Matemáticas I. Parte númerica del examen

Cristóbal Lecaros C.

Ejercicio 1

Ejercicio 1.1

Calcule los valores propios de de $\mathbf{A}$ y $\mathbf{A}^2$

Respuesta: se define $\mathbf{A}$, luego se computa con las funciones `%*%` y `eigen()`.

```
options(digits = 3)
A <- matrix(c(0.6,0.2,0.4,0.8), byrow = T, ncol = 2); A
```

```
##      [,1] [,2]
## [1,]  0.6  0.2
## [2,]  0.4  0.8
```

```
eigen(A)
```

```
## eigen() decomposition
## $values
## [1] 1.0 0.4
##
## $vectors
##        [,1]   [,2]
## [1,] -0.447 -0.707
## [2,] -0.894  0.707
```

```
A_A <- A %*% A; A_A
```

```
##      [,1] [,2]
## [1,] 0.44 0.28
## [2,] 0.56 0.72
```

```
eigen(A_A)
```

```
## eigen() decomposition
## $values
## [1] 1.00 0.16
##
## $vectors
##        [,1]   [,2]
## [1,] -0.447 -0.707
## [2,] -0.894  0.707
```

Nota. En los cálculos que siguen, es interesante notar algo. Incluso cuando uno define un decimal como 0.6, este valor se encuentra aproximado y almacenado de manera diferente en la memoria de R. esto puede verse explicitado de la siguiente manera.

```
options(digits = 22)
aproximacion <- 0.6
aproximacion
```

```
## [1] 0.5999999999999999777955
```

Ejercicio 1.2

¿Puede calcular los valores y vectores propios de $\mathbf{A}^{100}$?
Respuesta: Se define una función, y se computa la potencia.

```
options(digits = 4)
Mpow <- function(A, n) {
   if (n==1) return(list(A))
   L <- list(A)
   P <- A
   for (i in 2:n){
      P <- P %*% A
      L[[i]] <- P
   }
   return(L)
}

lista_A_100 <- Mpow(A, 100)
```

La matriz $\mathbf{A}^{100}$ es:

```
lista_A_100[[100]]
```

```
##        [,1]   [,2]
## [1,] 0.3333 0.3333
## [2,] 0.6667 0.6667
```

Sus valores y vectores propios:

```
eigen(lista_A_100[[100]])
```

```
## eigen() decomposition
## $values
## [1]  1.00e+00 -1.11e-16
##
## $vectors
##         [,1]    [,2]
## [1,] -0.4472 -0.7071
## [2,] -0.8944  0.7071
```

Ejercicio 1.3

¿Qué pasa cuando aplico sucesivas veces la matriz $\mathbf{A}$ a un vector P, que inicialmente tiene valores $P_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

Respuesta: Para esto definimos P, y luego aplicamos `lapply()` para ciclar sobre la lista.

```
P <- matrix(c(1,0), ncol = 1); P
```

```
##      [,1]
## [1,]    1
## [2,]    0
```

```
modificacion_P <- lapply(lista_A_100, '%*%', P)
```

Rescatamos el último valor y vemos que:

```
modificacion_P[100]
```

```
## [[1]]
##        [,1]
## [1,] 0.3333
## [2,] 0.6667
```

Se obtiene el vector propio asociado al valor propio 1, es decir $P = 1/3 \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

Ejercicio 2

Ejercicio 2.1

¿Cual es la proporción de personal en cada piso despues de un año? *Respuesta*: Para desarrollar este ejercicio, ocuparé el paquete `markovchain` para hacer los calculos y el paquete `diagram` para graficar. Lo primero es definir la matriz de transición.

```
library(markovchain)
```

```
## Package:   markovchain
## Version:   0.6.9.8-1
## Date:      2017-08-15
## BugReport: http://github.com/spedygiorgio/markovchain/issues
```

```
library(diagram)
```

```
## Loading required package: shape
```

```
matriz_cambio <- matrix(c(.8,.1,.1,.1,.8,.1,.1,.1,.8), byrow = T, nrow = 3)
matriz_cambio
```

```
##      [,1] [,2] [,3]
## [1,]  0.8  0.1  0.1
## [2,]  0.1  0.8  0.1
## [3,]  0.1  0.1  0.8
```

Con esta matriz, se define el objeto `markov` para realizar los cálculos.

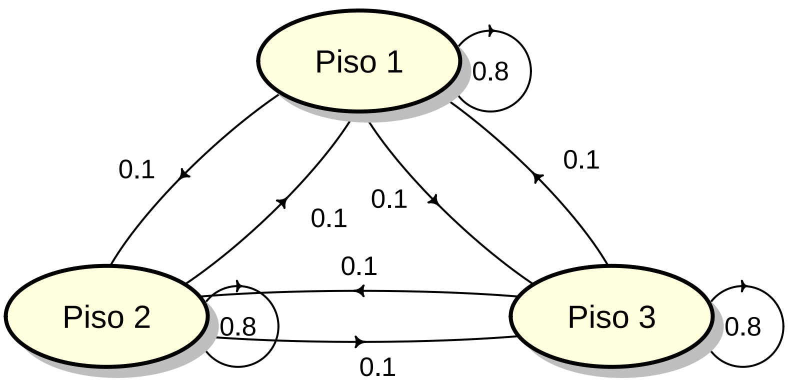
```
markov <- new("markovchain", transitionMatrix = matriz_cambio,
              states = c("Piso 1", "Piso 2", "Piso 3"),
              name =  "Distribucion del personal")
markov
```

```
## Distribucion del personal
##  A  3 - dimensional discrete Markov Chain defined by the following states:
##  Piso 1, Piso 2, Piso 3
##  The transition matrix  (by rows)  is defined as follows:
##        Piso 1 Piso 2 Piso 3
## Piso 1    0.8    0.1    0.1
## Piso 2    0.1    0.8    0.1
## Piso 3    0.1    0.1    0.8
```

Un esquema del problema es el siguiente:

```
nombres <- c("Piso 1", "Piso 2", "Piso 3")
row.names(matriz_cambio) <- nombres
```

```
plotmat(matriz_cambio,pos = c(1,2),
        lwd = 1, box.lwd = 2,
        cex.txt = 0.8,
        box.size = 0.1,
        box.type = "circle",
        box.prop = 0.5,
        box.col = "light yellow",
        arr.length=.1,
        arr.width=.1,
        self.cex = .4,
        self.shifty = -.01,
        self.shiftx = .13,
        main = "")
```



Para saber que ocurre en los primeros años, definimos las condiciones iniciales.

```
distribucion_inicial <- c(.4,.3,.3)
estado_y1 <- distribucion_inicial*markov^1; estado_y1
```

```
##      Piso 1 Piso 2 Piso 3
## [1,]   0.38   0.31   0.31
```

```
estado_y2 <- distribucion_inicial*markov^2; estado_y2
```

```
##      Piso 1 Piso 2 Piso 3
## [1,]  0.366  0.317  0.317
```

Despues del año 1, la proporción corresponde a `estado_y1` Despues del año 2, la proporción corresponde a `estado_y2` Lo que ocurre es que las proporciones están comenzando a equilibrarse.

Ejercicio 2.2

¿Se le ocurre otra situacion donde podria tener una situacion como la descrita?

Respuesta: Un ejemplo interesante es el presentado por (Hogendoorn et al., 2016). Se puede pensar la ocurrencia de una enfermedad y modelarla en tres categorias discretas: Sano, Enfermo, Muerto. En cada ciclo el paciente puede: moverse entre los estados Enfermo y Sano, mantenerse en el estado en que estaba a comienzo del ciclo (con una probabilidad determinada por la incidencia y prevalencia de la enfermedad), o ir al estado Muerto. En el largo plazo, en una población fija toda poblacion termina en el último estado.

Ejercicio 2.3

¿Cuál es la distribución para tiempos muy largos?

Respuesta: Para conocer la distribución en un tiempo largo, se computa la distribución en el estado estacionario.

```r
steadyStates(markov)
```

```
##      Piso 1 Piso 2 Piso 3
## [1,] 0.3333 0.3333 0.3333
```

Llegado este punto, en todos los pisos existe la misma proporción de personas.

Santiago, Mayo del 2018