

## Esercitazione 3

*Istruzioni trattate: error, ciclo while, printf.*

Nella prima parte di questa esercitazione vedremo una realizzazione del *metodo di bisezione*. Nella seconda parte utilizzeremo la realizzazione per approssimare uno zero della funzione  $f(x) = x^2 - 2$ .

### Prima parte

La definizione che segue è una realizzazione del *metodo di bisezione* che utilizza un criterio d'arresto di tipo assoluto. Le linee di commento immediatamente al di sotto dell'*intestazione* dichiarano lo scopo della funzione ed è buona norma inserirle.

```
function [z,v] = Bisezione(f,a,b,delta)
//
// Applica il metodo di bisezione alla funzione f a partire dall'intervallo
// [a,b]. L'iterazione si arresta quando la misura dell'ultimo intervallo
// calcolato è minore di delta (criterio di arresto di tipo assoluto)
// oppure si è trovato uno zero della funzione f.
//
// Ad ogni iterazione nella console vengono mostrati il numero dell'iterazione
// gli estremi e l'ampiezza dell'ultimo intervallo calcolato.
//
// La funzione restituisce l'approssimazione suggerita z e v = f(z).
//
if f(a) * f(b) < 0 then
    error('la funzione non assume valori di segno opposto agli estremi');
else
    k = 0;
    x = (a+b)/2;
    printf('k = %4.0f , a = %.2e , b = %.2e , ampiezza = %.3e\n', k,a,b,b-a);
    while ~(b-a) < delta | f(x) == 0,
        if sign(f(a)) ~= sign(f(x)) then b = x; else a = x; end;
        x = (a+b)/2;
        k = k + 1;
        printf('k = %4.0f , a = %.2e , b = %.2e , ampiezza = %.3e\n', k,a,b,b-a);
    end;
    z = x;
    v = f(z);
end;
endfunction
```

Nella definizione della procedura compaiono tre nuove istruzioni:

- **error**

Questa *funzione predefinita* ha lo scopo di *gestire un errore*. Precisamente, l'istruzione

```
error(<stringa descrittiva>)
```

ha due effetti: *interrompe l'esecuzione corrente e riporta l'interprete al livello del prompt e mostra* <stringa descrittiva> *nella console come parte di un messaggio di errore.*

- **ciclo while**

Questo costrutto predefinito consente di definire *iterazioni*. La sequenza:

```
while <condizione>, <istruzioni> end;
```

dove  $\langle \text{condizione} \rangle$  è una funzione a valori in  $\{T, F\}$ , significa:

`finché  $\langle \text{condizione} \rangle = T$ , ripeti  $\langle \text{istruzioni} \rangle$ ;`

Ad esempio:

```
-->i = 1; while i < 5, disp(i); i = i + 1; end;
```

- 1.
- 2.
- 3.
- 4.

- **printf**

Questa *funzione predefinita* ha lo scopo di *stampare nella console una stringa contenente il valore di una o più espressioni*. Precisamente, se  $v_1, \dots, v_n$  sono  $n$  espressioni, e  $\langle \text{frase} \rangle$  è una stringa che contiene  $n$  opportune sottostringhe  $\langle \text{formato } 1 \rangle, \dots, \langle \text{formato } n \rangle$  dette *stringhe segnaposto*, ed eventuali altre opportune *sottostringhe atte alla formattazione* del testo da stampare, l'istruzione:

`printf( $\langle \text{frase} \rangle, v_1, \dots, v_n$ )`

stampa sulla console, con la formattazione specificata, la stringa che si ottiene sostituendo, in  $\langle \text{frase} \rangle$ , alla sottostringa  $\langle \text{formato } i \rangle$  la stringa che rappresenta *l'arrotondato* del valore della  $i$ -esima espressione nell'insieme numerico e con il formato specificato dalla sottostringa stessa  $\langle \text{formato } i \rangle$ .

Ad esempio, si ha:

```
-->printf('\ne vale circa %12.3e, pi greco vale %5.3e\n\n...o quasi!','%e,%pi)
```

```
e vale circa      2.718e+00, pi greco vale 3.142e+00
```

```
...o quasi!
```

In questo caso sono presenti due *stringhe segnaposto* e due corrispondenti espressioni. Entrambe le *stringhe segnaposto* hanno la forma  $\%m.ne$  con  $m$  ed  $n$  numeri interi positivi. Ciascuna delle *stringhe segnaposto* viene sostituita dalla stringa ottenuta (a) calcolando l'arrotondato  $\xi$  del valore dell'espressione corrispondente in  $F(10, n + 1)$ , (b) calcolando la frazione con segno  $g$  e l'esponente  $b$  di  $\xi$  in base dieci, (c) costruendo le stringhe  $\sigma_1$  e  $\sigma_2$  che rappresentano, in base dieci, rispettivamente  $10g$  e  $b - 1$  e, infine, (d) antepoendo alla stringa  $\sigma_1\sigma_2$  il numero minimo di spazi necessario per ottenere una stringa di almeno  $m$  caratteri.

La stringa ottenuta dopo aver sostituito ciascuna delle *stringhe segnaposto* viene poi stampata nella console *interpretando le stringhe atte alla formattazione*. Nell'esempio è presente, tre volte, una sola di tali stringhe, `\n`, che comanda all'interprete di *andare a capo*.

I dettagli riguardanti le molte sottostringhe da usare per ottenere l'insieme numerico ed il formato di stampa desiderati si trovano nella pagina di *help* relativa al termine *printf\_conversion*. Qualche altro esempio si trova negli esercizi finali.

Si osservi che per decidere se l'algoritmo  $f$  restituisce valori diversi da zero e di segno opposto agli estremi dell'intervallo  $[a, b]$  si è utilizzata la disuguaglianza  $f(a) * f(b) < 0$ , ovvero:  $f(a) \otimes f(b) = \text{rd}(f(a) f(b)) < 0$ . In  $M = F(2, 53)$ , si ha:

$$\text{rd}(f(a) f(b)) < 0 \quad \text{se e solo se} \quad f(a) f(b) < 0$$

dunque le disuguaglianze  $f(a) * f(b) < 0$  e  $f(a) f(b) < 0$  sono *equivalenti*. Ma in *Scilab* si ha  $M = F_d(2, 53, -1021, 1024)$ . In tal caso zero non è punto di accumulazione di  $M$  e quindi le due

disuguaglianze *non sono* equivalenti. Infatti, detto  $m$  il punto medio dell'intervallo  $[\pi(0), 0]$ , si ha  $m > 0$  e:

$$\text{rd}(f(a) f(b)) < 0 \quad \text{se e solo se} \quad f(a) f(b) < m$$

La conseguenza pratica di questo è che *la procedura potrebbe segnalare che la funzione non assume valori di segno opposto agli estremi anche se ciò non è vero*. Una condizione che aggira questo problema è:

$$f(a) \neq 0 \ \& \ f(b) \neq 0 \ \& \ \text{sign}(f(a)) \neq \text{sign}(f(b))$$

che, sia in  $F(2, 53)$  che in  $F_d(2, 53, -1021, 1024)$ , è equivalente a:  $f(a) f(b) < 0$ .

## Seconda parte

Consideriamo la seguente realizzazione della funzione  $f(x) = x^2 - 2$ :

```
function y = fun(x)
//
// x, y: matrici di numeri reali della stessa dimensione
//
y = x .^ 2 - 2;
endfunction
```

Per ottenere un'approssimazione dello zero positivo di  $f$  ( $\sqrt{2} = 1.41421356237309504880\dots$ )<sup>1</sup> con errore assoluto non superiore a  $10^{-4}$  si applica la procedura **Bisezione** alla funzione **fun** a partire dall'intervallo  $[0, 2]$  e con **delta** =  $10^{-4}$ . Si ottiene:

```
-->a = 0; b = 2; delta = 1d-4;

-->[z4,v4] = Bisezione(fun,a,b,delta)
k = 0 , a = 0.00e+00 , b = 2.00e+00 , ampiezza = 2.000e+00
k = 1 , a = 1.00e+00 , b = 2.00e+00 , ampiezza = 1.000e+00
k = 2 , a = 1.00e+00 , b = 1.50e+00 , ampiezza = 5.000e-01
k = 3 , a = 1.25e+00 , b = 1.50e+00 , ampiezza = 2.500e-01
k = 4 , a = 1.38e+00 , b = 1.50e+00 , ampiezza = 1.250e-01
k = 5 , a = 1.38e+00 , b = 1.44e+00 , ampiezza = 6.250e-02
k = 6 , a = 1.41e+00 , b = 1.44e+00 , ampiezza = 3.125e-02
k = 7 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 1.562e-02
k = 8 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 7.812e-03
k = 9 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 3.906e-03
k = 10 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 1.953e-03
k = 11 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 9.766e-04
k = 12 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.883e-04
k = 13 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.441e-04
k = 14 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.221e-04
k = 15 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 6.104e-05
v4 =

0.0000043
z4 =

1.4142151

-->printf('\nz4 = %16.15e , fun(z4) = %16.15e',z4,v4)

z4 = 1.414215087890625e+00 , fun(z4) = 4.314817488193512e-06
```

La procedura si arresta per aver determinato, dopo 15 iterazioni come prevedibile, un intervallo di ampiezza minore di  $10^{-4}$  – il valore della funzione **fun** in **z4** è, infatti, diverso da zero. L'ultima

<sup>1</sup>La rappresentazione posizionale di  $\sqrt{2}$  in base dieci è stata ricopiata dalla pagina di Wikipedia relativa al termine radice quadrata: [https://it.wikipedia.org/wiki/Radice\\_quadrata](https://it.wikipedia.org/wiki/Radice_quadrata)

istruzione stampa gli arrotondati in  $F(10,16)$  dei valori determinati di  $z4$  e  $v4$ . Si osservi che l'approssimazione suggerita dalla procedura soddisfa la richiesta, infatti si ha (vedi Esercizio 3):  $|z4 - \sqrt{2}| < 10^{-5}$ .

Per ottenere un'approssimazione dello stesso zero di  $f$  ma con errore assoluto non superiore a  $10^{-8}$  si applica la procedura Bisezione alla funzione `fun` a partire dallo stesso intervallo ma con  $\text{delta} = 10^{-8}$ . Si ottiene:

```
-->a = 0; b = 2; delta = 1d-8;

-->[z8,v8] = Bisezione(fun,a,b,delta)
k = 0 , a = 0.00e+00 , b = 2.00e+00 , ampiezza = 2.000e+00
k = 1 , a = 1.00e+00 , b = 2.00e+00 , ampiezza = 1.000e+00
k = 2 , a = 1.00e+00 , b = 1.50e+00 , ampiezza = 5.000e-01
k = 3 , a = 1.25e+00 , b = 1.50e+00 , ampiezza = 2.500e-01
k = 4 , a = 1.38e+00 , b = 1.50e+00 , ampiezza = 1.250e-01
k = 5 , a = 1.38e+00 , b = 1.44e+00 , ampiezza = 6.250e-02
k = 6 , a = 1.41e+00 , b = 1.44e+00 , ampiezza = 3.125e-02
k = 7 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 1.562e-02
k = 8 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 7.812e-03
k = 9 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 3.906e-03
k = 10 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 1.953e-03
k = 11 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 9.766e-04
k = 12 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.883e-04
k = 13 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.441e-04
k = 14 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.221e-04
k = 15 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 6.104e-05
k = 16 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 3.052e-05
k = 17 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.526e-05
k = 18 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 7.629e-06
k = 19 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 3.815e-06
k = 20 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.907e-06
k = 21 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 9.537e-07
k = 22 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.768e-07
k = 23 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.384e-07
k = 24 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.192e-07
k = 25 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 5.960e-08
k = 26 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.980e-08
k = 27 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.490e-08
k = 28 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 7.451e-09
v8 =

5.300D-09
z8 =

1.4142136

-->printf('\nz8 = %16.15e , fun(z8) = %16.15e',z8,v8)

z8 = 1.414213564246893e+00 , fun(z8) = 5.299900962540960e-09
```

La procedura si arresta per aver determinato, dopo 28 iterazioni come prevedibile, un intervallo di ampiezza minore di  $10^{-8}$  – il valore della funzione `fun` in  $z8$  è, anche questa volta, diverso da zero. Si osservi che anche questa volta l'approssimazione suggerita dalla procedura soddisfa la richiesta:  $|z8 - \sqrt{2}| < 10^{-8}$ .

Infine, per ottenere un'approssimazione dello stesso zero di  $f$  ma con errore assoluto non superiore a  $10^{-16}$  si applica la procedura Bisezione alla funzione `fun` a partire dallo stesso intervallo ma con  $\text{delta} = 10^{-16}$ . Questa volta si ottiene:

```
-->a = 0; b = 2; delta = 1d-16;
```

```
-->[z16,v16] = Bisezione(fun,a,b,delta)
k = 0 , a = 0.00e+00 , b = 2.00e+00 , ampiezza = 2.000e+00
k = 1 , a = 1.00e+00 , b = 2.00e+00 , ampiezza = 1.000e+00
k = 2 , a = 1.00e+00 , b = 1.50e+00 , ampiezza = 5.000e-01
k = 3 , a = 1.25e+00 , b = 1.50e+00 , ampiezza = 2.500e-01
k = 4 , a = 1.38e+00 , b = 1.50e+00 , ampiezza = 1.250e-01
k = 5 , a = 1.38e+00 , b = 1.44e+00 , ampiezza = 6.250e-02
k = 6 , a = 1.41e+00 , b = 1.44e+00 , ampiezza = 3.125e-02
k = 7 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 1.562e-02
k = 8 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 7.812e-03
k = 9 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 3.906e-03
k = 10 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 1.953e-03
k = 11 , a = 1.41e+00 , b = 1.42e+00 , ampiezza = 9.766e-04
k = 12 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.883e-04
k = 13 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.441e-04
k = 14 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.221e-04
k = 15 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 6.104e-05
k = 16 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 3.052e-05
k = 17 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.526e-05
k = 18 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 7.629e-06
k = 19 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 3.815e-06
k = 20 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.907e-06
k = 21 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 9.537e-07
k = 22 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.768e-07
k = 23 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.384e-07
k = 24 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.192e-07
k = 25 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 5.960e-08
k = 26 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.980e-08
k = 27 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.490e-08
k = 28 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 7.451e-09
k = 29 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 3.725e-09
k = 30 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.863e-09
k = 31 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 9.313e-10
k = 32 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.657e-10
k = 33 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.328e-10
k = 34 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.164e-10
k = 35 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 5.821e-11
k = 36 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.910e-11
k = 37 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.455e-11
k = 38 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 7.276e-12
k = 39 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 3.638e-12
k = 40 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.819e-12
k = 41 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 9.095e-13
k = 42 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.547e-13
k = 43 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.274e-13
k = 44 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.137e-13
k = 45 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 5.684e-14
k = 46 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.842e-14
k = 47 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.421e-14
k = 48 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 7.105e-15
k = 49 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 3.553e-15
k = 50 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 1.776e-15
k = 51 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 8.882e-16
k = 52 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 4.441e-16
```

```
k = 53 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.220e-16
k = 54 , a = 1.41e+00 , b = 1.41e+00 , ampiezza = 2.220e-16
```

```
.
.
.
```

La procedura, apparentemente, *non si arresta* finché l'utilizzatore non obbliga *Scilab* ad interrompere l'esecuzione del comando selezionando, dalla finestra della console, il comando *Annulla* dalla tendina *Controllo*. Questo significa che *il criterio di arresto scelto per la procedura è inefficace*. In base all'analisi della procedura che utilizza il tipo *numero reale*, il criterio di arresto risulta *efficace*, ovvero la costruzione delle successioni viene interrotta *in ogni caso* dopo un numero finito di iterazioni. Per capire come mai questo non accade in *Scilab*, occorre analizzare la procedura trasformata che utilizza il tipo *numero in virgola mobile e precisione finita*.

Da un'osservazione più attenta delle righe scritte dalla procedura nella console si nota che dopo 53 iterazioni il valore dell'ampiezza (apparentemente) *non cambia* ed è maggiore di *delta*.

## Esercizi

Per l'uso del comando `printf` negli esercizi 1, 2 e 3 supponiamo di utilizzare *Scilab* su un calcolatore che ha un sistema operativo *diverso* da Windows.

1. Per determinare la scrittura in base dieci della precisione di macchina  $u = 2^{-53}$  riportata nella seconda pagina dell'Esercitazione 1, osserviamo che la scrittura posizionale in base dieci di  $u$  ha 53 cifre dopo il punto decimale. Infatti:  $10^{53}u$  è un numero *intero* (si ha  $10^{53}u = 5^{53}2^{53}u = 5^{53}2^{53}2^{-53} = 5^{53}$ ) mentre  $10^{52}u$  *non* è un numero *intero* (si ha  $10^{52}u = 5^{52}2^{52}u = 5^{52}2^{52}2^{-53} = 5^{52}2^{-1}$ ).

Prima parte dell'esercizio: Utilizzare la pagina di *help* relativa al termine *printf\_conversion* per determinare il valore di  $n$  da inserire nel comando `printf('\nu = %.nf', u)` in modo che, dopo l'assegnamento  $u = 2^{-53}$ , venga stampata nella console una stringa contenente la scrittura posizionale in base dieci di  $u$ :

```
u = 0.00000000000000011102230246251565404236316680908203125
```

Sappiamo che  $u \approx 1.11 \cdot 10^{-16}$ , dunque le prime 15 cifre dopo il punto decimale della scrittura posizionale in base dieci di  $u$  valgono 0. Allora  $u \in F(10, 53 - 15) = F(10, 37)$ .

Seconda parte dell'esercizio: Utilizzare la pagina di *help* relativa al termine *printf\_conversion* per determinare il valore di  $n$  da inserire nel comando `printf('\nu = %.ne', u)` in modo che venga stampata nella console la stringa:

```
u = 1.1102230246251565404236316680908203125e-16
```

2. Mostrare, ragionando come nell'esercizio precedente, che la scrittura posizionale in base dieci di  $\sigma(1)$ , il successore di 1 in  $F(2, 53)$ , ha 52 cifre dopo il punto decimale. Posto:

```
s1 = nearfloat('succ', 1)
```

utilizzare il comando `printf` per ottenere, nella console, la stringa:

```
s1 = 1.0000000000000002220446049250313080847263336181640625
```

Decidere infine se  $\sigma(1) \in F(10, 52)$ .

3. Siano  $x = \beta^b 0.c_1c_2 \dots$  ed  $x^* = \beta^b 0.c_1^*c_2^* \dots$  due numeri reali positivi *con lo stesso esponente in base  $\beta$* . Se  $c_1 = c_1^*, \dots, c_n = c_n^*$  allora:

$$|x - x^*| < \beta^{b-n}$$

(Infatti, posto  $t = 0.c_1 \dots c_n$  si ha:  $|x - x^*| = \beta^b |(t + \beta^{-n} 0.c_{n+1} \dots) - (t + \beta^{-n} 0.c_{n+1}^* \dots)| = \beta^{b-n} |0.c_{n+1} \dots - 0.c_{n+1}^* \dots|$ . Ma  $|0.c_{n+1} \dots - 0.c_{n+1}^* \dots| \leq \max\{0.c_{n+1} \dots, 0.c_{n+1}^* \dots\} < 1$ .)

Prima parte dell'esercizio: Dimostrare che si ha anche:

$$\frac{|x - x^*|}{|x|} < \beta^{1-n}$$

Seconda parte dell'esercizio: Utilizzare il comando `printf` per ottenere, nella console, le scritture posizionali di `z4` e `z8` in base dieci:

$$z4 = 1.414215087890625 \quad \text{e} \quad z8 = 1.4142135642468929290771484375$$

Terza parte dell'esercizio: Verificare, utilizzando la prima delle disuguaglianze ricavate sopra, che:

$$|z4 - \sqrt{2}| < 10^{-5} \quad \text{e} \quad |z8 - \sqrt{2}| < 10^{-8}$$

4. Modificare opportunamente la procedura `Bisezione` per realizzare una procedura di intestazione

```
function [z,v,ampiezza,iter] = Bisezione(f,a,b,delta)
```

che restituisce, oltre all'approssimazione `z` e `v = f(z)`, l'ampiezza dell'ultimo intervallo costruito e il numero di iterazioni eseguite.

5. Modificare opportunamente la procedura `Bisezione` per realizzare una procedura di intestazione

```
function [z,v] = Bisezione(f,a,b,epsilon)
```

che realizza il metodo di bisezione che utilizza, dato un numero reale positivo  $\epsilon$ , il criterio d'arresto di tipo relativo:

$$m_k = \min\{|a_k|, |b_k|\} \quad , \quad \text{se } \frac{\text{mis } I_k}{m_k} < \epsilon \text{ allora arresta la costruzione}$$

La procedura *deve* accertarsi anche che l'intervallo iniziale *non* includa zero. Discutere poi l'esecuzione dei seguenti assegnamenti:

```
[z, v] = Bisezione(fun,0,2,1e-5) \quad , \quad [z, v] = Bisezione(fun,1,2,1e-5)
```

e, posto prima `epsilon = 10-10` poi `epsilon = 10-16`:

```
[z, v] = Bisezione(fun,1,2,epsilon)
```

In ciascun caso, verificare se il valore restituito da *Scilab* soddisfa la richiesta di accuratezza dell'utilizzatore.