Francesca Mazzia
Dipartimento di Matematica
Università di Bari

R: matrici e vettori

Per costruire una matrice possiamo usare la function matrix:

```
> A <- matrix(nrow = 3, data = c(3, 8, 2, 5, 9, 3, 7, 10, 5))
> A

     [,1] [,2] [,3]
[1,]    3    5    7
[2,]    8    9   10
[3,]    2    3    5
```

la variabile A sarà una matrice con tre righe e tre colonne con elementi A[1,1]=3, A[1,2]=5, ..., A[2,1]=8, ..., A[3,3]=5. Ogni elemento di una matrice viene individuato da una coppia di indici, il primo indice è detto indice di riga, il secondo indice di colonna.

R consente anche di memorizzare una sequenza di istruzioni in un file, aggiungendo alle funzioni R predefinite funzioni definite dall'utente.

I nomi dei file R devono avere l'estensione .R, cioè devono essere del tipo *nomefile.R*. Essi possono essere scritti utilizzando un qualsiasi editore di testi.

Essi definiscono una nuova funzione R, accettano dei dati di input e restituiscono dei dati di output come risultato della loro elaborazione. L'insieme dei dati di input ed autput può essere vuoto. La prima istruzione in una function deve essere:

```
 nomefunction = function(i1,i2,...,im){

  istruzioni

return(list(o1,o2,...,on))
}
```

dove $o_1, o_2, \ldots, o_n$ sono le variabili di output e $i_1, i_2, \ldots, i_m$ sono le variabili di input. I dati di output sono organizzati in una lista con l'istruzione list. Se il dato e' uno solo si puo' non costruire la lista.

Per essere usata una nuova funzione deve essere caricata nell'ambiente con l'istruzione source("nomefunction.R")

Per esempio possiamo scrivere il file `testfun.R`

```
testfun = function(x){
  y=cos(c)
return(y)
}
```

al prompt possiamo caricare la nuova funzione nell'ambiente

```
> source("../R/testfun.R")
```

ed eseguirla

```
> y = testfun(10.1)
> y

[1] -0.7805682
```

Quando abbiamo terminato di lavorare con R possiamo uscire con il comando:

```
> q()
```

Quando si esce da R viene chiesto se si vuole salvare il contenuto delle variabili o meno.

Informazioni importanti relative alla rappresentazione dei numeri sono memorizzate in *.Machine*:

```
> .Machine

$double.eps
[1] 2.220446e-16

$double.neg.eps
[1] 1.110223e-16

$double.xmin
[1] 2.225074e-308

$double.xmax
[1] 1.797693e+308

$double.base
[1] 2

$double.digits
[1] 53

$double.rounding
[1] 5

$double.guard
[1] 0

$double.ulp.digits
[1] -52

$double.neg.ulp.digits
[1] -53

$double.exponent
```

```
[1] 11

$double.min.exp
[1] -1022

$double.max.exp
[1] 1024

$integer.max
[1] 2147483647

$sizeof.long
[1] 4

$sizeof.longlong
[1] 8

$sizeof.longdouble
[1] 12

$sizeof.pointer
[1] 4
```

Le costanti `T, F` vengono utilizzate nelle espressioni logiche

```
> F

[1] FALSE

> T

[1] TRUE
```

R utilizza l'aritmetica IEEE e permette di rappresentare infinito e NaN, per cambiare il floating point exception mode si usa la function `ieee`

```
> 1/0

[1] Inf
```

```
> 0/0

[1] NaN
```

Ci sono alcune function che permettono di verificare se un numero è infinito o se è uguale ad NaN. Eseguima le seguenti operazioni:

```
> is.infinite(1/0)

[1] TRUE

> is.finite(1/0)

[1] FALSE

> is.nan(1/0)

[1] FALSE
```

R consente di scrivere le function utilizzando la sequenza, la selezione e la ripetizione. Per la selezione si usa le parole chiavi `if then else`. Eseguiamo:

```
SINTASSI

   if ( instruzioni1 )
         istruzioni2
   else
         istruzioni3

 SINTASSI

    if ( statement1 )
         statement2
    else if ( statement3 )
         statement4
    else if ( statement5 )
         statement6
    else
         statement8
```

Per i cicli di ripetizione possiamo utilizzare l'istruzione `for` e l'istruzione `while` e `repeat`

```
DESCRIZIONE FOR

  for ( name in vector )
        statement1

DESCRIZIONE REPEAT

  repeat statement

 DESCRIZIONE WHILE

  while ( statement1 ) statement2
```

L'istruzione `break` all'interno di un loop, permette di interromperlo.