RecapCap2

2023-12-31

Apprendimento statistico - MASTRANTONIO

TIME SERIES pt. 1

```
currmar <- par()$mar
library(astsa)
library(forecast)
```

```
## Warning: il pacchetto 'forecast' è stato creato con R versione 4.2.3
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
##
## Caricamento pacchetto: 'forecast'
```

```
## Il seguente oggetto è mascherato da 'package:astsa':
##
##     gas
```

```
library(datasets)
library(lmtest)
```

```
## Warning: il pacchetto 'lmtest' è stato creato con R versione 4.2.3
```

```
## Caricamento del pacchetto richiesto: zoo
```

```
## Warning: il pacchetto 'zoo' è stato creato con R versione 4.2.3
```

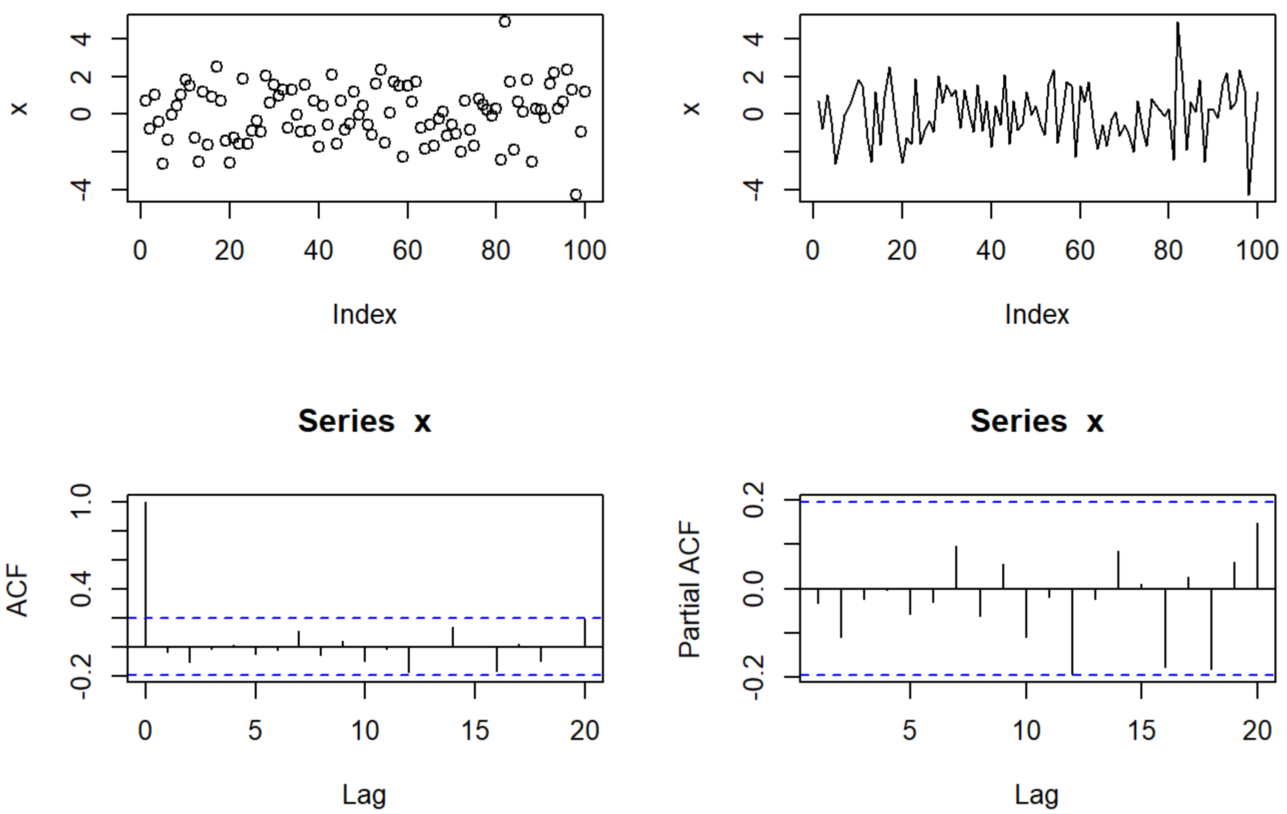
```
##
## Caricamento pacchetto: 'zoo'
```

```
## I seguenti oggetti sono mascherati da 'package:base':
##
##     as.Date, as.Date.numeric
```

SIMULAZIONE DI UN RUMORE BIANCO

Il codice genera un vettore di 100 numeri casuali distribuiti secondo una distribuzione normale$N(0, \sqrt{2}$.

```
# Genera un vettore x di lunghezza 100 contenente numeri casuali distribuiti secondo una distribuzione N(0, sqrt(2))
x = rnorm(100,0,2^0.5)

par(mfrow=c(2,2))
plot(x) # Crea un grafico a dispersione (scatter plot) dei valori nel vettore x
plot(x, type="l") # Crea un grafico a linea dei valori nel vettore x
acf(x) # Calcola e visualizza la funzione di autocorrelazione (ACF) dei valori nel vettore x
pacf(x) #  Calcola e visualizza la funzione di autocorrelazione parziale (PACF) dei valori nel vettore x
```

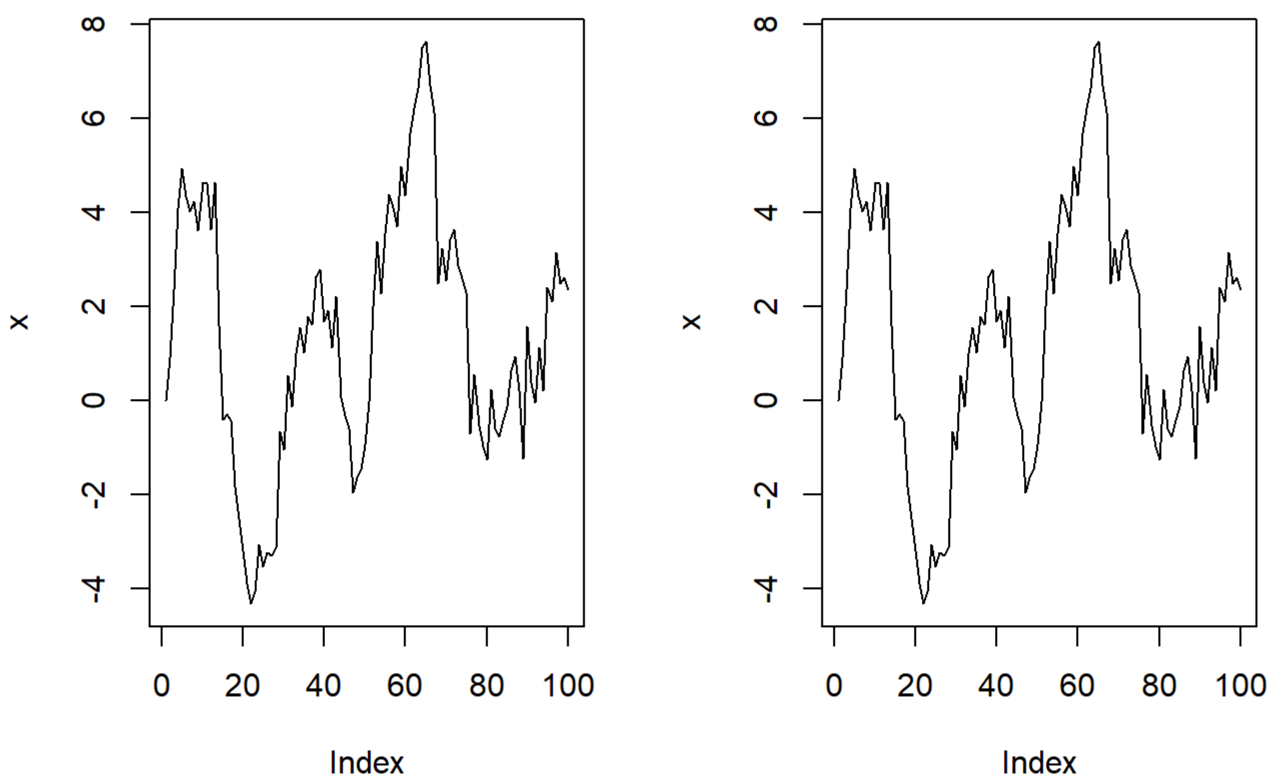


In sintesi, il codice crea quattro grafici in una griglia 2x2. Il primo grafico è uno scatter plot dei dati, il secondo è un grafico a linea, il terzo è la funzione di autocorrelazione (ACF), e il quarto è la funzione di autocorrelazione parziale (PACF).

SIMULAZIONE DI UN RANDOM WALK con $W_t \sim N(0,1)$

Il codice genera un processo stocastico noto come processo stocastico cumulativo o somma cumulativa di un processo di Wiener (anche noto come processo di Brownian motion). In altre parole, il vettore x rappresenta la somma cumulativa di un vettore di variabili casuali distribuite normalmente.

```
n = 100 # Definisce la lunghezza del vettore w
w = rnorm(n, 0, 1) # Genera un vettore w di lunghezza n contenente numeri casuali distribuiti secondo una distribuzione N(0,1)

x = c(0)
# Calcola la somma cumulativa dal secondo elemento
for(i in 2:n){
  # In questo modo, si ottiene una sequenza cumulativa dei valori nel vettore w
  x[i] = x[i-1]+ w[i]
}

par(mfrow=c(1,2))
plot(x, type="l") # Crea un grafico a linea della sequenza cumulativa x.

# In alternativa al for loop
# Aggiunge un zero iniziale e poi calcola la somma cumulativa degli elementi nel vettore w, tranne il primo
x = c(0,cumsum(w[-1]))

plot(x, type="l") # Crea un grafico a linea della sequenza cumulativa x.
```

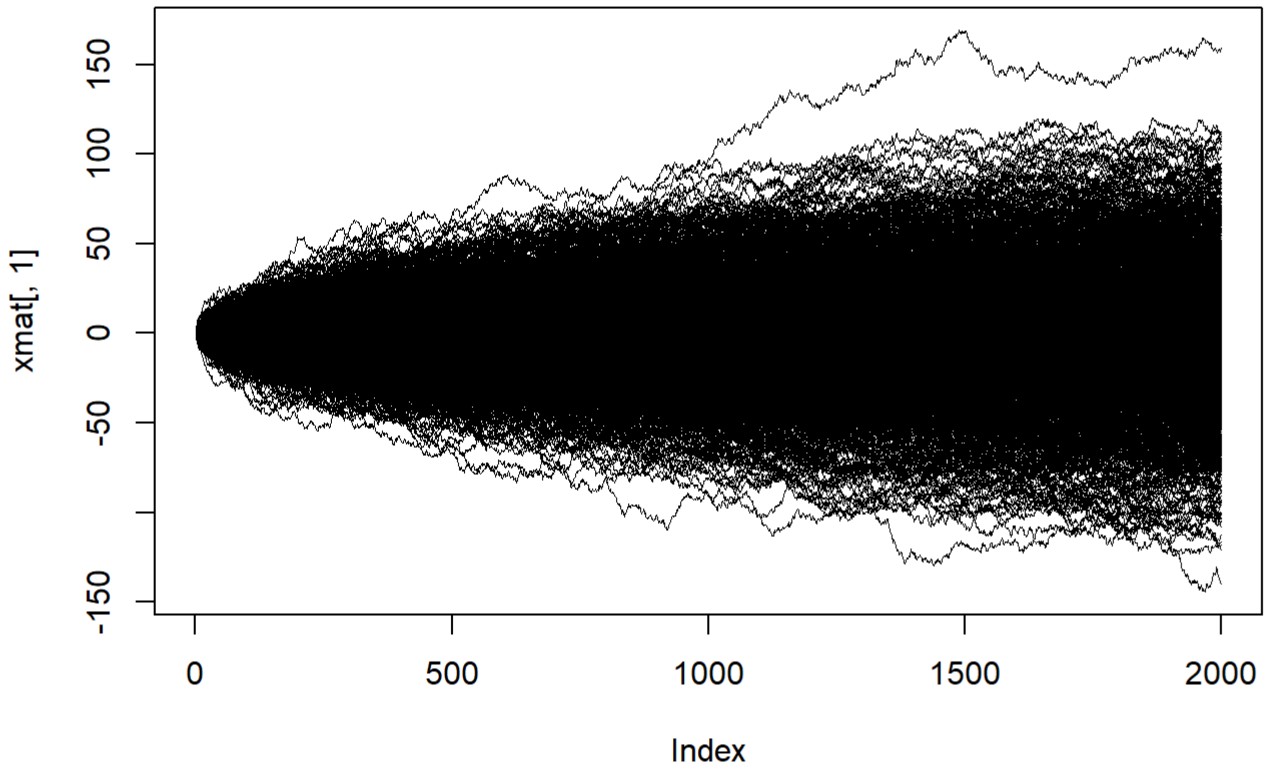


Entrambe le varianti del codice generano un grafico della somma cumulativa di un processo di Wiener o di un processo di Brownian motion.

STUDIO DELLA VARIABILITA’

Il codice genera un grafico che mostra le traiettorie di più simulazioni di un processo di Wiener (o processo di Brownian motion). Il processo di Wiener è un processo stocastico continuo caratterizzato da incrementi indipendenti e distribuiti normalmente.

```
n = 2000 #  Definisce la lunghezza di ciascuna traiettoria del processo
nsim = 1000 #  Definisce il numero di simulazioni da eseguire

xmat = matrix(0,ncol=nsim, nrow=n)

# Il doppio loop for esegue le simulazioni. Per ogni simulazione (isim), viene generato un vettore w di lunghezza n contenente numeri casuali distribuiti secondo una normale standard (rnorm(n, 0, 1)). Successivamente, la somma cumulativa viene calcolata e memorizzata nella matrice xmat.
for(isim in 1:nsim){
  w = rnorm(n, 0,1)
  for(i in 2:n){
    xmat[i,isim] = xmat[i-1, isim]+ w[i]
  }
}

# Crea il primo grafico di una delle traiettorie (simulazioni). La scala sull'asse y (ylim) è impostata in modo dinamico per includere tutte le traiettorie
plot(xmat[,1], type="l", ylim=range(c(xmat)), lwd=0.2)

# Il for aggiunge tutte le altre traiettorie al grafico con la funzione lines()
for(isim in 2:nsim){
  lines(xmat[,isim], type="l", lwd=0.2)
}
```



In conclusione, il grafico finale mostra visivamente come le traiettorie del processo di Wiener si sviluppano nel tempo. Le traiettorie sono stocastiche e si muovono in modo casuale, riflettendo la natura del processo di Wiener.

Chiaramente la varianza aumento con t. Se io invece facessi il seguente modello

$$x_t = \alpha x_{t-1} + w_t$$

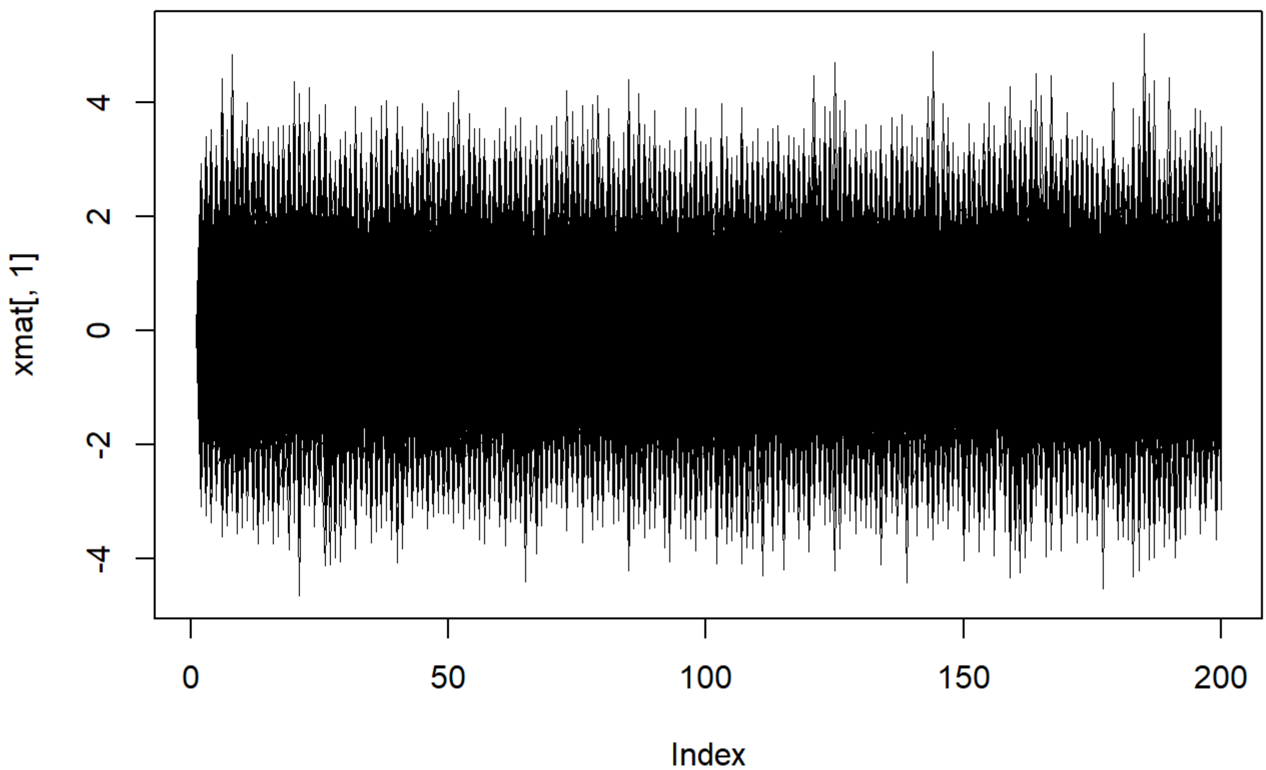
con $|\alpha| < 0$

Il codice simula un processo autoregressivo di ordine 1 (AR(1)) con un parametro di autoregressione (α) pari a -0.4. In altre parole, il processo segue la seguente equazione ricorsiva:

$$X_t = \alpha \cdot X_{t-1} + \varepsilon_t$$

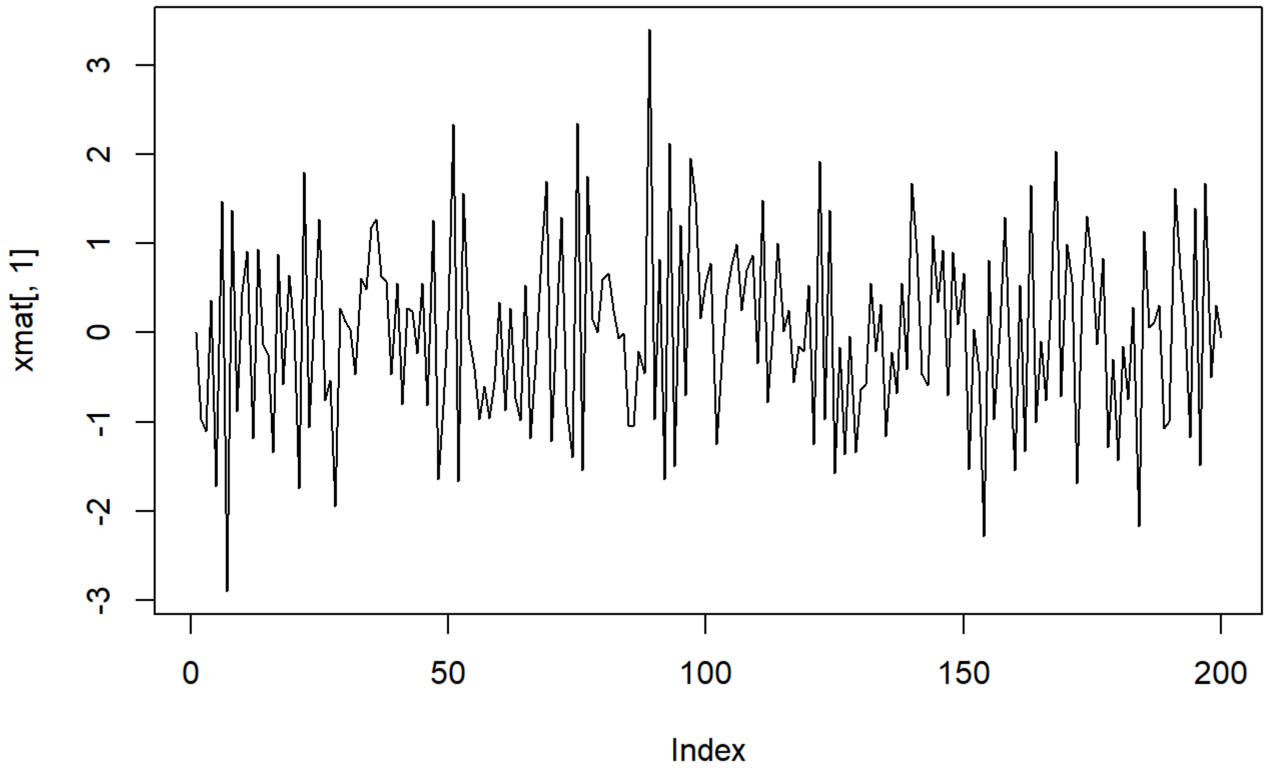
dove X_t è il valore al tempo t, α è il coefficiente di autoregressione, X_{t-1} è il valore al tempo precedente, e ε_t è un termine di errore distribuito normalmente con media zero e deviazione standard 1 ($\varepsilon_t \sim N(0, 1)$).

```
n = 200 # Definisce la lunghezza di ciascuna traiettoria del processo
nsim = 1000 # Definisce il numero di simulazioni da eseguire
alpha = -0.4 # Specifica il coefficiente di autoregressione
xmat = matrix(0,ncol=nsim, nrow=n)

# Il doppio loop for esegue le simulazioni. Per ogni simulazione (isim), viene generato un ve
ttore w di lunghezza n contenente numeri casuali distribuiti secondo una normale standard (rn
orm(n, 0, 1)). Successivamente, il processo AR(1) viene simulato utilizzando l'equazione rico
rsiva.
for(isim in 1:nsim){
  w = rnorm(n, 0,1)
  for(i in 2:n){
    xmat[i,isim] = alpha*xmat[i-1, isim]+ w[i]
  }
}

# Crea il primo grafico di una delle traiettorie. La scala sull'asse y (ylim) è impostata in
modo dinamico per includere tutte le traiettorie
plot(xmat[,1], type="l", ylim=range(c(xmat)), lwd=0.2)

# Aggiunge tutte le altre traiettorie al grafico con la funzione lines()
for(isim in 2:nsim){
  lines(xmat[,isim], type="l", lwd=0.2)
}
```



Il grafico finale mostra come le traiettorie del processo AR(1) si sviluppano nel tempo con il coefficiente di autoregressione $\alpha = -0.4$.

```
plot(xmat[,1], type="l") # genera un grafico a linee della prima traiettoria (simulazione) da
lla matrice xmat
```



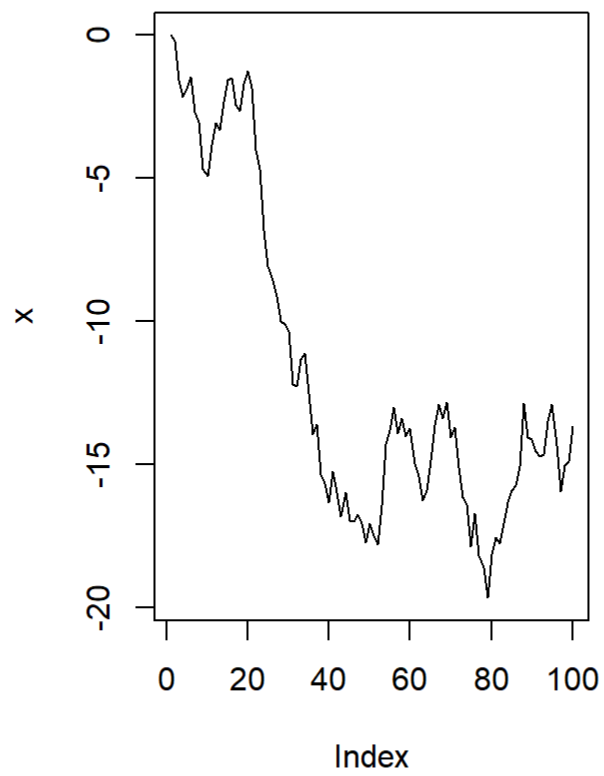
Il plot mostra come variano nel tempo i valori della prima simulazione del processo AR(1) con coefficiente di autoregressione $\alpha = -0.4$.

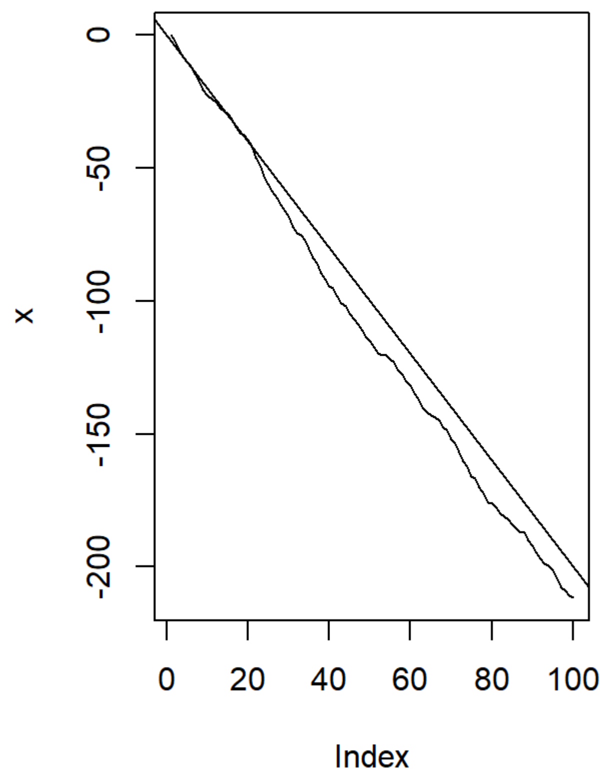
Nel contesto del processo AR(1) specificato nel codice, questo grafico a linee rappresenta come la prima traiettoria si sviluppa nel tempo in risposta a una sequenza di errori casuali. La connessione tra i punti rappresenta l'evoluzione del processo nel tempo.

Aggiungiamo un delta alla serie random walk.

Il codice simula due serie temporali utilizzando processi stocastici, e poi li visualizza in una griglia di due grafici.

```r
# Imposta il seed del generatore di numeri casuali per garantire la riproducibilità delle simulazioni. Il valore 10 è arbitrario e può essere sostituito con qualsiasi altro numero
set.seed(10)

# Primo processo
n = 100 # Definisce la lunghezza della serie temporale
w = rnorm(n, 0,1) # Genera un vettore di lunghezza n contenente numeri casuali distribuiti secondo una normale standard
x = c(0)

# Somma cumulativamente i valori casuali per generare una serie temporale x
for(i in 2:n){
  x[i] = x[i-1]+ w[i]
}

par(mfrow=c(1,2))
plot(x, type="l") # Crea un grafico a linee della prima serie temporale

# Secondo processo
delta = -2
x = c(0)

# simula un secondo processo in cui ad ogni passo viene sommato il parametro delta
for(i in 2:n){
  x[i] = delta+x[i-1]+ w[i]
}
# La serie temporale risultante viene visualizzata insieme a una retta orizzontale con intercetta delta (abline(a=0, b=delta, col=1))
plot(x, type="l")
abline(a = 0, b = delta, col=1)
```





In sostanza, il primo grafico mostra una serie temporale con incrementi casuali, mentre il secondo grafico mostra una serie temporale con incrementi casuali e un termine costante delta aggiunto ad ogni passo.

EFFETTI STAGIONALI

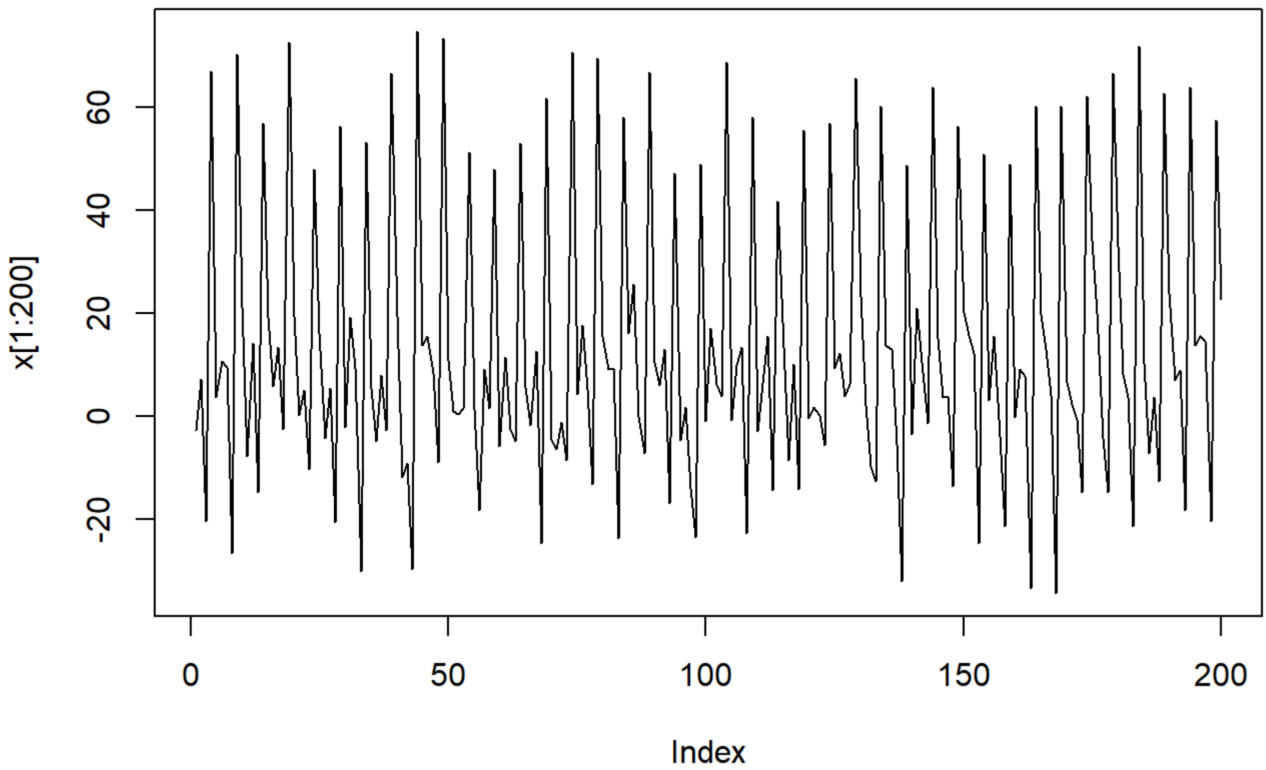
Assumiamo

$$x_t = a_{\mathrm{t\ mod\ s}} + w_t$$

simuliamo la serie storica.

Il codice genera una serie temporale x di lunghezza n utilizzando un modello di regressione ciclica. Il modello include un vettore di coefficienti a e un termine di errore distribuito normalmente.

```r
s = 5 # Definisce la lunghezza del vettore dei coefficienti a
n = 1000 # Definisce la lunghezza della serie temporale x
a = c(10,5,3,-10,60) # Specifica un vettore di coefficienti per il modello di regressione cic
lica
sigma2 = 100 # Definisce la varianza del termine di errore distribuito normalmente
x = c() # Inizializza un vettore vuoto x che conterrà la serie temporale simulata

# Il loop for simula la serie temporale. Ad ogni passo, viene selezionato un coefficiente cic
licamente da a usando l'operatore modulo (%%). A questo coefficiente viene aggiunto un termin
e di errore generato da rnorm(1, 0, sigma2^0.5), che è una variabile casuale distribuita norm
almente con media zero e deviazione standard sigma2^0.5
for(i in 1:n){
  x[i] = a[i%%(s)+1]+ rnorm(1,0,sigma2^0.5)
}

# Crea un grafico a linee dei primi 200 valori della serie temporale x
plot(x[1:200], type="l")
```

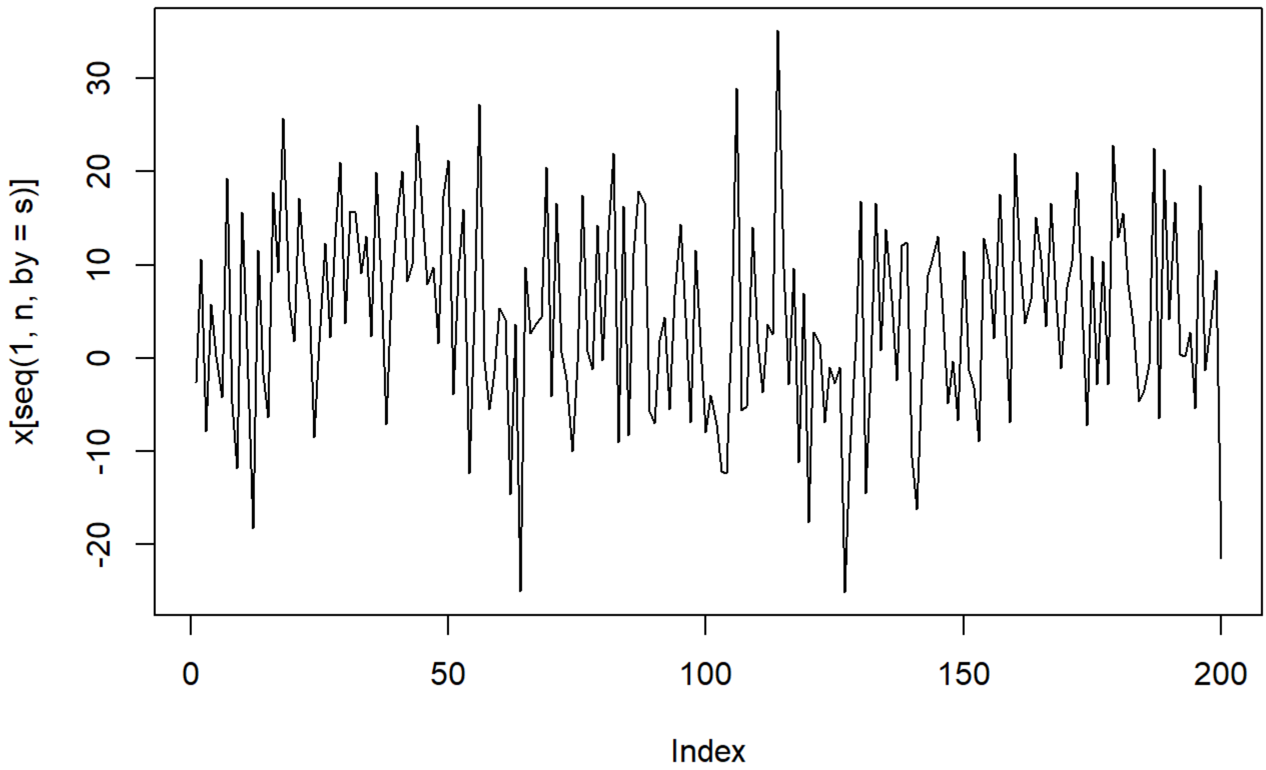


In sostanza, il grafico mostra la dinamica della serie temporale generata dalla combinazione di coefficienti ciclici e un termine di errore. Ogni coefficiente del vettore a viene utilizzato ciclicamente per influenzare i valori della serie temporale.

Prendo le sottoserie stazionarie

$$(x_1, x_6, x_{11}, x_{16}, \ldots)$$

```
#  seq seleziona i punti della serie temporale in cui viene cambiato il coefficiente ciclicam
ente
plot(x[seq(1, n, by = s)], type="l") # Crea un grafico a linee utilizzando solo i valori dell
a serie temporale x nei punti selezionati dalla sequenza
```



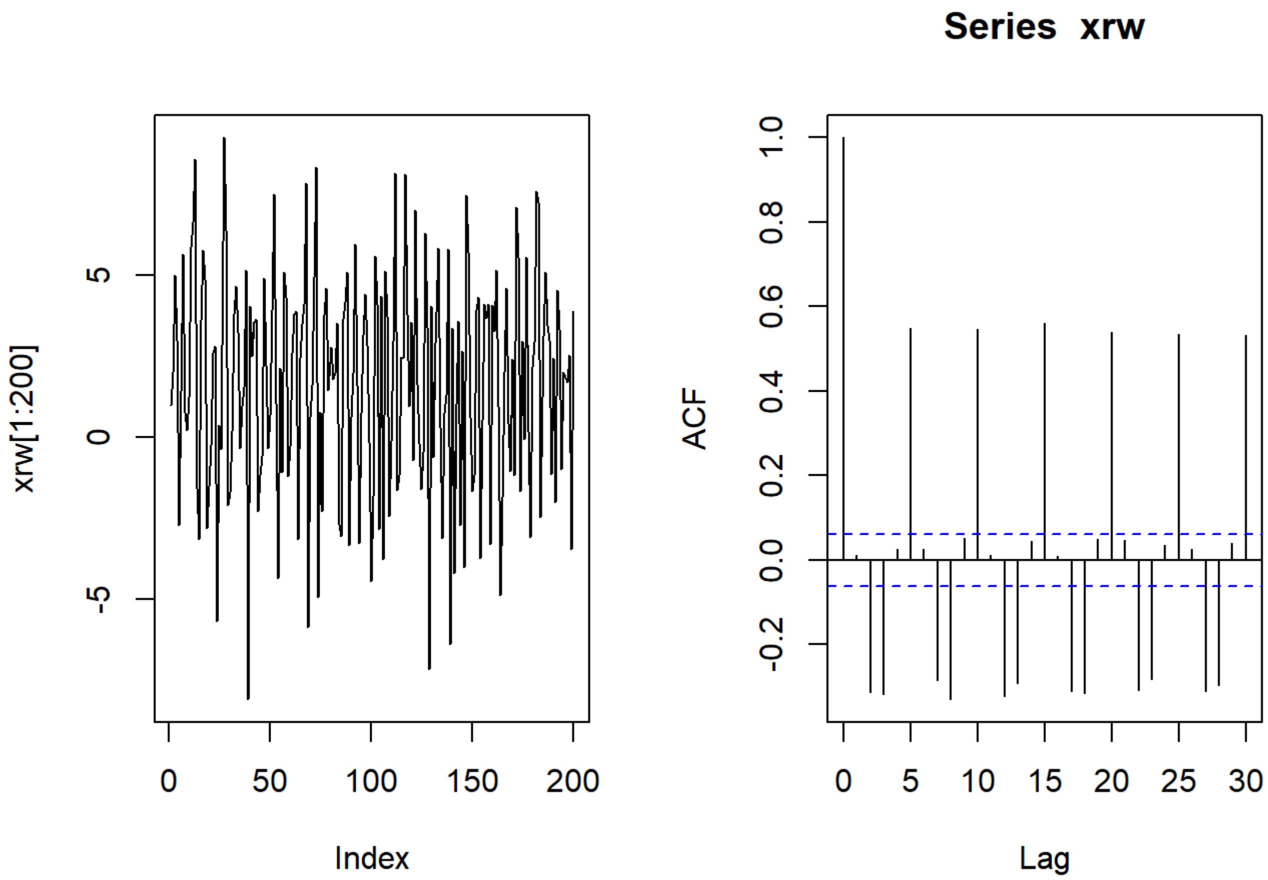
Questo grafico rappresenta la dinamica della serie temporale considerando solo i punti in cui cambiano i coefficienti ciclicamente. Ogni segmento di linea corrisponde a una selezione diversa di coefficienti dal vettore a.

Simulo un RW stagionale

$$x_t = x_{t-s} + w_t$$

Il codice simula due serie temporali: una serie temporale con coefficienti ciclici (xrw) e una serie temporale senza tali coefficienti (xew). Successivamente, viene creato un grafico con due sottografici (uno per ciascuna serie temporale).

```
xrw = c(1,2,5,3) # Inizializza la serie temporale xrw con i primi quattro valori
sigma2 = 5 # Definisce la varianza del termine di errore distribuito normalmente

# Il loop for simula la serie temporale xrw. Ad ogni passo, un coefficiente viene selezionato
ciclicamente da xrw e viene aggiunto un termine di errore generato da rnorm(1, 0, sigma2^0.5)
for(i in 5:(n+5)){
  xrw[i] = xrw[i%%s+1]+ rnorm(1,0,sigma2^0.5)
}

xew = xrw[-c(1,2,3,4,5)]# Crea la serie temporale xew rimuovendo i primi cinque valori da xrw

par(mfrow=c(1,2))
plot(xrw[1:200], type="l") #  Crea il primo grafico a linee mostrando i primi 200 valori dell
a serie temporale xrw
acf(xrw) # Calcola e visualizza la funzione di autocorrelazione (ACF) della serie temporale x
rw nel secondo grafico
```



In sintesi, il primo grafico mostra la dinamica della serie temporale xrw con coefficienti ciclici, mentre il secondo grafico mostra la funzione di autocorrelazione di questa serie temporale. La serie temporale xew è creata rimuovendo i primi cinque valori da xrw e non ha più coefficienti ciclici.