# Esercitazione sul MonteCarlo

Matteo Duranti

matteo.duranti@pg.infn.it

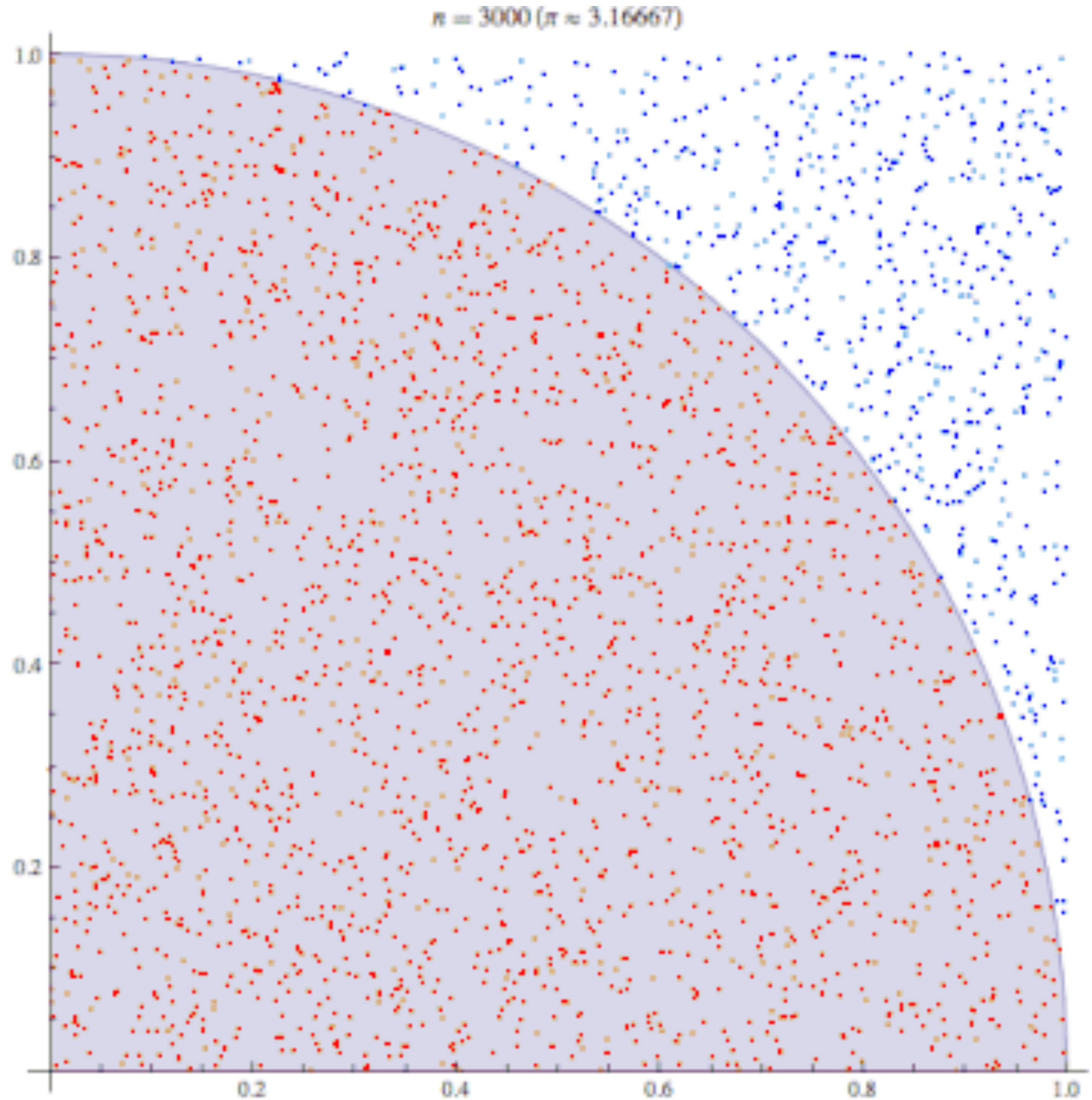
# Calcolo di $\pi$ tramite MonteCarlo

- l'area del quadrato è 1;

- l'area del cerchio è π/4;

→ l'integrale sotteso al cerchio può essere utilizzato per la stima di π

$$\left(\frac{\#\text{ pallini rossi}}{\#\text{ pallini totali}}\right) \approx \frac{\pi}{4}$$



# Calcolo di $\pi$ tramite MonteCarlo

L'algoritmo sarà quindi una cosa del tipo:

- si genera un coppia random *(x,y)*:

  $x$ in *[0,1]*

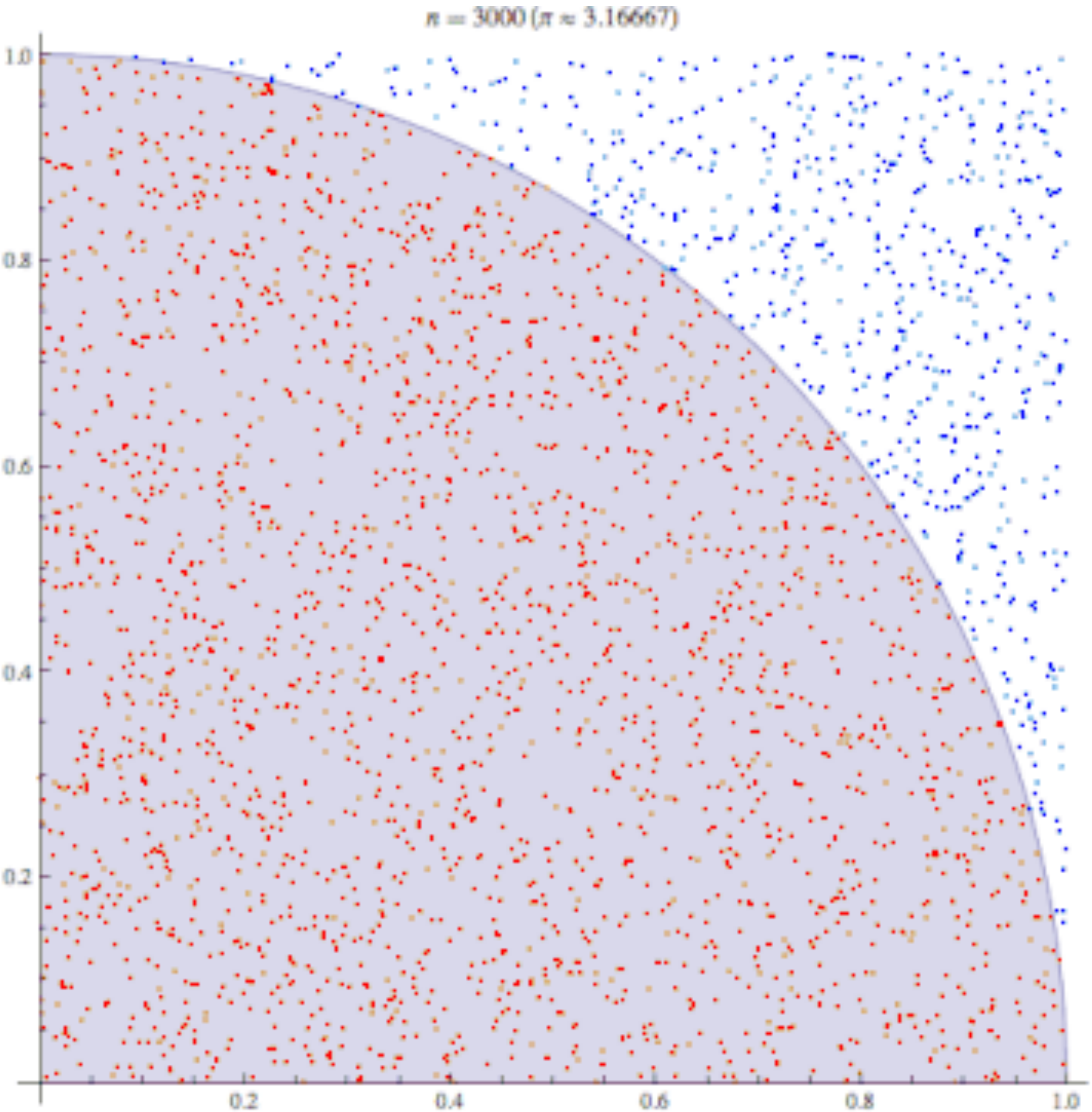
  $y$ in *[0,1]*

- si valuta:

  $x^2+y^2 < 1$

- si "conta" la frazione di *eventi* in cui la condizione era soddisfatta:

$$\left( \frac{\#\ \text{pallini rossi}}{\#\ \text{pallini totali}} \right) \approx \frac{\pi}{4}$$

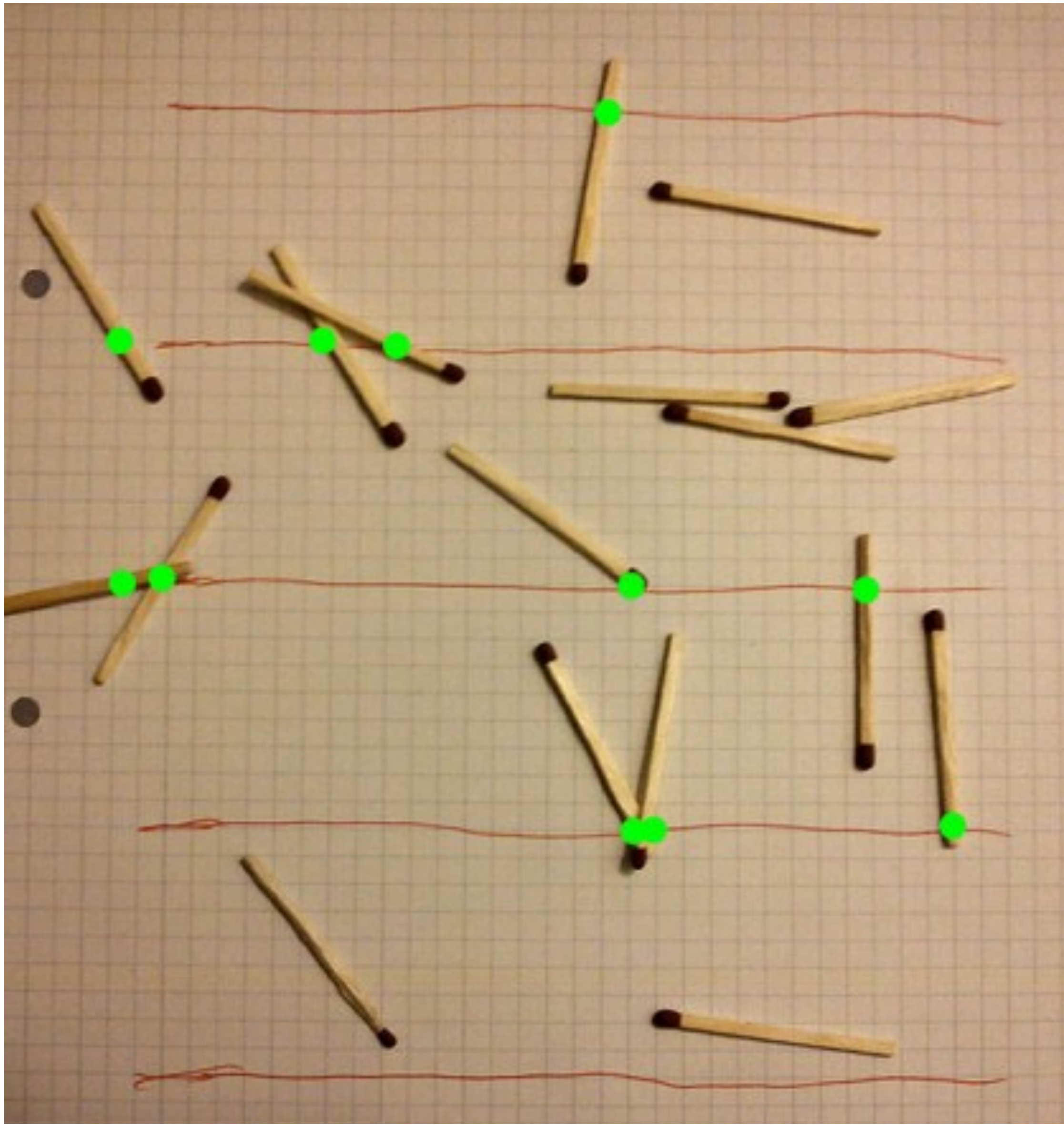


# L’ago di Buffon

Uno degli esempi “storici” di utilizzo di tecnica MonteCarlo è “l’ago di Buffon”:

- pattern di linee parallele a distanza *l;*
- bastoncino di lunghezza *d*;

→ quale è la probabilità che, tirando un bastoncino casualmente, questo intersechi la linea?



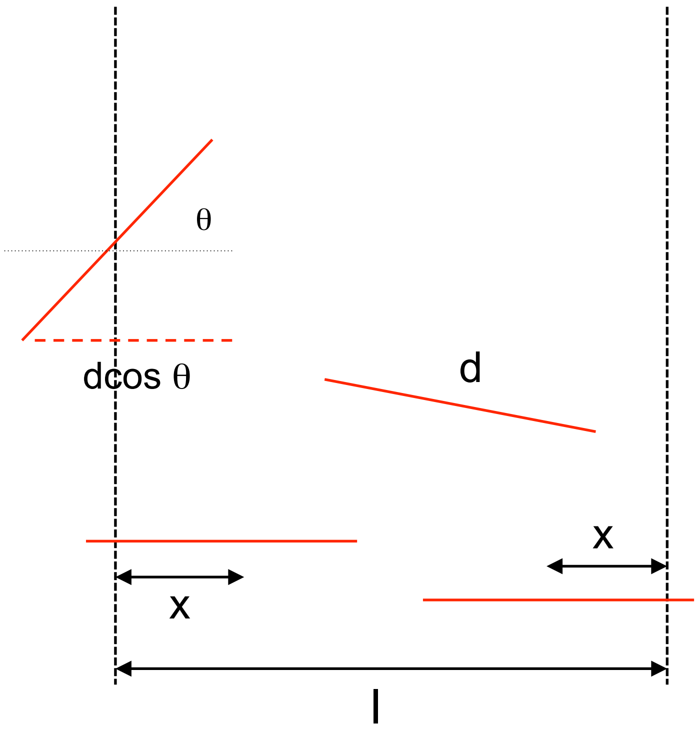
# L’ago di Buffon

L’algoritmo di calcolo del π, quindi, sarà qualcosa del tipo:

- si generano due numeri random per la posizione del centro e l’angolo;

$$|x| \in \left[0, \frac{l}{2}\right]$$

$$\theta \in \left[0, \frac{\pi}{2}\right]$$




- si valuta la condizione che deve essere soddisfatta per “l’intersecazione”:

$$|x| < d/2 \cos \theta$$

- si “conta” la frazione di *eventi* in cui c’è stata l’intersecazione:

$$\pi = \frac{2d}{l} \frac{n}{n_{int}}$$

# Calcolo di probabilità

Quale è la probabilità di ottenere 3, 6 o 9 volte testa lanciando 10 volte una moneta?

- il singolo “caso” (ad esempio: 3 volte “testa” in 10 lanci) è descrivibile da una binomiale:

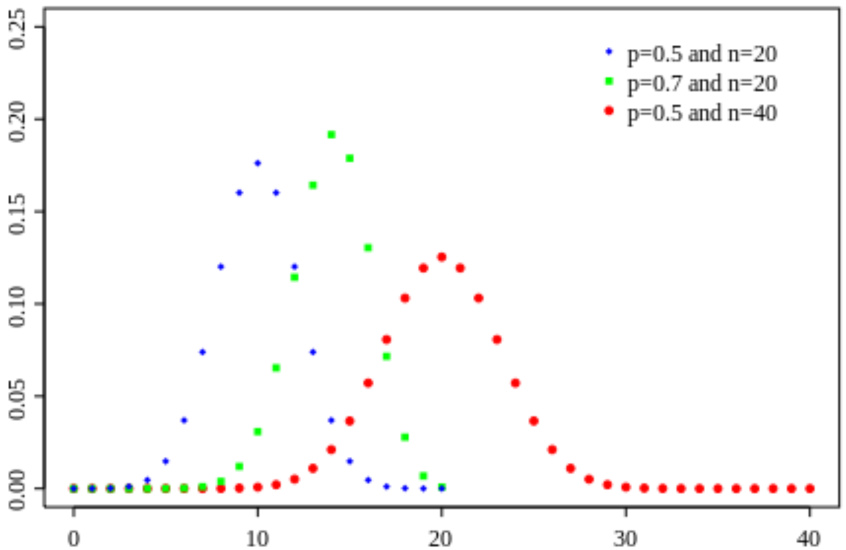
La distribuzione binomiale $\mathcal{B}(n,p)$ è caratterizzata da due parametri:[1]

- $n$: il numero di prove effettuate.
- $p$: la probabilità di successo della singola prova di Bernoulli $X_i$ (con $0 \le p \le 1$).

Per semplicità di notazione viene solitamente utilizzato anche il parametro $q = 1 - p$, che esprime la probabilità di fallimento per una singola prova.

La distribuzione di probabilità è:

$$P(k) = P(X_1 + X_2 + \cdots + X_n = k) = \binom{n}{k} p^k q^{n-k}$$




cioè ogni successione con $k$ successi e $n-k$ insuccessi ha probabilità $p^k q^{n-k}$, mentre il numero di queste successioni, pari al numero di modi (o combinazioni) in cui possono essere disposti i $k$ successi negli $n$ tentativi, è dato dal coefficiente binomiale $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

La formula del binomio di Newton mostra come la somma di tutte le probabilità nella distribuzione sia uguale a $1$:

$$\sum_{k=0}^{n} P(S_n = k) = \sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k} = (p+q)^n = (p+1-p)^n = (1)^n = 1$$

# Calcolo di probabilità

Quale è la probabilità di ottenere 3, 6 o 9 volte testa lanciando 10 volte una moneta?

- il singolo "caso" (ad esempio: 3 volte "testa" in 10 lanci) è descrivibile da una binomiale:

  ***B**(n,p)* = ***B**(10, ½)*

  e ha probabilità (*):

  $P(3) = 10!/(3!*7!)*0.5^3*0.5^7$

- il caso completo sarà:

  $P(3) + P(6) + P(9) \sim 0.33$

La distribuzione binomiale $\mathcal{B}(n,p)$ è caratterizzata da due parametri:[1]

- $n$: il numero di prove effettuate.
- $p$: la probabilità di successo della singola prova di Bernoulli $X_i$ (con $0 \leq p \leq 1$).

Per semplicità di notazione viene solitamente utilizzato anche il parametro $q = 1 - p$, che esprime la probabilità di fallimento per una singola prova.

La distribuzione di probabilità è:

$$P(k) = P(X_1 + X_2 + \cdots + X_n = k) = \binom{n}{k} p^k q^{n-k}$$

che ancora è risolvibile "a mano"…

(*) In ROOT *P(3)* la potete fare come:

```
[0] int k=3; TMath::Binomial(10, k)*pow(0.5, k)*pow(0.5, 10-k)
```

# Calcolo di probabilità

Quale è la probabilità di ottenere 3, 6 o 9 volte testa lanciando 10 volte una moneta?

Uno (pigro o che non ama la matematica) potrebbe invece scrivere un algoritmo che:

- genera ($M=10$) eventi uniformemente con x in $[0,1]$;
- definisce "testa" se $x<0.5$;
- conta il numero di volte in cui è uscito testa, $T$;
- ripete la procedura $N$ volte;
- stima la probabilità del singolo evento "i volte testa", come $\#(T=i)/N$;
- stima la probabilità del caso completo come somma delle tre singole (indipendenza da verificare!);

# Esercitazione

- Implementare l’algoritmo di valutazione del π con l’integrale del quarto di circonferenza;
- Implementare l’algoritmo di valutazione del π con il metodo dell’ago di Buffon;
- Valutare (*), in funzione del numero di “eventi”, la discrepanza fra la stima numerica del π (di entrambi i metodi), rispetto al valore esatto;

(*) come al solito: i grafici/istogrammi li potete fare con il metodo che volete: Excel, Calc, Numbers, Gnuplot, Scipy, Mathematica, Matlab, ROOT, etc…

# Esercitazione

- Verificare, con 1 milione di prove, che la stima effettuata tramite MonteCarlo sia conforme al conto “analitico”:

  $P(3) = 10!/(3!*7!)*0.5^3*0.5^7$

  $P(3) + P(6) + P(9) \sim 0.33$

- Verificare qualitativamente, con 1 milione di prove, che la distribuzione (*) del numero di volte in cui esce “testa” in 25 (non più 10) lanci sia quella prevista dalla binomiale. In ROOT potete plottare la binomiale così:

```
[0] TF1* f_binom = new TF1("binom", "TMath::Binomial([0],x)*pow(0.5,x)*pow(0.5,[0]-x)",0,26)
[2] f_binom->SetParameter(0,25)
[1] f_binom->Draw()
```

- Verificare qualitativamente, con 1 milione di ripetizioni, che la distribuzione del numero di volte, in *M*=1 milione di prove, in cui esce “testa” *N* volte (con *N* da 0 a 12) in 25 lanci, sia distribuita poissonianamente attorno *P(N)/M*. In ROOT potete plottare la poissoniana così:

```
[0] TF1* f_poiss = new TF1("poiss", "TMath::Poisson(x,[0])",0,15)
[1] f_poiss->SetParameter(0,3.4)
[2] f_poiss->Draw()
```

(*) come al solito: i grafici/istogrammi li potete fare con il metodo che volete: Excel, Calc, Numbers, Gnuplot, Scipy, Mathematica, Matlab, ROOT, etc…