3.11 Zero di una funzione

Per esercitarci con le strutture iterative e i numeri reali prendiamo in considerazione il problema del calcolo dello zero di una funzione continua *f(x)* con il cosiddetto metodo *dicotomico*. Ricordiamo che si dice *zero* di *f* un numero x_0 tale che $f(x_0)=0$.

Sia *f(x)* una funzione continua che negli estremi dell'intervallo [*a,b*] assume valori di segno discorde, ovvero uno negativo e uno positivo e quindi tale che *f(a)*f(b)*<0, come mostrato in Figura 3.2. Sia *m*=(*a*+*b*)/2 il punto medio dell'intervallo; se *f(m)*=0 abbiamo ottenuto lo zero cercato, altrimenti si considera:

- l'intervallo [*a,m*] se *f(a)* e *f(m)* hanno segno discorde;
- l'intervallo [*m,b*] se *f(a)* e *f(m)* hanno segno concorde.

Si itera quindi lo stesso procedimento nell'intervallo appena determinato e si prosegue fino a trovare uno zero di *f*.

È importante osservare come per la corretta soluzione del problema sia indispensabile che *f(a)* e *f(b)* abbiano segno discorde. È quindi opportuno, prima di effettuare i calcoli relativi alla determinazione dello zero, controllare la validità della condizione nel segmento [*a,b*] preso in esame. Non è difficile implementare quest'algoritmo usando, come esempio, la funzione

$f(x) = 2x^3-4x+1$

considerando che *f*(0)=+1 e f(1)=-1 sono di segno discorde e quindi l'intervallo [0,1] soddisfa le ipotesi fatte. Il programma è quello fornito nel Listato 3.8, dove utilizziamo la funzione `fabs` che, come la funzione `abs`, calcola il valore assoluto di un numero, ma può essere applicata a valori di tipo `float`.

```
/* Determina lo zero della funzione f(x) = 2x^3-4x+1 */

#include <stdio.h>
#include <math.h>

#define ERR 0.001
```

```
main()
{
float a, b, m;
float  fa, fb, fm;
char x;

/* controllo validità a, b */
do {
  printf("Inserire a: ");
  scanf("%f", &a);
  printf("Inserire b: ");
  scanf("%f", &b);
  fa = 2*a*a*a-4*a+1; /* Calcolo della funzione per x=a */
  fb = 2*b*b*b-4*b+1; /* Calcolo della funzione per x=b */
}
while(fa*fb>0);

/* calcolo zero f */
do {
  m  = (a+b)/2;
  fm = 2*m*m*m-4*m+1;   /* Calcolo della funzione per x=m */
  if(fm!=0) {
    fa = 2*a*a*a-4*a+1; /* Calcolo della funzione per x=a */
    fb = 2*b*b*b-4*b+1; /* Calcolo della funzione per x=b */
    if(fa*fm<0) b=m; else a=m;
    fm = 2*m*m*m-4*m+1;  /* Calcolo della funzione per x=m */
  }
}
while(fabs(fm) > ERR);

printf("Zero di f in %7.2f\n", m);
}
```

Listato 3.8 Programma per il calcolo dello zero di una funzione

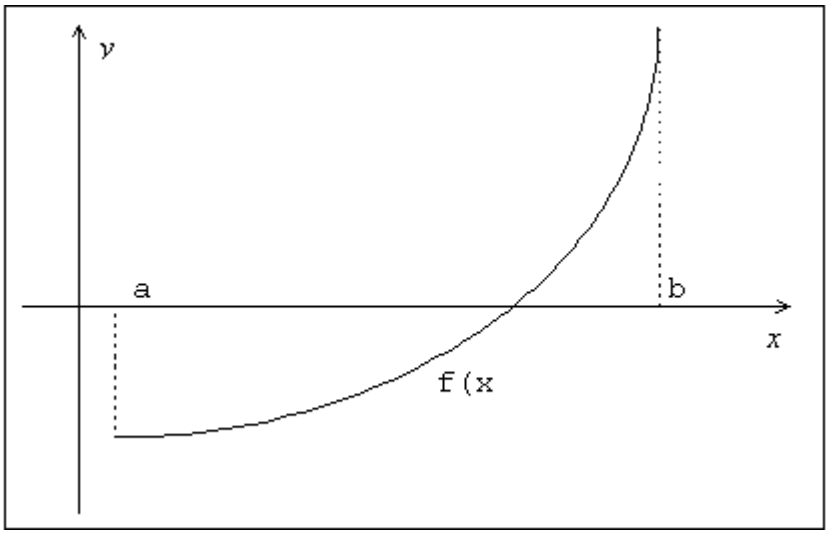

Figura 3.2 La funzione continua f assume segno discorde negli estremi dell'intervallo [a,b]

È importante osservare che a causa delle approssimazioni effettuate dalla macchina nell'esecuzione dei calcoli in genere non si ottiene uno zero effettivo della funzione ma solo una sua buona approssimazione. Per questa ragione nel `do-while` che controlla il ciclo di ricerca dello zero la condizione di controllo è `fabs(fm)>ERR` e non `fm=0`.

Effettuando il confronto con un errore ancora più piccolo (per esempio ERR=`1E-10`) si migliora il livello di approssimazione anche se questo può richiedere un tempo di calcolo molto maggiore. Abbiamo usato la funzione `fabs` che calcola il valore assoluto di un numero reale così come abbiamo visto `abs` fare di un intero.

Anticipiamo comunque che, una volta acquisite le conoscenze necessarie per creare noi stessi degli specifici sottoprogrammi, è possibile risolvere questo problema in modo molto più brillante e conciso.

Giustifichiamo infine il termine con cui è noto questo tipo di ricerca dello zero di una funzione ricordando che l'aggettivo dicotomico deriva da una parola greca che significa dividere a metà.