

METODE NUMERICE DE CALCUL A ECUAȚIILOR ALGEBRICE NELINIARE ȘI TRANSCENDENTE

Basoc Daniela

cl. a XII-a “B”

Probleme de modelare matematică proceselor mecanice și a situațiilor economice

nu pot fi rezolvate fără aplicarea metodelor de *calcul numeric*. Un specialist modern trebuie să cunoască bine metode de bază ale *matematicii aplicate* din care face parte și *analiza numerică*.

Practic, orice teoria inginerescă fără un suport matematic solid nu are nici o valoare științifică. Ca regulă, un inginer operează cu datele numerice, care trebuie să fie prelucrate într-un anumit mod pentru calculul și proiectarea dispozitivelor tehnice.

Ecuațiile matematice din domeniul mecanic, ca regulă, conțin derivate și integrale, iar cele din domeniu finanțiar modern utilizează, de exemplu, teoria ecuațiilor diferențiale stohastice, care nu pot fi rezolvate direct în mod analitic, ci numai prin *metode aproximative* cu care operează *analiza numerică*. Astfel, *metode numerice* de calcul sunt metode aproximative, iar disciplina aparține domeniului matematicii aplicate.

Programa analitică a cursului cuprinde studierea principalelor metode de analiză numerică, care prin intermediul computerelor devin tot mai precise și mai des utilizate, și anume:

- rezolvare numerică a ecuațiilor algebrice nelineare și transcendente;
- rezolvare sistemelor de ecuații algebrice liniare și nelineare;
- aproximare și interpolare funcțiilor algebrice complexe sau prezentate în formă tabulară (de tabele);
- determinare funcțiilor analitice pentru o bază de date numerice;
- derivare și integrare numerică .

Sub noțiunea de ecuații algebrice *neliniare* se înțelege acele ecuații care conținne cunoscută la puterea diferită de unu, de exemplu:

$$y = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \dots + a_n \cdot x^n = \sum_{i=0}^n a_i \cdot x^i, n \neq 1$$

Sub noțiunea de ecuații algebrice *transcendente* se înțeleg acele ecuații din care nu poate fi obținută soluție analitică în mod evident, de exemplu:

ecuațiile iraționale de tipul

$$y = \frac{\sqrt{8 \cdot (1-x)}}{1 + \sqrt{1-x}}$$

ecuațiile exponențiale

$$y = a_0 \cdot e^{b_0 \cdot x} + a_1 \cdot e^{b_1 \cdot x} + \dots + a_n \cdot e^{b_n \cdot x} = \sum_{i=0}^n a_i \cdot e^{b_i \cdot x}$$

ecuațiile logaritmice

$$y = \sum_{i=0}^n a_i \cdot \log_{b_i} x, i = 0, 1, \dots, n$$

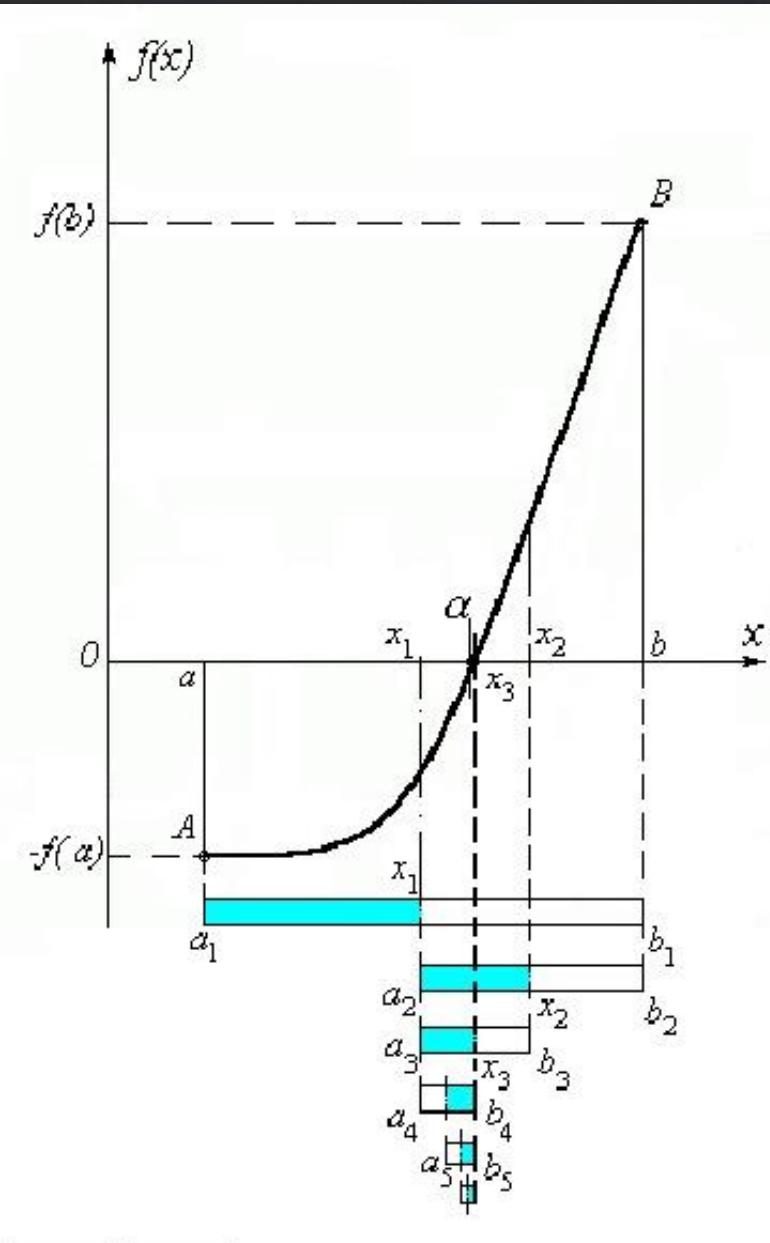
sau cele trigonometrice

$$y = a_0 + \sum_{k=1}^n (a_k \cdot \cos kx + b_k \cdot \sin kx), k = 0, 1, \dots, n$$

Pentru rezolvarea ecuațiilor neliniare și transcendente există următoarele metode principale : metoda *înjumătățirii intervalului* (*metoda bisecției*), metoda *coardei*, metoda *tangentei* (sau metoda *Newton*) .

Metoda BISECȚIEI

Este cea mai simplă și totodată cea mai sigură metoda de rezolvare numerică a ecuațiilor nelineare și transcențe. Singurul ei dezavantaj reprezintă un ritm lent de convergență, care cere un număr mare de operații matematice. Metoda are ca baza teorema *valorii intermediare* din analiza matematică și are ca o idee reducerea progresivă a intervalului de examinare a funcției date prin înjumătățirea pentru a localiza rădăcina căutată. Deci, fie o funcție $f(x)$ continuă pe un interval finit $a < x < b$, care are două valori $f(a)$ și $f(b)$ definite astfel că produsul $f(a) \cdot f(b) < 0$. În acest caz funcția $f(x)$ trebuie să aibă cel puțin o rădăcina, iar metoda localizării poate fi prezentată grafic în felul următor (fig. 1.1). Cum se vede reducerea progresivă a intervalului se face prin înjumătățirea lui și utilizarea punctului de mijloc drept limită nouă pentru intervalul urterior.



- ◊ Exemplul 1: Să se determine o rădăcină a functiei $x^4 + 2x^3 - x - 1 = 0$ pe segmentul $[0, 1]$ pentru 16 divizări consecutive. Deoarece numărul de aproximări succesive este fixat, iar extremitățile segmentului cunoscute, atribuirile se realizează nemijlocit în program.

```
◊ program cn05;
◊ var  a,b,c: real;
◊      i,n:integer;
◊ function f(x:real):real;
◊ begin f:=sqr(sqr(x))+2*x*sqr(x)-x-1; end;
◊ begin a:=0; b:=1; n:=16;
◊      for i:=1 to n do
◊          begin c:=(b+a)/2;
◊              writeln('i=',i:3,' x=',c:10:8,' f(x)=',f(c):12:8);
◊              if f(c)=0 then break else if f(c)*f(a)>0 then a:=c else b:=c;
◊          end;
◊      end.
```

- ❖ Rezultate: i= 1 x=0.50000000 f(x)= -1.18750000
- ❖ i= 2 x=0.75000000 f(x)= -0.58984375 ...
- ❖ i= 15 x=0.86679077 f(x)= 0.00018565
- ❖ i= 16 x=0.86677551 f(x)= 0.00009238

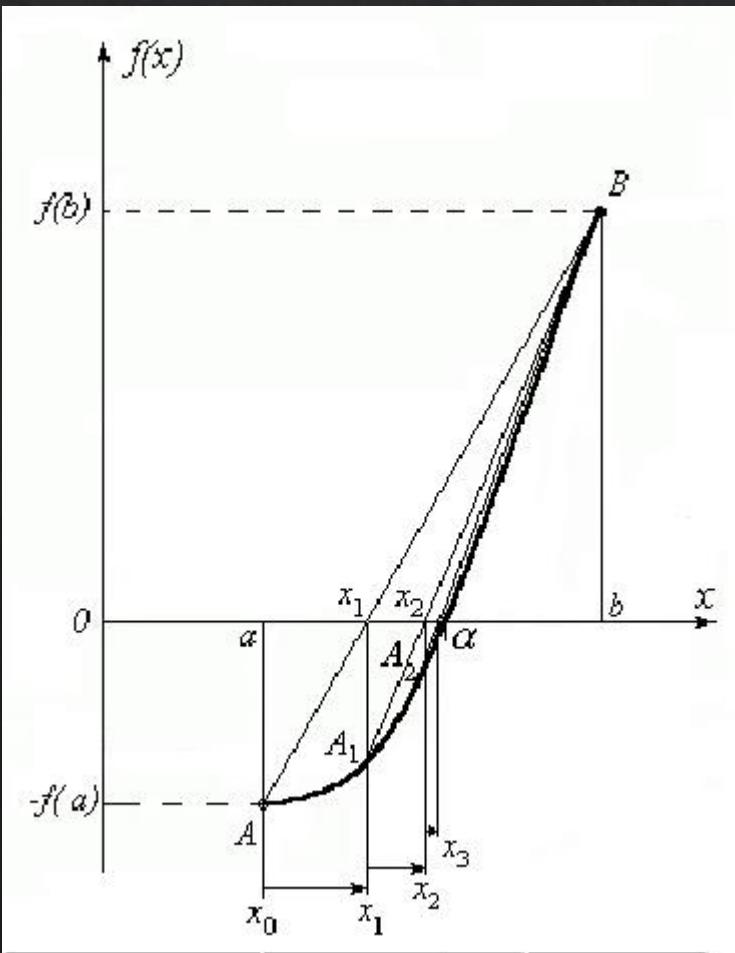
- ◆ Exemplul 2: Să se determine o rădăcină a ecuației $6\cos(x) + 8\sin(x) = 0$ pe segmentul $[2, 4]$ cu precizia $\varepsilon=0,00017$.

```
◆ program cn06;
◆ var a,b,c,eps:real;
◆ function f(x:real):real;
◆ begin f:=6*cos(x)+8*sin(x); end;
◆ begin a:=2; b:=4; eps:=0.00017;
◆     repeat c:=(b+a)/2;
◆         writeln('x=',c:10:8,' f(x)=',f(c):12:8);
◆         if f(c)=0 then break else
◆             if f(c)*f(a)>0 then a:=c else b:=c;
◆         until abs(b-a) <eps;
◆ end.
```

- ❖ Rezultate:
- ❖ $x=3.00000000$ $f(x)= -4.81099492$
- ❖ $x=2.50000000$ $f(x)= -0.01908454$
- ❖ ...
- ❖ $x=2.49829102$ $f(x)= -0.00199471$
- ❖ $x=2.49816895$ $f(x)= -0.00077401$

Metoda COARDELOR

- ◆ Metoda constă în inlocuirea funcției date $f(x)$, continua pe un interval $[a,b]$, printr-o dreapta $g(x)$ care trece prin extremitățile A și B definite astfel că $f(a) \cdot f(b) < 0$. Rădăcina precisă α a funcției date $f(x)$ se aproximeaza prin punctul de intersectie adreptei $g(x)$ cu axa ox (fig. 1.2.)



Exemplul 1: Fie dată funcția $f(x) = \ln(xsinx)$. Să se calculeze soluția aproximativă a ecuației $f(x) = 0$ pe segmentul $[0,5; 1,5]$ pentru 10 aproximări successive, utilizând metoda coardelor.

Pentru acest exemplu, preprocessarea matematică nu este necesară. Deoarece numărul de aproximări succesive este fixat, iar extremitățile segmentului cunoscute, atribuirile respective vor fi realizate direct în corpul programului.

```
◊ program cn07;
◊ Var  a,b,e,c,x: real;
◊      n,i: integer;
◊  function f(x:real):real;
◊  begin f:=ln(x*sin(x));end;
◊  begin a:=0.5; b:=1.5; n:=10;
◊          {determinarea extremitatii fixe e si a aproximarii initiale x0}
◊          c:=(a-(f(a)))/(f(b)-f(a))*(b-a);
◊          if f(c)*f(a)>0 then begin e:=b; x:=a; end
◊          else begin e:=a; x:=b; end;
◊          {calculul iterativ al solutiei}
◊          for i:=1 to n do
◊              begin x:= x-(f(x))/(f(e)-f(x))*(e-x);
◊                  writeln(x:10:8,' ',f(x):12:8); end;
◊          end.
```

- ❖ Rezultate:
- ❖ $i = 1 \ x = 1.27995775 \ f(x) = 0.20392348$
- ❖ $i = 2 \ x = 1.18251377 \ f(x) = 0.09028687$
- ❖ ...
- ❖ $i = 9 \ x = 1.11427651 \ f(x) = 0.00016577$
- ❖ $i = 10 \ x = 1.11420523 \ f(x) = 0.00006678$

Exemplul 2: Fie data functia $f(x) = x^4 - 3x^2 + 7.5x - 1$. Să se calculeze soluția aproximativă a ecuației $f(x) = 0$ pe segmentul $[-0,5; 0,5]$ cu exactitatea $\varepsilon = 0,0001$, utilizînd metoda coardelor.

Pentru funcția dată pe $[-0,5; 0,5]$ M_1 și m_1 sînt, respectiv, egale cu 10 și 5. Pentru simplitate, atribuirile necesare vor fi realizate direct în corpul programului.

program cn08;

```
var Msup,minf,a,b,e,x,xnou,xvechi,eps: real;
```

```
function f(x:real):real;
```

```
Begin
```

```
  f:=sqr(sqr(x))-3*sqr(x)+7.5*x-1;
```

```
end;
```

```
begin
```

```
  a:=-0.5; b:=0.5; eps:=0.0001;
```

```
  Msup:=10; minf:=5;
```

```
{determinarea extremitatii fixe si a aproximarii initiale}
```

```
  x:=a-(f(a))/(f(b)-f(a))*(b-a);
```

```
  if f(x)*f(a)>0 then begin e:=b; xnou:=a; end
```

```
  else begin e:=a; xnou:=b; end;
```

```
{calculul iterativ al solutiei}
```

```
  repeat
```

```
    xvechi:=xnou;
```

```
    xnou:= xvechi-(f(xvechi))/(f(e)-f(xvechi))*(e-xvechi);
```

```
    writeln(' x=',xnou:10:8,' f(x)=',f(xnou):12:8);
```

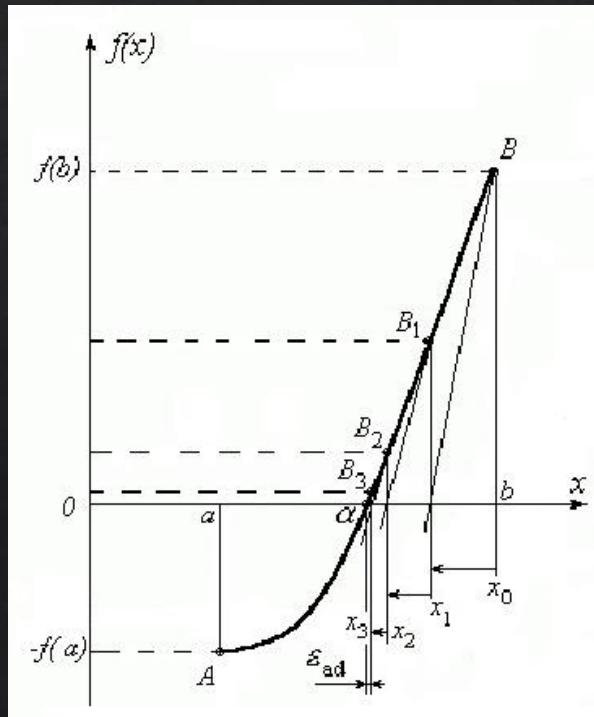
```
  until abs((Msup-minf)/minf*(xnou-xvechi))<eps;
```

End.

- ❖ Rezultate:
- ❖ $x=0.22500000 f(x)= 0.53818789$
- ❖ $x=0.15970438 f(x)= 0.12191694$
- ❖ ...
- ❖ $x=0.14130134 f(x)= 0.00026052$
- ❖ $x=0.14127062 f(x)= 0.00005579$

Metoda NEWTON

- ◆ Fie o ecuație neliniară sau trascendentă de forma $f(x)=0$ și fie ca funcția $f(x)$ are pe intervalul $[a,b]$ o singură rădăcina reală, iar prima derivate $f'(x)$ și cea a două $f''(x)$ sunt continue și nu mențin semnul constant în intervalul dat. Metoda constă în aproximarea rădăcinii precise α cu abscisa punctului de intersecție a tangentei cu axa Ox , care este dusă la curba $f(x)$ în punctul k cu coordonatele $\{x_k, f(x_k)\}$ alese în mod corespunzător.
- ◆ Altfel spus, arcul de curbă $f(x)$ se înlocuiește cu o tangentă la curbă într-un punct k care se deplasează în direcția rădăcini α (fig 1.3):



Exemplul 13 : Fie dată funcția $f(x) = x^3 - 2x^2 + x - 3$. Să se scrie un program care va calcula soluția ecuației $f(x) = 0$ pe segmentul $[2; 15]$ pentru 10 aproximări succesive, utilizând metoda Newton. Preprocesarea matematică: Se determină $f'(x)$:

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

Deoarece numărul de aproximări successive este fixat, iar extremitățile segmentului cunoscute, atribuirile necesare se vor realiza direct în corpul programului.

```
◊ program cn09;
◊ var    a, b, x, c : real;
◊      i, n: integer;
◊ function f(z:real):real;
◊      begin f:=z*z*z-2*z*z+z-3; end;
◊ function fd1(z:real):real;
◊      begin fd1:=3*z*z-4*z+1; end;
◊ begin a:=2.1; b:=15; n:=10; i:=0;
◊      c:=a-(f(a))/(f(b)-f(a))*(b-a);
◊      if f(c)*f(a)<0 then x:=a else x:=b; while i<n do
◊          Begin i:=i+1;
◊          x:=x-f(x)/fd1(x);
◊          writeln ('i=',i:2,' x=',x:15:12, ' f=',f(x):15:12);
◊      end;
◊ End.
```

- ❖ Rezultate:
- ❖ i= 1 x= 10.23214285700 f=869.11072454000
- ❖ i= 2 x= 7.06207637180 f=256.52261987000
- ❖ ...
- ❖ i= 9 x= 2.17455942470 f= 0.00000009329
- ❖ i=10 x= 2.17455941030 f= 0.00000000001

Exemplul 2: Fie dată funcția $f(x) = \cos^2(x) - \frac{x}{4}$. Să se scrie un program care va calcula soluția aproximativă a ecuației $f(x) = 0$ pe segmentul $[2,4; 3]$ cu exactitatea $\varepsilon = 0,0001$, utilizând metoda Newton. Pentru funcția dată pe segmentul $[2,4; 3]$ M_2 și m_1 sănt, respectiv, egale cu 2 și 0,03. Preprocesarea matematică:

$$f(x) = \cos^2(x) - \frac{x}{4}$$

$$f'(x) = -\sin(2x) - \frac{1}{4}$$

Deoarece ε este dat, extremitățile segmentului și valorile M_2 , m_1 – cunoscute, atribuirile vor fi realizate direct în program.

```

◊ program cn10;
◊ var a, b, xn, xv, M2, m1, e, c : real;
◊ function f(z:real):real;
◊     begin f:=cos(z)*cos(z)-z/4; end;
◊ function fd1(z:real):real;
◊     begin fd1:=-sin(2*z)-1/4; end;
◊ begin a:=2.4; b:=3; M2:=2; m1:=0.03; e:=0.0001;
◊     c:=(a-(f(a))/(f(b)-f(a))*(b-a));
◊     if f(c)*f(a)<0 then begin
◊         xn:=a; xv:=b;
◊     end
◊     else begin xn:=b; xv:=a; end;
◊     While M2*sqr(xn-xv)/(2*m1)>e do
◊     Begin xv:=xn;
◊         xn:=xv-f(xv)/fd1(xv)
◊         writeln('  x=‘, xn:15:12, ‘  f=‘, f(xn):15:12)
◊     End;
◊     End.

```

- ❖ Rezultate:
- ❖ $x = 2.47538619170$ $f = -0.00078052066$
- ❖ $x = 2.47646766320$ $f = -0.00000027700$
- ❖ $x = 2.47646804730$ $f = 0.00000000000$

Bibliografie

- ❖ <https://www.scribd.com/doc/13531262/METODE-NUMERICE-PENTRU-REZOLVAREA-ECUA%C5%A2ILOR-ALGEBRICE-NELINIARE-%C5%9EI-TRANSCENDENTE>
- ❖ [file:///C:/Users/User/Downloads/XII_Informatica%20\(in%20limba%20romana\).pdf](file:///C:/Users/User/Downloads/XII_Informatica%20(in%20limba%20romana).pdf)
- ❖ <https://www.codecogs.com/latex/eqneditor.php>