

# Cadenas de Markov

Matemáticas II

Diego Valero Bueno  
INGENIERÍA MULTIMEDIA

# Contenido

---

Introducción.....	2
Creador de las cadenas de Markov.....	2
Cuerpo.....	2
Desarrollo de las cadenas de Markov .....	2
Notación útil.....	2
Tipos de cadenas de Markov .....	4
Implementación de las cadenas de Markov en RStudio .....	5
Convergencia de las cadenas de Markov en RStudio .....	6
Ejemplo de cadena de Markov en RStudio .....	7
Ejemplo de la ruina del jugador .....	7
Movimiento de poblaciones .....	8
Aplicaciones .....	10
Meteorología .....	10
Internet .....	10
Juegos de azar.....	10
Economía y fianzas.....	10
Genética.....	10
Otros .....	10
Bibliografía .....	11

## Introducción

---

La cadena de Markov es un proceso estocástico donde la probabilidad de que se dé un evento depende solamente del estado anterior. A esta característica de falta de memoria se le denomina propiedad de Markov. No proporcionan un resultado exacto pero sí muy preciso a largo plazo.

Tienen un rango de uso muy amplio: meteorología, economía y finanzas, genética, estadística, matemáticas, redes neuronales, etc.

### Creador de las cadenas de Markov

Las cadenas de Markov fueron creadas en 1907 por Andréi Markov. Andréi fue un matemático ruso que hizo importantes aportaciones a la teoría de números y a la teoría de probabilidades. Fue alumno en la universidad de San Petersburgo en la facultad de Física y Matemáticas. Sus primeros trabajos fueron análisis, límites integrales y teoría de la aproximación y convergencia de series.



## Cuerpo

---

### Desarrollo de las cadenas de Markov

Antes de profundizar en las cadenas de Markov, vamos a pasar a explicar la notación y los elementos útiles de estas.

#### Notación útil

##### *Cadenas homogéneas y no homogéneas*

Las cadenas de Markov son homogéneas cuando se cumple la propiedad de que la probabilidad de que  $X_n = j$  no dependa del tiempo que se considere, es decir, las probabilidades de que ocurra un evento son las mismas en cada paso. Se explica en la siguiente igualdad:

$$P(X_n = j | X_{n-1} = i) = P(X_1 = j | X_0 = i) \text{ para todo } n \text{ y para cualquier } i, j.$$

##### *Matriz de probabilidad*

La matriz de probabilidad, también llamada matriz de transición, es usada para describir las transiciones en una cadena de Markov. Es también una matriz estocástica, la cual es una matriz cuadrada cuya suma de las filas o de las columnas suman 1.

Es obvio que la matriz de probabilidad tiene que ser cuadrada ya que tiene que tener un número de filas y columnas igual al número de estados posibles para que en el elemento  $P_{ij}$  defina la probabilidad de que se pase del estado  $i$  al estado  $j$ .

Si  $P$  es la matriz de transición de una cadena de Markov y  $P^k$  la matriz de transición  $P$  multiplicada por sí misma  $k$  veces, el elemento  $P^k_{ij}$  define la probabilidad de pasar del estado  $i$  al estado  $j$  en  $k$  transiciones.

Poniendo como ejemplo dos estados  $A$  y  $B$ , el 20% van de  $A$  a  $B$  y el 80% se quedan en  $A$ , y de  $B$  a  $A$  van 70% y el 30% se quedan. La matriz de transición de este ejemplo sería la siguiente.

$$\begin{array}{cc} & \begin{matrix} [1,] & [2,] \end{matrix} \\ \begin{matrix} [1,] \\ [2,] \end{matrix} & \begin{bmatrix} 0.8 & 0.7 \\ 0.2 & 0.3 \end{bmatrix} \end{array}$$

Como podemos ver, es una matriz estocástica porque la suma de sus columnas es 1.

### Vector de probabilidad invariante

Es un vector que cumple la siguiente igualdad:

$\nu P = \nu$  Donde  $P$  es la matriz de transición de la cadena de Markov. Cumple que al multiplicarlo por la matriz de transición, no varía. También se le denomina como vector de distribución estacionaria o distribución de equilibrio.

### Recurrencia

Teniendo  $E$  como espacio de estados de una cadena de Markov, si  $x$  pertenece a  $E$ , se define:

$$L_x = \mathbb{P}(X_n = x \text{ para algun } n \in \mathbb{N} | X_0 = x)$$

Y diremos que  $x$  es estado recurrente si  $L_x=1$ , transitorio si  $L_x<1$  o absorbente si  $p_{x,x}=1$

Sea  $\mu_x = \mathbb{E}(T_x | X_0 = x)$  si  $x$  pertenece a  $E$  diremos que:

$x$  es cero-recurrente si  $\mu_x=1$  o como positivo-recurrente si  $\mu_x<1$ .

### Periodicidad

El periodo de un estado  $x$  se define como:  $d(x) = \text{mcd}\{n : P_{x,x}^{(n)} > 0\}$

Si  $d(x)=1$ , diremos que el estado es aperiódico. Si todos los estados son aperiódicos, la cadena de Markov se denomina aperiódica.

## Tipos de cadenas de Markov

### *Cadenas irreducibles*

Se dice que una cadena de Markov es irreducible cuando cumple alguna de las siguientes premisas (todas son equivalentes entre sí):

1. Desde cualquier estado se puede acceder a otro.
2. Todos los estados están comunicados entre sí.
3.  $C(x)=E$  para algún  $x$  perteneciente a  $E$ .
4.  $C(x)=E$  para todo  $x$  perteneciente a  $E$ .
5. El único conjunto cerrado es el conformado por todos los estados.

Como ejemplos de cadenas de Markov irreducibles tenemos la cadena de Ehrenfest, que es una cadena de Markov en tiempo discreto que sirve para modelar el intercambio de moléculas de gas entre dos urnas.

### *Cadenas positivo-recurrentes*

Se da si todos sus estados son positivos-recurrentes. Además, su vector de estado invariable se define:  $\pi_x = 1/\mu_x$

### *Cadenas regulares*

Se dice que una cadena de Markov es regular cuando existe alguna potencia positiva de la matriz de transición en la que sus entradas son todas estrictamente mayores que 0. Se da que:

$\lim_{n \rightarrow \infty} P^n = W$  Donde  $W$  es un vector de probabilidad que resulta ser el vector de probabilidad invariante de la cadena. En el caso de que la cadena de Markov sea regular, este vector invariante  $W$  será único.

### *Cadenas absorbentes*

Se dice que una cadena de Markov es absorbente si cumple las siguientes condiciones:

1. Tiene al menos un estado absorbente.
2. De cualquier estado que no es absorbente se puede acceder a un estado absorbente.

Si llamamos  $A$  a todos los estados absorbentes de la cadena y  $D$  a los que no lo son, tenemos que su matriz de transición es la siguiente:

$$P = \begin{pmatrix} Q & R \\ 0 & I \end{pmatrix}$$

$Q$  es una submatriz correspondiente a los estados de  $D$ ,  $I$  es la identidad,  $0$  la matriz nula y  $R$  una submatriz cualquiera.

## Cadenas de Markov en tiempo continuo

Si en vez de considerar  $x_1, x_2, x_3, \dots, x_i$  con  $i$  dentro del conjunto de los números naturales, consideramos  $X_t$  como variable aleatoria que varía en un intervalo continuo del conjunto de los números reales, tendremos una cadena de Markov de tiempo continuo.

$$P(X(t_{n+1}) = x_{n+1} | X(t_n) = x_n, \dots, X(t_1) = x_1) = P(X(t_{n+1}) = x_{n+1} | X(t_n) = x_n) \text{ tal que } t_{n+1} > t_n > t_{n-1} > \dots > t_1$$

De aquí deducimos que la probabilidad de que se dé el estado  $x_{n+1}$  para  $t_{n+1}$  solo depende del estado anterior  $x_n$  para  $t_n$ .

## Implementación de las cadenas de Markov en RStudio

Para implementar las cadenas de Markov en RStudio solo necesitamos saber cómo crear matrices y como multiplicarlas.

Para crear las matrices, usaremos el siguiente comando: `> x <- matrix(c(0.8,0.2,0.3,0.7),2,2)`  
Vemos que en el primer parámetro de la función `matrix()` usada para crear matrices se introduce un vector con los valores que obtendrá la matriz, luego el número de filas y, por último, el número de columnas. La matriz que nos crea esta función es la siguiente:

Para multiplicar la siguiente matriz por otra, tendremos que usar el siguiente operador: `%*%`; y se usa de la siguiente forma:

```
> x %*% y
```

	<code>[,1]</code>	<code>[,2]</code>
<code>[1,]</code>	2.2	1.8
<code>[2,]</code>	1.8	2.2

He declarado la matriz y como una matriz de 2x2 con 2 en todas sus posiciones. Haciendo uso de este operador, ha multiplicado x por y con el producto matricial y nos ha mostrado por pantalla el resultado de la operación.

Sabiendo esto, podemos crear la matriz de transición y la matriz del estado inicial de la cadena de Markov. Teniendo estas dos matrices, cada iteración de la cadena será el producto matricial de la matriz de transición con la matriz de estado inicial. Estas iteraciones se pueden hacer manualmente o mediante un bucle creado en un script.

```
iter <- function(MTrans,estado,niter){
  for(i in 1:niter){
    estado <- MTrans %*% estado
  }
  estado
}
```

Con este sencillo programa podemos hacer una cadena de Markov muy fácilmente: simplemente tenemos que pasar como primer parámetro la matriz de transición, como segundo la matriz de estado inicial y como tercero el número de iteraciones que queramos para terminar la cadena de Markov.

## Convergencia de las cadenas de Markov en RStudio

Como bien sabemos, las cadenas de Markov tienden a converger hacia un vector de estado en el que por más iteraciones que hagamos, no va a variar. Hay varias formas de ver el vector de estado final: iterando muchas veces hasta encontrar uno que no varíe, usando la norma de los vectores propios de la matriz de transición y, por último, iterando hasta que la diferencia entre un elemento y ese mismo elemento pero del estado anterior sea prácticamente despreciable.

### *Norma de los vectores propios de la matriz de transición*

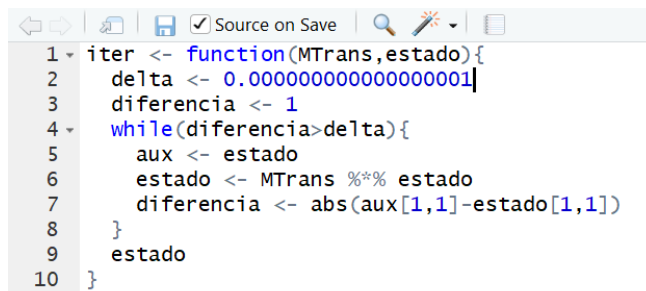
Mediante este método se puede saber a qué vector de estado final va a converger la matriz de transición pero, como podemos ver, no estamos teniendo en cuenta el vector de estado inicial. Ese es uno de los principales problemas del método de la norma, que no podemos asignar a la cadena de Markov un vector de estado inicial. Pero si para nuestro problema podemos prescindir del vector de estado inicial, el método de la norma es uno de los más rápidos ya que nos ahorramos mucho tiempo de ejecución al no iterar.

```
25 X <- eigen(P)
26 vectores <- abs(X$vectors[,1])
27 norma <- norm(as.matrix(vectores))
28 x1 <- vectores/norma
```

La función `eigen(P)` nos devuelve los valores y vectores propios de la matriz que se le pasa por parámetros. De ahí, obtenemos los valores propios de la primera columna del apartado de los vectores propios. Por último, sacamos la norma de la matriz formada por dicho vector propio y dividiremos ese vector propio entre la norma. Esto nos da como resultado el vector hacia el cual la matriz de transición ha convergido.

### *Diferencia de estados*

Este método se basa en declarar un valor prácticamente despreciable y crear un bucle que repita una serie de operaciones mientras que el valor absoluto de la resta entre el elemento de la primera fila y la primera columna de la matriz de estado actual y el elemento de la primera fila y la primera columna de la matriz de estado anterior sea mayor que el valor previamente establecido. Las operaciones que realiza el bucle son: guardar en una variable auxiliar la matriz del estado anterior, el producto matricial de la matriz de transición con la matriz del estado anterior y guardarnos el valor de la resta. Por último, imprimir la matriz de estado final por pantalla.



```
1 iter <- function(MTrans,estado){
2   delta <- 0.0000000000000001
3   diferencia <- 1
4   while(diferencia>delta){
5     aux <- estado
6     estado <- MTrans %%% estado
7     diferencia <- abs(aux[1,1]-estado[1,1])
8   }
9   estado
10 }
```

A esta función se le pasará por parámetros la matriz de transición en primer lugar y la matriz de estado inicial en segundo lugar. A diferencia del ejemplo anterior, en este no se introduce el número de iteraciones porque es un valor desconocido, ya que el bucle va a iterar tantas veces como sea necesario hasta que la diferencia de elementos sea menor o igual que delta.

Al contrario que el método de la norma, aquí sí que tenemos en cuenta el estado inicial de la cadena. Por tanto, este método es más fiable que el anterior. Para profundizar un poco más, dentro del bucle se podría poner un contador que vaya sumando valores cada vez que se ejecute el bucle para ver cuántas iteraciones ha necesitado la cadena de Markov para converger. Sería tan sencillo como inicializar al principio del script una variable llamada “contador” y asignarle el valor 0. Una vez dentro del bucle, al final, pondríamos que el valor de la variable “contador” pasa a ser el valor que tenía anteriormente más uno.

## Ejemplo de cadena de Markov en RStudio

Vamos a explicar dos ejemplos de resolución de problemas con cadenas de Markov: la ruina del jugador y el cambio de poblaciones.

### Ejemplo de la ruina del jugador

El problema de la ruina del jugador se nos plantea como una persona que tiene 20€ y necesita 50€ pero solo puede obtenerlos jugando en el casino. Solo puede apostar de 10 en 10 euros y hay 50% de probabilidades de ganar 10€ más o de perder los apostados. La pregunta es: cuál es la probabilidad de llegar a esos 50€ teniendo 20 al principio.

La matriz de transición de este problema es bastante sencilla: si tienes 0€, te quedas ahí siempre, si tienes 10€, tienes las mismas probabilidades de tener 20 que 0, y así para todos los casos. La matriz que nos queda es la siguiente:

$$\begin{pmatrix} 1 & .5 & 0 & 0 & 0 & 0 \\ 0 & 0 & .5 & 0 & 0 & 0 \\ 0 & .5 & 0 & .5 & 0 & 0 \\ 0 & 0 & .5 & 0 & .5 & 0 \\ 0 & 0 & 0 & .5 & 0 & 0 \\ 0 & 0 & 0 & 0 & .5 & 1 \end{pmatrix}$$

Como podemos ver, es una matriz estocástica como deben ser las matrices de transición. También podemos ver que si se tienen 0€, sigues en 0 al igual que si tienes 50.

Por ejemplo, después de haber jugado 1000 veces, la matriz de transición  $P$  nos queda como  $P^{1000}$ . Este cálculo es muy pesado, así que haremos  $P=P^2$  repetidas veces hasta llegar a  $P^{1024}$ .

La matriz final que nos queda después de hacer las operaciones es la siguiente:

$$P^{1024} = \begin{pmatrix} 1 & .8 & .6 & .4 & .2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .2 & .4 & .6 & .8 & 1 \end{pmatrix}$$

Antes de analizar la matriz, hay que comentar que los las columnas representan el estado inicial, es decir, el dinero con el que se empieza: columna 1  $\rightarrow$  0€, columna 2  $\rightarrow$  10€, etc. Las filas representan los estados finales: fila 1  $\rightarrow$  0€, fila 6  $\rightarrow$  50€, etc.

Sabiendo esto, podemos ver que independientemente del dinero con el que se empieza, siempre se va a acabar o con 0€ o con 50€, lo único que cambian son las probabilidades de que esto pase. Analicemos el caso que nos comenta el enunciado: nos decía que el jugador empezaba con 20€, es decir, nos fijaremos en la columna 3. Vemos que, con 20€, el 60% de las veces acabarás con 0€ y el 40% de las veces con 50€. De esta manera podemos ir comprobando qué probabilidades tenemos de acabar en un sitio u otro dependiendo del dinero con el que empezemos.



## Movimiento de poblaciones

Nos introducen el problema como que se ha realizado un estudio sobre los movimientos de población entre una ciudad y los suburbios. Los resultados han sido los siguientes: La migración poblacional entre estas dos partes de la región metropolitana es tal que cada año el 5% de la población de la ciudad se mueve a los suburbios, mientras que el 3% de la población de los suburbios se mueve a la ciudad. Supongamos que en 2014 hay 600 mil personas en la ciudad y 400 mil en los suburbios. Nos hacen las siguientes cuestiones:

- 1.- ¿Cuál será la distribución de la población en 2015?
- 2.- ¿Y en 2018? ¿Y en 2030?
- 3.- ¿Se estabilizarán algún día los movimientos de las poblaciones?

Primero que todo, tenemos que obtener la matriz de transición y el vector de estado inicial de la matriz. Llamaremos P a la matriz de transición y x0 al vector de estado inicial. Haremos la resolución de este problema apoyándonos en un script de RStudio.

Antes de nada, declaramos las matrices de transición y de estado inicial y las mostramos por pantalla.

```
P <- matrix(c(0.95, 0.05, 0.03, 0.97), nrow=2)
print(P)
x0 <- matrix(c(0.6, 0.4), nrow=2)
print(x0)
```

	[,1]	[,2]
[1,]	0.95	0.03
[2,]	0.05	0.97

	[,1]
[1,]	0.6
[2,]	0.4

La primera matriz es la matriz de transición y la segunda la del estado inicial. Como podemos ver, hemos reducido las poblaciones iniciales a probabilidades para simplificar más los resultados y trabajar con números más pequeños en vez de con cientos de miles.

Después de esto, pasaremos a la resolución del primer apartado, la distribución de la población en 2015.

```
x1<-P %*% x0
cat("La distribucion en 2015 sera la siguiente: \n")
print(x1)
```

Con este código, realizamos una iteración simple de la cadena de Markov y mostramos por

```
La distribucion en 2015 sera la siguiente:
      [,1]
[1,] 0.582
[2,] 0.418
```

pantalla el resultado.

Esto es lo que nos muestra el código anterior. La información que nos aporta es que, pasado un año, la probabilidad de vivir en la ciudad será de 0.582 y de vivir en los suburbios un 0.418.

Para la resolución del siguiente apartado, como ya tenemos el resultado de las probabilidades para 2015, solo tenemos que iterar 3 veces más para obtener las de 2018. Una vez tenemos esas, iteramos 12 veces más para tener las de 2030.

```
for(i in 1:3){
  #print(i)
  aux<-x1
  x1<-P %*% aux
}
cat("La distribución para 2018 sera la siguiente: \n")
print(x1)

for(i in 1:12){
  #print(i)
  aux <- x1
  x1 <- P %*% aux
}
cat("La distribución para 2030 sera la siguiente: \n")
print(x1)
```

Con este código se iteran las veces que he dicho y muestra por pantalla los resultados, que son los siguientes:

```
La distribución para 2018 sera la siguiente:
      [,1]
[1,] 0.5361884
[2,] 0.4638116
La distribución para 2030 sera la siguiente:
      [,1]
[1,] 0.4342636
[2,] 0.5657364
```

Nos dice que para 2018, los resultados serán esos y para 2030, los que van a continuación. Para 2030, la población de los suburbios ya empieza a ser mayor que la de la ciudad.

Para el último apartado, podemos usar tanto el método de la norma como el de la diferencia de elementos entre estados, explicados ambos previamente. En este caso, usaremos el método de la norma porque nos da el resultado real y es bastante más corto y sin iteraciones.

```
X <- eigen(P)
vectores <- abs(X$vectors[,1])
norma <- norm(as.matrix(vectores))
x1 <- vectores/norma
cat("Por mas que avancen los años, la distribución tiende hacia: \n")
print(x1)
```

Usamos lo previamente explicado, valores y vectores propios, norma de la matriz conformada por el vector propio de la primera columna y la división de ese vector entre su norma. El resultado nos lo muestra por pantalla y es el siguiente:

```
Por mas que avancen los años, la distribución tiende hacia:
[1] 0.375 0.625
```

Como bien dice, por más que avancen los años, la distribución se estabiliza en un vector invariable, diciendo que hay una probabilidad del 0.375 de gente que vivirá en la ciudad y un 0.625 de gente que vivirá en los suburbios.

# Aplicaciones

---

## Meteorología

Considerando el tiempo atmosférico como días sueltos, podemos asumir que el tiempo que va a hacer en un día depende solamente del estado del tiempo anterior. Gracias a esto, se han hecho modelos meteorológicos básicos para estudiar la recurrencia de las precipitaciones, entre otros.

## Internet

La pagerank de una página web es una cadena de Markov donde la posición que ocupe dentro de la cadena corresponde al peso que tiene en la distribución. Este método es el usado por Google para las búsquedas en internet. Clasifica las páginas web en torno a las interacciones que tiene y las ordena en una cadena de Markov para luego ofrecer las que más tengan.

Este concepto de la matriz de Google es muy extenso y las operaciones que realizan con ella son muy costosas dada la cantidad de datos que almacena esta matriz.

## Juegos de azar

Como en el ejemplo anterior de la ruina del jugador, las cadenas de Markov se usan para modelar juegos de azar. En este caso específico, la cadena de Markov orientada al problema de la ruina del jugador establece la probabilidad de en un juego de azar, se quede sin dinero.

## Economía y fianzas

Usadas en modelos como la bolsa de valores o para determinar la volatilidad de los precios. En comercios, se han usado para analizar los patrones de compra de los deudores morosos, reemplazo de equipo, planear necesidades personales, etc.

## Genética

Se usan en la teoría de poblaciones para describir los cambios de la genética en poblaciones pequeñas sometidas a la deriva genética. Se ha construido un método con las cadenas de Markov para analizar la difusión de los genes llamado Motō Kimura.

## Otros

También han sido usadas en música, operaciones, redes neuronales, etc.

## Bibliografía

---

<http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/PEst/tema4pe.pdf>

[https://es.wikipedia.org/wiki/Cadena\\_de\\_M%C3%A1rkov](https://es.wikipedia.org/wiki/Cadena_de_M%C3%A1rkov)

<https://www.pymesyautonomos.com/estrategia/la-cadena-de-markov-y-su-funcion-empresarial>

<https://www.ugr.es/~eaznar/markov.htm>

[https://es.wikipedia.org/wiki/Matriz\\_estoc%C3%A1stica](https://es.wikipedia.org/wiki/Matriz_estoc%C3%A1stica)

[https://www.cimat.mx/Eventos/taller\\_probabilidad08/cadenas2.pdf](https://www.cimat.mx/Eventos/taller_probabilidad08/cadenas2.pdf)

[http://www.dia.fi.upm.es/~ajimenez/Docu\\_IO/Transparencias/CMTD.pdf](http://www.dia.fi.upm.es/~ajimenez/Docu_IO/Transparencias/CMTD.pdf)

Libro teórico de Matemáticas II de Ingeniería Multimedia

Apuntes de práctica de RStudio sobre el álgebra lineal.