

R per l'analisi statistica multivariata

Unità I: metodi Monte Carlo

Tommaso Rigon

Università Milano-Bicocca



Argomenti affrontati

- Metodi Monte Carlo
 - Approssimazione di un evento tramite Monte Carlo
 - Integrazione Monte Carlo
 - Istogrammi & densità
-
- Esercizi **R** associati: https://tommasorigon.github.io/introR/exe/es_3.html

I metodi Monte Carlo

- Nell'**unità H** abbiamo dedicato moltissime energie per cercare di capire come **simulare** dei valori (pseudo) casuali da variabili aleatorie continue e discrete.
- Ciò che tuttavia non abbiamo spiegato è l'**utilità** di queste tecniche.
- Il motivo è semplice: le **possibili applicazioni** sono talmente numerose che è necessario introdurle separatamente in questa lezione...
- ...e probabilmente scalfiremo solamente la superficie.

Metodo Monte Carlo

Definiamo **metodo Monte Carlo** una qualsiasi procedura che coinvolga l'utilizzo di **numeri (pseudo) casuali**.

Alcuni cenni storici

- I metodi Monte Carlo hanno una lunga storia; alcuni di essi sono stati usati perfino **prima** dell'invenzione dei **computer**.
- I primi utilizzi moderni, ovvero basati su numeri pseudo-casuali, sono stati condotti (tra gli altri) da Enrico Fermi, Nicholas Metropolis, Richard Feynman e John von Neumann tra gli anni '30 e '40.
- Il neonato metodo Monte Carlo aveva quindi delle importanti applicazioni in **fisica**. In particolare, importanti passi avanti furono fatti all'interno del **progetto Manhattan**.
- L'**algoritmo di Metropolis**, sviluppato in quegli anni, è tutt'oggi ampiamente usato. Purtroppo è prematuro presentarlo in questo corso: lo vederete più avanti!

Approfondimento

- Hitchcock (2003). A history of the Metropolis-Hastings algorithm. *The American Statistician* 57(4), 254–257.

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION

Number 247

SEPTEMBER 1949

Volume 44

THE MONTE CARLO METHOD

NICHOLAS METROPOLIS AND S. ULAM

Los Alamos Laboratory

We shall present here the motivation and a general description of a method dealing with a class of problems in mathematical physics. The method is, essentially, a statistical approach to the study of differential equations, or more generally, of integro-differential equations that occur in various branches of the natural sciences.

Possibili applicazioni

- I metodi Monte Carlo hanno applicazioni in tutte le discipline scientifiche, incluse la fisica, biologia, medicina, genetica, informatica, matematica.
- Per ovvie ragioni, noi approfondiremo le applicazioni legate alla **probabilità** e alla **statistica**. Alcuni esempi sono riportati nel seguito.
- Il metodo **bootstrap** tramite Monte Carlo è valso il "Nobel per la Statistica" a Brad Efron nel 2019. Link: https://en.wikipedia.org/wiki/International_Prize_in_Statistics.
- La **statistica bayesiana** moderna fa uso intensivo dei metodi Monte Carlo.
- Concetti chiave di **data mining & machine learning**, come la suddivisione in insieme di stima & verifica o la convalida incrociata, sono per definizione basati sulla simulazione di numeri casuali.
- Infine, grazie alla simulazione è possibile verificare la validità dei risultati "asintotici" che vengono presentati nei corsi di **inferenza statistica**.

Approssimazione di una probabilità

- Si supponga di voler calcolare una determinata probabilità π di un certo **esperimento casuale**. Definiamo una variabile aleatoria di bernoulli Z tale che

$$\pi = \mathbb{P}(Z = 1),$$

ovvero un **indicatore binario** che denota se l'evento si è verificato o meno.

- In molti casi è **difficile** se non praticamente impossibile calcolare π analiticamente.
- **Esempio**. Si supponga che $X \sim N(0, 1)$ e si ponga $Y = \cos(X)$. Il calcolo di

$$\pi = \mathbb{P}(Y > 0) = \mathbb{P}\{\cos(X) > 0\},$$

non è affatto semplice usando solo "carta e penna": provateci, se volete.

- **Esempio**. La probabilità di vittoria della tombola π si potrebbe calcolare "carta e penna", ma questa operazione sarebbe lunga e faticosa.

Approssimazione di una probabilità

Approccio Monte Carlo (approssimazione di una probabilità)

- Il metodo Monte Carlo prevede di **simulare** tante volte l'esperimento casuale in questione e contare la frazione di volte che l'evento si è verificato (ovvero $Z = 1$).
- In altri termini, consideriamo delle variabili aleatorie binarie iid Z_1, \dots, Z_R aventi probabilità π . La probabilità π viene stimata tramite la frazione di successi.
- Il metodo Monte Carlo è di estrema utilità perché permette di **approssimare** una determinata **probabilità** senza fare alcun conto analitico.
- **Nota.** Il punto cruciale è che spesso è possibile simulare un esperimento casuale senza conoscere π , che infatti è la probabilità che siamo interessati ad approssimare.

Esempio

- Si supponga nuovamente che $X \sim N(0, 1)$ e si ponga $Y = \cos(X)$. Siamo interessati a calcolare la probabilità:

$$\pi = \mathbb{P}(Y > 0) = \mathbb{P}\{\cos(X) > 0\}.$$

- Definiamo quindi la **variabile binaria**

$$Z = \mathbb{1}(Y > 0) = \mathbb{1}\{\cos(X) > 0\},$$

ovvero una variabile aleatoria bernoulliana che vale 1 se $\cos(X) > 0$ e vale 0 altrimenti.

- É facile verificare (fatelo per esercizio!) che

$$\pi = \mathbb{P}(Z = 1) = \mathbb{P}(Y > 0).$$

- **Risultato chiave.** Simulare delle copie iid dalla legge di Z è molto semplice, nonostante la probabilità π sia ignota.

Esempio (continua)

- Per **approssimare** la probabilità $\pi = \mathbb{P}(Z = 1) = \mathbb{P}\{\cos(X) > 0\}$ dobbiamo quindi generare tante copie iid da questa legge, ovvero Z_1, \dots, Z_R .

```
R <- 5000 # Numero di repliche

set.seed(123)
X <- rnorm(R, 0, 1) # Ottengo R copie da una distribuzione gaussiana
Y <- cos(X) # Ottengo R copie dalla distribuzione di Y
Z <- Y > 0 # Vettore logico che verifica se Y > 0 o meno
Z[1:10]
# [1] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE
```

- Il numero R rappresenta il numero di **repliche** e determina, come vedremo, la precisione della nostra stima Monte Carlo.
- A questo punto, l'approssimazione si ottiene considerando la proporzione di successi

```
prop.table(table(Z)) # Considero la frequenza relativa
# FALSE TRUE
# 0.1158 0.8842
mean(Z) # Oppure, più semplicemente
# [1] 0.8842
```

Errare è l'unica certezza

- Come tutte le approssimazioni, anche il metodo Monte Carlo comporta un **errore**.
- La peculiarità delle **approssimazioni Monte Carlo** è che sono, per definizione, casuali.
- Questo significa che ogni volta che eseguiamo la procedura otteniamo un valore **leggermente diverso**. Ad esempio:

```
set.seed(100) # Imposto un seed diverso da prima
Z <- cos(rnorm(R, 0, 1)) > 0 # Calcolo gli indicatori (codice in forma compatta)
mean(Z)
# [1] 0.8878
```

- Per cui sia la **stima** ottenuta che l'**errore** commesso sono **aleatori**!
- Fortunatamente, questo è un contesto che dovrete conoscere molto bene. La nostra procedura Monte Carlo è infatti, a tutti gli effetti, uno **stimatore** di π .

Come mai funziona?

- Siano Z_1, \dots, Z_R delle variabili aleatorie binarie indipendenti ed identicamente distribuite, aventi la stessa distribuzione della variabile $Z \sim \text{Ber}(\pi)$. Lo **stimatore**

$$\hat{\pi} = \frac{1}{R} \sum_{r=1}^R Z_r = (\text{"Proporzione di successi"}),$$

coincide con l'**approssimazione Monte Carlo**.

- Lo stimatore $\hat{\pi}$ ha delle ottime proprietà inferenziali, che si studiano in un qualsiasi corso di **inferenza statistica**; torneremo a parlare di stimatori nell'**unità M**.

- In primo luogo, lo stimatore $\hat{\pi}$ è **non distorto**, infatti:

$$\mathbb{E}(\hat{\pi}) = \frac{1}{R} \sum_{r=1}^R \mathbb{E}(Z_r) = \frac{1}{R} \sum_{r=1}^R \mathbb{P}(Z = 1) = \mathbb{P}(Z = 1) = \pi.$$

- Inoltre, lo stimatore $\hat{\pi}$ è **consistente**, infatti per la legge (forte) dei grandi numeri si ottiene che:

$$\hat{\pi} = \frac{1}{R} \sum_{r=1}^R Z_r \xrightarrow{\text{q.c.}} \mathbb{P}(Z = 1) = \pi, \quad R \rightarrow \infty.$$

La varianza dello stimatore I

- Possiamo infine calcolare la **varianza** di $\hat{\pi}$, che risulta pari alla seguente quantità

$$\text{var}(\hat{\pi}) = \frac{1}{R^2} \sum_{r=1}^R \text{var}(Z_r) = \frac{1}{R^2} \sum_{r=1}^R \pi(1 - \pi) = \frac{\pi(1 - \pi)}{R}.$$

- Da questa equazione è evidente il ruolo chiave del numero di repliche R .
- Il numero di **repliche** R si può interpretare come se fosse una sorta di **numerosità campionaria**, che idealmente noi possiamo aumentare a piacere.
- Un numero di repliche elevato aumenta quindi la **precisione** ma ha un costo in termini di risorse computazionali (= il computer impiega più tempo).
- **Nota.** La varianza $\text{var}(\hat{\pi})$ dipende dal valore di π , che è ignoto. Per cui una stima della varianza si ottiene rimpiazzando π con la sua stima $\hat{\pi}$.

La varianza dello stimatore II

- Vogliamo valutare l'impatto della scelta di R ed implementiamo quindi la funzione MonteCarlo, che calcola sia l'approssimazione che la sua **deviazione standard**.

```
MonteCarlo <- function(R){  
  Z <- cos(rnorm(R, 0, 1)) > 0  
  estimate <- mean(Z)  
  std.error <- sqrt(estimate * (1 - estimate) / R)  
  out <- c(estimate, std.error)  
  names(out) <- c("estimate", "std.error") # Aggiungo solo per ragioni estetiche  
  out  
}
```

- Proviamo con alcuni valori diversi di R . Si nota un progressivo miglioramento:

```
MonteCarlo(100) # R = 100 conduce a uno std.error elevato  
# estimate std.error  
# 0.92000000 0.02712932  
MonteCarlo(5000) # R = 5000 conduce a uno std.error ragionevole  
# estimate std.error  
# 0.88420000 0.00452527  
MonteCarlo(10^6) # R = 10^6 conduce a uno std.error basso  
# estimate std.error  
# 0.883676000 0.000320613
```

Esercizio riassuntivo I

- Sia X una normale standard. Si approssimi tramite Monte Carlo la probabilità seguente

$$\pi = \mathbb{P}(1 < X < 2).$$

- Si ottenga quindi una stima Monte Carlo dell'errore commesso.
- Si ripeta la procedura per diversi valori del numero di repliche R .
- Si confrontino i risultati con il **vero valore** di $\mathbb{P}(1 < X < 2)$. Le approssimazioni Monte Carlo migliorano al crescere di R ?
- **Esercizio difficile**. Si ottenga un intervallo di confidenza (**?!**) per lo stimatore $\hat{\pi}$ di livello approssimato $1 - \alpha = 0.95$.

Schema della soluzione

```
MonteCarlo <- function(R){  
  X <- rnorm(R)  
  Z <- (X > 1) & (X < 2)  
  estimate <- mean(Z)  
  std.error <- sqrt(estimate * (1 - estimate) / R)  
  out <- c(estimate, std.error)  
  names(out) <- c("estimate", "std.error")  
  out  
}  
  
# Vero valore  
pnorm(2) - pnorm(1)  
# [1] 0.1359051  
  
MonteCarlo(100) # R = 100 conduce a std.error elevato  
# estimate std.error  
# 0.1400000 0.0346987  
MonteCarlo(5000) # R = 5000 conduce a std.error ragionevole  
# estimate std.error  
# 0.133600000 0.004811466  
MonteCarlo(10^6) # R = 10^6 conduce a std.error basso  
# estimate std.error  
# 0.1360790000 0.0003428724
```


Integrazione Monte Carlo

- L'idea di approssimare una probabilità tramite simulazione può essere **generalizzata**.
- In particolare, supponiamo di voler calcolare un generico **integrale** del tipo

$$\mathcal{I} = \int_{\mathcal{X}} g(x)f(x)dx = \mathbb{E}\{g(X)\},$$

dove $f(x)$ è la densità una variabile aleatoria X avente supporto \mathcal{X} .

- La **probabilità di un evento** è un **caso particolare** di questo contesto. Infatti se $g(x) = \mathbb{1}(x \in B)$ si ottiene

$$\mathcal{I} = \int_{\mathcal{X}} g(x)f(x)dx = \int_B f(x)dx = \mathbb{P}(X \in B).$$

- Esattamente come per la probabilità di un evento, vogliamo usare la simulazione per ottenere un'**approssimazione** di \mathcal{I} .

Integrazione Monte Carlo

Approccio Monte Carlo (Integrazione)

- Sia $X \sim f(x)$. Per approssimare l'integrale

$$\mathcal{I} = \int_{\mathcal{X}} g(x)f(x)dx = \mathbb{E}\{g(X)\}$$

si **simulano** dei valori X_1, \dots, X_R da $f(x)$. Si calcolano quindi i valori $g(X_1), \dots, g(X_R)$ ed infine si considera la loro **media campionaria**.

- La stima Monte Carlo $\hat{\mathcal{I}}$ è tale che

$$\hat{\mathcal{I}} = \frac{1}{R} \sum_{r=1}^R g(X_r) \approx \mathbb{E}\{g(X)\} = \mathcal{I}.$$

- **Nota.** Il metodo descritto può essere in realtà usato per approssimare un qualsiasi valore atteso $\mathbb{E}\{g(X)\}$, anche quando la variabile aleatoria X è **discreta**.

Esempio

- **Esempio.** Supponiamo di voler calcolare il valore del seguente integrale

$$\mathcal{I} = \int_0^1 [\cos(50x) + \sin(20x)]^2 dx.$$

- Si noti che questo integrale coincide con il valore atteso di una trasformazione di una variabile aleatoria uniforme U , ovvero

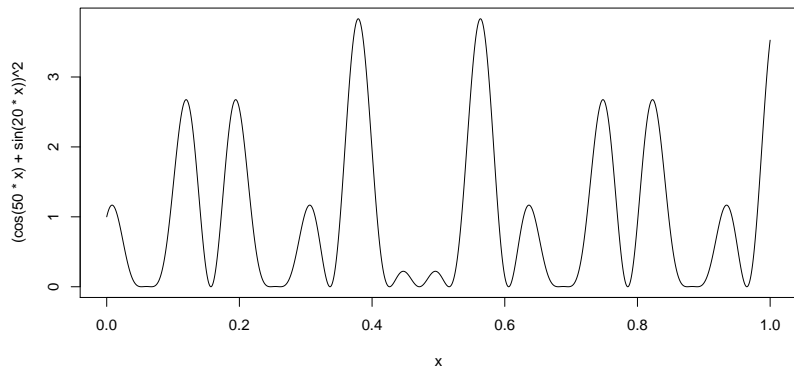
$$\mathcal{I} = \mathbb{E}[\{\cos(50U) + \sin(20U)\}^2], \quad U \sim \text{Unif}(0, 1).$$

- In **R** pertanto possiamo calcolare $\hat{\mathcal{I}}$ come segue

```
U <- runif(10^6)
I_hat <- mean((cos(50 * U) + sin(20 * U))^2)
I_hat
# [1] 0.9650047
```

- Questa funzione in realtà può essere integrata analiticamente: vale che $\mathcal{I} \approx 0.965201$. Pertanto, l'approssimazione Monte Carlo sembra essere accurata.

Esempio (continua)



- Grafico della funzione integranda nell'intervallo $(0, 1)$.

Come mai funziona? Procediamo come prima...

- Siano X_1, \dots, X_R delle copie iid aventi densità $f(x)$. Consideriamo quindi lo **stimatore** seguente

$$\hat{\mathcal{I}} = \frac{1}{R} \sum_{r=1}^R g(X_r),$$

ovvero l'**approssimazione Monte Carlo** di \mathcal{I} che abbiamo descritto.

- Anche in questo caso, otteniamo che $\hat{\mathcal{I}}$ è uno stimatore di \mathcal{I} con ottime proprietà inferenziali.
- Come in precedenza, lo stimatore $\hat{\mathcal{I}}$ risulta essere **non distorto**, infatti:

$$\mathbb{E}(\hat{\mathcal{I}}) = \frac{1}{R} \sum_{r=1}^R \mathbb{E}\{g(X_r)\} = \frac{1}{R} \sum_{r=1}^R \mathbb{E}\{g(X)\} = \mathbb{E}\{g(X)\} = \mathcal{I}.$$

- Inoltre, lo stimatore $\hat{\mathcal{I}}$ è **consistente**. Infatti per la legge (forte) dei grandi numeri

$$\hat{\mathcal{I}} = \frac{1}{R} \sum_{r=1}^R \mathbb{E}\{g(X_r)\} \xrightarrow{\text{q.c.}} \mathbb{E}\{g(X)\} = \mathcal{I}, \quad R \rightarrow \infty.$$

La varianza dello stimatore I

- Anche in questo caso possiamo calcolare la **varianza** dello stimatore $\hat{\mathcal{I}}$:

$$\text{var}(\hat{\mathcal{I}}) = \frac{1}{R^2} \sum_{r=1}^R \text{var}\{g(X_r)\} = \frac{1}{R} \text{var}\{g(X)\},$$

con $X \sim f(x)$.

- La varianza $\text{var}\{g(X)\}$ è tipicamente ignota, ma può essere stimata utilizzando gli stessi valori usati per stimare \mathcal{I} , ad esempio tramite la **varianza campionaria**:

$$\widehat{\text{var}\{g(X)\}} = \frac{1}{R} \sum_{r=1}^R g(X_r)^2 - \left(\frac{1}{R} \sum_{r=1}^R g(X_r) \right)^2.$$

- Come in precedenza, un numero di repliche R elevato aumenta quindi la **precisione** ma ha un **costo computazionale**.

La varianza dello stimatore II

- L'implementazione in **R** si ottiene come segue:

```
MonteCarlo <- function(R){  
  U <- runif(R)  
  hU <- (cos(50 * U) + sin(20 * U))^2  
  estimate <- mean(hU)  
  std.error <- sd(hU) / sqrt(R)  
  out <- c(estimate, std.error)  
  names(out) <- c("estimate", "std.error") # Aggiungo solo per ragioni estetiche  
  out  
}
```

- Proviamo con alcuni valori diversi di *R*. Si nota un progressivo miglioramento:

```
MonteCarlo(100) # R = 100 conduce a uno std.error elevato  
# estimate std.error  
# 0.9420754 0.1062872  
MonteCarlo(5000) # R = 5000 conduce a uno std.error ragionevole  
# estimate std.error  
# 0.9808332 0.0150004  
MonteCarlo(10^6) # R = 10^6 conduce a uno std.error basso  
# estimate std.error  
# 0.964881973 0.001044959
```

Esercizio riassuntivo I

- Si calcoli tramite Monte Carlo il valore del seguente integrale e se ne quantifichi l'incertezza

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \sin^2(x) e^{-x^2/2} dx.$$

- Si confronti il risultato con la funzione `integrate`.

- Schema della soluzione.

```
X <- rnorm(10^5)
hX <- sin(X)^2
mean(hX) # Estimate
sd(hX) / sqrt(10^5) # Std.error

# Integrazione numerica
integrate(function(x) sin(x)^2 * dnorm(x), -Inf, Inf)
```

Esercizio riassuntivo II

- Si supponga di voler approssimare tramite Monte Carlo il seguente valore atteso

$$\mathbb{E}(X^2), \quad X \sim \text{Ga}(3, 3).$$

- Si dica, motivando la risposta, quale codice produce il risultato corretto.

- Codice 1

```
mean(rgamma(10^5, 3, 3) * rgamma(10^5, 3, 3))
```

- Codice 2

```
X <- rgamma(10^5, 3, 3)  
mean(X * X)
```

Esercizi aggiuntivi (non risolti)

- **Esercizio.** Si ottenga un'approssimazione Monte Carlo del seguente integrale

$$\int_0^{\infty} x^4 e^{-x} dx$$

e si quantifichi l'errore commesso. Si confronti il risultato con la funzione integrale.

- **Esercizio.** Si ottenga un'approssimazione Monte Carlo del seguente integrale

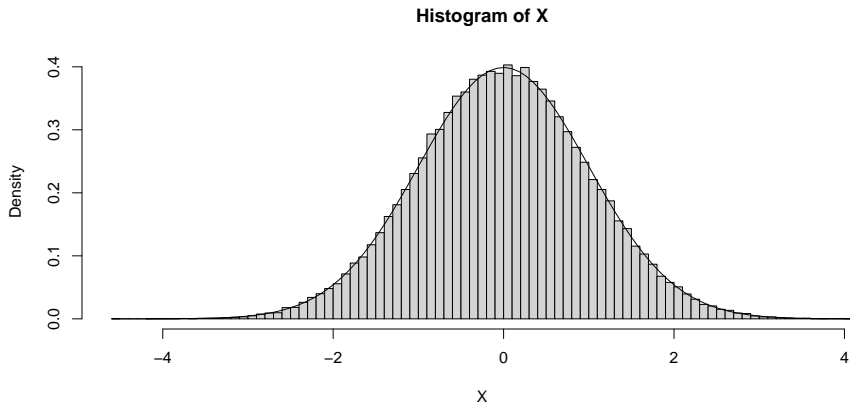
$$\int_0^1 x^{1/2} (1-x)^{1/2} dx$$

e si quantifichi l'errore commesso. Si confronti il risultato con la funzione integrale.

Istogrammi e densità

- Supponiamo di simulare delle variabili aleatorie continue X_1, \dots, X_R , aventi una certa **densità** $f(x)$.
- Se disegniamo l'**istogramma** di tali numeri, intuitivamente ci aspetteremo un'alta densità nell'istogramma in corrispondenza dei valori molto probabili.
- In realtà, il legame tra istogrammi e densità è molto più stretto.
- Consideriamo $\lambda = 1/R$ (si veda **unità D** per la definizione), ovvero il valore che rende la somma delle aree dei rettangoli pari a 1.
- In tale contesto, l'istogramma costituisce un'approssimazione della densità.
- In termini più precisi, diremo che l'**istogramma** è uno **stimatore** nonparametrico (!?) della **densità** $f(x)$.

Istogrammi e densità



```
X <- rnorm(10^5)
hist(X, freq = FALSE, breaks = 100)
curve(dnorm(x), add = TRUE) # add = TRUE Aggiunge la curva al grafico precedente
```

Istogrammi e densità: qualche intuizione

- L'idea è approssimare la funzione $f(x)$ con dei **rettangoli**. Quanti più rettangoli consideriamo, tanto più accurata sarà l'approssimazione.

- Ricordiamo che se $X \sim f(x)$ allora vale che

$$\mathbb{P}(a < X < b) = \int_a^b f(x)dx.$$

- Quindi idealmente l'altezza del rettangolo di base (a, b) dev'essere tale che

$$(\text{"Altezza rettangolo"}) = \frac{\mathbb{P}(a < X < b)}{b - a},$$

in maniera tale che l'**area del rettangolo** risulti pari a $\mathbb{P}(a < X < b)$.

- Le probabilità $\mathbb{P}(a < X < b)$ sono ulteriormente **approssimate** tramite Monte Carlo e sono poste pari alle proporzioni di valori X_1, \dots, X_R contenuti nell'intervallo (a, b) .

Approssimazione di una distribuzione discreta

- Un principio simile a visto per istogrammi / densità vale anche nel **caso discreto**.
- Sia X una variabile aleatoria discreta. In questo contesto, possiamo direttamente approssimare la **funzione di probabilità**

$$p(x) = \mathbb{P}(X = x),$$

utilizzando un metodo Monte Carlo.

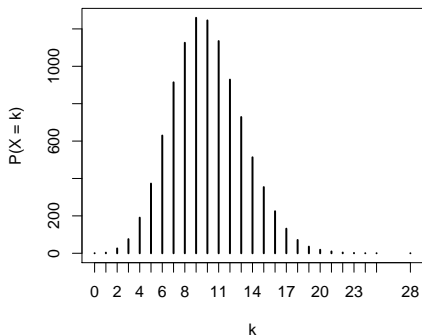
- Supponendo di poter simulare X_1, \dots, X_R . Allora, una stima per $p(x)$ è semplicemente pari alla proporzione di valori pari x che abbiamo ottenuto.

```
X <- rpois(10^5, 10) # Simulazione di R variabili Poisson con media 10
freq_rel <- table(X) / n # Calcolo delle frequenze relative

par(mfrow = c(1, 2)) # Grafici
plot(freq_rel, ylab = "P(X = k)", xlab = "k", main = "Distribuzione empirica")
plot(0:28, dpois(0:28, 10), type = "h", ylab = "P(X = k)",
     xlab = "k", main = "Distribuzione teorica")
```

Approssimazione di una distribuzione discreta

Distribuzione empirica



Distribuzione teorica

