Matrices

Sitio:	Plataforma de Formación On line del Instituto Andaluz de Administración Pública	Imprimido por:	ALFONSO LUIS MONTEJO RAEZ
Curso:	(I22F-PT05) Entorno de Programación R	Día:	lunes, 4 de abril de 2022, 10:26
Libro:	Matrices		

Tabla de contenidos

1. Que es una matriz

Una **matriz** de orden k por p es una tabla rectangular de elementos formada por k filas y p columnas que presenta las siguientes características.

- Si el número de filas y de columnas coinciden la matriz se denomina cuadrada.
- Todos los elementos almacenados en una matriz deben ser del mismo tipo, normalmente numéricos o carácter.
- Si mezclamos tipos distintos todos se convierten a un tipo que sea común a todos ellos, normalmente a carácter.

Un ejemplo de matriz sería la siguiente:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Este tipo de dato se parece más a una tabla de datos de las que se suele usar cuando trabajamos en estadística, pero presenta un problema: no podemos mezclar tipos de datos en una misma matriz. Una matriz o contiene números o contiene caracteres, pero no puede tener datos mezclados.

2. Creando matrices

Ya que sabemos lo que es una matriz vamos a ver como se crean.

Principalmente tenemos dos formas de crear una matriz en R:

- Usando la orden **matrix()**.
- Concatenando vectores con las órdenes **rbind()** o **cbind()**.

Vamos a ver como trabajan estas órdenes.

2.1. matrix()

Ya que sabemos lo que es una matriz vamos a ver como se crea. Para ello usamos la orden matrix() de la siguente manera:

matrix(data, nrow=nº de filas, byrow=T o F)

- **data**. Los datos que quiero almacenar en la matriz.
- **nrow**. Número de filas que va a tener la matriz.
- **byrow**. Si la matriz se va a construir por filas o por columnas. Toma los valores verdadero (T o TRUE) o falso (F o FALSE).

Ejemplo

Vamos a crear la matriz A que estamos usado como ejemplo.

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Si queremos crear en R la matriz la orden sería:

```
> matrix(1:9,nrow=3,byrow=T)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

Los datos para rellenar la matriz son la sucesión del 1 al 9 (recordar que eso se logra con los dos puntos), el número de filas sería 3 y queremos que se empiece a construir por filas (parámetro byrow=T), es decir, empieza a colocar los datos en la primera fila.

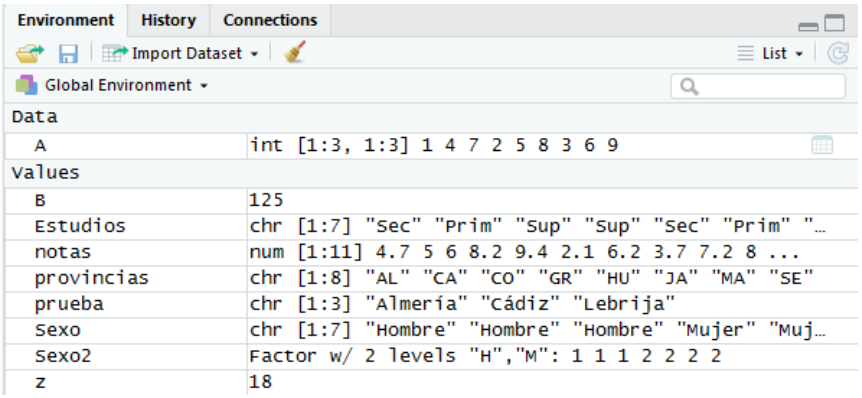
Para que veáis la diferencia en la construcción si utilizo el parámetro byrow=F, es decir, empezando a colocar los datos en la primera columna.

```
> matrix(1:9,nrow=3,byrow=F)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Como siempre en R, si la orden que usamos no se asigna a ningún objeto su resultado se pierde.

```
> matrix(1:9,nrow=3,byrow=T)->A
> class(A)
[1] "matrix"
```

Si la orden la asignamos a un objeto llamado A, creamos la matriz y esta aparece en la pestaña Environment como un objeto clasificado como Data.



Si os fijáis, RStudio clasifica los objetos como Data o Values en función de la dimensión de los datos. En el caso de valores sueltos o vectores, como solamente tienen una dimensión, serían **Values**, sin embargo, una matriz, al tener dos dimensiones ya es un **Data**.

2.2. Concatenando vectores

Otra forma de crear una matriz sería concatenando vectores como filas usando la función **rbind()** o concatenando como columnas usando **cbind()**. En ambos casos los parámetros a usar serían los vectores que queremos concatenar.

Vamos a ver un ejemplo.

Ejemplo

Si quiseramos construir la misma matriz A que hemos visto anteriormente pero usando la orden **rbind()**, tenemos que concatenar tres vectores: c(1,2,3) para la primera fila, c(4,5,6) para la segunda fila y c(7,8,9) para la tercera fila.

```
> rbind(c(1,2,3),c(4,5,6),c(7,8,9))
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

Si usamos la orden **cbind()**, lo que hacemos es concatenar los tres vectores anteriores pero por columnas y el resultado sería:

```
> cbind(c(1,2,3),c(4,5,6),c(7,8,9))
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

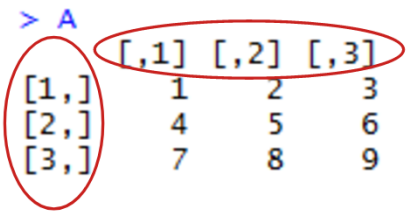
3. Trabajando con matrices

Ya sabemos como crear matrices ahora vamos a ver como podemos trabajar con ellas. Lo principal que vamos a ver va a ser:

- Acceder al contenido de una matriz.
- Renombrar el contenido de una matriz.
- Operaciones con una matriz.

3.1. Acceso

Si os habéis fijado, cada vez que creamos una matriz, tanto las filas como las columnas aparecen numeradas con unos índices.

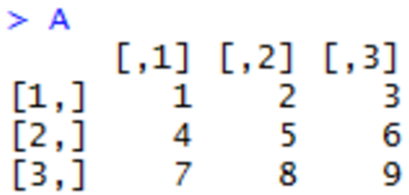


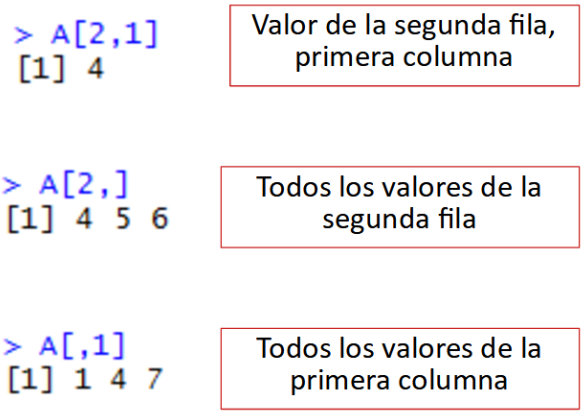
Nombres por defecto para las filas y columnas

Estos nos permite hacer referencia al contenido de la matriz usando sus índices para indicar que posición se quiere consultar teniendo en cuenta que **[i,j] indica la fila i y la columna j** (esto es igual que cuando jugamos a los barquitos diciendo que en que la fila i y en la columna j hay un barco escondido).

Ejemplo

Vamos a acceder al contenido de la matriz A de varias formas. Hay que tener en cuenta que **tenemos dos referencias: la primera es la fila y la segunda es la columna**. Si alguna de las dos referencias no la rellenamos se entiende que queremos acceder a todas las posiciones de la referencia que queda en blanco.





3.2. Renombrando contenido

Aunque R nombra las filas y las columnas de manera automática, también nos ofrece la posibilidad de cambiar esos nombres por otros que eligamos con las siguientes órdenes:

- **dimnames()**. Accede y cambia el nombre de filas y de columnas.
- **rownames()**. Accede y cambia el nombre de las filas.
- **colnames()**. Accede y cambia el nombre de las columnas.

Las tres funciones sirven para acceder y ver los nombres de filas y/o columnas. Podemos simplemente visualizarlos si usamos la función, o cambiarlos si le hacemos una asignación.

Evidentemente tenemos que darle nombre a todas las filas y/o columnas, es decir, si tenemos tres filas yo tengo que darle un vector con tres nombres. Si le doy un vector con dos nombres R da un error diciendo que faltan valores de nombres.

Una vez cambiados los nombres podemos acceder al contenido de la matriz usando los nuevos nombres que les hemos asignado.

Ejemplo 1

Vamos a renombrar las filas y columnas de la matriz A que hemos creado al principio usando **dimnames()**.

```
> dimnames(A)<-list(c('AL','CA','CO'),c('Población','Extensión','Municipios'))
```

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

Matriz A con valores por defecto

```
> A
   Población Extensión Municipios
AL         1         2          3
CA         4         5          6
CO         7         8          9
```

Matriz A con nombres en las filas y columnas

En la orden dimnames() tenemos que usar dos vectores; el primero indicaría el nombre de las filas y el segundo el nombre de las columnas. Para usar esta orden hay que pasarle los valores como una lista usando **list()**.

La lista es un tipo de dato de R que sirve para almacenar todo lo que queramos (un cajón desastre) y que veremos con mayor profundidad en próximos capítulos.

También podemos usar las órdenes **rownames()** o **colnames()** para conseguir el mismo efecto.

```
> rownames(A)
[1] "AL" "CA" "CO"
> colnames(A)
[1] "Población"  "Extensión"  "Municipios"
```

Ejemplo 2

Vamos a acceder al contenido de la matriz usando los nuevos nombres que les hemos puesto, aunque también podemos seguir usando los números de las filas o columnas para referirnos a su contenido.

```
> A['AL',]
 Población  Extensión Municipios
         1          2          3
> A[1,]
 Población  Extensión Municipios
         1          2          3
```

Accedemos al contenido de la primera fila por su nombre o por su posición

3.3. Operaciones con matrices

También podemos hacer operaciones con las matrices. Vamos a ver una serie de órdenes o funciones que nos sirven para trabajar con ellas.

La primera serie de ordenes nos ofrecen información sobre la dimensión de la matriz.

- **nrow()**. Número de filas que tiene la matriz.
- **ncol()**. Número de columnas que tiene la matriz.
- **dim()**. Número de filas y columnas que tiene la matriz.

La segunda serie de órdenes realizan operaciones aritméticas sobre las dimensiones de la matriz.

- **colSums()**. Suma los valores de cada columna.
- **rowSums()**. Suma los valores de caca fila.
- **colMeans()**. Media de los valores de cada columna.
- **rowMeans()**. Media de los valores de cada fila.

También podemos aplicar a una matriz funciones que ya conocemos como **summary()** que nos devuelve un resumen estadístico de las columnas de la matriz.

Ejemplo

Con nuestra vieja amiga la matriz A, vamos a aplicar las funciones que hemos visto para ver su resultado.

```
> A
   Población Extensión Municipios
AL         1         2          3
CA         4         5          6
CO         7         8          9
```

La primera serie de funciones nos ofrece información sobre las dimensiones de la matriz A.
En este caso todos los valores son iguales ya que A es una matriz cuadrada (mismo número de filas que de columnas)

```
> nrow(A)
[1] 3
> ncol(A)
[1] 3
> dim(A)
[1] 3 3
```

La segunda serie de funciones ofrece resúmenes estadísticos de las filas o columnas de la matriz.

```
> colSums(A)
 Población  Extensión Municipios
        12         15         18
> rowSums(A)
AL CA CO
 6 15 24
> colMeans(A)
 Población  Extensión Municipios
         4          5          6
> rowMeans(A)
AL CA CO
 2  5  8
```

Aplicando la función **summary()** a la matriz nos devuelve información descriptiva sobre las columnas.

```
> summary(A)
   Población      Extensión      Municipios
 Min.   :1.0   Min.   :2.0   Min.   :3.0
 1st Qu.:2.5   1st Qu.:3.5   1st Qu.:4.5
 Median :4.0   Median :5.0   Median :6.0
 Mean   :4.0   Mean   :5.0   Mean   :6.0
 3rd Qu.:5.5   3rd Qu.:6.5   3rd Qu.:7.5
 Max.   :7.0   Max.   :8.0   Max.   :9.0
```

Evidentemente estas funciones solo se pueden usar si el contenido de la matriz es numérico. En el caso de que la matriz tuviera texto las funciones darían un error (salvo summary que nos ofrece una tabla de frecuencias con los valores).