Lezione2
Ricerca di zeri

http://idefix.mi.infn.it/~palombo/didattica/Lab-TNDS/CorsoLab/LezioniFrontali

Fernando Palombo

Aritmetica Finita nel Computer

Nel computer l'aritmetica è a precisione finita cioè opera solo con numeri rappresentati da un numero finito di cifre perché ha a disposizione un numero finito di bit: oggi tipicamente per i numeri reali 32 (short) e 64 (long).

Ad un intero si assegna un numero massimo p di cifre. Quindi un intero n può essere rappresentato solo se $|n| < 2^p - 1$ (numerazione binaria) altrimenti si ha errore di overflow . Con gli interi l'aritmetica è esatta, sempre che non si abbiano degli overflow e tenendo presente che nel rapporto il risultato è un intero e si trascura il resto (7/2 = 3)

Numero reale a (float): $a = p\ 2^q$. 2 è la base della numerazione, p la mantissa e q l'esponente ($2^{-1} \leq |p| < 1$ normalizzazione)

Viene associato un numero finito t di bit per la mantissa e un numero finito r di bit per l'esponente.

Errori di Arrotondamento

Se il numero r di bit a disposizione per l'esponente non è sufficiente, allora si ha un errore di overflow o underflow.

Se la mantissa del numero reale ha bisogno di un numero di bit maggiore di t, allora la mantissa viene accorciata ed il numero reale è approssimato col numero di macchina più vicino (questo viene fatto o per arrotondamento o per troncamento).

2^{-t} granularità della mantissa (separazione in mantissa dei numeri rappresentabili dalla macchina).

ε_m precisione della macchina: la più piccola quantità che sommata ad 1 dà un numero diverso da 1. La sua conoscenza è fondamentale per scegliere le tolleranze e valutare i risultati.

x= numero reale, F(x) sua rappresentazione nell'aritmetica finita del computer, allora **|x-F(x)|** = errore di arrotondamento (round-off)

Errori di Arrotondamento

L'arrotondamento fa si che possano non valere le proprietà aritmetiche:
- Proprietà associativa dell'addizione e moltiplicazione: **(a+b)+c ≠ a+(b+c)**
- Proprietà distributiva della moltipl. rispetto alla somma**: a*(b+c) ≠ a*b +a*c**
- Legge di semplificazione**: (a*b)/b ≠ a**
-La somma può non produrre effetti su un addendo se questo è $1/\varepsilon_m$ ordini di grandezza più grande dell'altro.

Cancellazione numerica: sottrazione di numeri grandi e circa uguali porta a perdita di cifre significative. Esempio: a= 0.1234566789 e b=0.123456555 Se possiamo utilizzare solo le prime sette cifre della mantissa, si ha: a = 0.1234567 e b = 0.1234565 e quindi a-b = 0.2000000 10^{-6} (in notazione scientifica) invece che a-b = 0.2340000 10^{-6} (perdendo due cifre significative!)

Ogni operazione aritmetica introduce un errore di arrotondamento che via via **si propaga e si amplifica**. Ciò può portare ad un risultato finale completamente sbagliato.

Errore di Troncamento

Questo errore dipende unicamente dal tipo di problema trattato e dalla tecnica di approssimazione numerica adottata. Tipico esempio è il numero di termini di una serie per approssimare una funzione. Migliore è l'approssimazione della funzione, minore è l'errore commesso nel troncamento.

L'errore di troncamento è controllato completamente dall'analista!

Si può ripetere il calcolo con diversi livelli di approssimazione e se i risultati cambiano in maniera non accettabile, vuol dire che probabilmente l'errore di troncamento non è accettabile. Esistono comunque diversi modi per stimare l'errore di troncamento.

Questo errore sarebbe presente anche in presenza di aritmetica esatta.

Controllare sempre che l'accuratezza (o la precisione) richiesta dei risultati non sia persa a causa di questi errori di arrotondamento e troncamento

Derivazione Numerica

Calcolo numerico della derivata. Semplice concettualmente ma calcolo molto delicato e da fare con la massima cautela.

Il perché è semplice:

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \qquad \frac{df(x)}{dx} = \frac{\Delta f(x)}{\Delta x}$$

Diverse sorgenti di errori : arrotondamento nel calcolo di f (x) e del rapporto incrementale. Inoltre h molto piccolo rispetto ad x e tendente a zero (non più rappresentabile dalla macchina che lo approssima) e questo dà problemi:
a=10 , $\Delta x = 0.0000005000$ → x+ h = 10.0000009537
(x+h) – x = 0.0000009537 (~doppio del previsto !!)

Bisogna forzare il sistema perche assegni un incremento h che sia esattamente rappresentabile dalla macchina. Questo si può fare così:

Derivazione Numerica

float x, h, temp;
x= …; h = …; //Eventualmente definire temp come volatile
……
temp = x + h;
h= x – temp in questo modo x+ h e x differiscono per una quantità esattamente rappresentabile dalla macchina.

Detto ciò nel calcolo del rapporto incrementale restano gli errori di arrotoòndam. del rapporto incrementale e gli errori di troncamento nel modello matematico del calcolo della derivata.

Errore di arrotondamento: $\varepsilon_r \approx \varepsilon_f \, | f(x)/h |$ dove ε_f è l'errore relativo con cui è calcolato il valore della funzione f(x). In molti casi si può assumere $\varepsilon_f = \varepsilon_m$

L'errore di troncamento è dovuto all'aver trascurato il termine del secondo ordine nello sviluppo della serie di Taylor:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + h^2 \frac{f''(x_0)}{2!} + \cdots + h^n \frac{f^{(n)}(x_0)}{n!} + \cdots$$

Derivazione Numerica

Errore di troncamento ε_t dell'ordine di h^2 f''(x)

h va scelto in modo da minimizzare l'errore complessivo $\varepsilon = \varepsilon_r + \varepsilon_t$

Si potrebbe mostrare che in questo tipo di calcolo della derivata l'errore relativo che si commette va come la radice quadrata di ε_m (insufficiente in molti casi!!).

Vediamo di fare un primo miglioramento prendendo incrementi finiti centrali:

$$\frac{df(x)}{dx} \approx \frac{f(x+h/2) - f(x-h/2)}{h}$$

Si può far vedere che in questo caso si ha un miglioramento nell'errore relativo del calcolo della derivata che va come $\varepsilon_f^{2/3}$. Il calcolo della derivata migliora (anche di un ordine di grandezza in singola precisione) ma resta spesso inadeguata!).

Estrapolazione di Richardson

Esistono diverse tecniche per migliorare il calcolo numerico delle derivate. Ora ne presentiamo una che appare anche in diversi altri contesti (come la quadratura numerica): quella nota come estrapolazione di Richardson.

Non si tratta di estrapolazione vera e propria: qui si cerca di approssimare il risultato di una formula a differenza finita al limite che l'intervallo tenda a zero. Supponiamo di conoscere la funzione e le sue derivate nel punto x_0 e scriviamo gli sviluppi in serie di Taylor nei punti $x_0 + kh$ e $x_0 - kh$:

$$f(x_0 + kh) = f(x_0) + khf'(x_0) + (kh)^2 \frac{f''(x_0)}{2!} + (kh)^3 \frac{f^{(3)}(x_0)}{3!} + \cdots$$

$$f(x_0 - kh) = f(x_0) - khf'(x_0) + (kh)^2 \frac{f''(x_0)}{2!} - (kh)^3 \frac{f^{(3)}(x_0)}{3!} + \cdots$$

Estrapolazione di Richardson

Sottraiamo la seconda dalla prima e calcoliamo esplicitamente per i casi k=1, 2

$$\begin{cases} f(x_0+h)-f(x_0-h) & = 2hf'(x_0)+2h^3f^{(3)}(x_0)/6+\cdots+R_1(h^5) \\ f(x_0+2h)-f(x_0-2h) & = 4hf'(x_0)+16h^3f^{(3)}(x_0)/6+\cdots+R_2(h^5) \end{cases}$$

Ciò che non è esplicitato va come h^5. Se si elimina da queste equazioni la derivata terza e si risolve per la derivata prima si ha:

$$f'(x_0)=\frac{f(x_0-2h)-8f(x_0-h)+8f(x_0+h)-f(x_0+2h)}{12h}+O(h^4)$$

Si noti che l'errore sul calcolo della derivata va come h^4. Il calcolo della derivata prima necessita del calcolo della funzione f(x) in quattro punti. Questo algoritmo converge molto rapidamente. Si può fare meglio aumentando il numero di punti su cui si calcola f(x).

Comunque calcolare numericamente una derivata solo se non se ne può fare a meno!

Derivate con un Polinomio Interpolante

Una funzione complicata può essere approssimata mediante un polinomio interpolante. Questo lo vedremo nella prossima lezione.

Allora la derivata della funzione in un punto x_0 può essere calcolata utilizzando un polinomio che interpola la funzione nell'intorno di x_0 e calcolando in x_0 la derivata del polinomio interpolante.

Radici di una Funzione

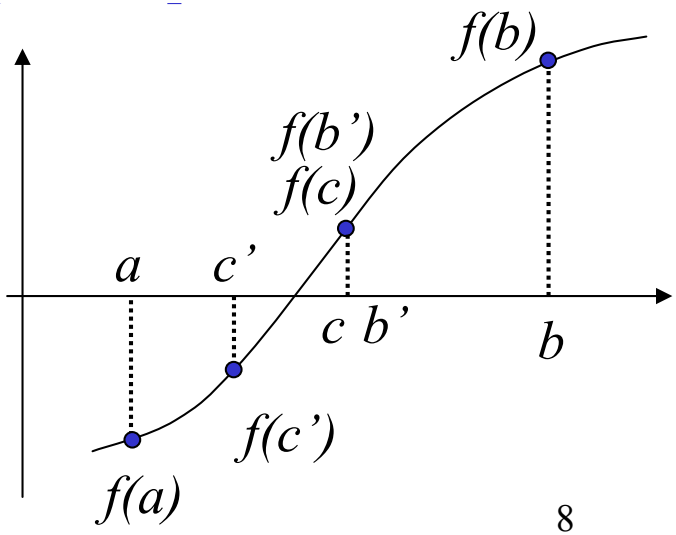
Ci occupiamo qui in generale di problemi non lineari dove le tecniche di ricerca degli zeri sono di tipo iterativo. Nei problemi lineari ci sono tecniche anche dirette (tipo la regola di Cramer per i sistemi di equazioni lineari a coefficienti costanti).

Come si può immaginare, a parte i casi molto semplici, il problema di trovare le radici di una funzione è un problema molto complesso e cercare le radici senza avere alcuna idea di dove si possano trovare è un problema ancora più arduo. Noi ci limitiamo qui a funzioni continue sull'intervallo considerato e ad una dimensione.

Quindi bisogna cercare di individuare una regione dove si ritiene ci possa essere uno (o più) zeri. Per fare questo possiamo ad esempio calcolare la funzione in una griglia di punti ed individuare un intervallo $[a_0, b_0]$ all'interno del quale la funzione cambi segno: **$\text{sign}(f(a_0) \bullet \text{sign}(f(b_0) < 0$** (bracheting)
Questo intervallo (detto di incertezza) sicuramente contiene almeno uno zero
(Teorema di Bolzano)

Metodo della bisezione

Tra i metodi classici che si basano sull'intervallo di incertezza quello più semplice è quello della bisezione
Indichiamo con α lo zero vero e con α_a il suo valore approssimato trovato dall'algoritmo. Se ε è la nostra tolleranza sull'errore nella misura di α, allora dovrà aversi : $|\alpha - \alpha_a| \leq \varepsilon$

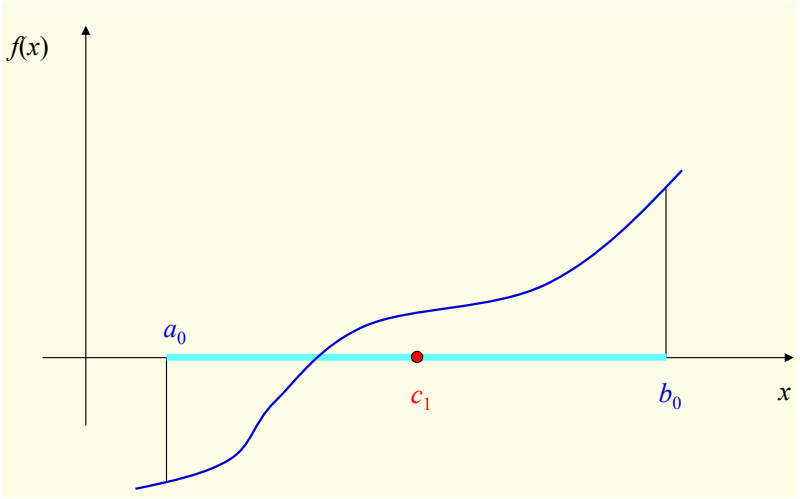


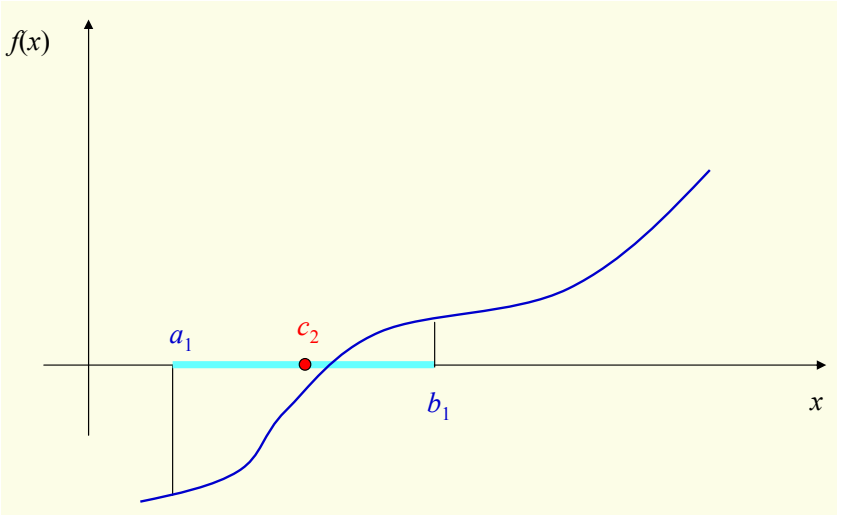
8

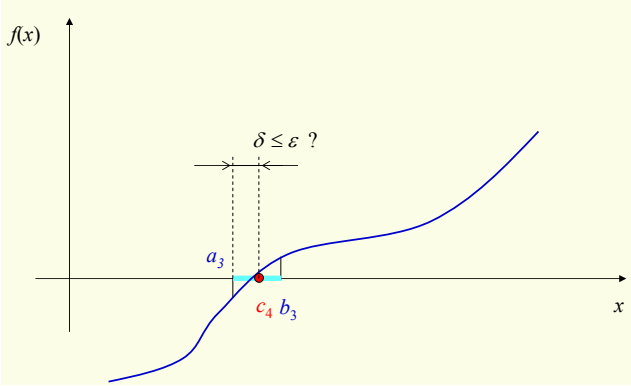
Dividiamo in due l'intervallo [a, b] all'interno del quale sappiamo esserci almeno uno zero: $c = a + (b-a)/2$
Se c dista da a o b per meno o uguale ad ε allora, allora c è lo zero cercato, altrimenti si scarta la metà dell'intervallo nel quale la funzione non cambia segno e la procedura si ripete nell'altra metà di intervallo. Si suddivide il sottointervallo in due, ecc. Alla fine sicuramente si trova lo zero con la desiderata risoluzione.

Un semplice ragionamento per induzione mostra che dopo n cicli di bisezione si ha che $|\alpha - c_n| \leq (b_n - a_n) \leq \varepsilon \leq (1/2)^n (b - a)$
→ # di iterazioni $\geq [\log(b - a) - \log\varepsilon]/\log 2$ per la tolleranza ε

Metodo della bisezione







$b_1 = c_1$ $a_1 = a_0$ per l'esempio in figura
e poi iterativamente sino ad $[a_n,b_n]$
quando il controllo sulla tolleranza blocca l'iterazione

Metodo della bisezione

Per esempio se a=1.6 e b=4.5 e vogliamo una risoluzione di 0.00005, allora saranno necessarie 16 bisezioni.

Questo metodo non può fallire. Se ci sono più zeri nell'intervallo considerato il metodo ne trova solo uno. Se c'è una singolarità, il metodo converge ad essa.

Permette di ottenere la precisione voluta (che però non confronta con la precisione della macchina). Questo controllo va fatto a monte.

Questo metodo è relativamente lento se confrontato con altri metodi in parte perché ad ogni iterazione il confronto lo fa sul segno della funzione ma ignora il suo modulo (che è una parte importante di informazione)

Accorgimenti

La condizione di appartenenza all'intervallo di incertezza meglio farla considerando il segno della funzione agli estremi dell'intervallo e non moltiplicando i valori delle funzioni (che potrebbe portare a zero a causa della precisione finita!)

Calcolare il valore medio come c = a + (b-a)/2 e non come c = (a+b)/2

Controllare che non si facciano richieste di precisione impossibili. Quando l'intervallo di incertezza diventa minore della precisione della macchina, la larghezza dell'intervallo viene messa uguale a zero e l'intervallo non viene più cambiato.

Va messo un controllo sul numero massimo di iterazioni accettate perché la procedura potrebbe entrare in un loop.

Applicazioni ricerca degli zeri: Bisezione

Vediamo ora alcune applicazioni pratiche. Il codice è solo di esemplificazione didattica e non è ottimizzato.

Potete trovare il codice a questo link:

http://www.mi.infn.it/~palombo/didattica/Lab-TNDS/CorsoLab/Applicazioni-Web/Zeri/Bisezione/

Scaricatelo sul vostro PC ed utilizzatelo.

Velocità di Convergenza

Una sequenza x_i si dice che converge **linearmente** alla soluzione α se esiste una costante c con $0 < c < 1$ e un intero $n \geq 0$ tali per cui:

$$|x_{n+1} - \alpha| \leq c\,|x_n - \alpha|$$

Nel metodo della bisezione il tipo di convergenza è lineare ($c = 0.5$)

Una sequnza x_i converge **superlinearmente** alla soluzione α se, per qualche sequenza c_n che converge a zero, esiste un intero $n \geq 0$ tale per cui: $|x_{n+1} - \alpha| \leq c_n\,|x_n - \alpha|$

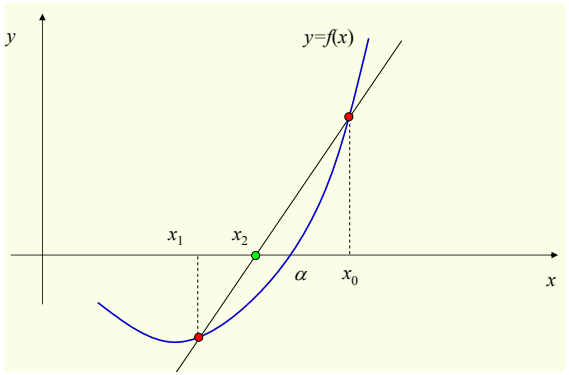
Una sequenza x_i si dice convergente con **ordine $p \geq 1$** alla soluzione α se per qualche $c > 0$ esiste un intero $n \geq 0$ tale per cui:

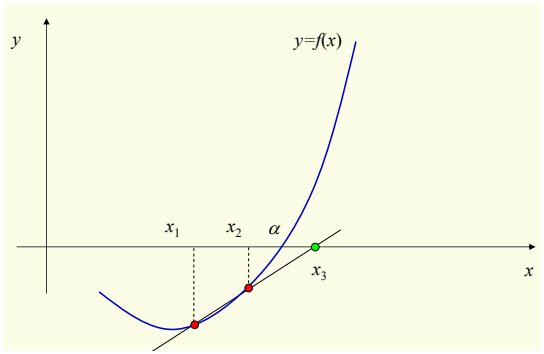
$$|x_{n+1} - \alpha| \leq c\,|x_n - \alpha|^p$$

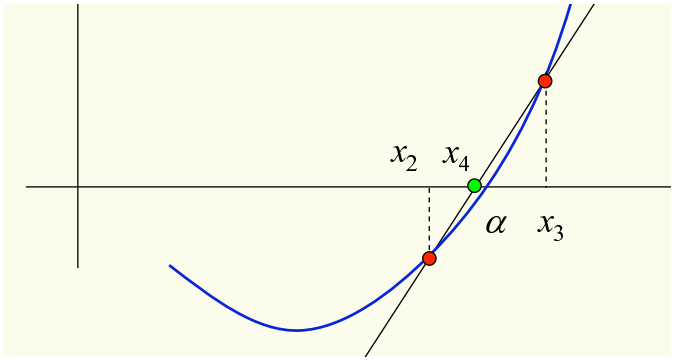
Metodo della Secante

Si applica quando la funzione nella regione dello zero può essere approssimata da una retta. Siano x_0 e x_1 due punti sufficientemente vicini alla radice e siano noti $f(x_0)$ e $f(x_1)$. Si considera la retta per questi due punti. La sua intersezione con l'asse x fornisce l'ascissa x_2 della nuova approssimazione della radice.

Una nuova secante viene fatta passare tra questo punto $(x_2, f(x_2))$ e e il punto dell'approssimazione precedente (c'è però una ambiguità nella prima iterazione).







Metodo della Secante

Importante: in questo metodo l'intersezione della secante può avvenire fuori dell'intervallo di incertezza.

La formula ricorsiva per questo metodo è:

$$x_{k+1} = x_k - f(x_k)\,(x_k - x_{k-1})/[f(x_k)-f(x_{k-1})]$$

Si può dimostrare che la convergenza è di tipo: $|x_{k+1} - \alpha| \approx M|x_k - \alpha|^{\phi}$

$\phi = \frac{\sqrt{5}+1}{2} \approx= 1.618$ e $M = \frac{f''(\alpha)}{2f'(\alpha)}$

Convergenza **di ordine 1.618**. Questo metodo converge più velocemente del metodo della bisezione.

Attenzione però perché se i punti iniziali non sono abbastanza vicini alla soluzione, l'algoritmo talvolta potrebbe non convergere.

Applicazioni del Metodo della Secante

Vediamo ora una applicazione di questo metodo. Il codice si trova a questo link:

http://idefix.mi.infn.it/~palombo/didattica/Lab-TNDS/CorsoLab/Applicazioni-Web/Zeri/Secante

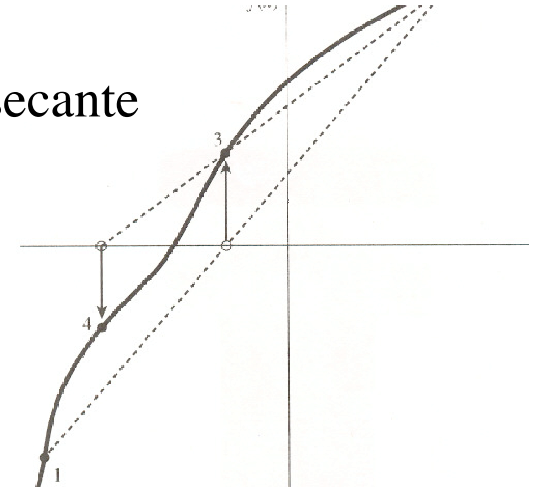
Scaricatelo sul vostro PC ed utilizzatelo.

Metodo della Regula Falsi (Falsa Posizione)

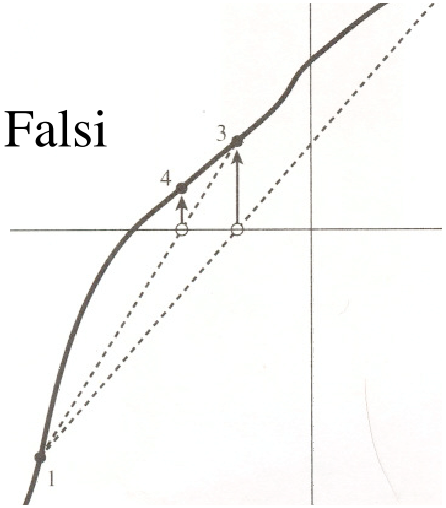
Questo metodo (di Fibonacci) di fatto è una variante del metodo delle secanti con questa sola differenza: in questo metodo la secante deve sempre definire un intervallo di incertezza (e non come nel metodo delle secanti che la nuova approssimazione può avvenire fuori dall'intervallo di incertezza).

La secante viene fatta passate tra l'ultima approssimazione e quella precedente. Se in punto intercettato sulla curva è fuori dall'intervallo di incertezza, allora la secante è fatta passare tra l'ultima approssimazione e la più recente tra quelle precedenti che rende definibile un intervallo di incertezza.

secante



Regula Falsi



Metodo della Regula Falsi (Falsa Posizione)

La formula ricorsiva è quella del metodo delle secanti

$$x_{k+1} = x_k - f(x_k)\,(x_k - x_{k-1})/[f(x_k)-f(x_{k-1})]$$

forzando però la secante a conservare l'intervallo di incertezza:

Se $\text{sign}(f(x_{k+1})\ \text{sign}(f(b_k) < 0 \rightarrow \quad a_{k+1} = x_{k+1}$ e $b_{k+1} = b_k$

altrimenti $\rightarrow \quad a_{k+1} = a_k$ e $b_{k+1} = x_{k+1}$

Questo metodo è più lento di quello delle secanti ma garantisce la convergenza alla soluzione