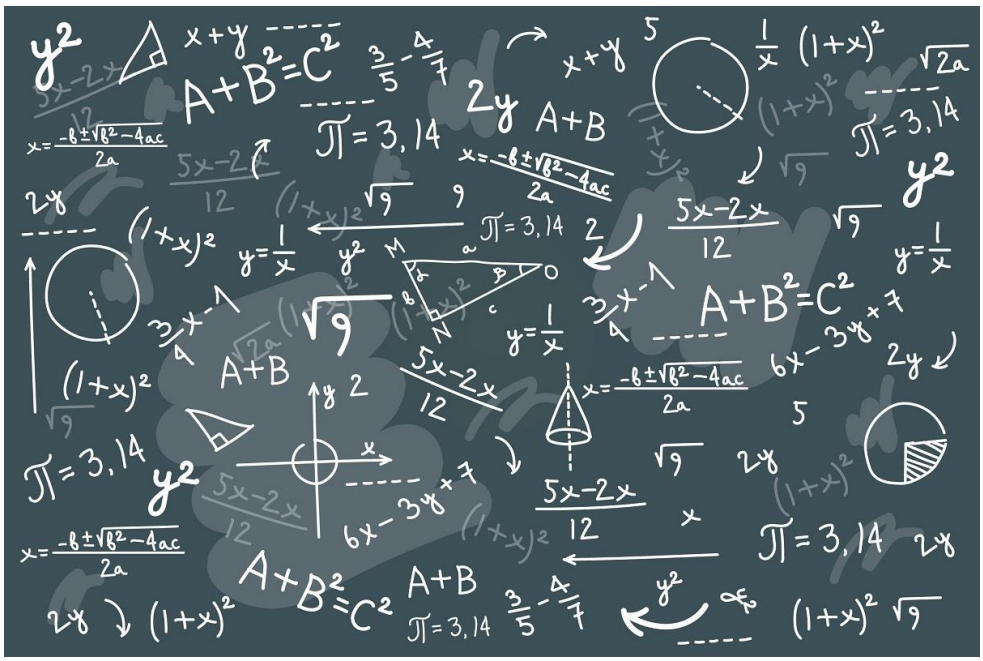
Universidad Rey Juan Carlos



MEMORIA PRÁCTICA MATES

MEMORIA 2

ÁLVARO GÓMEZ FUEYO//LUIS ORTIZ FERNÁNDEZ//ADRIAN ANTÓN PÉREZ

GRADO EN ING. DE CIBERSEGURIDAD

MEMORIA 2-MATEMATICAS DISCRETA Y ALGEBRA

EJERCICIO 1)

a)

Para este ejercicio, haremos una operación con matrices, para obtener la matriz C. El código lo haremos con la aplicación Rstudio:

MATRIZ A:

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    7    0    9    2    2    2    4    8
[2,]    5    1    7    0    7    1    6    0
[3,]    1    1    8    7    3    7    7    1
```

El resultado de la operación $C = 2\ A \cdot A^T$

```
     [,1] [,2] [,3]
[1,]  444  276  298
[2,]  276  322  264
[3,]  298  264  446
```

b)

Para el siguiente apartado crearemos una nueva matriz que sea C^2 luego con el comando sum(D) sumaremos los valores de todas las casillas. Cuyo resultado es:

```
> C
     [,1] [,2] [,3]
[1,]  444  276  298
[2,]  276  322  264
[3,]  298  264  446
> D=C**2
> sum(D)
[1] 969088
```

El código utilizado es el siguiente para todo el ejercicio es el siguiente:

```
#EJERCICIO 1
#A
A=matrix(c(7,0,9,2,2,2,4,8,
           5,1,7,0,7,1,6,0,
           1,1,8,7,3,7,7,1),ncol=8,nrow=3,byrow=T)
A
t(A)
C=(2*A)%*%(t(A))
C
#B
D=C**2
sum(D)
```

EJERCICO 2)

a)

En este ejercicio hallaremos las soluciones de un sistema de ecuaciones a la vez que su rango;

La matriz A utilizada es →

```
> A
     [,1] [,2] [,3]
[1,]   -1   -2    3
[2,]   -1    3   -1
[3,]    2   -5    5
```

Mientras que la de soluciones o también conocida como ampliada es →

```
> b
     [,1]
[1,]    9
[2,]   -6
[3,]   17
```

El siguiente paso será construir la matriz A ampliada y calcular mediante un comando el rango de esta; con esto sabremos que si los vectores son linealmente independientes.

```
> A2
     [,1] [,2] [,3] [,4]
[1,]   -1   -2    3    9
[2,]   -1    3   -1   -6
[3,]    2   -5    5   17
```

RANGO→

```
[1] 3
```

La solución de este ejercicio es que el rango de la matriz ampliada es 3, deduciendo que los 3 vectores son linealmente independientes.

b)

Para este apartado resolveremos un simple sistemas de ecuaciones AX=B; que con un simple comando resolveremos dando como resultado;

```
> solve(A,b)
            [,1]
[1,] -0.05263158
[2,] -1.31578947
[3,]  2.10526316
```

(ESTOS SON LOS RESULTADOS DE LA ECUACION)

El código utilizado para este ejercicio es el siguiente

```
#EJERCICIO 2
#A)
library(Matrix)
#Definición de matrices


A=matrix(c(-1,-2,3,
           -1,3,-1,
           2,-5,5),ncol=3,byrow=T)
A
b=matrix(c(9,-6,17),ncol=1,nrow=3,byrow=T)
b

A2=matrix(c(-1,-2,3,9,
            -1,3,-1,-6,
            2,-5,5,17),ncol=4,nrow=3,byrow=T)

#Rango (A|b)
A2
rankMatrix(A2)

#RANGO DE LA MATRIZ (A|b)=3
# Si son linealmente independientes
#B)
#Solucion del sistema lineal A*x=b
solve(A,b)
```

EJERCICIO 3)

Para este ejercicio usaremos la matriz del ejercicio 2;

```
> A
     [,1] [,2] [,3]
[1,]   -1   -2    3
[2,]   -1    3   -1
[3,]    2   -5    5
>
```

a) Para el primer apartado calcularemos sus autovalores y autovectores, con la formula:

|A-λi|,

Primero obtendremos los autovalores:

```
> autovalores
[1]  7.462834 -1.843714  1.380881
```

Tras obtener estos valores podemos conseguir los autovectores, los cuales formaran la siguiente matriz:

```
> autovectores
           [,1]       [,2]      [,3]
[1,] -0.3791182  0.9725073 0.3381581
[2,]  0.2824037  0.1673433 0.6368081
[3,] -0.8812023 -0.1619439 0.6929102
```

b) Para este apartado tendremos que comprobar si la matriz es diagonizable. Para estos existen diferentes procesos: uno de ellos seria mirar la multiplicidad de los autovalores obtenidos en al apartado anterior y ver si coincide con la dimensión de la matriz. Pero mejor utilizaremos la siguiente formula: $A = P \cdot D \cdot P^{-1}$ (siendo P D las correspondientes matrices de autovectores y autovalores). Esto lo pasaremos a R obteniendo lo siguiente:

```
> D=diag(autovalores)
> B=autovectores%%D%%solve(autovectores)
> B
           [,1]        [,2]      [,3]
[1,] -0.1753725         NaN       NaN
[2,]        NaN -0.01231518       NaN
[3,]        NaN         NaN 0.2620597
```

Como vemos el programa no nos da ningún error siendo esto un significativo de que cumple la fórmula y que esta es diagonalizable. A su vez hemos puesto ya su fórmula diagonalizada en la foto de arriba (los valores que ponen NaN es un 0)

El código utilizado para este ejercicio es →

```
A=matrix(c(-1, -2, 3,
           -1, 3, -1,
           2, -5, 5),ncol=3,nrow=3,byrow=T)
descomposicion=eigen(A)
autovalores=descomposicion$values
autovectores=descomposicion$vectors
autovalores
autovectores


D=diag(autovalores)
B=autovectores%*%D%*%solve(autovectores)
B
```

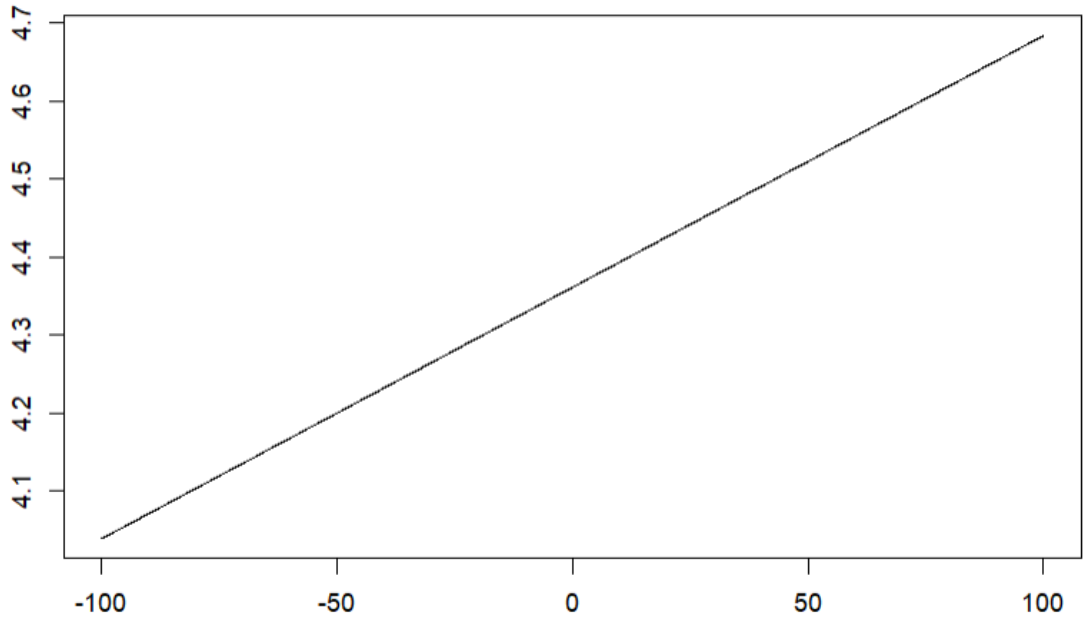
EJERCICIO 4)

Los DNIs utilizados son los siguientes: 70922248 // 11873771

Creamos los siguientes puntos

(7,1) (0,1) (9,8) (2,7) (2,3) (2,7) (4,7) (8,1)

Mediante R obtendremos la recta que más se ajusta por mínimos cuadrados. Primero tendremos que crear una nube de puntos, la cual avanzaremos de 0.1 en 0.1 y posteriormente tras unas líneas de código podemos mostrar nuestra primera recta:



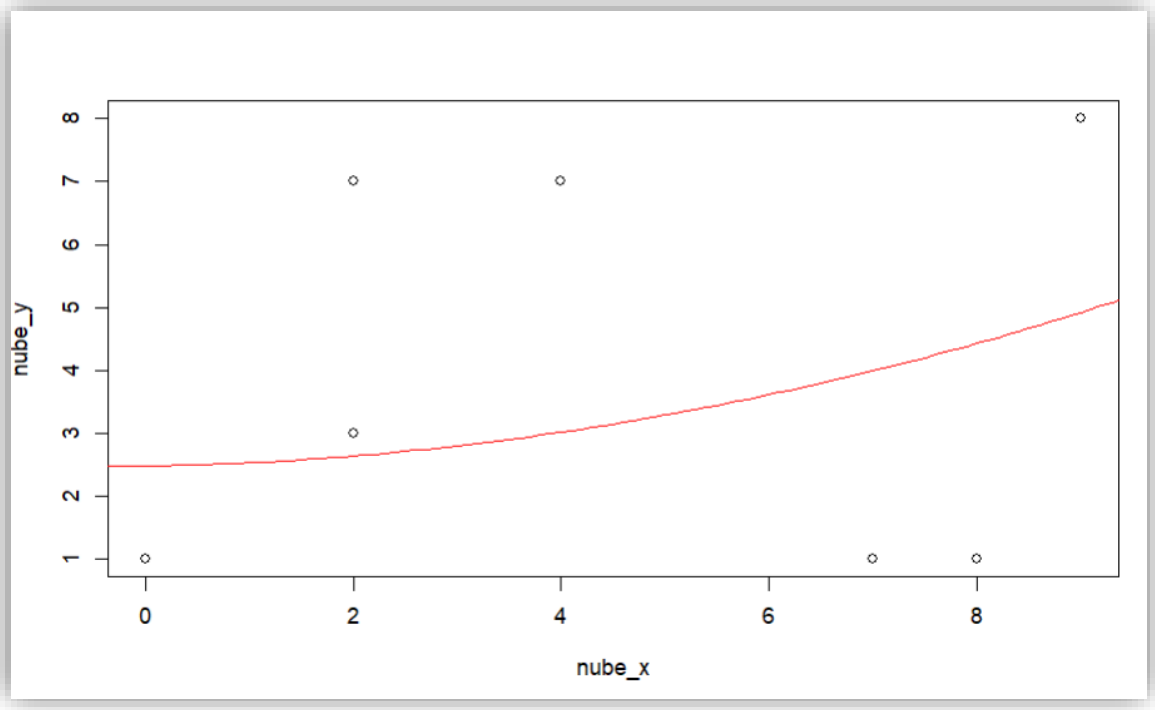
Para el siguiente apartado buscaremos la parábola que mejor se ajusta por mínimos cuadrados

Usaremos una matriz, cuya primera columna esta formada por las cifras del primer DNI que escribo al principio del ejercicio elevada al cuadrado. Para la matriz de la siguiente matriz (B) será el tercer DNI→ 51707160

Los resultados obtenidos tras hacer todas las operaciones que mas adelantes adjuntare con una foto, dan como resultado los valores A, B, C que serán → Ax^2+Bx+C, las cuales formarán la parábola

```
> T
             [,1]
[1,] 0.02735239
[2,] 0.02469064
[3,] 2.48325909
```

La parábola obtenida es la siguiente



Por último, el código utilizado para este ejercicio es el siguiente

```r
##EJERCICIO 4
nube_x=c(7,0,9,2,2,2,4,8)
nube_y=c(1,1,8,7,3,7,7,1)
plot(nube_x,nube_y)

A=matrix(c(7,1,
           0,1,
           9,1,
           2,1,
           2,1,
           2,1,
           4,1,
           8,1),ncol=2,byrow=T)
B=matrix(c(1,
           1,
           8,
           7,
           3,
           7,
           7,
           1),ncol=1,byrow=T)

C=t(A)%%A
D=t(A)%%B

S=solve(C)%%D
S
x=seq(-100,100,0.1)
y=0.003225806*x + 4.361290323
plot(x,y,type="l")
```

```r
#parabola
A=matrix(c(49,1,1,
           0,9,1,
           81,8,1,
           4,7,1,
           4,3,1,
           4,7,1,
           16,7,1,
           64,1,1),ncol=3,byrow=T)
A
t(A)
M=solve((t(A)%*%A))#multiplicamos la traspuesta por la matriz yhallamos la inversa
n=(M%*%t(A)) #lo multiplicamos por la traspuesta

B=matrix(c(5,
           1,
           7,
           0,
           7,
           1,
           6,
           0),ncol=1,byrow=T)

T=(n%*%B)
T
x=seq(-100,100,0.1)
y=0.02735239*(x)**2+0.02469064*x + 2.48325909     #crearemos una nube de puntos y la parabola
plot(x,y,type="l")
plot(nube_x,nube_y)
lines(x,y,col="red")
```