Metode Numerice – Lucrarea nr. 3

REZOLVAREA NUMERICĂ A ECUAȚIILOR ALGEBRICE ȘI TRANSCENDENTE – PARTEA II

Modelul matematic şi metodele numerice utilizate

Metoda bisecției (a înjumătățirii intervalului)

Metoda bisecţiei este una din cele mai simple metode iterative de rezolvare a unei ecuaţii neliniare care se bazează pe observaţia că funcţia $f(x)$ are semne contrare la capetele intervalul $[a,b]$ în interiorul căruia se află soluţia, adică $f(a)\cdot f(b)<0$. Se consideră deci ecuaţia de forma $f(x)=0,\ x\in[a,b]$, pentru care s-a separat în prealabil o soluţie în intervalul $[a,b]$, adică $f(a)\cdot f(b)<0$. Ştiind că funcţia este continuă pe intervalul $[a,b]$, se cere să se determine soluţia în cauză. Caracteristic metodei bisecţiei este că, pornind de la intervalul $[a,b]$, la fiecare pas se restrânge domeniul în care se caută soluţia, prin înjumătăţirea intervalului de la pasul anterior, până la atingerea preciziei dorite. Aplicarea repetată a înjumătăţirii intervalului de definiţie a unei funcţii, corespunzătoare unei ecuaţii a cărei soluţie trebuie determinată, conduce la scăderea gradului de incertitudine al localizării soluţiei. Această repetare se realizează în contextul probării unor condiţii şi a schimbării adaptate a capetelor intervalului. Ideea se transpune general după următorul algoritm:

Pasul 1. Se iniţializează intervalul $[a;b]=[a_0;b_0]$;

Pasul 2. Se pornește procesul iterativ $k=0$;

Pasul 3. Înjumătăţirea intervalului $c_{k+1}=a_k+\frac{1}{2}\cdot(b_k-a_k)$;

Pasul 4. Dacă $f(c_{k+1})\cdot f(a_k)<0 \Rightarrow a_{k+1}=a_k; b_{k+1}=c_{k+1}$;

Pasul 5. Dacă $f(c_{k+1})\cdot f(b_k)<0 \Rightarrow a_{k+1}=c_{k+1}; b_{k+1}=b_k$;

Pasul 6. Se incrementează $k=k+1$ şi se reia Pasul 3.

Paşii enumeraţi anterior reprezintă un algoritm scris în pseudolimbaj, pe baza căruia poate fi implementat uşor un program de calcul. Limitele intervalului aflat în restrângere în jurul soluţiei sunt notate cu a_k respectiv b_k. Se observă că la fiecare valoare a lui k se schimbă una din limitele intervalului, după cum condiţia de existenţă a soluţiei este verificată de prima sau a doua jumătate a intervalului partiţionat. Vizualizarea schimbării acestor limitele se arată în Fig. 1. Oricât de mult ar fi dusă restrângerea intervalului în jurul soluţiei, există posibilitatea ca valoarea considerată drept soluţie să nu fie cea adevărată. Procesul iterativ de înjumătăţire nu poate continua la infinit, ci trebuie oprit după un anumit număr de partiţionări determinat prin stabilirea unei erori limită între valoarea determinată ca şi soluţie şi valoarea adevărată x.

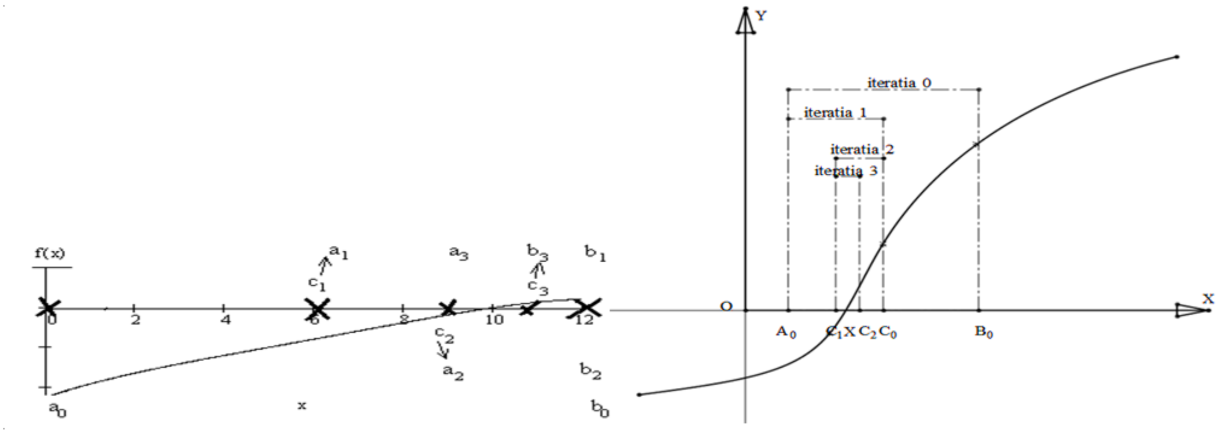


Fig. 1: *Metoda Bisecţiei*

Eroarea de aproximare a soluției, la iterația k, este mai mică sau egală cu lungimea intervalului de căutare de la iterația k: $\varepsilon = |x - c_k| \le (b_k - a_k)$, unde x este soluția exactă, a_k, b_k, c_k, sunt punctele de limită, respectiv de mijloc la iterația k. Lungimea intervalului de căutare se înjumătățește la fiecare iterație $(b_k - a_k) = \frac{1}{2}(b_{k-1} - a_{k-1})$, astfel, eroarea de aproximare a soluției la iterația k devine: $\varepsilon = (1/2)^k \cdot (b_0 - a_0)$.

Notând cu ε o limită de eroare absolută impusă şi cu n numărul până la care a ajuns iteraţia *k* se poate determina o măsură destul de precisă a numărului minim de iteraţii necesare pentru calcularea unei soluţii aproximative care să se încadreze într-o limită de eroare cunoscută:

$$n \ge \frac{\log(b-a) - \log(\varepsilon)}{\log(2)} \tag{1}$$

Astfel, în algoritmul prezentat mai sus, este posibilă adăugarea unei condiţii de oprire a procesului când se atinge o anumită precizie, adică după ce se efectuează un număr de n iteraţii. Implementarea în limbaj de programare a acestei metode trebuie să se concretizeze în aşa fel încât la fiecare iteraţie, în afară de prima, evaluarea funcţiei să fie făcută o singură dată.

Metoda lui Newton (tangentei)

Metoda Newton este una dintre cele mai eficiente metodele iterative de rezolvare a ecuaţiilor neliniare. Fie o ecuaţie de forma $f(x) = 0$, cu variabila $x \in [a, b]$ pentru care s-a separat în prealabil o soluţie în intervalul $[a, b]$, adică $f(a) \cdot f(b) < 0$. Funcţia este continuă şi de două ori derivabilă pe intervalul dat şi se doreşte determinarea soluţiei în cauză. Dezvoltarea în serie Taylor în jurul unui punct x_k, în cazul în care se reţin doar primii doi termeni ai dezvoltării şi restul, arată de forma:

$$f(x) = f(x_k) + (x - x_k) \cdot f'(x_k) + \frac{1}{2!} \cdot (x - x_k)^2 \cdot f''(\xi_k) \quad cu \ \xi_k \in [x; x_k] \tag{2}$$

Înlocuirea în expresia de mai sus, a unei aproximaţii x_{k+1} succesivă lui x_k, în locul lui x, pentru care presupunem că se anulează *f(x)*, neglijând restul (apare o eroare de metodă datorată neglijării restului seriei Taylor care se bazează pe un calcul recursiv) şi impunând (alegând) o aproximaţie iniţială a soluţiei x_0, ne conduce la o relaţie iterativă de determinare a soluţiei, care evidenţiază faptul că la fiecare iteraţie graficul funcţiei este aproximat cu tangenta dusă în punctul $(x_k, f(x_k))$:

$$f(x)=0 \Rightarrow x = x_k - \frac{f(x_k)}{f'(x_k)} - \frac{1}{2!}\cdot(x-x_k)^2 \cdot f''(\xi_k)/f'(x_k) \tag{3}$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{4}$$

Metoda lui Newton poate fi interpretată geometric ca o trasare repetată a tangentelor la funcţie, prin punctele de aproximare a soluţiei, până când tangenta intersectează punctul în care funcţia se anulează, ori un punct delimitat cu o anumită precizie de soluţia exactă. Ecuaţia dreptei tangentă în acest punct este:

$$y - f(x_k) = f'(x_k)\cdot(x - x_k) \tag{5}$$

Iar prin intersecţia ei cu axa Ox, deci pentru $y=0$ se obţine relaţia care coincide cu relaţia (4):

$$x = x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{6}$$

Din acest motiv metoda lui Newton este cunoscută şi sub numele de metoda tangentelor.

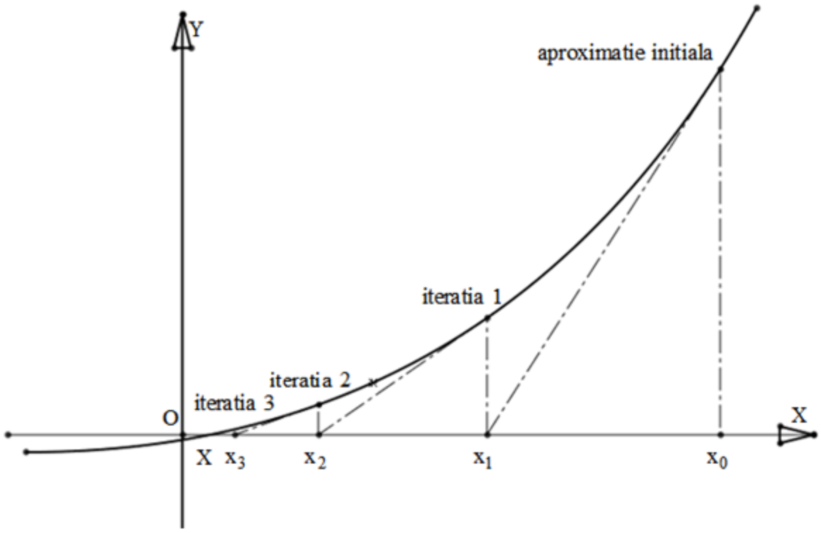


Fig. 2: *Trasarea repetată a tangentelor la funcţie*

Procesul se continuă până când se ajunge, în limita unei precizii, la soluţia considerată optimă. Algoritmul metodei se poate sintetiza astfel:

Pasul 1. Se iniţializează soluţia cu valoarea x_0 (aproximaţia iniţială la iteraţia 0);

Pasul 2. La un pas oarecare *k*, $k = 1, 2, 3...$, al procesului iterativ de calcul se determină $f(x_k)$ şi $f'(x_k)$, noua valoare a soluţiei rezultând din relaţia $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$;

Pasul 3. Calculul se consideră terminat când la impunerea unei abateri maxim admisibilă a aproximaţiei iniţiale faţă de soluţia adevărată, respectiv la impunerea unei erori de aproximare, este îndeplinită condiţia identificării unui număr minim de iteraţii necesare localizării soluţiei cu o anumită precizie impusă:

$$n \geq \log_2 \left[\frac{\log(\varepsilon) + \log(M)}{\log(M \cdot \varepsilon_{admisibil})} \right] \tag{7}$$

Valoarea acestui număr este condiţionată de abaterea admisibilă impusă şi de definirea numerică a raţiei M.

$$M = \frac{\max_{x \in I}\left(f''(x)\right)}{2 \cdot \min_{x \in I}\left(f'(x)\right)} < \infty \tag{8}$$

Forme îmbunătățite a Metodei lui Newton

Pentru a accelera convergenţa iterativă a soluţiei, pentru a creşte precizia, sau pentru a reduce efortul de calcul din metoda iniţială a lui Newton au fost dezvoltate o serie de forme îmbunătăţite ale acesteia:

Metoda Newton – Kantorovici (metoda tangentelor paralele)

În cazul în care evaluarea derivatei funcţiei, în fiecare nou punct de aproximare, este costisitoare ca timp de calcul, formula lui Newton poate fi adaptată, în sensul reţinerii valorii calculate a derivatei doar în primul punct de aproximare pe parcursul unui număr de iteraţii:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_0)} \tag{9}$$

Se reduce simţitor efortul computaţional, cu dezavantajul reducerii concomitente a convergenţei; sunt necesare mai multe iteraţii până la obţinerea unei soluţii dorite.

Metoda Newton discretă (metoda secantelor)

Dacă expresia derivatei funcţiei nu este cunoscută, atunci metoda lui Newton şi varianta sa simplificată nu pot fi aplicate. În acest caz se face apel la aproximarea numerică a derivatei după care se deduce relaţia iterativă de determinare a soluţiei:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \approx x_k - f(x_k) \cdot \left[\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} \right] \tag{10}$$

Pentru aplicarea acestei metode trebuie cunoscute (alese) primele două aproximaţii iniţiale x_0 şi x_1 (pot fi alese de exemplu capetele intervalului $x_0 = a,\ x_1 = b$).

Pentru ca metoda secantei să conveargă la o soluție, trebuie să se îndeplinească următoarele condiții:

$$\begin{gathered} f(a) \cdot f(b) < 0 \\ f(x_0) \cdot f''(x_0) > 0 \\ f(x_0) \cdot f(x_1) < 0 \end{gathered} \tag{11}$$

Observaţii: În funcţie de tipul aplicaţiei abordate trebuie să se realizeze un compromis între efortul de calcul implicat pentru atingerea unei precizii, convergenţă şi precizie, dificultatea de evaluare repetată a funcţiei şi a derivatelor ei. Trebuie să se facă apel pentru fiecare caracteristică aleasă la una din formulele expuse mai sus, ori la o formulă care să valorifice orientat sporirea preciziei, a convergenţei, ori reducerea timpului de calcul.

Metoda lui Halley

Pentru a obține o convergență mai rapidă decât cea a metodei lui Newton se folosește o dezvoltare în serie Taylor a funcției atașate ecuației studiate, folosind primele 3 componente:

$$f(x) = f(x_0) + f'(x_0)\cdot(x - x_0) + \frac{1}{2}\cdot f''(x_0)\cdot(x - x_0)^2 \tag{12}$$

De aici se poate ajunge la formula iterativă de calcul:

$$x_{k+1} = x_k - \frac{2\cdot f(x_k)}{f'(x_k) - \sqrt{f'(x_k)^2 - 2\cdot f(x_k)\cdot f''(x_k)}} \tag{13}$$

Instrumente folosite

Algoritmul general al metodei bisecției (înjumătățirii intervalului)

Relația (14) reprezintă algoritmul general de calcul al metodei bisecției. Se observă că algoritmul se folosește de instrucțiuniile de programare *for* și *if*, prezentate în **Lucrarea 2**. Indicii *k*, *k+1* și 0 se introduc utilizând paranteza pătrată („**[**”), deoarece variabilele *a*, *b*, respectiv *c* sunt variabile de tip vector.

$$\begin{array}{l|l} sol := & \text{for } k \in 0..n \\ & \quad a_0 \leftarrow A_0 \\ & \quad b_0 \leftarrow B_0 \\ & \quad c_{k+1} \leftarrow \dfrac{a_k + b_k}{2} \\ & \quad a_{k+1} \leftarrow \text{if}\left(f(a_k)\cdot f(c_{k+1}) < 0, a_k, c_{k+1}\right) \\ & \quad b_{k+1} \leftarrow \text{if}\left(f(b_k)\cdot f(c_{k+1}) < 0, b_k, c_{k+1}\right) \\ & \quad A_0 \wedge B_0 \\ & c \end{array} \tag{14}$$

\\ se apeeaza ciclul "for"
\\ se initializeaza marginile primului interval in care se afla solutia
\\ formula de aproximare a solutiei
\\ modificarea la fiecare iteratie a capetelor intervalului aflat in restrangere
\\ memorarea in vectori a acestor margini
\\ algoritmul returneaza vectorul aproximatiilor solutiei, care sunt de fapt injumatatirile intervalelor.

Operatorul de derivare

Algoritmul metodei Newton presupune utilizarea operației de derivare. Pentru utilizarea operatorului de derivare din *Mathcad*, este necesară apelarea toolbar-ului *Calculus*. Dacă toolbar-ul *Calculus* nu este vizibil, din meniul principal se bifează *View – Toolbars – Calculus* (a se vedea 3). Din toolbar-ul *Calculus* se selectează *Derivative.*

În fereastra de comandă se introduce numele funcției urmat de un apostrof (`) după care se introduce un operator de atribuire (:=). În dreapta operatorului de derivare se introduce funcția de derivat.

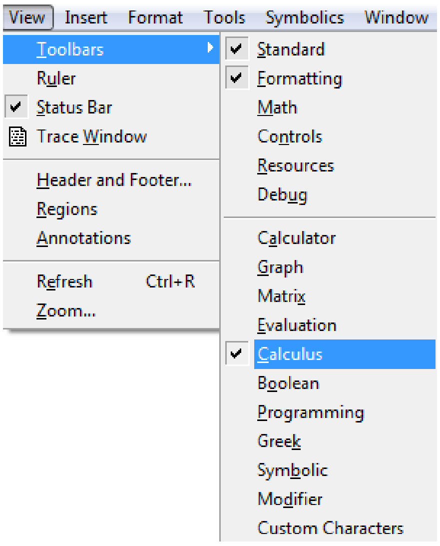


Fig. 3: *Activarea toolbar-ului* Calculus

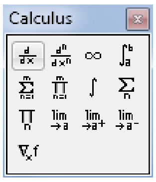


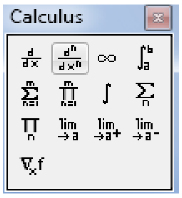
Fig. 4: *Inserarea operatorului de derivare*

$$f'(x) := \frac{d}{dx} f(x) \qquad (15)$$

Operatorul de derivare de ordin superior

Algoritmul metodei secantei presupune utilizarea operației de derivare de ordinul 2. Din toolbar-ul *Calculus* se selectează *Nth Derivative* (a se vedea Fig. 5)

În fereastra de comandă se introduce numele funcției urmat de doi apostrofi (``) după care se introduce un operator de atribuire (:=).



$$f''(x) := \frac{d^2}{dx^2} f(x) \qquad (16)$$

Fig. 5: *Inserarea operatorului de derivare de ordin superior*