UNIVERSIDAD AUTONOMA DE COAHUILA

FACULTAD DE SISTEMAS

ALUMNO: RICARDO GABRIEL RODRIGUEZ GONZALEZ

MATRICULA: 17001433

MODELOS COMPUTACIONALES

TAREA 3

PROFESORA: VALERIA SOTO MENDOZA

06 DE ABRIL DE 2022

Tarea 3

Ricardo Gabriel Rodriguez Gonzalez

2022-03-16

Ejercicio 1.

Un agente de ventas realiza su trabajo en tres localidades A, B y C. Para evitar desplazamientos innecesarios está todo el día en la misma ciudad y allí pernocta, desplazándose a otra localidad al día siguiente, si no tiene suficiente trabajo. después de estar trabajando un día en C, la probabilidad de tener que seguir trabajando en ella al día siguiente es 0.4, la de tener que viajar a B es 0.4, y la de tener que ir a A es 0.2. Si el viajante duerme un día en B, con probabilidad de 20% tendrá que seguir trabajando en la misma localidad al día siguiente en el 60% de los casos viajará a C mientras que irá a A con probabilidad 0.2. Por último si el agente de ventas trabaja todo un día en A, permanecerá en esa misma localidad, al día siguiente, con una probabilidad 0.1, irá a B con una probabilidad 0.3 y a C con una probabilidad de 0.6.

$X(t)$ =la ubicación del vendedor en el día t

Los estados: $M = A, B, C$ Correspondientes a las tres diferentes ciudades.

Temporalidad: Por día

$\left[\begin{array}0 0.1,0.3,0.6\\0,2,0.2,0.6\\0.2,0.4,0.4\end{array}\right]$

```
library(markovchain)
```

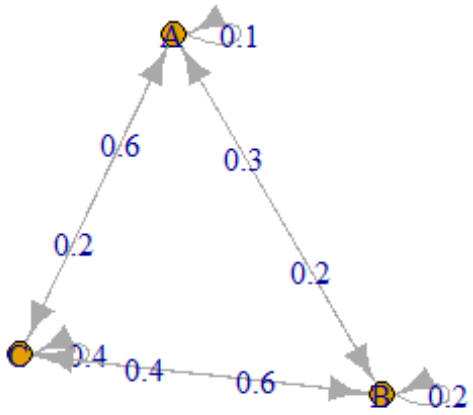
```
## Package:  markovchain
## Version:  0.8.6
## Date:     2021-05-17
## BugReport: https://github.com/spedygiorgio/markovchain/issues
```

```
statesNames = c("A","B","C")
mc_p1 <- new("markovchain",transitionMatrix = matrix(c(0.1,0.3,0.6,0.2,0.
2,0.6,0.2,0.4,0.4),byrow = TRUE,nrow = 3,dimnames = list(statesNames,stat
esNames)))
mc_p1
```

```
## Unnamed Markov chain
##  A  3 - dimensional discrete Markov Chain defined by the following sta
tes:
##  A, B, C
##  The transition matrix  (by rows)  is defined as follows:
##     A   B   C
## A 0.1 0.3 0.6
## B 0.2 0.2 0.6
## C 0.2 0.4 0.4
```

Digarama de transición de estados:

```
plot(mc_p1)
```



a) Si hoy el vendedor está en C, Cúal es la probabilidad de que también tenga que trabajar en C al cabo de cuatro días?

```
#Calculamos la matriz de 4 pasos
mc_p1^4
```

```
## Unnamed Markov chain^4
##  A  3 - dimensional discrete Markov Chain defined by the following sta
tes:
##  A, B, C
##  The transition matrix  (by rows)  is defined as follows:
##        A      B      C
## A 0.1819 0.3189 0.4992
## B 0.1818 0.3190 0.4992
## C 0.1818 0.3174 0.5008
```

La probabilidad de que también tenga que trabajar en C al cabo de cuatro días es 0.5008 o 50.08%

b) Cuales son los porcentajes de días en los que el agente de ventas está en cada una de las tres localidades?

```
print(mc_p1^7)
```

```
##           A         B         C
## A 0.1818181 0.3181755 0.5000064
## B 0.1818182 0.3181754 0.5000064
## C 0.1818182 0.3181882 0.4999936
```

```
print(mc_p1^30)
```

```
##           A         B   C
## A 0.1818182 0.3181818 0.5
## B 0.1818182 0.3181818 0.5
## C 0.1818182 0.3181818 0.5
```

```
print(mc_p1^365)
```

```
##           A         B   C
## A 0.1818182 0.3181818 0.5
## B 0.1818182 0.3181818 0.5
## C 0.1818182 0.3181818 0.5
```

Los porcentajes de días en los que el agente de ventas está en cada una de las tres localidades son 18.18%, 31.81% y 50%, respectivamente.

Ejercicio 2.

Una computadora se inspecciona cada hora. Se encuentra que está trabajando o que está descompuesta. En el primer caso, la probabilidad de que siga así la siguiente hora es de 0.95. Si está descompuesta, se repara, lo que puede llevar más de una hora. Siempre que la computadora esté descompuesta (sin importar cuánto tiempo pase), la probabilidad de que siga descompuesta una hora más es de 0.5

Mi variable aleatoria es: $X(T) = {$Inspección de una computadora en la hora t$}$

Los estados: $M = {"Trabajando","Descompuesta"}$ Dos estados.

Temporalidad por hora.

$\left[\begin{array}0 0.95,0.05\\0,5,0.5\end{array}\right]$

```
statesNames = c("Trabajando","Descompuesta")
mc_p2 <- new("markovchain",transitionMatrix = matrix(c(0.95,0.05,0.5,0.5)
,byrow = TRUE,nrow = 2,dimnames = list(statesNames,statesNames)))
mc_p2
```

```
## Unnamed Markov chain
## A  2 - dimensional discrete Markov Chain defined by the following stat
es:
##  Trabajando, Descompuesta
##  The transition matrix  (by rows)  is defined as follows:
##              Trabajando Descompuesta
## Trabajando         0.95         0.05
## Descompuesta       0.50         0.50
```

b)Encuentre el tiempo

```
print( meanFirstPassageTime(mc_p2))
```

```
##              Trabajando Descompuesta
## Trabajando            0           20
## Descompuesta          2            0
```

Ejercicio 3.

Una partícula se mueve sobre un círculo por puntos 0, 1, 2, 3, 4 (en el sentido de las manecillas del reloj). La partícula comienza en el punto 0. En cada paso tiene probabilidad de 0.5 de moverse un punto en el sentido de las manecillas del reloj (0 sigue al 4) y un aprobabilidad de 0.5 de moverse en punto en el sentido opuesto. Sea Xn(n>=0) la localización en el círculo después de paso n. {Xn} es entonces una cadena de Markov.

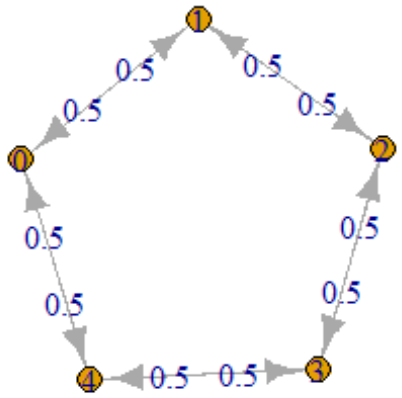
Mi variable aleatoria es: $X(T)$ =La posición de la partícula dentro de un circulot.

Los estados: $M = {“0”,“1”,“2”,“3”,“4”}$5 estados correspondientes a los puntos marcados en el círculo.

Temporalidad: Por paso
$\left[\begin{array}00,0.5,0,0,0.5\\0.5,0,0.5,0,0\\0,0.5,0,0.5,0\\0,0,0.5,0,0.5\\0.5,0,0,0.5,0\end{array}\right]$

a)Encuentre la matriz de transición(de un paso)

```
statesNames = c("0","1","2","3","4")
mc_p3 <- new("markovchain",transitionMatrix = matrix(c(0,0.5,0,0,0.5,0.5,
0,0.5,0,0,0,0.5,0,0.5,0,0,0,0.5,0,0.5,0.5,0,0,0.5,0),byrow = TRUE,nrow =
5,dimnames = list(statesNames,statesNames)))
plot(mc_p3)
```



b)Utilice el alguna herramienta computacional para procesar lo siguiente

i)Determinar la matriz de transición de n pasos P^n para n = 5,10,20,40, 80.

```
print(mc_p3^5)
```

```
##         0       1       2       3       4
## 0 0.06250 0.31250 0.15625 0.15625 0.31250
## 1 0.31250 0.06250 0.31250 0.15625 0.15625
## 2 0.15625 0.31250 0.06250 0.31250 0.15625
## 3 0.15625 0.15625 0.31250 0.06250 0.31250
## 4 0.31250 0.15625 0.15625 0.31250 0.06250
```

```
print(mc_p3^10)
```

```
##           0         1         2         3         4
## 0 0.2480469 0.1611328 0.2148438 0.2148438 0.1611328
## 1 0.1611328 0.2480469 0.1611328 0.2148438 0.2148438
## 2 0.2148438 0.1611328 0.2480469 0.1611328 0.2148438
## 3 0.2148438 0.2148438 0.1611328 0.2480469 0.1611328
## 4 0.1611328 0.2148438 0.2148438 0.1611328 0.2480469
```

```
print(mc_p3^20)
```

```
##           0         1         2         3         4
## 0 0.2057705 0.1953316 0.2017832 0.2017832 0.1953316
## 1 0.1953316 0.2057705 0.1953316 0.2017832 0.2017832
## 2 0.2017832 0.1953316 0.2057705 0.1953316 0.2017832
```

```
## 3 0.2017832 0.2017832 0.1953316 0.2057705 0.1953316
## 4 0.1953316 0.2017832 0.2017832 0.1953316 0.2057705

print(mc_p3^40)

##           0         1         2         3         4
## 0 0.2000832 0.1999327 0.2000257 0.2000257 0.1999327
## 1 0.1999327 0.2000832 0.1999327 0.2000257 0.2000257
## 2 0.2000257 0.1999327 0.2000832 0.1999327 0.2000257
## 3 0.2000257 0.2000257 0.1999327 0.2000832 0.1999327
## 4 0.1999327 0.2000257 0.2000257 0.1999327 0.2000832

print(mc_p3^80)

##     0   1   2   3   4
## 0 0.2 0.2 0.2 0.2 0.2
## 1 0.2 0.2 0.2 0.2 0.2
## 2 0.2 0.2 0.2 0.2 0.2
## 3 0.2 0.2 0.2 0.2 0.2
## 4 0.2 0.2 0.2 0.2 0.2
```

ii)Determinar las probabilidades de estado estable de los estados de la cadena de Markov. Compare las probabilidades de la matriz de transición de n pasos que se obtuvo en el inciso anterior con estas probabiidades de estado establo conforme n crece.

```
print(steadyStates(mc_p3))

##        0   1   2   3   4
## [1,] 0.2 0.2 0.2 0.2 0.2
```

Ejercicio 4.

Un juego de lanzamiento de dados utiliza una cuadrícula de cuatro casillas. Las casillas están designadas en sentido horario como A, B, C y D con retribuciones monetarias $4,$-2,$-6,$9, respectivamente. Comenzando en la casilla A, lanzamos el dado para determinar la siguiente casilla a la que nos moveremos en el sentido de las manecillas del reloj. Por ejemplo, si el dado muestra 2, nos movemos a la casilla C. El juego se repite utilizando la última casilla como punto inicial.

Mi variable aleatoria es: $X(T)$ =La ubicación en la cuadrícula en el lanzamiento t

Los estados: $M = {"A","B","C","D"}$ 4 estados correspondientes a las casillas.

Temporalidad: por lanzamiento $\left[\begin{array}0
1/6,2/6,2/6,1/6\\1/6,1/6,2/6,2/6\\2/6,1/6,1/6,2/6\\2/6,2/6,1/6,1/6\end{array}\right]$

```
statesNames = c("A","B","C","D")
mc_p4 <- new("markovchain",transitionMatrix = matrix(c(1/6, 2/6, 2/6, 1/6
```

```
, 1/6, 1/6, 2/6, 2/6, 2/6, 1/6, 1/6, 2/6, 2/6, 2/6, 1/6, 1/6),byrow = TRU
E,nrow = 4,dimnames = list(statesNames,statesNames)))
mc_p4

## Unnamed Markov chain
##  A  4 - dimensional discrete Markov Chain defined by the following sta
tes:
##  A, B, C, D
##  The transition matrix  (by rows)  is defined as follows:
##           A         B         C         D
## A 0.1666667 0.3333333 0.3333333 0.1666667
## B 0.1666667 0.1666667 0.3333333 0.3333333
## C 0.3333333 0.1666667 0.1666667 0.3333333
## D 0.3333333 0.3333333 0.1666667 0.1666667
```

a) Exprese el problema como una cadena de markov.

```
print(mc_p4)

##           A         B         C         D
## A 0.1666667 0.3333333 0.3333333 0.1666667
## B 0.1666667 0.1666667 0.3333333 0.3333333
## C 0.3333333 0.1666667 0.1666667 0.3333333
## D 0.3333333 0.3333333 0.1666667 0.1666667
```

b)Determine la ganancia o pérdida despues de lanzar el dado 5 veces

```
#Se calcula la matriz de 5 pasos
m5p4 <- mc_p4^5
print(m5p4)

##           A         B         C         D
## A 0.2502572 0.2497428 0.2497428 0.2502572
## B 0.2502572 0.2502572 0.2497428 0.2497428
## C 0.2497428 0.2502572 0.2502572 0.2497428
## D 0.2497428 0.2497428 0.2502572 0.2502572

#calcular la ganancia/pérdida esperados
Ct <- c(4,-2,-6,9)
print(sum(m5p4*Ct))

## [1] 5
```

La ganancia esperada despues de lanzr el dado 5 veces es de $5

Ejercicio 5.

Los estudiantes en una universidad han expresado un disgusto por el rápido paso al cual el departamento de matemáticas está impartiendo el Cal 1 de un semestre. para afrontar este problema, el departamento de matemáticas ahora está ofreciento Cal 1 en 4 módulosl Los estudiantes establecerán su paso individual para cada módulo y,

cuando estén listos, harán un examen que los llevará al siguiente módulo. Los exámenes se aplican una vez cada 4 semanas, de modo que un estudiante diligente puede completar 4 módulos en un semestre. Después de un par de años con este programa, 20% de los estudiantes completa el primer módulo a tiempo. Los porcentajes para los módulos del 2 al 4 fueros de 22, 25 y 30%, respectivamente.

Mi variable aleatoria es: $X(T)$ =El paso de cal 1 de un estudiante

Los estados: $M = {"Modulo 1","Modulo 2","Modulo 3","Modulo 4","Cal 2"}$ 5 Modulos de cal 1 e inicio de cal 2.

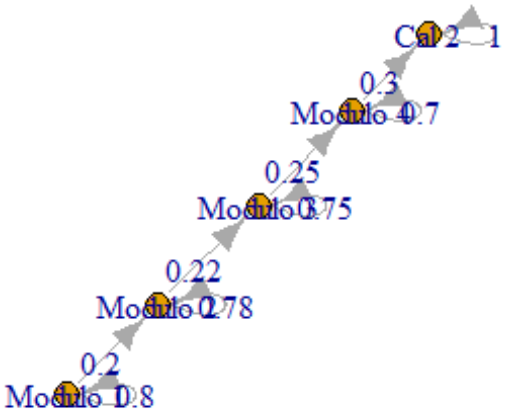
Temporalidad: por mes(4 Semanas) $\left[\begin{array}0 0.8,0.2,0,0,0\\0,0.78,0.22,0,0\\0,0,0.75,0.25,0\\0,0,0,0.7,0.3\\0,0,0,0,1\end{array}\right]$

a) Exprese el problema como una cadena de Markov. ***Propiedad markoviana para los 5 estados***

```
library(markovchain)
statesNames = c("Modulo 1","Modulo 2","Modulo 3","Modulo 4","Cal 2")
mc_p5 <- new("markovchain",transitionMatrix = matrix(c(0.8,0.2,0,0,0,0,0.
78,0.22,0,0,0,0,0.75,0.25,0,0,0,0,0.7,0.3,0,0,0,0,1),byrow = TRUE,nrow =
5,dimnames = list(statesNames,statesNames)))
mc_p5
```

```
## Unnamed Markov chain
##  A  5 - dimensional discrete Markov Chain defined by the following sta
tes:
##  Modulo 1, Modulo 2, Modulo 3, Modulo 4, Cal 2
##  The transition matrix  (by rows)  is defined as follows:
##          Modulo 1 Modulo 2 Modulo 3 Modulo 4 Cal 2
## Modulo 1      0.8     0.20     0.00     0.00   0.0
## Modulo 2      0.0     0.78     0.22     0.00   0.0
## Modulo 3      0.0     0.00     0.75     0.25   0.0
## Modulo 4      0.0     0.00     0.00     0.70   0.3
## Cal 2         0.0     0.00     0.00     0.00   1.0
```

```
plot(mc_p5)
```



b)En promedio, un estudiante que inicio el modulo 1 al principio el semestre actual ¿será capaz de llevar el módulo 2 el siguiente semestre? (El cal 1 es un prerrequisito para el cal 2)

```
#Debemos obtener la matriz a 5 pasos
print(mc_p5^5)
```

```
##          Modulo 1  Modulo 2  Modulo 3  Modulo 4     Cal 2
## Modulo 1  0.32768 0.3896256 0.2062455 0.0631499 0.0132990
## Modulo 2  0.00000 0.2887174 0.3770268 0.2262319 0.1080239
## Modulo 3  0.00000 0.0000000 0.2373047 0.3461734 0.4165219
## Modulo 4  0.00000 0.0000000 0.0000000 0.1680700 0.8319300
## Cal 2     0.00000 0.0000000 0.0000000 0.0000000 1.0000000
```

```
# La probabilidad que nos interesa es modulo 1 -> modulo 2
```

En promedio, un estudiante que que inició el modulo 1 al principio del semestre actual será capaz de llevar el módulo 2 el siguiente semestre con una probabilidad de 0.3896

c) Un estudiante que haya completado sólo un módulo el semestre anterior ¿será capaz de terminar el cal 1 al final del semestre actual?

```
# Como el terminar cal 1 quiere decir que llegara al estado cal 2. Y como
cal 2 es un estado absorbente, calculamos la probabilidad de absorbencia.

print(absorptionProbabilities(mc_p5))
```

```
##          Cal 2
## Modulo 1     1
## Modulo 2     1
## Modulo 3     1
## Modulo 4     1

print(meanAbsorptionTime(mc_p5))

##  Modulo 1  Modulo 2  Modulo 3  Modulo 4
## 16.878788 11.878788  7.333333  3.333333
```

El 1.3% de los estudiantes que hayan completado el modulo 1 al principio del semestre será capaz de terminar el cal 1 al final de semestre actual.El 10.80% de los estudiantes que hayan completado el modulo 2 al principio del semestre será capaz de terminar el cal 1 al final de semestre actual. El 41.6% de los estudiantes que hayan completado el modulo 3 al principio del semestre será capaz de terminar el cal 1 al final de semestre actual. El 83.1% de los estudiantes que hayan completado el modulo 4 al principio del semestre será capaz de terminar el cal 1 al final de semestre actual.

d) ¿Recoienda aplicar la idea del módulo a otras materias básicas? Explique.

```
 print(steadyStates(mc_p5))

##      Modulo 1 Modulo 2 Modulo 3 Modulo 4 Cal 2
## [1,]        0        0        0        0     1

print(mc_p5^10)

##           Modulo 1   Modulo 2   Modulo 3   Modulo 4     Cal 2
## Modulo 1 0.1073742 0.24016424 0.26342485 0.19084901 0.1981877
## Modulo 2 0.0000000 0.08335776 0.19832445 0.23385656 0.4844612
## Modulo 3 0.0000000 0.00000000 0.05631351 0.14032995 0.8033565
## Modulo 4 0.0000000 0.00000000 0.00000000 0.02824752 0.9717525
## Cal 2    0.0000000 0.00000000 0.00000000 0.00000000 1.0000000

print(mc_p5^15)

##            Modulo 1   Modulo 2   Modulo 3    Modulo 4     Cal 2
## Modulo 1 0.03518437 0.11117534 0.17520575 0.184380155 0.4940544
## Modulo 2 0.00000000 0.02406684 0.07849143 0.126817112 0.7706246
## Modulo 3 0.00000000 0.00000000 0.01336346 0.043079498 0.9435570
## Modulo 4 0.00000000 0.00000000 0.00000000 0.004747562 0.9952524
## Cal 2    0.00000000 0.00000000 0.00000000 0.000000000 1.0000000

print(mc_p5^120)

##              Modulo 1     Modulo 2     Modulo 3     Modulo 4 Cal 2
## Modulo 1 2.348543e-12 2.235991e-11 9.511191e-11 2.326957e-10     1
## Modulo 2 0.000000e+00 1.125518e-13 8.179211e-13 2.542022e-12     1
## Modulo 3 0.000000e+00 0.000000e+00 1.017072e-15 5.084068e-15     1
```

```
## Modulo 4 0.000000e+00 0.000000e+00 0.000000e+00 2.580862e-19     1
## Cal 2    0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00     1
```

Ejercicio 6.

Un fabricante de videograbadoras está tan seguro de su calidad que ofrece garantía de reposición total si un aparato falla dentro de los dos primeros años. Con base en datos compilados, la compañía ha notado que sólo 1% de sus grabadoras fallan durante el primer año, mientras que 5% de ellas sobreviven el primer año pero fallan durante el segundo. La garantía no cubre grabadoras ya reemplazadas.

Mi variable aleatoria es:

$X(t) = {$La garantía de reposición$}$

Estados:

$M={"A", "B", "C","D"}$

Temporalidad:

Por año

$\left[\begin{array} 00,0.99,0.01,0 \\ 0,0,0.05,0.95 \\ 0,0,1,0 \\ 0,0,0,1 \end{array}\right]$

a)Formule la evolución del estado de una grabadora como una cadena de Markov cuyos estados incluyen dos estados absorbentes que representan la necesidad de cubrir la garantía o el hecho de que una grabadora sobreviva el periodo de garantía. Después construya la matriz de transición (de un paso).

```
m <- matrix(c(0,0.99,0.01,0,0,0,0.05,0.95,0,0,1,0,0,0,0,1), ncol = 4, byr
ow = TRUE)
mc_p6 <- new('markovchain',states = c("Primer año funcionando", "Segundo
año funcionando", "Se aplica garantía", "Finaliza la garantía"), transiti
onMatrix=m)
mc_p6
```

```
## Unnamed Markov chain
##  A  4 - dimensional discrete Markov Chain defined by the following sta
tes:
##  Primer año funcionando, Segundo año funcionando, Se aplica garantía,
Finaliza la garantía
##  The transition matrix  (by rows)  is defined as follows:
##                          Primer año funcionando Segundo año funcionando
## Primer año funcionando                        0                    0.99
## Segundo año funcionando                       0                    0.00
## Se aplica garantía                            0                    0.00
## Finaliza la garantía                          0                    0.00
##                          Se aplica garantía Finaliza la garantía
```

```
## Primer año funcionando                0.01                 0.00
## Segundo año funcionando               0.05                 0.95
## Se aplica garantía                    1.00                 0.00
## Finaliza la garantía                  0.00                 1.00
```

b) Encuentre la probabilidad de que el fabricante tenga que cubrir una garantía.

```
absorptionProbabilities(mc_p6)

##                          Se aplica garantía Finaliza la garantía
## Primer año funcionando               0.0595               0.9405
## Segundo año funcionando              0.0500               0.9500
```

La probabilidad que hay de que el fabricante tenga que cubir alguna garantía es de 0.1095.

Ejercicio 7.

Una agencia de renta de automóviles tiene oficinas en Phoenix, Denver, Chicago y Atlanta. La agencia permite rentas de una y en dos direcciones de modo que los automóviles rentados en un lugar pueden terminar en otro. Las estadísticas muestran que al final de cada semana 70% de todas las rentas son en dos direcciones. En cuanto a las rentas en una dirección: Desde Phoenix, 20% van a Denver, 60% a Chicago, y el resto va a Atlanta; desde Denver, 40% va a Atlanta y 60% a chicago, 50% va a Atlanta y el resto a Denver; y desde Atlanta, 80% va a Chicago, 10% a Denver, y 10% a Phoenix.

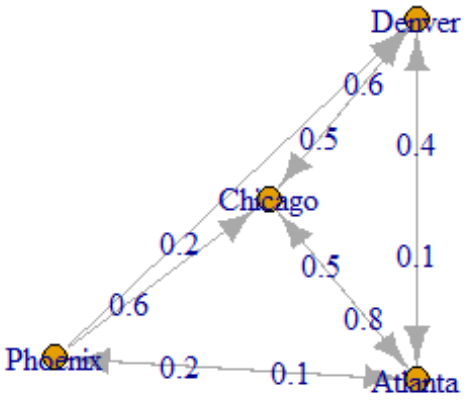
Mi variable aleatoria es: $X(T)$ =La ubicación donde se enccuentra un auto

Los estados: $M = {"Phoenix","Denver","Chicago","Atlanta"}$ 4 estados/ciudades.

Temporalidad: Por semana $\left[\begin{array}0
0,0.2,0.6,0.2\\0,0,0.6,0.4\\0,0.5,0,0.5\\0.1,0.1,0.8,0\end{array}\right]$

a) Exprese la situación como una cadena de Markov.

```
statesNames = c("Phoenix","Denver","Chicago","Atlanta")
mc_p7 <- new("markovchain",transitionMatrix = matrix(c(0,0.2,0.6,0.2,0,0,
0.6,0.4,0,0.5,0,0.5,0.1,0.1,0.8,0),byrow = TRUE,nrow = 4,dimnames = list(
statesNames,statesNames)))
print(mc_p7)

##         Phoenix Denver Chicago Atlanta
## Phoenix     0.0    0.2     0.6     0.2
## Denver      0.0    0.0     0.6     0.4
## Chicago     0.0    0.5     0.0     0.5
## Atlanta     0.1    0.1     0.8     0.0

plot(mc_p7)
```




```
print(summary(mc_p7))
```

```
## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## Phoenix Denver Chicago Atlanta
## Recurrent classes:
## {Phoenix,Denver,Chicago,Atlanta}
## Transient classes:
## NONE
## The Markov chain is irreducible
## The absorbing states are: NONE
## $closedClasses
## $closedClasses[[1]]
## [1] "Phoenix" "Denver"  "Chicago" "Atlanta"
##
##
## $recurrentClasses
## $recurrentClasses[[1]]
## [1] "Phoenix" "Denver"  "Chicago" "Atlanta"
##
##
## $transientClasses
## list()
```

b) Si la agencia inicia la semana con 100 autos en cada lugar, ¿cómo será la distribucion en dos semanas?

```
print(mc_p7^2)
```

```
##         Phoenix Denver Chicago Atlanta
## Phoenix    0.02   0.32    0.28    0.38
## Denver     0.04   0.34    0.32    0.30
## Chicago    0.05   0.05    0.70    0.20
## Atlanta    0.00   0.42    0.12    0.46
```

```
autos <- c(100,100,100,100)
print(autos*(mc_p7^2))
```

```
##      Phoenix Denver Chicago Atlanta
## [1,]      11    113     142     134
```

La distribución después de dos semanas será de 11, 113, 142y 134 autos para cada ciudad Phoenix, Denver, Chicago y Atlanta, respectivamente

c) Si cada lugar está diseñado para manejar un máximo de 110 autos, ¿habría a la larga un problema de disponibilidad de espacio en cualquiera de los lugares?

```
probEstadoEstable <- steadyStates(mc_p7)
print(autos*probEstadoEstable)
```

```
##       Phoenix   Denver  Chicago  Atlanta
## [1,] 3.108348 24.42274 41.38544 31.08348
```

A la larga no habria un problema de disponibilidad de espacio en cualquiera de los lugares.

d) Determine el promedioi de semanas que transcurren antes de que un auto regrese a su lugar de origen.

```
print(meanRecurrenceTime(mc_p7))
```

```
##   Phoenix    Denver   Chicago   Atlanta
## 32.171429  4.094545  2.416309  3.217143
```

El promedio de semanas que transcurren antes de que un auto regrese a su lugar de origen es de 32, 4, 2 y 3 semanas, respectivamente

Ejercicio 8.

Suponga que una red de comunicaciones transmite dígitos binarios, 0 o 1, y que cada dígito se transmite 10 veces sucesivas. Durante cada transmisión, la probabilidad de que ese dígito se transmita correctamente es de 0.995. En otras palabras, existe una probabilidad de 0.005 de que el dígito transmitido se registre con el valor opuesto al final de la transmisión. Para cada transmisión después de la primera, el dígito transmitido es el que se registra al final de la transmisión anterior. Si X0 denota el dígito binario que entra al sistema, X1 el dígito binario que se apunta después de la primera transmisión, X2 el dígito binario que se anota después de la segunda transmisión, . . ., entonces {Xn} es una cadena de Markov.

La variable aleatoria es: $X(t) = digitos$.

Los estados: $M = {"0","1"}$

Matriz de transición (a 1 paso):

$\left[\begin{array},0.995,0.005\\0.005,0.995\end{array}\right]$

```
library(markovchain)
statesNames = c("0","1")
mc_p8 <- new ("markovchain",transitionMatrix = matrix (c(0.995,0.005,0.00
5,0.995), byrow = TRUE, nrow = 2, dimnames=list(statesNames,statesNames))
)
mc_p8

## Unnamed Markov chain
##  A  2 - dimensional discrete Markov Chain defined by the following sta
tes:
##  0, 1
##  The transition matrix  (by rows)  is defined as follows:
##       0     1
## 0 0.995 0.005
## 1 0.005 0.995
```

a) Determine la matriz de transición (de un paso).

```
print(mc_p8)

##       0     1
## 0 0.995 0.005
## 1 0.005 0.995
```

b) Encuentre la matriz de transición de 10 pasos P(10). Utilice este resultado para identificar la probabilidad de que un dígito que entra a la red se registre correctamente después de la última transmisión.

```
print(mc_p8^10)

##            0          1
## 0 0.95219104 0.04780896
## 1 0.04780896 0.95219104
```

c) Suponga que la red sere diseña para mejorar la probabilidad de la exactitud de una sola transmisión de 0.995 a 0.998. Repita el inciso b) para encontrar la nueva probabilidad de que un dígito que entra a la red se registre correctamente después de la última transmisión.

```
library(markovchain)
statesNames = c("0","1")
mc2_p8 <- new ("markovchain",transitionMatrix = matrix (c(0.998,0.002,0.0
02,0.998), byrow = TRUE, nrow = 2, dimnames=list(statesNames,statesNames)
))
mc2_p8

## Unnamed Markov chain
##  A  2 - dimensional discrete Markov Chain defined by the following sta
```

```
tes:
##  0, 1
##  The transition matrix  (by rows)  is defined as follows:
##       0     1
## 0 0.998 0.002
## 1 0.002 0.998

print(mc2_p8^10)

##            0          1
## 0 0.98035619 0.01964381
## 1 0.01964381 0.98035619
```

Ejercicio 9.

Dadas las siguientes matrices de transición (de un paso) de una cadena de Markov, determine las clases de las cadenas de Markov y si son recurrentes o no.

a) $\left[\begin{array}0 0, 0, 1/3, 2/3\\1, 0, 0, 0\\ 0, 1, 0, 0\\0, 1, 0, 0\end{array}\right]$

```
library(markovchain)
statesNames = c("0", "1", "2", "3")
mc_p9a <- new("markovchain", transitionMatrix = matrix(c(0, 0, 1/3, 2/3,
1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0), byrow = TRUE, nrow = 4, dimnames = l
ist(statesNames,statesNames)))

summary(mc_p9a)

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## 0 1 2 3
## Recurrent classes:
## {0,1,2,3}
## Transient classes:
## NONE
## The Markov chain is irreducible
## The absorbing states are: NONE

print(mc_p9a)

##   0 1         2         3
## 0 0 0 0.3333333 0.6666667
## 1 1 0 0.0000000 0.0000000
## 2 0 1 0.0000000 0.0000000
## 3 0 1 0.0000000 0.0000000

#Estados Recurrentes
recurrentStates(mc_p9a)

## [1] "0" "1" "2" "3"
```

```
#Estados Transitorios
transientStates(mc_p9a)

## character(0)

#Clases Recurrentes
recurrentClasses(mc_p9a)

## [[1]]
## [1] "0" "1" "2" "3"

#Clases Transitorias
transientClasses(mc_p9a)

## list()
```

Tenemos que hay 4 clases que estan cerradas y una clase recurrente que utiliza 4 estados

b) $\left[\begin{array}0 1, 0, 0, 0\\0, 1/2, 1/2, 0\\ 0, 1/2, 1/2, 0\\1/2, 0, 0, 1/2\end{array}\right]$

```
statesNames = c("0", "1", "2", "3")

mc_p9b <- new("markovchain", transitionMatrix = matrix(c(1, 0, 0, 0, 0, 1
/2, 1/2, 0, 0, 1/2, 1/2, 0, 1/2, 0, 0, 1/2), byrow = TRUE, nrow = 4, dimn
ames = list(statesNames,statesNames)))

summary(mc_p9b)

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## 0
## 1 2
## Recurrent classes:
## {0},{1,2}
## Transient classes:
## {3}
## The Markov chain is not irreducible
## The absorbing states are: 0

print(mc_p9b)

##     0   1   2   3
## 0 1.0 0.0 0.0 0.0
## 1 0.0 0.5 0.5 0.0
## 2 0.0 0.5 0.5 0.0
## 3 0.5 0.0 0.0 0.5
```

No tenemos clases cerradas y nuestra clase recurrente esta en 0 y otras en 1 y 2, tambien tenemos clase transitoria que se crea gracias al estado 3

```
recurrentStates(mc_p9b)
```

```
## [1] "0" "1" "2"

transientStates(mc_p9b)

## [1] "3"
```

c) $\left[\begin{array}0 0, 1/3, 1/3, 1/3\\1/3, 0, 1/3, 1/3\\ 1/3, 1/3, 0, 1/3\\1/3, 1/3, 1/3, 0\end{array}\right]$

```
statesNames = c("0", "1", "2", "3")

mc_p9c <- new("markovchain", transitionMatrix = matrix(c(0, 1/3, 1/3, 1/3
, 1/3, 0, 1/3, 1/3, 1/3, 1/3, 0, 1/3, 1/3, 1/3, 1/3, 0), byrow = TRUE, nr
ow = 4, dimnames = list(statesNames,statesNames)))

summary(mc_p9c)

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## 0 1 2 3
## Recurrent classes:
## {0,1,2,3}
## Transient classes:
## NONE
## The Markov chain is irreducible
## The absorbing states are: NONE

print(mc_p9c)

##           0         1         2         3
## 0 0.0000000 0.3333333 0.3333333 0.3333333
## 1 0.3333333 0.0000000 0.3333333 0.3333333
## 2 0.3333333 0.3333333 0.0000000 0.3333333
## 3 0.3333333 0.3333333 0.3333333 0.0000000
```

Se cierran 4 clases y nos queda una clase que es recurrente que tiene 4 estados.

```
recurrentStates(mc_p9c)

## [1] "0" "1" "2" "3"
```

d) $\left[\begin{array}0 0, 0, 1\\1/2, 1/2, 0\\ 0, 1, 0\end{array}\right]$

```
statesNames = c("0", "1", "2")

mc_p9d <- new("markovchain", transitionMatrix = matrix(c(0, 0, 1, 1/2, 1/
2, 0, 0, 1, 0), byrow = TRUE, nrow = 3, dimnames = list(statesNames,state
sNames)))
summary(mc_p9d)

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## 0 1 2
```

```
## Recurrent classes:
## {0,1,2}
## Transient classes:
## NONE
## The Markov chain is irreducible
## The absorbing states are: NONE

print(mc_p9d)

##     0   1 2
## 0 0.0 0.0 1
## 1 0.5 0.5 0
## 2 0.0 1.0 0

recurrentStates(mc_p9d)

## [1] "0" "1" "2"
```

e) $\left[\begin{array}0 1/4, 3/4, 0, 0, 0\\3/4, 1/4, 0, 0, 0\\ 1/3, 1/3, 1/3, 0, 0\\ 0, 0, 0, 3/4, 1/4\\0, 0, 0, 1/4, 3/4\end{array}\right]$

```
statesNames = c("0", "1", "2", "3", "4")

mc_p9e <- new("markovchain", transitionMatrix = matrix(c(1/4, 3/4, 0, 0,
0, 3/4, 1/4, 0, 0, 0, 1/3, 1/3, 1/3, 0, 0, 0, 0, 0, 3/4, 1/4, 0, 0, 0, 1/
4, 3/4), byrow = TRUE, nrow = 5, dimnames = list(statesNames,statesNames)
))

summary(mc_p9e )

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## 0 1
## 3 4
## Recurrent classes:
## {0,1},{3,4}
## Transient classes:
## {2}
## The Markov chain is not irreducible
## The absorbing states are: NONE

print(mc_p9e )

##           0         1         2    3    4
## 0 0.2500000 0.7500000 0.0000000 0.00 0.00
## 1 0.7500000 0.2500000 0.0000000 0.00 0.00
## 2 0.3333333 0.3333333 0.3333333 0.00 0.00
## 3 0.0000000 0.0000000 0.0000000 0.75 0.25
## 4 0.0000000 0.0000000 0.0000000 0.25 0.75
```

Ahora nos quedan 2 clases reccurentes con 2 estados y 1 transitoria, en este caso tenemos 4 estados

```
recurrentStates(mc_p9e )
```

```
## [1] "0" "1" "3" "4"
```

Nuestra clase transitoria solamente tiene un estado

```
transientStates(mc_p9e )
```

```
## [1] "2"
```

Ejercicio 10.

En una Unidad de Cuidados Intensivos en un determinado hospital, cada paciente es clasificado de acuerdo a un estado crítico, serio o estable. Estas clasificaciones son actualizadas cada mañana por un médico internista,de acuerdo a la evaluación experimentada por el paciente. Las probabilidades con las cuales cada paciente se mueve de un estado a otro se resumen en la tabla que sigue:

Mi variable aleatoria es: $x(t) = {$Estado critico en el que el paciente es clasificado$}$

Los estados: $M = {"Critico", "Serio","Estable"}$ Estados Críticos

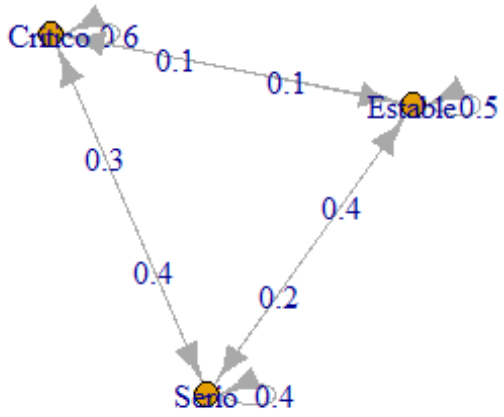
Temporalidad: Por día

a) ¿Cuál es la probabilidad que un paciente en estado crítico un día Jueves esté estable el día Sábado?

```
statesNames = c("Critico", "Serio","Estable")
mc_p10= new ("markovchain", transitionMatrix = matrix
(c(0.6,0.3,0.1,0.4,0.4,0.2,0.1,0.4,0.5), byrow = TRUE , nrow = 3, dimname
s = list(statesNames, statesNames)))
mc_p10
```

```
## Unnamed Markov chain
##  A  3 - dimensional discrete Markov Chain defined by the following sta
tes:
##  Critico, Serio, Estable
##  The transition matrix  (by rows)  is defined as follows:
##         Critico Serio Estable
## Critico     0.6   0.3     0.1
## Serio       0.4   0.4     0.2
## Estable     0.1   0.4     0.5
```

```
plot(mc_p10)
```




```
estado = c(1, 0, 0)
print(estado*(mc_p10^2))
```

```
##      Critico Serio Estable
## [1,]    0.49  0.34    0.17
```

La probabilidad que un paciente en estado crítico un día Jueves esté estable el día Sábado es de 17%

b) ¿Cuál es la probabilidad que un paciente que está en estado estable el Lunes experimente alguna complicación y no esté estable nuevamente el Miércoles?

```
estadoCom = c(0, 0, 1)
print(estadoCom*(mc_p10^2))
```

```
##      Critico Serio Estable
## [1,]    0.27  0.39    0.34
```

La probabilidad que un paciente que está en estado estable el Lunes experimente alguna complicación y no esté estable nuevamente el Miércoles es de 66%, que resulta de sumar la probabilidad del estado critico y el estado serio

c) ¿Qué porcentaje de la Unidad de Cuidados Intensivos usted diseñaría y equiparía para pacientes en estado crítico?

```
esta = c(1, 1, 1)
probEstadoEstable = steadyStates(mc_p10)
print(esta*probEstadoEstable)
```

```
##        Critico     Serio   Estable
## [1,] 0.4150943 0.3584906 0.2264151
```

El porcentaje de la Unidad de Cuidados Intensivos para pacientes en estado crítico es de 41.5%

Ejercicio 11

los estados Una empresa está considerando utilizar Cadenas de Markov para analizar los cambios en las preferencias de los usuarios por tres marcas distintas de un determinado producto. El estudio ha arrojado la siguiente estimación de la matriz de probabilidades de cambiarse de una marca a otra cada mes:

a) Si en la actualidad la participación de mercado de cada marca es de 45%, 25% y 30%, respectivamente.

¿Cuáles serán las participaciones de mercado de cada marca en dos meses más?

$\left [\begin{array} 0.7,0.2,0.1\\0.05,0.93,0.02\\0.4,0.08,0.52\end{array}\right]$

$X(t) =$ La marca

Particion = $\left [\begin{array} 0.45\\0.25\\0.3\end{array}\right]$ tiempo meses

Para saber como son las particiones del mercado en un plazo de 2 meses elevaremos la matriz de Markov al cuadrado y multiplicaremos por la particion que tiene el mercado

```
statesNames=c("Marca 1", "Marca 2","Marca 3")
mrc<- new ("markovchain", transitionMatrix= matrix(c(0.7,0.2,0.1,0.05,0.9
3,0.02,0.4,0.08,0.52),
byrow=TRUE,nrow=3,dimnames=list(statesNames,statesNames)))
mercado <-c(0.45,0.25,0.30)
Partmer<-(mercado*mrc^2)
print(Partmer)
```

```
##       Marca 1  Marca 2 Marca 3
## [1,] 0.412975 0.428225  0.1588
```

b) Cuál es la cuota de mercado en el largo plazo para cada una de las marcas?
Calculamos las probabilidades a largo plazo

```
ProbLargoPlazo<-steadyStates(mrc)
print(ProbLargoPlazo)
```

```
##        Marca 1  Marca 2    Marca 3
## [1,] 0.2176871 0.707483 0.07482993
```

Ejercicio 12.

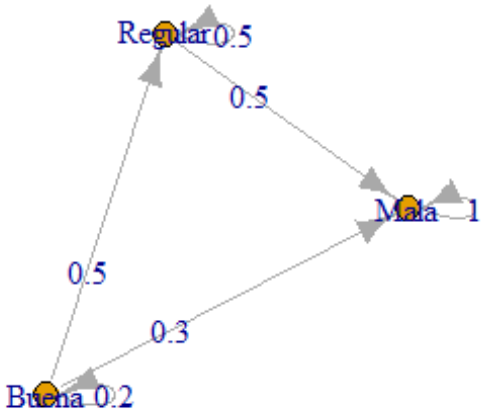
Cada año, durante la temporada de siembra de marzo a septiembre, un jardinero realiza una prueba química para verificar la condición de la tierra. Segúnel resultado de la prueba, la productividad en la nueva temporada puede ser uno de tres estados: (1)buena, (2)regular, (3)mala. A lo largo de los años, el jardinero ha observado que la condición de la tierra del año anterior afecta la probavilidad del año actual y que la situación se describe mediante la siguiente cadena de Markok:

Mi variable aleatoria es: $X(T)$ =La ubicación donde se enccuentra un auto

Los estados: $M = {"Phoenix","Denver","Chicago","Atlanta"}$ 4 estados/ciudades.

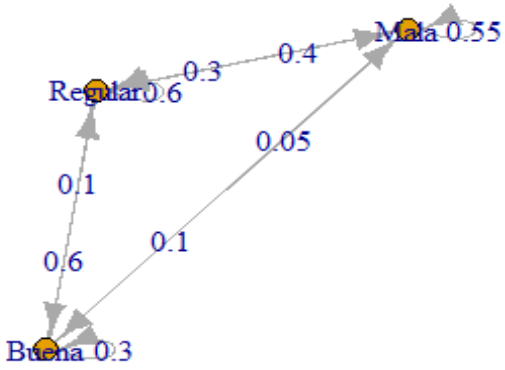
Temporalidad: Por año $\left[\begin{array}0
0,0.2,0.6,0.2\\0,0,0.6,0.4\\0,0.5,0,0.5\\0.1,0.1,0.8,0\end{array}\right]$

```
statesNames = c("Buena","Regular","Mala")
p_p12 <- new ("markovchain", transitionMatrix = matrix
(c(0.2,0.5,0.3,0,0.5,0.5,0,0,1), byrow = TRUE , nrow = 3, dimnames = list
(statesNames, statesNames)))
plot(p_p12)
```




```
print(summary(p_p12))

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## Mala
```

```
## Recurrent classes:
## {Mala}
## Transient classes:
## {Buena},{Regular}
## The Markov chain is not irreducible
## The absorbing states are: Mala
## $closedClasses
## $closedClasses[[1]]
## [1] "Mala"
##
##
## $recurrentClasses
## $recurrentClasses[[1]]
## [1] "Mala"
##
##
## $transientClasses
## $transientClasses[[1]]
## [1] "Buena"
##
## $transientClasses[[2]]
## [1] "Regular"

statesNames = c("Buena","Regular","Mala")
p1_p12 <- new ("markovchain", transitionMatrix = matrix(c(0.3,0.6,0.1,0.1
,0.6,0.3,0.05,0.4,0.55), byrow = TRUE , nrow = 3, dimnames = list(statesN
ames, statesNames)))

plot(p1_p12)
```



```
print(summary(p1_p12))
```

```
## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## Buena Regular Mala
## Recurrent classes:
## {Buena,Regular,Mala}
## Transient classes:
## NONE
## The Markov chain is irreducible
## The absorbing states are: NONE
## $closedClasses
## $closedClasses[[1]]
## [1] "Buena"   "Regular" "Mala"
##
##
## $recurrentClasses
## $recurrentClasses[[1]]
## [1] "Buena"   "Regular" "Mala"
##
##
## $transientClasses
## list()
```

a)¿Cúales son los tiempos esperados de recurrencia de cada estado?

```
#para la matriz P
print(meanRecurrenceTime(p_p12))
```

```
## Mala
##    1
```

```
#para la matriz P1
print(meanRecurrenceTime(p1_p12))
```

```
##    Buena  Regular     Mala
## 9.833333 1.903226 2.681818
```

b) Un jardín necesita dos sacos de fetilizante si la tierra es buena. La cantidad se incrementa en 25% si la tierra es regular, y 60% si la tierra es mala. El costo del fertilizante es de $50 por saco. El jardinero estima un rendimiento anual de $250 si no se utiliza fertilizante,

```
costos<- 50*c(1,1.25,1.6)
rendimientoAnualSF <- 250 #sin fertilizante
rendimientoAnualCF <- 420 #sin fertilizante
#print(sum(costos*steadyStates(p1_p12)))

print(rendimientoAnualCF - sum(costos*steadyStates(p1_p12)))
```

```
## [1] 352.2458
```

Es redituable utilizar fertilizante porque el rendimiento anual seria de $352 que es mayor a $250 sin utilizar fertilizante.

c)Considere la matrix de transicion del jardinero con fertilizantes, ccalcule el tiempo esperado de primera pasada desde los estados 2 y 3 (regular y mala) al estado 1 (bueno).

```
print(meanFirstPassageTime(p1_p12))
```

```
##            Buena  Regular     Mala
## Buena    0.00000 1.774194 4.545455
## Regular 12.50000 0.000000 3.636364
## Mala    13.33333 2.419355 0.000000
```

El tiempo esperado desde el estado Regular a buena es de 12.5 años y de Mala a Regular es de 13.3 años.

d) Considere la matriz de transicion del jardinero sin fertilizantes, calcule la probabilidad de absorcion al estado 3 (condición de tierra mala).

```
print(absorptionProbabilities(p_p12))
```

```
##         Mala
## Buena      1
## Regular    1
```

Ejercicio 13.

Considere una tienda departamental que clasifica el saldo de la cuenta de un cliente como pagada(estado 0), 1 a 30 días de retraso (estado 1), 31 a 60 días de retraso(estado 2), o mala deuda(estado 3). Las cuentas se revisan cada mes y se determina el estado de cada cliente. En general, los créditos no se extienden y se espera que los deudores paguen sus cuentas lo más pronto posible. En ocasiones, los clientes no pagan en la fecha límite. Si esto ocurre cuando els aldo queda dentro de los 30 días de retraso, la tienda considera que este cliente permanece en el estado 1. Si esto ocurre cuando el saldo est+a entre 31 y 60 días de retraso, la tienda considera que el cliente se mueve al estado 2. Los clientes que tienen más de 60 días de retraso se clasifican en la categoría de una mala deuda(estado 3), en cuyo caso envía las cuentas a una agencia de cobro.

Despues de examinar los datos de años anteriores en la progresión mes a mes de los clientes individuales de estado a estado, la tienda ha desarrollado la siguiente matriz de transición:

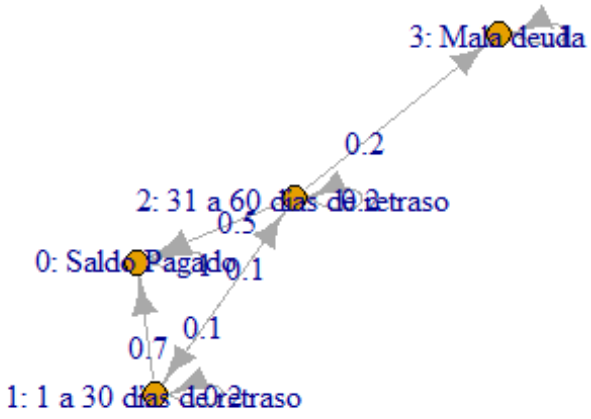
$$\left[\begin{array} _1, 0, 0, 0\\ 0.7, 0.2, 0.1, 0\\ 0.5, 0.1, 0.2, 0.2\\0, 0, 0, 1\end{array}\right]$$

a) Construya la matriz de transición de estados.

```
statesNames = c("0: Saldo Pagado", "1: 1 a 30 días de retraso", "2: 31 a
60 días de retraso", "3: Mala deuda")

mc_p13deudas <- new("markovchain", transitionMatrix = matrix(c(1, 0, 0, 0
, 0.7, 0.2, 0.1, 0, 0.5, 0.1, 0.2, 0.2, 0, 0, 0, 1), byrow = TRUE, nrow =
4, dimnames = list(statesNames,statesNames)))

summary(mc_p13deudas)

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## 0: Saldo Pagado
## 3: Mala deuda
## Recurrent classes:
## {0: Saldo Pagado},{3: Mala deuda}
## Transient classes:
## {1: 1 a 30 días de retraso,2: 31 a 60 días de retraso}
## The Markov chain is not irreducible
## The absorbing states are: 0: Saldo Pagado 3: Mala deuda

print(mc_p13deudas)

##                            0: Saldo Pagado 1: 1 a 30 días de retraso
## 0: Saldo Pagado                        1.0                       0.0
## 1: 1 a 30 días de retraso              0.7                       0.2
## 2: 31 a 60 días de retraso             0.5                       0.1
## 3: Mala deuda                          0.0                       0.0
##                            2: 31 a 60 días de retraso 3: Mala deuda
## 0: Saldo Pagado                                   0.0           0.0
## 1: 1 a 30 días de retraso                         0.1           0.0
## 2: 31 a 60 días de retraso                        0.2           0.2
## 3: Mala deuda                                     0.0           1.0
```

B) Dibuje el diagrama de transición de estados.

```
plot(mc_p13deudas)
```




```
absorbingStates(mc_p13deudas)

## [1] "0: Saldo Pagado" "3: Mala deuda"
```

C) Aunque cada cliente acaba por llegar al estado 0 o al estado 3, la tienda se interesa en determinar la probabilidad de que un cliente llegue a ser un mal deudor dado que la cuenta pertenece al estado de 1 a 30 días de retraso, y de igual forma, si se encuentra en 31 a 60 días de retraso.

Nuestro estado "1 a 30 dias" es un estado trancitorio nos queda los siguiete:

```
print(absorptionProbabilities(mc_p13deudas))

##                             0: Saldo Pagado 3: Mala deuda
## 1: 1 a 30 días de retraso         0.9682540    0.03174603
## 2: 31 a 60 días de retraso        0.7460317    0.25396825
```

Ejercicio 14.

Un proceso de producción incluye una máquina que se deteriora con rapidez tanto en la calidad como en la cantidad de producción con el trabajo pesado, por lo que se inspecciona al final de cada día. Después de la inspección se clásica la condición de la máquina en uno de cuatro estados posibles

Estado Condicion 0 Tan buena como nueva 1 Operable: deteriodo minimo 2 Operable: deteriodo mayor 3 Inoperable y Remplazable por una tan buena como nueva

El proceso se puede modelar como una cadena de Markov con matriz de transición (de un paso) P dada por: $\left [\begin{array}
0,0.2,7/8/16,1/16\\0,3/4,1/8,1/8\\0,0,1/2,1/2\\1,0,0,0\end{array}\right]$
$X(t)={$Estados de la maquina$}$ tiempo dias

a) Encuentre las probabilidades de estado estable.

```
statesNames=c("0", "1","2","3")
mc_p14<- new ("markovchain", transitionMatrix= matrix(c(0,7/8,1/16,1/16,0
,3/4,1/8,1/8,0,0,1/2,1/2,1,0,0,0),byrow=TRUE,nrow=4,dimnames=list(statesN
ames,statesNames)))

ProbLargoPlazo<-steadyStates(mc_p14)
print(ProbLargoPlazo)
```

```
##              0         1         2         3
## [1,] 0.1538462 0.5384615 0.1538462 0.1538462
```

b) Si los costos respectivos por estar en los estados 0, 1, 2, 3 son 0, 1 000, 3 000 y 6 000 dólares, ¿cuál es el costo diario esperado a largo plazo?

```
#Multiplicamos el estado estable por los costos y sumamos
costos <-c(0,1000,3000,6000)
print(sum(costos*ProbLargoPlazo))
```

```
## [1] 1923.077
```

```
#CT es de 1923.077
```

c) Encuentre el tiempo de recurrencia esperado del estado 0 (esto es, el tiempo esperado que una máquina se puede usar antes de tener que reemplazarla).

```
meanRecurrenceTime(mc_p14)
```

```
##        0        1        2        3
## 6.500000 1.857143 6.500000 6.500000
```

La maquina puede ser utilizada 6 dias y medio

Ejercicio 15.

Una unidad importante consta de dos componentes colocadas en paralelo. La unidad tiene un desempeño satisfactorio si una de las dos componentes está en operación. Por lo tanto, sólo se opera una de ellas a la vez, pero ambas se mantienen operativas (capaces de operar) tanto como sea posible, reparándolas cuando se necesite. Un componente operativo tiene una probabilidad de 0.2 de descomponerse en un periodo dado. Cuando ocurre,el componente en paralelo opera, si está operativo, al comenzar el siguiente periodo. Sólo se puede reparar un componente a la vez. Una reparación se inicia al principio del primer periodo disponible y termina al final del siguiente. Sea Xt un vector con dos elementos U y V , donde U es el número de componentes operativos al final del periodo t y V el número de periodos de reparación que transcurren para

componentes que todavía no son operativos. Entonces, V = 0 si U = 2 o si U = 1 y la reparación del componente no operativo se está realizando. Como la reparación toma dos periodos, V = 1 si U = 0 (pues el componente no operativo espera iniciar su reparación mientras la otra entra al segundo periodo) o si U = 1 y el componente no operativo está en su segundo periodo. Así, el espacio de estados contiene cuatro estados (2, 0), (1, 0), (0, 1) y (1, 1). Denote estos estados por 0, 1, 2, 3, respectivamente. {Xt}(t = 0, 1, . . .) es una cadena de Markov (suponga que X0 = 0)con matriz de transición (de un paso)

Mi variable aleatoria es: $x(t) = {$Componentes en operación$}$

Los estados: $M = {"0", "1","2","3"}$ Desempeño

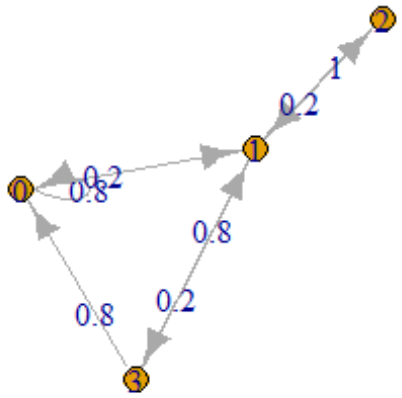
Temporalidad: Por periodo

Matriz de transición (a 1 paso):

$$\left[\begin{array}
00.8,0.2,0,0\\0,0,0.2,0.8\\0,1,0,0\\0.8,0.2,0,0\end{array}\right]$$

a) ¿Cuál es la probabilidad de que la unidad no esté operable después de n periodos (porque ambas componentes estén descompuestas), para n = 2, 5, 10, 20?

```
statesNames = c("0", "1","2","3")
mc_p15= new ("markovchain", transitionMatrix = matrix
(c(0.8,0.2,0,0,0,0,0.2,0.8,0,1,0,0,0.8,0.2,0,0), byrow = TRUE , nrow = 4,
dimnames = list(statesNames, statesNames)))
plot(mc_p15)
```




```
print(summary(mc_p15))

## Unnamed Markov chain  Markov chain that is composed by:
## Closed classes:
## 0 1 2 3
## Recurrent classes:
## {0,1,2,3}
## Transient classes:
## NONE
## The Markov chain is irreducible
## The absorbing states are: NONE
## $closedClasses
## $closedClasses[[1]]
## [1] "0" "1" "2" "3"
##
##
## $recurrentClasses
## $recurrentClasses[[1]]
## [1] "0" "1" "2" "3"
##
##
## $transientClasses
## list()

print(mc_p15^2)
```

```
##      0    1    2    3
## 0 0.64 0.16 0.04 0.16
## 1 0.64 0.36 0.00 0.00
## 2 0.00 0.00 0.20 0.80
## 3 0.64 0.16 0.04 0.16

print(mc_p15^5)

##         0       1       2       3
## 0 0.61952 0.19488 0.03712 0.14848
## 1 0.59392 0.17408 0.04640 0.18560
## 2 0.64000 0.23200 0.02560 0.10240
## 3 0.61952 0.19488 0.03712 0.14848

print(mc_p15^10)

##           0         1          2         3
## 0 0.6152913 0.1922044 0.03850086 0.1540035
## 1 0.6160138 0.1929815 0.03820093 0.1528037
## 2 0.6141051 0.1910047 0.03897805 0.1559122
## 3 0.6152913 0.1922044 0.03850086 0.1540035

print(mc_p15^20)

##           0         1          2         3
## 0 0.6153845 0.1923076 0.03846159 0.1538464
## 1 0.6153854 0.1923085 0.03846121 0.1538448
## 2 0.6153831 0.1923061 0.03846218 0.1538487
## 3 0.6153845 0.1923076 0.03846159 0.1538464
```

La probabilidad de que la unidad no esté operable después de n periodos es 4%, 3.7%, 3.9%, y 3.08 respectivamente

b) ¿Cuáles son las probabilidades de estado estable del estado de esta cadena de Markov?

```
desem = c(0, 1, 0,0)
print(desem*(mc_p15^20))

##              0         1          2         3
## [1,] 0.6153854 0.1923085 0.03846121 0.1538448
```

Las probabilidades de estado estable del estado de esta cadena de Markov son 61.53%, 19.23%, 3.84%, y 15.38%

c) Si cuesta 30 000 dólares por periodo que la unidad no opere (ambas componentes descompuestas) y cero en otro caso, ¿cuál es el costo promedio esperado (a la larga) por periodo?

```
p= 30000
costopromedio= p*0.038
print(costopromedio)
```

```
## [1] 1140
```

El costo promedio esperado por periodo es de 1140 dólares