```
# BREVE INTRODUCCIÓN AL MANEJO DE CADENAS DE MARKOV CON R:
# Para trabajar en R con Cadenas de Markov instalamos la librería
# especializada "markovchain":

library("markovchain")

# Aquí vamos a trabajar un ejemplo de cadena con solo tres estados,
# que representan los estados del tiempo en una posición geográfica
# concreta (por ejemplo, Ontario). Asumimos que hay solo tres estados:
# Lluvia, Sol y Viento. La cadena de Markov (X_n) nos indica, en
# el día n-ésimo, cuál es el estado del tiempo.

# Comenzamos, pues, definiendo los estados de la cadena:
Estados <- c("a", "b", "c", "d", "e", "f", "g")

# A partir de ahora weatherStates indica el espacio de estados de la cadena.

# Ahora introducimos la matriz de transición de probabilidades P=(p_{ij}).
# En nuestro caso, al haber solo tres estados, la matriz es 3x3.

byRow <- TRUE # Con esto indicamos que se introduce la matriz por filas:

Q <- matrix(data =
c(0,0,0,0,1/2,1/2,0,1/3,0,1/3,0,0,1/3,0,0,0,0,1/2,0,1/2,0,0,0,0,0,0,1,0,1/4,0,0,1/4,0,1/4,1/4,1/2,1/2,0,0,0,0,0,1/7,1/7,1/7,1/7,1/7,1/7,1/
byrow = byRow, nrow = 7,dimnames = list(Estados, Estados))
Q
A=matrix(data=rep(1/7,49),byrow = byRow, nrow = 7,dimnames = list(Estados, Estados))
A

p=0.85
P=p*Q+(1-p)*A
P
Matriz<-P
# A partir de ahora weatherMatrix indica la matriz de transición de la cadena.

mcRed <- new("markovchain", states = Estados, byrow = byRow,transitionMatrix = Matriz, name = "Red")
mcRed

# Por ejemplo, la probabilidad de pasar del "Lluvia" a "Lluvia" viene dada por:

transitionProbability(mcRed, "b", "f")

# Y también se puede obtener mirando la matriz:

mcRed[2,6]


# Vamos a proceder a simular la cadena, a partir de una distribución de probabilidades
# inicial conocida:

initialState <- c(0, 1, 0,0,0,0,0)

after2Days <- initialState * (mcRed * mcRed)
after2Days

#transitionProbability(mcWeather, "2", "3")
#mcWeather[2,3]

Evolucion <- rmarkovchain(n = 365000, object = mcRed, t0 = "a")
x=Evolucion
x[1:30]

# Una aproximación de la distribución estacionaria sería:
table(x[1:5000])/length(x[1:5000]) # Esto debería aproximar la distribución estacionaria

# O también:

barplot(table(x[1:5000])/length(x[1:5000])) # Esto también debería aproximar la distribución estacionaria
```

```
# Ahora bien, R nos la proporciona directamente:

print("La distribución estacionaria es:")

steadyStates(mcRed) # Esto nos proporciona la distribución estacionaria (si existe)

sort(steadyStates(mcRed))

barplot(steadyStates(mcRed))

# Si nos apetece visualizar la evolución de las frecuncias de los distintos estados,
# podemos proceder con un programa "a mano". Para ello, necesitamos numerar los estados
# cambiando por ejemplo weatherStates <- c("Lluvia", "Sol", "Viento") por:

Estados2 <- c("1", "2", "3","4","5", "6", "7")

Matriz2 <- matrix(data =
c(0,0,0,0,1/2,1/2,0,1/3,0,1/3,0,0,1/3,0,0,0,0,1/2,0,1/2,0,0,0,0,0,0,1,0,1/4,0,0,1/4,0,1/4,1/4,1/2,1/2,0,0,0,0,0,1/7,1/7,1/7,1/7,1/7,1/7,1/
byrow = byRow, nrow = 7,dimnames = list(Estados2, Estados2))
Matriz2

mcRed2 <- new("markovchain", states = Estados2, byrow = byRow,transitionMatrix = Matriz2, name = "Red")
mcRed2

# Y ahora volvemos a definir la cadena y a simularla:

Evolucion2 <- rmarkovchain(n = 365000, object = mcRed2, t0 = "2")

# El resultado lo podemos convertir ahora en una sucesión numérica y visualizarla
# en un gráfico (solo los primeros 1000 valores, por ejemplo):

x=as.numeric(Evolucion2)
show(plot(x[1:1000],xlab = paste("Evolución de la cadena de Markov")))

#
# prob_evol1=c()
# prob_evol2=c()
# prob_evol3=c()
# prob_evol4=c()
# prob_evol5=c()
# prob_evol6=c()
# prob_evol7=c()
#
# for(i in 1:50){
#   prob_evol1[i]=table(x[1:i][x==1])/i
#   prob_evol2[i]=table(x[1:i][x==2])/i
#   prob_evol3[i]=table(x[1:i][x==3])/i
#   prob_evol4[i]=table(x[1:i][x==4])/i
#   prob_evol5[i]=table(x[1:i][x==5])/i
#   prob_evol6[i]=table(x[1:i][x==6])/i
#   prob_evol7[i]=table(x[1:i][x==7])/i
# }
#
#
#
# colores=c("verde", "azul", "naranja")
# s1=prob_evol1
# plot(c(1:50),s1,type="l",col="green",xlab = paste("Evolución de las frecuencias de los distintos estados"),ylim
=c(0,1))
#
# s2=prob_evol2
# lines(c(1:50),s2,type="l",col="blue")
#
# s3=prob_evol3
# lines(c(1:50),s3,type="l",col="orange")
#
# s4=prob_evol4
# lines(c(1:50),s4,type="l",col="blue")
#
```

```
# s5=prob_evol5
# lines(c(1:50),s5,type="l",col="orange")
#
# s6=prob_evol6
# lines(c(1:50),s6,type="l",col="blue")
#
# s7=prob_evol7
# lines(c(1:50),s7,type="l",col="orange")


print("La distribución estacionaria es:")

steadyStates(mcRed2) # Esto nos proporciona la distribución estacionaria (si existe)


table(x[1:10000])/length(x[1:10000]) # Esto debería aproximar la distribución estacionaria


# IMPORTANTE: ES EVIDENTE QUE EL CÁLCULO ANTERIOR LLEVA DEMASIADO TIEMPO
# DE EJECUCIÓN Y TIENE, POR TANTO, SOLO UN INTERÉS DIDÁCTICO. AÚN ASÍ
# ES MUCHO MÁS RÁPIDO QUE EL MISMO PROGRAMA HECHO POR EJEMPLO EN SAGEMATH

# A markovchain le podemos preguntar muchas propiedades de las cadenas de Markov

absorbingStates(mcRed)


transientStates(mcRed)



is.irreducible(mcRed)


# Si le pedimos de golpe lo que sabe, hacemos esto:

summary(mcRed)


# Además, podemos dibujar el grafo asociado a la cadena:

plot(mcRed)
#
# mcDf<-as(mcRed,"data.frame")
# mcNew<-as(mcRed,"markovchain")
# mcIgraph<-as(mcRed,"igraph")
# mcIgraph
# plot(mcIgraph)
```