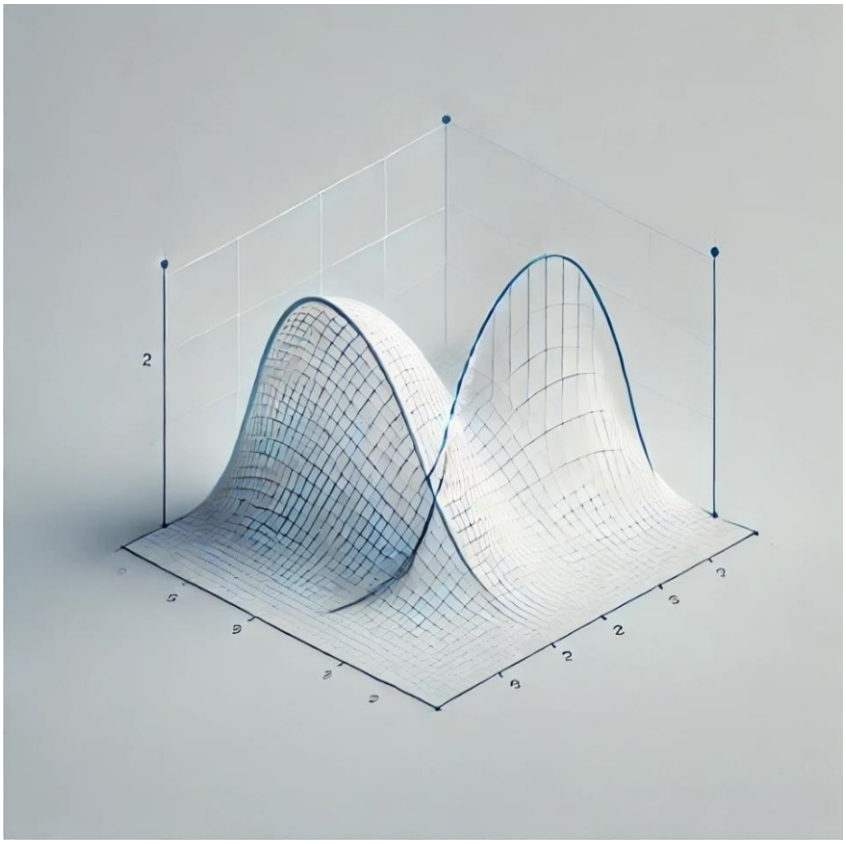
Ștefan - Marius BURU
Tudor MILCHIȘ

METODE NUMERICE

Aplicații și sinteze teoretice



U.T.PRESS
Cluj-Napoca, 2024
ISBN 978-606-737-747-7

Ștefan-Marius BURU
Tudor MILCHIȘ

METODE NUMERICE

Aplicații și sinteze teoretice

U.T.PRESS
Cluj-Napoca, 2024
ISBN 978-606-737-747-7

Editura U.T.PRESS
Str. Observatorului nr. 34
400775 Cluj-Napoca
Tel.: 0264-401.999
e-mail: utpress@biblio.utcluj.ro
www.utcluj.ro/editura

Recenzia: Prof.dr.ing. Cosmin G. Chiorean

Șl.dr.ing. Bianca R. Marton

Pregătire format electronic on-line: Gabriela Groza

ISBN 978-606-737-747-7

PREFAȚĂ

Prezenta lucrare reprezintă o introducere în studiul metodelor numerice aplicate și se adresează, în principal, studenților din domeniul ingineriei civile. Materialul acoperă programa analitică a cursului de Metode Numerice, din planul de învățământ al specializărilor Facultății de Construcții din cadrul Universității Tehnice din Cluj-Napoca și își propune să constituie un sprijin practic și teoretic pentru studenți. Această lucrare constituie atât un suport practic pentru activitățile de laborator, cât și o resursă utilă pentru studiul individual și înțelegerea noțiunilor teoretice fundamentale.

Materialul este structurat în trei capitole, fiecare abordând metode numerice fundamentale și tehnici de calcul esențiale pentru rezolvarea problemelor întâlnite în inginerie. Fiecare metodă discutată este prezentată într-o manieră accesibilă și organizată, cuprinzând patru secțiuni. Noțiunile teoretice oferă o prezentare concisă a principiilor fundamentale, a contextului matematic și a relațiilor esențiale necesare pentru înțelegerea și aplicarea metodei. Se continuă cu un cod sursă scris în sintaxă Matlab, care permite studenților să implementeze și să testeze algoritmii într-un limbaj de programare de curentă actualitate. Se prezintă apoi rezolvarea secvențială a unei aplicații, punându-se accent pe înțelegerea metodei de rezolvare și pe interpretarea rezultatelor. Ultima parte conține o serie de aplicații propuse pentru rezolvare în cadrul orelor de laborator sau pentru aprofundare individuală.

Autorii mulțumesc tuturor celor care au sprijinit realizarea acestui material și își exprimă speranța că lucrarea va fi utilă și apreciată de studenți, cadre didactice și profesioniști interesați de utilizarea metodele numerice în rezolvarea problemelor inginerești.

Cluj-Napoca, 2024 *Autorii*

CUPRINS

1 REZOLVAREA ECUAȚIILOR NELINIARE 7

1.1 Introducere 7

1.2 Localizarea soluțiilor unei ecuații neliniare de forma f(x)=0 10

1.3 Metoda bisecției pentru rezolvarea ecuațiilor neliniare de forma f(x)=0 12

1.3.1 Noțiuni teoretice 12

1.3.2 Aplicație Matlab 14

1.3.3 Aplicație rezolvată 16

1.3.4 Aplicații propuse 18

1.4 Metoda falsei poziții pentru rezolvarea ecuațiilor neliniare de forma f(x)=0 19

1.4.1 Noțiuni teoretice 19

1.4.2 Aplicație Matlab 21

1.4.3 Aplicație rezolvată 23

1.4.4 Aplicații propuse 24

1.5 Metoda secantei pentru rezolvarea ecuațiilor neliniare de forma f(x)=0 25

1.5.1 Noțiuni teoretice 25

1.5.2 Aplicație Matlab 27

1.5.3 Aplicație rezolvată 29

1.5.4 Aplicații propuse 31

1.6 Metoda Newton pentru rezolvarea ecuațiilor neliniare de forma f(x)=0 32

1.6.1 Noțiuni teoretice 32

1.6.2 Aplicație Matlab 35
1.6.3 Aplicație rezolvată 36
1.6.4 Aplicații propuse 38
1.7 Metoda punctului fix pentru rezolvarea ecuațiilor neliniare de forma x=g(x) 39
1.7.1 Noțiuni teoretice 39
1.7.2 Aplicație Matlab 43
1.7.3 Aplicație rezolvată 46
1.7.4 Aplicații propuse 48
2 REZOLVAREA SISTEMELOR DE ECUAȚII NELINIARE 49
2.1 Introducere 49
2.2 Metoda Newton 53
2.2.1 Noțiuni teoretice 53
2.2.2 Aplicație Matlab 54
2.2.3 Aplicație rezolvată 58
2.2.4 Aplicații propuse 60
3 REZOLVAREA SISTEMELOR DE ECUAȚII LINIARE 61
3.1 Introducere 61
3.2 Metoda eliminării Gauss 65
3.2.1 Noțiuni teoretice 65
3.2.2 Aplicație Matlab 67
3.2.3 Aplicație rezolvată 69
3.2.4 Aplicații propuse 72
3.3 Factorizarea Cholesky 73
3.3.1 Noțiuni teoretice 73
3.3.2 Aplicație Matlab 75

3.3.3 Aplicație rezolvată 77
3.3.4 Aplicații propuse 80
3.4 Metoda Jacobi 81
3.4.1 Noțiuni teoretice 81
3.4.2 Aplicație Matlab 82
3.4.3 Aplicație rezolvată 84
3.4.4 Aplicații propuse 87
3.5 Condiționarea sistemelor de ecuații liniare 88
3.5.1 Noțiuni teoretice 88
3.5.2 Aplicație Matlab 89
3.5.3 Aplicație rezolvată 90
3.5.4 Aplicații propuse 92

1

REZOLVAREA ECUAȚIILOR NELINIARE

1.1 Introducere

În cadrul acestui capitol se prezintă metode numerice pentru determinarea soluțiilor ecuațiilor neliniare de forma f(x)=0, respectiv x=g(x). Soluția unei ecuații de forma f(x)=0 se numește rădăcină. Notând o astfel de rădăcină cu α, are loc identitatea f(α)=0. Soluția unei ecuații de forma x=g(x) poartă denumirea de punct fix. Dacă un număr real β este punct fix al funcției g(x) atunci are loc identitatea β=g(β). Asupra acestei problematici se va reveni în cadrul metodei punctului fix.

O ecuație f(x)=0 este neliniară dacă nu poate fi redusă la o ecuație liniară de forma:

$$a \cdot x + b = 0 \tag{1.1}$$

Mai mult, o ecuație neliniară de forma f(x)=0 este algebrică dacă funcția f(x) este un polinom. În caz contrar, ecuația se numește transcendentă, iar în expresia ei pot interveni funcții trigonometrice, exponențiale, logaritmice, etc. Ecuațiile neliniare intervin frecvent în domeniul ingineriei civile și astfel soluționarea acestor tipuri de probleme devine esențială. În cazuri particulare (de ex. ecuații de gradul 2, anumite forme ale ecuațiilor de gradul 3, unele ecuații trigonometrice, etc.), ecuațiile neliniare pot fi rezolvate exact, folosind procedee analitice, însă în marea majoritate a ecuațiilor neliniare, soluțiile poate fi determinate doar numeric, folosind metode iterative de rezolvare. În continuare se prezintă o serie de aplicații din domeniul ingineriei civile care se reduc la rezolvarea unor ecuații neliniare. Aceste aplicații vor fi folosite ca exemple de test în cadrul metodelor numerice de rezolvare a ecuațiilor neliniare ce urmează a fi prezentate. Suplimentar, unde e cazul, se vor prezenta

ecuații neliniare formulate generic, pentru a evidenția diverse particularități care pot apărea pe parcursul procesului iterativ (divergență, proces staționar, soluții false, etc.).

Aplicația 1.

Fie rezervorul sferic din Figura 1. Cunoscând raza sferei (R) și nivelul apei (h), volumul de apă acumulat în rezervor poate fi calculat cu următoarea relație:

$$V = \pi \cdot h^2 \cdot \left(R - \frac{h}{3}\right) \qquad (1.2)$$

Știind că R=3 m, $\pi \approx 3.14159$, să se determine înălțimea coloanei de apă pentru care volumul de apă din rezervor este de 80 m^3.

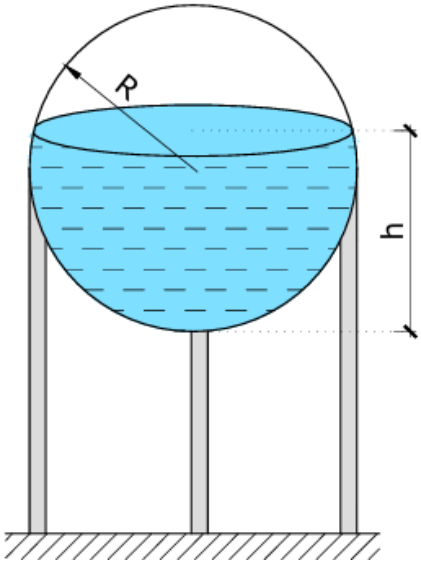


Figura 1

Aplicația 2.

Stâlpul din Figura 2 este solicitat la compresiune excentrică de forța concentrată P. Tensiunea extremă poate fi evaluată folosind următoarea relație:

$$\sigma_{max} = \frac{P}{A}\left(1 + \frac{e \cdot c}{\frac{I}{A}} \cdot \frac{1}{\cos\left(\frac{L}{2\sqrt{\frac{I}{A}}} \cdot \sqrt{\frac{P}{E \cdot A}}\right)}\right) \qquad (1.3)$$

unde:

- A – reprezintă aria secțiunii transversale;
- I – reprezintă momentul de inerție în raport cu axa după care are loc rotirea secțiunii;
- E – reprezintă modulul de elasticitate longitudinal al materialului.

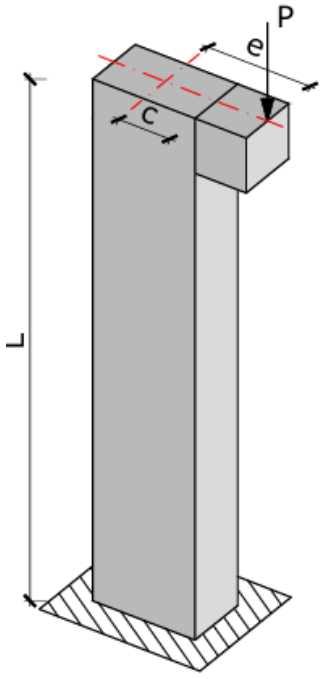


Figura 2

Se cere determinarea forței capabile P pentru care tensiunea maximă este 3 N/mm². Se cunosc: A=1125 cm², I=189500 cm⁴, E=30000 N/mm², c=22.5 cm, e=45cm.

Aplicația 3.

Ecuația axei deformate pentru grinda metalică de secțiune IPE300, reprezentată în figura de mai jos este:

$$w(x) = \frac{q}{120 \cdot E \cdot I \cdot L}(-x^5 + 2 \cdot L^2 \cdot x^3 - L^4 \cdot x) \tag{1.4}$$

unde:

- L – reprezintă lungimea grinzii;
- I – momentul de inerție în raport cu axa principală orizontală a secțiunii transversale;
- q – intensitatea maximă a sarcinii distribuite;
- E – reprezintă modulul de elasticitate longitudinal al materialului.

Să se determine valoarea maximă a săgeții. Se cunosc: E=210000 N/mm², L=10 m, q=100kN/m.

Obs.: Pentru determinarea secțiunii x din lungul elementului în care valoarea deplasării este maximă, trebuie să se calculeze în prealabil punctele de extrem ale funcției w(x), deci secțiunile x pentru care w'(x)=0.

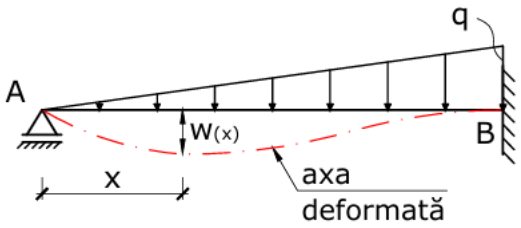


Figura 3

Aplicația 4.

Să se determine lățimea secțiunii transversale, b, a unui stâlp în consolă solicitat la încovoiere oblică spațială, știind că ecuația de dimensionare este:

$$235 \cdot b^3 - 0.9 \cdot b - 0.2 = 0 \tag{1.5}$$

1.2 Localizarea soluțiilor unei ecuații neliniare de forma f(x)=0

În vederea aplicării unui procedeu numeric pentru determinarea soluției/soluțiilor unei ecuații neliniare este necesară specificarea unor aproximații inițiale ale soluțiilor căutate. Așa cum se va vedea pe parcursul acestui capitol, unele metode au nevoie de o aproximație inițială, iar altele necesită două. Mai mult, cele două aproximații inițiale pot să încadreze soluția (de ex. în cazul metodei bisecției) sau pot fi de aceeași parte a ei (de ex. în cazul metodei secantei). Una dintre metodele cele mai folosite de localizare a soluțiilor unei ecuații neliniare și implicit a unor aproximații inițiale constă în inspectarea graficului funcției f(x) atașate ecuației. Reprezentarea grafică a variației funcției pe un anumit interval poate fi realizată într-o multitudine de aplicații dintre care amintim *Microsoft Excel* [1], *Matlab* [2] *GeoGebra* [3], *Desmos* [4], *WolframAlpha* [5].

x	f(x)
0	-80
0.3	-79.18
0.6	-76.8333
0.9	-73.1293
1.2	-68.2379
1.5	-62.3285
1.8	-55.571
2.1	-48.1348
2.4	-40.1897
2.7	-31.9054
3	-23.4513
3.3	-14.9973
3.6	-6.71293
3.9	1.232161
4.2	8.668311
4.5	15.42588
4.8	21.33521
5.1	26.22667
5.4	29.93061
5.7	32.27738
6	33.09734

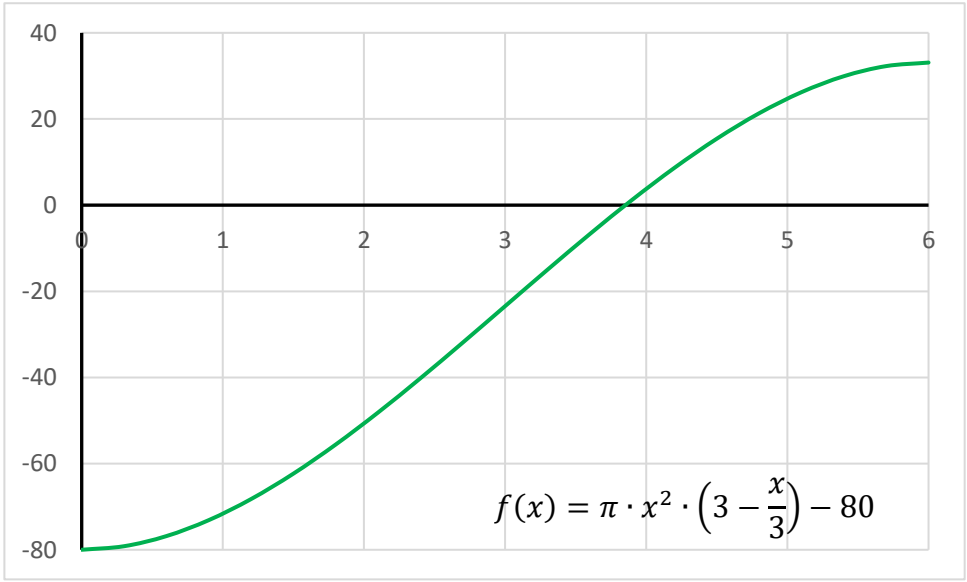


Figura 4. Grafic realizat în aplicația Microsoft Excel [1]

De exemplu, pentru Aplicația 1 descrisă anterior, graficul funcției atașate ecuației de forma f(x)=0 este prezentat în Figura 4 și a fost realizat în aplicația Microsoft Excel [1].

Spectrul de valori considerate pentru variabila problemei s-a considerat a fi în intervalul [0;6], definit de dimensiunile sferei.

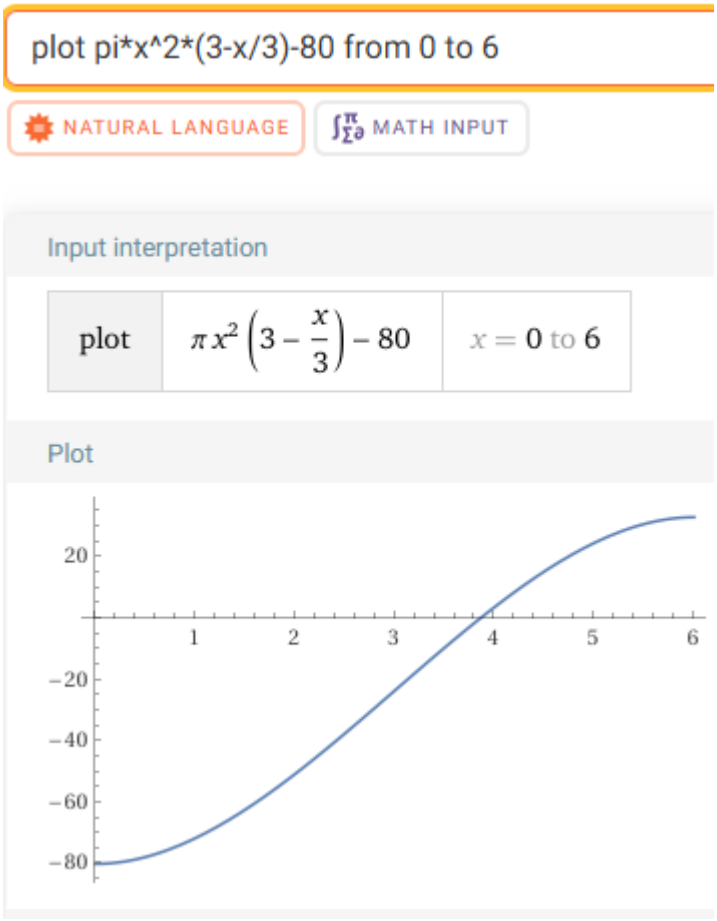


Figura 5. Grafic realizat online folosind WolframAlpha [5]

Analizând graficele prezentate anterior, se poate observa că soluția căutată (valoarea lui x pentru care funcția f(x) se anulează, deci punctul de intersecție dintre graficul funcției și axa absciselor) este localizată în intervalul [3; 4]. Din punct de vedere matematic, ecuația mai are două soluții reale, însă acestea nu sunt acceptabile (sunt înafara dimensiunilor sferei).

1.3 Metoda bisecției pentru rezolvarea ecuațiilor neliniare de forma f(x)=0

1.3.1 Noțiuni teoretice

Metoda bisecției sau metoda înjumătățirii intervalului se bazează pe observația conform căreia o ecuație de forma f(x)=0 are cel puțin o soluție localizată într-un interval închis [a; b] dacă funcția f(x) este continuă și ia valori de semne contrare la capetele intervalului. Algebric, existența a cel puțin unei soluții a ecuației f(x)=0 într-un interval [a; b] este asigurată dacă:

$$f(a) \cdot f(b) < 0 \tag{1.6}$$

Pentru ecuația reprezentată grafic in Figura 4, respectiv Figura 5, intervalul [3; 4] conține o soluție deoarece f(3)·f(4)<0.

Observații:

- dacă $f(a) \cdot f(b) > 0$, intervalul [a; b] poate să nu conțină nici o soluție a ecuației f(x)=0 (Figura 6) sau poate conține mai multe soluții (Figura 7).

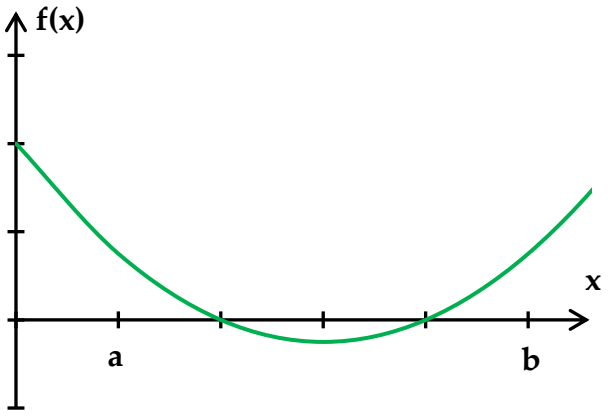


Figura 6. Interval [a; b] care conține 2 soluții.

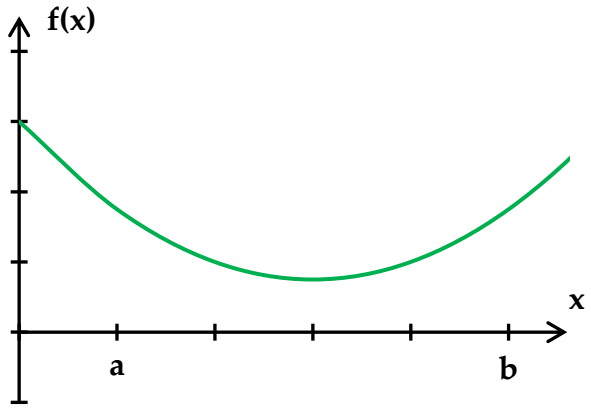


Figura 7. Interval [a; b] care nu conține nici o soluție.

- dacă $f(a) \cdot f(b) < 0$, intervalul [a; b] poate să conțină mai multe soluții (Figura 8).
- în continuare se va considera că intervalul [a; b] conține o singură rădăcină (notată cu α) a ecuației neliniare f(x)=0.

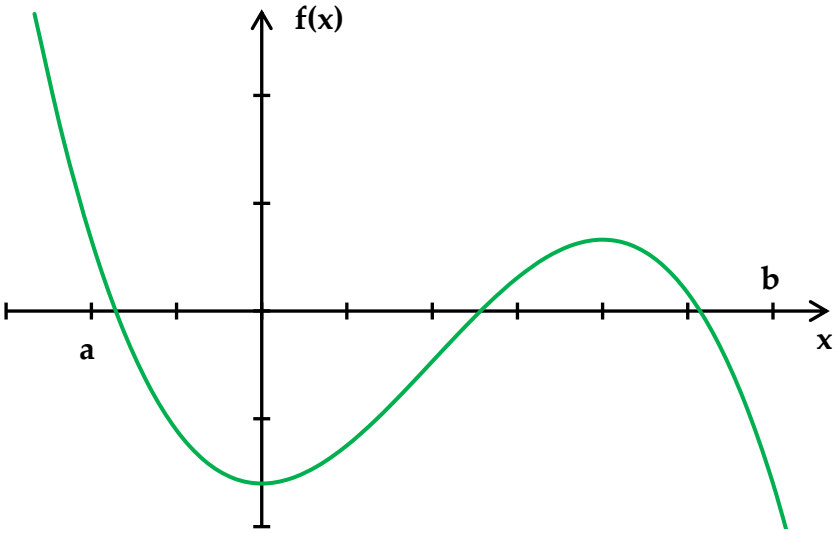


Figura 8. Interval [a; b] care conține 3 soluții.

Procesul iterativ

Iterația 1 – se calculează iterata x_1 situată la jumătatea intervalului [a; b], așa cum se poate observa în Figura 9. Astfel:

$$x_1 = \frac{a+b}{2} \tag{1.7}$$

Dintre cele două subintervale rezultate, se continuă procesul iterativ cu subintervalul care conține rădăcina, deci respectă condiția 1.6. Pentru situația studiată, noul interval de căutare a rădăcinii este [a; x_1]. In această etapă se și reinițializează capetele intervalului [a; b]. Astfel:

- Dacă $f(a) \cdot f(x_1) < 0$ atunci $b \coloneqq x_1$
- Dacă $f(x_1) \cdot f(b) < 0$ atunci $a \coloneqq x_1$

In consecință, iteratele următoare se calculează folosind aceeași relație ca în cazul iteratei x_1.

Iterația 2 – se calculează iterata x_2 situată la jumătatea noului interval [a; b], deci practic la jumătatea intervalului [a; x_1]. Se verifică din nou care subinterval conține rădăcina și se reinițializează limitele intervalului [a; b].

Procesul iterativ continuă într-o manieră similară până când lungimea intervalului curent este mai mică decât o toleranță de calcul impusă, notată cu ε. Toleranța de calcul semnifică practic eroarea acceptată dintre soluția exactă și cea determinată prin procedeul numeric descris anterior. Testul de convergență (de oprire a procesului iterativ), care se verifică în cadrul fiecărei iterații, poate fi exprimat astfel:

$$|x_{n+1} - x_n| < \varepsilon \tag{1.8}$$

Dacă relația 1.8 este îndeplinită atunci procesul iterativ se oprește, iar iterata x_{n+1} este rădăcina ecuației f(x)=0 în toleranța de calcul acceptată, ε. In acest caz $f(x_{n+1}) \approx 0$.

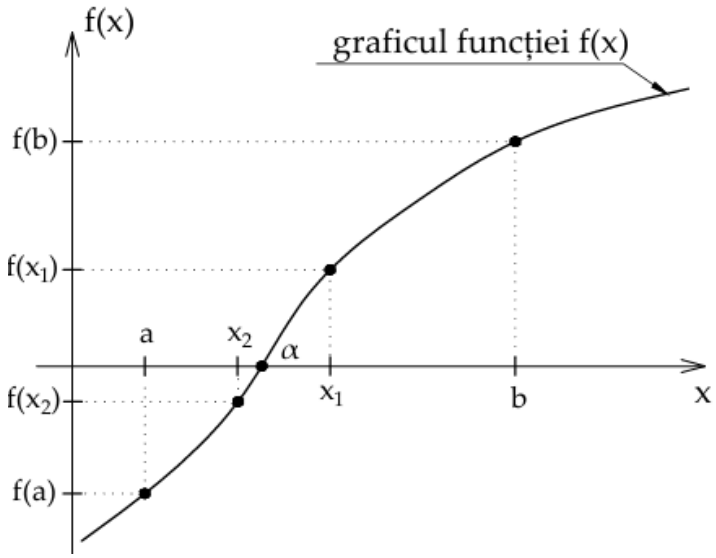


Figura 9. Interpretarea grafică a metodei bisecției

Observații:

- metoda bisecției converge necondiționat spre rădăcină;
- metoda este înceată, având ordin de convergență 1 (convergență liniară). Eroarea iteratelor față de rădăcina exactă descrește liniar între două iterații succesive;
- metoda permite calcularea numărului de iterații (n) necesare pentru determinarea soluției cu o toleranță de calcul impusă:

$$n \geq log_2 \frac{b-a}{\epsilon} \tag{1.9}$$

1.3.2 Aplicație Matlab

```
clearvars
clc
% Introducere date de intrare
% Functia f(x)
f=@(x) 3.14159*x^2*(3-x/3)-80;
%toleranta de calcul acceptata
TOL=1E-3;
% capetele intervalului (a si b)
```

```
a=3;
b=4;
% nr max de iteratii acceptat
iter_maxim = 1000;

%% Rezolvare
if f(a)==0
   disp('Solutie in punctul x = a')
   return % se opreste executia programului
else
   if f(b)==0
      disp('Solutie in punctul x = b')
      return % se opreste executarea programului
   end
end

if f(a)*f(b)>0
   disp('Intervalul [a,b] nu contine nicio radacina')
   return % se opreste executarea programului
end

nr_it = 0; %initializare variabila care retine numarul de iteratii
Test_conv=abs(a-b);
Iteratii=zeros();


while (Test_conv > TOL && nr_it<iter_maxim)
  nr_it = nr_it + 1;
  x = (a + b)/2;
  if f(a)*f(x)<0
     b = x;
     Test_conv=abs(x-a); % test de conv= modul (x_curent-x_precedent)
  else
     a = x;
     Test_conv=abs(x-b); % test de conv= modul (x_curent-x_precedent)
  end

  Iteratii(nr_it,1)=nr_it;
  Iteratii(nr_it,2)=x;

  Iteratii(nr_it,3)=Test_conv;
end
```

```
Iteratii(nr_it,1)=nr_it;
Iteratii(nr_it,2)=x; % solutia ecuatiei

%% Afisare rezultate
if nr_it==iter_maxim
  fprintf('\n Solutia nu s-a putut calcula dupa %2i iteratii \n ', nr_it);
  fprintf('\n Valoarea calculata in ultima iteratie este: %2.10f \n ', x);
  fprintf('\n Verificare in ecuatia f(x)=0: %2.10f <--> %2.10f \n', x, f(x));
  return
end

if abs(x)==inf
  fprintf('\n Solutia nu s-a putut calcula \n ');
  fprintf('\n Verificati datele de intrare \n');
  return
end

[n1,~]=size(Iteratii);
fprintf ( ' Nr. iteratie     x(i)              |x(i) - x(i-1)|  \n');
%disp(' ');
fprintf('   \n');
for i1=1:n1
  fprintf(' %d       %4.16f      %4.16f \n ', Iteratii(i1,1),Iteratii(i1,2),Iteratii(i1,3)  );
%Iteratii
end

fprintf('\n Solutia este: %2.10f \n Testul de convergenta a fost atins dupa %2i iteratii \n ',
x, nr_it);
%proba in ecuatia f(x)=0
fprintf('\n Proba solutiei in ecuatia f(solutie) = %g  \n', f(x));
```

1.3.3 Aplicație rezolvată

Se va prezenta rezolvarea aplicației 1 descrisă în subcapitolul 1.1, folosind toleranța de calcul $\epsilon = 10^{-3}$. Intervalul care conține rădăcina se considera [3; 4]. Rezolvarea se prezintă tabelar pentru o vizualizare clară și sistematică a procesului iterativ.

$$f(x) = 3.14159 \cdot x^2 \cdot \left(3 - \frac{x}{3}\right) - 80$$

a	b	f(a)	f(b)
3	4	-23.45138	3.77573333

Nr. Iterație	Valoare iterată	f(xi)	Reinitializare limite interval				$\lvert x_i-x_{i-1}\rvert$	Test de convergenta
i	x_i		a	b	f(a)	f(b)		$\lvert x_i-x_{i-1}\rvert$ <0.001
1	3.5	-9.445125	3.5	4	-9.4451246	3.7757333	0.5	Nu
2	3.75	-2.687434	3.75	4	-2.6874336	3.7757333	0.25	Nu
3	3.875	0.587101	3.75	3.875	-2.6874336	0.5871013	0.125	Nu
4	3.8125	-1.040195	3.8125	3.875	-1.0401953	0.5871013	0.0625	Nu
5	3.84375	-0.223958	3.84375	3.875	-0.2239584	0.5871013	0.03125	Nu
6	3.859375	0.182231	3.84375	3.859375	-0.2239584	0.1822306	0.015625	Nu
7	3.8515625	-0.020701	3.8515625	3.859375	-0.0207006	0.1822306	0.0078125	Nu
8	3.8554688	0.080806	3.8515625	3.8554688	-0.0207006	0.0808060	0.0039063	Nu
9	3.8535156	0.030063	3.8515625	3.8535156	-0.0207006	0.0300629	0.0019531	Nu
10	3.8525391	0.004684	3.8515625	3.8525391	-0.0207006	0.0046837	0.0009766	Da

Analizând tabelul de mai sus se poate observa că rădăcina s-a determinat dupa 10 iterații și are valoarea 3.852539 (cu șase cifre semnificative după virgulă). Proba soluției indică valoarea 0.004684≈0, deci soluția este acceptabilă.

Aplicând relația 1.9, numărul de iterații pentru determinarea soluției cu toleranța 0.001, a=3 și b=4 este $n \geq 9.21$. Rotunjind la numărul întreg, se obțin 10 iterații, exact ca în tabelul de mai sus.

Folosind aplicația Matlab, datele de ieșire pentru această problemă de test sunt:

Nr. iteratie	x(i)	|x(i) - x(i-1)|
1	3.5000000000000000	0.5000000000000000
2	3.7500000000000000	0.2500000000000000
3	3.8750000000000000	0.1250000000000000
4	3.8125000000000000	0.0625000000000000
5	3.8437500000000000	0.0312500000000000
6	3.8593750000000000	0.0156250000000000
7	3.8515625000000000	0.0078125000000000
8	3.8554687500000000	0.0039062500000000
9	3.8535156250000000	0.0019531250000000
10	3.8525390625000000	0.0009765625000000

Solutia este: 3.8525390625

Testul de convergenta a fost atins dupa 10 iteratii

Proba solutiei in ecuatia f(solutie) = 0.00468369.

Studii parametrice

In tabelul de mai jos sunt sintetizate rezultatele unor teste numerice privind influența datelor de intrare (toleranță, lungimea intervalului) asupra rezultatelor (precizia soluției, numărul de iterații).

Nr.crt.		1	2	3
Date de intrare	a	3	3	3
	b	4	5	5
	toleranța	1.00E-03	1.00E-03	1.00E-06
Date de ieșire	soluție	3.85253906	3.85253906	3.85235882
	f(soluție)	4.68E-03	4.68E-03	-1.114E-06
	nr. Iterații	10	11	21

Comparând testele 1 și 2, se poate observa că mărirea lungimii intervalului [a; b], dar păstrarea toleranței de calcul conduce la creșterea numărului de iterații necesare determinării soluției. Micșorarea toleranței de calcul (testul 3 comparativ cu testul 2) conduce la determinarea unei soluții mai precise, însă numărul de iterații este sporit.

1.3.4 Aplicații propuse

P1. Să se determine soluția aplicației 2 (în kN) enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-4}$.

P2. Să se determine soluția aplicației 4 (în cm) enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-3}$.

P3. Să se determine rădăcina din intervalul [0.5; 1.5] a următoarei ecuații neliniare, cu toleranța $\epsilon = 10^{-6}$:

$$x^8 - 2 = 0 \tag{1.10}$$

1.4 Metoda falsei poziții pentru rezolvarea ecuațiilor neliniare de forma f(x)=0

1.4.1 Noțiuni teoretice

S-a arătat că metoda bisecției converge necondiționat spre rădăcină dacă funcția f(x) este continuă și se cunoaște un interval [a; b] care conține o rădăcină. Totuși convergența este înceată, iar o îmbunătățire a vitezei de convergență poate fi făcută din analiza grafică a metodei. Se observă că la împărțirea intervalului [a; b] în jumătăți egale nu se ține cont de valorile f(a) și f(b). In Figura 9, se poate observa că valoarea f(a) este mai aproape de 0 decât valoarea f(b) și astfel este de așteptat ca soluția căutată să fie mai aproape de valoarea lui a. Astfel, o variantă alternativă metodei bisecției este ca prima iterată să se găsească la intersecția dreptei care trece prin punctele A(a; f(a)) și B(b; f(b)) cu axa orizontală Ox, așa cum se arată în Figura 10.

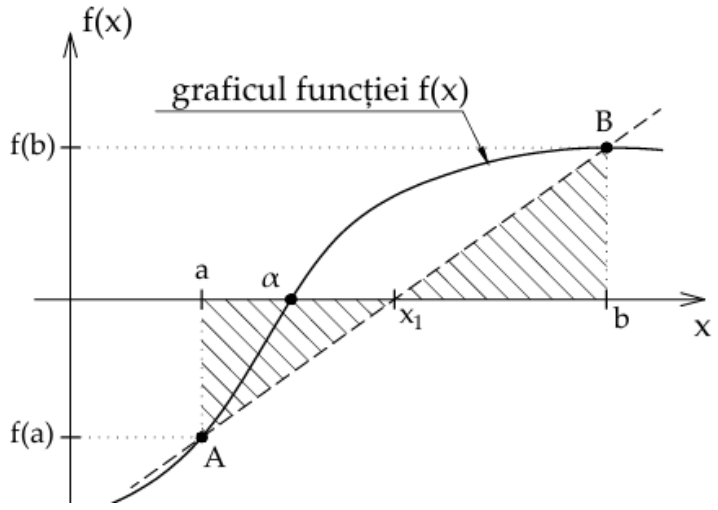


Figura 10. Metoda falsei poziții – determinarea grafică a primei iterate

Valoarea iteratei x_1 poate fi determinată, de exemplu, prin asemănarea celor două triunghiuri hașurate din Figura 10:

$$\frac{0 - f(a)}{x_1 - a} = \frac{0 - f(b)}{x_1 - b} \tag{1.11}$$

Prin operații algebrice elementare se obține:

$$x_1 = \frac{b \cdot f(a)}{f(a) - f(b)} - \frac{a \cdot f(b)}{f(a) - f(b)} \tag{1.12}$$

Adăugând și scăzând b in membrul drept se obține:

$$x_1 = b + \frac{b \cdot f(a)}{f(a) - f(b)} - b - \frac{a \cdot f(b)}{f(a) - f(b)} \tag{1.13}$$

Aducând apoi la același numitor al treilea termen al membrului drept, expresia pentru x_1 devine:

$$x_1 = b - \frac{f(b) \cdot (b - a)}{f(b) - f(a)} \tag{1.14}$$

Relația de calcul a primei iterate se poate determina alternativ și prin exprimarea ecuației dreptei care trece prin punctele A(a; f(a)) și B(b; f(b)):

$$\frac{y - f(b)}{f(b) - f(a)} = \frac{x - b}{b - a} \tag{1.15}$$

Considerând $x = x_1$ și $y = 0$ (coordonatele punctului de intersecția al dreptei AB cu axa Ox):

$$\frac{0 - f(b)}{f(b) - f(a)} = \frac{x_1 - b}{b - a} \tag{1.16}$$

Exprimându-l pe x_1 se obține:

$$x_1 = b - \frac{f(b) \cdot (b - a)}{f(b) - f(a)}, \tag{1.17}$$

relație care este identică cu expresia (1.13).

Procesul iterativ – interpretare grafică

Iterația 1 – se calculează iterata x_1 folosind relația (1.16). Dintre cele două subintervale rezultate, printr-un raționament identic cu cel descris în cadrul metodei bisecției, se alege subintervalul care conține rădăcina și se reinițializează capetele intervalului [a; b]. Astfel următoarele aproximări ale rădăcinii se evaluează cu aceeași relație.

Iterația 2 – se calculează iterata x_2:

$$x_2 = b - \frac{f(b) \cdot (b - a)}{f(b) - f(a)}, \tag{1.18}$$

În continuare, popularea șirului de iterate se face într-o manieră similară. Din punct de vedere grafic, interpretarea metodei este prezentată în Figura 11.

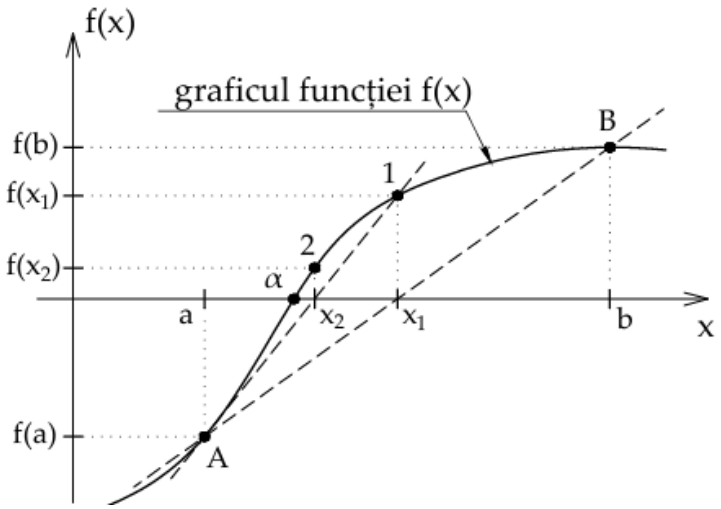


Figura 11. Interpretare grafică a metodei falsei poziții

Testul de convergență (de oprire a procesului iterativ), care se verifică în cadrul fiecărei iterații, poate fi exprimat astfel:

$$|x_{n+1} - x_n| < \varepsilon \tag{1.19}$$

unde ε reprezintă toleranța de calcul acceptată.

Observații:

- analizând Figura 11 se poate observa că iteratele x_i sunt de aceeași parte a soluției α.
- metoda falsei poziții este în general mai rapidă decât metoda bisecției, ordinul de convergență fiind 1, iar rata de convergență depinde de expresia funcției f(x) și de intervalul [a; b];
- testul practic de convergență (relația 1.18) nu este riguros. Astfel, la atingerea testului de oprire există posibilitatea ca diferența dintre ultimele două iterate ($|x_{n+1} - x_n|$) să fie mai mare decât diferența dintre rădăcina exactă α și soluția determinată numeric ($|\alpha - x_{n+1}|$).

1.4.2 Aplicație Matlab

```
clearvars
clc
% Introducere date de intrare
% Functia f(x)
```

```
f=@(x) 3.14159 * x^2 * (3-x/3)-80;
%f=@(x) x^8-2;
TOL=1e-3;
% aproximatile initiale x_1 si x_2
x(1) = 3;
x(2) = 4;
% nr max de iteratii
iter_maxim = 1000;

%% verificare solutii in aproximatiile initiale
if f(x(1))==0
   disp('Solutie in punctul x = x(1)')
   return % se opreste executia programului
else
   if f(x(2))==0
      disp('Solutie in punctul x = x(2)')
      return % se opreste executarea programului
   end
end

if f(x(1))*f(x(2))>0
   disp('Intervalul [x(1),x(2)] nu contine nicio radacina')
   return % se opreste executarea programului
end

%% start initializare iteratii
iteratie=0;
fprintf ( ' Nr. iteratie     x(i)              |x(i) - x(i-1)|  \n');
fprintf('   \n');
i1=3;
while i1<=iter_maxim
     x(i1) = x(1) - (f(x(1)))*((x(i1-1) - x(1))/(f(x(i1-1)) - f(x(1)))); % falsa pozitie in x1
     iteratie=iteratie+1;

     if f(x(1)) * f(x(3)) > 0 % radacina e in intervalul [x(3) , x(2)]=> se schimba punctul fals
        x(1) = x(2);
     end

     fprintf(' %3.0f        %4.16f      %4.16f  \n ', iteratie,x(i1),abs(x(i1)-x(i1-1)));

     if abs(x(i1)-x(i1-1))<TOL
      radacina=x(i1);
```

```
      disp(' ');
      fprintf(' Solutia ecuatiei in toleranta de calcul ceruta = %4.16f \n ', radacina);
      disp(' ');
      break
    end
    i1 = i1+1;
end

%proba in ecuatia f(radacina)=0
fprintf(' Verificarea solutiei obtinute: f(radacina) = %g \n', f(radacina));
```

1.4.3 Aplicație rezolvată

Se va prezenta rezolvarea aplicației 1 descrisă în subcapitolul 1.1, folosind toleranța de calcul $\epsilon = 10^{-3}$. Intervalul care conține rădăcina se consideră [3; 4]. Rezolvarea se prezintă tabelar pentru o vizualizare clară și sistematică a procesului iterativ.

$$f(x) = 3.14159 \cdot x^2 \cdot \left(3 - \frac{x}{3}\right) - 80$$

a	b	f(a)	f(b)
3	4	-23.45138	3.77573333

Nr. Iterație	Valoare iterată	f(xi)	Reinitializare limite interval				$\lvert x_i - x_{i-1} \rvert$	Test de convergenta
i	x_i		a	b	f(a)	f(b)		$\lvert x_i - x_{i-1} \rvert < 0.001$
1	3.861324508	0.232818	3	3.8613245	-23.4513800	0.2328182	0.8613245	Nu
2	3.852857597	0.012962	3	3.8528576	-23.4513800	0.0129624	0.0084669	Nu
3	3.8523865	0.000717	3	3.8523865	-23.4513800	0.0007171	0.0004711	Da

Analizând tabelul de mai sus, se poate observa că soluția a fost calculată după 3 iterații. In cadrul metodei bisecției au fost necesare 10 iterații, remarcându-se astfel performanța sporită a metodei falsei poziții. Trebuie menționat însă că sunt situații în care viteza de convergență a metodei bisecției este superioară celei aferente metodei falsei poziții (de exemplu, aplicația P1 din secțiunea 1.4.4).

Aplicând secvența Matlab descrisă anterior, în vederea rezolvării problemei de test, se afișează următoarele date de ieșire:

Nr. iteratie	x(i)	|x(i) - x(i-1)|
1	3.8613245081434755	0.1386754918565245
2	3.8528575973279016	0.0084669108155739

3 3.8523864520415567 0.0004711452863448

Solutia ecuatiei in toleranta de calcul ceruta = 3.8523864520415567

Verificarea solutiei obtinute f(radacina) = 0.000717144.

Studii parametrice

In tabelul de mai jos sunt sintetizate rezultatele unor teste numerice privind influența datelor de intrare (toleranță, lungimea intervalului) asupra rezultatelor (precizia soluției, numărul de iterații).

Nr.crt.		1	2	3
Date de intrare	a	3	3	3
	b	4	5	5
	toleranța	1.00E-03	1.00E-03	1.00E-06
Date de ieșire	soluție	3.85238645	3.85238113	3.85235880
	f(soluție)	7.17E-04	5.79E-04	9.79E-08
	nr. Iterații	3	4	7

Observațiile precizate în cadrul studiilor numerice asociate metodei bisecției sunt valabile și în acest caz. Totuși metoda se remarcă printr-un număr mai mic de iterații necesare pentru determinarea unei soluții cu o toleranță impusă comparativ cu metoda bisecției.

1.4.4 Aplicații propuse

P1. Să se determine rădăcina din intervalul [0.5; 1.5] a următoarei ecuații neliniare, cu toleranța $\epsilon = 10^{-6}$. Să se compare numărul de iterații cu cel aferent metodei bisecției.

$$x^8 - 2 = 0 \tag{1.20}$$

P2. Să se determine soluția aplicației 3 enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-4}$.

P3. Să se determine soluția aplicației 4 enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-3}$.

1.5 Metoda secantei pentru rezolvarea ecuațiilor neliniare de forma f(x)=0

1.5.1 Noțiuni teoretice

Cele două metode prezentate în secțiunile anterioare necesită cunoașterea unui interval care să conțină rădăcina căutată și astfel, cele două aproximații inițiale trebuie să încadreze soluția. Acest neajuns este eliminat în cadrul metodei secantei. Această metoda necesită, de asemenea, precizarea a două aproximații inițiale ale soluției, însă acestea pot încadra soluția sau pot fi de aceeași parte a ei.

Procesul iterativ – interpretare grafică

x_0, x_1 – aproximații inițiale cunoscute;

Iterația 1 – se evaluează x_2. Din punct de vedere grafic, iterata x_2 se află la intersecția secantei care trece prin puncte $0(x_0; f(x_0))$ și $1(x_1; f(x_1))$ cu axa absciselor Ox, așa cum se prezintă în Figura 12.

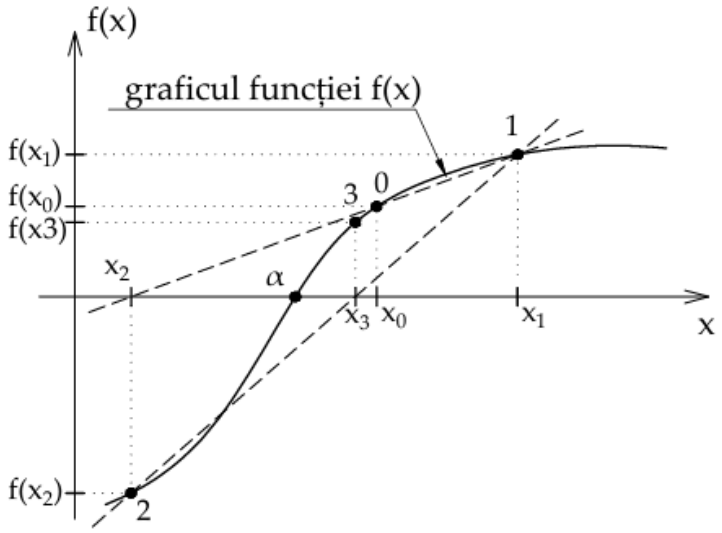


Figura 12. Interpretare grafică a metodei secantei

Expresia analitică a iteratei x_2 se poate deduce din ecuației dreptei care trece prin punctele $0(x_0; f(x_0))$ și $1(x_1; f(x_1))$:

$$\frac{y - f(x_1)}{f(x_1) - f(x_0)} = \frac{x - x_1}{x_1 - x_0} \tag{1.21}$$

Coordonatele punctului de intersecția al dreptei 12 cu axa Ox sunt $x = x_2$ și $y = 0$. Astfel relația (1.21) devine:

$$\frac{0 - f(x_1)}{f(x_1) - f(x_0)} = \frac{x_2 - x_1}{x_1 - x_0} \tag{1.22}$$

Exprimându-l pe x_2 se obține:

$$x_2 = x_1 - \frac{f(x_1)}{f(x_1) - f(x_0)}(x_1 - x_0) \tag{1.23}$$

Iterația 2 – se evaluează x_3. Din punct de vedere grafic, iterata x_3 se află la intersecția secantei care trece prin puncte $1(x_1; f(x_1))$ și $2(x_2; f(x_2))$ cu axa absciselor Ox. Analitic, x_3 se evaluează cu următoarea relație:

$$x_3 = x_2 - \frac{f(x_2)}{f(x_2) - f(x_1)}(x_2 - x_1) \tag{1.24}$$

Iteratele următoare pot fi evaluate cu următoarea formulă generală de iterare:

$$x_{n+1} = x_n - \frac{f(x_n)}{f(x_n) - f(x_{n-1})}(x_n - x_{n-1}), n \geq 1. \tag{1.25}$$

Testul de convergență (de oprire a procesului iterativ) poate fi exprimat astfel:

$$|x_{n+1} - x_n| < \varepsilon \tag{1.26}$$

unde ε reprezintă toleranța de calcul acceptată. Relația (1.26) nu reprezintă un text practic riguros de convergență, motiv pentru care, adesea, se verifică și ca:

$$f(x_{n+1}) < \varepsilon \tag{1.27}$$

Teorema – cu privire la convergența metodei [6, 7]

Dacă:

- funcția f(x) este continuă în vecinătatea rădăcinii α;
- există derivatele $f'(x)$, $f''(x)$ și sunt continue în vecinătatea rădăcinii α;
- $f'(\alpha) \neq 0$;
- x_0, x_1 sunt suficient de aproape de α;

Atunci:

- șirul de iterate $x_n \to \alpha$;
- ordinul de convergență este $p \approx 1.618$.

Observații:

- dacă cele două aproximații inițiale încadrează soluția atunci prima iterată a metodei secantei este identică cu cea asociată metodei falsei poziții. A doua iterată (x_3) a metodei secantei se obține folosind valorile x_1 $şi$ x_2, fără a se verifica dacă cele două valori încadrează soluția, așa cum se procedează în cadrul metodei falsei poziții;
- metoda poate suferi pierdere de semnificație la evaluarea fracției din formula metodei atunci când două iterate succesive au valori apropiate;
- în cadrul fiecărei iterații se face o singură evaluare de funcție ($f(x_n)$) considerând că $f(x_{n-1})$ s-a calculat în iterația anterioară și s-a stocat;
- dacă procesul iterativ este convergent spre rădăcina α, atunci viteza de convergență este superioară (ordin de convergență $p \approx 1.618$) celor aferente metodelor prezentate anterior, caracterizate prin ordin de convergență $p = 1$. Mai mult, trei iterații în metoda secantei au ordin de convergență echivalent cu două iterații efectuate cu o metodă care are convergență pătratică ($p = 2$):

$$1.618^3 \approx 2^2 \tag{1.28}$$

1.5.2 Aplicație Matlab

```
clearvars
clc
% Introducere date de intrare
% Functia f(x)
f=@(x) 3.14159*x^2*(3-x/3)-80;
% aproximatiile initiale
x(1)= 3;
x(2)= 4;
% toleranta de calcul ecceptată
TOL=1e-3;
% numarul maxim de iteratii
lnit=100;

%% verificare solutii in aproximatiile initiale
if f(x(1))==0
   disp('Solutie in punctul x = x(1)')
   return % se opreste executia programului
else
```

```
   if f(x(2))==0
      disp('Solutie in punctul x = x(2)')
      return % se opreste executarea programului
   end
end

%% start initializare iteratii
iteratie=0;
fprintf ( ' Nr. iteratie     x(i)              |x(i) - x(i-1)|  \n');
fprintf('   \n');
i=2;
solutie = 0;
while i<=lnit
   i = i+1;
   x(i) = x(i-1) - (f(x(i-1)))*((x(i-1) - x(i-2))/(f(x(i-1)) - f(x(i-2))));
   iteratie=iteratie+1;
   % verificare divergenta locala (se calculeaza distanta intre 3 iteratii
   % iterate consecutive
   div1 = abs (x(i-2) - x(i-1));
   div2 = abs (x(i-1)) - x(i);
   divergenta = "  ";
   if div1 < div2
      divergenta = " Diverg. locala ";
   end

   fprintf(' %3.0f       %4.16f      %4.16f  %s \n ', iteratie,x(i),abs(x(i)-x(i-1)), divergenta);

   if abs(x(i)-x(i-1))<TOL
      radacina=x(i);
      disp(' ');
      fprintf(' Solutia ecuatiei in toleranta de calcul ceruta = %4.16f \n ', radacina);
      disp(' ');
      fprintf(' Verificarea solutiei obtinute f(radacina) = %g \n', f(radacina));
      solutie = 1;
      return
   end

end

if solutie == 0
   fprintf ('Solutia nu s-a putut determina cu toleranta de calcul = %g \n', TOL);
end
```

1.5.3 Aplicație rezolvată

Se va prezenta rezolvarea aplicației 1 descrisă în subcapitolul 1.1, folosind toleranța de calcul $\epsilon = 10^{-3}$. In primul test numeric, aproximațiile inițiale se vor considera $x_0 = 3$ și $x_1 = 4$, deci identice cu cele folosite în cadrul metodei falsei poziții. Procesul iterativ este prezentat în tabelul de mai jos:

$f(x) = 3.14159 \cdot x^2 \cdot \left(3 - \frac{x}{3}\right) - 80$			
x0	x1	f(x0)	f(x1)
3	4	-23.45138	3.77573333

Nr. Iterație	**Valoare iterată**	**$f(x_{i+1})$**	**$\lvert x_i - x_{i-1} \rvert$**	**Test de convergenta**
i	x_{i+1}			$\lvert x_i - x_{i-1} \rvert < 0.001$
1	3.861324508	0.232818	0.86132451	Nu
2	3.852211624	-0.003827	0.00911288	Nu
3	3.8523590	0.000004	0.00014737	Da

Se poate observa că soluția a fost determinată după 3 iterații. Proba soluției, $f(x_4) = 0.00004 \approx 0$ certifică faptul că soluția calculată este corectă în limitele toleranței de calcul acceptate. Comparând rezultatele cu cele prezentate în cadrul metodei falsei poziții, se observă că primele iterate asociate celor două metode coincid. Totuși, următoarele sunt diferite datorită faptului că în metoda falsei poziții, cele două valori pe baza cărora se calculează iterata curentă trebuie să încadreze soluția, iar în metoda secantei se folosesc ultimele două iterate, fără a se verifica dacă respectă condiția menționată anterior.

Pentru a arăta versatilitatea metodei secantei comparativ cu metoda falsei poziții, se va prezenta în continuare rezolvarea ecuației de mai sus, folosind ca aproximații inițiale valorile $x_0 = 2$ și $x_1 = 3$ (intervalul [2; 3] nu conține rădăcină și astfel metoda falsei poziții, respectiv metoda bisecției nu vor converge spre soluție).

$f(x) = 3.14159 \cdot x^2 \cdot \left(3 - \frac{x}{3}\right) - 80$			
x0	x1	f(x0)	f(x1)
2	3	-50.67849333	-23.45138

Nr. Iterație	Valoare iterată	$f(x_{i+1})$	$\lvert x_i - x_{i-1} \rvert$	Test de convergenta
i	x_{i+1}			$\lvert x_i - x_{i-1} \rvert < 0.001$
1	3.861324508	0.232818	1.86132451	Nu
2	3.852857597	0.012962	0.00846691	Nu
3	3.8523584	-0.000012	0.0004992	Da

Se observă că procesul este convergent, iar soluția a fost calculată după 3 iterații.

Secvența Matlab, descrisă anterior, afișează următoarele date de ieșire pentru acest test numeric:

```
Nr. iteratie          x(i)                  |x(i) - x(i-1)|
      1        3.8613245081434755     0.8613245081434755
      2        3.8528575973279016     0.0084669108155739
      3        3.8523583980887359     0.0004991992391656
Solutia ecuatiei in toleranta de calcul ceruta = 3.8523583980887359
Verificarea solutiei obtinute f(radacina) = -1.20297e-05.
```

Studii parametrice

In tabelul de mai jos sunt sintetizate rezultatele unor teste numerice realizate pe de-o parte pentru compararea vitezei de convergență a metodei secantei cu cea a metoda falsei poziții (testul 1), iar pe de altă parte pentru evidențierea unor erori ce pot apărea datorită testului de convergență folosit (testul 2).

Nr.crt.		1	2
Date de intrare	x0	3	4.611
	x1	5	7.2
	toleranța	1.00E-06	1.00E-06
Date de ieșire	soluție	3.85235886	7.2000
	f(soluție)	-8.69E-10	1.77E+01
	nr. Iterații	5	3

Considerând cele două aproximații inițiale ca fiind 3, respectiv 5 și toleranța de calcul egală cu 1E-6, soluția e determinată după 5 iterații. Cu aceleași date de intrare, metoda falsei poziții a avut nevoie de 7 iterații, iar metoda bisecției de 21 de iterații, fiind deci mai lente decât metoda secantei. Testul 2 scoate în evidență o eroare ce poate apărea la folosirea testului de

convergență dat de relația (1.26). Pentru datele de intrare particulare folosite în acest test, soluția e estimată eronat, deoarece proba soluției este 17.7≠0. Acest neajuns poate fi eliminat prin cuplarea testului (1.26) cu testul suplimentar exprimat prin relația (1.27).

1.5.4 Aplicații propuse

P1. Să se determine rădăcina din intervalul [0.5; 1.5] a următoarei ecuații neliniare, cu toleranța $\epsilon = 10^{-6}$. Să se compare numărul de iterații cu cel aferent metodei falsei poziții.

$$x^8 - 2 = 0 \tag{1.29}$$

P2. Să se determine soluția aplicației 3 enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-4}$.

P3. Să se determine soluția aplicației 4 enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-3}$.

P4. Să se rezolve următoarea ecuație neliniară, cu toleranța $\epsilon = 10^{-6}$ considerând $x_0 = 0.1$ și $x_1 = 0.2$. Converge metoda spre soluția evidentă $x = 0$?

$$\sqrt[3]{x} - 0.1 \cdot x = 0 \tag{1.30}$$

1.6 Metoda Newton pentru rezolvarea ecuațiilor neliniare de forma f(x)=0

1.6.1 Noțiuni teoretice

Metoda Newton, denumită și metoda tangentei, este probabil cea mai utilizată metodă de rezolvare a ecuațiilor neliniare de forma f(x)=0. Spre deosebire de metodele discutate anterior, în cadrul metodei Newton este suficientă precizarea unei singure aproximații inițiale.

Procesul iterativ – interpretare grafică

x_0 – aproximație inițială cunoscută;

Iterația 1 – se evaluează x_1. Din punct de vedere grafic, iterata x_1 se află la intersecția tangentei la graficul funcției prin punctul $0(x_0; f(x_0))$ cu axa absciselor Ox, așa cum se prezintă în Figura 13. Panta acestei drepte este dată de valoarea derivatei $f'(x_0)$.

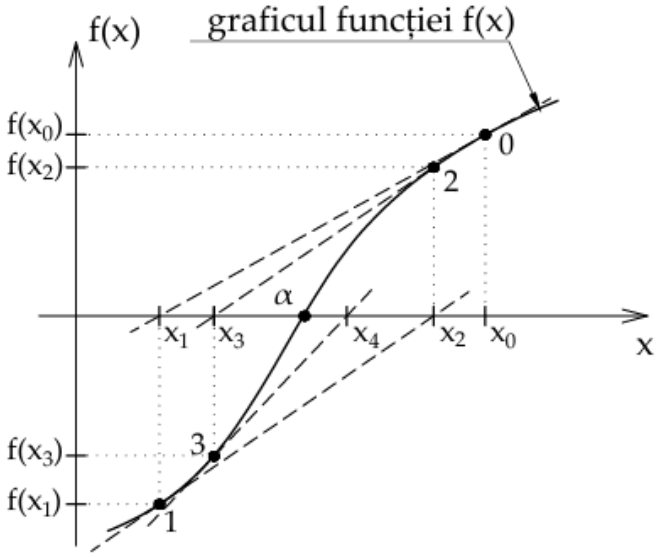


Figura 13. Interpretare grafică a metodei Newton

Expresia analitică a iteratei x_1 se poate deduce din ecuației tangentei la grafic prin punctul $0(x_0; f(x_0))$:

$$y - f(x_0) = f'(x_0) \cdot (x - x_0) \tag{1.31}$$

Coordonatele punctului de intersecția al acestei drepte cu axa Ox sunt $x = x_1$ și $y = 0$. Astfel relația (1.30) devine:

$$0 - f(x_0) = f'(x_0) \cdot (x_1 - x_0) \tag{1.32}$$

Exprimându-l pe x_1 se obține:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{1.33}$$

Iterația 2 – se evaluează x_2. Din punct de vedere grafic, iterata x_2 se află la intersecția tangentei la graficul funcției prin punctul $1(x_1; f(x_1))$ cu axa absciselor Ox. Analitic, x_2 se evaluează cu următoarea relație:

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} \tag{1.34}$$

Iteratele următoare pot fi evaluate cu următoarea formulă generală de iterare:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, n \geq 0 \tag{1.35}$$

Formula de iterare asociată metodei Newton poate fi dedusă și analitic, folosind dezvoltarea în serie Taylor a funcției f(x) pe baza funcției și a derivatelor funcției evaluate în x_0:

$$f(x) = f(x_0) + (x - x_0) \cdot f'(x_0) + (x - x_0)^2 \frac{f''(x_0)}{2!} + \cdots + (x - x_0)^k \frac{f^{(k)}(x_0)}{k!} + R_k \tag{1.36}$$

unde R_k reprezintă restul de ordin k și include efectul termenilor de ordin superior (k+1...∞):

$$R_k = \frac{f^{(k+1)}(\xi)}{(k+1)!}(x - x_0)^{(k+1)}, \qquad \xi \in (x_0;\, x) \tag{1.37}$$

Păstrând doar primii doi termeni ai sumei și considerând $f(x) = 0$, iar $x = x_1$, se obține:

$$0 = f(x_0) + (x_1 - x_0) \cdot f'(x_0) \tag{1.38}$$

Exprimându-l pe x_1 rezultă expresia (1.33). După n iterații se cunoaște valoarea x_n și poate fi folosită pentru evaluarea funcției în x_{n+1}:

$$f(x_{n+1}) = f(x_n) + (x_{n+1} - x_n) \cdot f'(x_n) \tag{1.39}$$

Pentru $n \to \infty$, $f(x_{n+1}) \to 0$ și astfel se deduce relația de calcul pentru x_{n+1}, care e identică cu expresia (1.35). Dezvoltarea în serie Taylor este utilă și pentru studiul convergenței și estimarea erorii dintre două iterații succesive. Testul de convergență (de oprire a procesului iterativ) poate fi exprimat prin relația (1.26). In cadrul metodei Newton, testul de convergență este riguros [7].

Teorema – cu privire la convergența metodei, estimarea erorii [8]

Dacă:

- funcția f(x) este continuă în vecinătatea rădăcinii α;
- există derivatele $f'(x)$, $f''(x)$ și sunt continue în vecinătatea rădăcinii α;
- $f'(\alpha) \neq 0$;
- x_0 este ales suficient de aproape de α;

Atunci:

- șirul de iterate $(x_n)_{n\geq 0} \to \alpha$;
- ordinul de convergență este $p = 2$ (convergență pătratică);
- eroarea din 2 pași succesivi este dată de expresia:

$$\alpha - x_{n+1} = (\alpha - x_n)^2 \cdot \frac{f''(\xi)}{2 \cdot f'(x_n)} \tag{1.40}$$

unde $\xi \in (x_n;\ \alpha)$. Relația anterioară arată că eroarea iteratei x_{n+1} în raport cu rădăcina α este proporțională cu pătratul erorii iteratei x_n. Astfel, dacă aproximația inițială este aleasă suficient de aproape de rădăcina căutată, atunci eroarea în iterațiile următoare va scădea rapid.

Observații:

- metoda poate suferi pierdere de semnificație la evaluarea fracției din formula metodei atunci când valoarea derivatei evaluată pentru iterata precedentă este ≈0;
- în cadrul fiecărei iterații se fac două evaluări ($f(x_n), f'(x_n)$). Astfel, chiar dacă ordinul de convergență al metodei Newton este superior celui aferent metodei secantei, timpul de calcul necesar determinării soluției cu un nivel de precizie impus poate fi mai mic în cadrul metodei secantei unde se face o singură evaluare de funcție per iterație.
- derivata funcției poate fi evaluată numeric folosind următoarea relație:

$$f'(x) \cong \frac{f(x+h) - f(x)}{h} \tag{1.41}$$

unde h este un increment mic (de ex. 1E-6). În acest caz, nu este necesară precizarea expresiei derivatei funcției.

1.6.2 Aplicație Matlab

```
clc
clearvars
% Introducere date de intrare
% Functia f(x)
f=@(x) 3.14159*x^2*(3-x/3)-80;
% Derivata functiei f(x)
fp = @(x)  6*3.14159*x-3.14159*x^2;
% alegerea aproximatiei initiale
x0 = 4;
% definirea tolerantei
tol = 1e-3;
% numarul maxim de iteratii
iter_maxim = 100;

%Rezolvare
x = zeros();
x(1) = x0;
i = 2;
nfinal = iter_maxim + 1;
iter = 1;
fprintf ( ' Nr. iteratie     x(i)              |x(i) - x(i-1)|  \n');
fprintf('   \n');
while (i <= iter_maxim + 1)
   fe = f(x(i - 1));
   fpe = fp(x(i - 1));
   x(i) = x(i - 1) - fe/fpe;
   if abs(x(i)-x(i-1))<= tol
      nfinal = i;
      fprintf(' %3.0f       %4.16f      %4.16f \n ', iter,x(i),abs(x(i)-x(i-1)));
      radacina=x(i);
      break; % se iese din ciclul "while"
   end
   fprintf(' %3.0f       %4.16f      %4.20f \n ', iter,x(i),abs(x(i)-x(i-1)));
   radacina=x(i);
   iter = iter + 1;
   i = i + 1;
end

%Afisare rezultate
if i> iter_maxim
```

```
   disp (' ');
   fprintf('  Nu s-a putut determina valoarea solutiei cu toleranta de calcul \n ');
   fprintf(' Aproximatia solutiei ecuatiei dupa %3.0f iteratii este %4.16f \n ', iter_maxim,
radacina);
   fprintf(' Proba ultimei iteratii in ecuatia f(solutie) = %g  \n', f(radacina));
else
   disp (' ');
   fprintf(' Solutia ecuatiei in toleranta de calcul ceruta = %4.16f \n ', radacina);
%disp (' ');
%fprintf(' Verificarea solutiei obtinute f(radacinaacina)= %4.20f \n', f(radacina));
   fprintf('\n Proba solutiei in ecuatia f(solutie) = %g  \n', f(radacina));
   disp (' ');
end
```

1.6.3 Aplicație rezolvată

Se va prezenta rezolvarea aplicației 1 descrisă în subcapitolul 1.1, folosind toleranța de calcul $\epsilon = 10^{-3}$. Aproximația inițială se consideră $x_0 = 3.5$. Această valoare s-a ales la jumătatea intervalului considerat în cadrul metodei bisecției, falsei poziții și secantei. S-a procedat astfel pentru a putea compara rezultatele cu cele obținute aplicând metodele menționate anterior. Procesul iterativ este prezentat în tabelul de mai jos:

$f(x) = 3.14159 \cdot x^2 \cdot \left(3 - \frac{x}{3}\right) - 80$		
$f'(x) = 6 \cdot 3.14159 \cdot x - 3.14159 \cdot x^2$		
x0	f(x0)	f'(x0)
3.5	-9.445124583	27.4889125

Nr. Iterație	Valoare iterată	$f(x_i)$	$f'(x_i)$	$\lvert x_i - x_{i-1} \rvert$	Test de convergenta
i	x_i				$\lvert x_i - x_{i-1} \rvert < 0.001$
1	3.843597608	-0.227926	26.038576	0.343597608	Nu
2	3.852351021	-0.000204	25.991938	0.008753413	Nu
3	3.852358861	0.00000000	25.991896	7.83972E-06	Da

Analizând datele afișate în tabelul de mai sus se poate observa că soluția a fost calculată după 3 iterații. Pentru acest exemplu de test, numărul de iterații necesare determinării soluției cu toleranța 1E-3 este identic cu cel aferent metodei secantei, respectiv metodei falsei

poziții. Totuși, pentru toleranțe de calcul mai mici, metoda Newton va converge spre soluție după un număr mai mic de iterații comparativ cu celelalte metode.

Pentru testul numeric considerat în această secțiune, secvența Matlab, descrisă anterior, afișează următoarele date de ieșire:

```
Nr. iteratie        x(i)                 |x(i) - x(i-1)|
    1        3.8435976080659189    0.34359760806591888027
    2        3.8523510211866405    0.00875341312072164257
    3        3.8523588609069415    0.0000078397203009
Solutia ecuatiei in toleranta de calcul ceruta = 3.8523588609069415
Proba solutiei in ecuatia f(solutie) = -1.64576e-10
```

Studii parametrice

In tabelul de mai jos sunt sintetizate rezultatele unor teste numerice realizate pe de-o parte pentru compararea vitezei de convergență a metodei Newton cu cea a secantei (testul 1), iar pe de altă parte pentru evidențierea pierderii de semnificație care poate apărea la evaluarea fracției din formula metodei, în cazul numitorului nul.

Nr.crt.		**1**	**2**
Date de intrare	**x0**	4	0
	toleranța	1.00E-06	1.00E-06
Date de ieșire	**soluție**	3.85235886	Inf
	f(soluție)	-1.28E-12	-
	nr. Iterații	3	-

Se poate observa că în cadrul testului 1, acceptând o toleranță de calcul de 1E-6, soluția a fost calculată după doar 3 iterații. Același test numeric a fost derulat și în cadrul metodei secantei, folosind ca aproximații inițiale valorile 3, respectiv 5, iar soluția a fost determinată după 5 iterații.

În cadrul testului 2, s-a folosit ca aproximație inițială valoarea 0. Pentru acest caz particular metoda Newton nu converge spre soluție, deoarece în cadrul primei iterații valoarea numitorului este 0, întrucât aproximația inițială introdusă anulează derivata funcției. Acest

neajuns poate fi eliminat prin evaluarea numerică aproximativă a derivatei funcției, folosind relația (1.41). Pentru situația studiată, aplicând evaluare numerică a derivatei, aplicația Matlab generează soluția după 42 de iterații:

Nr. iteratie	x(i)	|x(i) - x(i-1)|
1	8490949.5237000305205584	8490949.523700030520558357 24
2	5660581.5348531641066074	2830367.98884686641395092010
3	3774336.0359295625239611	1886245.49892360158264636993
4	2516199.5353629458695650	1258136.50056661665439605713
...		
40	7.7172947711391870	0.007632821683891677 37
41	7.7172740087047860	0.000020762434401078 17
42	7.7172740085439511	0.0000000001608349

Solutia ecuatiei in toleranta de calcul ceruta = 7.7172740085439511

Proba solutiei in ecuatia f(solutie) = 1.42109e-14.

1.6.4 Aplicații propuse

P1. Să se rezolve aplicația anterioară folosind $\epsilon = 10^{-6}$ și $x_0 = 6$. Derivata funcției se va evalua atât exact (folosind expresia derivatei funcției) cât și aproximativ (numeric).

Obs. Valoarea 6 anulează derivata funcției.

P2. Să se determine rădăcina următoarei ecuații neliniare, cu toleranța $\epsilon = 10^{-6}$. Aproximația inițială se va considera 0.5. Să se compare numărul de iterații cu cel aferent metodei secantei.

$$x^8 - 2 = 0 \tag{1.42}$$

P3. Fie ecuația [6]:

$$7 \cdot x - 5 \cdot x \cdot cos(x) + 3 \cdot sin(x) = 0 \tag{1.43}$$

Să se găsească o soluție a ecuației, folosind ca date de intrare $x_0 = 5$ și $\epsilon = 10^{-4}$.

P4. Să se determine soluția aplicației 2 enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-3}$.

1.7 Metoda punctului fix pentru rezolvarea ecuațiilor neliniare de forma x=g(x)

1.7.1 Noțiuni teoretice

Pornind de la ipoteza că o ecuație neliniară de forma $f(x) = 0$ poate fi transformată într-o ecuație echivalentă de forma $x = g(x)$, determinarea soluției (soluțiilor) poate fi realizată prin aplicarea metodei punctului fix pentru care formula de iterare (de generare a șirului de iterate) este direct determinată:

$$x_{n+1} = g(x_n);\ x_0 - specificat \tag{1.44}$$

În general, există mai multe moduri prin care transformarea menționată anterior este posibilă. O abordare generală, care asigura convergența metodei, va fi prezentată în secțiunile următoare, însă pentru început se va considera un exemplu pentru care transformarea poate fi făcută direct. Astfel, fie următoarea ecuație de forma $f(x) = 0$:

$$x^2 - 3 \cdot x + 2 = 0 \tag{1.45}$$

Rădăcinile acestei ecuații sunt valorile 1, respectiv 2 deoarece $f(1) = 0$ și $f(2) = 0$. Din punct de vedere grafic, rădăcinile se află la intersecția graficului funcției $y = f(x)$ cu axa orizontală Ox. Exprimându-l pe x din (1.45), se obține următoarea ecuație echivalentă, de forma $x = g(x)$:

$$x = \frac{x^2 + 2}{3} \equiv g(x) \tag{1.46}$$

Soluțiile acestei ecuații (denumite puncte fixe) sunt, de asemenea, valorile 1, respectiv 2 deoarece $g(1) = 1$ și $g(2) = 2$. Din punct de vedere grafic, aceste soluții sunt coordonatele orizontale ale punctelor de intersecție ale graficului funcției $y = g(x)$ cu dreapta de ecuație $y = x$ (prima bisectoare).

Analizând Figura 14, se poate observa că punctele de intersecție ale graficului funcției $y = f(x)$ cu axa Ox (rădăcinile ecuației $f(x) = 0$) sunt identice cu abscisele punctelor de intersecţie ale primei bisectoare cu graficul funcției $y = g(x)$. În consecință, pentru determinarea soluțiilor unei ecuații de forma $f(x) = 0$ se poate folosi o metodă numerică de determinare a soluțiilor unei ecuații echivalente de forma $x = g(x)$.

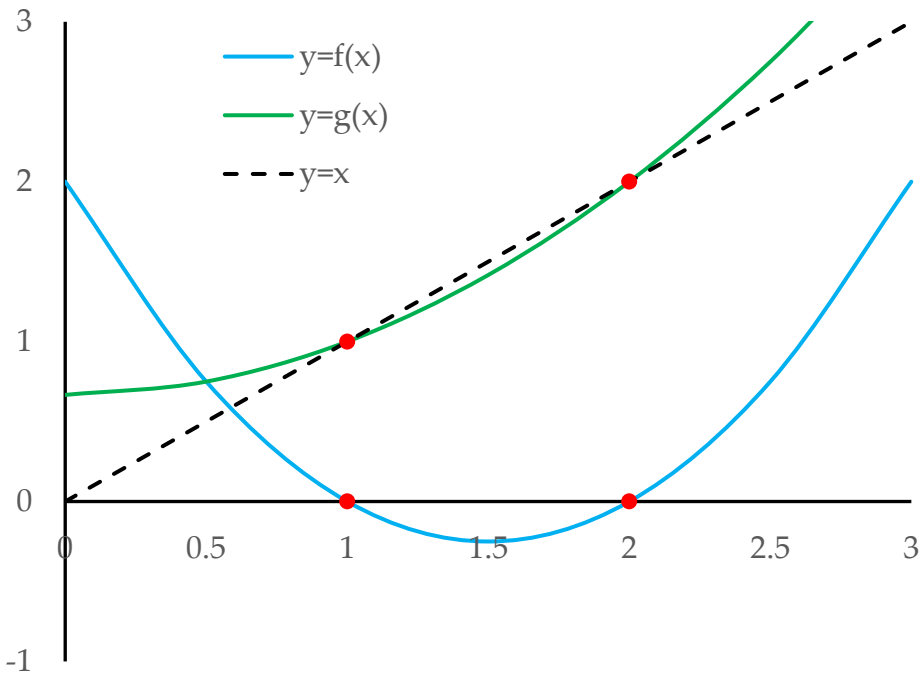


Figura 14. Reprezentare grafică comparativă rădăcini vs. puncte fixe.

Procesul iterativ

x_0 – aproximație inițială cunoscută;

Iterația 1 – se evaluează x_1:

$$x_1 = g(x_0) \tag{1.47}$$

Iterația 2 – se evaluează x_2:

$$x_2 = g(x_1) \tag{1.48}$$

Procesul iterativ continuă într-o manieră similară, iteratele următoare generându-se cu relația generală de iterare (1.44). Testul de convergență (de oprire a procesului iterativ) poate fi exprimat prin relația (1.26).

Teorema – cu privire la convergența metodei, unicitatea soluției, estimarea erorii. [7]

Dacă:

- $g: [a; b] \to [a; b]$ este continuă și derivabilă pe $[a; b]$;
- $|g'(x)| \leq \lambda < 1,\ (\forall)\ x \in [a; b]$;

Atunci:

- ecuația $x = g(x)$ are un singur punct fix $\alpha \in [a; b]$;
- $(\forall)\ x_0 \in [a; b]$ șirul $x_{n+1} = g(x_n)$ converge spre α;

- ordinul de convergență este $p = 1$ (convergență liniară);
- eroarea unei iterate curente poate fi estimată astfel:

$$|\alpha - x_n| \leq \frac{\lambda^n}{1-\lambda}|x_1 - x_0|, (\forall)\ n \geq 0 \tag{1.49}$$

Interpretarea grafică a metodei

Așa cum s-a menționat în secțiunile anterioare, din punct de vedere grafic, determinarea unei soluții (punct fix) a unei ecuații de forma $x = g(x)$ se reduce la determinarea unui punct de intersecție al graficului funcției $y = g(x)$ cu dreapta de ecuație $y = x$ (prima bisectoare). Pornind de la condiția de convergență $|g'(x)| < 1$ se disting două cazuri de convergență, respectiv două de divergență.

Observații:

- reamintim faptul că semnul derivatei de ordinul întâi indică monotonia funcției. Astfel, dacă derivata de ordinul întâi este pozitivă pe un interval, atunci funcția este strict crescătoare pe acel interval;
- din punct de vedere grafic, condiția $|g'(x)| < 1$ limitează panta tangentei la graficul funcției g(x) (pentru orice x din domeniul de definiție al funcției) la valoarea 1, care reprezintă panta primei bisectoare.
- cazuri de convergență:

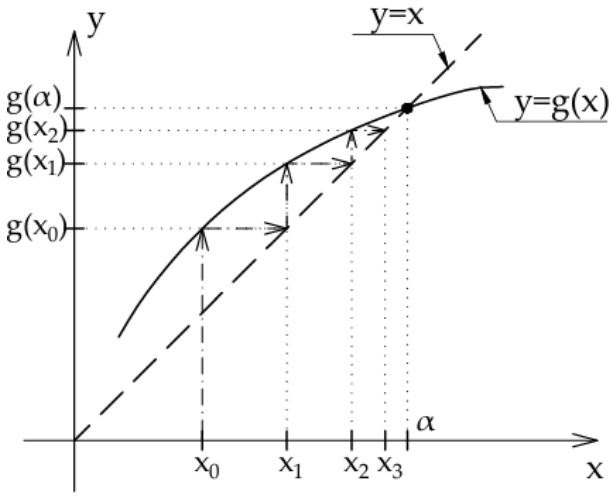


Figura 15. Convergență $0 < g'(x) < 1$

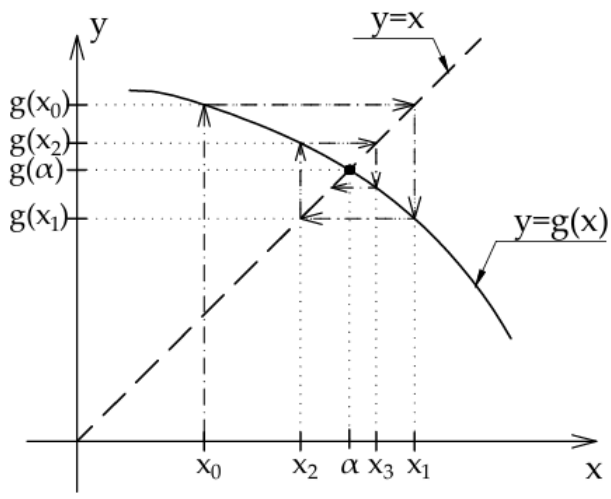


Figura 16. Convergență $-1 < g'(x) < 0$

- cazuri de divergență:

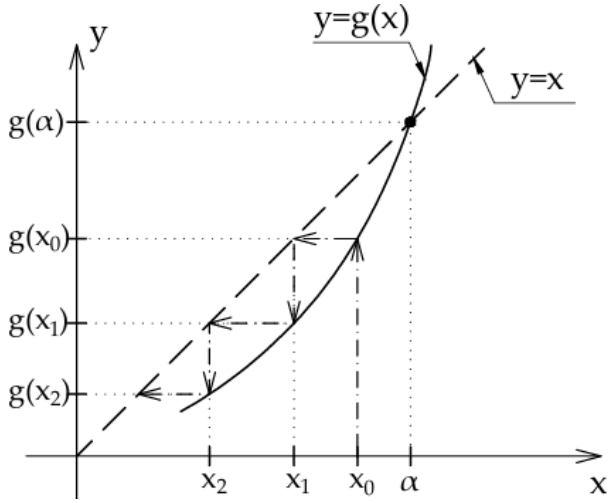


Figura 17. Divergență $g'(x) > 1$

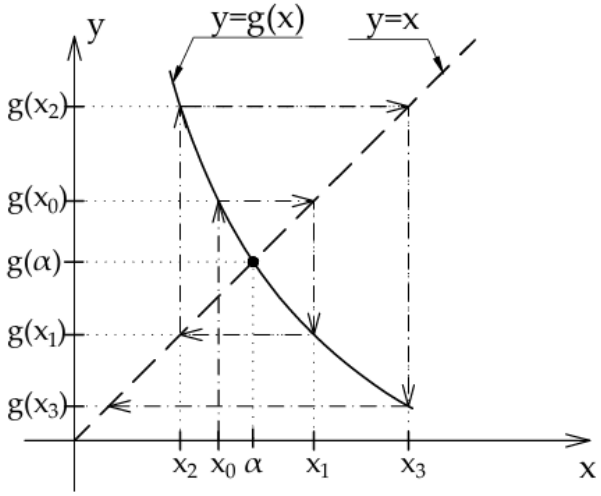


Figura 18. Divergență $g'(x) < -1$

Procedură explicită de punct fix

Trecerea de la o ecuație de forma $f(x) = 0$ într-una echivalentă de forma $x = g(x)$ poate fi făcută prin mai multe procedee. Adăugarea variabilei x în ambii membrii ai ecuației date sau scrierea ecuației într-o formă alternativă astfel încât într-o membru sa fie exprimat x, nu asigură îndeplinirea condiției de convergență $|g'(x)| < 1$ și implicit convergența metodei spre soluția căutata. O abordare generală, care elimină acest neajuns, poate fi realizată prin utilizarea unei proceduri de punct fix, așa cum se va prezenta în cele ce urmează. Cunoscând expresia funcției $f(x)$, se definește funcția $g(x)$ astfel:

$$g(x) = x - m \cdot f(x) \tag{1.50}$$

În relația de mai sus m este o constantă nenulă a cărei valoare se determină impunând condiția de convergență în $x = x0$:

$$|g'(x_0)| < 1 \tag{1.51}$$

Evaluând derivata de ordinul întâi a funcției $g(x)$ și explicitând modulul se obțin succesiv următoarele relații:

$$|1 - m \cdot f'(x_0)| < 1 \tag{1.52}$$

$$-1 < 1 - m \cdot f'(x_0) < 1 \tag{1.53}$$

$$0 < m \cdot f'(x_0) < 2 \tag{1.54}$$

Din relația (1.54) pot fi trasate următoarele consecințe:

- parametrul m trebuie să aibă același semn ca $f'(x_0)$;

- dacă $f'(x_0) < 0$ atunci:

$$m \in \left(\frac{2}{f'(x_0)};\ 0\right) \tag{1.55}$$

- dacă $f'(x_0) > 0$ atunci:

$$m \in \left(0;\ \frac{2}{f'(x_0)}\right) \tag{1.56}$$

Observații:

- alegerea parametrului m are influență directă asupra numărului de iterații necesar pentru determinarea soluției;
- valoarea optimă teoretică a parametrului m (care generează număr minim de iterații pentru determinarea soluției) este:

$$m = \frac{1}{f'(x_0)} \tag{1.57}$$

- observația anterioară este condiționată de calitatea aproximației inițiale. Dacă această nu este aleasă suficient de aproape de soluția căutată atunci, în general, afirmația precedentă nu este adevărată.
- în cazul utilizării valorii optime a parametrului m, formula de iterare (1.44) devine:

$$x_{n+1} = g(x_n) \equiv x_n - \frac{f(x_n)}{f'(x_0)} \tag{1.58}$$

- analizând relația anterioară se poate observa că prima iterată este identică cu cea evaluată folosind formula de iterare a metodei Newton. Mai mult, din punct de vedere grafic, o iterată curentă x_i se va afla la intersecția axei Ox cu dreapta de panta $f'(x_0)$ dusă prin punctul de coordonate $(x_i;\ f(x_i))$.

1.7.2 Aplicație Matlab

- Pentru determinarea intervalului de definiție al parametrului m:

```
clearvars
clc
% Introducere date de intrare:
% Expresia derivatei functiei
df=@(x) 6*3.14159*x-3.14159*x^2;
% Aproximatia initiala
```

```
x0=3.5;

%Evaluare limite parametru m
f_prim_x0=df(x0);

if f_prim_x0<0
   fprintf('\n Parametrul "m" apartine intervalului ( %2.16f; 0 ); \n', 2/f_prim_x0);
elseif f_prim_x0>0
   fprintf('\n Parametrul "m" apartine intervalului ( 0; %2.16f); \n', 2/f_prim_x0);
end

if f_prim_x0==0
   fprintf('\n Alege alta aproximatie initiala astfel incat f_prim(x0)≠0')
end

fprintf('\n Valoarea "optima" a parametrului m este %2.16f; \n', 1/f_prim_x0);
```

- Pentru rezolvarea ecuației neliniare:

```
clearvars;
clc;
% Introducere date de intrare
% Functia f(x)
f=@(x) 3.14159*x^2*(3-x/3)-80;
% parametrul m
m=0.0363783;
% Toleranta de calcul
TOL=1e-3;
% aproximatia initiala x0
x0=3.5;
% numarul maxim de iteratii
iter_maxim = 1000;


%verificare solutie in x_0
if f(x0)==0
   disp('Solutie in punctul x = x_0')
   return % se opreste executia programului
end
%% Rezolvare
% se defineste functia g(x) folosind parametrul m si functia f(x)
g=@(x) x-m*f(x);
```

```
%# Lansare proces iterativ
x1 = x0;

nr_it = 1; %variabila care retine numarul de iteratii
iter= zeros();
fprintf ( ' Nr. iteratie       x(i)              g( x(i) )          |x(i) - x(i-1)|  \n');
fprintf('   \n');

while  nr_it<iter_maxim % s-a limitat nr. maxim de iteratii la 1000
   x2 = g(x1); % x_curent
   Test_conv=abs(x2-x1); % test de conv= modul (x_curent-x_precedent)
   if Test_conv < TOL % s-a atins toleranta de calcul
      fprintf(' %3.0f       %4.16f         %4.16f    %4.16f \n ', nr_it, x2 ,g(x2),Test_conv );
      break
   end
   fprintf(' %3.0f       %4.16f         %4.16f    %4.16f \n ', nr_it, x2 ,g(x2),Test_conv );
   x1 = x2; % x_anterior
   nr_it = nr_it + 1;

end

%% Afisare rezultate
if nr_it==1000
  fprintf('\n Solutia nu s-a putut calcula dupa %2i iteratii \n ', nr_it);
  fprintf('\n Valoarea calculata in ultima iteratie este: %2.10f \n ', x2);
  fprintf('\n Verificare in ecuatia x=g(x): %2.10f <--> %2.10f \n', x2, g(x2));
  return
end

if abs(x2)==inf
  fprintf('\n Solutia nu s-a putut calcula \n ');
  fprintf('\n Verificati datele de intrare \n');
  return
end
fprintf('\n Solutia este: %2.16f \n Testul de convergenta a fost atins dupa %2i iteratii \n ',
x2, nr_it);
%proba in ecuatia x=g(x)
fprintf('\n Proba solutiei in ecuatia x=g(x): %2.16f <--> %2.16f \n', x2, g(x2));
%proba in ecuatia f(x)=0
fprintf('\n Proba solutiei in ecuatie f(rad)=0: %g  \n', f(x2));
```

1.7.3 Aplicație rezolvată

Se va prezenta rezolvarea aplicației 1 descrisă în subcapitolul 1.1, folosind toleranța de calcul $\epsilon = 10^{-3}$. Aproximația inițială se consideră $x_0 = 3.5$. Astfel, se vor putea compara rezultatele cu cele obținute aplicând metoda Newton. Procesul iterativ este prezentat mai jos:

$f(x) = 3.14159 \cdot x^2 \cdot \left(3 - \frac{x}{3}\right) - 80$			
$f'(x) = 6 \cdot 3.14159 \cdot x - 3.14159 \cdot x^2$			
$g(x) = x - m \cdot (3.14159 \cdot x^2 \cdot \left(3 - \frac{x}{3}\right) - 80)$			
x0	f(x0)	f'(x0)	interval pentru *m*
3.5	-9.445124583	27.4889125	(0; 0.072757)

m_ales	0.036

Nr. Iterație	**Valoare iterată**	**$f(x_i)$**	**$g(x_i)$**	**$\lvert x_i - x_{i-1} \rvert$**	**Test de convergenta**
i	**x_i**				**$\lvert x_i - x_{i-1} \rvert < 0.001$**
1	3.843597608	-0.227926	3.851889	0.343597608	Nu
2	3.85188918	-0.012208	3.852333	0.008291576	Nu
3	3.85233330	-0.00066427	3.852357	0.00044412	Da

Utilizând valoarea optimă a parametrului *m*, soluția a fost determinată după 3 iterații, la fel ca în cadrul metodei Newton. Mai mult, se poate observa că valoarea primei iterate este identică în ambele metode, concluzie care e în concordanță cu observația făcută asupra relației (1.58).

Folosind secvențele Matlab, descrise anterior, se obțin următoarele date de ieșire pentru exemplul de test considerat:

- Parametrul *m:*

Parametrul "m" apartine intervalului (0; 0.0727566068683146);

Valoarea "optima" a parametrului m este 0.0363783034341573.

- Proces iterativ:

Nr. iteratie	x(i)	g(x(i))	|x(i) - x(i-1)|
1	3.8435976039652489	3.8518891838680456	0.3435976039652489
2	3.8518891838680456	3.8523333042840653	0.0082915799027967
3	3.8523333042840653	3.8523574691899563	0.0004441204160197

Solutia este: 3.8523333042840653

Testul de convergenta a fost atins dupa 3 iteratii

Proba solutiei in ecuatia x=g(x): 3.8523333042840653 <--> 3.8523574691899563

Proba solutiei in ecuatie f(rad)=0: -0.000664267.

Studii parametrice

În această secțiune se va studia influența calității aproximației inițiale asupra numărului necesar de iterații pentru determinarea soluției. Se vor considera 3 valori pentru aproximația inițială x_0 (3, 3.5 și 4). Analizând rezultatele de mai jos, se poate observa că în cazul aproximațiilor inițiale $x_0 = 3$, respectiv $x_0 = 3.5$, numărul minim de iterații necesare determinării soluției a rezultat în cazul utilizării valorii $1.05 \cdot m_{opt}$ pentru parametrul m. Pentru cazul $x_0 = 4$, numărul minim de iterații a fost generat pentru valoarea optimă a parametrului m.

Date de intrare	**x0**	3				
	toleranta	1.00E-06				
Parametrul m	**interval m**	(0; 0.070735)				
	m_ales	m_opt	0.95*m_opt	0.25*m_opt	1.05*m_opt	1.75*m_opt
		0.0353675	0.01768375	0.008841875	0.05305125	0.061893125
Date de ieșire	**soluție**	3.85235791	3.85235861	3.85235522	3.85235968	3.85235931
	f(soluție)	-1.99E-06	-8.33E-07	7.29E-05	7.43E-07	-7.15E-06
	nr. Iterații	6	8	48	5	30

Date de intrare	**x0**	3.5				
	toleranta	1.00E-06				
Parametrul m	**interval m**	(0; 0.072756)				
	m_ales	m_opt	0.95*m_opt	0.25*m_opt	1.05*m_opt	1.75*m_opt
		0.036378	0.018189	0.0090945	0.054567	0.0636615
Date de ieșire	**soluție**	3.85235879	3.85235859	3.8523548	3.85235935	3.85235839
	f(soluție)	-1.074E-07	-7.29E-07	-8.067E-05	9.12E-08	7.9316E-06
	nr. Iterații	6	7	43	5	33

Date de intrare	**x0**	4				
	toleranta	1.00E-06				
Parametrul m	**interval m**	(0; 0.079578)				
	m_ales	m_opt	0.95*m_opt	0.25*m_opt	1.05*m_opt	1.75*m_opt
		0.039789	0.0198945	0.00994725	0.0596835	0.06963075
Date de ieșire	**soluție**	3.85235897	3.85235889	3.85236203	3.8523583	3.85235932
	f(soluție)	-9.28E-08	1.24E-08	6.11E-05	1.24E-06	-9.65E-06
	nr. Iterații	5	5	37	6	61

1.7.4 Aplicații propuse

P1. Să se rezolve aplicația anterioară folosind $\epsilon = 10^{-5}$ și $x_0 = 7$. Utilizarea valorii optime a parametrului *m* generează soluția cu număr minim de iterații?

P2. Să se determine rădăcina următoarei ecuații neliniare, cu toleranța $\epsilon = 10^{-6}$. Aproximația inițială se va considera 0.5. Folosind valoarea optimă a parametrului *m*, să se compare numărul de iterații cu cel aferent metodei Newton.

$$x^8 - 2 = 0 \tag{1.59}$$

P3. Fie ecuația:

$$e^{-x} - x = 0 \tag{1.60}$$

Să se găsească o soluție a ecuației, acceptând toleranța de calcul $\epsilon = 10^{-4}$.

P4. Să se determine soluția aplicației 2 enunțată în secțiunea 1.1 a acestui capitol. Toleranța de calcul acceptată este $\epsilon = 10^{-3}$. Comparați rezultatele cu cele obținute folosind metoda Newton.

2

REZOLVAREA SISTEMELOR DE ECUAȚII NELINIARE

2.1 Introducere

În capitolul anterior s-au prezentat o serie de metode numerice pentru rezolvarea ecuațiilor neliniare. O problematică conexă este reprezentată de determinarea soluției (soluțiilor) unui sistem de n ecuații neliniare cu n necunoscute, care poate fi exprimat în formă algebrică astfel:

$$\begin{cases} f_1(x_1, x_2, \dots, x_n) = 0 \\ f_2(x_1, x_2, \dots, x_n) = 0 \\ \quad\quad\quad\vdots \\ f_n(x_1, x_2, \dots, x_n) = 0 \end{cases} \tag{2.1}$$

Sistemul (2.1) poate fi scris în formă vectorială astfel:

$$\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{0} \tag{2.2}$$

unde:

$$\boldsymbol{X} = (x_1\ x_2\ \dots\ x_n)^T \tag{2.3}$$

$$\boldsymbol{F}(\boldsymbol{X}) = (f_1(\boldsymbol{X}) \quad f_2(\boldsymbol{X}) \quad \dots \quad f_n(\boldsymbol{X}))^T \tag{2.4}$$

Mai jos se prezintă un exemplu generic de un sistem format din 2 ecuații neliniare [9], în necunoscutele x și y:

$$\begin{cases} x^2 + y^2 - 4 = 0 \\ e^x + y - 1 = 0 \end{cases} \tag{2.5}$$

Din punct de vedere grafic, soluțiile sistemului (2.2) sunt reprezentate de punctele de intersecție dintre cercul de ecuație $x^2 + y^2 = 4$ și curba de ecuație $y = -e^x + 1$. Analizând Figura

19, se observă că soluțiile se află în proximitatea punctelor de coordonate (-2; 1), respectiv (1; 1.5). Aceste valori pot fi folosite ca aproximații inițiale în cadrul metodelor numerice de rezolvare a sistemelor de ecuații neliniare. Totuși, metoda grafică de identificare a aproximațiilor inițiale își pierde semnificația când sistemul are mult de 3 ecuații. În aceste situații, identificarea unor aproximații inițiale rezonabile este posibilă prin interpretarea fizică a problemei simulate prin sistemul de ecuații. Majoritatea metodelor dedicate rezolvării iterative a sistemelor de ecuații neliniare converg spre soluție dacă aproximațiile inițiale sunt alese suficient de aproape de soluția exactă.

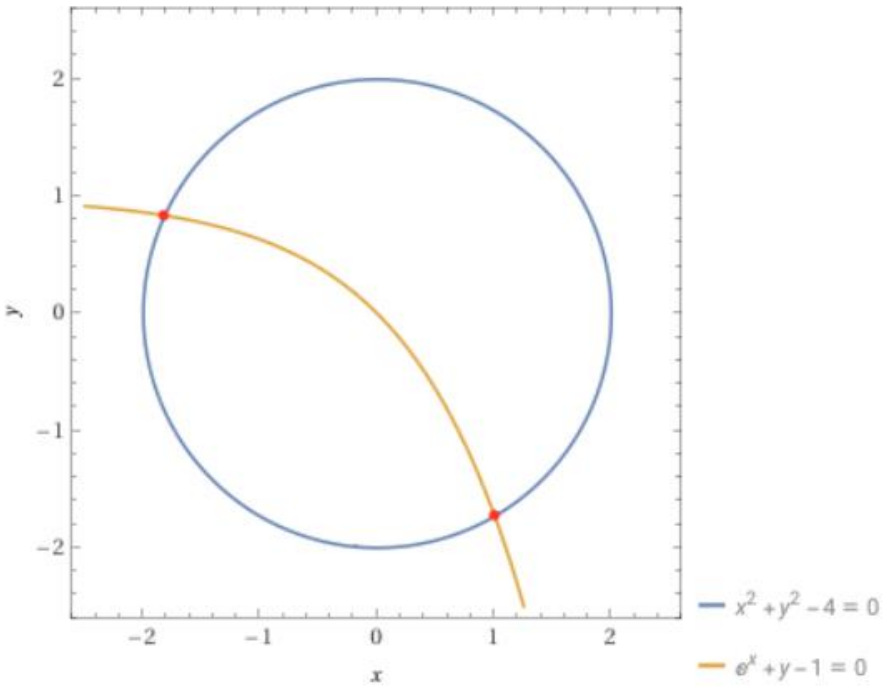


Figura 19. Reprezentarea grafică a ecuațiilor sistemului (2.2).

În continuare, se enumeră o serie de aplicații a căror soluționare implică rezolvarea unor sisteme de ecuații neliniare.

<u>Aplicația 1</u>. [10]

Traiectoriile a două corpuri (Figura 20) sunt descrise de următoarele ecuații parametrice:

$$\text{Corp 1}\begin{cases} x_1 = 200 \cdot t \\ y_1 = 80 \cdot t - 16 \cdot t^2 \end{cases} \qquad \text{Corp 2}\begin{cases} x_2 = 800 - 100 \cdot t \cdot \cos(\alpha) \\ y_2 = 50 \cdot t \cdot \sin(\alpha) - 0.8 \cdot t^2 \end{cases} \tag{2.6}$$

unde t reprezintă timpul de la lansare, iar α reprezintă unghiul de lansare al corpului 2. Așa cum se observă în Figura 20 traiectoriile celor două corpuri se intersectează, însă corpurile

vor intra în coliziune doar dacă ajung simultan în punctul teoretic de intersecție. Să se determine valorile *t și α* pentru care corpurile se ciocnesc.

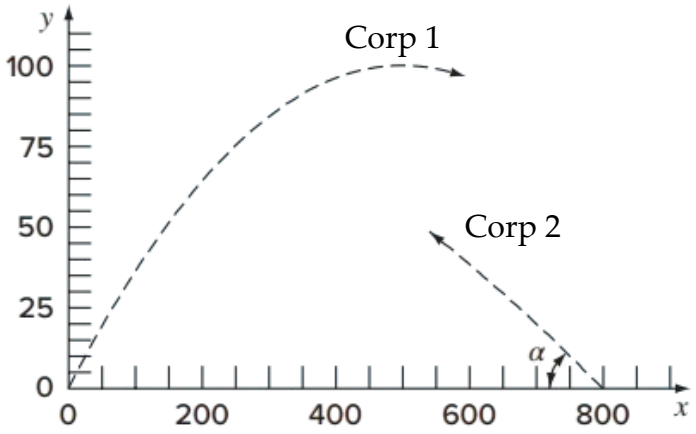


Figura 20.

Aplicația 2.

Fie următorul sistem de ecuații neliniare [9]:

$$\begin{cases} x - 3 \cdot y - z^2 = -3 \\ 2 \cdot x^3 + y - 5 \cdot z^2 = -2 \\ 4 \cdot x^2 + y + z = 7 \end{cases} \tag{2.7}$$

Două dintre punctele de intersecție ale suprafețelor tridimensionale generate de ecuațiile de mai sus sunt în proximitatea punctelor de coordonate (1; 1; 1), respectiv (1.3; 0.9; -1.2). Să se determine cele două puncte de intersecție.

Aplicația 3.

Pentru un stâlp metalic solicitat ca în Figura 21, relația de interacțiune neliniară efort axial N – moment încovoietor M este următoarea:

$$\frac{N}{N_{pl}} + \left(\frac{M}{M_{pl}}\right)^2 \leq 1 \tag{2.8}$$

Considerând efectul $P - \Delta$, momentul încovoietor la baza stâlpului poate fi evaluat folosind următoarea relație aproximativă [11]:

$$M = \frac{1}{1 - \frac{P \cdot L^2}{12 \cdot E \cdot I}} \cdot H \cdot L \tag{2.9}$$

unde:

- L – reprezintă lungimea stâlpului;
- I – reprezintă momentul de inerție în raport cu axa după care are loc rotirea secțiunii;
- E – reprezintă modulul de elasticitate longitudinal al materialului.

Să se calculeze valorile M și $N \equiv P$, pentru care stâlpul verifică la limită relația (2.8). Se cunosc: $L = 6\,m,\ \ I = 18260\ \text{cm}^4,\ \ E = 210000\ MPa,\ \ N_{pl} = 2644\ kN,\ \ M_{pl} = 325\ kNm,\ \ H = 30\ kN$.

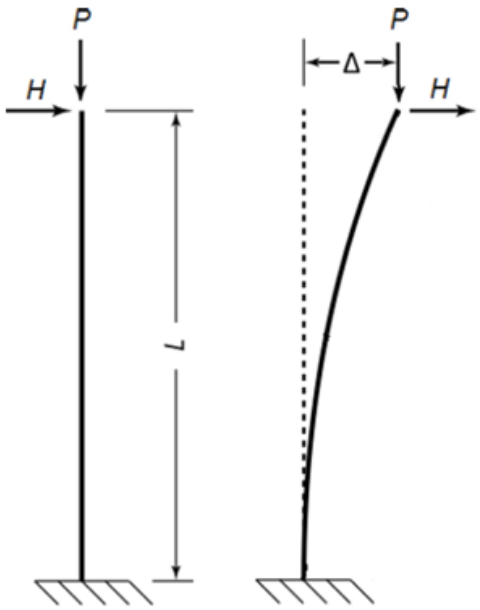


Figura 21.

2.2 Metoda Newton

2.2.1 Noțiuni teoretice

Dintre metodele dedicate rezolvării sistemelor de ecuații neliniare, în continuare se prezintă metoda Newton, aceasta fiind una dintre metodele care s-au impus în problematica studiată în acest capitol.

Fie:

$$\boldsymbol{X}^{(\boldsymbol{e})} = (x_1^{(e)} \;\; x_2^{(e)} \;\; \ldots \;\; x_n^{(e)})^T \tag{2.10}$$

soluția exactă a sistemului (2.2), iar:

$$\boldsymbol{X}^{(\boldsymbol{k})} = (x_1^{(k)} \;\; x_2^{(k)} \;\; \ldots \;\; x_n^{(k)})^T \tag{2.11}$$

o soluție aproximativă a aceluiași sistem. In aceste condiții poate fi scrisă următoarea relație:

$$\boldsymbol{X}^{(\boldsymbol{e})} = \boldsymbol{X}^{(\boldsymbol{k})} + \boldsymbol{\varepsilon}^{(\boldsymbol{k})} \tag{2.12}$$

unde:

$$\boldsymbol{\varepsilon}^{(\boldsymbol{k})} = (\varepsilon_1^{(k)} \;\; \varepsilon_2^{(k)} \;\; \ldots \;\; \varepsilon_n^{(k)})^T \tag{2.13}$$

reprezintă eroarea dintre soluția exactă $\boldsymbol{X}^{(\boldsymbol{e})}$ și aproximația $\boldsymbol{X}^{(\boldsymbol{k})}$. Cu aceste notații sistemul (2.2) poate fi scris în următoarea formă echivalentă:

$$\boldsymbol{F}\left(\boldsymbol{X}^{(\boldsymbol{k})} + \boldsymbol{\varepsilon}^{(\boldsymbol{k})}\right) = \boldsymbol{0} \tag{2.14}$$

Membrul stâng al egalității (2.14) poate fi aproximat prin considerarea primilor doi termeni din dezvoltarea în serie Taylor a funcției vectoriale $\boldsymbol{F}(\boldsymbol{X})$ în vecinătatea punctului $\boldsymbol{X}^{(\boldsymbol{k})}$. Rezultă astfel un sistem de ecuații liniare din care poate fi evaluat vectorul erorilor $\boldsymbol{\varepsilon}^{(\boldsymbol{k})}$. Urmând procedura descrisă în [7] se ajunge la următoarea relație matriceală de iterare:

$$\boldsymbol{X}^{(\boldsymbol{k+1})} = \boldsymbol{X}^{(\boldsymbol{k})} - \left[\boldsymbol{J}\left(\boldsymbol{X}^{(\boldsymbol{k})}\right)\right]^{-\boldsymbol{1}} \cdot \boldsymbol{F}\left(\boldsymbol{X}^{(\boldsymbol{k})}\right) \tag{2.15}$$

sau, în formă matriceală extinsă:

$$\begin{bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \\ \cdot \\ \cdot \\ \cdot \\ x_n^{(k+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \\ \cdot \\ \cdot \\ \cdot \\ x_n^{(k)} \end{bmatrix} - \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdot & \cdot & \cdot & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdot & \cdot & \cdot & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdot & \cdot & \cdot & \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{\boldsymbol{X}^{(k)}} \cdot \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)} \\ \cdot \\ \cdot \\ \cdot \\ f_n(x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)} \end{bmatrix} \tag{2.16}$$

Procesul iterativ se oprește când următorul test de convergență este îndeplinit:

$$\left\| \boldsymbol{X}^{(k+1)} - \boldsymbol{X}^{(k)} \right\| \leq \varepsilon \tag{2.17}$$

Observații:

- în relația (2.15), matricea $\boldsymbol{J}\left(\boldsymbol{X}^{(k)}\right)$ este jacobianul lui $\boldsymbol{X}^{(k)}$ și cuprinde valorile derivatelor parțiale ale funcțiilor f_i în raport cu variabilele x_i, așa cum se poate observa în relația (2.16). Elementele matricei jacobian pot fi evaluate analitic pe baza expresiilor derivatelor de ordinul întâi sau numeric, generalizând procedeul descris în secțiunea 1.6.1. Pentru a putea fi inversată, matricea $\boldsymbol{J}\left(\boldsymbol{X}^{(k)}\right)$ trebuie să fie nesingulară.
- în cadrul fiecărei iterații, evaluarea matricei jacobian presupune efectuarea a n^2 operații, motiv pentru care metoda Newton este costisitoare și ineficientă în cazul sistemelor de mari dimensiuni. O variantă modificată a metodei Newton presupune folosirea aceleași matrice jacobian în mai multe iterații succesive. În acest fel convergența pătratică este afectată, însă numărul de evaluări per iterație este diminuat, astfel micșorând-se și timpul efectiv de calcul.
- în vederea evaluării relației (2.17) se poate utiliza norma euclidiană a unui vector:

$$\|\boldsymbol{X}\| = \sqrt{\sum_{i=1}^{n} x_i^2} \tag{2.18}$$

2.2.2 Aplicație Matlab

- Varianta 1: elementele jacobinului sunt evaluate analitic, exact:

```
clc
clearvars
```

```
%% date de intrare
% vectorul functiilor F(x)=0
f = @(x) [x(1)^2+x(2)^2-4
          x(2)+exp(x(1))-1];
% Jacobianul sistemului de ecuatii
df = @(x) [2*x(1)         2*x(2)
           exp(x(1))       1];
% vectorul aproximatiilor initiale
x = [-2; 1];
% toleranta de calcul
x_tol=1e-3;

%% Programul principal
i=0;
eps_calc=1e3; % initializare toleranta calculata = 1000
dim=size(x,1);
while eps_calc>=x_tol
   x1 = x;
   x = x  -df(x)\f(x); % formula de iterare
   eps_calc=norm(x1-x, inf); % s-a utilizat norma liniilor
   i=i+1;
   disp('  ');
   fprintf ( 'Iteratia nr %4.0f  \n', i);
   for j1=1:dim
      fprintf ( 'Aproximatia x(%2.0f) = %2.20f  \n', j1,x(j1));
   end
end
disp('  ');
disp ('   ');
fprintf ( 'Solutia calculata in precizia ceruta s-a obtinut dupa %2.0f iteratii:  \n', i);

for i2=1:dim
   fprintf ( 'Solutia x(%2.0f)= %2.20f  \n',i2,x(i2))
end

%proba solutiei F(x) = 0 ?
disp ('   ');
disp('Proba solutiei F(rad)=0:')
rezultat =  f(x);

for k1=1:size(x,1)
   fprintf ( '  Ecuatia %1.0f: %g  \n', k1, rezultat(k1));
```

```
end

disp ('   ');
fprintf ( 'EPS calculat la ultima iteratie = %g   \n', eps_calc);
```

- Varianta 2: elementele jacobinului sunt evaluate numeric, aproximativ:

```
clc
clearvars

%% date de intrare
% vectorul functiilor F(x)=0
f = @(x)[ x(1)^2+x(2)^2-4
        x(2)+exp(x(1))-1];
% vectorul aproximatiilor initiale
x = [-2; 1];
% toleranta de calcul
x_tol=1e-3;
% interval calcul derivata numerica
h= 1e-6;

%% Programul principal
i=0;
eps_calc=1e3; % initializare toleranta calculata = 1000
dim=size(x,1);
while eps_calc >= x_tol
   x1 = x;

   x = x  - Jacobian_numeric (f,x,h) \ f(x); % formula de iterare
   eps_calc=norm(x1-x, inf); % s-a utilizat norma liniilor
   i=i+1;
   disp('  ');
   fprintf ( 'Iteratia nr %4.0f   \n', i);
   for j1=1:dim
      fprintf ( 'Aproximatia x(%2.0f) = %2.20f   \n', j1,x(j1));
   end
end
disp('  ');
disp ('   ');
fprintf ( 'Solutia calculata in precizia ceruta s-a obtinut dupa %2.0f iteratii:   \n', i);

for i2=1:dim
```

```
   fprintf ( 'Solutia x(%2.0f)= %2.20f  \n',i2,x(i2))
end

%proba solutiei F(x) = 0 ?
disp ('  ');
disp('Proba solutiei F(rad)=0:')
rezultat =  f(x);

for k1=1:size(x,1)
   fprintf ( '  Ecuatia %1.0f: %g  \n', k1, rezultat(k1));
end

disp ('  ');
fprintf ( 'EPS calculat la ultima iteratie = %g  \n', eps_calc);

function J = Jacobian_numeric(functie, x, h)
   % func: vectorul f
   % x: vectorul x
   % h: interval derivare

   n = length(x);
   F = functie(x);
   m = length(F);
   J = zeros(m, n);

   for j1 = 1:n
      e = zeros(n, 1);
      e(j1) = 1;

      x_forward = x + h * e;
      x_backward = x - h * e;

      F_forward = functie(x_forward);
      F_backward = functie(x_backward);

      J(:, j1) = (F_forward - F_backward) / (2 * h);
   end
end
```

2.2.3 Aplicație rezolvată

În continuare se prezintă rezolvarea sistemului de ecuații neliniare introdus în secțiunea 2.1. În Figura 19 se poate observa că o soluție a sistemului se află în apropierea punctului de coordonate (-2; 1) și astfel procesul iterativ se va iniția folosind aceste aproximații inițiale. Rezolvarea, pas cu pas, realizată în Microsoft Excel este prezentată tabelar mai jos. Se observă că testul de convergență a fost atins după 3 iterații, toleranța de calcul acceptată fiind setată la valoarea 1E-3. Soluția, cu 5 cifre semnificative după virgulă este reprezentată de perechea de valori: *x = -1.816264*, respectiv *y = 0.837368*.

$f_1(x,y) = x^2 + y^2 - 4$	
$f_2(x,y) = \exp(x) + y - 1$	
$J = \begin{bmatrix} 2x & 2y \\ \exp(x) & 1 \end{bmatrix}$	elementele matricei jacobian
$X^{(0)} = \begin{bmatrix} -2 \\ 1 \end{bmatrix}$	aproximația inițială

k	$X^{(k)}$	$F(X^{(k)})$	$J(X^{(k)})$		$\det(J(X^{(k)}))$	$J(X^{(k)})^{-1}$		$X^{(k+1)}$	$X^{(k+1)} - X^{(k)}$	$\|X^{(k+1)} - X^{(k)}\|$	Test de convergenta
0	-1.8	0.24	-3.6	2	-3.93060	-0.25441	0.50883	-1.82305	-0.02305	0.16313	Nu
	1	0.16530	0.165299	1		0.04205	0.91589	0.83851	-0.16149		
1	-1.823049	0.02661	-3.64610	1.67702	-3.91699	-0.25530	0.42814	-1.81627	0.00677	0.00687	Nu
	0.838511	0.00004	0.161532	1		0.04124	0.93084	0.83737	-0.00114		
2	-1.816275	0.00005	-3.63255	1.67475	-3.90491	-0.25609	0.42888	-1.816264	0.00001	0.00001	Da
	0.837373	3.7E-06	0.16263	1		0.04165	0.93025	0.837368	-0.00001		

Folosind secvențele Matlab, cu cele două variante de evaluarea a elementelor matricei jacobian, se obțin următoarele rezultate:

- cu evaluare analitică a jacobianului:

Iteratia nr 1

Aproximatia x(1) = -1.82922367291629717201

Aproximatia x(2) = 0.84155265416740587803

Iteratia nr 2

Aproximatia x(1) = -1.81630547632188310558

Aproximatia x(2) = 0.83738798709927619868

Iteratia nr 3

Aproximatia x(1) = -1.81626406930879258361

Aproximatia x(2) = 0.83736780010931932683

Solutia calculata in precizia ceruta s-a obtinut dupa 3 iteratii:

Solutia x(1)= -1.81626406930879258361

Solutia x(2)= 0.83736780010931932683

Proba solutiei F(rad)=0:

Ecuatia 1: 2.12206e-09

Ecuatia 2: 1.39416e-10

Se observă că rezultatele sunt identice cu cele evaluate pas cu pas. Suplimentar, proba soluției, certifică corectitudinea acesteia.

- cu evaluare numerică a jacobianului:

Iteratia nr 1

Aproximatia x(1) = -1.82922367293462939664

Aproximatia x(2) = 0.84155265416271018974

Iteratia nr 2

Aproximatia x(1) = -1.81630547632105932010

Aproximatia x(2) = 0.83738798709890849281

Iteratia nr 3

Aproximatia x(1) = -1.81626406930878969703

Aproximatia x(2) = 0.83736780010931799456

Solutia calculata in precizia ceruta s-a obtinut dupa 3 iteratii:

Solutia x(1)= -1.81626406930878969703

Solutia x(2)= 0.83736780010931799456

Proba solutiei F(rad)=0:

Ecuatia 1: 2.12204e-09

Ecuatia 2: 1.39415e-10

Comparând aceste rezultate cu cele obținute în varianta calculului explicit al elementelor jacobianului, se observă ca diferențe foarte mici apar la ultimele zecimale ale celor două soluții. Astfel, evaluarea numerică a derivatelor de ordin întâi este preferată, mai ales în situația în care expresiile exacte ale derivatelor sunt de complexitate ridicată.

2.2.4 Aplicații propuse

P1. Utilizând secvențele Matlab, se cere rezolvarea aplicației anterioare considerând $\epsilon = 10^{-5}$ și aproximația inițială $X^{(0)} = [1 \quad -1.5]^T$.

P2. Să se rezolve aplicația 1 descrisă în secțiunea 2.1. Toleranța de calcul acceptată este $\epsilon = 10^{-4}$.

P3. Să se rezolve sistemul (2.7), știind că $X^{(0)} = [1 \quad 1 \quad 1]^T$ și $\epsilon = 10^{-6}$.

P3. Folosind programul Matlab cu evaluare numerică a elementelor matricei jacobian, să se rezolve aplicația 4 din secțiune 2.1. Toleranța de calcul acceptată este $\epsilon = 10^{-3}$.

3

REZOLVAREA SISTEMELOR DE ECUAȚII LINIARE

3.1 Introducere

Fie următorul sistem de n ecuații liniare:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \qquad\qquad\qquad\qquad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases} \tag{3.1}$$

Necunoscutele sistemului sunt notate cu $x_i;\ i = \overline{1,n}$, coeficienții necunoscutelor sun notați cu $a_{ij}; i,j = \overline{1,n}$, iar termenii liberi cu $b_i;\ i = \overline{1,n}$. Separând termenii de același fel, sistemul (3.1) poate fi scris în următoarea formă matriceală extinsă:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{3.2}$$

sau, mai departe, în formă matriceală condensată:

$$\boldsymbol{A} \cdot \boldsymbol{X} = \boldsymbol{B} \tag{3.3}$$

unde:

A - poartă denumirea de matricea coeficienților;

X - este vectorul necunoscutelor;

B - este vectorul termenilor liberi.

Pentru ca sistemul de ecuații să aibă soluție (să fie rezolvabil), determinantul matricei coeficienților trebuie să fie diferit de 0. Mai mult, pentru a accepta soluție nebanală, termenii

liberi nu pot fi simultan nuli. Algebra elementară cuprinde metode pentru rezolvarea sistemelor de mici dimensiuni (Regula Cramer, substituții, eliminarea succesivă a necunoscutelor). Totuși, pentru n>4, aceste metode devin prohibitive, iar aplicarea lor într-un calcul manual este laborioasă, chiar imposibilă. În consecință, trebuie utilizate metode alternative mai eficiente, care pot utiliza puterea calculatoarelor pentru determinarea soluțiilor. Metodele de rezolvare a sistemelor de ecuații liniare pot și împărțite în două categorii: metode directe (introduc doar erori de rotunjire, trunchiere), respectiv metode iterative (soluția este acceptată în baza unui test de convergență, care permite o toleranță de calcul). În secțiunile următoare se vor descrie succint metoda eliminării Gauss, metoda Cholesky (metode directe) și metoda Jacobi (iterativă). Capitolul se încheie cu o discuție privind stabilitatea soluțiilor sistemelor de ecuații.

Sistemele de ecuații liniare intervin frecvent în studiul diferitelor probleme din sfera ingineriei și nu numai, având aplicabilitate ridicată și în analiza structurală. In continuare se prezintă o serie de aplicații, a căror rezolvare implică soluționarea unor sisteme de ecuații liniare.

Aplicația 1

Exprimând condițiile de echilibru static pentru fiecare corp al structurii din figura de mai jos, se obține următorul set de ecuații liniare:

$$\begin{cases} 62.5 - H_D = 0 \\ 2 \cdot V_D - 5 \cdot H_D + 168.33 = 0 \\ 2 \cdot V_A + 64.17 = 0 \\ H_D - H_B - H_E + 12.86 = 0 \\ -V_D + V_B + V_E - 96.32 = 0 \\ 4.2 \cdot V_D - 5 \cdot H_D + 5 \cdot H_E + 1.2 \cdot V_E + M_B + 16.45 = 0 \\ H_E = 0 \\ 2 \cdot V_F - 15 = 0 \\ 2 \cdot V_E - 15 = 0 \end{cases} \tag{3.4}$$

Rezolvând sistemul de ecuații (3.4) se determină reacțiunile din legăturile interioare și exterioare. Se poate observa că mulți dintre coeficienții sistemului de mai sus sunt nuli și astfel sistemul poate fi rezolvat relativ ușor și printr-un calcul manual.

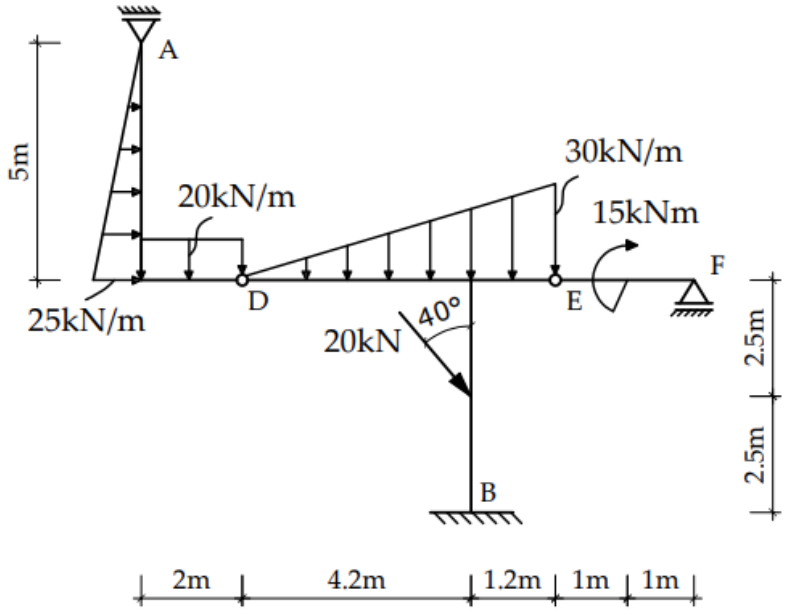


Figura 22.

Aplicația 2

Fie sistemul de bare articulate din Figura 23. Legătura dintre forțele aplicate și deplasările nodale este dată de următoarea relație matriceală:

$$\boldsymbol{K_s} \cdot \boldsymbol{D} = \boldsymbol{F} \tag{3.5}$$

unde:

$\boldsymbol{K_s}$ reprezintă matricea de rigiditate a structurii;

$\boldsymbol{D}$ este vectorul deplasărilor nodale și reprezintă necunoscutele problemei;

$\boldsymbol{F}$ este vectorul forțelor exterioare aplicate pe direcția gradelor de libertate și are următoarele elemente exprimate în kN: $\boldsymbol{F} = [0 \quad -110 \quad 0 \quad -110 \quad 0 \quad 0]^T$.

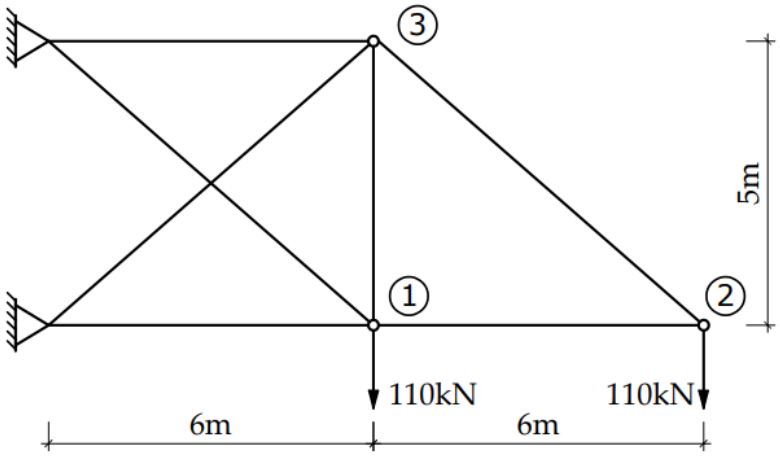


Figura 23.

Matricea de rigiditate a structurii este următoarea:

$$K_S = \begin{pmatrix} 4.055\times 10^5 & -4.912\times 10^4 & -1.733\times 10^5 & 0 & 0 & 0 \\ -4.912\times 10^4 & 1.969\times 10^5 & 0 & 0 & 0 & -1.56\times 10^5 \\ -1.733\times 10^5 & 0 & 2.519\times 10^5 & -6.549\times 10^4 & -7.859\times 10^4 & 6.549\times 10^4 \\ 0 & 0 & -6.549\times 10^4 & 5.457\times 10^4 & 6.549\times 10^4 & -5.457\times 10^4 \\ 0 & 0 & -7.859\times 10^4 & 6.549\times 10^4 & 3.108\times 10^5 & -1.637\times 10^4 \\ 0 & -1.56\times 10^5 & 6.549\times 10^4 & -5.457\times 10^4 & -1.637\times 10^4 & 2.515\times 10^5 \end{pmatrix} \cdot \frac{kN}{m} \quad (3.6)$$

Se cere determinarea deplasărilor nodale.

Observații:

- vectorul ***D*** are 6 componente. Primele două reprezintă deplasarea orizontală, respectiv verticală a nodului 1. Dacă sunt pozitive atunci nodul 1 se deplasează spre dreapta și în sus. Elementele 3 și 4 ale vectorului ***D*** sunt deplasările nodului 2, iar ultimele două elemente caracterizează poziția nodului 3;
- rezultatele se obțin în *m*.

3.2 Metoda eliminării Gauss

3.2.1 Noțiuni teoretice

În cadrul acestei metode, rezolvarea sistemului se face în două etape.

Etapa 1 – de triangularizare (eliminare)

Prin operații elementare, matricea coeficienților se transformă într-o matrice superior triunghiulară, deci elementele de sub diagonala principală devin nule. Această operație se realizează în *n-1* pași. Pentru evidențierea procesului de eliminare, se formează următoarea matrice extinsă:

$$[\boldsymbol{A}|\boldsymbol{B}] = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & & a_{2n} & b_2 \\ \vdots & & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array}\right] \tag{3.7}$$

Pas 1

Se elimina coeficienții de pe prima coloană, de sub diagonala principală. Astfel:

Ecuația 1 – rămâne neschimbată

Ecuația 2 $\rightarrow$ Ecuația 2 $- \frac{a_{21}}{a_{11}} \cdot$ Ecuația 1;

Ecuația 3 $\rightarrow$ Ecuația 3 $- \frac{a_{31}}{a_{11}} \cdot$ Ecuația 1;

....

Ecuația n $\rightarrow$ Ecuația n $- \frac{a_{n1}}{a_{11}} \cdot$ Ecuația 1;

Prin aceste operații elementare matricea extinsă (3.7) suferă următoarele modificări:

$$\boldsymbol{A}|\boldsymbol{B} = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} - \frac{a_{21}}{a_{11}} a_{11} & a_{22} - \frac{a_{21}}{a_{11}} a_{12} & \cdots & a_{2n} - \frac{a_{21}}{a_{11}} a_{1n} & b_2 - \frac{a_{21}}{a_{11}} b_1 \\ \vdots & & \ddots & \vdots & \vdots \\ a_{n1} - \frac{a_{n1}}{a_{11}} a_{11} & a_{n2} - \frac{a_{n1}}{a_{11}} a_{12} & \cdots & a_{nn} - \frac{a_{n1}}{a_{11}} a_{1n} & b_n - \frac{a_{n1}}{a_{11}} b_1 \end{array}\right] \tag{3.8}$$

sau, mai departe:

$$\boldsymbol{A}|\boldsymbol{B}^{(1)} = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} & b_n^{(1)} \end{array}\right] \tag{3.9}$$

Observații:

- în acest pas, coeficientul a_{11} se numește pivot și trebuie să fie diferit de 0. Dacă nu îndeplinește această condiție, atunci se pot face permutări de ecuații pentru a deveni nenul.
- în relația (3.9), exponentul (1) alăturat coeficienților se referă la pasul 1.

Pas 2

Se elimina coeficienții de pe a doua coloană, de sub diagonala principală. Pivotul va fi $a_{22}^{(1)}$ dacă este nenul. În caz contrar, se fac permutări între ecuația 2 și următoarele, astfel încât pivotul să devină nenul. Procesul de eliminare se conduce într-o manieră similară celei prezentate mai sus.

După *n-1* pași, matricea extinsă va avea următoarea formă:

$$\boldsymbol{A|B^{(n-1)}} = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & a_{nn}^{(n-1)} & b_n^{(n-1)} \end{array}\right] \tag{3.10}$$

Se observă că elementele de sub diagonala principală a matricei coeficienților sunt nule. Pentru simplitate în notații, elementele matricei coeficienților se notează cu u_{ij}, iar elementele vectorului liber cu b_i'. Astfel, la finalul etapei 1, sistemul de ecuații devine:

$$\begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1,n-1} & u_{1n} \\ 0 & u_{22} & \cdots & u_{2,n-1} & u_{2n} \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & u_{n-1,n-1} & u_{n-1,n} \\ 0 & 0 & \cdots & 0 & u_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1' \\ b_2' \\ \vdots \\ b_{n-1}' \\ b_n' \end{bmatrix} \tag{3.11}$$

sau, în formă matriceală condensată:

$$\boldsymbol{U \cdot X = B'} \tag{3.12}$$

unde $\boldsymbol{U}$ este o matrice superior triunghiulară.

Etapa 2 – de rezolvare

Având în vedere forma particulară a matricei coeficienților la finalul etapei 1, necunoscutele sistemului pot fi determinate prin substituție înapoi, deci începând rezolvarea cu ultima ecuație a sistemului (3.11). Astfel:

$$\text{Ecuația n: } u_{nn} \cdot x_n = b_n' \tag{3.13}$$

de unde:

$$x_n = \frac{b'_n}{u_{nn}} \tag{3.14}$$

Din penultima ecuație se calculează x_{n-1}:

$$x_{n-1} = \frac{b'_{n-1} - u_{n-1,n} \cdot \frac{b'_n}{u_{nn}}}{u_{n-1,n-1}} \tag{3.15}$$

Continuând substituția înapoi se determină toate necunoscutele sistemului.

Observații:

- determinantul matricei $\boldsymbol{A}$ este egal cu determinantul matricei $\boldsymbol{U}$, iar acesta din urmă poate fi evaluat cu următoarea relație:

$$\det(\boldsymbol{U}) = u_{11} \cdot u_{22} \cdot \ldots \cdot u_{nn} \tag{3.16}$$

- pentru sisteme având număr mare de ecuații, numărul secvențelor de calcul necesare rezolvare sistemului cu metoda eliminării Gauss este [10]:

$$nr.\ operații_{Gauss} \cong \frac{2 \cdot n^3}{3} \tag{3.17}$$

- în etapa de triangularizare se modifică și termenii liberi ai sistemului de ecuații. Acest aspect reprezintă un dezavantaj al metodei Gauss, deoarece există situații în care se dorește rezolvarea succesivă a unui sistem de ecuații pentru diferite seturi de termeni liberi (de exemplu, în calculul inversei unei matrice, așa cum se va arată în cadrul secțiunii 3.2.3).

3.2.2 Aplicație Matlab

```
clearvars
clc

%Date de intrare
%Matricea coeficientilor A.
A= [1  1 2
    3  2 1
    1 -2 3];
%Vectorul termenilor liberi B (se trec valorile pe linie)
B=[5 8 0];
```

```
%% verificare dimensiuni matrice
[n, m] = size(A);
   if n ~= m
      error('Matricea A trebuie să fie pătratică.');
   end
   if length(B) ~= n
      error('Dimensiunea vectorului b trebuie să corespundă cu matricea A.');
   end

%start proces eliminare
B = B'; % se transpune vectorul B
AB = [A B]; %matrice extinsa formata din A si B
[n, m] = size(AB);
i1 = 1; % initializare
while i1 <= n
   if AB(i1,i1) == 0 % se verifica daca termenul de pe diag principala are valoarea egala cu 0
      disp('Pivot nul in a matricea coeficientilor [A]. Modificati sistemul de ecuatii!')
          % se afisează mesajul dacă avem termen egal cu zero pe diag principala
      fprintf ('Eroarea este la termenul a(%1.0f,%1.0f)\n', i1,i1);
      return
   end
   % eliminarea elementelor de pe coloana i, de sub diagonala principala
   for k = (i1+1):n
      fractie=AB(k,i1)/AB(i1,i1);
      AB(k,i1:m) = AB(k,i1:m) - fractie*AB(i1,i1:m);
   end
   i1 = i1 +1;
end  %final proces eliminare

U=AB(1:n,1:n); %matricea superior triunghiulara
B1=AB(:,m); %termenii liberi dupa procesul de eliminare

%determinare solutie prin substitutie inapoi
x(n)=AB(n,m)/AB(n,n);
for i1=n-1:-1:1
   suma=B1(i1);
   for j1=i1+1:n
      suma=suma-U(i1,j1)*x(j1);
   end
   x(i1)=suma/U(i1,i1);
end
```

```
%afisare rezultate
disp('Matricea superior triunghiulara este:');
disp (U);

disp('Solutia sistemului de ecuatii este:')
for k=1:n
   fprintf ( '   x(%1.0f) = %2.10f   \n',k,x(k))
end
%proba solutiei; practic se face operatia A*X-B
disp ('   ');
disp('Proba solutiei A*X-B=0:')
%fprintf ( ' %2.20f   \n', A*x'-B');
for k1=1:n
   fprintf ( '  Ecuatia %1.0f: %g   \n', k1, A(k1,:)*x'-B(k1));
end
```

Observație: aplicația Matlab nu face pivotare (interschimbare de ecuații) dacă se identifică pivot nul.

3.2.3 Aplicație rezolvată

Pentru exemplificarea operațiilor ce trebuie parcurse pentru rezolvarea unui sistem de ecuații liniare cu metoda Gauss, se consideră următorul exemplu:

$$\begin{cases} x_1 + x_2 + 2x_3 = 5 \\ 3x_1 + 2x_2 + x_3 = 8 \\ x_1 - 2x_2 + 3x_3 = 0 \end{cases} \tag{3.18}$$

Matricea extinsă este:

$$\boldsymbol{A|B} = \left[\begin{array}{ccc|c} 1 & 1 & 2 & 5 \\ 3 & 2 & 1 & 8 \\ 1 & -2 & 3 & 0 \end{array}\right] \tag{3.19}$$

Etapa 1 – Pas 1:

Ecuația 1 – rămâne neschimbată

Ecuația 2 → Ecuația 2 − 3 · Ecuația 1;

Ecuația 3 → Ecuația 3 − 1 · Ecuația 1;

Matricea extinsă devine:

$$\boldsymbol{A}|\boldsymbol{B}^{(1)} = \begin{bmatrix} 1 & 1 & 2 & 5 \\ 3-3\cdot 1 & 2-3\cdot 1 & 1-3\cdot 2 & 8-3\cdot 5 \\ 1-1\cdot 1 & -2-1\cdot 1 & 3-1\cdot 2 & 0-1\cdot 5 \end{bmatrix} = \left[\begin{array}{ccc|c} 1 & 1 & 2 & 5 \\ 0 & -1 & -5 & -7 \\ 0 & -3 & 1 & -5 \end{array}\right] \tag{3.20}$$

Pas 2:

Ecuația 1 – rămâne neschimbată

Ecuația 2 – rămâne neschimbată

Ecuația 3 → Ecuația 3 − 3 · Ecuația 2;

Matricea extinsă devine:

$$\boldsymbol{A}|\boldsymbol{B}^{(2)} = \left[\begin{array}{ccc|c} 1 & 1 & 2 & 5 \\ 0 & -1 & -5 & -7 \\ 0 & -3-3(-1) & 1-3(-5) & -5-3(-7) \end{array}\right] = \left[\begin{array}{ccc|c} 1 & 1 & 2 & 5 \\ 0 & -1 & -5 & -7 \\ 0 & 0 & 16 & 16 \end{array}\right] \tag{3.21}$$

La finalul etapei 1 sistemul de ecuații are următoarea formă:

$$\begin{bmatrix} 1 & 1 & 2 \\ 0 & -1 & -5 \\ 0 & 0 & 16 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5 \\ -7 \\ 16 \end{bmatrix} \tag{3.22}$$

Etapa 2:

Din ecuația 3 de obține:

$$x_3 = 1 \tag{3.23}$$

Folosind această valoare în ecuația 2, obținem:

$$x_2 = 2 \tag{3.24}$$

In final, din ecuația 1 rezultă:

$$x_1 = 1 \tag{3.25}$$

Aplicând secvența Matlab, se afișează următoarele:

Matricea superior triunghiulara este:

1 1 2

0 -1 -5

0 0 16

Solutia sistemului de ecuatii este:

x(1) = 1.0000000000

x(2) = 2.0000000000

x(3) = 1.0000000000

Proba solutiei A*X-B=0:

Ecuatia 1: 0

Ecuatia 2: 0

Ecuatia 3: 0

Se observă că soluția este identică cu cea calculată pas cu pas. Mai mult, se afișează și matricea superior triunghiulară, putându-se astfel evalua determinantul matricei coeficienților. Nu în ultimul rând, programul face și proba soluției, observându-se astfel că soluția este corectă, deoarece există identitate cu 0 în fiecare ecuație după introducerea valorilor calculate.

Metoda eliminării Gauss poate fi folosită și pentru evaluarea inversei unei matrice. Astfel, în continuare se prezintă modul de evaluare a inversei matricei coeficienților utilizând aplicația Matlab. Pornind de la identitatea:

$$\boldsymbol{A}\cdot\boldsymbol{A^{-1}}=\boldsymbol{I}\ \equiv \begin{bmatrix}1 & 1 & 2\\ 3 & 2 & 1\\ 1 & -2 & 3\end{bmatrix}\cdot\begin{bmatrix}x_{11} & x_{12} & x_{13}\\ x_{21} & x_{22} & x_{23}\\ x_{31} & x_{32} & x_{33}\end{bmatrix}=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} \tag{3.26}$$

se evaluează elementele fiecărei coloane a matricei $\boldsymbol{A^{-1}}$ prin aplicarea repetată a metodei Gauss considerând ca termeni liberi, succesiv, câte o coloana din matricea ***I.*** Astfel, pentru evaluarea primei coloane a matricei inverse, se rezolvă următorul sistem de ecuații:

$$\begin{bmatrix}1 & 1 & 2\\ 3 & 2 & 1\\ 1 & -2 & 3\end{bmatrix}\cdot\begin{bmatrix}x_{11}\\ x_{21}\\ x_{31}\end{bmatrix}=\begin{bmatrix}1\\ 0\\ 0\end{bmatrix} \tag{3.27}$$

Aplicând secvența Matlab, se obțin următoarele valori:

x(1) = -0.5000000000

x(2) = 0.5000000000

x(3) = 0.5000000000

Continuând în aceeaşi manieră, se evaluează și celelalte elemente. Alăturând rezultatele se obține următoarea matrice:

$$\boldsymbol{A^{-1}}=\begin{bmatrix}-0.5 & 0.4375 & 0.1875\\ 0.5 & -0.0625 & -0.3125\\ 0.5 & -0.1875 & 0.0625\end{bmatrix} \tag{3.28}$$

3.2.4 Aplicații propuse

P1. Utilizând algoritmul Matlab, să se rezolve aplicația 2 descrisă în secțiunea 3.1. Să se evalueze și numărul aproximativ de secvențe de calcul necesare rezolvării sistemului, folosind relația (3.17).

P2. Să se determine soluția următorului sistem de ecuații liniare:

$$\begin{cases} 2x_2 + 3x_3 = 8 \\ 4x_1 + 6x_2 + 7x_3 = -3 \\ 2x_1 + x_2 + 6x_3 = 5 \end{cases} \tag{3.29}$$

P3. Să se determine reacțiunile structurii din Figura 22.

P4. Să se calculeze determinantul și inversa următoarei matrice:

$$\boldsymbol{A} = \begin{bmatrix} -2 & 3 & 2 & 8 \\ 1 & 3 & -2 & 6 \\ 5 & -1 & 3 & 9 \\ 2 & 3 & 8 & -1 \end{bmatrix} \tag{3.30}$$

3.3 Factorizarea Cholesky

3.3.1 Noțiuni teoretice

O variantă mai eficientă de rezolvare a sistemelor de ecuații liniare poate fi evidențiată în cazul sistemelor având matricea coeficienților $\boldsymbol{A}$:

- simetrică:

$$a_{ij} = a_{ji}, (\forall)\ i, j = \overline{1, n} \tag{3.31}$$

- pozitiv definită:

$$\boldsymbol{X}^T \cdot \boldsymbol{A} \cdot \boldsymbol{X} > 0, (\forall)\ \boldsymbol{X} \neq \boldsymbol{0} \tag{3.32}$$

Observație: această proprietate poate fi verificată folosind teorema (criteriul) Sylvester:

O matrice este pozitiv definită dacă toți minorii principali primari sunt pozitivi.

Acest criteriu va fi folosit în cadrul secțiunii 3.3.3 pentru a arăta că matricea coeficienților din exemplul de test este pozitiv definită.

Dacă condițiile (3.31) și (3.32) sunt îndeplinite atunci matricea coeficienților poate fi astfel descompusă:

$$\boldsymbol{A} = \boldsymbol{L} \cdot \boldsymbol{L}^T \tag{3.33}$$

unde $\boldsymbol{L}$ este o matrice inferior triunghiulară (are elemente nule deasupra diagonalei principale). În aceste condiții, aplicarea metodei Cholesky presupune parcurgerea următoarelor 2 etape:

Etapa 1 – de factorizare (descompunere)

În această etapă se determină elementele matricei $\boldsymbol{L}$. Pornind de la relația (3.33), scrisă în forma extinsă (3.34), elementele l_{ij} se determină prin identificarea termen cu termen a produsului matriceal (3.34). Relațiile de recurență pentru evaluarea coeficienților l_{ij} pot fi consultate, de exemplu, în [6, 7, 12]. În cadrul aplicației rezolvate, se va arata pas cu pas modul de evaluarea e elementelor unei matrice $\boldsymbol{L}$*(3x3).*

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & & 0 \\ \vdots & & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \cdot \begin{bmatrix} l_{11} & l_{21} & \cdots & l_{n1} \\ 0 & l_{22} & & l_{n2} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & l_{nn} \end{bmatrix} \tag{3.34}$$

La finalul etapei 1, sistemul de ecuații devine:

$$\begin{bmatrix} l_{11} & 0 & \cdots & 0 & 0 \\ l_{21} & l_{22} & & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{n-1,1} & l_{n-1,2} & \cdots & l_{n-1,n-1} & 0 \\ l_{n1} & l_{n2} & \cdots & l_{n,n-1} & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & \cdots & l_{n-1,1} & l_{n1} \\ 0 & l_{22} & & l_{n-1,2} & l_{n2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & l_{n-1,n-1} & l_{n,n-1} \\ 0 & 0 & \cdots & 0 & l_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix} \tag{3.35}$$

sau, în formă matriceală condensată:

$$\boldsymbol{L \cdot L^T \cdot X = B} \tag{3.36}$$

Etapa 2 – de rezolvare

Determinarea soluției se face în doi pași. Astfel:

Pas 1:

Cu notația:

$$\boldsymbol{L^T \cdot X = Y} \tag{3.37}$$

Sistemul (3.36) devine:

$$\boldsymbol{L \cdot Y = B} \tag{3.38}$$

sau, în formă extinsă:

$$\begin{bmatrix} l_{11} & 0 & \cdots & 0 & 0 \\ l_{21} & l_{22} & & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{n-1,1} & l_{n-1,2} & \cdots & l_{n-1,n-1} & 0 \\ l_{n1} & l_{n2} & \cdots & l_{n,n-1} & l_{nn} \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix} \tag{3.39}$$

Având în vedere forma particulară a matricei ***L***, elementele vectorului ***Y*** se determină prin substituție înainte, deci începând cu prima ecuație a sistemului.

Pas 2:

Cunoscând elementele vectorului ***Y,*** se revine la notația (3.37), care în formă extinsă are următoarea formă:

$$\begin{bmatrix} l_{11} & l_{21} & \cdots & l_{n-1,1} & l_{n1} \\ 0 & l_{22} & & l_{n-1,2} & l_{n2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & l_{n-1,n-1} & l_{n,n-1} \\ 0 & 0 & \cdots & 0 & l_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} \tag{3.40}$$

Necunoscutele sistemului se determină prin substituție înapoi.

Observații:

- determinantul matricei $\boldsymbol{A}$ este egal cu pătratul determinantului matricei $\boldsymbol{L}$, iar acesta din urmă poate fi evaluat cu următoarea relație:

$$\det(\boldsymbol{L}) = l_{11} \cdot l_{22} \cdot \ldots \cdot l_{nn} \tag{3.41}$$

- în etapa de factorizare, termenii liberi nu sunt afectați. Astfel, odată realizată factorizarea (3.33), ea poate fi folosită pentru mai multe seturi de termeni liberi.
- metoda poate fi utilizată pentru calcularea inversei unei matrice simetrice și pozitiv definite, printr-un proces similar celui detaliat în secțiunea 3.2.3.
- pentru sisteme având număr mare de ecuații, numărul secvențelor de calcul necesare factorizării Cholesky este [6]:

$$nr.\ operații_{Cholesky} \cong \frac{n^3}{6} \tag{3.42}$$

Comparativ cu metoda Gauss, numărul operațiilor este redus la aproximativ jumătate.

- un alt avantaj, este reprezentat de posibilitatea stocării parțiale a matricei coeficienților, datorită simetriei acesteia.

3.3.2 Aplicație Matlab

```
%Date  de intrare
%Matricea coeficientilor A. Trebuie sa fie simetrica si pozitiv definita
A= [ 4  -1   1
   -1  2.5  2
    1   2  3.5];
%Vectorul termenilor liberi B
B=[4.5 6.75 11];
B = B';

%% verificare matice pozitiv definita si simetrica
[n, m] = size(A);
if m ~= size(B,1)
   error (' Matriea A si B au numar diferit de linii')
end
if n~=m
   error ( ' Matricea A nu este patratica')
end
A1=A;
```

```
if isequal(A, A')
   pozitiv = 1;
else
   error (' Matricea A nu e simetrica ')
end
for i1 = 1 :n
   if det(A1)<= 0
      error (' Matricea nu e pozitiv definita');
   end
   A1 (end, :) = []; % se elimina ultima linie din A
   A1 (:, end) = []; % se elimina ultima coloana din A
end

%Etapa 1 factorizare: transformare A=L*LT
L = zeros (n,n);
   for i1 = 1:n
      % Calcul elemente de pe diagonala L(i, i)
      suma_diag = 0;
      for k1 = 1:i1-1
         suma_diag = suma_diag + L(i1, k1)^2;
      end
      L(i1, i1) = sqrt(A(i1, i1) - suma_diag);

      for j1 = i1+1:n
         sum_off_diag = 0;
         for k1 = 1:i1-1
            sum_off_diag = sum_off_diag + L(j1, k1) * L(i1, k1);
         end
         L(j1, i1) = (A(j1, i1) - sum_off_diag) / L(i1, i1);
      end
   end

%Etapa 2 - determinare solutii
%Pas 1: substitutie inainte; practic se rezolva sistemul L*y=B
%unde cu y s-a notat LT*X
y = zeros (n,1);
y(1)=B(1)/L(1,1);
for i1= 2:n
   S=B(i1);
   for j1=1:i1-1
      S=S-L(i1,j1)*y(j1);
   end
```

```
    y(i1)=S/L(i1,i1);
end

%Pas 2:substitutie inapoi; practic se rezolva sistemul LT*X=Y
LT=L';
x = zeros(n,1);
x(n)=y(n)/LT(n,n);
for i1=n-1:-1:1
    S=y(i1);
    for j1=i1+1:n
        S=S-LT(i1,j1)*x(j1);
    end
    x(i1)=S/LT(i1,i1);
end

%% afisare rezultate
disp(" Matricea L")
disp(L)
%afisare solutie
disp('Solutia sistemului de ecuatii este:')
for k1=1:n
    fprintf ( '   x(%1.0f) = %2.10f   \n',k1,x(k1))
end
%proba solutiei; practic se face operatia A*X-B
disp ('   ');
disp('Proba solutiei A*X-B=0:')
%fprintf ( ' %2.20f   \n', A*x'-B');
for k1=1:n
    fprintf ( '  Ecuatia %2.0f: %g   \n', k1, A(k1,:)*x-B(k1));
end
```

3.3.3 Aplicație rezolvată

Pentru exemplificarea operațiilor ce trebuie parcurse pentru rezolvarea unui sistem de ecuații liniare cu metoda Cholesky, se consideră următorul exemplu:

$$\begin{cases} 4x_1 - x_2 + x_3 = 4.5 \\ -x_1 + 2.5x_2 + 2x_3 = 6.75 \\ x_1 + 2x_2 + 3.5x_3 = 11 \end{cases} \tag{3.43}$$

Matricea coeficienților este:

$$\boldsymbol{A} = \begin{bmatrix} 4 & -1 & 1 \\ -1 & 2.5 & 2 \\ 1 & 2 & 3.5 \end{bmatrix} \tag{3.44}$$

Se observă că matricea $\boldsymbol{A}$ este simetrică, iar în continuare se verifică dacă e și pozitiv definită, folosind criteriul Sylvester. Astfel, minorul principal de ordin 1 este reprezentat de elementul $D_1 = a_{11} = 4$, care e pozitiv. Minorul de ordin 2 este determinantul matricei care cuprinde elementele comune ale primelor 2 linii, respectiv primelor două coloane:

$$D_2 = \begin{vmatrix} 4 & -1 \\ -1 & 2.5 \end{vmatrix} = 9 > 0 \tag{3.45}$$

Minorul principal de ordin 3 este chiar determinantul matricei $\boldsymbol{A}$:

$$D_3 = \det(\boldsymbol{A}) = 9 > 0 \tag{3.46}$$

Toți minorii principali fiind pozitivi rezultă că matricea coeficienților este pozitiv definită.

Etapa 1:

$$\begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \cdot \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix} = \begin{bmatrix} 4 & -1 & 1 \\ -1 & 2.5 & 2 \\ 1 & 2 & 3.5 \end{bmatrix} \tag{3.47}$$

Se efectuează secvențial produsul matriceal:

Linia din $\boldsymbol{L}$	·	Coloana din $\boldsymbol{L^T}$	=	Element din $\boldsymbol{A}$	Ecuație	Coeficient calculat
1		1		a_{11}	$l_{11}^2 = 4$	$l_{11} = 2$
1		2		a_{12}	$l_{11} \cdot l_{21} = -1$	$l_{21} = l_{12} = -0.5$
1		3		a_{13}	$l_{11} \cdot l_{31} = 1$	$l_{31} = l_{13} = 0.5$
2		2		a_{22}	$l_{21}^2 + l_{22}^2 = 2.5$	$l_{22} = 1.5$
2		3		a_{23}	$l_{21} \cdot l_{31} + l_{22} \cdot l_{32} = 2$	$l_{32} = l_{23} = 1.5$
3		3		a_{33}	$l_{31}^2 + l_{32}^2 + l_{33}^2 = 3.5$	$l_{33} = 1$

Etapa 2 - Pas 1

Se rezolvă sistemul (3.38):

$$\begin{bmatrix} 2 & 0 & 0 \\ -0.5 & 1.5 & 0 \\ 0.5 & 1.5 & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 4.5 \\ 6.75 \\ 11 \end{bmatrix} \tag{3.48}$$

Aplicând substituția înainte, rezultă:

$$\begin{aligned} &\text{Ecuația } 1 \rightarrow y_1 = 2.25 \\ &\text{Ecuația } 2 \rightarrow y_2 = 5.25 \\ &\text{Ecuația } 3 \rightarrow y_3 = 2 \end{aligned} \tag{3.49}$$

Pas 2

Se rezolvă sistemul (3.37):

$$\begin{bmatrix} 2 & -0.5 & 0.5 \\ 0 & 1.5 & 1.5 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2.25 \\ 5.25 \\ 3 \end{bmatrix} \tag{3.50}$$

Aplicând substituția înapoi, rezultă:

$$\begin{aligned} &\text{Ecuația } 3 \rightarrow x_3 = 2 \\ &\text{Ecuația } 2 \rightarrow x_2 = 1.5 \\ &\text{Ecuația } 1 \rightarrow x_1 = 1 \end{aligned} \tag{3.51}$$

Aplicând secvența Matlab, se afișează următoarele:

Matricea L

2.0000 0 0

-0.5000 1.5000 0

0.5000 1.5000 1.0000

Solutia sistemului de ecuatii este:

x(1) = 1.0000000000

x(2) = 1.5000000000

x(3) = 2.0000000000

Proba solutiei A*X-B=0:

Ecuatia 1: 0

Ecuatia 2: 0

Ecuatia 3: 0

3.3.4 Aplicații propuse

P1. Utilizând algoritmul Matlab, să se rezolve aplicația 2 descrisă în secțiunea 3.1. Să se evalueze și numărul aproximativ de secvențe de calcul necesare rezolvării sistemului, folosind relația (3.42) și să se compare cu numărul de secvențe necesar în cadrul metodei Gauss.

P2. Să se rezolve următorul sistem de ecuații liniare:

$$\begin{cases} 5x_1 - x_2 + 2x_3 + x_4 - x_5 = 12 \\ -x_1 + 5x_2 + x_3 - x_4 - 2x_5 = 13 \\ 2x_1 + x_2 + 6x_3 + 2x_4 - x_5 = 15 \\ x_1 - x_2 + 2x_3 + 6x_4 + 2x_5 = -12 \\ -x_1 - 2x_2 - x_3 + 2x_4 + 7x_5 = 14 \end{cases} \tag{3.52}$$

P3. Să se calculeze determinantul și inversa următoarei matrice:

$$\boldsymbol{A} = \begin{bmatrix} 4 & 1 & 2 & 3 \\ 1 & 9 & -2 & 1.5 \\ 2 & -2 & 16 & -1 \\ 3 & 1.5 & -1 & 8 \end{bmatrix} \tag{3.53}$$

3.4 Metoda Jacobi

3.4.1 Noțiuni teoretice

O alternativă a metodelor directe de rezolvare a sistemelor de ecuații liniare este reprezentată de metodele iterative, în cadrul cărora soluția $\boldsymbol{X} = (x_1\ x_2\ ...\ x_n)^T$ se determină prin aproximații succesive generate de o relație de iterare, pornind de la o aproximație inițială cunoscută $\boldsymbol{X^{(0)}} = \left(x_1^{(0)}\ x_2^{(0)}\ ...\ x_n^{(0)}\right)^T$. În cadrul metodei Jacobi, în vederea obținerii relației de iterare se explicitează din fiecare ecuație $i = \overline{1,n}$ a sistemului (3.1) necunoscuta x_i:

$$\begin{cases} x_1 = \dfrac{b_1}{a_{11}} - \dfrac{a_{12}}{a_{11}}x_2 - \cdots - \dfrac{a_{1n}}{a_{11}}x_n \\ x_2 = \dfrac{b_2}{a_{22}} - \dfrac{a_{21}}{a_{22}}x_1 - \cdots - \dfrac{a_{2n}}{a_{22}}x_n \\ \vdots \\ x_i = \dfrac{b_i}{a_{ii}} - \dfrac{a_{i1}}{a_{ii}}x_1 - \cdots - \dfrac{a_{in}}{a_{ii}}x_n \\ \vdots \\ x_n = \dfrac{b_n}{a_{nn}} - \dfrac{a_{n1}}{a_{n2}}x_1 - \cdots - \dfrac{a_{n,n-1}}{a_{nn}}x_{n-1} \end{cases} \tag{3.54}$$

Se fac următoarele notații:

$$\begin{gathered} g_i = \frac{b_i}{a_{ii}}, i = \overline{1,n} \\ m_{ij} = -\frac{a_{ij}}{a_{ii}}, i,j = \overline{1,n}, i \neq j \end{gathered} \tag{3.55}$$

Trecând în formă matriceală sistemul (3.54) și folosind notațiile (3.55), se obține:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_i \\ \vdots \\ g_n \end{bmatrix} \begin{bmatrix} 0 & m_{12} & \cdots & m_{1i} & \cdots & m_{1n} \\ m_{21} & 0 & \cdots & m_{2i} & \cdots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ m_{i1} & m_{i2} & \cdots & 0 & \cdots & m_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{ni} & \cdots & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix} \tag{3.56}$$

sau, în formă condensată:

$$\boldsymbol{X} = \boldsymbol{g} + \boldsymbol{M} \cdot \boldsymbol{X} \tag{3.57}$$

Considerând cunoscută aproximația inițială $\boldsymbol{X^{(0)}}$, suntem astfel conduși spre următoarea relație de iterare:

$$\boldsymbol{X}^{(\boldsymbol{k+1})} = \boldsymbol{g} + \boldsymbol{M} \cdot \boldsymbol{X}^{(\boldsymbol{k})}, k \geq 0 \tag{3.58}$$

Fiecare necunoscută a vectorului $\boldsymbol{X}$ se aproximează în iterația (k+1) folosind elementele vectorului necunoscutelor evaluate în iterația precedentă:

$$x_i^{(k+1)} = g_i + \sum_{j=1}^{n} m_{ij} \cdot x_j^{(k)} \tag{3.59}$$

Ca și criteriu de terminare a procesului iterativ se poate folosi relația de mai jos. Norma euclidiană a unui vector se poate evalua cu relația (2.18).

$$\left\| \boldsymbol{X}^{(\boldsymbol{k+1})} - \boldsymbol{X}^{(\boldsymbol{k})} \right\| \leq \varepsilon \tag{3.60}$$

Observații:

- metoda converge spre soluția exactă, independent de aproximația inițială folosită, dacă matricea coeficienților este diagonal dominantă. Astfel, pe fiecare linie, modulul elementului diagonal trebuie să fie mai mare decât suma modulelor celorlalte elemente de pe linie:

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}, i = \overline{1,n} \tag{3.61}$$

- în mod obișnuit, aproximația inițială se consideră a fi vectorul $\boldsymbol{g}$ sau un vector nul.
- o variantă accelerată a metodei Jacobi se obține prin utilizarea în cadrul unei iterații curente (k+1) a valorilor $x_i^{(k+1)}$ deja calculate pentru determinarea necunoscutelor $x_{i+1}^{(k+1)} \ldots x_n^{(k+1)}$:

$$x_i^{(k+1)} = g_i + \sum_{j=1}^{i-1} m_{ij} \cdot x_j^{(k+1)} + \sum_{j=i+1}^{n} m_{ij} \cdot x_j^{(k)} \tag{3.62}$$

Această varianta modifică este denumită metoda Gauss-Seidel. Folosirea acesteia conduce la determinarea soluției după un număr mai mic de iterații decât cel necesar metodei Jacobi.

3.4.2 Aplicație Matlab

```
clearvars
clc
```

```
%% Date de intrare
% Matricea coeficientilor A
A= [ 10    1    1
       1   20    2
    -1.5  -1.5  15];
% Vectorul termenilor liberi
B = [25.5 58 38.25];
B = B';
n=size(B,1);
% Toleranta
tol=1e-3;
% nr max de iteratii
max_iter = 1e3;
%Aproximatia initiala
%Varianta 1: zerouri
%x=zeros(n,1);
%Varianta 2: vectorul g (b(i)/a(i,i))
for i=1:n
   g(i)=B(i)/A(i,i);
end
x=g';

%% Programul principal
for i=1:n
   suma_rand=0;
   for j=1:n
      suma_rand=suma_rand+abs(A(i,j));
   end
   if suma_rand-abs(A(i,i)) > abs(A(i,i))  % verificam daca matricea este diagonal domi-
nanta
      disp('OBSERVATIE: Matricea nu este diagonal dominanta');
      disp(' ')
   end
end

for iteratie = 1 : max_iter
   x_anterior=x;
   for i2=1:n
      suma=0;
      for j2=1:n
         if j2~=i2
            suma=suma+A(i2,j2)*x_anterior(j2);
```

```
        end
      end
      x(i2)=(1/A(i2,i2))*(B(i2)-suma);
    end
    norma=norm(x_anterior-x);
    if norma <= tol
      break % s-a atins solutin in precizia de calcul impusa
    end
end

%% Afisare rezultate
if iteratie == max_iter
    disp ('   ');
    disp('Solutia nu s-a putut calcula');
    fprintf ( 'Numarul curent de iteratii este %1.0f  \n',iteratie);
    return
end
disp('Solutia sistemului de ecuatii este:')
for i1=1:n
    fprintf ( '   x(%1.0f) = %2.10f  \n',i1,x(i1))
end
disp ('   ');
fprintf ( 'Solutia s-a determinat dupa %1.0f iteratii  \n',iteratie);
%proba solutiei; practic se face operatia A*X-B
disp ('   ');
disp('Proba solutiei A*X-B=0:')
for i1=1:n
    fprintf ( '  Ecuatia %1.0f: %g  \n', i1, A(i1,:)*x-B(i1));
end
```

3.4.3 Aplicație rezolvată

Pentru exemplificarea operațiilor ce trebuie parcurse pentru rezolvarea unui sistem de ecuații liniare cu metoda Jacobi, se consideră următorul exemplu:

$$\begin{cases} 10x_1 + x_2 + x_3 = 25.5 \\ x_1 + 20x_2 + 2x_3 = 58 \\ -1.5x_1 - 1.5x_2 + 15x_3 = 38.25 \end{cases} \tag{3.63}$$

Matricea coeficienților este:

$$\boldsymbol{A} = \begin{bmatrix} 10 & 1 & 1 \\ 1 & 20 & 2 \\ -1.5 & -1.5 & 15 \end{bmatrix} \tag{3.64}$$

Analizând elementele matricei coeficienților, se observă cu ușurință că matricea $\boldsymbol{A}$ este diagonal dominantă. Vectorul $\boldsymbol{g}$ și matricea $\boldsymbol{M}$ cuprind următoarele elemente:

$$\boldsymbol{g} = \begin{bmatrix} 2.55 \\ 2.9 \\ 2.55 \end{bmatrix}; \boldsymbol{M} = \begin{bmatrix} 0 & -0.1 & -0.1 \\ -0.05 & 0 & -0.1 \\ 0.1 & 0.1 & 0 \end{bmatrix} \tag{3.65}$$

Aproximația inițială se consideră vectorul $\boldsymbol{g}$**.** Toleranța de calcul acceptată este $\epsilon = 10^{-3}$.

Iterația 1:

$$\boldsymbol{X^{(1)}} = \boldsymbol{g} + \boldsymbol{M} \cdot \boldsymbol{X^{(0)}} \rightarrow$$

$$\boldsymbol{X^{(1)}} = \boldsymbol{g} + \boldsymbol{M} \cdot \boldsymbol{g} \rightarrow$$

$$\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ x_3^{(1)} \end{bmatrix} = \begin{bmatrix} 2.55 \\ 2.9 \\ 2.55 \end{bmatrix} + \begin{bmatrix} 0 & -0.1 & -0.1 \\ -0.05 & 0 & -0.1 \\ 0.1 & 0.1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 2.55 \\ 2.9 \\ 2.55 \end{bmatrix} \rightarrow \tag{3.66}$$

$$\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ x_3^{(1)} \end{bmatrix} = \begin{bmatrix} 2.005 \\ 2.5175 \\ 3.095 \end{bmatrix}$$

Verificarea testului de convergență:

$$\left\| \boldsymbol{X^{(1)}} - \boldsymbol{X^{(0)}} \right\| \leq \varepsilon \rightarrow$$

$$\left\| \begin{bmatrix} 2.005 \\ 2.5175 \\ 3.095 \end{bmatrix} - \begin{bmatrix} 2.55 \\ 2.9 \\ 2.55 \end{bmatrix} \right\| \leq 0.001 \rightarrow \tag{3.67}$$

$$\left\| \begin{bmatrix} -0.545 \\ -0.3825 \\ 0.545 \end{bmatrix} \right\| \leq 0.001 \rightarrow$$

$$0.86044 > 0.001$$

Se observă că testul de convergență nu este îndeplinit și astfel, se continuă procesul iterativ care este prezentat în totalitate în tabelul de mai jos. Soluția, cu toleranța de calcul acceptată, este obținută după 5 iterații. Valorile calculate sunt foarte apropiate de soluția exactă care e reprezentată de valorile 2, 2.5 și 3.

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\|\boldsymbol{X}^{(\boldsymbol{k+1})} - \boldsymbol{X}^{(\boldsymbol{k})}\|$	Test de convergență îndeplinit?
1	2.005	2.5175	3.095	0.860440	Nu
2	1.98875	2.49025	3.00225	0.098026	Nu
3	2.00075	2.5003375	2.9979	0.016269	Nu
4	2.000176	2.500173	3.000109	0.002288	Nu
5	1.999972	2.499980	3.000035	0.000290	Da

Micșorând toleranța de calcul la valoarea $\epsilon = 10^{-6}$, soluția calculată se apropie și mai mult de soluția exactă, însă numărul de iterații este mai mare.

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\|\boldsymbol{X}^{(\boldsymbol{k+1})} - \boldsymbol{X}^{(\boldsymbol{k})}\|$	Test de convergență îndeplinit?
1	2.005	2.5175	3.095	0.86043957	Nu
2	1.98875	2.49025	3.00225	0.09802646	Nu
3	2.00075	2.5003375	2.9979	0.01626899	Nu
4	2.000176	2.500173	3.000109	0.00228801	Nu
5	1.999972	2.499980	3.000035	0.00029011	Nu
6	1.999998	2.499998	2.999995	0.00005090	Nu
7	2.000001	2.500001	3.000000	0.00000560	Nu
8	1.999999981	2.500000002	3.000000124	0.00000102	Nu
9	1.999999987	2.499999989	2.999999998	0.00000013	Da

Aplicând secvența Matlab ($\epsilon = 10^{-6}$), se afișează următoarele:

Solutia sistemului de ecuatii este:

x(1) = 1.9999999874

x(2) = 2.4999999886

x(3) = 2.9999999982

Solutia s-a determinat dupa 9 iteratii

Proba solutiei A*X-B=0:

Ecuatia 1: -1.38923e-07

Ecuatia 2: -2.44772e-07

Ecuatia 3: 9.42891e-09

Aplicația Matlab afișează și proba soluției, atestând suplimentar că valorile calculate sunt corecte.

Observație: este posibil ca procesul iterativ să conveargă spre soluție chiar dacă matricea coeficienților nu este diagonal dominantă. De exemplu, pentru aplicația anterioară, modificând coeficientul a_{11} în valoarea 1, algoritmul Matlab afișează următoarele rezultate:

OBSERVATIE: Matricea nu este diagonal dominanta

Solutia sistemului de ecuatii este:

x(1) = 19.3971294905

x(2) = 1.4665072175

x(3) = 4.6363636485

Solutia s-a determinat dupa 14 iteratii

Pentru această situație, procesul iterativ determină soluția după 14 iterații, deci cu un efort de calcul mărit.

3.4.4 Aplicații propuse

P1. Utilizând algoritmul Matlab, să se rezolve aplicația 2 descrisă în secțiunea 3.1. Toleranța de calcul acceptată este $\epsilon = 10^{-4}$.

P2. Să se rezolve sistemul de ecuații liniare de mai jos, acceptând toleranța $\epsilon = 10^{-3}$.

$$\begin{cases} 4x_1 + 6x_2 + x_3 = 8 \\ 5x_1 + 2x_2 + 2x_3 = -3 \\ 2x_1 + x_2 + 6x_3 = 5 \end{cases} \tag{3.68}$$

Este procesul iterativ convergent spre soluție?

Ce se întâmplă dacă primele două ecuații sunt inversate?

P3. Să se rezolve următorul sistem de ecuații liniare:

$$\begin{cases} 8x_1 - x_2 + 2x_3 + x_4 - x_5 = 20 \\ -x_1 + 5x_2 + x_3 - x_4 - 2x_5 = 30 \\ 2x_1 + x_2 + 6x_3 + 2x_4 - x_5 = 40 \\ x_1 - x_2 + 2x_3 + 6x_4 + 2x_5 = -10 \\ -x_1 - 2x_2 - x_3 + 2x_4 + 7x_5 = 10 \end{cases} \tag{3.69}$$

3.5 Condiționarea sistemelor de ecuații liniare

3.5.1 Noțiuni teoretice

Stabilitatea soluției unui sistem de ecuații liniare poate fi examinată prin introducerea unor mici perturbări în termenii liberi ai sistemului și compararea soluțiilor astfel obținute cu cele ale sistemului inițial (neperturbat). Dacă mici perturbări ale termenilor liberi generează mici modificări (de același ordin) ale soluțiilor atunci sistemul este bine condiționat. În caz contrar, se spune că sistemul este rău condiționat. Sensibilitatea soluției sistemului la mici perturbări ale termenilor liberi este semnalată de mărimea numărului de condiție al matricei coeficienților, care se poate evalua cu următoarea relație:

$$cond\ (\boldsymbol{A}) = \|\boldsymbol{A}\| \cdot \left\|\boldsymbol{A^{-1}}\right\| \tag{3.70}$$

unde $\|\boldsymbol{A}\|$ reprezintă norma matricei coeficienților. Dacă $cond\ (\boldsymbol{A}) \cong 1$ atunci sistemul de ecuații este bine condiționat, iar perturbări mici ale termenilor liberi vor genera modificări de același ordin în rândul soluțiilor. Această concluzie este evidențiată prin următoarea relație [6..9]:

$$\frac{1}{cond\ (\boldsymbol{A})} \cdot \frac{\|\boldsymbol{r}\|}{\|\boldsymbol{B}\|} \leq \frac{\|\boldsymbol{e}\|}{\|\boldsymbol{X}\|} \leq cond\ (\boldsymbol{A}) \frac{\|\boldsymbol{r}\|}{\|\boldsymbol{B}\|} \tag{3.71}$$

unde, $\|\boldsymbol{r}\|$ este norma vectorului perturbator, iar $\|\boldsymbol{e}\|$ reprezintă norma vectorului care conține modificarea soluției sistemului. Analizând relația (3.71), se poate deduce că pentru $cond\ (\boldsymbol{A}) \cong 1$ modificarea soluției, indicată de raportul $\frac{\|\boldsymbol{e}\|}{\|\boldsymbol{X}\|}$, este de același ordin cu perturbarea introdusă (cuantificată prin raportul $\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{B}\|}$). Dacă numărul de condiție este mare, atunci există posibilitatea ca mici perturbări ale termenilor liberi să conducă la modificări mari ăn rândul soluțiilor sistemului.

Observații:

- valoarea numărului de condiție este influențată de norma utilizată în evaluarea lui;
- cele mai des întâlnite norme ale unei matrice sunt:
  - norma liniilor (valoarea maximă dintre suma modulelor elementelor de pe fiecare linie):

$$\|\boldsymbol{A}\|_{\infty} = \max_{1 \le i \le n} \sum_{j=1}^{n} |a_{ij}| \tag{3.72}$$

- norma coloanelor (valoarea maximă dintre suma modulelor elementelor de pe fiecare coloană):

$$\|\boldsymbol{A}\|_{\mathbf{1}} = \max_{1 \le j \le n} \sum_{i=1}^{n} |a_{ij}| \tag{3.73}$$

- dacă determinantul matricei coeficienților este apropiat de 0 (matricea este aproape singulară) atunci $cond\ (\boldsymbol{A})$ este mare și, în general, sistemul de ecuații este rău condiționat. Multiplicarea ecuațiilor sistemului cu factori nenuli conduce la modificarea determinantului matricei coeficienților (nu și a soluției sistemului) și, astfel, utilizarea mărimii determinantului în evaluarea stabilității soluției sistemului nu este un criteriu riguros.

3.5.2 Aplicație Matlab

```
clearvars
clc
%Date de intrare
%Matricea coeficientilor A
A= [ 1.002    1
        1     0.998];
%Vectorul termenilor liberi B
B=[2.002   1.998];
%Vectorul perturbare R
R=[0.0001    0];
n=size(A,1);
B = B';
R = R';
B_pert=B+R;
perturbare=norm(B_pert)/norm(B);

% Determinarea numarului de conditie
numar_conditie=cond(A,"inf");  %   calculul numarului de conditie folosind norma liniilor
                               %   ( maximul sumei modulellor elementelor de pe linie)
fprintf ('Numarul de conditie a matricei A este: %6.2f \n', numar_conditie)
disp(" ")
```

```
%Determinarea cu metoda Gauss a solutiilor sistemului neperturbat
x = functie_Gauss(A,B); % se apeleaza metoda eliminarii Gauss
%afisare rezultate sistem perturbat
disp('Solutia sistemului neperturbat este:')
for k=1:n
   fprintf ( '   x(%1.0f) = %2.10f   \n',k,x(k))
end
disp ( " ")
%Determinarea cu metoda Gauss a solutiilor sistemului perturbat
B_perturbat=B+R; % termenul liber perturbat
x_perturbat = functie_Gauss(A,B_perturbat); % se apeleaza metoda eliminarii Gauss
%afisare rezultate sistem perturbat
disp('Solutia sistemului perturbat este:')
for k=1:n
   fprintf ( '   x_perturb(%1.0f) = %2.10f   \n',k,x_perturbat(k))
end
%afisare rezultate
fprintf('\n')
disp('Perturbarea privita in raport cu efectul pe care il provoaca asupra rezultatelor:')
D=x-x_perturbat;
fprintf ('Perturbarea in B = %2.6f \n', perturbare )
efect=norm(D)/norm(x);
fprintf ('Perturbarea in X = %2.6f \n', efect )
```

3.5.3 Aplicație rezolvată

Fie următorul sistem de ecuații liniare:

$$\begin{cases} 1.002x_1 + x_2 = 2.002 \\ x_1 + 0.998x_2 = 1.998 \end{cases} \tag{3.74}$$

Se cere:

- evaluarea numărului de condiție al matricei coeficienților;
- verificarea stabilității soluției sistemului folosind următorul vector perturbator:

$$\boldsymbol{r} = \begin{bmatrix} 0.0001 \\ 0 \end{bmatrix} \tag{3.75}$$

Pentru evaluarea numărului de condiție trebuie apriori evaluată inversa matricei $\boldsymbol{A}$. Fiind o matrice pătratică de ordin 2, inversa se va evalua clasic, pe baza adjunctei:

$$\boldsymbol{A}^{-1} = \frac{1}{\det(\boldsymbol{A})} \cdot \boldsymbol{A}^{*} \rightarrow \tag{3.76}$$

$$\boldsymbol{A}^{-1} = \frac{1}{-0.000004}\begin{bmatrix} 0.998 & -1 \\ -1 & 1.002 \end{bmatrix} \rightarrow$$

$$\boldsymbol{A}^{-1} = \begin{bmatrix} -249500 & 250000 \\ 250000 & -250500 \end{bmatrix}$$

Având în vedere că matricea $\boldsymbol{A}$ este simetrică, norma ei și a inversei nu depinde de norma aplicată. Astfel:

$$\begin{aligned} \|\boldsymbol{A}\|_\infty &= \max(2.002, 1.998) = 2.002 \\ \left\|\boldsymbol{A^{-1}}\right\|_\infty &= \max(499500, 500500) = 500500 \end{aligned} \tag{3.77}$$

În baza relației (3.70), numărul de condiție este:

$$cond\ (\boldsymbol{A}) = 2.002 \cdot 500500 = 1002001 \tag{3.78}$$

Se observă că numărul de condiție este foarte mare și astfel soluția sistemului poate fi sensibilă la mici perturbări ale termenilor liberi. Soluția sistemului inițial (3.74) este:

$$\boldsymbol{X} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{3.79}$$

Introducând perturbarea impusă, sistemul de ecuații devine:

$$\begin{cases} 1.002x_1 + x_2 = 2.0021 \\ x_1 + 0.998x_2 = 1.998 \end{cases} \tag{3.80}$$

iar soluția acestuia:

$$\boldsymbol{X}^* = \begin{bmatrix} -23.95 \\ 26 \end{bmatrix} \tag{3.81}$$

Comparând soluțiile (3.79), respectiv (3.81) se observă o modificare considerabilă a acestora ca urmare a micilor perturbări introduse. În consecință, se poate afirma că sistemul este rău condiționat.

Folosind secvența Matlab, descrisă mai sus, se afișează următoarele:

Numarul de conditie al matricei A este: 1002001.00

Solutia sistemului neperturbat este:

x(1) = 1.0000000001

x(2) = 0.9999999999

Solutia sistemului perturbat este:

x_perturb(1) = -23.9500000003

x_perturb(2) = 26.0000000003

Perturbarea privita in raport cu efectul pe care il provoaca asupra rezultatelor:

Perturbarea in B = 1.000025

Perturbarea in X = 24.975013

În cadrul aplicației Matlab, rezolvarea celor două sisteme de ecuații se face folosind metoda eliminării Gauss, iar norma utilizată pentru evaluarea numărului de condiție este cea a liniilor. Suplimentar rezultatelor calculate manual, secvența Matlab evaluează și rapoartele $\frac{\|\boldsymbol{e}\|}{\|\boldsymbol{X}\|}$, respectiv $\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{B}\|}$. Se observă că ordinul de mărime al efectului $\frac{\|\boldsymbol{e}\|}{\|\boldsymbol{X}\|}$ este diferit de cel al cauzei $\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{B}\|}$, consecință caracteristică sistemelor de ecuații rău condiționate.

3.5.4 Aplicații propuse

P1. Să se multiplice cu factorul 10 prima ecuație a sistemului (3.74), iar apoi să se calculeze numărul de condiție și să se studieze stabilitatea soluției sistemului folosind același vector perturbator. Ce se observă?

P2. Să se calculeze numărul de condiției al matricei de rigiditate a structurii descrise în aplicația 2 a secțiunii 3.1. Să se studieze apoi stabilitatea soluției sistemului de ecuații asociat, considerând că forțele aplicate devin 111 kN. Sistemul este bine sau rău condiționat?

P3. Fie matricea Hilbert de rang 4:

$$\boldsymbol{H_4} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{bmatrix} \tag{3.82}$$

Să se calculeze numărul de condiție și să se studieze stabilitatea sistemului cunoscând:

$$\begin{gathered} \boldsymbol{B} = [25/12 \quad 77/60 \quad 57/60 \quad 319/420] \\ \boldsymbol{r} = [0.1 \quad 0 \quad 0.1 \quad 0] \end{gathered} \tag{3.83}$$

BIBLIOGRAFIE

[1] Microsoft Corporation. (2021). Microsoft Excel. https://office.microsoft.com/excel

[2] The MathWorks Inc. (2022). MATLAB version: 9.13.0 (R2022b), Natick, Massachusetts. https://www.mathworks.com

[3] https://www.geogebra.org/graphing

[4] https://www.desmos.com/calculator

[5] https://www.wolframalpha.com/examples/mathematics/plotting-and-graphics

[6] Chisalita A.: Numerical Analysis, Universitatea Tehnică din Cluj-Napoca, 2002.

[7] Chiorean C.G.: Metode numerice – Note de curs online, https://heyzine.com/flip-book/326d0e8caa.html

[8] Atkinson K., Weimin H.: Elementary numerical analysis - Third edition, John Wiley & Sons, Inc., 2004.

[9] Gerald C., Wheatley P.O.: Applied Numerical Analysis – Seventh edition, Pearson Education, Inc., 2004.

[10] Chapra S.C., Canale R.P.: Numerical Methods for Engineers – Eighth edition, McGraw Hill, 2021.

[11] Burgos R.B., Silva L.E.: Evaluation of the P- Δ (P-Delta) effect in columns and frames using the two-cycle method based on the solution of the beam-column differential equation, MethodsX 11 (2023), 102248.

[12] Atkinson K.: An introduction to.numerical analysis – Second edition, John Wiley & Sons, Inc., 1989.