ȘTEFAN V. ȘTEFĂNESCU

ANALIZA NUMERICĂ

NOTE DE CURS

EDITURA UNIVERSITĂȚII DIN BUCUREȘTI
2000

Ştefan V. Ştefănescu

ANALIZA NUMERICĂ

Note de curs

Editura Universităţii din Bucureşti
2000

Referenţi ştiinţifici: Conf. dr. ing. **Constantin PENE**
Prof. dr. ing. **Aurelian NEGUŢ**

Descrierea CIP a Bibliotecii Naţionale
ŞTEFĂNESCU, ŞTEFAN
Analiză numerică / Ştefan Ştefănescu. - Bucureşti, Editura Universităţii din Bucureşti, 2000
176 p.; 28 cm.
Bibliogr.
ISBN 973-575-435-5

519.6

Tiparul s-a executat sub cda 486/2000
la Tipografia Editurii Universităţii din Bucureşti

CUPRINS

Prefata

Prezenta lucrare reprezinta primul volum destinat ca suport al cursului de Analiza numerica, curs de un semestru pe care autorul l-a tinut în ultimii cinci ani la Facultatea de geologie si geofizica a Universitatii din Bucuresti.

Capitole din acest material pot fi cu succes folosite de studenti si la cursurile de Matematici generale, Topografie, Statistica geologica, Informatica (de exemplu : rezolvarea sistemelor liniare, vectori si valori proprii, aproximarea functiilor, polinoame Cebâsev, siruri definite recurent, calculul aproximativ al integralei unei functii, scheme cu diferente, rezolvarea ecuatiilor diferentiale, proceduri Monte Carlo).

In acest volum se urmareste prezentarea unor tehnici de baza privind analiza numerica. Urmatorul volum este destinat solutionarii efective pe calculator, prin intermediul unui limbaj de programare, a unui mare numar de probleme ce apar frecvent în practica, în special în domeniul geologiei si geofizicii.

Pe lânga o tratare clasica a erorilor de calcul am evidentiat situatii inedite, gasite de autor, în care rezultate teoretice cunoscute nu pot fi implementate identic pe calculator. Astfel datorita memorarii în calculator a valorilor numerice cu un numar finit de cifre semnificative, proprietatile de comutativitate, asociativitate si distributivitate caracteristice operatiilor aritmetice de adunare si înmultire cu numere reale nu se mai pastreaza într-un calcul automat. Neglijarea acestui aspect poate conduce la erori grave de programare, erori greu detectabile deoarece ele apar extrem de rar.

Intr-un context asemanator este studiat comportamentul sirului de reduse ale unei fractii continue. Trunchierea unei fractii continue la o redusa de ordin n nu este însa analoga cu trunchierea unui numar real prin pastrarea a n cifre semnificative ale sale. Remarcam astfel situatia în care sirul reduselor unei fractii continue ar putea sa nu convearga. Acest fapt nu este însa adevarat si pentru sirul de aproximatii succesive ale unui numar real, aproximatii a caror precizie difera prin numarul de cifre semnificative ce sunt utilizate.

Am evitat pe cât posibil prezentarea unor demonstratii matematice greoaie care nu contribuie la o rezolvare practica a problemei studiate. In acest context mentionam abordarea diferentiata a unor probleme de convergenta ce necesita de regula efectuarea unor demonstratii complexe. Datorita facilitatilor de calcul dictate de sistemele actuale de calcul deosebit de rapide, nu este întotdeauna necesara stabilirea conditiilor teoretice care sa garanteze manifestarea unui fenomen de convergenta. Convergenta unui proces poate fi relativ usor verificata pe calculator iar în cazul nerealizarii ei algoritmul de calcul poate propune o multitudine de alte date de intrare posibile cu care sa se încerce realizarea convergentei procesului în cauza.

In lucrare sunt evidentiate si situatii în care acest mod experimental de solutionare a problemei convergentei trebuie tratat cu mare atentie. Mentionam în acest sens dificultati în a demonstra din punct de vedere practic, bazându-ne numai pe rezultatele unui program de calcul, convergenta seriilor numerice infinite (de exemplu, prin intermediul unui program sa confirmam experimental faptul ca seria $S = \sum_{k \geq 1} \frac{1}{k}$ este divergenta, rezultat cunoscut din punct de vedere teoretic).

De multe ori nu este neaparat necesar sa se obtina efectiv un sir convergent catre solutia problemei numerice respective ci sa se gaseasca o aproximatie foarte buna a solutiei. Acest lucru poate fi realizat si prin procedee euristice, bazându-ne pe resursele uriase ale calculatorului : o viteza de calcul foarte mare precum si o capacitate de memorare enorma.

Privind problematica abordata în aceasta lucrare, pe lânga evidentierea unor aspecte teoretice cu justificarile corespunzatoare, mentionam si o tratare din punct de vedere algoritmic mergându-se pâna la implementarea algoritmului în limbajul BASIC precum si interpretarea si comentarea prin tabele comparative a rezultatelor rularii programului respectiv.

In actualul material nu s-a urmarit prezentarea unor algoritmi sofisticati ci s-au sugerat mai multe moduri de rezolvare a problemei care sa conduca efectiv la obtinerea unei solutii. Permanent s-a avut în vedere dezvoltarea unei metodologii de lucru.

Pe parcursul lucrarii sunt date multe exemple si facute diverse comentarii. Privita în ansamblu întreaga problematica are însa un numitor comun, anume modul de abordare constructiv, folosindu-se nemijlocit calculatorul, pentru solutionarea problemelor de analiza numerica. Procedurile prezentate sunt pe scurt comentate din punct de vedere teoretic si experimental. Sunt propusi diversi algoritmi analizându-se atât punctele lor slabe cât si performantele, posibilitatile de a evita erorile de programare, modificari ale variantei initiale care sa permita utilizarea unui set extins de date de intrare. Uneori sunt sugerate posibile generalizari.

Datorita spatiului restrâns alocat acestei lucrari, aspecte importante ale analizei numerica nu si-au gasit locul (ca de exemplu rezolvarea ecuatiilor liniare si neliniare). De altfel si probleme practice deosebit de utile în geologie si geofizica au fost total neglijate în actualul volum, urmând a fi studiate într-o lucrare separata. In acest context mentionam necesitatea studierii unei problematici privind : estimari de arii si volume, prognoza, modele statistice, utilizarea graficii pe calculator, alocare de resurse si operarea cu liste, clasificare, analiza datelor experimentale, etc.

In cadrul seminariilor de analiza numerica tinute de autor, pe lânga programarea efectiva, în limbajul BASIC, a algoritmilor studiati au fost analizate diverse tehnici de programare structurata, impunerea unui stil de programare, realizarea unei interfete prietenoase cu utilizatorul. Prezentul volum nu trateaza asemenea aspecte, ele urmând a fi pe larg discutate într-un alt volum.

Avându-se permanent în vedere dezvoltarea capacitatii de analiza si a abilitatilor de programare pentru studenti, algoritmii de calcul studiati au fost efectiv implementati în limbajul Quick Basic (MS-DOS, Qbasic, versiunea 1,1) si apoi executati pe calculator. Rezultatele rularilor au fost amplu comentate si analizate comparativ în raport cu alte variante posibile.

In întreaga lucrare s-a urmarit cu perseverenta ideea centrala a analizei numerice. In acest sens de cele mai multe ori nu ne intereseaza obtinerea unei solutii "exacte", teoretice (eventual cu o infinitate de zecimale), daca aceasta solutie nu ar putea fi folosita efectiv în practica. In schimb suntem profund interesati în gasirea unei solutii aproximative a carui prag de eroare poate fi oricând adaptat unor noi cerinte.

In încheierea acestei prezentari tin sa multumesc parintilor mei pentru suportul moral pe care în mod permanent mi l-au acordat .

Bucuresti, octombrie 1998 Stefan Stefanescu

ERORI DE CALCUL

1. Erori relative si absolute

Datorita unor cauze multiple, în locul unei valori exacte x este folosita o aproximatie a sa x^*. Se introduce astfel eroarea ε_x ce exprima diferenta dintre cantitatea reala x si aproximatia respectiva x^*, adica $\varepsilon_x = x - x^*$. Datorita necunoasterii de cele mai multe ori a valorii exacte x, în realitate este dificil de a stabili marimea erorii ε_x utilizând formula precedenta.

Vom nota prin Δ_x eroarea absoluta ce este definita de expresia $\Delta_x = |x - x^*|$. Eroarea relativa δ_x va fi data de expresia $\delta_x = \Delta_x / |x|$. Daca eroarea absoluta Δ_x este mica atunci $x \approx x^*$ fapt ce justifica calculul experimental al erorii relative δ_x dupa formula $\delta_x = \Delta_x / |x^*|$.

Suntem interesati de a stabili relatii între erorile absolute si relative atunci când se efectueaza diverse operatii asupra cantitatilor x, y cu care se opereaza.

Propozitia 1. Sunt satisfacute relatiile

$$\Delta_{x+y} \le \Delta_x + \Delta_y \qquad \Delta_{x-y} \le \Delta_x + \Delta_y \qquad \Delta_{x\cdot y} \le |x^*|\Delta_y + |y^*|\Delta_x + \Delta_x\Delta_y$$
$$\Delta_{x/y} \le [\,|x^*|\Delta_y + |y^*|\Delta_x\,] / [\,(|y^*| - \Delta_y)\,|y^*|\,]$$
$$\delta_{xy} \le \delta_x + \delta_y + \delta_x\delta_y \qquad \delta_{x/y} = (\delta_x + \delta_y)/(1 - \delta_y) \qquad (1)$$

Demonstratie. In urma unui calcul elementar deducem :

$$\Delta_{x+y} = |(x+y) - (x^*+y^*)| = |(x-x^*) + (y-y^*)| \le |x-x^*| + |y-y^*| = \Delta_x + \Delta_y$$
$$\Delta_{x-y} = |(x-y) - (x^*-y^*)| = |(x-x^*) - (y-y^*)| \le |x-x^*| + |y-y^*| = \Delta_x + \Delta_y$$
$$\Delta_{xy} = |(xy) - (x^*y^*)| = |x^*(y-y^*) + y^*(x-x^*) + (x-x^*)(y-y^*)| \le$$
$$\le |x^*||y-y^*| + |y^*||x-x^*| + |x-x^*||y-y^*| = |x^*|\Delta_y + |y^*|\Delta_x + \Delta_x\Delta_y$$
$$\Delta_{x/y} = |(x/y) - (x^*/y^*)| = |y^*(x-x^*) - x^*(y-y^*)| / |y\,y^*| \le$$
$$\le (|y^*||x-x^*| + |x^*||y-y^*|) / [\,(|y^*| - \Delta_y)\,|y^*|\,] \le$$
$$\le (|x^*|\Delta_y + |y^*|\Delta_x) / [\,(|y^*| - \Delta_y)\,|y^*|\,]$$
$$\delta_{xy} = \Delta_{xy} / |x^*y^*| \le (|x^*|\Delta_y + |y^*|\Delta_x + \Delta_x\Delta_y) / (|x^*||y^*|) = \delta_x + \delta_y + \delta_x\delta_y$$
$$\delta_{x/y} = \Delta_{x/y} / |x^*/y^*| \le [\,|x^*|\Delta_y + |y^*|\Delta_x\,] / [\,(|y^*| - \Delta_y)\,|x^*|\,] =$$
$$= (\Delta_y / |y^*| + \Delta_x / |x^*|) / (1 - \Delta_y / |y^*|) = (\delta_x + \delta_y)/(1 - \delta_y)$$

Fie $z = f(x,y)$. Cunoscând aproximatiile x^* si y^* ale lui x, respectiv y, sa determinam eroarea absoluta de aproximare a lui z.

Propozitia 2. $\Delta_z \le M_x^{(1)}\Delta_x + M_y^{(1)}\Delta_y \qquad (2)$

unde : $M_x^{(1)} = \sup\{\, |f_x^{(1)}(\alpha,\beta)| \;;\; x^* \le \alpha \le x,\ y^* \le \beta \le y \,\}$

$M_y^{(1)} = \sup\{\, |f_y^{(1)}(\alpha,\beta)| \;;\; x^* \le \alpha \le x,\ y^* \le \beta \le y \,\}$

$f_x^{(1)}(\alpha,\beta)$, $f_y^{(1)}(\alpha,\beta)$ fiind derivatele partiale de ordinul 1 în raport cu x, respectiv y ce sunt calculate în punctul (α, β).

Demonstratie. Presupunem, de exemplu, $x^* \le x$, $y^* \le y$. In evaluarile urmatoare vom aplica formula lui Lagrange :

$$f(x,y) - f(x^*,y^*) = [\,f(x,y) - f(x^*,y)\,] + [\,f(x^*,y) - f(x^*,y^*)\,] =$$
$$= f_x^{(1)}(\alpha,y)(x - x^*) + f_y^{(1)}(x^*,\beta)(y - y^*)$$

unde $\alpha \in (x^*, x)$ si $\beta \in (y^*, y)$. Asadar

$$| f(x,y) - f(x^*,y^*) | \le | f_x^{(1)}(\alpha,y) | | x - x^* | + | f_x^{(1)}(x^*,\beta) | | y - y^* | \le M_x^{(1)} \Delta_x + M_y^{(1)} \Delta_y$$

Observatia 1. In aplicatiile practice eroarea absoluta Δ_z este calculata dupa formula

$$\Delta_z \approx | f_x^{(1)}(x^*, y^*) | \Delta_x + | f_y^{(1)}(x^*, y^*) | \Delta_y \qquad (3)$$

Justificarea acestei formule rezulta prin utilizarea în demonstratia Propozitiei 2 a aproximarilor $f_x^{(1)}(\alpha,y) \approx f_x^{(1)}(x^*,y^*)$, $f_y^{(1)}(x^*,\beta) \approx f_y^{(1)}(x^*,y^*)$ ce sunt adevarate în cazul în care derivatele partiale de ordin 1 ale functiei $f(x,y)$ sunt continue în dreptunghiul $\{ (s,t) \mid x_0 \le s \le x,\ y_0 \le t \le y \}$.

Aplicând Propozitia 2 pentru diferite cazuri particulare regasim unele rezultate anterioare

Corolarul 1. $\Delta_{x+y} \le \Delta_x + \Delta_y$ $\qquad$ $\Delta_{x-y} \le \Delta_x + \Delta_y$

Demonstratie. Fie $z = f(x,y) = x + y$. In acest caz $f_x^{(1)}(\alpha,\beta) = 1$, $f_y^{(1)}(\alpha,\beta) = 1$ pentru orice $\alpha, \beta \in \mathbf{R}$ si deci $M_x^{(1)} = M_y^{(1)} = 1$. Aplicând Propozitia 2 rezulta

$$\Delta_z = \Delta_{x+y} \le M_x^{(1)} \Delta_x + M_y^{(1)} \Delta_y = \Delta_x + \Delta_y$$

Considerând $z = f(x,y) = x - y$, în urma unui calcul similar se obtine $\Delta_{x-y} \le \Delta_x + \Delta_y$.

Corolarul 2. Daca $z = g(x)$ atunci $\Delta_z = \Delta_{g(x)} \le M_1 \Delta_x$ unde M_1 este maximul modulului derivatei functiei $g(x)$ pe intervalul (x^*, x), $M_1 = \sup \{ | g^{(1)}(\alpha) |,\ x^* \le \alpha \le x \}$

Demonstratie. Rezultatul mentionat se obtine daca în Propozitia 2 vom considera $f(x,y) = g(x)$. In adevar, în aceasta situatie avem $f_x^{(1)}(\alpha,\beta) = g^{(1)}(\alpha,\beta)$, $f_y^{(1)}(\alpha,\beta) = 0$ si deci $M_x^{(1)} = M_1$, $M_y^{(1)} = 0$ fapt ce implica $\Delta_{g(x)} = \Delta_z \le M_x^{(1)} \Delta_x + M_y^{(1)} \Delta_y = M_1 \Delta_x$.

Caz particular. Daca $1 \le x \le 2$ atunci $\Delta_{exp(x)} \le e^2 \Delta_x$ deoarece pentru $g(x) = e^x$, $1 \le x \le 2$, rezulta $M_1 = \sup \{ | e^{\alpha} |,\ 1 \le \alpha \le 2 \} = e^2$

2. Erori ale solutiilor sistemelor de ecuatii liniare

Nu întotdeauna se pot gasi procedee eficiente de micsorarea erorilor de calcul. Este însa util de a cunoaste eventualele surse de eroare pentru a se evita crearea "conditiilor favorabile" de aparitie a lor. In continuare este evidentiat un asemenea aspect.

Aplicatia 1. Vom studia variatia solutiilor unui sistem liniar de doua ecuatii cu doua necunoscute atunci când în locul valorilor reale ale coeficientilor sistemului sunt utilizate aproximatii ale lor.

Un calcul elementar ne arata ca solutiile x_1, x_2 ale unui sistem liniar de doua ecuatii cu coeficienti reali,

$$\begin{aligned} a_{11} \cdot x_1 + a_{12} \cdot x_2 &= b_1 \\ a_{21} \cdot x_1 + a_{22} \cdot x_2 &= b_2 \end{aligned} \qquad (4)$$

sunt de forma

$$\begin{aligned} x_1 &= [b_1 \cdot a_{22} - b_2 \cdot a_{12}] / [a_{11} \cdot a_{22} - a_{12} \cdot a_{21}] \\ x_2 &= [b_2 \cdot a_{11} - b_1 \cdot a_{21}] / [a_{11} \cdot a_{22} - a_{12} \cdot a_{21}] \end{aligned} \qquad (5)$$

Obtinerea valorilor efective ale solutiei (5) pentru sistemul (4) este realizata de programul BASIC listat în continuare.

```
REM  Erori la solutionarea unui sistem liniar
INPUT "Linia 1 : "; a11, a12, b1:          INPUT "Linia 2 : "; a21, a22, b2
```

```
x1 = (b1 * a22 - b2 * a12) / (a11 * a22 - a12 * a21)
x2 = (b2 * a11 - b1 * a21) / (a11 * a22 - a12 * a21)
PRINT "Solutie :  x1 = "; x1; "       x2 = "; x2
```

Exemplul 1. Prin calcul direct deducem ca solutia urmatorului sistem de doua ecuatii liniare

$$\begin{aligned} x_1 + 2x_2 &= 5 \\ 3x_1 + 4x_2 &= 11 \end{aligned} \qquad (6)$$

este $x_1 = 1$ si $x_2 = 2$. Rularea programului BASIC prezentat în Aplicatia 1 a condus evident la aceeasi solutie (1 , 2) a sistemului (6).

Exemplul 2. In practica coeficientii sistemului liniar (4) pot fi supusi unor erori de calcul. Vom analiza acest aspect considerînd urmatorul sistem liniar

$$\begin{aligned} x_1 + 2x_2 &= 5 \\ a_{21}x_1 + a_{22}x_2 &= b_2 \end{aligned} \qquad (7)$$

asemanator sistemului (6) prin pastrarea nealterata a primei ecuatii.

Sistemul (6) se obtine din (7) pentru $a_{21} = 3$, $a_{22} = 4$, $b_2 = 11$.

Variind cu circa o zecime coeficientii celei de a doua ecuatii a sistemului liniar (6) se obtin noi sisteme liniare de forma (7) ale caror solutii nu difera "prea mult" de solutia "fara eroare" (1 , 2) a sistemului (6). O sinteza a rezultatelor obtinute este prezentata în Tabelul 1.

Tabelul 1. Variatia solutiei (x_1 , x_2) a sistemului (7), simulând erori la preluarea coeficientilor celei de a doua ecuatii din (6).

a_{21}	a_{22}	b_2	x_1	x_2	a_{21}	a_{22}	b_2	x_1	x_2
3.0	4.0	11.0	1.000000	2.000000	3.0	4.0	11.1	1.100000	1.950000
3.0	4.0	10.9	0.900000	2.050000	3.1	4.0	11.0	0.909091	2.045455
2.9	4.1	11.1	1.000001	2.000000	3.0	4.1	11.1	0.894738	2.052631
3.0	3.9	10.9	1.095238	1.952381	3.1	3.9	11.0	1.086957	1.956522
2.9	4.0	10.9	1.000000	2.000000	3.1	4.0	11.1	1.000000	2.000000

Urmarind rezultatele din Tabelul 1 am fi tentati sa credem ca o eroare de cel mult 0.1 impusa coeficientilor sistemului liniar (6) ar antrena neaparat o eroare în jurul pragului de 0.1 pentru solutia noului sistem rezultat. Un asemenea rationament gresit, nefondat din punct de vedere teoretic, este adesea promovat în practica datorita necunoasterii suficiente a modului de compunere a erorilor.

Exemplul 2. Vom relua studiul precedent luând si alte cazuri în considerare. Rezultatele sunt centralizate în Tabelul 2. Exista astfel variante pentru care o variatie de 0.1 a coeficientilor celei de a doua ecuatie a sistemului (6) antreneaza erori ale componentelor solutiei de peste 0.4 (Tabelul 2).

Asadar trebuie întreprins un studiu teoretic al erorilor rezultate prin rezolvarea sistemelor liniare atunci când coeficientii acestor sisteme sunt supusi unor erori sistematice.

Exemplul 3. Considerând în sistemul (7) $a_{21} = 2.5$, $a_{22} = 4.5$, $b_2 = 11.0$ se obtine solutia $x_1 = -1$, $x_2 = 3$. Fata de sistemul initial (6) valorile coeficientilor a_{21} si a_{22} au fost modificati cu 0 5 Noua solutie (-1 , 3) rezultata în acest mod difera pe componente cu 2 , respectiv -1 unitati în raport cu solutia (1 , 2) a sistemului (6).

Tabelul 2. Componentele x_1 , x_2 ale solutiei sistemului (7) pot fluctua cu mai mult de 0.1 în cazul în care valorile coeficientilor a_{21} , a_{22} , b_2 variaza cu 0.1.

a_{21}	a_{22}	b_2	x_1	x_2	a_{21}	a_{22}	b_2	x_1	x_2
3.0	4.0	11.0	1.000000	2.000000	3.1	4.0	10.9	.8181815	2.090909
3.0	3.9	11.0	1.190476	1.904762	3.0	4.1	11.0	0.789474	2.105263
3.0	3.9	11.1	1.285715	1.857143	2.9	4.1	10.9	0.764706	2.117647
2.9	3.9	11.1	1.421053	1.789474	3.1	4.1	10.9	0.619048	2.190476

Aplicatia 2. Erorile privind solutiile sistemelor liniare sunt cu mult mai mari în situatia unei proaste "conditionari" a sistemului în cauza. Evidentiem faptul ca solutia sistemului liniar este supusa unor erori mari atunci când determinantul matricei asociate acestui sistem este aproximativ nul .

Concret, în cazul sistemului liniar (4) de numai doua ecuatii, determinantul d este dat de expresia $d = a_{11} \cdot a_{22} - a_{12} \cdot a_{21}$. Atunci când valoarea determinantului d este aproape nula rezulta ca dreptele de ecuatii $a_{11} \cdot x_1 + a_{12} \cdot x_2 - b_1 = 0$, respectiv $a_{21} \cdot x_1 + a_{22} \cdot x_2 - b_2 = 0$, sunt aproximativ paralele. Coordonatele punctului de intersectie al acestor doua drepte reprezinta de fapt solutia sistemului liniar respectiv. In situatia în care cele doua drepte sunt "aproape paralele" pozitia punctului de intersectie al acestora (si deci solutia sistemului) se va modifica foarte mult chiar la o usoara schimbare a pozitiei initiale a dreptelor (la o usoara modificare a coeficientilor a_{ij} , $1 \le i, j \le 2$). A se vedea în acest sens si forma (5) a solutiei sistemului (4) unde se împarte cu cantitatea d .

In concluzie, valori aproximativ nule ale determinantului d antreneaza fluctuatii mari ale solutiei sistemului liniar chiar daca intervin alterari relativ mici ale valorilor coeficientilor sistemului.

Exemplul 4. In Tabelul 3 vom evidentia fluctuatiile mari ale solutiei unui sistem "prost conditionat" de forma

$$\begin{aligned} x_1 + 2\,x_2 &= 5 \\ 2\,x_1 + 4.1\,x_2 &= 10.2 \end{aligned} \qquad (8)$$

Sistemele (6) si (8) admit aceeasi solutia (1 , 2) , dar sistemul (8) este mai "prost conditionat" decât sistemul (6). Determinantul sistemului (8) este egal cu 0.1 , o valoare de 20 de ori mai mica decât modulul determinantului sistemului (6). Coeficientii celor doua sisteme au valori cam de acelasi ordin de marime.

Vom prezenta comparativ solutii ale sistemului liniar (7) atunci când cele doua ecuatii ale acestui sistem definesc drepte aproximativ paralele ca în cazul sistemului (8) unde $a_{21} = 2.0$, $a_{22} = 4.1$, $b_2 = 10.2$. Datorita proastei conditionari a sistemului (8), considerând în sistemul (7) $a_{21} \approx 2.0$, $a_{22} \approx 4.1$, $b_2 \approx 10.2$, solutia (1 , 2) a sistemului (8) este substantial alterata la variatii relativ minore ale valorilor coeficientilor a_{21} , a_{22} , b_2 (Tabelul 3).

Observatia 2. "Proasta conditionare" a sistemului (4) dictata de o valoare aproximativ nula a determinantului sau principal d trebuie însa privita dintr-un punct de vedere relativ. In adevar, o împartire a ambelor ecuatii ale sistemului (4) cu o constanta c nu modifica solutia initiala dar, în schimb, antreneaza o micsorare de c^2 ori a determinantului sistemului.

Devine astfel clara necesitatea efectuarii unui studiu sistematic al erorilor ce apar în procesul de solutionare a sistemelor liniare. Este utila introducerea unor indicatori invarianti la transformari

matriciale care nu afecteaza solutia sistemului liniar. In plus acesti indicatori trebuie sa evalueze amplitudinea variatiilor solutiei sistemului ai carui coeficienti sunt supusi la modificari.

Tabelul 3. Fluctuatia solutiei (x_1, x_2) a sistemului (7) în situatia unei proaste conditionari (cazul sistemului (8)).

a_{21}	a_{22}	b_2	x_1	x_2	a_{21}	a_{22}	b_2	x_1	x_2
2.00	4.10	10.20	1.000000	2.000000	2.10	4.00	10.20	2.000000	1.500000
2.00	4.15	10.20	2.333338	1.333331	1.95	4.15	10.15	1.800005	1.599998
2.00	4.10	10.10	2.99999	1.000005	2.00	4.10	10.30	-1.00001	3.000005
1.90	4.00	10.30	-3.000001	4.000000	1.95	4.05	10.25	-1.666659	3.333333
2.03	4.07	10.15	5.000048	-0.000024	2.03	4.07	10.20	-4.999762	4.999881
2.05	4.05	10.20	2.999999	1.000005	2.05	4.15	10.15	8.999990	-1.999995
2.10	4.19	10.20	-55.00143	30.00072	2.10	4.19	10.10	-75.00182	40.00091

Observatia 3. Putem studia fluctuatiile solutiei (x_1, x_2) a sistemului (4) folosind inegalitati de tipul (3). Mentionam mai multe dificultati privind o asemenea abordare :

- Un grad relativ ridicat de complexitate pentru functiilor f_1, f_2 utilizate pentru a calcula componentele x1 , x2 ale solutiei,

$$x_1 = f_1(a_{11}, a_{12}, a_{21}, a_{22}) = [\, b_1 \cdot a_{22} - b_2 \cdot a_{12} \,] / [\, a_{11} \cdot a_{22} - a_{12} \cdot a_{21} \,]$$

$$x_2 = f_2(a_{11}, a_{12}, a_{21}, a_{22}) = [\, b_2 \cdot a_{11} - b_1 \cdot a_{21} \,] / [\, a_{11} \cdot a_{22} - a_{12} \cdot a_{21} \,]$$

- Necesitatea tratarii unitare a perechii de valori (x_1, x_2) si nu o interpretare unilaterala, ce presupune studierea separata a erorile produse asupra fiecarei componente x_1, respectiv x_2 ;
- Extinderea acestui studiu în cazul unui sistem liniar general, de n ecuatii cu n necunoscute.

EROARI CE APAR LA INTERPRETAREA UNEI "BALANTE"

Operarea în calculator cu un numar finit, relativ mic, de cifre semnificative conduce nemijlocit la nesatisfacerea proprietatilor obisnuite de asociativitate si distributivitate ale operatiilor aritmetice de adunare-scadere sau înmultire-împartire. Necunoasterea acestor aspecte favorizeaza aparitia unor erori de programare foarte greu detectabile în aplicatii.

In acest paragraf este sugerat un nou mod de abordare atunci când se intentioneaza implementarea unor proceduri de tip "balanta" aplicatate seturilor multidimensionale de date numerice

1. Prezentarea problemei

La proiectarea de aplicatii pe calculator se impune adesea luarea unor decizii în mod automat în functie de realizarea sau nerealizarea unei anumite "balante" raportata la un set precizat de valori numerice. Asemenea actiuni apar frecvent în viata de zi cu zi. Astfel, în activitatea contabila, verificarea efectiva a corectitunii unor sume numerice se realizeaza adesea prin schimbarea ordinii de însumare a termenilor respectivi.

Un exemplu concludent îl constituie desfasurarea unor operatii aritmetice cu valorile numerice v_{ij}, $1 \le i \le m$, $1 \le j \le n$, ale unui tabel bidimensional V.

In activitatea curenta, în tabele multiplu-dimensionale de tipul V, se efectueaza "centralizari" pe fiecare dimensiune în parte, prin calcularea sumelor l_i, $1 \le i \le m$, "pe linii", respectiv a sumelor c_j, $1 \le j \le n$, "pe coloane", adica :

$$l_i = v_{i1} + v_{i2} + v_{i3} + \ldots + v_{in} \quad , \qquad 1 \le i \le m$$

$$c_j = v_{1j} + v_{2j} + v_{3j} + \ldots + v_{mj} \quad , \qquad 1 \le j \le n$$

De multe ori, în final se evalueaza si "totalul general" t ce presupune însumarea tuturor valorilor v_{ij}, $1 \le i \le m$, $1 \le j \le n$, ale tabelului V. Putem deduce valoarea totala t prin adunarea rezultatelor partiale $l_1, l_2, l_3, \ldots, l_m$ ce au fost obtinute "pe linii", $t = t_{lin} = l_1 + l_2 + l_3 + \ldots + l_m$, sau si prin însumarea celor n rezultate partiale $c_1, c_2, c_3, \ldots, c_n$ ce au fost anterior calculate "pe coloane", $t = t_{col} = c_1 + c_2 + c_3 + \ldots + c_n$.

Verificarea corectitudinii operatiilor aplicate unor date numerice din tabelul bidimensional V se realizeaza indirect prin urmarirea echilibrului balantei dintre totalul t_{lin} obtinut "pe linii" si totalul t_{col} calculat "pe coloane". Mentionam însa si posibilitatea efectuarii de calcule aritmetice gresite cu setul de valori v_{ij}, $1 \le i \le m$, $1 \le j \le n$, calcule ce pot conduce cu totul întâmplator la realizarea unei "balante corecte", prin verificarea egalitatii $t_{lin} = t_{col}$.

In cazul tabelului bidimensional V, realizarea balantei $t_{lin} = t_{col}$ se bazeaza nemijlocit pe utilizarea proprietatilor de comutativitate si asociativitate ale operatiei aritmetice de adunare a valorilor numerice v_{ij}.

De cele mai multe ori efectuarea operatiilor aritmetice complexe se realizeaza în practica cu ajutorul calculatorului. Calculatorul însa opereaza cu un numar finit de cifre semnificative atribuite valorilor numerice respective. Datorita acestei restrictii, în calculele aritmetice pe calculator apar erori de "rotunjire" si de "trunchiere" (a se vedea, de exemplu, Donald E. Knuth : Tratat de programare a calculatoarelor - Algoritmi seminumerici. Editura Tehnica, Bucuresti, 1983). Prezenta acestor erori ar

putea afecta proprietatile de asociativitate si distributivitate ale operatiilor aritmetice de adunare-scadere si înmultire-împartire.

In cele ce urmeaza vom încerca sa stabilim daca pe calculator se mentine proprietatea de asociativitate a operatiilor de adunare si înmultire cu valori numerice si daca avem proprietatea de distributivitate a operatiei de înmultire în raport cu operatia de adunare.

2. O varianta de solutionare

Vom desemna prin $\mathbf{R}_0$ multimea valorilor numerice reale ce pot fi memorate în calculator. In mod evident multimea $\mathbf{R}_0$ este finita si deci diferita de multimea infinita $\mathbf{R}$ a numerelor reale.

Intentionam sa stabilim daca pentru orice elemente a , b , c ale multimii $\mathbf{R}_0$ se mentin urmatoarele egalitati :

$a+(b+c)=(a+b)+c$	P1 : Asociativitatea operatiei de adunare
$a*(b*c)=(a*b)*c$	P2 : Asociativitatea operatiei de înmultire
$a*(b+c)=(a*b)+(a*c)$	P3 : Distributivitatea înmultirii fata de adunare

Verificarea identitatilor (P1)-(P3) prin enumerarea tuturor posibilitatilor (a, b, c) , unde $a \in \mathbf{R}_0$, $b \in \mathbf{R}_0$, $c \in \mathbf{R}_0$, este practic imposibil de realizat datorita unui numar astronomic de cazuri ce vor trebui luate în considerare. Negarea veridicitatii uneia dintre proprietatile (P1), (P2) sau (P3) revine la a gasi un triplet (a_0 , b_0, c_0) , $(a_0 , b_0 , c_0) \in \mathbf{R}_0^3$, astfel încât sa nu fie satisfacuta egalitatea mentionata de proprietatea respectiva. Infirmarea proprietatilor (P1)-(P3) ridica aceleasi dificultati tehnice prin numarul imens de triplete (a , b , c) ce vor trebui parcurse, cu singura deosebire ca algoritmul de verificare se va opri dupa primul caz ("fericit") în care egalitatea în cauza nu mai este satisfacuta.

O enumerare secventiala "regulata" a indicilor a, b, c nu este de dorit a se realiza. Motivam aceasta decizie si prin faptul ca valori mici ale indicilor a, b, c nu conduc la aparitia unor erori de "rotunjire", egalitatile (P1)-(P3) nefiind în acest caz afectate.

Valori "comparabile" ca marime atribuite termenilor a, b, c ar putea micsora eficacitatea operatiunii de depistare a unui triplet (a_0 , b_0 , c_0) care sa nu satisfaca relatia de egalitate studiata.

Din punct de vedere practic, prin efectuarea unor nenumarate teste pe calculator, am constatat ca nerespectarea cerintelor mentionate nu a condus, dupa ore întregi de calcul, la obtinerea unui triplet (a_0 , b_0 , c_0) care sa nu verifice una dintre proprietatile (P1)-(P3). Deoarece nu au fost luate în considerare toate cazurile posibile, negasirea unui astfel de triplet (a_0 , b_0 , c_0) nu confirma si nici nu infirma veridicitatea egalitatilor (P1)-(P3).

Avându-se în vedere aceste considerente am preferat eliminarea unei regularitati în parcurgerea elementelor domeniului $\mathbf{R}_0^3$. Propunem în acest sens o selectare "la întâmplare" a valorilor a, b, c . Astfel elementele a, b, c vor fi generate în mod aleator în multimea $\mathbf{R}_0$, în acest sens putându-se utiliza una dintre procedurile specializate în producerea de numere întâmplatoare (a se vedea, de exemplu, lucrarea lui Ion Sacuiu si Dan Zorilescu : Numere aleatoare - Aplicatii în economie, industrie si studiul fenomenelor naturale. Editura Academiei, Bucuresti, 1978).

Tinând seama de aceste observatii, algoritmul de verificare a egalitatilor (P1)-(P3) poate fi proiectat în orice limbaj de programare.

In vederea solutionarii prezentei probleme am optat pentru scrierea algoritmului de verificare în limbajul BASIC . Aceasta alegere este motivata si de faptul ca BASIC-ul este cel mai vechi limbaj de programare de "nivel înalt", programarea în acest limbaj fiind facila pentru o gama larga de utilizatori (

atât de economisti, matematicieni, ingineri, cât si de informaticieni sau umanisti). In plus limbajul BASIC este livrat adesea împreuna cu sistemul de operare DOS fiind astfel prezent pe foarte multe calculatoare si deci accesibil. Luînd în considerare toate aceste aspecte rezultatele programelor ce vor fi prezentate în continuare vor putea fi cu usurinta reproduse pe aproape orice calculator.

Mentionam ca fiecare limbaj de programare de nivel înalt are câte o procedura specializata în generarea de valori pseudoaleatoare ce sunt uniform repartizate în intervalul (0, 1].

3. Implementarea si rezultatele algoritmului de verificare

Algoritmul ce urmareste verificarea proprietatilor (P1)-(P3) a fost programat si rulat în limbajul MS-DOS Quick-Basic, versiunea 1.1, programul astfel obtinut putând fi preluat fara probleme în oricare dintre versiunile Quick-Basic ulterioare (eventual Visual Basic).

Pentru simplificare expunerii vom considera ca domeniul $\mathbf{R}_0$ cuprinde numai acele valori numerice reale ce sunt memorate în calculator în precizie dubla.

Procedura **RND** din limbajul Q-BASIC realizeaza generarea de numere aleatoare uniform repartizate în intervalul (0 , 1] . Aceasta procedura utilizeaza o metoda multiplicativ congruentiala pentru producerea termenilor pseudo-aleatori x_n , $x_n \in (0 , 1]$, $n \in N$, valorile x_n satisfacând relatia

$$x_{n+1} \equiv d \, . \, x_n \quad (\text{ modulo } M)$$

Constantele d si M sunt selectate în urma aplicarii unor teste statistice complexe ce vizeaza proprietatile statistice ale termenilor x_n , $n \in \mathbf{N}$, dedusi recurent.

Tinând seama de modul de proiectare a generatorului **RND** rezulta ca pentru obtinerea unui sir $\{ x_n \}_n$ de numere pseudo-aleatoare este nevoie sa se precizeze valoarea de început x_0 a acestui sir. Asadar pentru producerea unor secvente diferite $x_1 , x_2 , \ldots , x_n , \ldots$ generatorul **RND** de numere pseudo-aleatoare ar trebui initializat, de fiecare data indicându-se "samânta" sa x_0 .

In continuare intentionam de a pune în evidenta, prin exemple concrete, anumite caracteristici privind particularitatile de desfasurare în calculator a operatiilor aritmetice .

3.1. Operatia de înmultire cu calculatorul nu este asociativa

Daca U_1 , U_2 , U_3 sunt trei numere pseudo-aleatoare produse cu ajutorul generatorului **RND** vom considera valorile $a = U_1$, $b = a^4 + U_2$, $c = a^6 + U_3$.

Urmarim sa verificam daca, în acest caz, expresiile aritmetice e_1 , e_2 iau valori diferite, $e_1 \neq e_2$, unde $e_1 = a \, . \, (b \, . \, c)$, $e_2 = (a \, . \, b) \, . \, c$.

Urmatorul program va evidentia faptul ca operatia de înmultire cu calculatorul a numerelor reale a#, b#, c# (considerate în precizie dubla) nu este asociativa.

```
CLS :          INPUT "Initializare generator = "; x% :          RANDOMIZE x%
n = 30000 :          m = 0
FOR j = 1 TO n :          a# = RND :          b# = a# * a# * a# * a# + RND
c# = a# * a# * a# * a# * a# * a# + RND :     e1# = a# * (b# * c#) :     e2# = (a# * b#) * c#
IF e1# <> e2# THEN m = m + 1:          PRINT "j = "; j; "     a = "; a#
NEXT j
```

PRINT "In "; m; " cazuri din "; n; " nu s-a verificat asociativitatea inmultirii"

Rulând acest program BASIC s-au evidentiat cazuri în care nu este verificata proprietatea de asociativitate a operatiei de inmultire (valoarea expresiei e1# este diferita de valoarea expresiei e2#). Pentru a putea fi regasite cu usurinta toate aceste situatii de exceptie, au fost listate atât valoarile $a\# \equiv U_j$ ce au fost folosite în calcule cât si pozitia j a termenului pseudo-aleator $x_j \equiv U_j$ utilizat, precizându-se de fiecare data "samânta" x_0 a generatoruluiu **RND** . Prin executarea programului prezentat s-au obtinut rezultatele :

```
Initializare generator = ? 32456
j =  718     a =  .1698784828186035          j =  4552    a =  .3435925245285034
j =  5837    a =  .188832700252533           j =  6928    a =  .9925082921981812
j =  6958    a =  .2716565132141113          j =  8029    a =  .3550838828086853
j =  9614    a =  5.150461196899414D-02      j =  10714   a =  .1855247020721436
j =  11837   a =  2.134740352630615D-03      j =  12153   a =  .4234206080436707
j =  14438   a =  .960261344909668           j =  16021   a =  .7946633696556091
j =  19756   a =  .8023220300674438          j =  20670   a =  .8730692863464355
j =  21447   a =  .2427275776863098          j =  25030   a =  .8427858352661133
j =  25474   a =  .4021952152252197          j =  28301   a =  .3995298743247986
j =  28919   a =  .6338456273078918          j =  29717   a =  .8280267119407654
In  20  cazuri din 30000 nu s-a verificat asociativitatea inmultirii
```

In urma acestui rezultat aparent surprinzator, ne asteptam ca si alte proprietati ale operatiilor aritmetice sa nu fie verificate pe calculator.

3.2. Inmultirea cu calculatorul nu este distributiva în raport cu operatia de adunare

Datorita erorilor de "rotunjire" din calculator, nu întotdeauna avem distributivitatea înmultirii fata de adunare.

Vom pastra în programul BASIC anterior aceleasi valori a, b, c ($a = U_1$, $b = a^4 + U_2$, $c = a^6 + U_3$, unde U_1, U_2, U_3 sunt valori aleatoare uniform repartizate pe intervalul [0 , 1]), schimbând însa corespunzator forma expresiilor aritmetice e_1 si e_2 . Avem astfel $e_1 = a \cdot (b + c)$, $e_2 = (a \cdot b) + (a \cdot c)$.

Rulând noul program astfel modificat, pentru diferite valori de initializare x_0 ale generatorului **RND** , s-au obtinut triplete de numere (a, b, c) ce nu verifica proprietatea de distributivitate a înmultirii fata de adunare. Pentru cazurile de exceptie gasite sunt listate valorile $a \equiv x_j$, j , x_0

```
Initializare generator = ? 22334
j =  4735    a =  .7724292874336243          j =. 7931    a =  .8537970185279846
j =  13397   a =  .9158830046653748          j =  21198   a =  .6997551918029785
In  4  cazuri din 30000 nu s-a verificat distributivitatea inmultirii
Initializare generator = ? 29123
j =  5087    a =  .9845359921455383          j =  13970   a =  .4067223072052002
In  2  cazuri din 30000 nu s-a verificat distributivitatea inmultirii
```

Initializare generator = ? 28876
j = 2764 a = .361341118812561 j = 7574 a = .3680839538574219
j = 8167 a = .700153648853302 j = 20046 a = .5844826698303223
j = 29106 a = .3929646015167236
In 5 cazuri din 30000 nu s-a verificat distributivitatea inmultirii

3.3. Operatia de adunare cu calculatorul nu este asociativa

Pentru a demonstra ca operatia de adunare pe calculator nu este asociativa s-a utilizat un program similar cu cel deja prezentat în sectiunea 3.1, dar în care $a = U$, $b = a^3$, $c = -a^6$, unde U este un numar aleator uniform repartizat în intervalul (0 , 1].

De asemenea forma expresiilor aritmetice e_1 si e_2 a fost adaptata corespunzator. Efectuând aceste modificari s-au obtinut cazuri U pentru care, datorita erorilor de rotunjire pe calculator, valorile date de expresiile $e_1 = a + (b + c)$ si $e_2 = (a + b) + c$ sunt distincte. Asadar operatia aritmetica de adunare nu este asociativa pe calculator, fapt confirmat experimental de rezultatele executarii noului program BASIC propus :

Initializare generator = ? 22222
j = 24121 a = 1.935356855392456D-02
In 1 cazuri din 30000 nu s-a verificat asociativitatea adunarii
Initializare generator = ? 16162
j = 2637 a = 7.551133632659912D-03
In 1 cazuri din 30000 nu s-a verificat asociativitatea adunarii
Initializare generator = ? 6666
j = 24889 a = 1.935356855392456D-02
In 1 cazuri din 30000 nu s-a verificat asociativitatea adunarii
Initializare generator = ? 10203
j = 24527 a = 6.588995456695557D-03
In 1 cazuri din 30000 nu s-a verificat asociativitatea adunarii

Rularea ultimului program considerând diverse initializari x_0 a pus în evidenta foarte putine cazuri de exceptie. Nu de putine ori au existat secvente întregi de câte n = 30000 triplete de valori numerice (a, b, c) pentru care s-a verificat în totalitate proprietatea de asociativitate a operatiei de adunare. In vederea evidentierii unor situatii de exceptie a fost schimbata de mai multe ori "samânta" x_0 a generatorului **RND** de numere pseudoaleatoare.

4. Concluzii

Operatiile aritmetice desfasurate pe calculator presupun utilizarea unui numar finit, relativ restrâns, de cifre semnificative pentru valorile numerice, aproximarea numerelor efectuându-se prin "rotunjire". Intr-o astfel de situatie, la evaluarea unor expresii aritmetice pe calculator, nu sunt respectate întotdeauna proprietatile obisnuite de asociativitate si distributivitate ale operatiilor aritmetice cu numere reale.

Prin exemplele de test propuse a fost semnalata aceasta "anomalie".

Mentionam faptul ca sunt relativ putine cazurile în care, pe calculator, nu sunt adevarate proprietatile (P1), (P2), (P3). Necunoasterea acestei particularitati a calculatorului poate însa conduce la erori de programare grave, foarte greu detectabile deoarece se întâmpla extrem de rar, numai în unele "situatii de exceptie".

Tinând seama de aceste observatii, la proiectarea aplicatiilor de tip "balanta" nu se va efectua testarea egalitatii a doua expresii aritmetice e_1 si e_2 printr-o instructiune de control de forma

" **daca** $e_1 = e_2$ **atunci** actiune ... "

ci se va prefera varianta

" **daca** $|e_1 - e_2| < \varepsilon$ **atunci** actiune ... "

unde ε este o cantitate pozitiva, relativ mica, destinata sa anihileze efectul erorilor aritmetice de "rotunjire".

Problemele mentionate pot apare frecvent în practica în operatiuni ce vizeaza interpretarea automata, pe calculator, a rezultatelor unor "balante contabile" în functie de care sunt luate anumite decizii (Stefan Stefanescu : Erori de calcul la evaluarea unei balante, Studii si Cercetari de Calcul Economic si Cibernetica Economica, vol. XXXIII, nr. 2 / 1999).

TEHNICI DE SORTARE

In literatura de specialitate întâlnim o mare varietate de proceduri de sortare ce sunt analizate în raport cu performantele pe care le realizeaza, ca de exemplu durata medie a unei sortari, numarul cel mai mare de operatii necesare unui anume tip de sortare , memoria interna si externa utilizata de algoritmul de sortare.

Vom prezenta în detaliu câteva tehnici ce stau la baza algoritmilor de sortare (sortare prin insertie, aflarea minimului relativ, interclasare) fara a descrie si analiza unele proceduri subtile si performante ce urmaresc sortarea elementelor unei liste (de exemplu proceduri probabiliste ce tin seama de distributia initiala a datelor de intrare, algoritmi rapizi bazati pe principiul "divide si stapâneste" , implementarea unor proceduri recurente). Detalii suplimentare privind descrierea unor noi algoritmi de sortare pot fi obtinute consultând monografia lui Donald E. Knuth, " Tratat de programare a calculatoarelor - Sortare si cautare ", Editura Tehnica, Bucuresti, 1976.

1. Sortare prin insertie

In cazul în care se doreste sortarea (crescatoare) a celor n elemente ale unui vector V putem opta pentru tehnica insertiei fiecarui nou element V(m+1) într-o pozitie corespunzatoare. Concret, sa presupunem ca în vectorul V s-a reusit sortarea crescatoare a primelor m elemente, $m < n$. Initial $m = 1$. In acest context urmeaza sa stabilim noua pozitie a elementului curent V(m+1). Elementul V(m+1) va migra succesiv permutând cu elementele V(j) situate pe pozitii din stânga sa, $j = m, m-1, m-2, k+1, k$, $k \geq 1$, operatiunea oprindu-se în momentul în care $V(k-1) \leq V(m+1) < V(k)$. In urma acestor operatii de permutare a componentelor lui V este gasit locul corespunzator elementului V(m+1) astfel încât, în noua situatie, primele m+1 elemente ale vectorului V sunt ordonate crescator (adica $V(1) \leq V(2) \leq V(3) \leq \ldots \leq V(m-1) \leq V(m) \leq V(m+1)$). Aceasta etapa se va repeta pentru fiecare element V(m+1) nou introdus, unde $m = 1, 2, 3, \ldots, n-1$.

Aplicatia 1. Listam programul sursa ce realizeaza sortarea prin insertie a primelor n elemente din vectorul V .

```
REM  Sortare prin insertie
DATA 1,4,-5,2,4,2,7,5,3,8,9,-4
INPUT "Numar de date = "; n :                DIM v(n)
REM  Citeste datele de intrare
FOR j = 1 TO n:          READ v(j):          NEXT j
FOR m = 2 TO n:            w = v(m):         j = m - 1
WHILE ((j > 0) AND (v(j) > w))
  v(j + 1) = v(j):       j = j - 1 :         WEND
v(j + 1) = w :           NEXT m
PRINT "Rezultate : "; :        FOR j = 1 TO n:     PRINT v(j); " , "; :     NEXT j
```

Exemplul 1. Vom executa programul de sortare proiectat în Aplicatia 1, considerând o dimensiune variabila n a vectorului V. Mentionam faptul ca valorile vectorului V sunt retinute într-

un bloc de date, din care, în momentul rularii programului, sunt citite succesiv n valori numerice. In continuare prezentam rezultatele de test.

```
Numar de date = ? 11
Rezultate : -5 , 1 , 2 , 2 , 3 , 4 , 4 , 5 , 7 , 8 , 9 ,
Numar de date = ? 12
Rezultate : -5 , -4 , 1 , 2 , 2 , 3 , 4 , 4 , 5 , 7 , 8 , 9 ,
```

Observatia 1. Metoda descrisa poate fi extinsa si în situatia utilizarii memoriei externe, de exemplu în cazul sortarii a n înregistrari ale unui fisier, sortarea efectuându-se dupa un câmp dat ce ia valorile V(1), V(2), V(3), ..., respectiv V(n) . In acest context, pentru micsorarea volumului de calcule, este de dorit sa se efectueze un numar cât mai mic de permutari ale înregistrarilor fisierului respectiv. In acest scop înregistrarile ce trebuiesc sortate vor fi aduse pe pozitiile corespunzatoare numai în faza finala, anume atunci când au fost deja aranjate în ordine (crescatoare) toate numerele V(m) de identificare a înregistrarilor prelucrate.

Observatia 2. Sugeram si o posibila îmbunatatire a performantei prezentei proceduri. Astfel, tinându-se seama de distributia initiala a datelor de intrare, se anticipeaza aproximativ (în mod determinist sau probabilist) pentru fiecare element V(m) , $1 \le m \le n$, viitoarea sa pozitie ce ar trebui sa rezulte în urma operatiei de sortare. In acest mod se poate reduce volumul de calcule prin evitarea efectuarii unor permutari suplimentare ale componentelor vectorului V.

2. Aflarea minimului relativ

Daca dorim o sortare (crescatoare) a celor n componente ale vectorului V vom putea aplica si urmatoarea procedura axata pe aflarea minimului valorilor unei multimi date.

Luându-se în considerare toate cele n elemente ale vectorului V se determina în primul rând cel mai mic element $V(k_1)$ ce va fi apoi adus în locul elementului V(1) prin efectuarea unei permutari între V(1) si $V(k_1)$. In etapa urmatoare se obtine minimul $V(k_2)$ al elementelor lui V de pe pozitiile 2, 3, 4, ... , n , $V(k_2)$ = minimum { V(2) , V(3) , V(4) , ... , V(n) }. Dupa aflarea valorii indicelui k_2 elementul $V(k_2)$ va fi trecut în vectorul V pe pozitia a doua printr-o permutare între elementele V(2) si $V(k_2)$. In general, prin determinarea succesiva a minimului $V(k_j)$, $1 \le j \le n-1$, relativ la multimea { V(j) , V(j+1) , V(j+2) , ... , V(n) } si aducerea elementului $V(k_j)$ pe cea de a j-a pozitie în V (printr-o permutare a elementelor V(j) si $V(k_j)$) se obtine în final o ordonare crescatoare a componentele vectorului V.

Aplicatia 2. Programul urmator exemplifica procedura de sortare crescatoare bazata pe aflarea minimului relativ al elementelor unei submultimi.

```
REM  Sortare prin determinarea minimului relativ
DATA 1,4,-5,2,4,2,7,5,3,8,9,-4
INPUT "Numar de date = "; n :                    DIM v(n)
REM  Citeste datele de intrare
FOR j = 1 TO n:          READ v(j):              NEXT j
FOR m = 1 TO n - 1:           kmin = m:
   FOR k = m + 1 TO n
   IF v(k) < v(kmin) THEN kmin = k
```

```
   NEXT k
   IF kmin <> m THEN vmin = v(kmin) :        v(kmin) = v(m):        v(m) = vmin
NEXT m
PRINT "Vector sortat : "; :      FOR j = 1 TO n:      PRINT v(j); " , "; :      NEXT j
```

Exemplul 2. Prezentam rezultatele de test obtinute prin rularea programului BASIC din Aplicatia 2. Valorile componentelor vectorului V sunt citite succesiv dintr-un bloc DATA prin intermediul instructiunii READ.

```
Numar de date = ? 5
Vector sortat : -5 ,  1 ,  2 ,  4 ,  4 ,
Numar de date = ? 10
Vector sortat : -5 ,  1 ,  2 ,  2 ,  3 ,  4 ,  4 ,  5 ,  7 ,  8 ,
Numar de date = ? 12
Vector sortat : -5 ,  -4 ,  1 ,  2 ,  2 ,  3 ,  4 ,  4 ,  5 ,  7 ,  8 ,  9 ,
```

3. Sortare prin interclasare

Vom prezenta un algorim de interclasare a doi vectori A si B de dimensiune m , respectiv n , cu elemente numere reale, unde $A = (a_1 , a_2 , \dots , a_m)$, $B = (b_1 , b_2 , \dots , b_n)$. Este esential ca cei doi vectori A, B ale caror valori urmeaza a fi interclasate sa aiba elementele ordonate crescator, adica $a_1 \le a_2 \le a_3 \le \dots \le a_m$, respectiv $b_1 \le b_2 \le b_3 \le \dots \le b_n$. Vom construi vectorul V care sa contina toate elementele vectorilor A si B si în plus elementele lui V sa fie aranjate în ordine crescatoare. In final dimensiunea vectorului V devine m+n . Aducem urmatoarele precizari :

Indicii i , j , k definesc pozitia elementului curent din vectorii A , B , respectiv V .

Initial i = 1 , j = 1 si k = 0 .

Vom analiza pe rând perechile de valori (A(i) , B(j)) pentru $1 \le i \le m$ si $1 \le j \le n$. Astfel daca A(i) < B(j) atunci vom opera modificarile k = k+1 , V(k) = A(i) , având grija ca în final sa fie actualizata valoarea curenta a indicelui i , adica i = i+1. Asadar valoarea elementului A(i) va fi retinuta în vectorul V pe pozitia k . In mod analog, în situatia în care A(i) > B(j) vom efectua operatiile k = k+1 , V(k) = B(j) , j = j+1, valoarea B(j) fiind astfel atribuita componentei curente V(k) . Daca A(i) = B(j) atunci vom efectua secventa de instructiuni : k = k+1 , V(k) = A(i) , i = i+1 , k = k+1 , V(k) = B(j) , j = j+1 , ambii termeni A(i) , B(j) fiind inclusi printre elementele vectorului V.

In final se ajunge la una dintre situatiile j > n , sau i > m . In acest caz toate elementele A(s) , $i \le s \le m$, respectiv toate elementele B(t) , $j \le t \le n$, vor deveni în mod automat elemente ale vectorului V. Prin aceste operatii vectorul V ramâne ordonat crescator.

Aplicatia 3. Algoritmul de interclasare descris anterior este implementat în limbajul BASIC . Componentele vectorilor A si B ce urmeaza a fi interclasate sunt retinute într-un bloc de date ce reuneste toate secventele de date definite prin instructiunile DATA ale programului.

Listam programul sursa de interclasare.

```
REM  Interclasarea vectorilor A si B ordonati crescator
DECLARE SUB intercl (x!(), y!(), z!(), m!, n!)
DECLARE SUB scrie (x!(), p!) :              DECLARE SUB ordinecr (x!(), p!)
m = 7 :        n = 5 :        DIM a(m), b(n), v(m + n)
```

```
DATA 1,5,9,17,23,24,25                  ' Valorile vectorului A
REM  Citirea vectorului A
FOR i = 1 TO m :        READ a(i) :        NEXT i
PRINT "Valorile vectorului A : " ;:        CALL scrie(a(), m)
CALL ordinecr(a(), m)        ' Verificarea ordonarii crescatoare a elementelor vectorului A
DATA 2,3,5,19,23                        ' Valorile vectorului B
REM  Citirea vectorului B
FOR j = 1 TO n :        READ b(j) :        NEXT j
PRINT "Valorile vectorului B : "; :        CALL scrie(b(), n)
CALL ordinecr(b(), n)        ' Verificarea ordonarii crescatoare a elementelor vectorului B
CALL intercl(a(), b(), v(), m, n)          ' Interclasarea lui A cu B
REM  Editarea vectorului V
PRINT "Interclasare : "; :                 CALL scrie(v(), m + n)

SUB intercl (x(), y(), z(), m, n)
REM  Se va realiza interclasarea vectorilor X si Y ce au dimensiunile m , n
i = 1 :          j = 1 :          k = 0
DO WHILE (i <= m) AND (j <= n)
  IF x(i) = y(j) THEN
    k = k + 1:      z(k) = x(i):      i = i + 1
    k = k + 1:      z(k) = y(j):      j = j + 1 :      GOTO 1
  END IF
  IF x(i) < y(j) THEN      k = k + 1 :      z(k) = x(i) :      i = i + 1 :      GOTO 1
  IF x(i) > y(j) THEN      k = k + 1 :      z(k) = y(j) :      j = j + 1 :      GOTO 1
1 :  LOOP
DO WHILE (i <= m)
  k = k + 1 :          z(k) = x(i) :          i = i + 1
LOOP
DO WHILE (j <= n)
  k = k + 1 :          z(k) = y(j) :          j = j + 1
LOOP
END SUB

SUB ordinecr (x(), p)
REM  Se verifica ordonarea crescatoare a valorilor elementelor vectorului X
REM  Dimensiunea vectorului X este p
FOR k = 2 TO p
IF x(k - 1) > x(k) THEN PRINT "Eroare, valorile nu sunt in ordine crescatoare":   STOP
NEXT k
END SUB

SUB scrie (x(), p)
REM  Editarea vectorului X de dimensiune p
FOR k = 1 TO p :          PRINT x(k); " , "; :          NEXT k :          PRINT
END SUB
```

Observatia 3. In urma procesului de interclasare, în matricea finala V aceeasi valoare poate apare de mai multe ori. Procedura de interclasare presupune în mod obligatoriu ordonarea crescatoare a vectorilor A si B . In caz contrar, prin aplicarea algoritmului descris, componentele vectorului V ar putea în final sa nu fie ordonate crescator. Subrutina **ordinecr** ne va avertiza printr-un mesaj, dupa care va întrerupe automat executia programului, în situatia în care elementele vectorilor A si B nu sunt ordonate crescator.

Exemplul 3. Executarea programului prezentat în Aplicatia 3 a condus la urmatoarele rezultate de test :

Valorile vectorului A : 1 , 5 , 9 , 17 , 23 , 24 , 25 ,
Valorile vectorului B : 2 , 3 , 5 , 19 , 23 ,
Interclasare : 1 , 2 , 3 , 5 , 5 , 9 , 17 , 19 , 23 , 23 , 24 , 25 ,

Observatia 4. Procedura de interclasare a valorilor a doi vectori A si B poate fi cu usurinta extinsa si în cazul operarii cu mai multi vectori . Vectorii ce urmeaza a fi interclasati, ordonati în prealabil, ar putea fi memorati în coloanele unei matrici auxiliare.

Siruri definite recurent

Fie sirul $\{ x_n \}_n$, ai carui termeni sunt legati printr-o relatie de recurenta de tipul

$$x_{n+k} = f(x_n , x_{n+1} , x_{n+2} , \dots , x_{n+k-1} , n+k) \qquad n \in N \tag{1}$$

si care, în plus, respecta conditiile initiale

$$x_j = c_j \quad , \quad 1 \le j \le k \tag{2}$$

unde forma functiei $f(t_1 , t_2 , \dots , t_k)$ este data iar constantele $c_1 , c_2 , c_3 , \dots , c_k$ sunt precizate

Relatii (1) si (2) sunt suficiente pentru determinarea tuturor termenilor sirului $\{ x_n \}_n$. In adevar, stiind valorile $x_1, x_2 , x_3 ,\dots, x_k$ obtinem valoarea x_{k+1}, unde $x_{k+1} = f(x_1 ,x_2 ,x_3 ,\dots ,x_k)$ Urmeaza apoi a fi evaluata valoarea x_{k+2} aplicând formula $x_{k+2} = f (x_2 , x_3 , x_4 , \dots , x_{k+1})$, dupa care se calculeaza termenii $x_{k+3} , x_{k+4} ,\dots , x_{k+n} , \dots$ s.a.m.d.

Aplicatia 1. Fie functia liniara f de k argumente,

$$f (t_1 , t_2 , t_3 , \dots , t_k) = a_1 t_1 + a_2 t_2 + a_3 t_3 + \dots + a_{k-1} t_{k-1} + a_k t_k \tag{3}$$

unde $a_1 , a_2 , a_3 , \dots , a_k$ sunt numere reale.

In acest caz relatia de recurenta (1) devine

$$x_{n+k} = a_1 x_n + a_2 x_{n+1} + a_3 x_{n+2} + \dots + a_{k-1} x_{n+k-2} + a_k x_{n+k-1} \tag{4}$$

Programul urmator editeaza primii n + k termeni ai sirului $\{ x_j \}_j$ utilizând formula (4). Valorile $x_1 , x_2 , x_3 , \dots , x_k$ sunt introduse ca date de intrare.

```
REM  Siruri numerice obtinute prin relatii de recurenta
REM  x(n+k) = f(x(n), x(n+1), x(n+2),..., x(n+k-1))
DECLARE FUNCTION f! (x!(), a!(), k!, m!) :      INPUT "Numarul valorilor initiale = "; k
INPUT "Numarul termenilor sirului X ce urmeaza a se calcula = "; n :      DIM a(k), x(n + k)
PRINT "Specificarea parametrilor functiei f"
FOR j = 1 TO k :      PRINT USING "a(## ) = "; j; :      INPUT a(j) :      NEXT j
PRINT "Precizarea celor k valori initiale ale sirului X"
FOR j = 1 TO k :      PRINT USING "x(## ) = "; j; :      INPUT x(j) :      NEXT j
REM  Determinarea termenilor x(k+1) , x(k+2) , x(k+3) ,... , x(k+n)
FOR j = k + 1 TO k + n :          x (j) = f (x(), a(), k, j) :          NEXT j
REM  Editarea primilor n+k termeni ai sirului X
PRINT "Primii "; n + k; " termeni ai sirului X"
FOR j = 1 TO k + n :          PRINT x(j); "    "; :          NEXT j :          PRINT

FUNCTION f (x(), a(), k, m)
s = 0 :      FOR j = 1 TO k :      s = s + x(m - j) * a(k + 1 - j) :      NEXT j
f = s :      END FUNCTION
```

Considerând $k = 2$, $n = 15$, $a_1 = a_2 = 1$ si $x_1 = 1$, $x_2 = 2$ programul va edita primele 17 valori ale unui sir de tip Fibonacci (sir în care valoarea unui termen este suma celor doi termeni anteriori). In urma testarii programului BASIC s-au obtinut rezultatele :

```
Numarul valorilor initiale = ?  2
Numarul termenilor sirului  X  ce urmeaza a se calcula = ?  15
```

Specificarea parametrilor functiei f a(1) = ? 1 a(2) = ? 1
Precizarea celor k valori initiale ale sirului X x(1) = ? 1 x(2) = ? 2
Primii 17 termeni ai sirului X
1 2 3 5 8 13 21 34 55 89 144 233 377
610 987 1597 2584

Observatia 1. Valorile parametrilor $a_1, a_2, a_3, \dots, a_k$ ce definesc functia f pot sa nu fie constante ci sa depinda efectiv de n .

In continuare vom analiza siruri $\{ x_j \}_j$ a caror termeni verifica egalitatea (4). Acestor siruri li se va atasa ecuatia caracteristica $P(x) = 0$, unde

$$P(x) = x^k - a_k x^{k-1} - a_{k-1} x^{k-2} - \dots - a_3 x^2 - a_2 x - a_1 \qquad (5)$$

$a_1 \in R$, $a_2 \in R$, ... , $a_k \in R$.

In aceste conditii avem :

Propozitia 1. Daca sirurile $\{ x_n \}_n$, $\{ y_n \}_n$ sunt solutii ale ecuatiei (4) atunci pentru orice $\alpha \in R$, $\beta \in R$ sirul $\{ z_n \}_n$ verifica relatia (4) , unde $z_n = \alpha x_n + \beta y_n$, $\forall n \in N$.

Justificare. In adevar, cum $\{ x_n \}_n$, $\{ y_n \}_n$ sunt solutii ale ecuatiei (4), adica

$$x_{n+k} = a_1 x_n + a_2 x_{n+1} + a_3 x_{n+2} + \dots + a_{k-1} x_{n+k-2} + a_k x_{n+k-1}$$
$$y_{n+k} = a_1 y_n + a_2 y_{n+1} + a_3 y_{n+2} + \dots + a_{k-1} y_{n+k-2} + a_k y_{n+k-1}$$

pentru orice $n \in N$. Considerând $\alpha \in R$, $\beta \in R$, din egalitatile anterioare deducem

$$\alpha x_{n+k} + \beta y_{n+k} = a_1 (\alpha x_n + \beta y_n) + a_2 (\alpha x_{n+1} + \alpha y_{n+1}) + \dots + a_k (\alpha x_{n+k-1} + \beta y_{n+k-1})$$

Tinând seama de definitia termenului z_n avem

$$z_{n+k} = a_1 z_n + a_2 z_{n+1} + a_3 z_{n+3} + \dots + a_{k-1} z_{n+k-2} + a_k z_{n+k-1}$$

adica sirul $\{ z_n \}_n$ este solutie a ecuatiei (4).

Propozitia 2. Daca ecuatia caracteristica $P(x) = 0$ are o radacina t atunci sirul $\{ x_n \}_n$ definit prin $x_n = t^n$, $n \in N$, este solutie a ecuatiei (4) .

Demonstratie. Deoarece valoarea t , nu neaparat numar real, este radacina a polinomului P (x) definit de (5) , atunci

$$t^k - a_k t^{k-1} - a_{k-1} t^{k-2} - \dots - a_3 t^2 - a_2 t - a_1 = 0$$

Inmultind aceasta relatie cu t^n se obtine egalitatea

$$t^{n+k} - a_k t^{n+k-1} - a_{k-1} t^{n+k-2} - \dots - a_3 t^{n+2} - a_2 t^{n+1} - a_1 t^n = 0$$

ce este adevarata pentru orice $n \in N$.

Cu notatia $x_n = t^n$, expresia anterioara se rescrie în forma

$$x_{n+k} - (a_1 . x_n + a_2 . x_{n+1} + a_3 . x_{n+2} + \dots + a_k . x_{n+k-1}) = 0$$

adica sirul $\{ x_n \}_n$ este solutie a ecuatiei (4) .

Propozitia 3. Daca $t = \lambda (\cos(\theta) + i . \sin(\theta))$ este o radacina complexa a polinomului caracteristic (5) atunci sirurile $\{ y_n \}_n$, $\{ z_n \}_n$, unde $y_n = \lambda^n \cos(n\theta)$, $z_n = \lambda^n \cos(n\theta)$, $\forall n \in N$, sunt solutii ale ecuatiei (4).

Demonstratie. Cum t este o radacina a polinomului caracteristic (5), conform Propozitiei 2 rezulta ca sirul $\{ x_n \}_n$, unde $x_n = t^n = \lambda^n (\cos(n\theta) + i \sin(n\theta))$, este o solutie a ecuatiei (4) si deci

$$\lambda^{n+k} (\cos((n+k)\theta) + i \sin((n+k)\theta)) = x_{n+k} = a_1 x_n + a_2 x_{n+1} + a_3 x_{n+2} + \dots + a_k x_{n+k-1} =$$

$= a_1 [\lambda^n (\cos (n\theta) + i \sin (n\theta))] + a_2 [\lambda^{n+1} (\cos ((n+1)\theta) + i \sin ((n+1)\theta))] +$

$+ a_3 [\lambda^{n+2} (\cos ((n+2)\theta) + i \sin ((n+2)\theta))] + \ldots + a_k [\lambda^{n+k-1} (\cos ((n+k-1)\theta) + i \sin ((n+k-1)\theta))]$

Cum coeficientii $a_1, a_2, a_3, \ldots, a_k$ sunt numere reale, identificând partile reale si partile imaginare din egalitatea anterioara în care se opereaza cu numere complexe, rezulta

$\lambda^{n+k} \cos ((n+k)\theta) = a_1 \lambda^n \cos (n\theta) + a_2 \lambda^{n+1} \cos ((n+1)\theta) + \ldots + a_k \lambda^{n+k-1} \cos ((n+k-1)\theta)$

$\lambda^{n+k} \sin ((n+k)\theta) = a_1 \lambda^n \sin (n\theta) + a_2 \lambda^{n+1} \sin ((n+1)\theta) + \ldots + a_k \lambda^{n+k-1} \sin ((n+k-1)\theta)$

adica sirurile $y_n = \lambda^n \cos (n\theta)$ si $z_n = \lambda^n \cos (n\theta)$, $n \in N$, sunt solutii ale ecuatiei (4).

Propozitia 4. Daca t este o solutie reala, dubla, a ecuatiei caracteristice $P(x) = 0$ definita de relatia (5) atunci sirul $\{ y_n \}_n$, unde $y_n = n t^n$, $n \in N$, este solutie pentru ecuatia (4).

Demonstratie. Cum t este o solutie reala, dubla, a ecuatiei caracteristice $P(x) = 0$ atunci t va fi solutie dubla si pentru polinomul $Q(x)$,

$$Q(x) = x^{n+k} - a_k x^{n+k-1} - a_{k-1} x^{n+k-2} - \ldots - a_3 x^{n+2} - a_2 x^{n+1} - a_1 x^n$$

Prin urmare $Q(t) = 0$ si $(\partial Q(x) / \partial x)(t) = 0$, unde

$$\partial Q(x) / \partial x = (n+k) x^{n+k-1} - (n+k-1) a_k x^{n+k-2} - \ldots - (n+1) a_2 x^n - n a_1 x^{n-1}$$

si deci

$$(n+k) x^{n+k} - a_k (n+k-1) x^{n+k-1} - \ldots - a_3 (n+2) x^{n+2} - a_2 (n+1) x^{n+1} - a_1 n x^n = 0$$

adica sirul $y_n = n t^n$, $n \in N$, este solutie pentru ecuatia (4).

Urmând un rationament analog, aplicând Propozitia 1 si utilizând derivatele de ordin superior ale polinomului $Q(x)$ rezulta

Propozitia 5. Daca t este o radacina reala, multipla de ordinul p, $p \in N_*$, a polinomului caracteristic $P(x)$ definit de expresia (5), atunci sirul $\{ y_n \}_n$, unde $y_n = S(n) t^n$, $n \in N$, este solutie pentru ecuatia (4), unde $S(x)$ este un polinom oarecare, cu coeficienti reali, de grad cel mult p - 1.

Observatia 2. Propozitiile 2-5 ne sugereaza o procedura pentru a obtine solutii ale ecuatiei (4), fara însa ca aceste solutii sa verifice restrictiile (2). Stiind solutii ale ecuatiei (4) si aplicând Propozitia 1 putem construi noi solutii care în final sa satisfaca si conditiile initiale (2). Astfel, în cazul în care se cunosc k solutii "independente" $\{ x_n^{(1)} \}_n$, $\{ x_n^{(2)} \}_n$, $\ldots$, $\{ x_n^{(k)} \}_n$, ale ecuatiei (4), vom defini sirul $\{ y_n \}_n$ ce depinde de k parametri b_j, $1 \le j \le k$,

$$y_n = b_1 x_n^{(1)} + b_2 x_n^{(2)} + b_3 x_n^{(3)} + \ldots + b_{k-1} x_n^{(k-1)} + b_k x_n^{(k)}, \quad n \ge 1$$

Conform Propozitiei 1 sirul $\{ y_n \}_n$ astfel obtinut este solutie a ecuatiei (4). Parametrii b_j, $1 \le j \le k$, vor fi determinati astfel încât sa fie satisfacute conditiile (2). Deducerea coeficientilor b_j, $1 \le j \le k$, se reduce la solutionarea unui sistem liniar de k ecuatii cu k necunoscute,

$$\begin{aligned}
&b_1 x_1^{(1)} + b_2 x_1^{(2)} + b_3 x_1^{(3)} + \ldots + b_{k-1} x_1^{(k-1)} + b_k x_1^{(k)} = c_1 \\
&b_1 x_2^{(1)} + b_2 x_2^{(2)} + b_3 x_2^{(3)} + \ldots + b_{k-1} x_2^{(k-1)} + b_k x_2^{(k)} = c_2 \\
&b_1 x_3^{(1)} + b_2 x_3^{(2)} + b_3 x_3^{(3)} + \ldots + b_{k-1} x_3^{(k-1)} + b_k x_3^{(k)} = c_3 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&b_1 x_k^{(1)} + b_2 x_k^{(2)} + b_3 x_k^{(3)} + \ldots + b_{k-1} x_k^{(k-1)} + b_k x_k^{(k)} = c_k
\end{aligned}$$

Sirul $\{ y_n \}_n$ definit prin intermediul coeficientilor b_j, $1 \le j \le k$, este solutie a ecuatiei (4) ce verifica în plus si conditiile initiale (2), adica $y_j = c_j$, $1 \le j \le n$.

Exemplul 1. Sa determinam forma sirului $\{ x_n \}_n$ definit recurent de relatia $x_{n+2} = 5\, x_{n+1} - 6\, x_n$, cu conditiile initiale $x_1 = 1$, $x_2 = 5$.

In acest caz se obtine urmatoarea ecuatie caracteristica $t^2 - 5\, t + 6 = 0$ cu radacinile $t_1 = 2$ si $t_2 = 3$. Deci solutia generala $\{ x_n \}_n$ a ecuatiei $x_{n+2} = 5\, x_{n+1} - 6\, x_n$ este de forma $x_n = \alpha\, t_1^{\,n} + \beta\, t_2^{\,n} = \alpha\, 2^n + \beta\, 3^n$ unde parametrii α, β sunt numere reale. Vom determina valorile α si β astfel încât $x_1 = 1$ si $x_2 = 5$. Rezulta astfel sistemul liniar de doua ecuatii în necunoscutele α si β,

$$\begin{cases} \alpha 2^1 + \beta 3^1 = 1 \\ \alpha 2^2 + \beta 3^2 = 5 \end{cases}$$

de unde se deduce $\alpha = -1$ si $\beta = 1$.

Prin urmare solutia $\{ x_n \}_n$ a ecuatiei $x_{n+2} = 5\, x_{n+1} - 6\, x_n$ ce satisface conditiile initiale $x_1 = 1$, $x_2 = 5$ este de forma $x_n = 3^n - 2^n$, $n \geq 1$.

Prezentam rezultatele rularii programului BASIC din Aplicatia 1 privind determinarea primilor 15 termeni din sirul $\{ x_n \}_n$ ce este definit recurent conform relatiei $x_{n+2} = 5\, x_{n+1} - 6\, x_n$, $n \geq 1$, astfel încât $x_1 = 1$, $x_2 = 5$:

```
Numarul valorilor initiale = ?  2
Numarul termenilor sirului  X  ce urmeaza a se calcula = ?  15
Specificarea parametrilor functiei  f          a( 1 ) = ?  -6          a( 2 ) = ?  5
Precizarea celor  k  valori initiale ale sirului  X          x( 1 ) = ?  1          x( 2 ) = ?  5
Primii  17  termeni ai sirului  X
 1      5      19      65      211      665      2059      6305      19171      58025      175099
527345      1586131      4766585      1.431614E+07      4.298118E+07      1.290091E+08
```

Ultimele trei valori din sirul anterior nu au fost listate cu toate cifrele lor semnificative deoarece în programul respectiv vectorul X a fost definit ca fiind de tip real, în precizie simpla.

Urmatorul program va estima termenii sirului $\{ x_n \}_n$ aplicând formula $x_n = 3^n - 2^n$, $n \geq 1$, calculele desfasurându-se în precizie dubla. Functia g# este definita de relatia $g\#(n) = 3^n - 2^n$.

```
DECLARE FUNCTION g# (n!)
REM  Editarea valorilor unui sir definit explicit printr-o formula de calcul
INPUT "n = "; n :          FOR j = 1 TO n :          PRINT g#(j); "    "; :          NEXT j

FUNCTION g# (n) :          g# = 3# ^ n - 2# ^ n :          END FUNCTION
```

Rezultatele rularii ultimului program BASIC (pentru n = 17) confirma aspectele teoretice mentionate. De aceasta data valorile termenilor $\{ x_n \}_n$ sunt listate cu un numar suficient de cifre semnificative :

```
 1      5      19      65      211      665      2059      6305      19171      58025      175099      527345
1586131      4766585      14316139      42981185      129009091
```

Exemplul 2. Vom deduce forma sirului $\{ x_n \}_n$ definit recurent de relatia $x_{n+2} = 4\, x_{n+1} - 4\, x_n$, unde $x_1 = x_2 = -4$

Ecuatia caracteristica atasata relatiei recurente $x_{n+2} = 4\, x_{n+1} - 4\, x_n$ este de forma

$t^2 - 4t + 4 = 0$ si are radacinile $t_1 = t_2 = 2$. Asadar conform Propozitiei 5, solutia generala $\{ x_n \}_n$ ce verifica relatia $x_{n+2} = 4 x_{n+1} - 4 x_n$ este data de expresia $x_n = \alpha t_1^n + \beta n t_2^n$, adica $x_n = (\alpha + \beta n) 2^n$, unde parametrii α, β sunt numere reale. Valorile α si β vor fi obtinute impunându-se satisfacerea conditiilor initiale pentru termenii x_1 si x_2. Din sistemul

$$\begin{cases} (\alpha+\beta)2^1 = -4 \\ (\alpha+2\beta)2^2 = -4 \end{cases}$$

rezulta $\alpha = -3$ si $\beta = 1$.

Prin urmare solutia $\{ x_n \}_n$ a ecuatiei $x_{n+2} = 4 x_{n+1} - 4 x_n$ ce satisface conditiile initiale $x_1 = x_2 = -4$ este de forma $x_n = (n - 3) 2^n$, $n \geq 1$.

Prin executarea programului din Exemplul 1 pentru $n = 15$ unde functia $g\#(m)$ este de forma $g\#(m) = (m - 3) 2^m$ rezulta urmatoarea secventa de valori

-4 -4 0 16 64 192 512 1280 3072 7168
16384 36864 81920 180224 393216

Rezultate similare sunt obtinute folosind programul BASIC din Aplicatia 1 pentru deducerea succesiva a termenilor sirului $\{ x_n \}_n$ ce satisface relatia $x_{n+2} = 4 x_{n+1} - 4 x_n$, cu conditiile initiale $x_1 = x_2 = -4$,

Numarul valorilor initiale = ? 2
Numarul termenilor sirului X ce urmeaza a se calcula = ? 13
Specificarea parametrilor functiei f a(1) = ? -4 a(2) = ? 4
Precizarea celor k valori initiale ale sirului X x(1) = ? -4 x(2) = ? -4
Primii 15 termeni ai sirului X
-4 -4 0 16 64 192 512 1280 3072 7168
16384 36864 81920 180224 393216

Exemplul 3. Sirului $\{ x_n \}_n$ este definit recurent de relatia $x_{n+2} = 2 . 3^{1/2} x_{n+1} - 4 x_n$, cu conditiile initiale $x_1 = 2 . 3^{1/2} + 1$, $x_2 = 2 . 3^{1/2} + 4$.

In acest caz ecuatie caracteristica este de forma $t^2 - 2 . 3^{1/2} t + 4 = 0$, cu radacinile t_1, t_2 numere complexe, $t_1 = 2 (\cos (\pi / 6) + i \sin (\pi / 6))$, $t_2 = 2 (\cos (\pi / 6) - i \sin (\pi / 6))$.

Asadar solutia generala $\{ x_n \}_n$ a ecuatiei $x_{n+2} = 5 x_{n+1} - 6 x_n$ este data de expresia $x_n = \alpha 2^n \cos (n \pi / 6) + \beta 2^n \sin (n \pi / 6)$, unde parametrii α, β sunt numere reale. Parametrii α, β vor fi dedusi din conditiile initiale $x_1 = 2 . 3^{1/2} + 1$, $x_2 = 2 . 3^{1/2} + 4$, adica $\alpha = 2$, $\beta = 1$.

Prin urmare solutia $\{ x_n \}_n$ a ecuatiei $x_{n+2} = 2 . 3^{1/2} x_{n+1} - 4 x_n$ ce satisface conditiile initiale $x_1 = 2 . 3^{1/2} + 1$, $x_2 = 2 . 3^{1/2} + 4$ este $x_n = 2^{n+1} \cos (n \pi / 6) + 2^n \sin (n \pi / 6)$, $n \geq 1$

Prezentam rezultatele rularii programului proiectat în Aplicatia 1 în vederea calcularii primilor 15 termeni din sirul $\{ x_n \}_n$. Mentionam faptul ca sirul $\{ x_n \}_n$ este definit recurent conform relatiei $x_{n+2} = 2 . 3^{1/2} x_{n+1} - 4 x_n$, $n \geq 1$ cu conditiile initiale $x_1 = 2 . 3^{1/2} + 1$, $x_2 = 2 . 3^{1/2} + 4$

Numarul valorilor initiale = ? 2
Numarul termenilor sirului X ce urmeaza a se calcula = ? 13
Specificarea parametrilor functiei f a(1) = ? -4 a(2) = ? 3.464102

Precizarea celor k valori initiale ale sirului X x(1) = ? 4.464102 x(2) = ? 7.464102
Primii 15 termeni ai sirului X

4.464102	7.464102	8.000004	-2.143578	-39.42559	-127.9999	-285.7025	-477.7029
-512.0015	137.1862	2523.233	8191.991	18284.96	30573.01	32768.16	

Termenii sirului $\{ x_n \}_n$ pot fi calculati si direct dupa formula $x_n = g\#(n)$, functia g#(n) având expresia $g\#(n) = 2^{n+1} \cos (n \pi / 6) + 2^n \sin (n \pi / 6)$, $n \geq 1$. Vom considera procedura g# (n) utilizata în Exemplul 1 având forma

```
FUNCTION g# (n) :          u = n * 3.141592 / 6
g# = 2# ^ (n + 1) * COS(u) + 2# ^ n * SIN(u) :          END FUNCTION
```

In aceasta ultima varianta se obtin valori x_n comparabile cu cele deja deduse prin folosirea functiei $f(t_1 , t_2) = 2 . 3^{1/2} t_2 - 4 t_1$ (Aplicatia 1). Micile diferente numerice rezultate în cele doua variante prezentate sunt datorate preciziei impuse. Spre exemplificare listam primii 15 termeni ai sirului $\{ x_n \}_n$ calculati cu ajutorul functiei g#(n), anume

4.464101643175619	7.464102646500394	8.000005022663265	-2.143578591661494
-39.4255946008054	-127.9999598186655	-285.7025159297139	-477.7027673975264
-512.0009643511954	137.1875948786918	2523.235655111217	8191.994856785956
18284.96170854884	30572.98978680572	32768.10286411136	

SISTEME DE ECUATII LINIARE

In cadrul acestei sectiuni vom indica metode pentru obtinerea solutiei unui sistem liniar de n ecuatii cu n necunoscute. Aceste proceduri vor fi apoi aplicate la inversarea matricilor sau la calculul valorii unui determinant. Vom trata si problema "solutionarii" unui sistem linear cu mult mai multe ecuatii decât necunoscute.

1. Rezolvarea unui sistem liniar

Fie urmatorul sistem liniar de n ecuatii cu n necunoscute :

$$\begin{aligned}
&a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1,n-1}x_{n-1} + a_{1n}x_n = b_1\\
&a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2,n-1}x_{n-1} + a_{2n}x_n = b_2\\
&a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \ldots + a_{3,n-1}x_{n-1} + a_{3n}x_n = b_3\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&a_{j1}x_1 + a_{j2}x_2 + a_{j3}x_3 + \ldots + a_{j,n-1}x_{n-1} + a_{jn}x_n = b_j \qquad (1)\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{n,n-1}x_{n-1} + a_{nn}x_n = b_n
\end{aligned}$$

O solutie a sistemului (1) este un set de n valori numerice $x_1, x_2, \ldots, x_n$ ce verifica toate ecuatiile sistemului, coeficientii a_{jk} si b_j, $1 \le j, k \le n$, fiind numere reale date.

Sistemul (1) poate fi scris si sub forma matriciala $A\,x = b$ unde $A = (a_{jk})_{1 \le j,\, k \le n}$, $b = (b_j)_{1 \le j \le n}$, $x = (x_j)_{1 \le j \le n}$, adica

$$\begin{pmatrix}
a_{11} & a_{12} & a_{13} & & a_{1,n-1} & a_{1n}\\
a_{21} & a_{22} & a_{23} & & a_{2,n-1} & a_{2n}\\
a_{31} & a_{32} & a_{33} & & a_{3,n-1} & a_{3n}\\
\ldots & \ldots & \ldots & & & \ldots\\
a_{j1} & a_{j2} & a_{j3} & \ldots & a_{j,n-1} & a_{jn}\\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots\\
a_{n1} & a_{n2} & a_{n3} & & a_{n,n-1} & a_{nn}
\end{pmatrix}
\begin{pmatrix} x_1\\ x_2\\ x_3\\ \ldots\\ x_j\\ \ldots\\ x_n \end{pmatrix}
=
\begin{pmatrix} b_1\\ b_2\\ b_3\\ \ldots\\ b_j\\ \ldots\\ b_n \end{pmatrix} \qquad (2)$$

Precizam faptul ca sistemul (1) admite solutie unica daca determinantul matricii asociate A este nenul, $\det(A) \neq 0$.

1.1. Formula lui Cramer

O metoda generala de rezolvare a sistemului liniar de ecuatii (1) este data de procedura lui Cramer în care obtinerea solutiei $x = (x_1, x_2, \ldots, x_{n-1}, x_n)$ se reduce la calculul unor determinanti.

Aceasta tehnica nu este în general implementata pe calculator deoarece necesita o procedura specializata în calculul valorii unui determinant. Astfel componenta x_k, $1 \le k \le n$, a solutiei x a sistemului (1) este data de formula

$$x_k = \det(A_k) / \det(A) \qquad (3)$$

unde matricea A_k, $1 \le k \le n$, este obtinuta din matricea A prin înlocuirea coloanei k de vectorul coloana $(b_1, b_2, b_3, \ldots, b_n)$ al termenilor liberi, adica

$$A_k = \begin{pmatrix} a_{11} & a_{12} & & a_{1,k-1} & b_1 & a_{1,k+1} & & a_{1n} \\ a_{21} & a_{22} & & a_{2,k-1} & b_2 & a_{2,k+1} & & a_{2n} \\ \\ a_{n-1,1} & a_{n-1,2} & & a_{n-1,k-1} & b_{n-1} & a_{n-1,k+1} & & a_{n-1,n} \\ a_{n1} & a_{n2} & & a_{n,k-1} & b_n & a_{n,k+1} & & a_{nn} \end{pmatrix} \qquad (4)$$

Din (3) rezulta ca solutia $x = (x_1, x_2, \ldots, x_n)$ a sistemului (1) exista si este unica daca determinantul sistemului este nenul, adica $\Delta = \det(A) \neq 0$.

1.2. Sisteme triunghiulare

Un caz particular interesant de sistem liniar este sistemul inferior sau superior triunghiular în care matricea A asociata are nule toate elementele de deasupra diagonalei principale, respectiv de sub diagonala principala. Gasirea solutiei unui sistem triunghiular se realizeaza succesiv, prin metoda substitutiei, valoarea x_j a unei necunoscute obtinuta din una dintre ecuatii fiind apoi folosita în toate celelalte ecuatii.

Cazul 1. Pentru exemplificare vom considera sistemul superior triunghiular

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1,n-2}x_{n-2} + a_{1,n-1}x_{n-1} + a_{1n}x_n &= b_1 \\ a_{22}x_2 + a_{23}x_3 + \ldots + a_{2,n-2}x_{n-2} + a_{2,n-1}x_{n-1} + a_{2n}x_n &= b_2 \\ a_{33}x_3 + \quad + a_{3,n-2}x_{n-2} + a_{3,n-1}x_{n-1} + a_{3n}x_n &= b_3 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & \\ a_{n-2,n-2}x_{n-2} + a_{n-2,n-1}x_{n-1} + a_{n-2,n}x_n &= b_{n-2} \\ a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n &= b_{n-1} \\ a_{n,n}x_n &= b_{n-2} \end{aligned} \qquad (5)$$

Sistemul (5) va admite solutie daca $\det(A) \neq 0$ fapt ce revine la verificarea relatiei $a_{11}a_{22}a_{33}\cdots a_{n-1,n-1}a_{nn} \neq 0$ adica toate elementele diagonalei principale a matricei A sunt nenule. In aceasta situatie valoarea x_n este obtinuta din ultima ecuatie dupa care aceasta valoare este utilizata în penultima ecuatie pentru aflarea lui x_{n-1} s.a.m.d. In general, având din ultimele n-j ecuatii valorile x_n, x_{n-1}, x_{n-2}, $\ldots$, x_{j+1} se deduce si componenta x_j a solutiei x, $j = n-1, n-2, \ldots, 3, 2, 1$, în acest sens folosindu-se cea de a j-a ecuatie a sistemului triunghiular (5).

Deci $x_n = b_n / a_{nn}$ iar pentru $1 \le j \le n-1$ avem

$$x_j = (b_j - a_{j,j+1}x_{j+1} - a_{j,j+2}x_{j+2} - \cdots - a_{j,n-1}x_{n-1} - a_{j,n}x_n) / a_{jj} \qquad (6)$$

Aplicatia 1. Bazat pe formulele (6) programul BASIC urmator determina solutia unui sistem triunghiular de forma (5). Este semnalata situatia în care sistemul (5) nu admite solutie (din (6) nu poate fi dedusa valoarea x_j atunci când $a_{jj} = 0$).

```
REM   Solutia sistemelor superior triunghiulare
DECLARE SUB ssuptri (a!(), b!(), x!(), n!) :      INPUT "n = "; n :      DIM a(n, n), x(n), b(n)
PRINT "Matricea  A  :"              '  Initializarea matricii  A
```

```
FOR j = 1 TO n :   FOR k = j TO n :   PRINT "   a (", j, ", "; k; ") = ";   INPUT ; a(j, k)
NEXT k :          NEXT j
PRINT :          PRINT "Vectorul termenilor liberi : "
FOR j = 1 TO n :     PRINT "   b ("; j; ") = "; :     INPUT ; b(j) :     NEXT j       PRINT
CALL ssuptri(a(), b(), x(), n)
PRINT "Sistemul superior triunghiular de "; n; " ecuatii are solutia :"      ' Listarea solutiei
FOR j = 1 TO n :      PRINT USING "X(##) = ###.## , "; j; x(j); :       NEXT j
PRINT "Verificare solutiei gasite. Erori : "          ' Precizarea erorilor
FOR j = 1 TO n :       s = 0
FOR k = j TO n :        s = s + a(j, k) * x(k) :       NEXT k
s = b(j) - s :       PRINT "Er"; j; " = "; s; " , "; :        NEXT j

SUB ssuptri (a(), b(), x(), n)
IF a(n, n) = 0 THEN PRINT "Sistemul superior triunghiular nu admite solutie":     STOP
x(n) = b(n) / a(n, n)
FOR j = n - 1 TO 1 STEP -1
IF a(j, j) = 0 THEN PRINT "Sistemul superior triunghiular nu admite solutie":     STOP
s = 0 :       FOR k = j + 1 TO n :        s = s + a(j, k) * x(k) :       NEXT k
x(j) = (b(j) - s) / a(j, j) :          NEXT j :          END SUB
```

Pentru testarea programului prezentat în Aplicatia 1, dupa aflarea solutiei x a sistemului liniar $x = (x_1, x_2, \ldots, x_n)$, se calculeaza erorile ε_j, $1 \le j \le n$, unde

$$\varepsilon_j = b_j - a_{j1} x_1 - a_{j2} x_2 - \ldots - a_{jj} x_j - \ldots - a_{jn} x_n \tag{7}$$

In cazul particular al sistemului triunghiular (5) deoarece $a_{ij} = 0$ pentru orice $1 \le i < j \le n$ relatia (7) devine

$$\varepsilon_j = b_j - a_{jj} x_j - a_{j,j+1} x_{j+1} - a_{j,j+2} x_{j+2} - \ldots - a_{jn} x_n$$

Exemplul 1. Vom utiliza programul BASIC anterior pentru un sistem superior triunghiular $A x = b$ unde $n = 4$ si $a_{11} = -7$, $a_{12} = 9$, $a_{13} = 2$, $a_{14} = 3$, $a_{22} = -5$, $a_{23} = -8$, $a_{24} = 3$, $a_{33} = 4$, $a_{34} = 1$, $a_{44} = 9$, $a_{21} = a_{31} = a_{32} = a_{41} = a_{42} = a_{43} = 0$; $b_1 = -4$, $b_2 = 7$, $b_3 = 4$, $b_4 = 6$

Prin rularea programului s-a obtinut solutia : $x = (-1.90, -2.33, 0.83, 0.67)$.

Erorile ε_j, $1 \le j \le 4$, date de (7) au valori foarte mici, anume : $\varepsilon_1 = -1.430511E-06$, $\varepsilon_2 = 4.768372E-07$, $\varepsilon_3 = 0$, $\varepsilon_4 = 0$. Este astfel confirmata indirect corectitudinea programului BASIC propus în Aplicatia 1.

Cazul 2. Rezolvarea unui sistem inferior triunghiular se va desfasura într-un mod asemanator. Asfel solutia sistemului triunghiular

$$\begin{array}{llr} a_{11} x_1 & = b_1 & \\ a_{21} x_1 + a_{22} x_2 & = b_2 & \\ a_{31} x_1 + a_{32} x_2 + a_{33} x3 & = b_3 & \\ \ldots\ldots\ldots\ldots\ldots\ldots & & (8) \\ a_{n-2,1} x_1 + a_{n-2,2} x_2 + a_{n-2,3} x_3 + \quad + a_{n-2,n-2} x_{n-2} & = b_{n-2} & \\ a_{n-1,1} x_1 + a_{n-1,2} x_2 + a_{n-1,3} x_3 + \ldots + a_{n-1,n-2} x_{n-2} + a_{n-1,n-1} x_{n-1} & = b_{n-1} & \\ a_{n1} x_1 + a_{n2} x_2 + a_{n3} x_3 + \ldots + a_{n,n-2} x_{n-2} + a_{n,n-1} x_{n-1} + a_{nn} x_n & = b_n & \end{array}$$

va fi dedusa recurent dupa formulele

$$x_1 = b_1 / a_{11}$$

$x_j = (b_j - a_{j1} x_1 - a_{j2} x_2 - \dots - a_{j\,j-2} x_{j-2} - a_{j,j-1} x_{j-1}) / a_{jj}$, $2 \le j \le n$ (9)

Cum determinantul sistemului (8) este dat de produsul $\det(A) = a_{11} a_{22} a_{33} \dots a_{nn}$, sistemul (8) va admite solutie unica de forma (9) daca $\det(A) \neq 0$, adica $a_{ii} \neq 0$ pentru orice $1 \le i \le n$.

Aplicatia 2. Programul BASIC listat în continuare determina solutia $x = (x_1 , x_2 , \dots , x_n)$ a sistemului inferior triunghiular (8), semnalând si situatia în care acest sistem nu admite solutie.

```
REM  Solutia sistemelor inferior triunghiulare
DECLARE SUB sinftri (a!(), b!(), x!(), n!)
INPUT "n = "; n :                DIM a(n, n), x(n), b(n)
PRINT "Matricea A :" :           ' Initializarea matricii A
FOR j = 1 TO n
FOR k = 1 TO j :   PRINT "   a ("; j; ", "; k; ") = "; :   INPUT ; a(j, k) :   NEXT k :   NEXT j
PRINT :   PRINT "Vectorul termenilor liberi : "
FOR j = 1 TO n :   PRINT "   b ("; j; ") = "; :   INPUT ; b(j) :   NEXT j :   PRINT
CALL sinftri(a(), b(), x(), n)
PRINT "Sistemul inferior triunghiular de "; n; " ecuatii are solutia :" :   ' Listarea solutiei
FOR j = 1 TO n :   PRINT USING "X(##) = ###.##  , "; j; x(j); :   NEXT j
PRINT "Verificare solutiei gasite. Erori : "
FOR j = 1 TO n :   s = 0
FOR k = 1 TO j :   s = s + a(j, k) * x(k) :   NEXT k
s = b(j) - s :   PRINT "Er"; j; " = "; s; "  , "; :   NEXT j

SUB sinftri (a(), b(), x(), n)
IF a(1, 1) = 0 THEN PRINT "Sistemul inferior triunghiular nu admite solutie":   STOP
x(1) = b(1) / a(1, 1) :   FOR j = 2 TO n
IF a(j, j) = 0 THEN PRINT "Sistemul inferior triunghiular nu admite solutie":   STOP
s = 0 :   FOR k = 1 TO j - 1 :   s = s + a(j, k) * x(k) :   NEXT k
x(j) = (b(j) - s) / a(j, j) :   NEXT j :   END SUB
```

Verificarea acuratetii solutiei obtinute s-a realizat prin listarea erorilor ε_j, $1 \le j \le n$, date de formula (7), adica

$\varepsilon_j = b_j - a_{j1} x_1 - a_{j2} x_2 - a_{j,j-1} x_{j-1} - a_{jj} x_j$

Exemplul 2. Vom rezolva sistemul inferior triunghiular $A x = b$ unde $n = 5$ si $a_{11} = -3$, $a_{21} = 7$, $a_{22} = 4$, $a_{31} = 1$, $a_{32} = -9$, $a_{33} = 7$, $a_{41} = 2$, $a_{42} = 4$, $a_{43} = 6$, $a_{44} = 8$, $a_{51} = 5$, $a_{52} = 6$, $a_{53} = -4$, $a_{54} = -2$, $a_{55} = 9$, $a_{12} = a_{13} = a_{14} = a_{15} = a_{23} = a_{24} = a_{25} = a_{34} = a_{35} = a_{45} = 0$; $b_1 = 8$, $b_2 = 0$, $b_3 = -5$, $b_4 = 7$, $b_5 = 8$.

Prin executarea programului BASIC prezentat s-a obtinut solutia x având componentele $x = (-2.67 , 4.67 , 5.67 , -5.04 , 0.66)$.

Erorile ε_j, $1 \le j \le 5$, estimate dupa formula (7) sunt foarte mici confirmând indirect corectitudinea programului descris în Aplicatia 2. Astfel $\varepsilon = (0 , 0 , -9.536743E-07 , 0 , 0)$.

Observatia 1. La scrierea programului ce utilizeaza termenii a_{jk} a unei matrici triunghiulare A se poate economisi memorie calculator prin retinerea din matricea A numai a partii inferior sau superior triunghiulare (ce poate contine eventual elemente diferite de zero, restul elementelor fiind deja nule).

1.3. Matrici A simetrice

Vom studia modul de rezolvare pentru sistemul liniar $A\,x = b$ atunci când matricea A este simetrica, adica $a_{jk} = a_{kj}$ oricare ar fi $1 \le j, k \le n$. Pentru început vom presupune ca atât elementele matricei A si ci cele ale vectorului b pot lua valori numere complexe.

Transpusa $C^{(t)} = (c_{st}^{(t)})_{st}$ a unei matrici $C = (c_{jk})_{jk}$ cu m linii si n coloane este o matrice cu n linii si m coloane ce se obtine prin "inversarea" în matricea C a liniilor cu coloanele, adica $c_{kj}^{(t)} = c_{jk}$, $\forall j, k$, $1 \le j \le m$, $1 \le k \le n$. Se observa ca matricea C este simetrica daca si numai daca $C^{(t)} = C$.

In mod evident $(W^{(t)})^{(t)} = W$ oricare ar fi matricea W.

Propozitia 1. Pentru orice doua matrici $Y = (y_{ij})_{ij}$, $Z = (z_{jk})_{jk}$, $1 \le i \le n$, $1 \le j \le m$, $1 \le k \le p$, avem egalitatea $(Y Z)^{(t)} = Z^{(t)} Y^{(t)}$

Demonstratie. In adevar elementul w_{ki}, $1 \le k \le p$, $1 \le i \le n$, al matricei $Z^{(t)} Y^{(t)}$ este de forma

$$w_{ki} = z_{k1}^{(t)} y_{1i}^{(t)} + z_{k2}^{(t)} y_{2i}^{(t)} + z_{k3}^{(t)} y_{3i}^{(t)} + \dots + z_{km}^{(t)} y_{mi}^{(t)} =$$
$$= y_{i1} z_{1k} + y_{i2} z_{2k} + y_{i3} z_{3k} + \dots + y_{im} z_{mk} = v_{ik}$$

unde v_{ik} este componenta de pe linia i si coloana k din matricea produs $Y Z$ adica elementul de pe pozitia (k, i) din matricea $(Y Z)^{(t)}$ (transpusa matricei $Y Z$).

Propozitia 2. Daca A este o matrice simetrica de dimensiune n atunci exista o matrice superior triunghiulara C de dimensiune n, cu coeficienti numere complexe, astfel încât

$$A = C^{(t)} C \qquad (10)$$

Demonstratie. Tinând seama de forma triunghiulara a matricilor C si $C^{(t)}$, unde

$$C = \begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{1n} \\ 0 & c_{22} & c_{23} & c_{2n} \\ 0 & 0 & c_{33} & c_{3n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & c_{nn} \end{pmatrix} \qquad C^{(t)} = \begin{pmatrix} c_{11} & 0 & 0 & 0 \\ c_{12} & c_{22} & 0 & 0 \\ & & & \dots \\ c_{1,n-1} & c_{2,n-1} & c_{n-1,n-1} & 0 \\ c_{1n} & c_{2n} & c_{n-1,n} & c_{nn} \end{pmatrix}$$

din egalitatea $A = C^{(t)} C$ rezulta urmatoarele relatii ce sunt adevarate pentru orice $1 \le j \le k \le n$,

$$c_{1j} c_{1k} + c_{2j} c_{2k} + c_{3j} c_{3k} + \dots + c_{j-1,j} c_{j-1,k} + c_{jj} c_{jk} = a_{jk}$$

Printr-o rearanjare a acestor relatii în succesiunea

$$c_{11} = (a_{11})^{1/2} \qquad c_{1k} = a_{1k} / c_{11}, \quad 2 \le k \le n \qquad (11)$$
$$c_{jj} = (a_{jj} - c_{1j} c_{1j} - c_{2j} c_{2j} - \dots - c_{j-1,j} c_{j-1,j})^{1/2}, \quad 2 \le j \le n$$
$$c_{jk} = (a_{jk} - c_{1j} c_{1k} - c_{2j} c_{2k} - c_{3j} c_{3k} - \dots - c_{j-1,j} c_{j-1,k}) / c_{jj}, \quad 2 \le j < k \le n$$
$$c_{jk} = 0 \text{ pentru orice } 1 \le k < j \le n$$

deducem toti coeficientii c_{jk}, $1 \le j \le k \le n$, din partea superior triunghiulara a matricei C

Observatia 2. Reciproca Propozitiei 2 este de asemenea adevarata. In adevar orice matrice W de forma $W = Y^{(t)} Y$, unde Y este o matrice patratica oarecare (nu neaparat matrice triunghiulara) este o matrice simetrica, aceasta deoarece

$W^{(t)} = (Y^{(t)} Y)^{(t)} = Y^{(t)} (Y^{(t)})^{(t)} = Y^{(t)} Y = W$.

Propozitia 3. Daca A este o matrice simetrica atunci aflarea solutiei sistemului liniar $A x = b$ se reduce la rezolvarea a doua sisteme liniare triunghiulare.

Demonstratie. Folosind relatiile (11) construim matricea C astfel încât $A = C^{(t)} C$ (Propozitia 2). In acest context sistemul linear $A x = b$ se rescrie în forma $C^{(t)} C x = b$ solutia x rezultând din rezolvarea a doua sisteme lineare triunghiulare, anume : $C^{(t)} y = b$, $C x = y$

Observatia 3. Daca matricea simetrica A are elemente numere reale atunci nu este întotdeauna posibil de a se gasi o matrice superior triunghiulara C având numai elemente numere reale, astfel încât sa se verifice egalitatea $A = C^{(t)} C$.

Exemplul 3. Consideram matricea A de dimensiune 2 , unde $a_{11} = a_{22} = 1$, $a_{12} = a_{21} = a$. Din conditiile (11) rezulta : $c_{11} = 1$, $c_{12} = a$, $c_{21} = 0$, $c_{22}^2 = 1 - a^2$. In cazul în care $|a| > 1$ atunci elementul c_{22} nu poate avea o valoare numerica reala.

Imposibilitatea gasirii unei matrici triunghiulare C , de numere reale, astfel încât sa avem descompunerea $A = C^{(t)} C$, este pusa în evidenta prin obtinerea unor valori negative pentru expresiile a_{11} , $a_{jj} - c_{1j} c_{1j} - c_{2j} c_{2j} - \ldots - c_{j-1,j} c_{j-1,j}$. Daca c_{ij} sunt numere reale atunci expresiile mentionate ar trebui sa ia valori nenegative deoarece sunt echivalente cu c_{11}^2 , respectiv c_{jj}^2 , $2 \le j \le n$.

Aplicatia 3. Urmatorul program determina solutia numerica reala x a sistemului $A x = b$ precizând mai întâi vectorul y ca solutie a sistemului inferior triunghiular $C^{(t)} y = b$ iar apoi rezolvând sistemul $C x = y$ de forma superior triunghiulara. In situatia în care nu se poate gasi o solutie reala x programul se opreste cu mesaj de avertizare.

```
REM  Solutia sistemelor simetrice
DECLARE SUB s1tri (a!(), b!(), x!(), n!) :          DECLARE SUB s2tri (a!(), b!(), x!(), n!)
DECLARE SUB descsim (a!(), c!(), n!)
INPUT "n = "; n :          DIM a(n, n), c(n, n), x(n), y(n), b(n)
PRINT "Matricea  A  :"          ' Initializarea matricii simetrice  A
FOR j = 1 TO n :       FOR k = 1 TO j
PRINT "   a ("; j; ", "; k; ") = "; :          INPUT ; a(j, k) :          a(k, j) = a(j, k)
NEXT k :       NEXT j
PRINT :        PRINT "Vectorul termenilor liberi : "
FOR j = 1 TO n
PRINT "   b ("; j; ") = ";  :          INPUT ; b(j) :          NEXT j :          PRINT
CALL descsim(a(), c(), n)          ' Obtinerea descompunerii  C  a matricei  A
CALL s1tri(c(), b(), y(), n)          ' Rezolvarea sistemului  C' y = b
CALL s2tri(c(), y(), x(), n)          ' Rezolvarea sistemului  C x = y
PRINT "Sistemul simetric  A x = b  de "; n; " ecuatii are solutia :"          ' Listarea solutiei
FOR j = 1 TO n :       PRINT USING "X(##) = ###.##  ,  "; j; x(j); :       NEXT j
PRINT "Verificare solutiei gasite. Erori : "
FOR j = 1 TO n :       s = 0
FOR k = 1 TO n :       s = s + a(j, k) * x(k) :       NEXT k :
s = b(j) - s :       PRINT "Er"; j; " = "; s; "  ,  "; :       NEXT j

SUB descsim (a(), c(), n)          ' Descompunerea matricei simetrice  A  in forma  A = C' C
```

```
REM  Matricea  C  este superior triunghiulara
IF a(1, 1) < 0 THEN PRINT "Sistemul simetric nu are solutie reala"  STOP
c(1, 1) = SQR(a(1, 1))
FOR k = 2 TO n
IF a(1, 1) = 0 THEN PRINT "Sistemul simetric nu are solutie reala"  STOP
c(1, k) = a(1, k) / c(1, 1) :        NEXT k
FOR j = 2 TO n :        s = 0
FOR t = 1 TO j - 1 :        s = s + c(t, j) * c(t, j)        NEXT t
s = a(j, j) - s
IF s < 0 THEN PRINT "Sistemul simetric nu are solutie reala":  STOP
c(j, j) = SQR(s)
FOR k = j + 1 TO n :        s = 0
FOR t = 1 TO j - 1 :        s = s + c(t, j) * c(t, k) :        NEXT t
IF c(j, j) = 0 THEN PRINT "Sistemul simetric nu are solutie reala": STOP
c(j, k) = (a(j, k) - s) / c(j, j) :        NEXT k
NEXT j :        END SUB

SUB s1tri (c(), b(), x(), n)        ' Rezolvarea sistemului  C' x = b
REM  Matricea  C  este superior triunghiulara
IF c(1, 1) = 0 THEN PRINT "Sistemul triunghiular 1 nu admite solutie":        STOP
x(1) = b(1) / c(1, 1)
FOR j = 2 TO n
IF c(j, j) = 0 THEN PRINT "Sistemul triunghiular 1 nu admite solutie":  STOP
s = 0 :        FOR k = 1 TO j - 1 :        s = s + c(k, j) * x(k) :        NEXT k
x(j) = (b(j) - s) / c(j, j) :        NEXT j :        END SUB

SUB s2tri (c(), b(), x(), n)        ' Rezolvarea sistemului  C x = b
REM  Matricea  C  este memorata superior triunghiular
IF c(n, n) = 0 THEN PRINT "Sistemul triunghiular 2 nu admite solutie":  STOP
x(n) = b(n) / c(n, n)
FOR j = n - 1 TO 1 STEP -1
IF c(j, j) = 0 THEN PRINT "Sistemul triunghiular 2 nu admite solutie":  STOP
s = 0 :        FOR k = j + 1 TO n :        s = s + c(j, k) * x(k) :        NEXT k
x(j) = (b(j) - s) / c(j, j) :        NEXT j :        END SUB
```

Exemplul 4. Sa rezolvam sistemul liniar A x = b , unde A este o matrice simetrica de dimensiune n = 4 , cu : $a_{11} = 1$, $a_{21} = a_{12} = -2$, $a_{22} = 13$, $a_{31} = a_{13} = 3$, $a_{32} = a_{23} = -9$, $a_{33} = 19$, $a_{41} = a_{14} = 2$, $a_{42} = a_{24} = 8$, $a_{43} = a_{34} = -13$, $a_{44} = 49$, terrmenii liberi fiind $b_1 = 3$, $b_2 = -5$, $b_3 = 7$, $b_4 = -12$.

Rulând programul prezentat în Aplicatia 3 s-a obtinut solutia x = (50.95 , 4.35 , -9.40 , -5.53)

Eroarea totala ε rezultata în urma aplicarii acestei metode este foarte mica fapt ce confirma corectitudinea implementarii efectuate. Astfel componentele ε_j , $1 \le j \le 4$, ale erorii ε , calculate dupa formula (7), au luat valorile :

ε_1 = 9.536743E-07 , $\varepsilon_2 = 0$, ε_3 = 4.768372E-06 , ε_4 = -8.583069E-06

1.4. Metoda Gauss

Din sectiunile anterioare a reiesit ca sistemele liniare de tip (inferior sau superior) triunghiular pot fi cu usurinta solutionate. Ideea metodei lui Gauss consta în aplicarea unor transformari sistemului de ecuatii liniare (1) astfel încât sa se ajunga la o forma triunghiulara.

In primul rând vom preciza doua transformari ce nu afecteaza solutia sistemului (1). Astfel :

T1. Daca se înmulteste orice ecuatie j, $1 \le j \le n$, a sistemului (1) cu o cantitate c, $c \neq 0$, atunci solutia x a sistemului (1) nu se modifica.

T2. Solutia sistemului (1) se pastreaza daca la o linie j a acestui sistem se aduna orice alta linie t ce este înmultita cu o constanta arbitrara d (nu neaparat nenula).

Fie tablourile $A^{(1)} = (a_{jk}^{(1)})_{jk}$, $b^{(1)} = (b_j^{(1)})_j$, $A^{(2)} = (a_{jk}^{(2)})_{jk}$, $b^{(2)} = (b_j^{(2)})_j$, $1 \le j \le n$, $1 \le k \le n$, ale caror elemente sunt obtinute din tablourile A si b dupa formulele

$$
\begin{aligned}
&a_{jk}^{(1)} = c\, a_{jk}\ ,\ 1 \le k \le n\ ; \quad b_j^{(1)} = c\, b_j\ ; \qquad c \neq 0 \\
&a_{sk}^{(1)} = a_{sk}\ ,\ 1 \le k \le n\ ; \quad b_s^{(1)} = b_s\ ; \quad \forall\, s,\ 1 \le s \neq j \le n \\
&a_{tk}^{(2)} = a_{tk} + d\, a_{jk}\ ,\ 1 \le k \le n\ ; \quad b_t^{(2)} = b_t + d\, b_j\ ; \\
&a_{sk}^{(2)} = a_{sk}\ ,\ 1 \le k \le n\ ; \quad b_s^{(s)} = b_s\ ; \quad \forall\, s,\ 1 \le s \neq t \le n
\end{aligned}
\qquad (12)
$$

pentru j, t fixati $1 \le j, t \le n$, $t \neq j$.

Observatia 4. Prin aplicarea transformarilor de tip T1 sau T2 sistemului (1) rezulta sisteme de ecuatii liniare ce admit aceeasi solutie. Concret, pastrând notatiile din (12) solutiile sistemelor

$$A\,x = b \qquad A^{(1)}\,x^{(1)} = b^{(1)} \qquad A^{(2)}\,x^{(2)} = b^{(2)} \qquad (13)$$

verifica egalitatea $x^{(1)} = x^{(2)} = x$.

Notam prin G matricea extinsa obtinuta prin adaugarea la matricea A a unei ultime coloane determinata de vectorul coloana b al termenilor liberi, adica $G \equiv (A\ b)$, $G = (g_{jk})_{jk}$, $1 \le j \le n$, $1 \le k \le n+1$; $g_{ju} = a_{ju}$, $1 \le u \le n$; $g_{j,\,n+1} = b_j$.

Aceasta notatie conduce la simplificarea expunerii, aplicarea transformarilor T1 si T2 desfasurându-se în mod unitar, numai asupra matricei G. In acest mod se evitata tratarea diferentiata a tablourilor A si b.

Initial matricea G este de forma ($A\ b$). In cazul în care $g_{11} \neq 0$, prin aplicarea transformarii T1 în matricea $G \equiv G^{(0)}$ (împartirea primei linii cu g_{11}), rezulta o noua matrice $G^{(1)}$ în care $g_{11}^{(1)} = 1$. Din matricea $G^{(1)}$ prin transformari succesive de tipul T2 (la linia j a matricei $G^{(j-1)}$ se aduna linia 1 multiplicata cu $-\,a_{j1}^{(j-1)}$, $2 \le j \le n$) se obtin pe rând matricile $G^{(2)}, G^{(3)}, \ldots, G^{(n)}$ în care elementele $g_{21}^{(2)}, g_{31}^{(3)}, \ldots, g_{n1}^{(n)}$ devin pe rând nule. In acest mod, dupa n pasi, în matricea $G^{(n)}$ rezultata avem $g_{j1}^{(n)} = 0$, $2 \le j \le n$ si $g_{11}^{(n)} = 1$, adica

$$
G^{(1)} = \begin{pmatrix}
1 & g_{12}^{(1)} & g_{13}^{(1)} & \cdots & g_{1n}^{(1)} & g_{1,n+1}^{(1)} \\
g_{21} & g_{22} & g_{23} & \cdots & g_{2n} & g_{2,n+1} \\
g_{31} & g_{32} & g_{33} & \cdots & g_{3n} & g_{3,n+1} \\
\cdots & & & & & \\
g_{n1} & g_{n2} & g_{n3} & \cdots & g_{nn} & g_{n,n+1}
\end{pmatrix}
\qquad
G^{(2)} = \begin{pmatrix}
1 & g_{12}^{(1)} & g_{13}^{(1)} & \cdots & g_{1n}^{(1)} & g_{1,n+1}^{(1)} \\
0 & g_{22}^{(2)} & g_{23}^{(2)} & \cdots & g_{2n}^{(2)} & g_{2,n+1}^{(2)} \\
g_{31} & g_{32} & g_{33} & \cdots & g_{3n} & g_{3,n+1} \\
\cdots & & & & & \\
g_{n1} & g_{n2} & g_{n3} & \cdots & g_{nn} & g_{n,n+1}
\end{pmatrix}
$$

$$G^{(3)} = \begin{pmatrix} 1 & g_{12}^{(1)} & g_{13}^{(1)} & g_{1n}^{(1)} & g_{1,n+1}^{(1)} \\ 0 & g_{22}^{(2)} & g_{23}^{(2)} & g_{2n}^{(2)} & g_{2,n+1}^{(2)} \\ 0 & g_{32}^{(3)} & g_{33}^{(3)} & g_{3n}^{(3)} & g_{3,n+1}^{(3)} \\ \dots & & & & \\ g_{n1} & g_{n2} & g_{n3} & g_{nn} & g_{n,n+1} \end{pmatrix} \qquad G^{(n)} = \begin{pmatrix} 1 & g_{12}^{(1)} & g_{13}^{(1)} & g_{1n}^{(1)} & g_{1,n+1}^{(1)} \\ 0 & g_{22}^{(2)} & g_{23}^{(2)} & g_{2n}^{(2)} & g_{2,n+1}^{(2)} \\ 0 & g_{32}^{(3)} & g_{33}^{(3)} & g_{3n}^{(3)} & g_{3,n+1}^{(3)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & g_{n2}^{(n)} & g_{n3}^{(n)} & g_{nn}^{(n)} & g_{n,n+1}^{(n)} \end{pmatrix}$$

Aceeasi procedura de folosire a transformarii T1 si apoi utilizarea repetata, de n-2 ori a transformarii T2 , va fi aplicata si pentru a doua coloana a matricei $G^{(n)}$ ce a fost dedusa anterior Astfel pornind cu elementul $g_{22}^{(n)}$ se obtin pas cu pas matricile $G^{(n+1)}$, $G^{(n+2)}$, ... , $G^{(2n-1)}$. In final $g_{j1}^{(2n-1)} = g_{t2}^{(2n-1)} = 0$, $2 \le j \le n$, $3 \le t \le n$, iar $g_{11}^{(2n-1)} = g_{22}^{(2n-1)} = 1$,

$$G^{(2n-1)} = \begin{pmatrix} 1 & g_{12}^{(2n-1)} & g_{13}^{(2n-1)} & g_{1n}^{(2n-1)} & g_{1,n+1}^{(2n-1)} \\ 0 & 1 & g_{23}^{(2n-1)} & g_{2n}^{(2n-1)} & g_{2,n+1}^{(2n-1)} \\ 0 & 0 & g_{33}^{(2n-1)} & g_{3n}^{(2n-1)} & g_{3,n+1}^{(2n-1)} \\ \dots & & \dots & \dots & \dots \\ 0 & 0 & g_{n3}^{(2n-1)} & g_{nn}^{(2n-1)} & g_{n,n+1}^{(2n-1)} \end{pmatrix}$$

Utilizarea procedurii prezentate conduce dupa $n(n+1)/2$ pasi la deterrminarea matricei $G^{(n(n+1)/2)}$ ce are o forma triunghiulara

$$G^{(n(n+1)/2)} = \begin{pmatrix} 1 & g_{12}^{(n(n+1)/2)} & g_{13}^{(n(n+1)/2)} & g_{1n}^{(n(n+1)/2} & g_{1,n+1}^{(n(n+1)/2)} \\ 0 & 1 & g_{23}^{(n(n+1)/2)} & g_{2n}^{(n(n+1)/2)} & g_{2,n+1}^{(n(n+1)/2)} \\ 0 & 0 & 1 & g_{3n}^{(n(n-1)/2)} & g_{3,n+1}^{(n(n+1)/2)} \\ \dots & & \dots & \dots & \dots \\ 0 & 0 & 0 & 1 & g_{n,n+1}^{(n(n+1)/2)} \end{pmatrix} \qquad (14)$$

Tinând seama de Observatia 4 rezulta ca sistemele $A\,x = b$ si $G^*\,x^* = b^*$ admit aceeasi solutie, $x = x^*$. Matricea Matricea G^* si vectorul b^* reprezinta primele n coloane, respectiv a n+1 coloana a matricii $G^{(n(n+1)/2)}$ definita de (14), adica $G^{(n(n+1)/2)} = (G^* , b^*)$.

In concluzie solutia x a sistemului liniar $A\,x = b$ este data de relatiile

$$x_n = g_{n,n+1}^{(p)}$$

$$x_j = g_{j,\,n+1}^{(p)} - g_{j,\,j+1}^{(p)} x_{j+1} - g_{j,\,j+2}^{(p)} x_{j+2} - \dots - g_{j,\,n-1}^{(p)} x_{n-1} - g_{j,\,n}^{(p)} x_n \qquad (15)$$

unde $j = n-1, n-2, \dots, 2, 1$ iar $p = n(n+1)/2$.

Remarcam faptul ca prin aplicarea transformarilor T1 si T2 mentinând o parcurgere a matricei initiale G în ordinea "subdiagonala" $g_{11}, \dots, g_{1n}, g_{22}, \dots, g_{2n}, g_{33}, \dots, g_{3n}, \dots, g_{n-1,\,n-1}, g_{n-1,\,n}, g_{nn}$ nu vor fi afectate elementele g_{jk} , $k \le j \le n$, ce au fost deja facute 1 sau 0 .

Observatia 5. S-ar putea ca la un pas j , $1 \le j \le n$, elementul g_{jj} sa fie nul fiind astfel imposibila utilizarea transformarilor T1 si T2 (pentru a avea în final $g_{jj}^{(p)} = 1$ si $g_{jt}^{(p)} = 0$, $j < t \le n$). Intr-o asemenea situatie ar trebui cautat pe linia j un element "pivot" a_{js} nenul , $j+1 \le s \le n$ In cazul nerealizarii acestui deziderat determinantul sistemului liniar $A\,x = b$ va fi în mod evident nul, $\det(A) = 0$, matricea sistemului transformat prezentând o linie cu toate elementele egale cu zero Se ajunge asadar la imposibilitatea gasirii unei solutii x .

In urma optarii pentru un pivot nenul g_{js}, $s > j$, vor trebui inversate coloanele j si s astfel încât pivotul ales sa fie transferat pe pozitia (j, j). Permutarea coloanelor j si s în matricea G nu va afecta valorile componentelor solutiei x. In schimb componenta x_s a solutiei se va regasi în vectorul x pe pozitia j si nu pe pozitia s.

Observatia 6. In cazul alegerii unor pivoti ce nu sunt situati pe diagonala principala a matricei A va fi necesara retinerea tuturor permutarilor de coloane aplicate matricilor de tip G. In acest mod pot fi regasite în final pozitiile corecte ale componentelor soluției x.

In acest context se va introduce în matricea G o linie suplimentara cu elementele $g_{n+1,k} = k$, $1 \le k \le n$. O permutare a coloanelor j si s a matricei G dictata de alegerea pivotului g_{js}, se va aplica si liniei n+1 nou introduse.

Daca în final, valoarea elementului $g_{n+1,k}^{(p)}$ din matricea $G^{(p)}$, $p = n(n+1)/2$, este egala cu j, iar y este solutia dedusa din matricea triunghiulara $G^{(p)}$, $G^{(p)} = (G^*, b^*)$, $G^* y = b^*$, atunci valoarea componentei x_j a solutiei sistemului liniar $A x = b$ este y_k si nu y_j.

Observatia 7. Practic, la aplicarea metodei de reducere Gauss, în vederea micsorarii erorilor de calcul, se prefera ca pentru fiecare linie j, $1 \le j \le n$, a matricei curente $G^{(\#)}$ sa fie ales drept pivot cel mai mare element în valoare absoluta $g_{js}^{(\#)}$. Va fi astfel necesara permutarea în matricea $G^{(\#)}$ a coloanelor j si s.

Aplicatia 4. Subrutina **sgauss** determina, prin metoda lui Gauss descrisa anterior, solutia x a sistemului liniar $A x = b$. De fiecare data pe o linie j, $1 \le j \le n$, a transformatei matricei A se alege pivotul ce are valoarea absoluta cea mai mare :

```
REM  Rezolvarea sistemelor liniare prin metoda Gauss
DECLARE SUB sgauss (a!(), b!(), x!(), n!)
INPUT "n = "; n :              DIM a(n, n), x(n), b(n)
PRINT "Matricea A :"           ' Initializarea matricei A
FOR j = 1 TO n
FOR k = 1 TO n :      PRINT USING "   a(##;##) = "; j; k;  :      INPUT ; a(j, k)
NEXT k :      NEXT j :          PRINT
PRINT "Vectorul termenilor liberi : "
FOR j = 1 TO n :     PRINT "   b ("; j; ") = "; :      INPUT ; b(j) :      NEXT j :      PRINT
CALL sgauss(a(), b(), x(), n)      ' Rezolvarea sistemului  A x = b  prin metoda Gauss
PRINT "Sistemul  A x = b  de "; n; " ecuatii are solutia :"          ' Listarea solutiei
FOR j = 1 TO n :       PRINT USING "X(##) = ###.##  ,  "; j; x(j); :       NEXT j
PRINT "Verificare solutiei gasite. Erori : "
FOR j = 1 TO n :          s = 0
FOR k = 1 TO n :          s = s + a(j, k) * x(k) :          NEXT k
s = b(j) - s :          PRINT "Er"; j; " = "; s; "  ,  "; :          NEXT j

SUB sgauss (a(), b(), x(), n)          ' Rezolvarea sistemului  A x = b  folosind eliminarea Gauss
REM  Continutul matricei  A  este transferat in matricea  G  ce va fi in final modificata
REM  Linia  g(n+1,..)  retine permutarea elementelor solutiei prin aplicarea procedurii Gauss
REM  Initializarea matricei de lucru  g
DIM g(n + 1, n + 1), y(n)          ' Tablouri auxiliare
FOR j = 1 TO n
FOR k = 1 TO n          g(j, k) = a(j, k) :          NEXT k
```

```
g(n + 1, j) = j :          g(j, n + 1) = b(j) :          NEXT j
REM  Aducerea matricei g  la forma triunghiulara Gauss
FOR j = 1 TO n - 1          ' Determinarea pozitiei (j,s) cu cel mai mare pivot in modul
s = j :          gm = ABS(g(j, j))
FOR k = j + 1 TO n
IF ABS(g(j, k)) > gm THEN gm = g(j, k) :          s = k
NEXT k
REM  Permutarea coloanei j cu coloana s
IF j <> s THEN FOR t = 1 TO n + 1 :          SWAP g(t, j), g(t, s) :          NEXT t
IF g(j, j) = 0 THEN PRINT "Determinantul sistemului este nul" :          STOP
REM  Anularea elementelor g(t,j)  de sub diagonala principala
FOR t = j + 1 TO n :          c = -g(t, j) / g(j, j) :          g(t, j) = 0
FOR k = j + 1 TO n + 1 :          g(t, k) = g(t, k) + c * g(j, k) :          NEXT k
NEXT t :          NEXT j
REM  Rezolvarea sistemului superior triunghiular obtinut
IF g(n, n) = 0 THEN PRINT "Determinantul sistemului este nul" :          STOP
y(n) = g(n, n + 1) / g(n, n)
FOR t = n - 1 TO 1 STEP -1
s = 0 :          FOR k = t + 1 TO n :          s = s + g(t, k) * y(k) :          NEXT k
y(t) = (g(t, n + 1) - s) / g(t, t) :          NEXT t
REM  Permutarea elementelor solutiei
FOR k = 1 TO n :          x(g(n + 1, k)) = y(k) :          NEXT k :          END SUB
```

Exemplul 5. Vom rula procedura **sgauss** prezentata în Aplicatia 4 considerând datele de intrare $n = 4$ si $a_{11} = -1$, $a_{12} = 2$, $a_{13} = 3$, $a_{14} = -5$, $a_{21} = .2$, $a_{22} = -3$, $a_{23} = 5$, $a_{24} = 4$, $a_{31} = 9$, $a_{32} = 7$, $a_{33} = 8$, $a_{34} = 1$, $a_{41} = 3$, $a_{42} = 6$, $a_{43} = 5$, $a_{44} = -8$; $b_1 = 11$, $b_2 = 23$, $b_3 = 45$, $b_4 = 36$.

In urma executarii procedurii **sgauss** a rezultat solutia $x = (6.39 , -4.11 , 2.48 , -3.63)$.

In vederea verificarii corectitudinii algoritmului implementat s-au evaluat dupa formula (7) erorile ε_j, $1 \le j \le 4$, obtinându-se efectiv $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = \varepsilon_4 = 0$.

1.5. Matrici A de tip banda

Un caz particular interesant, ce apare des în aplicatiile practice, este acela în care toate elementele matricei A ce nu apartin unei "bande diagonale" sunt nule. In cazul sistemelor liniare tridiagonale avem $a_{js} = 0$ daca $s < j-1$ sau $s > j+1$, $1 \le s, j \le n$, adica

$$A = \begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 & 0 & & 0 & 0 & 0 & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 & 0 & & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & \dots & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & a_{n-1,n-2} & a_{n-1,n-1} & a_{n-1,n} \\ 0 & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & 0 & a_{n,n-1} & a_{nn} \end{pmatrix} \qquad (16)$$

Observatia 8. Intr-o asemenea situatie, pentru rezolvarea sistemului liniar $A x = b$, poate fi

aplicata o varianta simplificata a procedurii de reducere Gauss. Astfel vor trebui anulate pe fiecare coloana, prin aplicarea transformarilor de tip T1 si T2 (sectiunea 1.4), a cel mult doua elemente de sub diagonala principala, restul elementelor fiind deja nule. In plus, la alegerea pivotului având valoarea absoluta maxima pe linia i , $1 \le i \le n$,vor participa cel mult trei elemente ale transformatei matricei A, restul elementelor de pe linia i fiind nule.

In continuare vom sugera o procedura directa de determinare a solutiei $x = (x_1, x_2, \dots, x_n)$ pentru sistemul liniar $A x = b$ de n ecuatii, unde matricea patratica A este de tip banda având forma tridiagonala (16).

Daca x este vectorul solutie cautat, fie coeficientii c_j , d_j , $1 \le j \le n-1$, alesi astfel încât sa avem relatiile

$$x_j = c_j x_{j+1} + d_j \qquad (17)$$

Din prima ecuatie a sistemului liniar $A x = b$ rezulta $x_1 = -(a_{12} / a_{11}) x_2 + b_1 / a_{11}$. Deci

$$c_1 = - a_{12} / a_{11} \qquad d_1 = b_1 / a_{11} \qquad (18)$$

Pentru $2 \le j \le n - 1$, înlocuind în a j-a ecuatie a sistemului $A x = b$ pe x_{j-1} cu expresia data de formula (17) obtinem

$$a_{j,j-1} (c_{j-1} x_j + d_{j-1}) + a_{j,j} x_j + a_{j,j+1} x_{j+1} = b_j$$

adica

$$x_j = - \frac{a_{j,j+1}}{a_{j,j-1} c_{j-1} + a_{jj}} x_{j+1} + \frac{b_j - a_{j,j-1} d_{j-1}}{a_{j,j-1} c_{j-1} + a_{jj}}$$

Comparând aceasta ultima egalitate cu formula (17) deducem

$$c_j = - \frac{a_{j,j+1}}{a_{j,j-1} c_{j-1} + a_{jj}} \qquad d_j = \frac{b_j - a_{j,j-1} d_{j-1}}{a_{j,j-1} c_{j-1} + a_{jj}} \qquad (19)$$

Aplicarea succesiva a relatiei (19) cu conditia initiala (18) permite precizarea tuturor valorilor coeficientilor c_1 , d_1 , c_2 , d_2 , ... , c_{n-1} , d_{n-1} .

Prin utilizarea formulei $x_{n-1} = c_{n-1} x_n + d_{n-1}$ în ultima ecuatie a sistemului liniar $A x = b$ de forma (16) rezulta $a_{n,n-1} (c_{n-1} x_n + d_{n-1}) + a_{nn} x_n = b_n$. Asadar ultima componenta x_n a solutiei x pentru sistemul $A x = b$ este data de expresia

$$x_n = \frac{b_n - a_{n,n-1} d_{n-1}}{a_{n,n} + a_{n,n-1} c_{n-1}} \qquad (20)$$

Pornind în sens invers, cu valoarea x_n precizata de (20), dupa folosirea iterativa a formulei (17) pentru j = n-1, n-2, ..., 3, 2, 1, unde coeficientii c_j , d_j au fost în prealabil dedusi din (19), aflam succesiv si valorile componentelor x_{n-1} , x_{n-2} ,..., x_3 , x_2 , x_1 ale solutiei sistemului liniar tridiagonal $A x = b$.

Aplicatia 5. Urmatorul program BASIC determina solutia oricarui sistem liniar tridiagonal utilizând procedura descrisa anterior. In acest program variabilele a(j,1) , a(j,2) , a(j,3) sunt asociate respectiv elementelor $a_{j,j-1}$, a_{jj} , $a_{j,j+1}$ din matricea A .

```
REM  Rezolvarea sistemelor tridiagonale  A x = b
INPUT "Dimensiunea sistemului = "; n :          DIM a(n, 3), b(n), x(n), c(n - 1), d(n - 1)
REM  Initializarea matricei A si a vectorului b ( termenii liberi )
a(1, 2) = -3 :          a(1, 3) = 1 :          b(1) = -1
FOR j = 2 TO n - 1 :          a(j, 1) = 2 * j + 1 :          a(j, 2) = -3 * j
```

```
a(j, 3) = 2 * j - 1 :          b(j) = j * j - 2 :          NEXT j
a(n, 1) = 2 * n + 1 :          a(n, 2) = -3 * n :          b(n) = -n * n - n - 1
c(1) = -a(1, 3) / a(1, 2) :          d(1) = b(1) / a(1, 2)        ' Calcularea coeficientilor  c(j) , d(j)
FOR j = 2 TO n - 1 :          t = a(j, 1) * c(j - 1) + a(j, 2)
c(j) = -a(j, 3) / t :          d(j) = (b(j) - a(j, 1) * d(j - 1)) / t :          NEXT j
x(n) = (b(n) - a(n, 1) * d(n - 1)) / (a(n, 2) + a(n, 1) * c(n - 1))        ' Obtinerea solutiei
FOR j = n - 1 TO 1 STEP -1 :          x(j) = c(j) * x(j + 1) + d(j) :          NEXT j
PRINT "Solutia sistemului liniar  A x = b"
FOR j = 1 TO n :          PRINT USING "###.## ; "; x(j); :          NEXT j
```

Exemplul 6. Fie A o matrice tridiagonala unde $a_{j,j-1} = 2j + 1$, $a_{j,j} = -3j$, $a_{j,j+1} = 2j - 1$, $1 \le j \le n$, cu conventia $a_{jk} = 0$ daca $|j - k| \ge 2$. Termenii liberi vor lua valorile : $b_1 = -1$, $b_n = -n^2 - n - 1$, $b_j = j^2 - 2$ pentru $2 \le j \le n-1$. Prin calcul direct se deduce ca $x = (1, 2, 3, \ldots, n-1, n)$ este solutia sistemului liniar tridiagonal $A x = b$.

In urma rularii programului prezentat în Aplicatia 5, pentru diferite dimensiuni n , $3 \le n \le 15$, ale sistemului liniar tridiagonal $A x = b$, s-a obtinut experimental solutia $x = (1, 2, 3, \ldots, n-1, n)$.

1.6. Rezolvarea "în bloc" a sistemelor liniare

Avantajul metodei Gauss consta în facilitatea solutionarii simultane a mai multor sisteme de ecuatii liniare ce au însa aceeasi matrice A dar alti termeni liberi b .

Astfel rezolvarea simultana a m sisteme liniare de forma $A x^{(u)} = b^{(u)}$, cu $b^{(u)}$ vector coloana de componente $b^{(u)} \equiv (b_1^{(u)}, b_2^{(u)}, b_3^{(u)}, \ldots, b_n^{(u)})$, $1 \le u \le m$, revine la a initializa matricea Gauss G de n+1 linii si n+m coloane cu valorile :

$$G = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{1n} & b_1^{(1)} & b_1^{(2)} & b_1^{(m)} \\ a_{21} & a_{22} & a_{23} & a_{2n} & b_2^{(1)} & b_2^{(2)} & b_2^{(m)} \\ a_{31} & a_{32} & 33 & a_{3n} & b_3^{(1)} & b_3^{(2)} & b_3^{(m)} \\ & & & & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & a_{n3} & a_{nn} & b_n^{(1)} & b_n^{(2)} & b_n^{(m)} \\ 1 & 2 & 3 & n & & & \end{pmatrix} \qquad (21)$$

In final se va obtine matricea $G^{(*)}$ atasata unui numar de m sisteme triunghiulare cu termenii liberi dati de vectorii $b^{(*u)} = (b_1^{(*u)}, b_2^{(*u)}, b_3^{(*u)}, \ldots, b_n^{(*u)})$, $1 \le u \le m$,

$$G^{(*)} = \begin{pmatrix} g_{11}^{(*)} & g_{12}^{(*)} & g_{13}^{(*)} & g_{1n}^{(*)} & b_1^{(*1)} & b_1^{(*2)} & b_1^{(*m)} \\ 0 & g_{22}^{(*)} & g_{23}^{(*)} & g_{2n}^{(*)} & b_2^{(*1)} & b_2^{(*2)} & b_2^{(*m)} \\ 0 & 0 & g_{33}^{(*)} & g_{3n}^{(*)} & b_3^{(*1)} & b_3^{(*2)} & b_3^{(*m)} \\ & & & & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & g_{nn}^{(*)} & b_n^{(*1)} & b_n^{(*2)} & b_n^{(*m)} \\ g_{n+1,1}^{(*)} & g_{n+1,2}^{(*)} & g_{n+1,3}^{(*)} & g_{n+1,n}^{(*)} & & & \end{pmatrix} \qquad (22)$$

Toate sistemele triunghiulare bazate pe matricea (22) vor fi rezolvate în mod direct (a se

vedea sectiunea 1.2.) . Se vor obtine astfel solutiile $y^{(u)}$, $1 \le u \le m$. Componentele solutiei $x^{(u)}$ ale unui sistem liniar $A\,x^{(u)} = b^{(u)}$, $1 \le u \le m$, se regasesc în elementele vectorului $y^{(u)}$ dar pe pozitii eventual diferite (Observatia 5). Conform Observatiei 6 indicii acestor componente au suferit permutarea ($g_{n+1,1}^{(*)}$, $g_{n+1,2}^{(*)}$, $g_{n+1,3}^{(*)}$, ... , $g_{n+1,n}^{(*)}$) memorata în cea de a n+1 linie a matricei $G^{(*)}$

Observatia 9. Prin procedeul propus pentru rezolvarea în bloc a unui set de m sisteme liniare având toate aceeasi matrice A, matricea comuna A nu va fi adusa la forma triunghiulara pentru fiecare sistem liniar în parte. Operatiunea de transformare triunghiulara a matricei initiale A se va desfasura o singura data, actiunea fiind extinsa si asupra tuturor termenilor liberi asociati celor m sisteme liniare considerate (a se vedea forma (22) a matricei Gauss $G^{(*)}$).

Observatia 10. In cadrul acestui capitol nu vom discuta o clasa întreaga de proceduri iterative ce ar putea fi utilizate la rezolvarea sistemului liniar $A\,x = b$. Am evitat abordarea acestui subiect datorita unor analize relativ complexe privind initializarea procesului iterativ. In plus timpul de rulare a unei proceduri iterative este de regula substantial mai mare decât în cazul metodelor deja discutate.

Remarcam de asemenea posibilitatea folosirii unor tehnici probabiliste de tip Monte Carlo pentru precizarea unui numar restrâns de componente x_j ale solutiei x sau pentru determinarea valorii unei combinatii liniare date a componentelor x_j , $1 \le j \le n$, unde $A\,x = b$.

2. Inversarea matricilor

Aflarea inversei $C \equiv (c_{jk})_{jk}$ a unei matrici $A \equiv (a_{jk})_{jk}$, $1 \le j, k \le n$, se reduce la rezolvarea a n sisteme liniare de forma $A\,c^{(k)} = d^{(k)}$, $1 \le k \le n$. Solutia $c^{(k)}$ reprezinta tocmai coloana k a matricei inverse C , $c^{(k)} \equiv (c_{1k}, c_{2k}, \ldots, c_{nk})$, unde vectorul coloana $d^{(k)}$ al termenilor liberi are elementele $d_j^{(k)}$ nule cu exceptia elementului $d_k^{(k)}$, cu $d_k^{(k)} = 1$.

Prin urmare matricea Gauss G va fi initializata astfel încât sa fie posibila solutionarea simultana a n sisteme de ecuatii liniare ale caror solutii vor defini chiar coloanele inversei C a matricei A. Luând în considerare aspectele discutate în sectiunea 1.6 vom aplica transformarile T1 si T2 (sectiunea 1.4) matricei G pentru aducerea ei la o forma triunghiulara,

$$G = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{1n} & 1 & 0 & 0 & \ldots & 0 \\ a_{21} & a_{22} & a_{23} & a_{2n} & 0 & 1 & 0 & \ldots & 0 \\ a_{31} & a_{32} & 33 & a_{3n} & 0 & 0 & 1 & \ldots & 0 \\ & & & & & & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & a_{n3} & a_{nn} & 0 & 0 & 0 & & 1 \\ 1 & 2 & 3 & n & & & & & \end{pmatrix} \qquad (23)$$

Cea de a n+1 linie a matricei G având forma initiala (23) este utilizata în vederea memorarii permutarilor elementelor solutiilor gasite (Observatia 6).

Aplicatia 6. Prezentam în continuare subprogramul **invmat** de calcul al inversei C a matricei A . Programul **invmat** foloseste procedeul solutionarii "în bloc" a n sisteme liniare (sectiunea 1.6), coeficientii sistemelor fiind retinuti în matricea G de forma (23).

```
SUB invmat (a(), c(), n)          ' Determinarea inversei C a matricei patratice A
```

```
REM  Continutul matricei  A  este transferat in matricea  G  ce va fi in final modificata
REM  Linia  g(n+1,.)  retine permutarea elementelor solutiei prin aplicarea procedurii Gauss
REM  Initializarea matricei de lucru  g  ce va contine  A , matricea identitate, permutarea solutiei
DIM g(n + 1, 2 * n), y(n)        ' Tablouri auxiliare
FOR j = 1 TO n
FOR k = 1 TO n :        g(j, k) = a(j, k) :        g(j, n + k) = 0! :        NEXT k
g(n + 1, j) = j :        g(j, n + j) = 1! :        NEXT j
REM  Aducerea matricei  g  la forma triunghiulara Gauss
FOR j = 1 TO n - 1
s = j :        gm = ABS(g(j, j))        ' Determinarea pozitiei  (j,s)  cu cel mai mare pivot in modul
FOR k = j + 1 TO n
IF ABS(g(j, k)) > gm THEN gm = g(j, k) :        s = k
NEXT k
REM  Permutarea coloanei  j  cu coloana  s
IF  j <> s  THEN FOR  t = 1  TO   n + 1 :        SWAP g(t, j), g(t, s) :        NEXT t
IF g(j, j) = 0 THEN PRINT "Determinantul sistemului este nul" :        STOP
REM  Anularea elementelor  g(t,j)  de sub diagonala principala
FOR t = j + 1 TO n :        c = -g(t, j) / g(j, j) :        g(t, j) = 0
FOR k = j + 1 TO 2 * n :        g(t, k) = g(t, k) + c * g(j, k) :        NEXT k :        NEXT t
NEXT j
IF g(n, n) = 0 THEN PRINT "Determinantul sistemului este nul" :        STOP
REM  Obtinerea coloanelor matricei inverse
FOR nc = n + 1 TO 2 * n :        y(n) = g(n, nc) / g(n, n)
FOR t = n - 1 TO 1 STEP -1 :        s = 0
FOR k = t + 1 TO n :        s = s + g(t, k) * y(k) :        NEXT k
y(t) = (g(t, nc) - s) / g(t, t) :        NEXT t
REM  Permutarea elementelor solutiei si stocarea in coloana  nc
FOR k = 1 TO n :        c(g(n + 1, k), nc - n) = y(k) :        NEXT k
NEXT nc :        END SUB
```

La testarea programului s-a folosit aceeasi matrice A ca si în cazul sistemului liniar solutionat anterior prin metoda Gauss (Exemplul 5). Validarea faptului ca matricea C rezultata este inversa matricei A poate fi realizata experimental prin verificarea relatiilor matriciale $A\,C = C\,A = I$, unde I este matricea unitate de dimensiune n .

Aplicatia 7. Obtinerea inversei C a matricei A ar putea fi utilizata si la rezolvarea sistemului liniar $A\,x = b$.

In adevar, înmultind la stânga cu matricea C, relatia matriciala $A\,x = b$ devine $C\,A\,x = C\,b$. Folosind identitatea $C\,A = A\,C = I$ deducem în final solutia x a sistemului $A\,x = b$. Mai precis $x = C\,b$, matricea inversa C fiind obtinuta în Aplicatia 6 .

Listam programul de solutionare a sistemului liniar $A\,x = b$ ce utilizeaza procedura **invmat** de obtinere a inversei C a matricei A .

```
DECLARE SUB invmat (a!(), c!(), n!)        ' Inversarea matricilor prin metoda Gauss
CLS        PRINT "Determinarea inversei  C  a matricei  A"
INPUT "n = "; n :        DIM a(n, n), c(n, n)
PRINT "Matricea  A  "        ' Initializarea matricei  A
FOR j = 1 TO n        FOR k = 1 TO n
```

```
PRINT USING "  a(##,##) = "; j; k; :        INPUT ; a(j, k) :       NEXT k :        NEXT j
CALL invmat(a(), c(), n)              ' Obtinerea inversei C prin metoda Gauss
PRINT "Matricea inversa C :"
FOR j = 1 TO n :     FOR k = 1 TO n :     PRINT USING "#####.##  , "; c(j, k); :     NEXT k
PRINT :       NEXT j
PRINT :    PRINT "Verificare solutiei gasite. Matricea  A * C  : "
FOR j = 1 TO n :          FOR k = 1 TO n
s = 0 :       FOR t = 1 TO n :       s = s + a(j, t) * c(t, k) :       NEXT t
PRINT USING "#####.##  , "; s; :          NEXT k :          PRINT :          NEXT j
PRINT "Rezolvarea sistemului liniar  A x = b  utilizand inversa  C  a matricei A"
DIM b(n), x(n) :        PRINT "Vectorul termenilor liberi : "
FOR j = 1 TO n :       PRINT "   b ("; j; ") = "; :       INPUT ; b(j) :       NEXT j :       PRINT
PRINT "Sistemul  A x = b  de "; n; " ecuatii are solutia :"
FOR j = 1 TO n :     x(j) = 0 :     FOR k = 1 TO n :     x(j) = x(j) + c(j, k) * b(k) :     NEXT k
PRINT USING "X(##) = ###.##  , "; j; x(j); :          NEXT j
```

Exemplul 7. Vom rezolva sistemul liniar $A\,x = b$ utilizând inversa C a matricei A.

Testarea metodei prezentate în Aplicatia 7 s-a realizat pentru $n = 4$, matricea A si vectorul fiind definiti în Exemplul 5. Rulând programul **invmat** s-a obtinut inversa C a matricei A, cu : $c_{11} = -0.62$, $c_{12} = 0.15$, $c_{13} = -0.14$, $c_{14} = 0.45$, $c_{21} = 0.43$, $c_{22} = -0.25$, $c_{23} = 0.23$, $c_{24} = -0.37$, $c_{31} = 0.29$, $c_{32} = 0.06$, $c_{33} = 0.07$, $c_{34} = -0.14$, $c_{41} = 0.27$, $c_{42} = -0.09$, $c_{43} = 0.16$, $c_{44} = -0.32$.

Ca o verificare a determinarii corecte a matricii inverse C s-au efectuat produsele de matrici $A\,C$ si $C\,A$ rezultatele fiind matricea identitate I.

Solutia sistemului liniar $A\,x = b$ este de forma $x = C\,b$. Pentru $b = (11, 23, 45, 36)$ s-a obtinut solutia $x = (6.39, -4.11, 2.48, -3.63)$. Aceeasi solutie a fost dedusa si prin aplicarea metodei Gauss de rezolvare a sistemelor liniare (Exemplul 5).

3. Calculul determinantilor

Intentionam sa calculam valoarea determinantului unei matrici patratice $A = (a_{jk})_{jk}$, $1 \le j, k \le n$, de dimensiune n.

Este cunoscuta definitia determinantului $\det(A)$ a matricei A,

$$\det(A) = \sum_{\sigma \in S} (-1)^{\mathrm{sgn}(\sigma)} a_{1,\sigma(1)} a_{2,\sigma(2)} a_{3,\sigma(3)} \cdots a_{n,\sigma(n)} \qquad (24)$$

unde S este multimea tuturor permutarilor elementelor $1, 2, 3, \ldots, n$, σ fiind o permutare arbitrara. Expresia $(-1)^{sgn(\sigma)}$ va lua valoarea 1 sau -1 dupa cum permutarea σ este para, respectiv impara.

Calculul determinantului $\det(A)$ bazat pe formula (24) este deosebit de anevoios. In acest sens precizam si numarul imens de operatii numerice rezultat din parcurgerea tuturor permutarilor σ de n obiecte, $\sigma \in S$. Utilizând un calculator performant, pentru valori nu prea mari ale dimensiunii n, estimarea determinantului $\det(A)$ dupa formula (24) ar dura zile întregi.

In cazul în care matricea A se aduce prin aplicarea transformarilor T1 si T2 (sectiunea 1.4) la forma G triunghiulara de tip Gauss,

$$G = \begin{pmatrix} g_{11} & g_{12} & g_{13} & g_{14} & \cdots & g_{1,n-1} & g_{1n} \\ 0 & g_{22} & g_{23} & g_{24} & \cdots & g_{2,n-1} & g_{2n} \\ 0 & 0 & g_{33} & g_{34} & \cdots & g_{3,n-1} & g_{3n} \\ \cdots & & & \cdots & & & \\ 0 & 0 & 0 & 0 & \cdots & g_{n-1,n-1} & g_{n-1,n} \\ 0 & 0 & 0 & 0 & \cdots & 0 & g_{n-1,n} \end{pmatrix} . \qquad (25)$$

atunci calculul determinantului det (G) este simplu de realizat, deoarece

$$\det (G) = g_{11}\, g_{22}\, g_{33}\, g_{44} \cdots g_{n-1,\, n-1}\, g_{nn} \qquad (26)$$

Vom urmari ideea evaluarii determinantului det (A) prin reducerea matricei A la forma Gauss. Privind acest procedeu mentionam mai multe aspecte.

Observatia 11. In varianta Gauss de rezolvare a sistemelor liniare aplicarea transformarilor T1 si T2 (sectiunea 1.4) nu afecteaza solutia x a sistemului A x = b . In cazul evaluarii determinantului det (A) , folosirea transformarii T1 (înmultirea unei linii cu o constanta c) afecteaza valoarea determinantului matricii rezultate. In adevar, pentru matricile $A^{(1)}$ si $A^{(2)}$ deduse din matricea A prin aplicarea transformarilor T1 , respectiv T2 (relatiile (12)), avem

$$\det (A^{(1)}) = c \ \det (A) \qquad \det (A^{(2)}) = \det (A) \qquad (27)$$

Asadar la exprimarea valorii determinantului det (A) în functie de det (G) se va tine seama si de factorii multiplicativi c ce pot aparea prin utilizarea transformarii T1 (formula (27)).

Observatia 12. In procedura Gauss de alegere a pivotului (ce are cea mai mare valoare absoluta pe linia curenta) devine necesara permutarea a doua coloane ale matricei A . Aceasta operatiune afecteaza însa valoarea determinantului det (A) schimbându-i semnul.

Aplicatia 8. Bazat pe Obsérvatiile 11-12 a fost proiectata în limbajul BASIC functia det (A,n) pentru evaluarea determinantului matricei A de dimensiune n .

Listam în continuare programul sursa al functiei **det** ,

```
FUNCTION det (a(), n)        ' Calculul determinantului matricei A de dimensiune n
REM  Continutul matricei A este transferat intr-o matrice auxiliara G adusa la forma Gauss
p = 1! :              DIM g(n, n)      ' Matrice auxiliara
FOR j = 1 TO n
FOR k = 1 TO n :          g(j, k) = a(j, k) :          NEXT k :          NEXT j
FOR j = 1 TO n - 1                ' Aducerea matricei g la forma triunghiulara Gauss
s = j :        gm = ABS(g(j, j))      ' Determinarea pozitiei (j,s) cu cel mai mare pivot in modul
FOR k = j + 1 TO n
IF ABS(g(j, k)) > gm THEN gm = g(j, k) :          s = k
NEXT k
REM  Permutarea coloanei j cu coloana s
IF j <> s THEN p = -p :        FOR t = 1 TO n :        SWAP g(t, j), g(t, s) :        NEXT t
IF g(j, j) = 0 THEN det = 0! :          GOTO 1
p = p * g(j, j)
REM  Anularea elementelor g(t,j) de sub diagonala principala
FOR t = j + 1 TO n :          c = -g(t, j) / g(j, j) :          g(t, j) = 0
FOR k = j + 1 TO n :          g(t, k) = g(t, k) + c * g(j, k) :          NEXT k
NEXT t :          NEXT j
det = p * g(n, n)
1                 END FUNCTION
```

Aplicatia 9. Având functia **det** ce estimeaza valoarea determinantului unei matrici oarecare, putem rezolva sistemul liniar A x = b prin metoda Cramer. In acest caz componenta x_k , $1 \le k \le n$, a solutiei x este data de formula (3), adica $x_k = \det(A_k) / \det(A)$, matricea A_k având forma (4).

Subrutina **cramer** listata în continuare utilizeaza functia **det (a,n)** pentru rezolvarea sistemului liniar A x = b prin metoda Cramer.

```
REM  Rezolvarea sistemelor liniare prin metoda  Cramer
DECLARE SUB cramer (a!(), b!(), x!(), n!) :       DECLARE FUNCTION det! (a!(), n!)
CLS :      PRINT "Metoda Cramer pentru gasirea solutiei  x  a sistemului liniar  A x = b"
INPUT "n = "; n :          DIM a(n, n), b(n), x(n)
PRINT "Matricea  A  :"          ' Initializarea matricei  A
FOR j = 1 TO n :          FOR k = 1 TO n :          PRINT USING "   a(##,##) = "; j; k;
INPUT ; a(j, k) :          NEXT k :                  NEXT j
PRINT :    PRINT "Vectorul termenilor liberi : "
FOR j = 1 TO n :    PRINT "   b ("; j; ") = "; :    INPUT ; b(j) :    NEXT j :    PRINT
CALL cramer(a(), b(), x(), n)          ' Aplicarea procedurii Cramer

PRINT "Sistemul  A x = b  de "; n; " ecuatii are solutia :"
FOR j = 1 TO n :          PRINT USING "X(##) = ###.##  ,  "; j; x(j); :          NEXT j

SUB cramer (a(), b(), x(), n)          ' Rezolvarea sistemelor liniare prin procedura Cramer
DIM c(n, n) :          d = det(a(), n)
FOR j = 1 TO n
FOR k = 1 TO n :          FOR t = 1 TO n :          c(k, t) = a(k, t) :          NEXT t
c(k, j) = b(k) :          NEXT k
x(j) = det(c(), n) / d :          NEXT j :          END SUB
```

Exemplul 8. Vom rezolva prin metoda Cramer sistemul liniar A x = b unde n = 4 , matricea A si vectorul b fiind definiti în Exemplul 5. Prin rularea procedurii **cramer** a rezultat solutia x = (6.39 , -4.11 , 2.48 , -3.63) . Aceeasi solutie a fost obtinuta prin aplicarea metodei Gauss (Exemplul 5) sau utilizând inversa C a matricei A (Exemplul 7).

4. Sisteme liniare cu mai multe ecuatii decât necunoscute

In practica se impune adesea determinarea unor solutii atunci când numarul de restrictii depaseste numarul de necunoscute. De regula o asemenea problema nu are solutie datorita numarului mare de restrictii , neexistând valori numerice care sa respecte întocmai toate constrângerile impuse.

Fiind obligati de a gasi neaparat o "solutie", aceasta problema va fi reformulata fiind în final redusa la o problema de aproximare. In continuare vom studia o asemenea abordare în cazul unor restrictii liniare, considerând doua sau trei necunoscute.

4.1. Cazul a doua necunoscute

Pentru început vom presupune numai doua necunoscute x, y ce terbuie sa satisfaca n restrictii liniare, n > 2 , adica

$a_j x + b_j y = c_j \quad , \qquad 1 \le j \le n$ (28)

Precizând doua ecuatii ale sistemului (28), în general se poate determina o solutie (x_0 , y_0) ce verifica cele doua restrictii liniare alese. Perechea (x_0 , y_0) astfel obtinuta nu va satisface obligatoriu toate ecuatiile (28). Asadar optând pentru o pereche (x_0 , y_0), valoarea expresiei $a_j x_0 + b_j y_0$ nu este neaparat c_j, aparând astfel o eroare ε_j, $\varepsilon_j = a_j x_0 + b_j y_0 - c_j$, pentru fiecare ecuatie j , $1 \le j \le n$, a sistemului (28).

Cea mai potrivita "solutie" (x_0 , y_0) va trebui sa minimizeze setul de erori "punctuale" ε_j, $1 \le j \le n$. Acceptând expresia $\varepsilon = (\varepsilon_1^{\,2} + \varepsilon_2^{\,2} + \varepsilon_3^{\,2} + \ldots + \varepsilon_n^{\,2}) / n$ drept "masura globala" în multimea erorilor $\{ \varepsilon_1 , \varepsilon_2 , \varepsilon_3 , \ldots , \varepsilon_n \}$ vom determina perechea (x_0 , y_0) care sa minimizeze functia h (x,y) , unde

$$h(x,y) = \left(\sum_{j=1}^{n} (a_j x + b_j y - c_j)^2 \right) / n \qquad (29)$$

Functia h (x,y) fiind infinit derivabila si marginita inferior de zero isi va atinge valoarea minima în punctul (x_0 , y_0) ce anuleaza derivatele sale partiale,

$$\begin{cases} \dfrac{\partial h(x,y)}{\partial x} (x_0 , y_0) = 0 \\ \dfrac{\partial h(x,y)}{\partial y} (x_0 , y_0) = 0 \end{cases} \qquad (30)$$

Din (29) si (30) se obtine un sistem de doua ecuatii liniare având necunoscutele x_0 , y_0 ,

$$\begin{cases} \left(\sum_{j=1}^{n} a_j^2 \right) x_0 + \left(\sum_{j=1}^{n} a_j b_j \right) y_0 = \sum_{j=1}^{n} a_j c_j \\ \left(\sum_{j=1}^{n} a_j b_j \right) x_0 + \left(\sum_{j=1}^{n} b_j^2 \right) y_0 = \sum_{j=1}^{n} b_j c_j \end{cases} \qquad (31)$$

Cu notatia $S_{\alpha\beta} = \left(\sum_{j=1}^{n} \alpha_j \beta_j \right) / n$ solutia sistemului (31) este

$$x_0 = \frac{S_{ac} S_{bb} - S_{ab} S_{bc}}{S_{aa} S_{bb} - (S_{ab})^2} \qquad y_0 = \frac{S_{aa} S_{bc} - S_{ab} S_{ac}}{S_{aa} S_{bb} - (S_{ab})^2} \qquad (32)$$

Aplicatia 10. Listam în continuare programul BASIC de rezolvare a sistemului liniar (28), cu n ecuatii liniare si doua necunoscute x, y . Acest sistem va admite întotdeauna o "solutie" (x_0 , y_0) ce va minimiza expresia (29). Acuratetea solutiei (x_0 , y_0) rezultata prin solutionarea sistemului (31) este data de "eroarea patratica medie" $\varepsilon = h (x_0 , y_0)$ precizata de formula (29).

```
REM     a(j) x  +  b(j) y  =  c(j)  ,     1 <= j <= n
PRINT "Rezolvarea sistemelor liniare cu doua necunoscute si  n  ecuatii ( n > 2 )"
INPUT "Numar de ecuatii = "; n          ' Citirea coeficientilor ecuatiilor liniare
saa = 0!:       sab = 0!:       sac = 0!:       sbb = 0!:       sbc = 0!:       scc = 0!
```

```
FOR j = 1 TO n :        PRINT "Ecuatia ", j; "    a, b, c = ";:        INPUT a, b, c
REM  Determinarea sumelor  saa , sab , sac , sbb , sbc
saa = saa + a * a :        sab = sab + a * b :        sac = sac + a * c
sbb = sbb + b * b :        sbc = sbc + b * c :        scc = scc + c * c
NEXT j
saa = saa / n :            sab = sab / n :            sac = sac / n
sbb = sbb / n :            sbc = sbc / n              scc = scc / n
d = saa * sbb - sab * sab
x = (sac * sbb - sab * sbc) / d          y = (saa * sbc - sab * sac) / d
PRINT USING "Solutia :  x = ###.##  ,   y = ###.##"; x; y
er = scc + saa * x * x + sbb * y * y - 2! * sac * x - 2! * sbc * y + 2! * sab * x * y
PRINT "Eroarea medie = "; er
```

Exemplul 9. Supunând variabilele x, y la urmatoarele 5 restrictii

$2x + 3y = 7$, $x - 4y = -2$, $4x + 5y = 13$, $-3x + y = -5$, $3x + y = 7$

prin rularea programului prezentat în Aplicatia 10 s-a obtinut solutia $(x_0 , y_0) = (1.00 , 2.00)$ cu o eroare patratica medie de $\varepsilon = h(1, 2) = -2.074833E\text{-}13$ (formula (29)).

4.2. Cazul a trei necunoscute

In mod similar poate fi abordat cazul unui sistem liniar cu n ecuatii si numai 3 necunoscute x, y, z, unde $n \geq 3$,

$$\begin{aligned} a_1 x + b_1 y + c_1 z &= d_1 \\ a_2 x + b_2 y + c_2 z &= d_2 \\ a_3 x + b_3 y + c_3 z &= d_3 \\ &\cdots\cdots\cdots \\ a_n x + b_n y + c_n z &= d_n \end{aligned} \qquad (33)$$

Daca $n > 3$ atunci, de regula, sistemul (33) nu admite o solutie în sensul clasic, un triplet (x_0 , y_0 , z_0) ce verifica primele 3 ecuatii nesatisfacând obligatoriu si restul de n-3 ecuatii.

Asadar o solutie a sistemului (33) este un triplet (x_0 , y_0 , z_0) astfel încât

$$a_j x_0 + b_j y_0 + c_j z_0 \approx d_j \quad , \qquad \forall j , \ 1 \leq j \leq n \qquad (34)$$

Conditiile (34) sunt însa vagi nefiind precizata riguros relatia " $\approx$ " (" aproximativ ")

Mai precis "solutia" (x_0 , y_0 , z_0) a sistemului (33) va trebui aleasa astfel încât sa minimizeze toate erorile ε_j , $1 \leq j \leq n$, unde

$$\varepsilon_j = a_j x_0 + b_j y_0 + c_j z_0 - d_j \qquad (35)$$

Tripletul (x_0 , y_0 , z_0) se va deduce astfel încât sa minimizeze suma patratelor erorilor ε_j , $1 \leq j \leq n$, adica

$$h(x_0 , y_0 , z_0) = \text{minimum} \{ h(x, y, z) \mid x \in \mathbf{R} , \ y \in \mathbf{R} , \ z \in \mathbf{R} \} \qquad (36)$$

unde functia $h(x, y, z)$ este data de expresia

$$h(x,y,z) = \sum_{j=1}^{n} \varepsilon_j^2 = \sum_{j=1}^{n} \left(a_j x + b_j y + c_j z - d_j \right)^2 \qquad (37)$$

Punctul de extrem (x_0 , y_0 , z_0) va anula derivatele partiale ale functiei $h(x , y , z)$ si deci va verifica urmatorul sistem liniar de trei ecuatii cu trei necunoscute

$$\begin{cases} \dfrac{\partial h(x_0,y_0,z_0)}{\partial x} = 2\sum_{j=1}^{n}\left(a_j x_0 + b_j y_0 + c_j z_0 - d_j\right)a_j = 0 \\ \dfrac{\partial h(x_0,y_0,z_0)}{\partial y} = 2\sum_{j=1}^{n}\left(a_j x_0 + b_j y_0 + c_j z_0 - d_j\right)b_j = 0 \\ \dfrac{\partial h(x_0,y_0,z_0)}{\partial z} = 2\sum_{j=1}^{n}\left(a_j x_0 + b_j y_0 + c_j z_0 - d_j\right)c_j = 0 \end{cases} \qquad (38)$$

Explicitând relatiile (38) obtinem un sistem liniar de 3 ecuatii în necunoscutele x_0 , y_0 , z_0 ,

$$\begin{cases} \left(\sum_{j=1}^{n} a_j a_j\right)x_0 + \left(\sum_{j=1}^{n} a_j b_j\right)y_0 + \left(\sum_{j=1}^{n} a_j c_j\right)z_0 = \left(\sum_{j=1}^{n} a_j d_j\right) \\ \left(\sum_{j=1}^{n} a_j b_j\right)x_0 + \left(\sum_{j=1}^{n} b_j b_j\right)y_0 + \left(\sum_{j=1}^{n} b_j c_j\right)z_0 = \left(\sum_{j=1}^{n} b_j d_j\right) \\ \left(\sum_{j=1}^{n} a_j c_j\right)x_0 + \left(\sum_{j=1}^{n} b_j c_j\right)y_0 + \left(\sum_{j=1}^{n} c_j c_j\right)z_0 = \left(\sum_{j=1}^{n} c_j d_j\right) \end{cases} \qquad (39)$$

Notând prin $S_{\alpha\beta}$ sumele ce apar între paranteze în ecuatiile (39), rescriem sistemul respectiv în forma

$$\begin{cases} S_{aa} x_0 + S_{ab} y_0 + S_{ac} z_0 = S_{ad} \\ S_{ab} x_0 + S_{bb} y_0 + S_{bc} z_0 = S_{bd} \\ S_{ac} x_0 + S_{bc} y_0 + S_{cc} z_0 = S_{cd} \end{cases} \qquad (40)$$

Solutia (x_0 , y_0 , z_0) poate fi dedusa aplicând metoda Cramer, adica

$$x_0 = \Delta_x / \Delta \qquad y_0 = \Delta_y / \Delta \qquad z_0 = \Delta_z / \Delta \qquad (41)$$

unde $\Delta , \Delta_x , \Delta_y , \Delta_z$ sunt urmatorii determinanti

$$\Delta = \det\begin{pmatrix} S_{aa} & S_{ab} & S_{ac} \\ S_{ab} & S_{bb} & S_{bc} \\ S_{ac} & S_{bc} & S_{cc} \end{pmatrix} \qquad \Delta_x = \det\begin{pmatrix} S_{ad} & S_{ab} & S_{ac} \\ S_{bd} & S_{bb} & S_{bc} \\ S_{cd} & S_{bc} & S_{cc} \end{pmatrix}$$

$$\Delta_y = \det\begin{pmatrix} S_{aa} & S_{ad} & S_{ac} \\ S_{ab} & S_{bd} & S_{bc} \\ S_{ac} & S_{cd} & S_{cc} \end{pmatrix} \qquad \Delta_z = \det\begin{pmatrix} S_{aa} & S_{ab} & S_{ad} \\ S_{ab} & S_{bb} & S_{bd} \\ S_{ac} & S_{bc} & S_{cd} \end{pmatrix} \qquad (42)$$

Valoarea unui determinant de forma (42) poate fi obtinuta efectiv aplicând de exemplu formula lui Sarrus. Astfel

$$\Delta = S_{aa} S_{bb} S_{cc} + S_{ab} S_{bc} S_{ac} + S_{ac} S_{ab} S_{bc} - S_{ac} S_{bb} S_{ac} - S_{bc} S_{aa} S_{bc} - S_{ab} S_{cc} S_{ab}$$

Aplicatia 11. Listam programul BASIC de determinarea solutiei (x_0 , y_0 , z_0) .

```
REM  Rezolvarea unui sistem liniar cu n ecuatii si 3 necunoscute
```

```
CLS :          INPUT "n = "; n :          DIM a(n), b(n), c(n), d(n)
FOR j = 1 TO n :          PRINT "Ecuatia "; j; " :          a, b, c, d = ";
INPUT a(j), b(j), c(j), d(j) :          NEXT j
REM  Calculul sumelor  saa , sab , ... , scd
saa = 0 :      sab = 0 :      sac = 0 :      sad = 0
sbb = 0 :      sbc = 0 :      sbd = 0 :      scc = 0 :      scd = 0
FOR j = 1 TO n
saa = saa + a(j) * a(j) :      sab = sab + a(j) * b(j) :      sac = sac + a(j) * c(j)
sad = sad + a(j) * d(j) :      sbb = sbb + b(j) * b(j) :      sbc = sbc + b(j) * c(j)
sbd = sbd + b(j) * d(j) :      scc = scc + c(j) * c(j) :      scd = scd + c(j) * d(j)
NEXT j
REM  Calculul determinantilor
det = saa * sbb * scc + sab * sbc * sac + sac * sab * sbc - sac * sbb * sac - sbc * saa * sbc - sab * scc
* sab
detx = sad * sbb * scc + sbd * sbc * sac + scd * sab * sbc - sac * sbb * scd - sbc * sad * sbc - sab *
scc * sbd
dety = saa * sbd * scc + sab * scd * sac + sac * sad * sbc - sac * sbd * sac - sbc * scd * saa - scc * sab
* sad
detz = saa * sbb * scd + sab * sbc * sad + sac * sab * sbd - sac * sbb * sad - sbc * sbd * saa - sab *
scd * sab
REM  Obtinerea solutiei prin metoda Cramer
x = detx / det :          y = dety / det :          z = detz / det
PRINT "Solutia sistemului :    x ="; x; "     y = "; y; "     z = "; z
PRINT "Erorile rezultate" :          ser = 0!
FOR j = 1 TO n
er = a(j) * x + b(j) * y + c(j) * z - d(j) :          ser = ser + ABS (er)
PRINT "Er("; j; " ) = "; er; "  ,  "; :          NEXT j
ser = ser / n :          PRINT :          PRINT "Eroarea totala = "; ser
```

Programul a fost proiectat avându-se în vedere solutionarea unui sistem liniar cu n ecuatii si numai trei necunoscute x, y, z. Numele variabilelor utilizate au fost alese astfel încât sa sugereze operatiile respective. Astfel variabilele saa, sab, det, detx retin valorile expresiilor S_{aa} , S_{ab} , Δ , respectiv Δ_x . Memorarea coeficientilor a_j , b_j , c_j , d_j , $1 \le j \le n$, s-a realizat în vectorii a, b, c, d

Odata obtinut tripletul $(x_0\ y_0 , z_0)$ ce minimizeaza functia h(x,y,z) data de formula (37) sunt listate toate erorile ε_j , $\varepsilon_j = a_j x_0 + b_j y_0 + c_j z_0 - d_j$, $1 \le j \le n$, produse prin acceptarea unei astfel de solutii, cât si eroarea totala ε privita ca o medie a modulelor erorilor partiale ε_j , $1 \le j \le n$,

$$\varepsilon = [\,|\varepsilon_1| + |\varepsilon_2| + |\varepsilon_3| + \ldots + |\varepsilon_{n-1}| + |\varepsilon_n|\,] / n \qquad (43)$$

Observatia 13. Precizia solutiei (x_0 , y_0) a sistemului liniar (28) a fost masurata prin eroarea patratica medie $h(x_0 , y_0)$, functia h având expresia (29). In schimb pentru sistemul (33) eroarea ε produsa prin acceptarea solutiei (x_0 , y_0 , z_0) are forma (43) , fiind de aceasta data o medie a modulelor erorilor ε_j asociate fiecarei ecuatii, $1 \le j \le n$.

Exemplul 10. Validarea programului din Aplicatia 11 s-a efectuat luând în considerare un sistem compatibil de 3 ecuatii cu 3 necunoscute a carui solutie "exacta" este cunoscuta.
Astfel pentru n = 3 considerând ecuatiile 2 x - 3 y + z = -1 , x + y + 2 z = 9 , x + 2 y - z = 2

a rezultat solutia "exacta" (1, 2, 3) a sistemului (33) cu erorile $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = 0$

Exemplul 11. Pentru urmatorul sistem de 7 ecuatii în necunoscutele x, y, z ,

$10x + 15y + 20z = 80$, $10x + 15y + 20z = 81$, $10x + 15y + 20z = 78$, $9x + 15y + 20z = 78$

$10x + 15y + 19z = 79$, $10x + 15y + 19z = 80$, $10x + 14y + 19z = 80$

a rezultat solutia (3.804694 , .8589156 , 1.489344) cu erorile $\varepsilon_1 = .7175624$, $\varepsilon_2 = -.2824376$, $\varepsilon_3 = 2.717562$, $\varepsilon_4 = -1.087132$, $\varepsilon_5 = .228218$, $\varepsilon_6 = -.771782$, $\varepsilon_7 = -1.630698$.

Eroarea medie absoluta este $\varepsilon = 1.062199$ (formula (43)) .

Observatia 14. O adaptare a acestui program pentru un numar m mai mare de necunoscute, $m > 3$, presupune folosirea unor tablouri în locul variabilelor de tipul S_{aa} , S_{ab} , ... , S_{cd} . De asemenea solutionarea în final a unui sistem liniar de dimensiune m ce extinde forma (40) este de preferat a se realiza prin metoda Gauss si nu aplicând procedura Cramer.

VECTORI SI VALORI PROPRII

Fie A o matrice patratica de dimensiune n, $A = (a_{jk})$, $a_{jk} \in \mathbf{R}$, $\forall\ 1 \le j, k \le n$, iar $v = (v_1, v_2, v_3, \ldots, v_n)$ un vector nenul (v are cel putin o componenta v_j nenula), $v_k \in \mathbf{R}$, $1 \le k \le n$

Definitie. Vectorul v nenul este vector propriu pentru matricea A daca exista $\lambda \in \mathbf{R}$, $\lambda \neq 0$, care sa verifice relatia matriciala

$$A v = \lambda v \qquad (1)$$

Constanta numerica nenula λ se numeste valoare proprie.

Propozitia 1. Daca v este un vector propriu corespunzator valorii proprii λ atunci pentru orice scalar $\alpha \neq 0$ vectorul $w = \alpha v$ este de asemenea vector propriu atasat aceleasi valori proprii λ

Demonstratie. Cum v este vector nenul si $\alpha \neq 0$ rezulta vectorul nenul $w = \alpha v$ In plus

$$A w = A (\alpha v) = \alpha (A v) = \alpha (\lambda v) = (\lambda \alpha) v = \lambda (\alpha v) = \lambda w$$

adica w este vector propriu cu valoarea proprie λ

Asadar pentru aceeasi valoare λ exista o infinitate de vectori proprii w

Corolarul 1. Daca $v = (v_1, v_2, v_3, \ldots, v_n)$ este vector propriu de valoare proprie λ si $v_j \neq 0$ atunci exista un vector propriu $w = (w_1, w_2, w_3, \ldots, w_n)$ având valoarea proprie λ si $w_j = a$, pentru orice $a \neq 0$ fixat.

Demonstratie. Consideram în Propozitia 1 $\alpha = a / v_j \neq 0$ caz în care pentru vectorul propriu $w = \alpha v$ avem $w_j = \alpha v_j = a$.

1. Proprietati ale valorilor si vectorilor proprii

In cele ce urmeaza vom analiza un procedeu general de determinare a valorilor si vectorilor proprii pentru o matrice A precizata.

Propozitia 2. Valorile proprii sunt radacinile ecuatiei polinomiale de gradul n

$$\det (A - \lambda E) = 0 \qquad (2)$$

unde $E = (e_{jk})$ este matricea identitate de dimensiune n ($e_{jk} = 0$, $e_{jj} = 1$, $\forall\ 1 \le j \neq k \le n$).

Demonstratie. In adevar, reinterpretând relatia (1) deducem componentele v_j, $1 \le j \le n$, ale vectorului propriu v drept solutie a sistemului liniar si omogen

$$\begin{aligned}
&(a_{11} - \lambda) v_1 + a_{12} v_2 + a_{13} v_3 + a_{14} v_4 + \ldots + a_{1,n-1} v_{n-1} + a_{1n} v_n = 0 \\
&a_{21} v_1 + (a_{22} - \lambda) v_2 + a_{23} v_3 + a_{24} v_4 + \quad + a_{2,n-1} v_{n-1} + a_{2n} v_n = 0 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \qquad (3) \\
&a_{n-1,1} v_1 + a_{n-1,2} v_2 + a_{n-1,3} v_3 + \quad + (a_{n-1,n-1} - \lambda) v_{n-1} + a_{n-1,n} v_n = 0 \\
&a_{n1} v_1 + a_{n2} v_2 + a_{n3} v_3 + a_{n4} v_4 + \quad + a_{n,n-1} v_{n-1} + (a_{nn} - \lambda) v_n = 0
\end{aligned}$$

Cum vectorul propriu v este nenul rezulta ca sistemul liniar si omogen (3) admite pe lânga solutia nula si o solutie nenula. Acest lucru este însa echivalent cu a cere ca determinantul principal al sistemului (3) sa fie nul (sistemul (3) nu admite solutie unica), adica tocmai relatia (2)

Observatia 1. Valoarea proprie λ este radacina a polinomului (2), de gradul n. Putem astfel avea cel mult n valori proprii reale $\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n$.

Pentru o valoare proprie λ ce a fost precizata, determinarea componentelor v_j, $1 \le j \le n$, ale vectorului propriu corespunzator v revine la a gasi o solutie nenula pentru sistemul omogen (3)

Tinând seama de Corolarul 1 putem impune de exemplu restrictia $v_1 = 1$ (în cazul în care prima componenta a vectorului propriu v este nenula).

Exemplul 1. Sa determinam valorile si vectorii proprii pentru matricea patratica $A = (a_{jk})$, de dimensiune 2 , unde $a_{11} = 5$, $a_{12} = 4$, $a_{21} = 6$, $a_{22} = 15$.

Relatia (2) conduce la urmatoarea ecuatie $\lambda^2 - 20\lambda + 51 = 0$ a carei solutii sunt $\lambda_1 = 3$ si $\lambda_2 = 17$. Gasirea vectorilor proprii $v^{(1)} = (1 , a)$, $v^{(2)} = (1 , b)$ corespunzatoare valorilor proprii $\lambda_1 = 3$, respectiv $\lambda_2 = 17$, revine la rezolvarea unor sisteme liniare derivate din (3) pentru $\lambda = \lambda_j$ si $v_1 = 1$, adica

$5 + 4a = 3$; $6 + 15a = 3a$; $5 + 4b = 17$; $6 + 15b = 17a$

Se obtine astfel $a = -1/2$ si $b = 3$.

Asadar cei doi vectori proprii ai matricei A sunt de forma $v^{(1)} = (1 , -1/2)$, $v^{(2)} = (1 , 3)$ valorile proprii atasate fiind $\lambda_1 = 3$, respectiv $\lambda_2 = 17$.

Observatia 2. Pentru o matrice A având drept elemente numere reale pot rezulta si valori proprii ce sunt numere complexe. Astfel considerând matricea $A = (a_{jk})$ de dimensiune 2 , unde $a_{11} = 2$, $a_{12} = -1$, $a_{21} = 4$, $a_{22} = 3$ se obtine ecuatia caracteristica $(2 - \lambda)(3 - \lambda) + 4 = 0$ ale carei radacini sunt numere complexe.

Propozitia 3. Daca o matrice A admite valori proprii reale si distincte atunci cei n vectori proprii corespunzatori sunt liniar independenti, elementele vectorilor proprii fiind numere reale.

Demonstratia va fi facuta prin inductie matematica în raport cu numarul m al vectorilor proprii utilizati. In cazul $m = 1$, fapt ce presupune existenta unui singur vector propriu $v^{(1)}$, acest unic vector $v^{(1)}$ nenul va fi liniar independent. In adevar, notând prin $\underline{0}$ vectorul nul, cum $v^{(1)} \neq \underline{0}$ din egalitatea $\alpha v^{(1)} = \underline{0}$ pentru o valoare fixata $\alpha \in \mathbf{R}$ rezulta în mod obligatoriu $\alpha = 0$

Presupunem ca m vectori proprii $v^{(1)}$, $v^{(2)}$, ... , $v^{(m)}$ ai matricii A sunt liniar independenti Sa demonstram ca prin adaugarea unui nou vector propriu $v^{(m+1)}$ toti cei $m+1$ vectori proprii ramân liniar independenti.

Fie asadar $\alpha_j \in \mathbf{R}$, $1 \leq j \leq m+1$, astfel încât

$$\alpha_1 v^{(1)} + \alpha_2 v^{(2)} + \alpha_3 v^{(3)} + \quad + \alpha_m v^{(m)} + \alpha_{m+1} v^{(m+1)} = \underline{0} \qquad (4)$$

Inmultind la stânga cu matricea A , egalitatea (4) devine

$$\underline{0} = A\,\underline{0} = A (\alpha_1 v^{(1)} + \alpha_2 v^{(2)} + \alpha_3 v^{(3)} + \quad + \alpha_m v^{(m)} + \alpha_{m+1} v^{(m+1)}) =$$

$$= \alpha_1 A v^{(1)} + \alpha_2 A v^{(2)} + \alpha_3 A v^{(3)} + \quad + \alpha_m A v^{(m)} + \alpha_{m+1} A v^{(m+1)}$$

Tinând seama ca $v^{(j)}$, $1 \leq j \leq m+1$, sunt vectori proprii, din formula anterioara rezulta

$$\alpha_1 \lambda_1 v^{(1)} + \alpha_2 \lambda_2 v^{(2)} + \alpha_3 \lambda_3 v^{(3)} + \ldots + \alpha_m \lambda_m v^{(m)} + \alpha_{m+1} \lambda_{m+1} v^{(m+1)} = \underline{0} \qquad (5)$$

Scazând din relatia (5) relatia (4) multiplicata cu λ_{m+1} deducem

$$\alpha_1 (\lambda_1 - \lambda_{m+1}) v^{(1)} + \alpha_2 (\lambda_2 - \lambda_{m+1}) v^{(2)} + \ldots + \alpha_m (\lambda_m - \lambda_{m+1}) v^{(m)} = \underline{0}$$

Din ipoteza de independenta a primilor vectori proprii $v^{(1)}$, $v^{(2)}$, ... , $v^{(m)}$ se obtine $\alpha_j (\lambda_j - \lambda_{m+1}) = 0$ pentru orice $1 \leq j \leq m$. Cum $\lambda_j \neq \lambda_{m+1}$ pentru $1 \leq j \leq m$ (valorile proprii sunt distincte) rezulta $\alpha_j = 0$, $\forall\, j$, $1 \leq j \leq m$. In aceasta situatie relatia (4) devine $\alpha_{m+1} v^{(m+1)} = \underline{0}$, deci $\alpha_{m+1} = 0$ Prin urmare vectorii proprii $v^{(1)}$, $v^{(2)}$, , $v^{(m)}$, $v^{(m+1)}$ sunt liniar independenti deoarece orice combinatie liniara a acestor vectori proprii ce este tocmai vectorul $\underline{0}$ a condus în final

la egalitatea $\alpha_1 = \alpha_2 = \ldots = \alpha_m = \alpha_{m+1} = 0$

In cazul în care valoarea proprie λ este un numar real atunci vectorul propriu v are toate componentele sale numere reale (a se analiza în acest sens sistemul (3)).

Din Propozitia 3 rezulta :

Corolarul 2 Daca toate valorile proprii ale matricei A sunt reale si distincte atunci cei n vectori proprii $v^{(k)}$, $1 \le k \le n$, determina o baza a spatiului $\mathbf{R}^n$.

Exemplul 2. Fie matricea patratica $A = (a_{jk})$ de dimensiune 3, unde $a_{11} = 8$, $a_{12} = -7$, $a_{13} = 1$, $a_{21} = 12$, $a_{22} = -11$, $a_{23} = 1$, $a_{31} = 12$, $a_{32} = -14$, $a_{33} = 4$.

Cele trei valori proprii $\lambda_1, \lambda_2, \lambda_3$ ale matricei A sunt radacinile ecuatiei caracteristice (2),

$$\det\left[\begin{pmatrix} 8 & -7 & 1 \\ 12 & -11 & 1 \\ 12 & -14 & 4 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\right] = \det\begin{bmatrix} 8-\lambda & -7 & 1 \\ 12 & -11-\lambda & 1 \\ 12 & -14 & 4-\lambda \end{bmatrix} = 0$$

fapt ce revine la a rezolva ecuatia de gradul 3

$$\lambda^3 - \lambda^2 - 14\lambda + 24 = 0$$

ale carei radacini sunt $\lambda_1 = -4$, $\lambda_2 = 3$, $\lambda_3 = 2$.

Daca $v^{(k)} = (1, v_2^{(k)}, v_3^{(k)})$ este vectorul propriu asociat valorii proprii λ_k, $1 \le k \le 3$, atunci componentele acestor vectori vor satisface un sistem liniar si omogen de forma (3). Avem respectiv relatiile

$$\begin{cases} 7v_2^{(1)} - v_3^{(1)} = 12 \\ 14v_2^{(1)} - 8v_3^{(1)} = 12 \end{cases} \qquad \begin{cases} 7v_2^{(2)} - v_3^{(2)} = 5 \\ 14v_2^{(2)} - v_3^{(2)} = 12 \end{cases} \qquad \begin{cases} 7v_2^{(3)} - v_3^{(3)} = 6 \\ 13v_2^{(3)} - v_3^{(3)} = 12 \end{cases}$$

Prin rezolvarea acestor trei sisteme de doua ecuatii cu doua necunoscute se obtin valorile componentelor $v_2^{(k)}, v_3^{(k)}$, $1 \le k \le 3$, pentru vectorii proprii $v^{(1)}, v^{(2)}, v^{(3)}$ ce sunt asociati valorilor proprii $\lambda_1 = -4$, $\lambda_2 = 3$, respectiv $\lambda_3 = 2$.

Deducem astfel $v^{(1)} = (1, 2, 2)$; $v^{(2)} = (1, 1, 2)$; $v^{(3)} = (1, 1, 1)$.

2. Cea mai mare valoare proprie

In cele ce urmeaza vom presupune ipoteza simplificatoare :

Ipoteza 1. Matricea A admite numai valori proprii λ_j reale pentru care exista în mod unic o cea mai mare valoare proprie în modul. Notând aceasta valoare cu λ_1 avem

$$|\lambda_1| > |\lambda_2| \ge |\lambda_3| \ge |\lambda_4| \ge \ldots \ge |\lambda_{n-1}| \ge |\lambda_n| \qquad (6)$$

Observatia 3. In vederea simplificarii expunerii am acceptat Ipoteza 1 prin care toate valorile proprii ale matricei A sunt numere reale. Aceasta varianta poate fi însa adaptata cu mici modificari si în cazul unor valori proprii numere complexe. Mentionam ca într-o asemenea situatie exista cel putin doua valori proprii egale în modul. In adevar daca $\lambda_1 = \alpha + \beta i$ este o solutie, numar complex, a ecuatiei caracteristice (2) atunci si conjugata sa $\lambda_2 = \alpha - \beta i$ verifica de asemenea egalitatea (2). In plus cele doua solutii λ_1 si λ_2 au acelasi modul, adica $|\lambda_1| = |\lambda_2| = (\alpha^2 + \beta^2)^{1/2}$.

Vom analiza un algoritm pentru determinarea celei mai mari valori proprii λ_1.

Ipoteza 2. Fie vectorul $x^{(0)} \in \mathbf{R}^n$ ales astfel încât sa fie satisfacuta egalitatea vectoriala

$$x^{(0)} = \alpha_1 v^{(1)} + \alpha_2 v^{(2)} + \alpha_3 v^{(3)} + \ldots + \alpha_{n-1} v^{(n-1)} + \alpha_n v^{(n)} \tag{7}$$

unde $\alpha_k \in \mathbf{R}^n$ sunt numere reale arbitrare cu $\alpha_1 \neq 0$, iar $v^{(k)}$ este vectorul propriu asociat valorii proprii λ_k, $1 \leq k \leq n$.

Considerând un vector $x^{(0)}$ ce satisface Ipoteza 2, definim iterativ sirul vectorilor $\{ x^{(t)} \}_{t \in N}$, dupa urmatoarea formula

$$x^{(t+1)} = A\, x^{(t)} \tag{8}$$

Tinând seama de Ipoteza 2 unde $v^{(k)}$, $1 \leq k \leq n$, sunt vectori proprii avem succesiv

$$x^{(1)} = A\, x^{(0)} = A (\alpha_1 v^{(1)} + \alpha_2 v^{(2)} + \alpha_3 v^{(3)} + \ldots + \alpha_{n-1} v^{(n-1)} + \alpha_n v^{(n)}) =$$

$$= \alpha_1 Av^{(1)} + \alpha_2 Av^{(2)} + \ldots + \alpha_n Av^{(n)} = \alpha_1 \lambda_1 v^{(1)} + \alpha_2 \lambda_2 v^{(2)} + \ldots + \alpha_{n-1} \lambda_{n-1} v^{(n-1)} + \alpha_n \lambda_n v^{(n)}$$

$$x^{(2)} = A\, x^{(1)} = A (\alpha_1 \lambda_1 v^{(1)} + \alpha_2 \lambda_2 v^{(2)} + \ldots + \alpha_{n-1} \lambda_{n-1} v^{(n-1)} + \alpha_n \lambda_n v^{(n)}) =$$

$$= \alpha_1 \lambda_1 Av^{(1)} + \alpha_2 \lambda_2 Av^{(2)} + \ldots + \alpha_n \lambda_n Av^{(n)} = \alpha_1 \lambda_1^2 v^{(1)} + \alpha_2 \lambda_2^2 v^{(2)} + \ldots + \alpha_n \lambda_n^2 v^{(n)}$$

Pornind de la expresiile vectorilor $x^{(1)}$ si $x^{(2)}$ vom presupune

$$x^{(t)} = \alpha_1 \lambda_1^t v^{(1)} + \alpha_2 \lambda_2^t v^{(2)} + \ldots + \alpha_{n-1} \lambda_{n-1}^t v^{(n-1)} + \alpha_n \lambda_n^t v^{(n)} \tag{9}$$

Egalitatea (9) se confirma prin inductie matematica pentru orice $t \in \mathbf{N}$. In adevar vectorul $x^{(0)}$ specificat prin Ipoteza 2 are forma (9). In plus

$$x^{(t+1)} = A\, x^{(t)} = A (\alpha_1 \lambda_1^t v^{(1)} + \alpha_2 \lambda_2^t v^{(2)} + \ldots + \alpha_{n-1} \lambda_{n-1}^t v^{(n-1)} + \alpha_n \lambda_n^t v^{(n)}) =$$

$$= \alpha_1 \lambda_1^t Av^{(1)} + \alpha_2 \lambda_2^t Av^{(2)} + \ldots + \alpha_n \lambda_n^t Av^{(n)} = \alpha_1 \lambda_1^{t+1} v^{(1)} + \alpha_2 \lambda_2^{t+1} v^{(2)} + \ldots + \alpha_n \lambda_n^{t+1} v^{(n)}$$

Rescriind pe componente relatia vectoriala (9), pentru $1 \leq j \leq n$ rezulta egalitatile

$$x_j^{(t)} = \alpha_1 \lambda_1^t v_j^{(1)} + \alpha_2 \lambda_2^t v_j^{(2)} + \ldots + \alpha_{n-1} \lambda_{n-1}^t v_j^{(n-1)} + \alpha_n \lambda_n^t v_j^{(n)}$$

Deci pentru orice $1 \leq j \leq n$, avem

$$\frac{x_j^{(t+1)}}{x_j^{(t)}} = \frac{\alpha_1 \lambda_1^{t+1} v_j^{(1)} + \alpha_2 \lambda_2^{t+1} v_j^{(2)} + \ldots + \alpha_{n-1} \lambda_{n-1}^{t+1} v_j^{(n-1)} + \alpha_n \lambda_n^{t+1} v_j^{(n)}}{\alpha_1 \lambda_1^t v_j^{(1)} + \alpha_2 \lambda_2^t v_j^{(2)} + \ldots + \alpha_{n-1} \lambda_{n-1}^t v_j^{(n-1)} + \alpha_n \lambda_n^t v_j^{(n)}} =$$

$$= \frac{\alpha_1 \lambda_1 v_j^{(1)} + \alpha_2 \lambda_2 \left(\lambda_2 / \lambda_1\right)^t v_j^{(2)} + \ldots + \alpha_n \lambda_n \left(\lambda_n / \lambda_1\right)^t v_j^{(n)}}{\alpha_1 v_j^{(1)} + \alpha_2 \left(\lambda_2 / \lambda_1\right)^t v_j^{(2)} + \ldots + \alpha_n \left(\lambda_n / \lambda_1\right)^t v_j^{(n)}} \tag{10}$$

Propozitia 4. Sirul de vectori $\{ x^{(t)} \}_{t \in N}$ definit iterativ de (8), în conditiile Ipotezelor 1 si 2, are proprietatea

$$\lim_{t \to \infty} \frac{x_j^{(t+1)}}{x_j^{(t)}} = \lambda \qquad \forall\ 1 \leq j \leq n \tag{11}$$

Demonstratie. Conform Ipotezei 1 avem $|\lambda_k| / |\lambda_1| < 1$, $\forall\ 2 \leq k \leq n$. Pentru un indice k fixat, $2 \leq k \leq n$, sirul $\{ (\lambda_k / \lambda_1)^t \}_{t \in N}$ este convergent la zero si deci tinând seama de (10) se obtine (11).

Propozitia 4 sugereaza determinarea celei mai mari valori proprii în modul dupa urmatorul algoritm :

Algoritmul VPMM (Valoare Proprie Maxima în Modul)

Pas 0. Initializarea vectorului $x^{(0)}$; $t = 0$

Pas 1. $x^{(t+1)} = A\,x^{(t)}$; $p_j = x_j^{(t+1)} / x_j^{(t)}$, $1 \le j \le n$; $t = t + 1$

Pas 2. Daca ($p_i \approx p_j$ pentru $1 \le i < j \le n$) atunci $\lambda_1 = (p_1 + p_2 + \ldots + p_n) / n$; STOP

altfel Mergi la Pasul 1 .

Observatia 4. Pentru a decide în Pasul 2 al Algoritmului **VPMM** daca pentru orice $1 \le i < j \le n$ avem $p_i = x_i^{(t+1)} / x_i^{(t)} \approx x_j^{(t+1)} / x_j^{(t)} = p_j$, vom verifica inegalitatea $p_{max} - p_{min} < \varepsilon$ unde $p_{max} = \max \{ p_j \mid 1 \le j \le n \}$, $p_{min} = \min \{ p_j \mid 1 \le j \le n \}$, ε fiind eroarea acceptata în procesul de aproximare.

Observatia 5. Daca vectorul initial $x^{(0)}$ nu satisface Ipoteza 2 atunci Propozitia 4 nu poate fi aplicata (nu este asigurata convergenta (11)). In aceasta situatie este imposibil de evaluat cu ajutorul algoritmului **VPMM** cea mai mare valoare proprie λ_1 . Vom evita acest neajuns generând aleator, cu ajutorul procedurii RND , componentele vectorului $x^{(0)}$ ce initializeaza procesul iterativ (8). In situatia în care, la executii repetate ale algoritmului **VPMM** pentru diferiti vectori initiali $x^{(0)}$, se obtin valori proprii comparabile vom retine una dintre variante în vederea estimarii lui λ_1 .

Aplicatia 1. Urmatorul program constituie o implementare a algoritmului **VPMM** în limbajul BASIC . La proiectarea programului nu s-a retinut întregul sir de vectori $x^{(0)}$, $x^{(1)}$, $x^{(2)}$, ... , $x^{(t)}$, $x^{(t+1)}$. In fiecare moment algoritmul **VPMM** opereaza numai cu doi vectori $x^{(k)}$ si $x^{(k+1)}$ consecutivi, rolul lor fiind preluat în programul BASIC de vectorii auxiliari x si y de dimensiune n .

```
REM  Aflarea celei mai mari valori proprii in modul
CLS :       RANDOMIZE TIMER :       INPUT "n = "; n :       DIM a(n, n), x(n), y(n), p(n)
PRINT "Matricea  A : " :       FOR j = 1 TO n :       PRINT "linia "; j; " :  ";
FOR k = 1 TO n :       INPUT ; a(j, k) :       PRINT " , "; :       NEXT k
PRINT :       NEXT j
REM  Initializarea vectorului  x
FOR j = 1 TO n :       x(j) = RND :       NEXT j
eps = .01 :       tmax = 35 :       t = 0 :       PRINT :       PRINT "x( 0) = ";
FOR j = 1 TO n :       PRINT USING "######^^^^ , "; x(j); :       NEXT j
DO :       t = t + 1
FOR j = 1 TO n :       y(j) = 0
FOR k = 1 TO n :       y(j) = y(j) + a(j, k) * x(k) :       NEXT k
IF x(j) = 0 THEN p(j) = .00000001# ELSE p(j) = y(j) / x(j)
NEXT j
pmin = p(1) :       pmax = p(1) :       s = 0
FOR j = 1 TO n :       s = s + p(j)
IF p(j) < pmin THEN pmin = p(j)
IF p(j) > pmax THEN pmax = p(j)
x(j) = y(j) :       NEXT j :       s = s / n
PRINT :       PRINT USING "x(##) = "; t;
FOR j = 1 TO n :       PRINT USING "#######^^^^ , "; x(j); :       NEXT j
PRINT :       PRINT USING "    p(##) = "; t;
FOR j = 1 TO n :       PRINT USING "####.## , ": p(j); :       NEXT j
```

```
PRINT USING "    Medie = ####.##      Variatie = ##.###"; s; pmax - pmin;
LOOP UNTIL  (t >= tmax) OR ((pmax - pmin) < eps)
```

Observatia 6. In programul BASIC anterior s-a optat pentru oprirea algoritmului **VPMM** dupa cel mult $t_{max} = 35$ de iteratii, $0 \le t \le t_{max}$, aceasta deoarece exista situatii în care procesul iterativ (8) nu converge (datorita nesatisfacerii Ipotezei 2). In acest caz algoritmul **VPMM** va putea fi reluat dupa o noua reinitializare a componentelor vectorului $x^{(0)}$ cu valori numerice generate aleator.

Exemplul 3. Fie $n = 3$ si matricea patratica $A = (a_{ij})_{1 \le i, j \le 3}$ cu $a_{11} = 8$, $a_{12} = -7$, $a_{13} = 1$, $a_{21} = 12$, $a_{22} = -11$, $a_{23} = 1$, $a_{31} = 12$, $a_{32} = -14$, $a_{33} = 4$. Matricea A are valorile proprii $\lambda_1 = -4$, $\lambda_2 = 3$, $\lambda_3 = 2$ (a se vedea Exemplul 2) fiind satisfacuta Ipoteza 1 ($|\lambda_1| > |\lambda_2| > |\lambda_3|$). Acceptând o eroare de $\varepsilon = 0.01$ s-a asigurat convergenta procesului iterativ (8) dupa 21 de iteratii.

Aproximarea $\lambda_{1*}^{(t)}$ a valorii proprii λ_1 s-a calculat la fiecare iteratie t, $t \le t_{max}$, utilizându-se formula $\lambda_{1*}^{(t)} = (p_1 + p_2 + p_3 + \ldots + p_n) / n$, unde $p_j = x_j^{(t+1)} / x_j^{(t)}$, $1 \le j \le n$.

In Tabelul 1, pe lânga valorile componentelor $x_j^{(t)}$, $1 \le j \le n$, este listata si lungimea p_{var} a intervalului de variatie a valorilor p_j, $1 \le j \le n$, $p_{var} = p_{max} - p_{min}$, apreciindu-se astfel precizia în procesul de aproximare a valorii proprii λ_1.

Tabelul 1. Evaluarea experimentala $\lambda_{1*}^{(t)}$, la pasul t, a valorii proprii λ_1.

t	$x_1^{(t)}$	$x_2^{(t)}$	$x_3^{(t)}$	p_1	p_2	p_3	$\lambda_{1*}^{(t)}$	p_{var}
0	9819E-07	4180E-06	7383E-06					
1	-1402E-03	-2682E-03	-1721E-03	-14.28	-6.42	-2.33	-7.68	11.95
2	5832E-03	1095E-02	1383E-02	-4.16	-4.08	-8.04	-5.43	3.955
3	-1616E-02	-3662E-02	-2798E-02	-2.77	-3.35	-2.02	-2.71	1.322
4	9914E-02	1810E-01	2070E-01	-6.14	-4.94	-7.40	-6.16	2.455
5	-2670E-01	-5945E-01	-5167E-01	-2.69	-3.28	-2.50	-2.82	0.788
6	1509E+00	2819E+00	3052E+00	-5.65	-4.74	-5.91	-5.43	1.166
7	-4609E+00	-9849E+00	-9148E+00	-3.05	-3.49	-3.00	-3.18	0.497
8	2292E+01	4388E+01	4598E+01	-4.97	-4.46	-5.03	-4.82	0.571
9	-7781E+01	-1617E+02	-1553E+02	-3.39	-3.68	-3.38	-3.49	0.305
10	3537E+02	6891E+02	7080E+02	-4.55	-4.26	-4.56	-4.46	0.295
20	3528E+08	7044E+08	7055E+08	-4.03	-4.01	-4.03	-4.02	0.015
21	-1403E+09	-2810E+09	-2806E+09	-3.98	-3.99	-3.98	-3.98	0.011
22	5636E+09	1126E+10	1127E+10	-4.02	-4.01	-4.02	-4.01	0.008

Observatia 7. Din formula (9) rezulta ca pentru $|\lambda_1| > 1$ componentele vectorului $x^{(t)}$

devin foarte mari odata cu cresterea lui t (a se vedea în Tabelul 1 iteratiile t = 20, 21, 22). In schimb aceleasi componente $x_j^{(t)}$ tind la valori aproape nule daca $|\lambda_1| < 1$. Aceste aspecte pot conduce la aparitia în programul BASIC a unor erori de calcul majore. Pentru evitarea acestor neajunsuri se recomanda "normalizarea" la fiecare iteratie a vectorilor $x^{(t)}$ prin împartirea componentelor sale cu o constanta c, de exemplu $c = |x_1^{(t)}| + |x_2^{(t)}| + \ldots |x_n^{(t)}|$.

Observatia 8. Rationamentul utilizat în algoritmul **VPMM** ar putea fi extins si pentru celelalte valori proprii. In acest sens vom aduce unele precizari. Propozitia 4 se mentine adevarata si în situatia în care $|\lambda_1| > |\lambda_2| > |\lambda_3|$ iar vectorul initial $x^{(0)}$ nu depinde de vectorul propriu $V^{(1)}$ (în relatia (7) coeficientul α_1 este nul). Dificultatea aplicarii procedurii **VPMM** consta tocmai în alegerea unui vector initial $x^{(0)}$ de forma (7) în care $\alpha_1 = 0$. In acest context vectorul propriu $v^{(1)}$ este cunoscut.

Vom sugera un mod de solutionare a acestei situatii. Fie $y = (y_1, y_2, \ldots, y_n)$ un vector arbitrar din $\mathbf{R}^n$. Definim vectorul $x^{(0)} = y - \mu V^{(1)}$. Parametrul $\mu \in \mathbf{R}$ va fi determinat astfel încât sa eliminam din $x^{(0)}$ prezenta oricarei "influente" a vectorului propriu $V^{(1)}$. In acest sens vom cere ca $< x^{(0)}, V^{(1)} > = 0$, unde $< x, y >$ este produsul scalar al vectorilor x si y, adica $< x, y > = x_1 y_1 + x_2 y_2 + x_3 y_3 + \ldots + x_n y_n$.

Asadar $\mu = < y, V^{(1)} > / < V^{(1)}, V^{(1)} >$ si deci vectorul initial $x^{(0)}$ utilizat în procedura iterativa (8) este de forma $x^{(0)} = y - (< y, V^{(1)} > / < V^{(1)}, V^{(1)} >) V^{(1)}$.

Prin urmare, odata precizata cea mai mare valoare proprie în modul λ_1 precum si vectorul propriu corespunzator $V^{(1)}$, vom putea relua procedura **VPMM** ce este initializata cu vectorul $x^{(0)}$ determinat anterior. Se obtine astfel valoarea proprie λ_2, procedeul putând fi extins în mod analog si pentru celelalte valori proprii distincte.

Daca matricea A admite numai valori proprii reale si distincte, atunci din Propozitia 3 deducem ca vectorii proprii corespunzatori $V^{(1)}, V^{(2)}, V^{(3)}, \ldots, V^{(n)}$ definesc o baza a spatiului $\mathbf{R}^n$. Acest fapt va simplifica calculele, orice vector al spatiului $\mathbf{R}^n$ fiind o combinatie liniara a vectorilor proprii $V^{(j)}$, $1 \le j \le n$. In acest caz orice vector $x^{(0)} \in \mathbf{R}^n$ va putea fi de forma (7) fara însa a fi neaparat satisfacuta conditia suplimentara $\alpha_1 \neq 0$ ceruta de Ipoteza 2.

3. Determinarea vectorilor proprii

In cazul în care este cunoscuta o valoare proprie λ atunci vectorul propriu corespunzator $v = (v_1, v_2, v_3, \ldots, v_n)$ va fi dedus utilizând relatiile (3).

Pentru simplificarea expunerii vom presupune ca $v_n \neq 0$. Conform Corolarului 1 determinarea vectorului propriu $v = (v_1, v_2, v_3, \ldots, v_{n-1}, 1)$ se reduce la rezolvarea urmatorului sistem liniar, neomogen, de n ecuatii în necunoscutele $v_1, v_2, v_3, \ldots, v_{n-1}$,

$$\begin{aligned}
&(a_{11} - \lambda) v_1 + a_{12} v_2 + a_{13} v_3 + a_{14} v_4 + \ldots + a_{1,n-1} v_{n-1} = -a_{1n} \\
&a_{21} v_1 + (a_{22} - \lambda) v_2 + a_{23} v_3 + a_{24} v_4 + \ldots + a_{2,n-1} v_{n-1} = -a_{2n} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \qquad (12) \\
&a_{n-1,1} v_1 + a_{n-1,2} v_2 + a_{n-1,3} v_3 + \ldots + (a_{n-1,n-1} - \lambda) v_{n-1} = -a_{n-1,n}
\end{aligned}$$

$a_{n1} v_1 + a_{n2} v_2 + a_{n3} v_3 + a_{n4} v_4 + \quad + a_{n, n-1} v_{n-1} \quad = \lambda - a_{n n}$

unde valoarea proprie λ este cunoscuta.

Din Propozitia 1 avem $\det (A - \lambda E) = 0$. Prin urmare cel putin una dintre cele n ecuatii ale sistemului (12) este o combinatie liniara a celorlalte ecuatii. Asadar aflarea solutiei $(v_1, v_2, \ldots, v_{n-1})$ a sistemului (12) revine la rezolvarea unui sistem liniar clasic, de n-1 ecuatii, sistem dedus din (12) prin renuntarea la acea ecuatie m, $1 \le m \le n$, ce se exprima ca o combinatie liniara de celelalte ecuatii.

Solutionarea sistemului liniar rezultat în urma eliminarii ecuatiei m se poate efectua, de exemplu, prin metoda Gauss. Detalii suplimentare privind algoritmul Gauss sunt date în sectiunea referitoare la rezolvarea sistemelor liniare.

Practic, în sistemul (12), nu cunoastem valoarea indicelui m a ecuatiei ce urmeaza a fi eliminata. Precizam faptul ca metoda Gauss nu poate fi aplicata, determinantul sistemului astfel rezultat fiind nul, atunci când se renunta la o alta ecuatie decât aceea ce este o combinatie liniara a celorlalte ecuatii. In acest caz vom reveni la sistemul (12) optând pentru o noua ecuatie m ce nu va fi utilizata în calcule.

Daca $v_n = 0$ nu vom putea aplica Corolarul 1 astfel încât sa operam cu $v_n = 1$. In aceasta situatie locul elementului v_n va fi preluat de orice element nenul v_k, $1 \le k \le n-1$ (din punct de vedere algoritmic nu vom întâmpina dificultati, în matricea $C - \lambda E$ permutându-se coloanele n si k).

Aplicatia 2. Pentru o matrice patratica $A = (a_{ij})_{1 \le i, j \le n}$ data, programul urmator determina prin metoda Gauss solutia $(v_1, v_2, v_3, \ldots, v_{n-1})$ a sistemului liniar (12) ce are n ecuatii. Una dintre aceste ecuatii va fi în mod obligatoriu eliminata fiind o combinatie liniara a celorlalte. In ipoteza $v_n \neq 0$ s-a considerat $v_n = 1$ (Corolarul 1). Ca date de intrare, pe lânga matricea A, se va preciza valoarea proprie λ precum si numarul m al ecuatiei ce urmeaza a se elimina din (12). In cazul semnalarii faptului ca determinantul principal al sistemului rezultat dupa eliminarea ecuatiei m este nul, se va modifica valoarea indicelui m (în sistemul(12) se va renunta la o alta ecuatie).

```
REM  Aflarea vectorului propriu pentru o valoare proprie  d  data
CLS :          INPUT "n = "; n:          n1 = n - 1 :        DIM a(n, n), c(n, n), x(n), y(n1)
x(n) = 1:      PRINT "Matricea  A : "
FOR j = 1 TO n:        PRINT "linia "; j; " :  ";
FOR k = 1 TO n:    INPUT ; a(j, k):     PRINT " , ";:     NEXT k :     PRINT :     NEXT j
INPUT "Valoare proprie = "; d :          INPUT "Se renunta la ecuatia numarul "; m
REM  Initializarea matricei  C
FOR j = 1 TO n
FOR k = 1 TO n1:       c(j, k) = a(j, k):       NEXT k
c(j, n) = -a(j, n) :     c(j, j) = a(j, j) - d :     NEXT j :     c(n, n) = -c(n, n)
REM  Eliminarea ecuatiei  m
FOR k = 1 TO n :       c(m, k) = c(n, k) :       c(n, k) = k :       NEXT k
REM  Rezolvarea sistemului  C x = b  folosind eliminarea Gauss
REM  Aducerea matricei  C  la forma triunghiulara Gauss
FOR j = 1 TO n1
REM  Determinarea pozitiei  (j,s)  cu cel mai mare pivot in modul
s = j :        cm = ABS(c(j, j))
FOR k = j + 1 TO n1
IF ABS(c(j, k)) > cm THEN cm = c(j, k) :       s = k
NEXT k
```

```
REM  Permutarea coloanei j cu coloana s
IF j <> s THEN FOR t = 1 TO n :      SWAP c(t, j), c(t, s) :      NEXT t
IF c(j, j) = 0 THEN PRINT "Determinantul sistemului este nul" :      STOP
REM  Anularea elementelor c(t,j)  de sub diagonala principala
FOR t = j + 1 TO n1 :      cj = -c(t, j) / c(j, j) :      c(t, j) = 0
FOR k = j + 1 TO n :      c(t, k) = c(t, k) + cj * c(j, k) :      NEXT k
NEXT t :      NEXT j
REM  Rezolvarea sistemului superior triunghiular obtinut
IF  c(n1, n1) = 0  THEN     PRINT "Determinantul sistemului este nul" :      STOP
y(n1) = c(n1, n) / c(n1, n1)
FOR t = n1 - 1 TO 1 STEP -1
s = 0 :      FOR k = t + 1 TO n1 :      s = s + c(t, k) * y(k) :      NEXT k
y(t) = (c(t, n) - s) / c(t, t) :      NEXT t
REM  Permutarea elementelor solutiei
FOR k = 1 TO n1 :      x(c(n, k)) = y(k) :      NEXT k
PRINT "Vector propriu =  ";
FOR j = 1 TO n :      PRINT USING "####.## ,   "; x(j); :      NEXT j
```

Exemplul 4. Fie $n = 3$ si matricea $A \equiv (a_{ij})$ cu $a_{11} = 8$, $a_{12} = -7$, $a_{13} = 1$, $a_{21} = 12$, $a_{22} = -11$, $a_{23} = 1$, $a_{31} = 12$, $a_{32} = -14$, $a_{33} = 4$. Matricea A are valorile proprii $\lambda_1 = -4$, $\lambda_2 = 3$, $\lambda_3 = 2$ (Exemplul 2). In aceasta varianta, în Tabelul 2 sunt sintetizate rezultatele rularii programului precedent atunci când indicele m al ecuatiei din (12), la care se va renunta ulterior, este modificat succesiv. Vectorii proprii dedusi experimental cu ajutorul programului prezentat în Aplicatia 2 (Tabelul 2) nu difera de cei obtinuti teoretic în Exemplul 2 .

Tabelul 2. Vectorii proprii (v_1, v_2, v_3) dedusi experimental cu programul din Aplicatia 2 pentru matricea A din Exemplul 4, variind parametrii de intrare λ si m .

λ	m	Vectorul propriu (v_1, v_2, v_3)	λ	m	Vectorul propriu (v_1, v_2, v_3)
-4	1	(0.50 , 1.00 , 1.00)	-4	2	(0.50 , 1.00 , 1.00)
-4	3	deterrminant nul	3	1	deterrminant nul
3	2	(0.50 , 0.50 , 1.00)	3	3	(0.50 , 0.50 , 1.00)
2	1	(1.00 , 1.00 , 1.00)	2	2	deterrminant nul
2	3	(1.00 , 1.00 , 1.00)			

Observatia 9. In literatura de specialitate întâlnim o mare varietate de proceduri specializate pentru determinarea vectorilor si valorilor proprii ale unei matrici patratice A oarecare (Demidovich, Maron, 1981, p. 410-458). Mentionam de asemenea existenta unor algoritmi performanti pentru aflarea vectorilor proprii ai unor matrici A de forma particulara (de exemplu matrici simetrice si pozitiv definite).

FRACTII CONTINUE

Vom nota prin $F = [\, a_0 ;\ b_1 \mid a_1 , b_2 \mid a_2 , \ldots , b_n \mid a_n , \ldots ..]$ urmatoarea fractie continua (infinita) dependenta de termenii reali a_0 , b_1 , a_1 , b_2 , a_2 , ... , b_n , a_n , ...

$$F = a_0 + \cfrac{b_1}{a_1 + \cfrac{b_2}{a_2 + \cfrac{b_3}{a_3 + \cfrac{b_4}{a_4 + \ldots + \cfrac{b_n}{a_n + \cfrac{b_{n+1}}{a_{n+1} + \ldots .}}}}}} \qquad (1)$$

Fractia continua este simpla (sau standard ; notatatie prescurtata **fcs**) daca este de forma :

$$[a_0 ; a_1 , a_2 , \ldots , a_n \ldots] = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \ldots \cfrac{1}{a_n + \cfrac{1}{a_{n+1} + \ldots}}}}} \qquad (2)$$

adica $b_1 = b_2 = b_3 = \ldots = b_n = b_{n+1} = \ldots = 1$.

Exemplul 1. Vom rescrie fractia ordinara p_0 / p_1 în forma **fcs** (fractie continua simpla) unde numaratorul p_0 si numitorul p_1 sunt numere naturale, $p_1 \neq 0$, .

Prin aplicarea succesiva a algoritmului lui Euclid rezulta egalitatile :

$$\begin{aligned}
p_0 &= a_0 \cdot p_1 + p_2 \quad , \quad 0 \le p_2 < p_1 \\
p_1 &= a_1 \cdot p_2 + p_3 \quad , \quad 0 \le p_3 < p_2 \\
p_2 &= a_2 \cdot p_3 + p_4 \quad , \quad 0 \le p_4 < p_3 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
p_n &= a_n \cdot p_{n+1} + p_{n+2} \quad , \quad 0 \le p_{n+2} < p_{n+1} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots
\end{aligned} \qquad (3)$$

Tinând seama de aceste relatiile (3) fractia ordinara p_0 / p_1 se rescrie în forma echivalenta :

$$\frac{p_0}{p_1} = \frac{a_0 \cdot p_1 + p_2}{p_1} = a_0 + \frac{1}{\frac{p_1}{p_2}} = a_0 + \frac{1}{\frac{a_1 \cdot p_2 + p_3}{p_2}} = a_0 + \frac{1}{a_1 + \frac{1}{\frac{p_2}{p_3}}} = a_0 + \frac{1}{a_1 + \frac{1}{\frac{a_2 \cdot p_3 + p_4}{p_3}}} =$$

$$= a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\frac{p_3}{p_4}}}} = \qquad = [a_0 ; a_1 , a_2 , a_3 , \ldots , a_n , \ldots]$$

Observatia 1. Orice fractie continua simpla $[\, a_0 ;\ a_1 , a_2 , a_3 , \ldots , a_n , \ldots \,]$ având drept valoare o fractie ordinara (numar rational) are un numar finit de termeni.

In adevar, deoarece termenii $p_1, p_2, p_3, \dots, p_n, \dots$ sunt numere naturale si cum $p_1 > p_2 > p_3 > \dots > p_{n-1} > p_n > \dots$ rezulta o valoare $n \in N$ astfel încât $p_{n+2} = 0$. In aceasta situatie fractia continua simpla (infinita) $[a_0 ; a_1, a_2, a_3, \dots, a_n, \dots]$ se rescrie de fapt sub forma $[a_0 ; a_1, a_2, a_3, \dots, a_{n-1}, a_n]$, unde toti coeficientii a_j, $a_j > n$, sunt nuli.

Aplicatia 1. Urmatorul program BASIC calculeaza, pentru o fractie ordinara p_0 / p_1 precizata prin numaratorul si numitorul ei p_0 , $p_1 \in N$, coeficientii $a_0, a_1, a_2, \dots, a_n$ ai dezvoltarii lui p_0 / p_1 în fractie continua simpla (forma **fcs** (2)).

Pentru a economisi memorie calculator programul va evita retinerea, într-un vector auxiliar p , a tuturor rezultatelor intermediare $p_2, p_3, p_4, \dots, p_n, p_{n+1}$. Algoritmul descris se opreste la prima valoare p_j nula. Pe masura ce sunt obtinute valorile variabilelor consecutive p_j si p_{j+1} este estimat si coeficientul a_j. In aceasta situatie coeficientii a_j ai dezvoltarii în fractie continua simpla a fractiei ordinare p_0 / p_1 vor putea fi listati succesiv, în momentul deducerii lor. In esenta procedura descrisa foloseste numai valorile lui p_j si p_{j+1} pentru determinarea lui p_{j+2}.

In programul BASIC propus variabilele numerice p0% , p1% , p2% , a% utilizate sunt de tip întreg.

```
REM  Dezvoltarea unei fractii ordinare in fractie continua simpla
INPUT "Fractia ordinara  p0 / p1  = "; p0%, p1%
PRINT "Coeficientii  fcs : "; :
DO :      a% = INT(p0% / p1%) :      PRINT a%; "      "; :      p2% = p0% - p1% * a%
p0% = p1% :      p1% = p2% :      LOOP WHILE p2% <> 0 :      PRINT
```

Programul descris este mai general deoarece determina termenii a_j, $0 \le j \le n$, ai dezvoltarii în fractie continua simpla si pentru fractii ordinare p_0 / p_1 negative (unde, de exemplu, $p_0 \in Z$ iar $p_1 \in N$). Acest fapt este confirmat si de urmatoarele rezultate de test :

```
Fractia ordinara  p0 / p1  = ?  23  ,  61
Coeficientii  fcs :   0      2      1      1      1      7
Fractia ordinara  p0 / p1  = ? -23  ,  61
Coeficientii  fcs :  -1      1      1      1      1      1      7
Fractia ordinara  p0 / p1  = ?  61  ,  23
Coeficientii  fcs :   2      1      1      1      7
Fractia ordinara  p0 / p1  = ? -61  ,  23
Coeficientii  fcs :  -3      2      1      7
```

Asadar prin rularea programului anterior au fost obtinute urmatoarele dezvoltari în forma **fcs** :

$$\frac{23}{61} = 0 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{7}}}}} \qquad \frac{-23}{61} = -1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{7}}}}}}$$

$$\frac{61}{23} = 2 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{7}}}} \qquad \frac{-61}{23} = -3 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{7}}}$$

Observatia 2. In vederea unei utilizari eficiente a memoria interne a calculatorului, valorile p_j , p_{j+1} , p_{j+2} obtinute succesiv au fost memorate în variabilele de tip real p_0 , p_1 , p_2 si nu într-un vector p de dimensiune variabila. In acest context precizam faptul ca programul prezentat nu va retine în final valorile numaratorului si numitorului fractiei ordinare p_0 / p_1 studiate. Valorile initiale ale variabilelor p_0 si p_1 sunt modificate succesiv pe parcursul desfasurarii algoritmului de calcul.

Orice numar real, nu neaparat numar rational, poate fi dezvoltat în fractie continua. Vom sugera pe un exemplu procedura de dezvoltare în fractie continua a unui numar irational.

Exemplul 2. Sa determinam termenii a_0 , a_1 , a_2 , ... , a_n , ... ai dezvoltarii **fcs** (fractie continua standard) a numarului irational $39^{1/2}$. Algoritmul descris în continuare va putea fi cu usurinta adaptat si pentru precizarea coeficientilor din dezvoltarea **fcs** a altor numere irationale.

Tinând seama de faptul ca $6 < 39^{1/2} < 7$ obtinem succesiv :

$39^{1/2} = 6 + (39^{1/2} - 6) = 6 + 1 / a_1$

$a_1 = 1 / (39^{1/2} - 6) = (39^{1/2} + 6) / 3 = 4 + (39^{1/2} - 6) / 3 = 4 + 1 / a_2$

$a_2 = 3 / (39^{1/2} - 6) = 3 . (39^{1/2} + 6) / 3 = 12 + (39^{1/2} - 6) = 12 + 1 / a_3$

$a_3 = 1 / (39^{1/2} - 6) = a_1$

$a_4 = a_2$

si în general $a_{n+2} = a_n$, $\forall\, n \in N$, $n \geq 1$.

Am obtinut asadar urmatoarea dezvoltarea în fractie continua standard (**fcs**) .

$$\sqrt{39} = \left[6;4,12,4,12,4,12,\ldots \right] = 6 + \cfrac{1}{4 + \cfrac{1}{12 + \cfrac{1}{4 + \cfrac{1}{12 + \cfrac{1}{4 + \ldots}}}}}$$

Intr-un interval de timp finit nu este posibil de a evalua, chiar cu ajutorul calculatorului, valoarea unei fractii continue (standard sau nu) ce are un numar infinit de termeni. O astfel de activitate presupune efectuarea unui numar nelimitat de operatii. Asadar vom accepta un compromis, aproximând o fractie continua infinita $F \equiv [a_0 ; b_1 | a_1 , b_2 | a_2 , \ldots , b_n | a_n , \quad]$ prin redusa sa $P_n / Q_n = [a_0 ; b_1 | a_1 , b_2 | a_2 , \ldots , b_n | a_n]$, de ordin n .

In continuare suntem interesati în a stabili conditiile în care sirul reduselor $\{ P_n / Q_n \}_{n \in N}$ de ordin n ale unei fractii continue F este convergent la o valoare α . In caz afirmativ limita finita α astfel rezultata va defini valoarea fractiei continue infinite F .

Nu este prea comod de a proiecta un algoritm de calcul care sa estimeze valoarea redusei P_n / Q_n pornind de la definitia sa, $P_n / Q_n = [a_0 ; b_1 | a_1 , b_2 | a_2 , \ldots , b_n | a_n]$. O solutionare eleganta a acestei probleme se va baza pe faptul ca redusele unei fractii continue F pot fi obtinute succesiv utilizându-se o relatie de recurenta. Având în vedere acest lucru vom adopta o convenție ce ne va permite calculul iterativ al reduselor fractiei continue (infinite) F .

Conventia 1. $P_{-1} = 1$, $Q_{-1} = 0$, $P_0 = a_0$, $Q_0 = 1$.

Urmând definitia redusei de ordin n avem o dezvoltare a sa sub forma de fractii suprapuse

$$\frac{P_n}{Q_n} = \left[a_0 ; \frac{b_1}{a_1}, \frac{b_2}{a_2}, \ldots, \frac{b_n}{a_n} \right] = a_0 + \cfrac{b_1}{a_1 + \cfrac{b_2}{a_2 + \cfrac{b_3}{a_3 + \cfrac{b_4}{a_4 + \cfrac{\ddots}{\ddots \cfrac{b_{n-1}}{a_{n-1} + \cfrac{b_n}{a_n}}}}}}} \qquad (4)$$

Propozitia 1. Respectând Conventia 1, pentru orice $n \geq 1$ sunt adevarate urmatoarele formule recurente

$$P_n = a_n P_{n-1} + b_n P_{n-2} \qquad Q_n = a_n Q_{n-1} + b_n Q_{n-2} \qquad (5)$$

Vom utiliza metoda inductiei matematice pentru a demonstra veridicitatea identitatilor (5).

Cum $P_1 / Q_1 = [a_0 ; b_1 | a_1] = a_0 + b_1 / a_1 = (a_0 a_1 + b_1) / a_1$ si tinând seama de Conventia 1 rezulta

$$P_1 = a_0 a_1 + b_1 = a_1 P_0 + b_1 P_{-1} \qquad Q_1 = a_1 = a_1 Q_0 + b_1 Q_{-1}$$

egalitatile (5) fiind astfel adevarate pentru $n = 1$ (pasul initial în procedura de inductie).

In continuare vom justifica pasul inductiv de trecere de la rangul n-1 la rangul n . Astfel, vom presupune ca formulele (5) sunt adevarate pentru redusele P_k / Q_k , de rang k , $1 \leq k \leq n-1$, ale oricarei fractii continue (infinite) , urmând sa aratam ca aceste formule continua sa ramâna adevarate si pentru redusa P_n / Q_n de ordin n ale fractiei continue F .

Prin efectuarea unor operatii aritmetice simple în redusa P_n / Q_n , vom rescrie expresia $b_{n-1} / (a_{n-1} + b_n / a_n)$ sub forma unei fractii ordinare $b_{n-1}' / a_{n-1}' = (b_{n-1} a_n) / (a_{n-1} a_n + b_n)$. Asadar redusa P_n / Q_n de ordin n data de expresia (4) ar putea fi interpretata si ca o redusa $P_{n-1}' / Q_{n-1}' = [a_0 ; a_1' | b_1' , a_2' | b_2' , \ldots , b_{n-1}' | a_{n-1}']$ de ordin n-1 , dar pentru o alta fractie continua F' , anume $F' \equiv [a_0 ; a_1' | b_1' , a_2' | b_2' , \ldots , b_n' | a_n'' , \ldots]$, unde $a_j' = a_j$, $0 \leq j \leq n-2$, $b_k' = b_k$, $1 \leq k \leq n-2$, $a_{n-1}' = a_{n-1} a_n + b_n$, $b_{n-1}' = b_{n-1} a_n$.

In mod evident sunt adevarate egalitatile $P_j' = P_j$, $Q_j' = Q_j'$, pentru orice $1 \leq j \leq n-2$.

Tinând seama de presupunerea facuta în cadrul pasului de inductie, avem succesiv :

$$P_n = P_{n-1}' = a_{n-1}' P_{n-2}' + b_{n-1}' P_{n-3}' = (a_{n-1} a_n + b_n) P_{n-2} + b_{n-1} a_n P_{n-3} =$$
$$= a_n (a_{n-1} P_{n-2} + b_{n-1} P_{n-3}) + b_n P_{n-2} = a_n P_{n-1} + b_n P_{n-2}$$

$$P_n = Q_{n-1}' = a_{n-1}' Q_{n-2}' + b_{n-1}' Q_{n-3}' = (a_{n-1} a_n + b_n) Q_{n-2} + b_{n-1} a_n Q_{n-3} =$$
$$= a_n (a_{n-1} Q_{n-2} + b_{n-1} Q_{n-3}) + b_n Q_{n-2} = a_n Q_{n-1} + b_n Q_{n-2}$$

fapt ce încheie demonstratia prin inductie matematica a egalitatilor (5).

Caz particular. In cazul unei fractii continue infinite $F \equiv [a_0 ; a_1 , a_2 , a_3 , \ldots , a_n , \ldots]$ ce este data în forma standard, redusa sa P_n / Q_n de ordin n este definita recurent de formulele

$$P_n = a_n P_{n-1} + P_{n-2} \qquad Q_n = a_n Q_{n-1} + Q_{n-2} \qquad (6)$$

cu respectarea Conventiei 1.

In cele ce urmeaza suntem interesati în a preciza conditiile în care sirul $\{ P_n / Q_n \}_n$ al reduselor unei fractii continue F este convergent la o valoare c . In caz afirmativ cantitatea c va defini valoarea fractiei continue infinite respective.

In acest context vom studia diferenta dintre oricare doua reduse succesive

Propozitia 2. Pentru orice $n \geq 1$ avem :

$$\frac{P_n}{Q_n} - \frac{P_{n-1}}{Q_{n-1}} = (-1)^{n-1} \frac{b_1 b_2 b_3 \ldots b_{n-1} b_n}{Q_{n-1} Q_n} \tag{7}$$

Pentru demonstrarea acestei propozitii este suficient sa verificam egalitatea

$$P_n Q_{n-1} - P_{n-1} Q_n = (-1)^{n-1} b_1 b_2 b_3 \ldots b_{n-1} b_n \tag{8}$$

Tinând seama de Conventia 1 rezulta ca egalitatea (8) este adevarata pentru $n = 1$ deoarece

$$P_1 Q_0 - P_0 Q_1 = (a_0 a_1 + b_1) \cdot 1 - a_0 \cdot a_1 = b_1 = (-1)^{1-1} \cdot b_1$$

Presupunând adevarata relatia (8) pentru valoarea n si utilizând (5) , deducem

$$P_{n+1} Q_n - P_n Q_{n+1} = (a_{n+1} P_n + b_{n+1} P_{n-1}) Q_n - P_n (a_{n+1} Q_n + b_{n+1} Q_{n-1}) =$$

$$= b_{n+1} (P_{n-1} Q_n - P_n Q_{n-1}) = - b_{n+1} \cdot [(-1)^{n-1} b_1 b_2 b_3 \ldots b_{n-1} b_n] = (-1)^n b_1 b_2 b_3 \ldots b_n b_{n+1}$$

afirmatie cu care se termina demonstratia prin inductie a identitatii (8) .

Sa estimam diferenta dintre doua reduse succesive pare, respectiv impare.

Propozitia 3. Pentru orice $n \geq 1$ avem

$$\frac{P_n}{Q_n} - \frac{P_{n-2}}{Q_{n-2}} = (-1)^n \frac{b_1 b_2 b_3 \ldots b_{n-1} a_n}{Q_{n-2} Q_n} \tag{9}$$

Justificarea acestei propozitii este analoga celei precedente, demonstratia reducându-se la verificarea prin inductie a egalitatii

$$P_n Q_{n-2} - P_{n-2} Q_n = (-1)^n b_1 b_2 b_3 \ldots b_{n-1} a_n \tag{10}$$

Utilizând Conventia 1 rezulta

$$P_1 Q_{-1} - P_{-1} Q_1 = (a_0 a_1 + b_1) \cdot 0 - 1 \cdot a_1 = b_1 = (-1)^{1-1} \cdot b_1$$

relatia (10) fiind astfel adevarata pentru $n = 1$.

In ipoteza în care egalitatea (10) este adevarata pentru o valoare fixata a indicelui n , utilizând relatiile (5) si (8) deducem succesiv

$$P_{n+1} Q_{n-1} - P_{n-1} Q_{n+1} = (a_{n+1} P_n + b_{n+1} P_{n-1}) Q_{n-1} - P_{n-1} (a_{n+1} Q_n + b_{n+1} Q_{n-1}) =$$

$$= a_{n+1} (P_n Q_{n-1} - P_{n-1} Q_n) = a_{n+1} \cdot [(-1)^{n-1} b_1 b_2 b_3 \ldots b_{n-1} b_n] = (-1)^{n+1} b_1 b_2 b_3 \ldots b_n a_{n+1}$$

Prin urmare etapa inductiva de trecere de la n la $n+1$ este realizata, identitatea (10) fiind astfel adevarata pentru orice $n \in \mathbf{N}$.

Propozitia 4. Daca exista $k \in \mathbf{N}$ astfel încât $b_k = 0$ atunci

$$\frac{P_{k-1}}{Q_{k-1}} = \frac{P_k}{Q_k} = \frac{P_{k+1}}{Q_{k+1}} = \ldots = \frac{P_{k+j}}{Q_{k+j+1}} = \ldots \tag{11}$$

In adevar, aplicând Propozitia 2 pentru $b_k = 0$, din formula formula (7) rezulta

$$P_n / Q_n - P_{n-1} / Q_{n-1} = 0 \quad , \qquad \forall\ n \geq k$$

Asadar se verifica egalitatea (11) , valorile tuturor reduselor P_n / Q_n mentinându-se neschimbate pentru orice $n \geq k - 1$.

In continuare vom studia convergenta sirului $\{ P_n / Q_n \}_n$ al reduselor unei fractii continue F într-o situatie particulara, anume atunci când coeficientii fractiei continue F sunt nenegativi.

Ipoteza 1. $a_j \geq 0$, $b_j \geq 0$, $\forall\ j \in \mathbf{N}$.

Teorema 1. In conditiile precizate de Ipoteza 1 redusele $R_n = P_n / Q_n$ ale fractiei continue F satisfac inegalitatile

$$R_0 \leq R_2 \leq R_4 \leq \ldots \leq R_{2m} \leq \ldots \leq R_{2n+1} \leq R_{2n-1} \leq \ldots \leq R_3 \leq R_1 \leq R_{-1} \tag{12}$$

Sa justificam inductiv inegalitatile (12) Tinând seama de Ipoteza 1, Conventia 1 si formulele recurente (5) , deducem succesiv ca $P_n \geq 0$ si $Q_n \geq 0$, $\forall\ n \in \mathbf{N}$.

Din Propozitia 3 , formula (9), rezulta ca semnul expresiei $P_n / Q_n - P_{n-2} / Q_{n-2}$ este $(-1)^n$ si deci $P_{2m} / Q_{2m} \geq P_{2m-2} / Q_{2m-2}$, $P_{2m+1} / Q_{2m+1} \leq P_{2m-1} / Q_{2m-1}$, $\forall\ m \in \mathbf{N}$. Din Propozitia 2 (formula (7)), deducem ca inegalitatile $P_{2m} / Q_{2m} \leq P_{2m+1} / Q_{2m+1}$ sunt adevarate pentru orice $m \in \mathbf{N}$.

In concluzie pentru orice $m, k \in \mathbf{N}$, considerând de exemplu $m \leq k$, avem succesiv $P_{2m} / Q_{2m} \leq P_{2k} / Q_{2m} \leq P_{2m} / Q_{2m}$ fapt ce implica inegalitatile (12).

Deoarece subsirul definit de redusele pare, respectiv subsirul reduselor impare, este monoton si marginit rezulta ca el este si convergent. Asadar sirurile $\{ P_{2m} / Q_{2m} \}_{m \in N}$, $\{ P_{2m+1} / Q_{2m+1} \}_{m \in N}$ converg la valorile α , respectiv β , unde $\alpha \leq \beta$. In cazul în care $\alpha = \beta$ vom defini valoarea fractiei continue F ca fiind egala cu α . Valoarea lui F nu este precizata în situatia $\alpha \neq \beta$.

Nu toate multimile $\{ a_n \}_n$, $\{ b_n \}_n$, de numere reale, nenegative, pot caracteriza o fractie continua, infinita, $F \equiv [a_0 ; b_1 | a_1 , b_2 | a_2 , \dots , b_n | a_n , \dots]$ convergenta (sirul reduselor lui F sa fie convergent). In acest sens sugeram urmatorul exemplu :

Exemplul 3. Fie fractia continua $F \equiv [a_0 ; b_1 | a_1 , b_2 | a_2 , \dots , b_n | a_n , \dots]$ unde termenii sai a_j , b_j sunt definiti de formulele :

$$a_0 = 0 ; \quad a_{2m-1} = 1 / [m^2 + m - 1] ; \quad a_{2m} = 1 / [m (m + 2)] ; \qquad (13)$$

$$b_{2m-1} = [m (m + 2)] / [m^2 + m - 1] ; \quad b_{2m} = [m^2 + 3 m + 1] / [m (m + 2)] ; \quad m \geq 1$$

In aceste conditii vom demonstra inductiv ca redusele P_n / Q_n de ordin n sunt date de expresiile :

$$P_{2m-1} = 2 m + 1 ; \quad Q_{2m-1} = m ; \quad P_{2m} = m ; \quad Q_{2m} = m + 1 ; \quad m \geq 1 \qquad (14)$$

Sa verificam în primul rând ca egalitatile (14) sunt adevarate pentru $m = 1$, adica : $P_1 = 3$, $Q_1 = 1$, $P_2 = 1$, $Q_2 = 2$. In adevar, tinând seama de formulele (13) ce ne dau expresiile coeficientilor a_n si b_n , obtinem

$P_1 = b_1 = 3 ; \quad Q_1 = a_1 = 1 ; \quad P_2 = b_1 a_2 = 3 (1 / 3) = 1 ;$

$Q_2 = a_1 a_2 + b_2 = 1 (1 / 3) + (5 / 3) = 2$.

etapa initiala a procedurii de inductie fiind astfel satisfacuta.

Conform procedurii inductiei matematice vom presupune ca formulele (14) sunt adevarate pentru o valoare fixata $m \in \mathbf{N}$,urmând sa aratam ca egalitatile (14) se mentin adevarate si pentru urmatoarea valoare $m + 1$, adica :

$$P_{2m+1} = 2 m + 3 ; \quad Q_{2m+1} = m + 1 ; \quad P_{2m+2} = m + 1 ; \quad Q_{2m+2} = m + 2 ; \qquad (15)$$

Aplicând formulele recurente (5) obtinem :

$$P_{2m+1} = a_{2m+1} P_{2m} + b_{2m+1} P_{2m-1} = m / (m^2 + 3 m + 1) +$$

$$+ (m + 1) (m + 3) (2 m + 1) / (m^2 + 3 m + 1) = (2 m^3 + 9 m^2 + 11 m + 3) / (m^2 + 3 m + 1) =$$

$$= 2 m + 3$$

$$Q_{2m+1} = a_{2m+1} Q_{2m} + b_{2m+1} Q_{2m-1} = (m + 1) / (m^2 + 3 m + 1) +$$

$$+ m (m + 1) (m + 3) / (m^2 + 3 m + 1) = (m^3 + 4 m^2 + 4 m + 1) / (m^2 + 3 m + 1) = m + 1$$

$$P_{2m+2} = a_{2m+2} P_{2m+1} + b_{2m+2} P_{2m} = [2 m + 3] / [(m + 1) (m + 3)] +$$

$$+ [m (m^2 + 5 m + 5)] / [(m + 1) (m + 3)] = [m^3 + 2 m^2] / [(m + 1) (m + 3)] = m + 1$$

$$Q_{2m+2} = a_{2m+2} Q_{2m+1} + b_{2m+2} Q_{2m} = (m + 1) / [(m + 1) (m + 3)] +$$

$+ (m^2 + 5 m + 5) (m + 1) / [(m + 1) (m + 3)] = (m^2 + 5 m + 6) / (m + 3) = m + 2$

Asadar formulele (14) sunt adevarate pentru orice $m \geq 1$.

Printr-un calcul direct deducem ca sirul reduselor impare R_{2m+1} este un sir descrescator , unde $R_{2m+1} = P_{2m+1} / Q_{2m+1} = (2 m + 3) / (m + 1)$ tinde la valoarea 2 . Analog, sirul reduselor pare R_{2m} este un sir crescator, iar $R_{2m} = P_{2m} / Q_{2m} = m / (m + 1)$ tinde la valoarea 1 si $1 \neq 2$.

Prin urmare sirul reduselor $\{ R_n \}_n$ nu converge. Asadar valoarea fractiei continue infinite $F \equiv [a_0 ; b_1 | a_1 , b_2 | a_2 , \ldots , b_n | a_n , \ldots]$ unde termenii a_n , b_n au expresiile (13), nu poate fi precizata (cele doua subsiruri de reduse pare sau impare converg la valori distincte 1 , respectiv 2).

Aplicatia 2. Bazat pe relatiile de recurenta (5), programul urmator editeaza valorile a 38 de reduse consecutive $R_n = P_n / Q_n$ ale caror coeficienti sunt definiti de formulele (13) .

```
REM  Editarea valorilor unor reduse consecutive ale fractiei continue  F
p1 = 3:    q1 = 1:    p2 = 1:    q2 = 2
FOR n = 3 TO 38 :          m = FIX(n / 2)
IF n = (2 * m) THEN a = 1 / (m * (m + 2)) :       b = (m * m + 3 * m + 1) / (m * (m + 2))
IF n = (2 * m + 1) THEN a = 1 / (m * m + 3 * m + 1) :       b = a * (m + 1) * (m + 3)
p3 = a * p2 + b * p1 :     q3 = a * q2 + b * q1 :      PRINT USING "R(##) = #.## ;  "; n; p3 / q3;
p1 = p2 :        p2 = p3 :        q1 = q2 :        q2 = q3 :        NEXT n
```

In urma rularii prezentului program constatam în mod experimental convergenta reduselor pare (impare) la valoarea 1 (respectiv valoarea 2) .

```
R( 3) = 2.50 ;  R( 4) = 0.67 ;  R( 5) = 2.33 ;  R( 6) = 0.75 ;  R( 7) = 2.25 ;  R( 8) = 0.80 ;
R( 9) = 2.20 ;  R(10) = 0.83 ;  R(11) = 2.17 ;  R(12) = 0.86 ;  R(13) = 2.14 ;  R(14) = 0.88 ;
R(15) = 2.13 ;  R(16) = 0.89 ;  R(17) = 2.11 ;  R(18) = 0.90 ;  R(19) = 2.10 ;  R(20) = 0.91 ;
R(21) = 2.09 ;  R(22) = 0.92 ;  R(23) = 2.08 ;  R(24) = 0.92 ;  R(25) = 2.08 ;  R(26) = 0.93 ;
R(27) = 2.07 ;  R(28) = 0.93 ;  R(29) = 2.07 ;  R(30) = 0.94 ;  R(31) = 2.06 ;  R(32) = 0.94 ;
R(33) = 2.06 ;  R(34) = 0.94 ;  R(35) = 2.06 ;  R(36) = 0.95 ;  R(37) = 2.05 ;  R(38) = 0.95 ;
```

La implementarea acestui program am preferat ca valorile reduselor succesive P_n / Q_n sa fie retinute în variabilele reale simple P1 , P2 , P3 , Q1 , Q2 , Q3 si nu definind vectori P , Q de dimensiune variabila. In acest mod memoria calculator utilizata de program este independenta de numarul reduselor ce urmeaza a fi calculate.

In continuare vom stabili o conditie suficienta pentru a asigura convergenta sirului reduselor $\{ P_n / Q_n \}_n$. In acest sens vom extinde Ipoteza 1 impunând noi restrictii pentru termenii (pozitivi) ai fractiei continue F . Sirul reduselor unei fractii continue poate fi însa convergent si într-un context mai general, de exemplu în cazul în care termeii a_n , b_n ai fractiei F iau si valori negative (a se vedea în acest sens lucrarea lui Demidovich si Maron : Computational mathematics. MIR Publishers, Moscow, 1981, pag. 71).

Ipoteza 2. $a_j \geq \lambda > 0 , \ 0 \leq b_j \leq a_j , \ \forall j \in \mathbf{N}$.

Teorema 2. In conditiile Ipotezei 2 sirul reduselor $\{ P_n / Q_n \}_n$ este convergent.

Demonstratie. Tinând seama de Teorema 1, ne ramâne sa aratam ca limita subsirului reduselor pare coincide cu limita subsirului reduselor impare, adica diferenta $P_{2m+1} / Q_{2m+1} - P_{2m} / Q_{2m}$ poate fi facuta oricât de mica pentru indici n ce depasesc un anumit prag.

Vom minora valoarea produsului $Q_{2m} Q_{2m+1}$. Utilizând Ipoteza 2 obtinem relatiile

$$Q_n = a_n Q_{n-1} + b_n Q_{n-2} \geq b_n (Q_{n-1} + Q_{n-2}) = b_n (a_{n-1} Q_{n-2} + b_{n-1} Q_{n-3} + Q_{n-2}) \geq$$
$$\geq b_n (a_{n-1} Q_{n-2} + Q_{n-2}) \geq b_n (1 + \lambda) Q_{n-2}$$

ce ne conduc la inegalitatile

$$Q_{2m+1} Q_{2m} \geq b_{2m+1} b_{2m} (1 + \lambda)^2 \geq b_{2m+1} b_{2m} b_{2m-1} b_{2m-2} (1 + \lambda)^4 \geq \ldots$$
$$\geq b_{2m+1} b_{2m} b_{2m-1} b_{2m-2} \ldots b_5 b_4 b_3 b_2 (1 + \lambda)^{2m} \qquad (16)$$

Aplicând Propozitia 2 (formula (7)) si folosind inegalitatea (16), vom estima marimea diferentei dintre doua reduse consecutive. Avem succesiv

$$0 \leq P_{2m+1} / Q_{2m+1} - P_{2m} / Q_{2m} = b_1 b_2 b_3 \ldots b_{2m} b_{2m+1} / (Q_{2m+1} Q_{2m}) \leq$$
$$\leq b_1 b_2 b_3 \ldots b_{2m} b_{2m+1} / [b_2 b_3 \ldots b_{2m} b_{2m+1} (1 + \lambda)^{2m}] \leq b_1 / (1 + \lambda)^{2m} \qquad (17)$$

Din inegalitatea (17), cum $\lambda > 0$ concluzionam ca atât subsirul reduselor pare si cel al reduselor impare au o limita comuna ce defineste de fapt valoarea fractiei continue infinite F .

Observatia 3. Sirul reduselor $\{ P_n / Q_n \}_n$ ale fractiei continue F poate converge si în cazul în care nu sunt satisfacute conditiile Ipotezei 2 . Aceste conditii suficiente pentru asigurarea convergentei nu sunt neaparat si necesare. In sprijinul acestei afirmatii vom analiza exemplul urmator.

Exemplul 4. Fie fractia continua $F_1 \equiv [a_0 ; b_1 | a_1 , b_2 | a_2 , \ldots , b_n | a_n , \ldots]$ în care $a_0 = a_1 = a_2 = a_3 = \ldots = 2$ si $b_1 = b_2 = b_3 = \ldots = 3$. In acest caz nu sunt verificate conditiile Teoremei 2 deoarece $b_n = 3 > 2 = a_n$ pentru orice $n \in N$. Tinând seama de Conventia 1 si de relatiile recurente (5) , se verifica imediat prin inductie matematica ca redusa P_n / Q_n de ordinul n a fractiei continue F_1 este data de $P_n = [3^{n+2} + (-1)^{n+1}] / 4$ si $Q_n = [3^{n+1} + (-1)^n] / 4$.

Cunoscând astfel valorile expresiilor P_n , Q_n deducem imediat ca sirul reduselor $\{ P_n / Q_n \}_n$ tinde la valoarea $c = 3$. Interpretând valoarea fractiei continue (infinite) F_1 ca limita a sirului reduselor sale deducem ca $F_1 = 3$.

Aplicatia 3. Vom confirma experimental faptul ca $F_1 = 3$ (Exemplul 4).

Urmatorul program BASIC editeaza valorile reduselor $R_n = P_n / Q_n$, unde termenii P_n si Q_n sunt obtinuti succesiv aplicând formulele (5) cu respectarea Conventiei 1, considerând $a_n = a = 2$ si $b_n = b = 3$, $\forall n \geq 1$.

```
REM     Calculul reduselor Pk / Qk de ordin k ale unei fractii continue F
REM     F = [ a0 ; a1 / b1 , a2 / b2 , ... , an / bn , ... ]
n = 15        ' P1 / Q1 , P2 / Q2 , P3 / Q3  sunt valorile a trei reduse consecutive
REM  Initializarea procesului recurent de calcul succesiv al reduselor
p1 = 1 :        p2 = 2 :        q1 = 0 :        q2 = 1 :        a = 2 :        b = 3
REM  Editarea valorilor consecutive a  n  reduse
FOR k = 1 TO n :        p3 = a * p2 + b * p1 :        q3 = a * q2 + b * q1
PRINT USING "R(### ) = ##.#####  ;   "; k; p3 / q3;
p1 = p2 :       q1 = q2 :       p2 = p3 :       q2 = q3 :       NEXT k
```

In adevar, se constata experimental ca prin rularea acestui program sirul reduselor R_k tinde la valoarea 3 în mai putin de $n = 15$ pasi (acceptând o eroare de 10^{-5}) fapt ce confirma rezultatul teoretic mentionat în Exemplul 4 , anume $F_1 = 3$.

```
R( 1 ) = 3.50000 ;    R( 2 ) = 2.85714 ;    R( 3 ) = 3.05000 ;    R( 4 ) = 2.98361 ;
```

$R(5) = 3.00549$; $R(6) = 2.99817$; $R(7) = 3.00061$; $R(8) = 2.99980$;
$R(9) = 3.00007$; $R(10) = 2.99998$; $R(11) = 3.00001$; $R(12) = 3.00000$;
$R(13) = 3.00000$; $R(14) = 3.00000$; $R(15) = 3.00000$;

Convergenta sirului $\{ P_n / Q_n \}_n$ al reduselor unei fractii continue este însa asigurata si daca sunt satisfacute inegalitatile $0 \le b_n \le a_n$, $a_n \ge \lambda > 0$, $\forall\ n \ge n_0$, conditii ce sunt mai putin restrictive decât acelea precizate în Teorema 2 (unde $n_0 = 1$). Asadar

Ipoteza 3. Fie $s \in \mathbf{N}$ astfel încât $a_j \ge \lambda > 0$, $0 \le b_j \le a_j$, $\forall\ j \ge s$.

Teorema 3. In conditiile Ipotezei 3 sirul reduselor $\{ P_n / Q_n \}_n$ este convergent.

Demonstratie. Fie fractiile continue $F \equiv [\, a_0 ;\ b_1 \mid a_1 ,\ b_2 \mid a_2 , \ldots , b_s \mid a_s , \ldots , b_{s+k} \mid a_{s+k} , \ldots \,]$, $F^{(1)} \equiv [\, 0 ;\ b_s \mid a_s ,\ b_{s+1} \mid a_{s+1} , \ldots , b_{s+k} \mid a_{s+k} , \ldots \,]$ ale caror reduse sunt P_n / Q_n, respectiv $P_n^{(1)} / Q_n^{(1)}$. Atunci redusa P_{s+k} / Q_{s+k} de ordin $s + k$ a fractiei continue F,

$$\frac{P_{s+k}}{Q_{s+k}} = a_0 + \cfrac{b_1}{a_1 + \cfrac{b_2}{a_2 + \cfrac{b_3}{a_3 + \ldots + \cfrac{b_{s+k-1}}{a_{s+k-1} + \cfrac{b_{s+k}}{a_{s+k}}}}}}$$

se poate exprima în functie de redusa $P_{k+1}^{(1)} / Q_{k+1}^{(1)}$ de ordin $k + 1$ a fractiei $F^{(1)}$, adica

$$\frac{P_{s+k}}{Q_{s+k}} = a_0 + \cfrac{b_1}{a_1 + \cfrac{b_2}{a_2 + \cfrac{b_3}{a_3 + \ldots + \cfrac{b_{s-1}}{a_{s-1} + \cfrac{P_{k+1}^{(1)}}{Q_{k+1}^{(1)}}}}}} \qquad (18)$$

Tinând seama de Ipoteza 3 si aplicând Teorema 2 pentru fractia continua $F^{(1)}$ deducem ca sirul reduselor $\{ P_k^{(1)} / Q_k^{(1)} \}_k$ este convergent. Conform formulei (18) acest fapt va antrena convergenta reduselor P_{s+k} / Q_{s+k} ale fractiei continue F, Teorema 3 fiind astfel demonstrata.

Observatia 4. Orice numar real r este caracterizat de sirul, finit sau infinit, al cifrelor sale c_j într-o baza de numeratie precizata. Considerând $r \in [0 , 1]$ si baza de numeratie 10 obtinem reprezentarea zecimala a numarului r, $r = 0{,}c_1 c_2 c_3 c_4 \ldots c_{n-1} c_n c_{n+1} \ldots$, cu cifrele c_j, $0 \le c_j \le 9$, $j = 1, 2, 3, \ldots$. O aproximare a numarului r este $r_n = 0{,}c_1 c_2 c_3 c_4 \ldots c_{n-1} c_n$, valoare rezultata prin pastrarea numai a primelor n zecimale ale numarului real r . In aceasta situatie avem eroarea de aproximare $\varepsilon = | r - r_n | < 10^{-n}$, precizia fiind cu atât mai buna cu cât se pastreaza mai multe zecimale. Practic nu se poate determina valoarea exacta a numarului real r atunci când acesta are un numar infinit de zecimale. Tinând seama de acest aspect valoarea numarului r va fi estimata fiind interpretata ca limita sirului r_n, sir ce are proprietatea de a fi întotdeauna convergent.

Prin analogie, în cazul unei fractii continue F, redusa $F_n = P_n / Q_n$ de ordin n va fi utilizata ca aproximare a lui F. In aceasta situatie însa, spre deosebire de cazul numerelor reale, sirul $\{ F_n \}_n$ nu este neaparat convergent.

Asadar extrapolarea pentru fractii continue a rezultatelor privind numerele reale poate conduce la erori grave de interpretare.

Exemplul 5. In literatura de specialitate sunt adesea prezentate, fara justificari, dezvoltari în fractie continua infinita pentru diferite functii larg utilizate în practica : sin (x) , cos (x) , tg (x) , ctg (x) , arcsin (x) , arccos (x) , arctg (x) , arcctg (x) , sinh (x) , cosh (x) , ln (x) , exp (x) , gama (x) , beta (x) , functia gama incompleta etc. Astfel Demidovich, Maron (1981, p. 74) sugereaza, fara demonstratie însa, urmatoarea dezvoltare în fractie continua a functiei exponentiale $f(x) = e^x$:

$$e^x = \left[0 ; \frac{1}{1}, \frac{-2x}{x+2}, \frac{x^2}{6}, \frac{x^2}{10}, \cdots, \frac{x^2}{4n+2}, \cdots \right] \qquad (19)$$

Aproximam aceasta fractie continua "infinita" cu redusa P_n / Q_n de ordin n unde P_n si Q_n sunt definiti recursiv prin relatiile (5) , adica

$$P_n = a_n \cdot P_{n-1} + b_n \cdot P_{n-2} \quad ; \qquad Q_n = a_n \cdot Q_{n-1} + b_n \cdot Q_{n-2} \quad ; \quad n \geq 1 \qquad (20)$$

cu respectarea Conventiei 1 adica $P_{-1} = 1$, $Q_{-1} = 0$, $P_0 = 0$, $Q_0 = 1$.

Din (19) deducem $a_1 = 1$, $b_1 = 1$, $a_2 = x + 2$, $b_2 = -2.x$, $a_{k+2} = 4k + 2$, $b_{k+2} = x^2$, $\forall\, k \geq 1$. Rescriind relatiile (20) obtinem formulele de calcul efectiv al redusele fractiei continue definita prin relatia (19),

$$P_1 = P_0 + P_{-1} = 1 \quad ; \qquad Q_1 = Q_0 + Q_{-1} = 1 \quad ;$$

$$P_2 = (x+2) P_1 - 2x P_0 = x + 2 \quad ; \qquad Q_2 = (x+2) Q_1 - 2x Q_0 = 2 - x \quad ;$$

$$P_n = a_n P_{n-1} + b_n P_{n-2} \quad ; \qquad Q_n = a_n Q_{n-1} + b_n Q_{n-2} \quad ; \qquad (21)$$

unde $a_n = 4n - 6$ si $b_n = x^2$, $\forall\, n \geq 3$.

In acest caz Ipoteza 3 este satisfacuta deoarece $a_k \geq 0$, $b_k \geq 0$ pentru $k \geq 3$ si în plus avem inegalitatea $b_k = x^2 < 4k - 6 = a_k$ pentru indici k ce depasesc un anumit prag (pragul este evident dependent de argumentul x). Conform Teoremei 3 avem convergenta reduselor P_n / Q_n catre valoarea atribuita fractiei continue infinite (19).

Aplicatia 4. Bazându-ne pe expresia fractiei continue (19) propunem urmatoarea procedura BASIC f(x) de aproximare a valorii functiei e^x , acest lucru realizându-se cu o eroare $\varepsilon = 10^{-4}$:

```
REM     Calculul redusei f(x) a functiei Exp(x) dezvoltata in fractie continua
FUNCTION f (x)
p1 = 1:     p2 = 2 + x :     q1 = 1:     q2 = 2 - x :     r1 = p1 / q1 :     r2 = p2 / q2
n = 3 :     b = x * x :     eps = .0001     ' eps = eroarea de aproximare
DO UNTIL (ABS(r1 - r2) < eps) :     a = 4 * n - 6     ' a = termen curent al fractiei continue F
p3 = a * p2 + b * p1 :     q3 = a * q2 + b * q1     ' Estimarea urmatoarei reduse
REM  Initializarea pasului urmator
n = n + 1 :  p1 = p2 :  p2 = p3 :  q1 = q2 :  q2 = q3 :  r1 = p1 / q1 :  r2 = p2 / q2 :  LOOP
f = r2 :     END FUNCTION
```

De fapt procedura f(x) reprezinta redusa de ordin n a fractiei continue (19). Ordinul n este determinat astfel încât sa fie asigurata precizia ε în aproximarea prin f(x) a functiei e^x .

Ca si în programele precedente si în acest program valorile termenilor P_n , respectiv Q_n , nu au fost retinute în tablouri (vectori) de dimensiune variabila. Astfel pentru memorarea termenilor

consecutivi P_{n-2} , P_{n-1} , P_n (Q_{n-2} , Q_{n-1} , Q_n) au fost utilizate variabilele numerice de tip real (în simpla precizie) P_1 , P_2 , P_3 (respectiv Q_1 ,Q_2 , Q_3) ce urmeaza a fi apoi reactualizate la fiecare crestere cu o unitate a valorii indicelui n (formulele (5)).

Având asigurata convergenta reduselor sale, fractia continua (19) este aproximata cu acea redusa $R_n = P_n / Q_n$ de ordin n a carei valoare "nu difera prea mult" de redusa P_{n-1} / Q_{n-1} de ordin n-1 , adica $| R_n - R_{n-1} | < \varepsilon$, ε fiind "eroarea de aproximare" acceptata .

Aplicatia 5. Verificarea corectitudinii formulei (19) poate fi efectuata indirect de urmatorul program ce urmareste evaluarea erorilor de calcul rezultate prin estimarea valorii expresiei $E(x,y) = e^x \cdot e^y - e^{x+y}$. Teoretic expresia E (x,y) este nula pentru orice x , y $\in$ **R** .

```
REM  Functia exponentiala este dezvoltata in fractie continua
REM  Verificarea identitatii       exp(x) * exp(y)  =  exp(x+y)
DECLARE FUNCTION f! (x!)
FOR x = .5 TO 1.5 STEP .5 :        FOR y = .2 TO  0.9 STEP .2
PRINT USING "F(#.#) * F(#.#) - F(#.#) = ###.##### * ###.##### - ###.#####  =  ##.########";
   x; y; x + y; f(x); f(y); f(x + y); f(x) * f(y) - f(x + y)
NEXT y :        NEXT x
```

Functia f(x) a fost proiectata în Aplicatia 4 defineste cu o eroare $\varepsilon = 0.0001$ redusa fractei continue (19). Rezultatele rularii programului sunt listate în continuare.

```
F(0.5) * F(0.2) - F(0.7) =  1.64872 *  1.22140 -  2.01375  =  0.00000001
F(0.5) * F(0.4) - F(0.9) =  1.64872 *  1.49182 -  2.45960  =  0.00000040
F(0.5) * F(0.6) - F(1.1) =  1.64872 *  1.82212 -  3.00417  =  0.00000041
F(0.5) * F(0.8) - F(1.3) =  1.64872 *  2.22554 -  3.66930  =  0.00000073
F(1.0) * F(0.2) - F(1.2) =  2.71828 *  1.22140 -  3.32012  = -0.00000016
F(1.0) * F(0.4) - F(1.4) =  2.71828 *  1.49182 -  4.05520  = -0.00000012
F(1.0) * F(0.6) - F(1.6) =  2.71828 *  1.82212 -  4.95303  = -0.00000033
F(1.0) * F(0.8) - F(1.8) =  2.71828 *  2.22554 -  6.04965  = -0.00000019
F(1.5) * F(0.2) - F(1.7) =  4.48169 *  1.22140 -  5.47395  = -0.00000070
F(1.5) * F(0.4) - F(1.9) =  4.48169 *  1.49182 -  6.68589  = -0.00000045
F(1.5) * F(0.6) - F(2.1) =  4.48169 *  1.82212 -  8.16617  = -0.00000056
F(1.5) * F(0.8) - F(2.3) =  4.48169 *  2.22554 -  9.97418  =  0.00000084
```

Se observa ca pentru $\varepsilon = 10^{-4}$ valoarea expresiei $| E^*(x,y) |$, cu $E^*(x,y) = f(x)\, f(y) - f(x+y)$, nu depaseste pragul de 0.00000084 pentru x $\in$ { 0.5 , 1 , 1.5 } si y $\in$ { 0.2 , 0.4 , 0.6 , 0.8 }. Acest rezultat caracterizeaza acuratetea aproximarii functiei e^x prin redusa f(x) a fractiei continue descrisa de relatia (19).

Observatia 5. Daca în subrutina de tip functie f(x) definita în Aplicatia 4 calculele se vor efectua în precizie dubla, considerând în plus $\varepsilon = 10^{-5}$ în loc de $\varepsilon = 10^{-4}$, atunci vom obtine o îmbunatatire substantiala a aproximatiei f(x) a functiei e^x . Aceasta afirmatie este sustinuta experimental prin listarea valorilor expresiei $E^*(x,y) = f(x)\, f(y) - f(x+y)$ ce sunt aproape nule.

Lucrarea "Asupra convergentei fractiilor continue" (Stefan Stefanescu : Studii si Cercetari de Calcul Economic si Cibernetica Economica, vol XXXIII, 1999) trateaza si alte aspecte legate de convergenta fractiilor continue.

CALCULUL VALORILOR FUNCTIILOR

In aceasta sectiune vom evidentia mai multe tehnici utilizate la calculul valorilor functiilor.

De cele mai multe ori în practica nu este posibila estimarea valorii exacte $b = f(a)$ a unei functii $f(x)$ într-un punct a. In acest caz vom opera cu o aproximare b^* a lui b, cu avantajul de a modifica oricând, dupa dorinta, acuratetea aproximarii. Cu alte cuvinte eroarea $|b - b^*|$ este întotdeauna sub control.

Problemele ce vor fi abordate în continuare privesc tehnica dezvoltarilor în serie Taylor, estimarea valorilor seriilor ce au un numar infinit de termeni, extragerea radacinii de ordin p dintr-un numar, calculul valorii unui polinom. Scopul urmarit este de a sugera proceduri de solutionare cât mai variate.

1. Dezvoltarea functiilor în serie Taylor

Vom considera o functie $f : \mathbf{R} \to \mathbf{R}$ $n+1$ derivabila si a o valoare reala fixata. Suntem interesati sa aproximam functia $f(x)$ printr-un polinom $T_n(x)$ de gradul n astfel încât "local", într-o vecinatate V_a a punctului a, polinomul T_n sa descrie foarte bine comportamentul functiei f. Intr-o astfel de situatie, în loc de a calcula direct valoarea expresiei $f(b)$, pentru $b \in V_a$, vom utiliza aproximatia sa $T_n(b)$ ce se poate obtine usor cu ajutorul schemei lui Horner.

Polinomul $T_n(x)$ este caracterizat de cei $n + 1$ coeficienti ai sai $a_0, a_1, a_2, \dots, a_n$,

$$T_n(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \dots + a_{n-2} x^2 + a_{n-1} x + a_n \qquad (1)$$

Polinomul $T_n(x)$ simuleaza foarte bine comportamentul functiei $f(x)$ daca valoarea polinomului si a derivatelor sale $T_n^{(k)}$ calculate "local " în punctul a coincid cu valoarea functiei f, respectiv cu valorile derivatelor $f^{(k)}$ ale functiei f, estimate de asemenea în acelasi punct a.

Obtinem astfel un sistem liniar de $n + 1$ ecuatii, anume : $T_n(a) = f(a)$, $T_n^{(1)}(a) = f^{(1)}(a)$, $T_n^{(2)}(a) = f^{(2)}(a), \dots, T_n^{(n-1)}(a) = f^{(n-1)}(a)$, $T_n^{(n)}(a) = f^{(n)}(a)$. Solutia acestui sistem reprezinta cei $n + 1$ coeficienti necunoscuti a_j, $0 \le j \le n$, ai polinomului $T_n(x)$.

Forma Taylor a polinomului $T_n(x)$ este :

$$T_n(x) = f(a) + [\, f^{(1)}(a) / 1! \,](x - a) + \dots + [\, f^{(j)}(a) / j! \,](x - a)^j + \dots + [\, f^{(n)}(a) / n! \,](x - a)^n \qquad (2)$$

Calculând derivatele succesive ale polinomului $T_n(x)$ sunt verificate în mod direct egalitatile: $T_n^{(k)}(a) = f^{(k)}(a)$, aceasta pentru orice $0 \le k \le n$ (cu conventia $f^{(0)}(x) = f(x)$).

Forma Lagrange a restului $R_n(x) = f(x) - T_n(x)$ rezultat prin aproximarea functiei $f(x)$ cu polinomul Taylor $T_n(x)$ are expresia

$$R_n(x) = [\, f^{(n+1)}(c) / (n+1)! \,].(x - a)^{n+1} \qquad (3)$$

unde c este o valoare intermediara, necunoscuta, cuprinsa între a si x ($a < c < x$ sau $x < c < a$).

Rezultatele enuntate pot fi obtinute si în conditii mai restrictive (de exemplu exista derivata $f^{(n+2)}(x)$) pe un interval în care punctul a este punct interior). Pentru detalii a se studia si capitolul privind aproximarea functiilor.

Aplicatia 1. Vom estima valoarea functiei $f(x) = e^x$ cu o eroare ε data.

Cum $f^{(n)}(x) = e^x$ pentru orice $n \in N$, atunci considerând în formula (2) $a = 0$ se obtine urmatorul polinom Taylor $T_n(x)$ atasat functiei $f(x) = e^x$,:

$$T_n(x) = 1 + x/1! + x^2/2! + x^3/3! + x^4/4! + \quad + x^n/n! \qquad (4)$$

Restul $R_n(x)$ al aproximarii functiei $f(x)$ prin polinomul Taylor $T_n(x)$ are expresia

$$R_n(x) = e^c \cdot [\, x^{n+1}/(n+1)! \,] \qquad (5)$$

unde $|c| < x$ (c se gaseste între 0 si x ; a = 0).

Vom aborda mai întâi cazul $x \geq 0$.

De fapt valoarea $T_n(b)$, $b \geq 0$, poate rezulta estimând direct suma $S = u_0 + u_1 + u_2 + \ldots + u_n$ unde $u_j = b^j/j!$, $0 \leq j \leq n$. La evaluarea sumei S, pentru micsorarea volumului de calcule vom folosi relatia recurenta $u_{j+1} = u_j \cdot [\, b/(j+1) \,]$ cu ajutorul careia se vor calcula succesiv termenii u_j, $1 \leq j \leq n$, pornind de la valoarea $u_0 = 1$.

Va trebui însa sa precizam gradul n al polinomului Taylor $T_n(x)$ astfel încât eroarea de aproximare sa nu depaseasca valoarea ε. In vederea determinarii gradului polinomului Taylor $T_n(x)$ este mai comod sa ne referim la eroarea relativa de aproximare impunând, de exemplu, conditia $|R_n(b)/f(b)| < \varepsilon$.

Pentru $b \geq 0$, cum $0 < c < b$, avem

$$|R_n(b)/f(b)| = |\, b^{n+1} \cdot [\, f^{(n+1)}(c)/(n+1)! \,]/f(b) \,| = |\, [\, e^c \cdot b^{n+1}/(n+1)! \,]/e^b \,| <$$
$$< |\, [\, e^b \cdot b^{n+1}/(n+1)! \,]/e^b \,| = |\, b^{n+1}/(n+1)! \,| = |\, u_{n+1} \,| \qquad (6)$$

si deci vom determina gradul n al polinomului Taylor astfel încât $u_{n+1} < \varepsilon$.

Programul BASIC urmator calculeaza suma $S = u_0 + u_1 + u_2 + \ldots + u_n$ însumând termenii u_j pâna la primul coeficient $u_{n+1} < \varepsilon$.

Valorile termenilor u_n se deduc iterativ dupa formula $u_{n+1} = b\, u_n/(n+1)$

```
REM  Estimarea valorii functiei exponentiale  exp (x)
DECLARE FUNCTION f1! (x!)
INPUT "x = "; x :                    PRINT "Exp("; x; " ) = "; f1(x)
FUNCTION f1 (x)      ' Polinomul Taylor pentru functia exponentiala
eps = 10 ^ (-5) :      s = 1 :      n = 0 :      u = 1
DO
n = n + 1 :      u = u * x / n :      s = s + u :      LOOP UNTIL u < eps
f1 = s :                END FUNCTION
```

Exemplul 1. Pentru programul BASIC din Aplicatia 1 admitem o eroare de aproximare $\varepsilon = 10^{-5}$. S-au obtinut astfel urmatoarele rezultate de test :

$e^0 \approx 1.0$ $\qquad$ $e^1 \approx 2.718282$ $\qquad$ $e^2 \approx 7.389055$

Observatia 1. Ne putem asigura de corectitudinea programului propus bazându-ne pe identitatea $e^{2x} = e^x e^x$ ce este caracteristica functiei exponentiale pentru orice $x \in \mathbf{R}$.

Asadar, aplicând succesiv procedura f1 cu argumentele x si respectiv 2x ar trebui ca valoarea expresiei E(x), unde $E(x) = (\, f1(x) \,)^2 / f1(2x)$, sa fie întotdeauna foarte apropiata de 1, aceasta pentru orice valoare reala a parametrului x.

Rezultatele obtinute prin rularea programului urmator sunt listate în Tabelul 1.

```
REM  Verificarea corectitudinii estimarii valorilor functiei exponentiale
REM  Se listeaza valorile expresiei  f1(x) . f1(x) / f(2.x)
DECLARE FUNCTION f1! (x!)
FOR x = -10 TO 10 :          e1 = f1(x) :          e2 = f1(2 * x)
PRINT "f1("; x; ") * f1("; x; ") / f1(2 *("; x; ")) = "; e1 * e1 / e2 :          NEXT x
```

Tabelul 1. Valoarea expresiei $E(x) = (f1(x))^2 / f1(2x)$, functia $f1(x)$ fiind definita în Aplicatia 1 (conditia de oprire a algoritmului : $u_{n+1} \le \varepsilon = 10^{-5}$)

x	E(x)	x	E(x)	x	E(x)	x	E(x)
-10	-4.263158	-9	-3.764706	-8	-3.266667	-7	-2.769231
-6	-2.272727	-5	-1.777778	-4	-1.285714	-3	-.8 000000
-2	-.3333333	-1	0.000000	0	1.000000	1	1.000000
2	.9999998	3	1.000000	4	1.000000	5	1.000000
6	.9999998	7	1.000000	8	1.000000	9	.9999996
10	1.000000						

Observatia 2.. Analizând datele din Tabelul 1 remarcam inexactitati flagrante pentru valori negative ale argumentului x . De altfel rationamentul ce a condus la proiectarea functiei $f1(x)$ a fost dezvoltat în ipoteza $x \ge 0$.

Inlocuirea conditiei de "oprire" $u_{n+1} < \varepsilon$ ce precizeaza numarul termenilor u_n ce urmeaza a se însuma, printr-o noua conditie de "oprire" de forma $| u_{n+1} | < \varepsilon$ nu elimina erorile de estimare decât pentru acele valori negative ale argumentului x ce sunt apropiate de 0 . Efectuarea acestui tip de modificare nu conduce întotdeauna la verificarea relatiei $(f1(x))^2 / f1(2x) \approx 1$, aceasta pentru valori negative ale argumentului x . A se vedea în acest sens datele din Tabelul 2.

Tabelul 2. Valoarea expresiei $E(x) = (f1(x))^2 / f1(2x)$, functia $f1(x)$ fiind definita în Aplicatia 1 (conditia de oprire a algoritmului este $| u_{n+1} | \le \varepsilon = 10^{-5}$)

x	E(x)	x	E(x)	x	E(x)	x	E(x)
-10	7.388389E-09	-9	1.275071E-07	-8	1.873012E-06	-7	-2.540566E-04
-6	-1.316584E-02	-5	-.6067871	-4	.959567	-3	1.000095
-2	1.000054	-1	.9999902	0	1.000000		

Exemplul 2. Rezultatele contradictorii din Tabelul 2 se datoresc în fapt utilizarii majorantului e^b pentru expresia e^c cu $0 < c < b$ inegalitate ce nu se mai este valabila atunci când $b < c < 0$.

Aceasta eroare subtila de proiectare a functiei $f1(x)$ poate fi eliminata tinând seama de urmatoarea egalitate $e^{-x} = 1 / e^x$. In acest mod estimarea valorilor functiei $f1(x)$ pentru argumente x negative se reduce la estimarea expresiei $1 / f1(-x)$, unde evident $-x \ge 0$.

Functia BASIC $f1(x)$ din Aplicatia 1 va fi reproiectata astfel :

```
FUNCTION f1 (x)        ' Polinomul Taylor pentru functia exponentiala
eps = 10 ^ (-5) :      b = ABS(x) :      s = 1 :      n = 0
u = 1
DO
n = n + 1 :      u = u * b / n :      s = s + u :      LOOP UNTIL u < eps
IF x < 0 THEN s = 1 / s
f1 = s :                END FUNCTION
```

De aceasta data rezultatele obtinute dupa utilizarea noii proceduri f1 sunt corecte. A se vedea în acest sens rezultatele sintetizate în Tabelul 3.

Tabelul 3. Valoarea expresiei $E(x) = (f1(x))^2 / f1(2x)$, functia f1(x) fiind reproiectata în Exemplul 2 ($\varepsilon = 10^{-5}$; conditia de oprirea algoritmului $| u_{n+1} | < \varepsilon = 10^{-5}$)

x	E(x)	x	E(x)	x	E(x)	x	E(x)
-10	.9999999	-9	1.000000	-8	.9999997	-7	.9999996
-6	1.000000	-5	.9999999	-4	.9999999	-3	.9999996
-2	1.000000	-1	.9999999	0	1.000000	10	1.000000
20	.9999999	30	.9999995	40	1.000001		

Exemplul 3. Pentru un grad n precizat, polinomul Taylor $T_n(x)$ aproximeaza "local" comportamentul functiei f(x) , eroarea de aproximare fiind mica numai într-o vecinatate a punctului a Acuratetea aproximarii depinde de valoarea lui n (formula (3))

Astfel prin aproximarea functiei $f(x) = e^x$ cu ajutorul polinomului

$$T_n(x) = 1 + x / 1! + x^2 / 2! + x^3 / 3! + x^4 / 4! + \ldots + x^n / n! \qquad (7)$$

unde, de aceasta data gradul n al polinomului $T_n(x)$ va ramâne fixat, aceasta independent de valoarea argumentului x , s-au obtin rezultatele din Tabelul 4. Putem astfel aprecia comportamentul expresiei $E_n(x) = (T_n(x))^2 / T_n(2x)$ pentru n = 15 :

Tabelul 4. Valoarea expresiei $E_n(x) = (T_n(x))^2 / T_n(2x)$ pentru n = 15
(polinomul Taylor $T_n(x)$ de gradul n este definit de formula (7))

x	$E_{15}(x)$	x	$E_{15}(x)$	x	$E_{15}(x)$	x	$E_{15}(x)$
0	1.000000	1	1.000000	2	1.000005	3	1.000510
5	1.051093	7	1.486782	9	3.336488	10	5.781590
13	41.15107	15	165.7146	20	4483.245		

Observatia 3. Daca polinomul $T_n(x)$ constituie o buna aproximare "globala" a functiei $f(x) = e^x$ atunci ar trebuie sa fie satisfacuta relatia $E_n(x) = (T_n(x))^2 / T_n(2x) \approx 1$ pentru orice $x \in R$ Aproximatia $E_n(x) \approx 1$ nu se realizeaza însa decât "local" , pentru valori ale argumentului x ce

apartin unei vecinatati a punctului $a = 0$ ce defineste polinomul Taylor (7) (a se vedea formula (2)).

Observatia 4. Precizia aproximarii va creste odata cu marirea gradului n al polinomului de aproximare $T_n(x)$. Astfel pentru $n = 30$ rezultatele obtinute sunt în mod evident mai precise. A se compara rezultatele din Tabelul 4 (unde $n = 15$) cu cele din Tabelul 5 (obtinut pentru $n = 30$).

Tabelul 5. Valoarea expresiei $E_n(x) = (T_n(x))^2 / T_n(2x)$ pentru $n = 30$
(polinomul Taylor $T_n(x)$ de gradul n este definit de formula (7)).

x	$E_n(x)$	x	$E_n(x)$	x	$E_n(x)$	x	$E_n(x)$
0	1.000000	1	1.000000	2	1.000000	3	1.000000
5	1.000000	7	1.000060	8	1.000568	9	1.003342
10	1.013659	13	1.229298	15	1.822928	20	15.77512

Observatia 5. In varianta propusa în Exemplul 2 gradul n al polinomului Taylor $T_n(x)$ era variabil, fiind determinat în raport de valoarea absoluta a argumentului x. In acest caz estimarile valorilor functiei e^x au o precizie mare, fapt evidentiat si de valori foarte aproape de 1 pentru expresia $E_n(x) = (T_n(x))^2 / T_n(2x)$ atunci când argumentul x ia valori relativ mari (Tabelul 3).

Aceasta afirmatie este sustinuta si de un studiu comparativ al rezultatelor din Tabelele 3-5 pentru $x = 10$ si $x = 20$. Atât în cazul în care se fixeaza gradul polinomului Taylor de aproximare cât si în situatia unui polinom $T_n(x)$ de grad n dependent de valoarea parametrului x, pot apare erori de calcul de tipul "overflow" (aceasta pentru valori absolute relativ mari ale argumentului x).

Aplicatia 2. Vom evalua valoarea functiei $f(x) = \ln (1 + x)$ pentru $x > -1$.

Printr-un calcul direct se obtin expresiile derivatelor multiple $f^{(k)}(x)$ ale functiei $f(x) = \ln(1 + x)$, anume $f^{(1)}(x) = 1 / (1 + x)$; $f^{(k)}(x) = (-1)^{k-1} . [(k - 1)!] / (1 + x)^k$ pentru $k \geq 2$.

Considerând în (2) $a = 0$ deducem urmatoarea forma a polinomului Taylor de aproximare

$$T_n(x) = x - x^2 / 2 + x^3 / 3 - x^4 / 4 + x^5 / 5 - x^6 / 6 + \ldots + (-1)^{n-1} . x^n / n \qquad (8)$$

Restul $R_n(x)$ rezultat în procesul de aproximare este dat de expresia (3), adica

$$R_n(x) = f(x) - T_n(x) = [(-1)^n / (1 + c)^{n+1}] [x^{n+1} / (n + 1)] \qquad (9)$$

unde $0 < c < x$ sau $x < c < 0$.

Vom aborda în primul rând cazul $0 \leq x \leq 1$.

Pentru $0 \leq x \leq 1$ avem $0 < c < 1$ si deci $0 < 1 / (1 + c) < 1$. Stabilim astfel inegalitatea

$$| R_n(x) | < | x / (1 + c) |^{n+1} / (n + 1) < 1 / (n + 1) \qquad (10)$$

Prin urmare pentru orice valoare reala x fixata, $0 \leq x \leq 1$, sirul $R_n(x)$ definit de (10) tinde la zero atunci când n tinde la infinit. Asadar, pentru orice $0 \leq x \leq 1$, odata cu marirea gradului n al polinomului $T_n(x)$, asistam la o îmbunatatire a acuratetii procesului de aproximare a functiei $f(x)$ prin polinomul Taylor $T_n(x)$ atasat. Mentionam faptul ca pentru $x = 1$ convergenta restului $R_n(1)$ la valoarea 0 poate sa fie destul de lenta.

Cum sirul de valori reale $| (-1)^{n-1} / n | / | (-1)^n / (n + 1) |$ tinde la valoarea 1, rezulta ca raza de convergenta a seriei infinite de puteri

$$S(x) = x - x^2 / 2 + x^3 / 3 - x^4 / 4 + x^5 / 5 - x^6 / 6 + \ldots + (-1)^{n-1} x^n / n + \ldots \qquad (11)$$

este 1. Asadar convergenta seriei $S(x)$ data de (11) este asigurata pentru orice $x \in (-1, 1)$. Deci

polinomul $T_n(x)$ definit de relatia (8) aproximeaza functia $f(x) = \ln(1 + x)$ pentru orice $x \in (-1 , 1)$

Proiectarea procedurii f2(y) de estimare a valorii functiei $f(y) = \ln(y)$, $0 < y \leq 2$, se bazeaza pe utilizarea polinomului de interpolare $T_n(x)$ precizat de (8) pentru $y = 1 + x$.

```
FUNCTION f2 (y)          ' Estimarea valorii ln(y) pentru 0 < x = y - 1 <= 2
eps = 10 ^ (-5) :        x = y - 1 :        s = 0 :        n = 0 :        u = -1
DO
n = n + 1 :        u = -u * x :        s = s + u / n :        LOOP UNTIL ABS(u) < n * eps
f2 = s :                END FUNCTION
```

Exemplul 4. Stabilirea acuratetii aproximarii functiei $f(y) = \ln (y)$ prin polinomul Taylor $T_n(y-1)$ definit de relatia (8) o vom realiza utilizând identitatea $e^{\ln(y)} = y$ ce este valabila pentru orice $y > 0$. Asadar ar trebui ca $f1(f2(y)) \approx y$, adica $f1(f2(y)) - y \approx 0$ oricare ar fi $0 < y \leq 2$, unde $f1(x) \approx e^x$ (Aplicatia 1) si $f2(y) \approx \ln (y)$ (Aplicatia 2).

Programul urmator evalueaza expresia f1 (f2(y)) - y , având astfel o imagine a erorii rezultate în procesul de aproximare.

```
DECLARE FUNCTION f1! (x!)
DECLARE FUNCTION f2! (y!)
REM  Verificarea corectitudinii aproximarii lui ln(y) prin f2(y)
FOR y = .2 TO 2.1 STEP .2
z = f2(y)   ' z  estimeaza valoarea ln(y)
t = f1(z)   ' t  estimeaza valoarea exp(z)
e = t - y   ' e = eroarea aparuta
PRINT "f2("; y; ") = "; z; "      f1("; z; ") = "; t; "     f1( f2("; y; ") ) - "; y; " = "; e
NEXT y
```

Rezultatele rularii programului prezentat în Exemplul 4 confirma obtinerea unor bune aproximatii f1(x) si f2(y) pentru functiile e^x, respectiv ln (y) .

In Tabelul 6 sunt listate valorile expresiei D(y) = f1(f2(y)) - y , functiile f1(x) si f2(y) fiind proiectate în Aplicatia 1, respectiv Aplicatia 2.

Tabelul 6. Valoarea expresiei D(y) = f1(f2(y)) - y (functiile f1(x) si f2(y) sunt definite în Aplicatia 1, respectiv Aplicatia 2)

y	D(y)	y	D(y)	y	D(y)	y	D(y)
0.2	6.407499E-06	0.4	5.245209E-06	0.6	1.430511E-06	0.8	4.172325E-07
1.0	0.000000	1.2	2.384186E-07	1.4	1.192093E-06	1.6	5.722046E-06
1.8	-7.510185E-06	2.0	-7.033348E-06				

Observatia 6. Precizia valorilor D (x) nu mai este atât de aproape de 0 (ca în Tabelul 6) atunci când argumentul x nu ia valori în intervalul (0 , 2] . In adevar pentru $x > 1$ sirul de numere reale $T_n(x)$ nu converge la valoarea ln (1+x) atunci când n tinde la ∞ , $T_n(x)$ fiind polinomul Taylor atasat functiei $f(x) = \ln (1+x)$ în punctul 0 (formula (8)).

Observatia 7. Daca vom încerca sa estimam valoarea ln (3) apelând procedura f2(3) va aparea o eroare de tip "overflow" datorita imposibilitatii determinarii gradului n al polinomului $T_n(x)$ folosit pentru aproximarea functiei f (x) cu o acuratete ε precizata.

Fixând gradul n al polinomului $T_n(x)$ apar erori mari de aproximare. Astfel pentru n = 8 expresia D(y) = f1(f2(y)) - y nu ia valori apropiate de zero (ca în Exemplul 6). Prin rularea programului BASIC din Exemplul 4 s-au obtinut rezultatele :

```
f2( 2.0 ) =  .7456349     f1( .7456349 ) = 2.107779     f1( f2( 2.0 ) ) - 2.0 =  .107779
f2( 2.2 ) = 1.085315      f1( 1.085315 ) = 2.960371     f1( f2( 2.2 ) ) - 2.2 =  .7603714
f2( 2.4 ) = 2.151835      f1( 2.151835 ) = 8.600631     f1( f2( 2.4 ) ) - 2.4 = 6.200631
```

Datorita neconvergentei sirului $T_n(y)$ la ln(y) pentru y > 2 , o crestere a gradului n în procedura f2(y) nu va conduce la o micsorare a erorilor de aproximare D(y) = f1(f2(y)) - y , ci dimpotriva asistam la o marire a acestor erori. Acest aspect este evidentiat pentru n = 16 :

```
f2( 2.0 ) =  .7216952     f1( .7216952 ) = 2.057919     f1( f2( 2.0 ) ) - 2.0 = 5.791879E-02
f2( 2.2 ) = 1.481209      f1( 1.481209 ) = 4.398260     f1( f2( 2.2 ) ) - 2.2 = 2.19826
f2( 2.4 ) = 11.07967      f1( 11.07967 ) = 64839.23     f1( f2( 2.4 ) ) - 2.4 = 64836.84
```

2. Serii numerice

In practica se pune adesea problema estimarii valorii S a unei serii ce are o infinitate de termeni,

$$S = a_1 + a_2 + a_3 + a_4 + \ldots + a_n + \ldots \qquad (12)$$

Valoarea S este interpretata ca o limita a sirului sumelor partiale S_n ,

$$S_n = a_1 + a_2 + a_3 + a_4 + \ldots + a_n \qquad (13)$$

atunci când n tinde la infinit.

Nu putem defini valoarea seriei infinite S în cazul în care sirul $\{ S_n \}_n$ al sumelor sale partiale nu este convergent. In literatura de specialitate sunt indicate mai multe criterii în vederea stabilirii convergentei sirului $\{ S_n \}_n$ (a se vedea, de exemplu, M. Nicolescu, N. Dinculeanu, S. Marcus , 1966, vol.1, p.678).

Practic suntem pusi în imposibilitatea adunarii unui numar infinit de termeni a_n în vederea precizarii valorii sumei S . Vom accepta asadar un compromis aproximând valoarea S cu o suma partiala, finita, S_n . In aceasta situatie va trebui însa sa controlam eroarea de aproximare facuta, adica sa determinam numarul n de termeni a_j , $1 \le j \le n$, ce se vor însuma astfel încât diferenta $| S - S_n |$ sa nu depaseasca un prag impus ε (ce se poate modifica în functie de cerinte).

Probleme deosebite apar chiar si în cazul estimarii valorii unei serii cu un numar finit de termeni. Un exemplu concludent este evidentiat de situatia în care termenii seriei studiate sunt obtinuti dupa formule de calcul complicate. In acest caz vom fi deosebit de atenti la modul de programare, în vederea evitarii omiterii unor termeni, permanent constienti de necesitatea asigurarii controlului fenomenului de propagarea erorilor aritmetice de rotunjire. In acest context vom prezenta o noua aplicatie.

Aplicatia 3. Sa calculam valorile coeficientilor δ_{km} , $1 \le k \le k_0$, $1 \le m \le m_0$, dati de urmatoarea suma ce se efectueaza dupa indicii j , s :

$$\delta_{km} = -\sum_{j=1}^{n}\left[\frac{1}{[j.(k.m+1)+m+k.m+1].[j.(k.(m+1)+1)+m+1+k.(m+1)+1]} - \right.$$

$$\left. -\sum_{s=1}^{n}\frac{1}{[(s+1).(j.(k.m+1)+m)+k.m+1].[(s+1).(j.(k.(m+1)+1)+m+1)+k.(m+1)+1]}\right] \qquad (14)$$

Expresia (14) a rezultat în urma unor calcule statistice. Datorita unui numar relativ mare de paranteze, aceasta expresie nu este usor de programat de un începator. Multiplele calcule numerice fac ca erorile de transcriere în cod sursa sa fie cu greutate depistabile.

Având în vedere aceste aspecte, pentru estimarea coficientilor δ_{km} este recomandabila o restructurare a expresiei (14), rescriind-o într-o forma simplificata, ca de exemplu

$$\delta_{km} = -\sum_{j=1}^{n}\left[\frac{1}{t_{11}.t_{12}} - \sum_{s=1}^{n}\frac{1}{t_{21}.t_{22}}\right] \qquad (15)$$

unde termenii $t_{11}, t_{12}, t_{21}, t_{22}$ sunt de forma

$$t_{11} = j.(k.m+1)+m+k.m+1$$
$$t_{12} = j.(k.(m+1)+1)+m+1+k.(m+1)+1$$
$$t_{21} = (s+1).(j.(k.m+1)+m)+k.m+1$$
$$t_{22} = (s+1).(j.(k.(m+1)+1)+m+1)+k.(m+1)+1 \qquad (16)$$

Este de dorit ca în formula (15) efectuarea calculelor sa se faca în precizie dubla. In plus, în programul BASIC urmator s-au estimat numai o singura data valorile unor expresii mai des utilizate, ca de exemplu : m + 1 , k . m , k . (m + 1) .

```
REM  Calculul coeficientilor d(k,m)
INPUT "Numar de valori pentru  k , m = "; nvk, nvm :          INPUT "n = "; n
FOR k = 1 TO nvk
FOR m = 1 TO nvm :          k# = k :          m# = m :          m1# = m# + 1!
km# = k# * m# :          km1# = k# * m1# :          sumj# = 0!
FOR j = 1 TO n :          j# = j :          t11# = j# * (km# + 1!) + m# + km# + 1!
t12# = j# * (km1# + 1!) + m1# + km1# + 1! :          sums# = 0!
FOR s = 1 TO n :          s1# = s + 1! :          t21# = s1# * (j# * (km# + 1!) + m#) + km# + 1!
t22# = s1# * (j# * (km1# + 1!) + m1#) + km1# + 1! :          sums# = sums# + 1! / (t21# * t22#)
NEXT s
sumj# = sumj# + 1! / (t11# * t12#) - sums# :          NEXT j
dkm# = -sumj# :          PRINT "d ("; k; " , "; m; " ) = "; dkm# :          NEXT m
NEXT k :          PRINT
```

Exemplul 5. In urma rularii programului BASIC din Aplicatia 3 pentru k = m = 2 si n = 30 s-au obtinut rezultatele :

δ_{11} = -6.031794868696926D-03 δ_{12} = -3.631296213849436D-03

δ_{21} = -1.097065583396969D-03 δ_{22} = -6.260932979971371D-04

Aplicatia 4. Fie $\{a_n\}_n$ un sir de valori reale pozitive a carui limita este 0 . Intentionam sa estimam valoarea sumei infinite $S = a_1 + a_2 + a_3 + a_4 + \ldots + a_{n-1} + a_n + a_{n+1} + \ldots$ cu o eroare ε ce este precizata.

Daca sirul $\{ S_n \}_n$ al sumelor partiale (13) este convergent (la o valoare finita) atunci limita S a acestui sir este interpretata drept valoarea sumei seriei infinite $a_1 + a_2 + a_3 + \ldots + a_n + a_{n+1} + \ldots$.

Observatia 7. Deoarece $a_n = (a_1 + a_2 + \ldots + a_{n-1} + a_n) - (a_1 + a_2 + \ldots a_{n-1}) = S_n - S_{n-1}$, convergenta sirului $\{ S_n \}$ la S, S finit, va impune ca sirul $\{ a_n \}_n$ sa conveaarga la zero. Reciproca acestei afirmatii nu este adevarata, sirul $\{ a_n \}_n$ putând converge la 0 fara ca seria infinita S sa aiba neaparat o limita finita. Un exemplu în acest sens îl constituie seria

$$S = 1/1 + 1/2 + 1/3 + 1/4 + \ldots + 1/n + \ldots$$

valoarea lui S fiind în acest caz ∞.

In practica se cere adesea evaluarea sumei infinite S cu o eroare ε impusa. Aproximând S prin suma partiala S_n se obtine restul R_n,

$$R_n = a_{n+1} + a_{n+2} + a_{n+3} + a_{n+4} + \ldots \qquad (17)$$

Va trebui determinata valoarea indicelui n pentru care $| R_n | \le \varepsilon$.

Ipoteza 1. Exista $n_0 \in N$ astfel încât pentru orice $m \ge n_0$ sa avem

$$a_{m+1} / a_m \le q < 1$$

In conditiile Ipotezei 1, daca $n \ge n_0$ atunci deducem succesiv inegalitatile

$$a_{n+1} \le q\, a_n \qquad a_{n+2} \le q\, a_{n+1} \le q^2 a_n \qquad a_{n+3} \le q\, a_{n+2} \le q^3 a_n$$

adica $a_{n+k} \le q^k a_n$, fapt ce ne va permite evaluarea restului R_n, anume

$$R_n = a_{n+1} + a_{n+2} + a_{n+3} + a_{n+4} + \ldots \le a_n [q + q^2 + q^3 + \ldots + q^k + \ldots]$$

Prin urmare

$$R_n \le a_n q / (1 - q) \qquad (18)$$

Deci va trebui sa determinam $n \ge n_0$ pentru care

$$a_n q / (1 - q) \le \varepsilon \qquad (19)$$

Respectând conditiile (19) suntem siguri ca aproximatia S_n a lui S are precizia ceruta, adica

$$S - S_n = R_n \le a_n q / (1 - q) \le \varepsilon$$

Exemplul 6. Vom estima cu o precizie $\varepsilon = 10^{-5}$ suma

$$S = 1/3 + 2/3^2 + 3/3^3 + 4/3^4 + 5/3^5 + \ldots + n/3^n + \ldots \qquad (20)$$

Cum $(n + 1) / n \le 9 / 8$ pentru $n \ge 8$ deducem ca

$$a_{n+1} / a_n \le (n + 1) / (3 . n) \le 3 / 8 = q < 1$$

Conform relatiei (18) vom determina $n \ge 8$ astfel încât

$$a_n q / (1 - q) = (n / 3^n) (3 / 8) / (1 - 3 / 8) = n / (5 . 3^{n-1}) \le \varepsilon = 10^{-5}$$

Prezentam în continuare un program BASIC ce va preciza prima valoare $n \ge n_0$ pentru care $a_n . q / (1 - q) \le \varepsilon$, unde $n_0 = 8$, $a_n = n / 3^n$, $q = 3 / 8$.

```
REM  Determinarea valorii  n0 <= n <= nmax  astfel incat  a(n) . q / (1 - q) < eps
REM  Calculul sumei  S = a(1) + a(2) + a(3) + ... + a(n)
DECLARE FUNCTION a! (n!)
n0 = 8 :          nmax = 99 :          q = 3 / 8
FOR j = 3 TO 8 :          eps = 10 ^ (-j) :          s = 0
FOR n = 1 TO nmax :          s = s + a(n)
IF (a(n) * q / (1 - q) <= eps) AND (n >= n0) THEN PRINT "eps = "; eps; " ;    S("; n; " ) = "; s
GOTO 1
```

```
NEXT n
PRINT "eps = "; eps; "    Nu a fost gasita solutie pentru valoriale lui  n  in intervalul  [ "; n0; " , ", nmax; " ]"
1 : NEXT j
FUNCTION a (n) :          a = n / 3 ^ n :          END FUNCTION
```

Programul va cauta secvential o valoare întreaga n cuprinsa în intervalul $[n_0 , n_{max}]$ astfel încât $| S - S_n | < \varepsilon$. Suma infinita S data de formula (20) este aproximata prin suma partiala S_n ,

$$S_n = 1/3 + 2/3^2 + 3/3^3 + 4/3^4 + 5/3^5 + \quad + n/3^n$$

In Tabelul 7 , pentru $\varepsilon = 10^{-p}$, $3 \le p \le 8$, sunt date valorile indicelui n si a sumei partiale S_n . Este confirmat experimental un rezultat teoretic privind sirul de aproximatii succesive S_n ce este monoton crescator la valoarea S.

Table 7. Aproximarea sumei S data de formula (20) prin sumele partiale S_n
(Exemplul 6 ; conditia de oprire a algoritmului de calcul : $S - S_n < \varepsilon$)

ε	n	S_n	ε	n	S_n	ε	n	S_n
1E-03	8	.7492760	1E-04	11	.7499647	1E-05	13	.7499955
1E-06	15	.7499995	1E-07	17	.7499999	1E-08	19	.7500000

Aplicatia 5. Sa estimam valoarea urmatoarei serii infinite

$$S^{(p)} = 1/1^p + 1/2^p + 1/3^p + 1/4^p + \quad + 1/n^p + \qquad (21)$$

cu o eroare ε precizata (de exemplu $\varepsilon = 10^{-8}$) , unde $p \in \mathbf{R}$, $p > 1$.

Suma infinita $S^{(p)}$ este limita sumelor partiale $S_n^{(p)}$,

$$S_n^{(p)} = 1/1^p + 1/2^p + 1/3^p + 1/4^p + \quad + 1/n^p \qquad (22)$$

Intentionam sa aproximam valoarea seriei infinite $S^{(p)}$ printr-o suma partiala $S_n^{(p)}$ Dificultatea majora în aceasta etapa consta în a preciza valoarea indicelui n astfel încât eroarea facuta prin aceasta aproximare sa nu depaseasca un prag ε impus, adica $| S^{(p)} - S_n^{(p)} | < \varepsilon$. Cum seria infinita $S^{(p)}$ are termeni pozitivi în mod evident vom fi satisfacute si inegalitatile $| S^{(p)} - S_{n+k}^{(p)} | < \varepsilon$ pentru orice k numar natural.

Aceasta abordare presupune convergenta sirului sumelor partiale $\{ S_n^{(p)} \}_n$ fapt ce se realizeaza numai în cazul $p > 1$ (a se vedea, de exemplu, Miron Nicolescu, Nicolae Dinculeanu, Solomon Marcus, 1966, p. 698) .

In continuare vom evalua numarul n de termeni din seria infinita $S^{(p)}$ ce vor trebui însumati astfel încât eroarea R_n obtinuta, $R_n = S - S_n^{(p)}$, sa fie în modul mai mica decât o valoare ε fixata. Asadar restul R_n rezultat în procesul de aproximare nu trebuie sa depaseasca un prag ε precizat, adica $R_n = S^{(p)} - S_n^{(p)} \le \varepsilon$. In urma unor calcule elementare deducem :

$$R_n = S^{(p)} - S_n^{(p)} = \frac{1}{(n+1)^p} + \frac{1}{(n+2)^p} + \frac{1}{(n+3)^p} + \ldots + \frac{1}{(n+k)^p} + \ldots <$$

$$< \int_{n}^{n+1} \frac{dx}{x^p} + \int_{n+1}^{n+2} \frac{dx}{x^p} + \int_{n+2}^{n+3} \frac{dx}{x^p} + \ldots + \int_{n+k-1}^{n+k} \frac{dx}{x^p} + \ldots = \int_{n}^{\infty} \frac{dx}{x^p} = \frac{1}{(p-1)n^{p-1}} \qquad (23)$$

Deci, tinând seama de inegalitatea (23), $0 < R_n < 1 / [(p - 1) n^{p-1}]$, rezulta ca $R_n < \varepsilon$ daca $1 / [(p - 1) n^{p-1}] \le \varepsilon$ fapt ce revine la satisfacerea relatiei $n \ge [1 / ((p - 1) \varepsilon)]^{1/(p-1)}$

Prin urmare daca $n \ge] (1 / ((p - 1) \varepsilon))^{1/(p-1)} [+ 1$ atunci $S^{(p)} - S_n^{(p)} < \varepsilon$ unde prin $] a [$ am notat partea întreaga a numarului real a. Asadar pentru a asigura o precizie ε la evaluarea sumei infinite $S^{(p)}$ trebuiesc însumati cel putin primii $] 1 / ((p - 1) \varepsilon) [^{1/(p-1)} + 1$ termeni ai sumei respective.

O noua abordare în situatii speciale.

In anumite cazuri particulare putem imagina si alte proceduri de evaluare a restului R_n, $R_n = S^{(p)} - S_n^{(p)}$. In continuare vom dezvolta o noua tehnica de estimare a marimii restului R_n considerând $p = 2$ si $p = 3$.

Exemplul 7 (cazul p = 2). Pentru $p = 2$ restul $R_n = S^{(2)} - S_n^{(2)}$ este de forma

$$R_n = 1 / (n + 1)^2 + 1 / (n + 2)^2 + 1 / (n + 3)^2 + \ldots + 1 / (n + k)^2 + \qquad (24)$$

si deci sunt satisfacute inegalitatile

$$R_n \le 1 / [n (n+1)] + 1 / [(n+1) (n+2)] + 1 / [(n+2) (n+3)] + \ldots + 1 / [(n+k-1) (n+k)] + \ldots =$$
$$= [1 / n - 1 / (n+1)] + [1 / (n+1) - 1 / (n+2)] + [1 / (n+2) - 1 / (n+3)] + \ldots = 1 / n \qquad (25)$$
$$R_n \ge 1 / [(n+1) (n+2)] + 1 / [(n+2) (n+3)] + \ldots + 1 / [(n+k-1) (n+k)] + \ldots =$$
$$= [1 / (n+1) - 1 / (n+2)] + [1 / (n+2) - 1 / (n+3)] + [1 / (n+3) - 1 / (n+4)] + \ldots = 1 / (n+1)$$

Cele doua inegalitati (25) permit o apreciere a marimii erorii R_n. Daca se doreste o precizie ε la estimarea sumei $S^{(2)}$ atunci din $R_n \le 1 / n \le \varepsilon$ deducem $n \ge 1 / \varepsilon$ fapt ce implica însumarea a cel putin $] 1 / \varepsilon [+ 1$ termeni ai seriei infinite $S^{(2)}$.

Acelasi rezultat poate fi dedus si prin aplicarea variantei bazate pe inegalitatea (23) cu $p = 2$.

Exemplul 8 (cazul p = 3). In cazul $p = 3$ restul R_n verifica inegalitatea

$$R_n = 1 / (n+1)^3 + 1 / (n+2)^3 + 1 / (n+3)^3 + 1 / (n+4)^3 + \ldots + 1 / (n+k)^3 + \ldots \le$$
$$\le 1 / [(n-1) n (n+1)] + 1 / [n (n+1) (n+2)] + 1 / [(n+1) (n+2) (n+3)] + \ldots =$$
$$= \{ 1 / [2 (n-1) n] - 1 / [2 n (n+1)] \} + \{ 1 / [2 n (n+1)] - 1 / [2 (n+1) (n+2)] \} +$$
$$+ \{ 1 / [2 (n+1) (n+2)] - 1 / [2 (n+2) (n+3)] \} + \ldots = 1 / [2 (n-1) n] \qquad (26)$$

Impunând $R_n \le \varepsilon$, din (26) deducem ca pentru a obtine o precizie ε la evaluarea sumei infinite $S^{(3)}$, este suficient sa fie însumati primii n termeni, unde n verifica inegalitatea $1 / [2 (n-1) n] \le \varepsilon$, adica $(n-1) n \ge 1 / (2 \varepsilon)$.

Considerând $p = 3$ în varianta de estimare a restului R_n bazata pe formula (23), deducem

$$n \ge [1 / ((p - 1) \varepsilon)]^{1/(p-1)} = [1 / ((3 - 1) \varepsilon)]^{1/(3-1)} = [1 / (2 \varepsilon)]^{1/2}$$

si deci $n^2 \ge 1 / (2 \varepsilon)$.

Deoarece exista $n \in \mathbf{N}$ astfel încât $n^2 \ge 1 / (2 \varepsilon)$ fara însa ca $(n-1) n \ge 1 / (2 \varepsilon)$ rezulta ca în varianta initiala (23) avem o estimare mai precisa a restului R_n decât în evaluarea (26).

Aplicatia 6. Vom extinde studiul început în Aplicatia 1 dând o noua exprimare restului de

aproximare $R_n(x)$ pentru functia $f(x) = e^x$.

Este cunoscuta dezvoltarea în serie infinita MacLaurin a functiei $f(x) = e^x$, anume :

$$f(x) = e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots + \frac{x^{n-1}}{(n-1)!} + \frac{x^n}{n!} + \frac{x^{n+1}}{(n+1)!} + \qquad (27)$$

Asadar vom aproxima valoarea functiei f(x) în punctul x prin suma finita $S_n(x)$,

$$S_n(x) = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots + \frac{x^{n-1}}{(n-1)!} + \frac{x^n}{n!} \qquad (28)$$

eroarea facuta fiind $R_n(x) = f(x) - S_n(x)$. Evident $S_n(x) < f(x) = e^x$, $\forall\ x \in \mathbf{R}$.

Cum raza de convergenta a seriei infinite f (x) data de formula (27) este $+\infty$ rezulta ca sirul sumelor partiale $S_n(x)$ definite prin (28) converge la valoarea f(x) pentru orice $x \in \mathbf{R}$ (Nicolescu, Dinculeanu, Marcus, 1966, vol. 1, p.756).

Sa evaluam numarul n de termeni ce vor fi însumati în suma partiala $S_n(x)$ definita de relatia (28) astfel încât restul $R_n(x)$ sa nu depaseasca un prag impus ε .

Urmând un calcul elementar deducem succesiv

$$R_n(x) = f(x) - S_n(x) = [\, x^{n+1} / (n+1)! \,] + [\, x^{n+2} / (n+2)! \,] + \ldots + [\, x^{n+p} / (n+p)! \,] + \ldots =$$
$$= [\, x^{n+1} / (n+1)! \,] \{\, 1 + x / (n+2) + x^2 / [\, (n+2)\,(n+3) \,] + \ldots + x^p / [\, (n+2)\,(n+3) \ldots (n+p+1) \,] + \ldots \,\} <$$
$$< [\, x^{n+1} / (n+1)! \,] \{\, 1 + x / (n+2) + x^2 / (n+2)^2 + \ldots + x^p / (n+2)^p + \ldots \,\} =$$
$$= [\, x^{n+1} / (n+1)! \,] \{\, 1 / [\, 1 - x / (n+2) \,] \,\} = a_{n+1} / [\, 1 - x / (n+2) \,] \qquad (29)$$

unde prin $a_n = x^n / n!$ am desemnat termenul general al seriei infinite f(x) data de relatia (27) .

Prin urmare eroarea $R_n(x)$ a aproximarii lui $f(x) = e^x$ prin $S_n(x)$ nu va fi mai mare de ε daca $a_{n+1} / [\, 1 - x / (n+2) \,] < \varepsilon$. Remarcam faptul ca termenii a_n pot fi obtinuti recursiv aplicând formula $a_{n+1} = a_n\, [\, x / (n+1) \,]$, algoritmul de evaluare a valorilor acestor termeni fiind în aceasta ultima varianta mult mai rapid.

Asadar calculul sumei $S_n(x)$ se va realiza adunând pe rând termenii a_0 , a_1 , a_2 , a_3 , ... procesul iterativ oprindu-se în momentul când este satisfacuta inegalitatea

$$a_n / |\, 1 - x / (n+1) \,| < \varepsilon \qquad (30)$$

Observatia 8. Estimatia (29) a restului $R_n(x)$ este diferita de estimatia (6) din Aplicatia 1 .

La implementarea procedurii de estimare a valorii $f(x) = e^x$ nu este nevoie sa se memoreze întregul sir $\{ a_j \}_j$, fiind suficienta retinerea numai a termenului curent a_n . In vederea asigurarii unei precizii sporite în calcule programul BASIC va utiliza variabile reale declarate în precizie dubla.

```
FUNCTION f (x)          ' Calculul functiei exponentiale utilizand dezvoltarea in serie Taylor
eps = 10 ^ (-8) :       n = 0 :        a# = 1 :        s# = 1
DO
n = n + 1 :        a# = a# * x / n :        s# = s# + a# :        LOOP UNTIL a# < eps
f = s# :                END FUNCTION
```

Stiind ca valoarea expresiei $E(x,y) = e^x \cdot e^y - e^{x+y}$ este nula pentru orice $x , y \in \mathbf{R}$, putem verifica indirect precizia rezultata prin folosirea polinoamelor Taylor $S_n(x)$ în vederea aproximarii expresiei $f(x) = e^x$. In acest sens vor fi calculate valorile expresiei $E^*(x,y) = S_n(x)\, S_n(y)$

- $S_n(x+y)$.

```
DECLARE FUNCTION f (x!)          ' Evaluarea valorilor functiei  f(x) = exp(x)
REM  Verificarea identitatii     exp(x) * exp(y)  =  exp(x+y)
FOR x = .5 TO 1.5 STEP .5 :              FOR y = .2 TO .9 STEP .2
PRINT USING "F(#.#) * F(#.#) - F(#.#) = ###.##### * ###.##### - ###.#####  =  ##.########"; x;
y; x + y; f(x); f(y); f(x + y); f(x) * f(y) - f(x + y)
NEXT y :                 NEXT x
```

Rezultatele obtinute sunt listate în Tabelul 8. Urmarind valorile expresiei E*(x,y) ce sunt aproape nule, $E^*(x,y) = S_n(x)\, S_n(y) - S_n(x+y)$, este confirmata corectitudinea procedurii ce utilizeaza dezvoltarea în serie Taylor (28) pentru aproximarea functiei $f(x) = e^x$.

Table 8. Evaluarea expresiei $E^*(x,y) = (S_n(x)\, S_n(y)) / S_n(x+y)$ (functia $S_n(x)$ este calculata cu formula (28) ; oprirea algoritmului este dictata de inegalitatea (30))

x	y	E*(x,y)	x	y	E*(x,y)	x	y	E*(x,y)
0.5	0.2	-0.00000005	0.5	0.4	0.00000004	0.5	0.6	-0.00000007
0.5	1.3	-0.00000004	1.0	0.2	-0.00000016	1.0	1.4	-0.00000012
1.0	0.6	-0.00000001	1.0	0.8	-0.00000019	1.5	0.2	-0.00000022
1.5	0.4	0.00000003	1.5	0.6	0.00000092	1.5	0.8	-0.00000012

Observatia 9. Rezultate asemanatoare au fost deja obtinute la aproximarea lui e^x prin dezvoltarea acestei functii în fractie continua infinita (a se vedea exemplul corespunzator din capitolul referitor la fractiile continue). Se vor compara în acest sens valorile expresiei $E^*(x,y) = g(x)\, g(y) - g(x+y)$ atunci când expresia g (x) este, pe rând, polinomul de aproximare Taylor de grad n (formula (28)), respectiv redusa de ordin n a functiei e^x . Bineînteles, rezultatele deduse în cele doua variante nu sunt identice.

Aplicatia 7. Sa estimam valoarea sumei infinite

$$S = 1 + [\,1/2\,] + [\,1/3\,] + [\,1/4\,] + \ldots + [\,1/n\,] + [\,1/(n+1)\,] + \ldots \qquad (31)$$

Urmatorul program BASIC îsi propune sa calculeze sume partiale S_n finite,

$$S_n = 1 + [\,1/2\,] + [\,1/3\,] + [\,1/4\,] + [\,1/5\,] + \ldots + [\,1/n\,] \qquad (32)$$

pentru diverse valori ale parametrului n :

```
REM  Calculul sumei  s = 1 + [ 1/2 ] + [ 1/3 ] + [ 1/4 ] + ...
u = 0 :              p = 500000 :          s = 0
1 : u = u + 1 :              s = s + 1 / u
IF u < p THEN GOTO 1
PRINT "n = "; u; "      s = "; s :          p = p + 500000 :          GOTO 1
```

Suma infinita S va fi limita sumelor finite S_n atunci când valoarea variabilei n tinde la ∞

Rezultatele obtinute prin rularea programului anterior sunt listate în Tabelul 9.

Urmarind datele din Tabelul 9 se observa o "stabilizare" la valoarea 15.40368 a sumelor partiale S_n . Cum sirul sumelor partiale S_n este crescator si (aparent) marginit concluzionam

convergenta acestui sir la o valoare $v \approx 15.40368$. Deci valoarea sumei infinite S definita de (31) este aproximativ 15.40368 .

Aceasta afirmatie este însa eronata.

In adevar, aplicând succesiv inegalitatea

$$[1/(2^p+1)] + [1/(2^p+2)] + [1/(2^p+3)] + \ldots + [1/(2^p+2^p)] > 2^p/2^{p+1} = 1/2 \qquad (33)$$

pentru sirul termenilor sumei infinite S definita de (31) deducem

$$S > 1 + [1/2] + [1/2] + [1/2] + \ldots + [1/2] + \ldots = \infty \qquad (34)$$

adica $S = \infty$.

Remarcam faptul ca sirul sumelor partiale S_n (formula (32)) creste lent (Tabelul 9).

Tabelul 9. Valoarea sumei S_n data de formula (32)

n	S_n	n	S_n	n	S_n	n	S_n
10	2.92897	100	5.18738	1000	7.48548	10000	9.78761
100000	12.09085	500000	13.69069	1000000	14.35736	1500000	14.83420
2000000	15.31103	2500000	15.40368	3000000	15.40368	3500000	15.40368
4000000	15.40368	10000000	15.40368				

Observatia 10. Valorile sumelor partiale S_n sunt retinute de calculator cu un numar fix de cifre semnificative. Odata cu cresterea gradului n creste si ordinul de marime al sumei S_n . Cum $S_{n+1} = S_n + [1/(n+1)]$, teremenul $1/(n+1)$ nou introdus descreste atât de mult încât, la un moment dat, printr-o operatie de adunare el nu mai poate afecta cifrele semnificative ale sumei S_n .

Asadar pentru valori mari ale indicelui n termenul $1/(n+1)$ are efectiv o influenta nula la calcularea sumei $S_{n+1} = S_n + 1/(n+1)$. Practic rezulta egalitatea $S_{n+1} = S_n$ când de fapt $S_n < S_{n+1} = S_n + 1/(n+1)$.

3. Estimarea radacinii de ordin p

Vom studia si o alta tehnica utilizata în evaluarea valorilor unor functii.

Pentru $p > 1$, $p \in \mathbf{N}$, intentionam sa calculam radacina de ordin p dintr-un numar real $a \geq 0$

Fie functia $h : [0, \infty) \to [0, \infty)$ definita prin formula

$$h(t) = [(p-1)\,t + a/t^{p-1}]/p \qquad (35)$$

Functia h (t) are punctul fix $a^{1/p}$ deoarece $h(a^{1/p}) = [(p-1)\,a^{1/p} + a/a^{(p-1)/p}]/p = a^{1/p}$

Cazul a > 1 .

Considerând $x_0 = a$, construim iterativ sirul $\{ x_n \}_n$, unde

$$x_{n+1} = h(x_n) = [(p-1)\,x_n + a/x_n^{p-1}]/p \qquad (36)$$

Sa demonstram prin inductie ca sirul $\{ x_n \}_n$ definit de relatia (36) este descrescator, adica

$a = x_0 \geq x_1 \geq x_2 \geq x_3 \geq \ldots \geq x_{n-1} \geq x_n \geq x_{n+1} \geq \ldots$ In plus $x_n \geq a^{1/p}$, $\forall\ n \in \mathbf{N}$.

Cum $a > 1$ si $p > 1$ deducem succesiv inegalitatile $a^{p-1} > 1$; $(p-1)\,a + a^{2-p} < p\,a$ si deci

$x_1 = h(x_0) = h(a) = [(p-1)\,a + a/a^{p-1}]/p = [(p-1)\,a + a^{2-p}]/p < a = x_0$.

Derivata $h^{(1)}(t)$ a functiei $h(t)$ are expresia

$h^{(1)}(t) = [(p-1) + (1-p)\, a / t^p] / p = [(p-1) / p] [1 - a / t^p]$

Asadar $h^{(1)}(t) > 0$ atunci când $t^p > a$.

Prin urmare functia h este strict crescatoare pentru orice $t > a^{1/p}$ si strict descrescatoare daca $t < a^{1/p}$.

Din inegalitatile $1 < a^{1/p} < a = x_0$, cum functia $h(t)$ este strict crescatoare pe intervalul $[a^{1/p}, \infty)$, rezulta ca $x_1 = h(x_0) > h(a^{1/p}) = a^{1/p}$

In concluzie $x_0 > x_1 > a^{1/p}$, etapa initiala în procedura de inductie fiind astfel verificata.

Presupunem ca pentru o valoare data a indicelui n sunt adevarate inegalitatile $x_{n-1} > x_n > a^{1/p}$

Utilizând proprietatea de monotonie a functiei $h(t)$ deducem ca $h(x_{n-1}) > h(x_n) > h(a^{1/p})$ si deci $x_n > x_{n+1} > a^{1/p}$ fapt ce justifica etapa iterativa a procedurii de inductie.

Prin urmare sirul $\{ x_n \}_n$ definit de relatia (36) este un sir monoton si marginit, deci un sir convergent la o valoare finita c, $c \geq a^{1/p}$

Facând ca indicele n sa tinda la infinit în egalitatea $x_{n+1} = [(p-1)\, x_n + a / x_n^{p-1}] / p$ rezulta ca limita c a sirului $\{ x_n \}_n$ satisface relatia $c = [(p-1)\, c + a / c^{p-1}] / p$ adica $c = a / c^{p-1}$

Prin urmare :

Propozitia 1. Pentru $a > 1$, sirul $\{ x_n \}_n$ cu $x_0 = a$ si $x_{n+1} = [(p-1)\, x_n + a / x_n^{p-1}] / p$ este convergent (descrescator) la radacina de ordin p din a.

Cazul $0 < a < 1$.

Observatia 11. Propozitia 1 ar putea fi folosita la obtinerea unei aproximatii a radacinii de ordin p din numarul real a si în cazul $0 < a < 1$. In adevar, utilizarea egalitatii $a^{1/p} = 1 / [1 / a]^{1/p}$ va permite reducerea cazului $0 < a < 1$ la varianta $a > 1$ ($a > 1$ daca si numai daca $0 < 1/a < 1$).

Situatiile $a = 1$ sau $a = 0$ vor fi tratate separat, cazuri în care, în mod evident, valoarea lui $a^{1/p}$ este cunoscuta ($a^{1/p} = a$).

Observatia 12. Daca $0 < a < 1$, atunci sirul $\{ x_n \}_n$ definit recurent prin formulele $x_0 = a$ si $x_{n+1} = [(p-1)\, x_n + a / x_n^{p-1}] / p$ nu este neaparat un sir monoton (se constata $x_0 < x_1 > x_2$).

Printr-un rationament asemanator vom cauta sa extindem Propozitia 1 si în varianta $0 < a < 1$.

Astfel vom demonstra ca pentru $0 < a < 1$ termenii sirului $\{ x_n \}_n$ verifica inegalitatile

$a = x_0 \leq x_1 \geq x_2 \geq x_3 \geq \ldots \geq x_{n-1} \geq x_n \geq \ldots \geq a^{1/p}$

In adevar, pentru $p > 1$, cum functia $h(t) = [(p-1)\, t + a / t^{p-1}] / p$ este strict descrescatoare atunci când $t \in (0, a^{1/p})$, din $a \leq a^{1/p}$ rezulta $h(a) \geq h(a^{1/p}) = a^{1/p}$. Deci

$$x_1 = h(x_0) = h(a) \geq a^{1/p} \geq a = x_0$$

Functia $h(t)$ fiind însa strict crescatoare pe intervalul $(a^{1/p}, \infty)$, din $x_1 \geq a^{1/p}$ deducem si inegalitatea $x_2 = h(x_1) \geq h(a^{1/p}) = a^{1/p}$

Construim functia $h_1(t) = t - h(t) = [t - a / t^{p-1}] / p$ ce este crescatoare pe intervalul $(0, \infty)$ si deci $h_1(t) \geq h_1(a^{1/p}) = 0$ pentru orice $t \geq a^{1/p}$ adica $t \geq h(t)$, $\forall\ t \in [a^{1/p}, \infty)$

Cum $x_1 \geq a^{1/p}$ obtinem si $x_1 \geq h(x_1) = x_2$

Aplicând functia strict crescatoare $h(t)$, $t \geq a^{1/p}$, dublei inegalitati $x_1 \geq x_2 \geq a^{1/p}$ rezulta în final, dupa n-1 pasi, ca $x_n \geq x_{n+1} \geq a^{1/p}$, $\forall\ n \in \mathbf{N}$, $n \geq 1$.

In concluzie, pentru $n \geq 1$ sirul $\{ x_n \}_n$ este descrescator si marginit, fiind convergent la $a^{1/p}$

Propozitia 2. Daca $0 < a < 1$, atunci sirul $\{ x_n \}_n$ definit recurent prin formula (36) cu $x_0 = a$ este convergent la radacina de ordin p din a.

O analiza separata a cazului $a = 1$ împreuna cu rezultatele Propozitiilor 1 si 2 ne conduc la

Teorema 1. Sirul $\{ x_n \}_n$, unde $x_0 = a$ si $x_{n+1} = [(p-1)\, x_n + a / x_n^{p-1}] / p$, este convergent la $a^{1/p}$, $\forall\ a \in \mathbf{R}$, $a > 0$.

Aplicatia 8. Teorema 1 sta la baza algoritmului **Rad (p)** pentru calculul radacinii de ordin p dintr-un numar real a, strict pozitiv. Esential este de stabilit numarul n de iteratii ce trebuiesc efectuate astfel încât eroarea aproximarii lui $a^{1/p}$ prin x_{n+1} sa nu depaseasca un prag impus ε, adica $| x_{n+1} - a^{1/p} | < \varepsilon$ ($\varepsilon > 0$ ia valori în vecinatatea lui zero, de exemplu $\varepsilon = 10^{-8}$).

Necunoscând însa valoarea lui $a^{1/p}$ nu este posibila compararea directa a termenului x_{n+1} cu $a^{1/p}$. Putem accepta însa un compromis, anume oprirea algoritmului **Rad (p)** în momentul în care modulul diferentei dintre doi termeni consecutivi ai sirului recurent $\{ x_n \}_n$ definit de (36) este suficient de mica, de exemplu $| x_{n+1} - x_n | < \varepsilon$.

Listam în pseudocod algoritmul **Rad (p)**

Algoritmul Rad (p) (Aproximarea radacinii de ordin p din numarul $a > 0$ cu o eroare ε)

Pas 0.	Input a	$\varepsilon = 10^{-8}$ $\quad$ x2 = a
Pas 1.	x1 = x2	x2 = [(p - 1) x1 + a / x1^(p-1)] / p
Pas 2.	Daca | x2 - x1 | > ε	atunci Mergi la Pasul 1
	rad = x2	Tipareste rad (radacina de ordin p a numarului strict pozitiv a)

In algoritmul prezentat nu este nevoie de a se retine toate valorile $x_0, x_1, x_2, \ldots, x_{n-1}, x_n, x_{n+1}$, în calcule fiind folosite întotdeauna numai doua valori succesive x_j si x_{j+1}, $0 \leq j \leq n$, valori ce vor fi memorate temporar în variabilele x1, respectiv x2.

Decizia de oprire a calculului iterarativ în momentul satisfacerii inegalitatii $| x_{n+1} - x_n | < \varepsilon$ pare hazardata, cel putin la prima vedere. Astfel putem avea o diferenta dintre doi termeni consecutivi x_n si x_{n+1} mai mica decât ε si totusi termenul x_{n+1} sa fie destul de departat de limita sa $a^{1/p}$ Vom arata ca o asemenea situatie nu este însa posibila în cazul algoritmului **Rad (p)**.

Justificarea modului de oprire a algoritmului Rad (p)

Din demonstratiile Propozitiilor 1-2 a rezultat ca pentru orice $a > 0$ avem $x_1 > x_2 > x_3 > \ldots > x_{n-1} > x_n > \ldots > a^{1/p}$ si în plus sirul $\{ x_n \}_n$ este convergent la valoarea $a^{1/p}$

Utilizând teorema lui Lagrange vom estima diferenta dintre doi termeni consecutivi ai sirului $\{ x_n \}_n$ definit de (36),

$$x_n - x_{n+1} = h(x_{n-1}) - h(x_n) = h^{(1)}(\xi)\,(x_{n-1} - x_n) = [(p-1)/p]\,(1 - a/\xi^p)\,(x_{n-1} - x_n) <$$
$$< [(p-1)/p]\,(1 - a^{1-p})\,(x_{n-1} - x_n) = d\,(x_{n-1} - x_n) < d^2\,(x_{n-2} - x_{n-1}) < d^3\,(x_{n-3} - x_{n-2}) <$$

$< \ldots < d^{n-1} (x_1 - x_2)$

unde $x_n < \xi < x_{n-1}$ si $0 \le d = [(p - 1) / p] (1 - a^{1-p}) < 1$.

Sa evaluam distanta dintre x_n si $a^{1/p}$

$$x_n - a^{1/p} = (x_n - x_{n+1}) + (x_{n+1} - x_{n+2}) + (x_{n+2} - x_{n+3}) + \ldots + (x_{n+k-1} - x_{n+k}) + \ldots <$$
$$< (x_n - x_{n+1}) + d (x_n - x_{n+1}) + d^2 (x_n - x_{n+1}) + \ldots + d^k (x_n - x_{n+1}) + \ldots =$$
$$= (x_n - x_{n+1}) [1 + d + d^2 + d^3 + \ldots + d^k + \ldots] = (x_n - x_{n+1}) / (1 - d)$$

Asadar folosirea în algoritmul **Rad (p)** a criteriului de oprire $| x_{n+1} - x_n | < \varepsilon$ este justificata deoarece în acest caz avem

Propozitia 3. Pentru sirul $\{ x_n \}_n$ definit recurent de relatiile (36) cu $x_0 = a$, daca $0 \le d = [(p - 1) / p] (1 - a^{1-p}) < 1$ atunci

$$(x_n - a^{1/p}) < (x_n - x_{n+1}) / (1 - d) < \varepsilon / (1 - d) \qquad (37)$$

Este util de a se studia **viteza de convergenta** a termenilor sirului $\{ x_n \}_n$ catre limita $a^{1/p}$

Astfel, presupunând ca x_n aproximeaza pe $a^{1/p}$ cu o eroare relativa δ , $x_n = a^{1/p} (1 + \delta)$ rezulta $x_{n+1} = [(p - 1) x_n + a / x_n^{p-1}] / p = a^{1/p} [(p - 1) (1+ \delta) + (1 + \delta)^{1-p}] / p$

Utilizând dezvoltarea functiei $h_2(\delta) = (1 + \delta)^{1-p}$ în serie Taylor în jurul punctului $\delta = 0$, deducem $(1 + \delta)^{1-p} \approx 1 - (p - 1) \delta + p (p - 1) \delta^2 / 2$ si deci

$$x_{n+1} \approx a^{1/p} [(p - 1) (1+ \delta) + 1 - (p - 1) \delta + p (p - 1) \delta^2 / 2] / p = a^{1/p} [1 + (p - 1) \delta^2 / 2] \qquad (38)$$

Cum $x_n = a^{1/p} (1 + \delta)$, din relatia (38) rezulta

Propozitia 4. Pentru $| \delta | < 1$ termenul x_{n+1} difera de limita sa $a^{1/p}$ cu o eroare relativa $(p - 1) \delta^2 / 2$, eroare ce este substantial mai mica decât eroarea relativa δ ce caracterizeaza deosebirea dintre x_n si $a^{1/p}$.

Aplicatia 9 (cazul particular p = 2). Daca $p = 2$ atunci sirul $\{ x_n \}_n$, cu $x_0 = a$ si $x_{n+1} = (x_n + a / x_n) / 2$, converge la radacina patrata a numarului a , $a > 0$.

Programul urmator aproximeaza cu o eroare $\varepsilon = 10^{-8}$, radacina patrata a numerelor naturale $1 \le x \le 9$,

```
REM  Program pentru calculul radacinii patratice dintr-un numar real  x
DECLARE FUNCTION rad2! (a!)
FOR x = 1! TO 9! :                  y = rad2(x)
PRINT USING "[ ## ^ (1/2) ] * [ ## ^ (1/2) ] - ## = [ #.######## ] * [ #.######## ] - ## =
####^^^^ "; x; x; x; y; y; x; y * y - x :                    NEXT x
FUNCTION rad2 (a)       ' Calculul radacinii patratice  rad2  dintr-un numar real  a > 0
eps = 10 ^ (-8) :           x2 = a
DO
x1 = x2 :           x2 = (x1 + a / x1) / 2! :           LOOP UNTIL ABS(x2 - x1) < eps
rad2 = x2 :           END FUNCTION
```

Rezultatele obtinute prin aplicarea functiei **rad2** (a) = g(a) sunt verificate indirect prin listarea valorilor expresiei $E_1(a) = g(a) g(a) - a$. Precizam faptul ca expresia $E_1(a)$ este nula pentru orice $a \ge$

0 atunci când functia g(a) este chiar $a^{1/2}$. Datorita acceptarii în algoritmul **rad2** de extragere a radacinii patratice o eroare de aproximare ε, expresia $E_1(a)$ nu va fi neaparat nula, dar va trebui sa ia valori foarte aproape de zero. Acest lucru este confirmat de rezultatele sintetizate în Tabelul 10.

Observatia 13. In algoritmul **rad2** din Aplicatia 9 este foarte important sa initializam termenul x_0 chiar cu valoarea a deoarece în aceasta situatie s-a demonstrat teoretic convergenta sirului $\{ x_n \}_n$ (Teorema 1). In cazul în care se atribuie lui x_0 o valoare arbitrara atunci convergenta acestui sir s-ar putea sa nu fie asigurata.

Astfel prin initializarea lui x_0 cu o valoare strict negativa, cum $x_{n+1} = (x_n + a / x_n) / 2$, $a > 0$, rezulta ca $x_n < 0$ pentru orice $n \in N$. Prin urmare sirul $\{ x_n \}_n$ ce are numai termeni negativi nu poate sa convearga la $a^{1/2} > 0$.

Dupa o analiza corespunzatoare procedura **rad(p)** va putea fi adaptata cu succes si altor situatii, ca de exemplu : α) p impar si $a \in \mathbf{R}$ (nu neaparat $a \geq 0$), β) $p > 1$, $p \in \mathbf{R}$.

Tabelul 10. Valoarea expresiei $E_1(a) = (g(a))^2 - a$, unde $g(a)$ = **rad2 (a)** (Aplicatia 9)

a	$E_1(a)$	a	$E_1(a)$	a	$E_1(a)$	a	$E_1(a)$	a	$E_1(a)$
1	0E+00	2	-685E-10	3	-108E-09	4	0E+00	5	147E-09
6	437E-09	7	-387E-09	8	-274E-09	9	0E+00		

4. Estimarea valorii unui polinom într-un punct dat

Vom aborda problema estimarii valorii $P_n(c)$ a polinomului $P_n(x)$, de gradul n cu coeficienti reali, în punctul c, $c \in R$, unde

$$P_n(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_j x^{n-j} + \ldots + a_{n-1} x + a_n \qquad (39)$$

Aplicatia 10. Vom proiecta în limbajul BASIC doua programe diferite, ambele urmând sa evalueze valoarea polinomului

$$P_n(x) = 1 - 3 x + 5 x^2 - 2 x^{n-1} + x^n \qquad (40)$$

de gradul $n \geq 4$, în diferite puncte c.

Se observa ca $P_n(1) = 2$ si $P_n(2) = 15$ pentru orice numar natural n, $n \geq 4$.

Practic aceste egalitati ar putea sa nu se realizeze datorita erorilor de calcul, valorile numerice fiind retinute în calculator cu un numar fix de cifre semnificative.

Exemplul 9. In programul urmator calculele aritmetice pentru evaluarea lui $P_n(x)$ definit de (40) se vor face începând cu puterile cele mai mici ale variabilei x. In acest caz, datorita erorilor aritmetice de rotunjire specifice calculatorului, valorile $P_n(2)$ vor fi dependente de gradul n al polinomului atunci când n devine prea mare. Astfel prin rularea programului **MB** (Metoda Bruta) se obtin rezultate în sprijinul afirmatiilor facute (Tabelul 11).

```
REM  Calculul valorii polinomului (40) - Metoda bruta
FOR n = 40 TO 84 STEP 4
x = 1 :              p = 1 - 3 * x + 5 * x ^ 2 - 2 * x ^ (n - 1) + x ^ n
```

```
PRINT "P"; n; "("; x; ")="; p; "  ,  ";
x = 2 :                p = 1 - 3 * x + 5 * x ^ 2 - 2 * x ^ (n - 1) + x ^ n
PRINT "P"; n; "("; x; ")="; p; "  ;  "; :                NEXT n
```

Datele aparent contradictorii din Tabelul 11 se explica prin faptul ca adunând valoarea 15 la cantitatea 2^n se obtine practic tot valoarea 2^n atunci când n este mare. Acest lucru este datorat imposibilitatii operarii calculatorului cu un numar nelimitat de cifre semnificative.

Tabelul 11. Valoarea polinomului $P_n(x)$ (forma (40)) estimata cu procedura **MB** (Metoda Bruta ; gradul n al polinomului $P_n(x)$ este variabil)

n	$P_n(1)$	$P_n(2)$	n	$P_n(1)$	$P_n(2)$	n	$P_n(1)$	$P_n(2)$
40	2	15	44	2	15	48	2	15
52	2	15	56	2	15	60	2	15
64	2	15	68	2	0	72	2	0
76	2	0	80	2	0	84	2	0

Exemplul 10. Diminuarea erorilor de rotunjire se poate realiza efectuând operatii aritmetice cu valori comparabile. Astfel utilizând proprietatile de comutativitate si asociativitate ale operatiilor aritmetice de adunare si înmultire vom modifica ordinea de efectuare a calculelor pentru estimarea valorii polinomului $P_n(x)$. Vom demara procesul de calcul luînd în considerare puterile cele mai mari ale variabilei x , operând pe cât posibil cu valori relativ comparabile. Propunem astfel urmatoarea desfasurare a operatiilor :

$$P_n(x) = x^{n-1} (x - 2) + x (5 x - 3) + 1 \qquad (41)$$

Prin modificarea programului BASIC anterior în forma **RC** (Reorientarea Calculelor) se obtin de aceasta data rezultatele reale, fara eroare (Tabelul 12).

```
REM  Calculul valorii polinomului (40) scris in forma (41) - Reorientarea calculelor
FOR n = 40 TO 84 STEP 4
x = 1 :                p = (x ^ (n - 1)) * (x - 2) + x * (5 * x - 3) + 1
PRINT "P"; n; "("; x; ")="; p; "  ,  ";
x = 2 :                p = (x ^ (n - 1)) * (x - 2) + x * (5 * x - 3) + 1
PRINT "P"; n; "("; x; ")="; p; "  ;  "; :                NEXT n
```

Tabelul 12. Valoarea polinomului $P_n(x)$ (forma (41)) estimata cu procedura **RC** (Reorientarea Calculelor ; gradul n al polinomului $P_n(x)$ este variabil)

n	$P_n(1)$	$P_n(2)$	n	$P_n(1)$	$P_n(2)$	n	$P_n(1)$	$P_n(2)$
40	2	15	44	2	15	48	2	15
52	2	15	56	2	15	60	2	15
64	2	15	68	2	15	72	2	15
76	2	15	80	2	15	84	2	15

Aplicatia 11. Pentru marirea preciziei în calcule, procesul de estimare a valorii $P_n(c)$ pentru un polinom oarecare $P_n(x)$, de gradul n,

$$P_n(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_j x^{n-j} + \ldots + a_{n-1} x + a_n \qquad (42)$$

se va efectua urmând o schema de calcul de tipul

$$P_n(c) = (c (c (\ldots (c (c (c (c a_0 + a_1) + a_2) + a_3) + a_4) + \ldots + a_{n-2}) + a_{n-1}) + a_n) \qquad (43)$$

Incepând în (43) cu grupul de paranteze interioare, vom desemna succesiv continutul dintre orice doua paranteze rotunde corespunzatoare prin b_1, b_2, ..., b_{n-1}, respectiv b_n. Deducen astfel egalitatile :

$$\begin{aligned} b_0 &= a_0 \\ b_1 &= c \cdot b_0 + a_1 \\ b_2 &= c \cdot b_1 + a_2 \\ &\ldots\ldots\ldots\ldots \\ b_j &= c \cdot b_{j-1} + a_j \\ &\ldots\ldots\ldots\ldots \\ b_n &= c \cdot b_{n-1} + a_n \end{aligned} \qquad (44)$$

Valoarea b_n a expresiei delimitata de ultimul grup de paranteze a expresiei (43) este chiar $P_n(c)$, adica valoarea polinomului $P_n(x)$ în punctul c.

Relatiile iterative (44) ce definesc coeficientii b_j, $0 \le j \le n$, pot servi la estimarea valorii polinomului $P_n(x)$ în punctul $x = c$.

Aplicatia 12. Sa determinam câtul si restul împartirii polinomului $P_n(x)$ la $(x - c)$.

Printr-un calcul elementar se verifica egalitatea

$$P_n(x) = (x - c) Q_{n-1}(x) + b_n \qquad (45)$$

unde coeficientii $b_0, b_1, b_2, \ldots, b_{n-1}, b_n$ sunt definiti de relatiile (44) iar câtul $Q_{n-1}(x)$ are forma

$$Q_{n-1}(x) = b_0 x^{n-1} + b_1 x^{n-2} + b_2 x^{n-3} + \ldots + b_{n-2} x + b_{n-1} \qquad (46)$$

Asadar coeficientii b_j, $0 \le j \le n$, ce satisfac relatiile (44) caracterizeaza câtul $Q_{n-1}(x)$ si restul b_n ce rezulta prin împartirea polinomului $P_n(x)$ dat de formula (42) la polinomul $(x - c)$ de gradul unu .

Procedura recurenta (44) folosita la obtinerea coeficientilor b_j, $0 \le j \le n$, este cunoscuta sub numele de "schema lui Horner".

Listam în continuare programul BASIC conceput pentru determinarea coeficientilor câtului Q_{n-1} precum si evaluarea restului b_n al împartirii polinomului $P_n(x)$ la polinomul $x - c$:

```
REM  Calculul citului si restului impartirii unui polinom  P(x)  la polinomul  x -  c
REM  Schema lui  Horner
CLS :              LOCATE 24, 35 :              INPUT "Gradul polinomului = "; n
DIM a(n), b(n) :        LOCATE 24, 35 :        PRINT "Definirea coeficientilor polinomului  P"
FOR j = 0 TO n :              LOCATE 24, 35 :              PRINT "a("; j; ")= ";
INPUT a(j) :             NEXT j
LOCATE 24, 15 :                INPUT "Impartirea polinomului  P  la  x  -  c      ,    c = "; c
REM  calculul coeficientilor citului si restului
b(0) = a(0) :          FOR j = 1 TO n :          b(j) = c * b(j - 1) + a(j)          NEXT j
LOCATE 1, 1 :        PRINT "Restul impartirii polinomului  P(x)  la  x  - "; c; "  este "; b(n)
LOCATE 2, 1 :        PRINT "Coeficientii citului impartirii polinomului  P(x)  la  x  - "; c; "  sunt  "
```

```
FOR j = 0 TO n - 1 :        PRINT "b("; j; ") = "; b(j); "  ,  "; :        NEXT j
```

Exemplul 11. In urma rularii programului BASIC prezentat în Aplicatia 12 s-a testat împartirea polinomului $P_3(x) = x^3 - 2x^2 + 3x + 5$ la $x - 1$ obtinându-se restul $b_3 = 7$ si câtul $Q_2(x) = x^2 - x + 2$.

Observatia 14. Un studiu similar poate fi dezvoltat în cazul în care polinomul $P_n(x)$ are coeficienti numere complexe. La implementarea acestei variante în limbajul BASIC vor trebui însa proiectate subrutinele de efectuarea calculelor aritmetice cu numere complexe (un numar complex fiind asimilat ca o pereche de numere reale).

APROXIMAREA FUNCTIILOR

1. Formule de medie

Vom analiza mai multe formule de aproximare ale unor expresii, formule des folosite în practica.

Notatie. Desemnam prin $C^{(k)}(D)$, $D \subset R$, multimea functiilor cu valori reale ce sunt definite pe domeniul D si care au derivata de ordin k continua pe D.

Pentru simplificare vom folosi si notatia $C(D) \equiv C^{(0)}(D)$ pentru a desemna multimea functiilor reale ce sunt continue în orice punct din domeniul $D \subset R$.

Propozitia 1. Daca $f \in C([a, b])$ atunci pentru orice puncte $x_1, x_2, x_3, \ldots, x_n$ ce apartin intervalului închis $[a, b]$ si pentru orice "ponderi" $p_j \geq 0$, cu $p_1 + p_2 + p_3 + \ldots + p_n = 1$, exista $c \in [a, b]$ astfel încât

$$p_1 f(x_1) + p_2 f(x_2) + p_3 f(x_3) + \ldots + p_{n-1} f(x_{n-1}) + p_n f(x_n) = f(c) \qquad (1)$$

Demonstratie. Avem inegalitatile $M_1 \leq E \leq M_2$ si $M_1 \leq f(x) \leq M_2$ pentru orice $x \in [a, b]$, unde

$$M_1 = \text{minim}\{ f(x) \mid a \leq x \leq b \} \qquad M_2 = \text{maxim}\{ f(x) \mid a \leq x \leq b \}$$

$$E = p_1 f(x_1) + p_2 f(x_2) + p_3 f(x_3) + \ldots + p_{n-1} f(x_{n-1}) + p_n f(x_n)$$

Functia f fiind continua pe un interval închis $[a, b]$ deducem ca cele doua margini M_1, M_2 sunt efectiv atinse, adica exista $c_j \in [a, b]$ astfel încât $M_j = f(c_j)$, $j = 1, 2$.

Cum $f(c_1) = M_1 \leq E \leq M_2 = f(c_2)$, conform proprietatii lui Darboux rezulta $c \in [a, b]$ astfel încât $f(c) = E$, fapt ce justifica propozitia enuntata.

Acelasi rationament poate fi aplicat si în cazul unor ponderi q_j, $1 \leq j \leq n$, nenegative, a caror suma nu este neaparat 1. Considerând în Propozitia 1 $p_j = q_j / (q_1 + q_2 + \ldots + q_n)$ deducem

Propozitia 2. Pentru orice $f \in C([a, b])$, $x_j \in [a, b]$, $q_j \geq 0$ cu $1 \leq j \leq n$ exista $c \in [a, b]$ astfel încât urmatoarea egalitate este adevarata

$$q_1 f(x_1) + q_2 f(x_2) + \ldots + q_{n-1} f(x_{n-1}) + q_n f(x_n) = (q_1 + q_2 + \ldots + q_n) f(c) \qquad (2)$$

Observatia 1. Din (2), pentru $q_j = 1 / n$, $1 \leq j \leq n$, obtinem o formula de medie des utilizata în aplicatii,

$$f(x_1) + f(x_2) + f(x_3) + \ldots + f(x_{n-1}) + f(x_n) = n f(c) \qquad (3)$$

pentru $c \in [a, b]$.

Observatia 2. Formula (2) poate sa nu fie adevarata daca ponderile q_j nu sunt toate nenegative. In adevar pentru $f(x) = 1 / (x+1)$, $a = 1/2$, $b = 5/2$, $n = 2$, $x_1 = 1$, $x_2 = 2$, $q_1 = 2$, $q_2 = -7$ nu poate exista relatia (2), anume $q_1 f(x_1) + q_2 f(x_2) = (q_1 + q_2) f(c)$, deoarece rezolvând aceasta ecuatie în necunoscuta c se obtine $c = 11 / 4 \notin [a, b] \equiv [1/2, 5/2]$.

2. Aproximarea derivatelor

Fie $a \in D$ un punct arbitrar pentru care intentionam sa estimam valoarea $f^{(m)}(a)$ a derivatei de ordinul m a functiei $f(x)$. Vom utiliza si notatia $f^{(0)}(a)$ pentru a desemna valoarea $f(a)$ a functiei $f(x)$ în punctul a. Daca $n \in N^*$ atunci

$$f^{(m)}(a) = \lim_{h \to 0} \frac{f^{(m-1)}(a+h) - f^{(m-1)}(a)}{h} \qquad (4)$$

si în particular

$$f^{(1)}(a) = \lim_{h \to 0} \frac{f^{(0)}(a+h) - f^{(0)}(a)}{h} = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h} \qquad (5)$$

Considerând o valoare foarte mica atribuita lui h putem aproxima limita (5) prin diferenta

$$f^{(1)}(a) \approx [\, f(a+h) - f(a) \,] / h \qquad (6)$$

Formula (6) nu este singura aproximatie posibila a derivatei functiei f(x) calculata în punctul a . Astfel vom opera si cu aproximatia

$$f^{(1)}(a) \approx [\, f(a + h/2) - f(a - h/2) \,] / h \qquad (7)$$

In practica este util de a alege aproximatia cea mai buna. Va trebui asadar sa comparam cele doua aproximatii ale derivatei $f^{(1)}(a)$ date de formulele (6) si (7).

Propozitia 3. Daca $f \in C^{(2)}([\, a, a+h \,])$ atunci

$$| f^{(1)}(a) - [\, f(a+h) - f(a) \,] / h | \le (h/2) \sup \{ | f^{(2)}(x) | , a \le x \le a+h \} \qquad (8)$$

Demonstratie. Vom evalua "eroarea" facuta prin utilizarea aproximarii (6).

Utilizând dezvoltarea în serie Taylor cu rest de ordin 2 avem

$$f(a+h) = f(a) + h f^{(1)}(a) + (h^2/2) f^{(2)}(c)$$

unde $c \in (a, a+h)$. Deci

$$[\, f(a+h) - f(a) \,] / h - f^{(1)}(a) = (h^2/2) f^{(2)}(c)$$

Prin aplicarea modulului în ultima egalitate se obtine inegalitatea (8).

Propozitia 4. Daca $f \in C^{(3)}([\, a, a+h \,])$ atunci

$$| f^{(1)}(a) - [\, f(a+h/2) - f(a-h/2) \,] / h | \le (h^2/24) \sup \{ | f^{(3)}(x) | , a-h/2 \le x \le a+h/2 \} \qquad (9)$$

Demonstratie. De aceasta data vom aplica formula lui Taylor considerând un rest de ordinul 3 în vederea estimarii valorilor functiei f(x) în punctele $a + h/2$ si $a - h/2$. Asadar

$$f(a + h/2) = f a) + (h/2) f^{(1)}(a) + (h^2/8) f^{(2)}(a) + (h^3/48) f^{(3)}(c_1)$$

$$f(a - h/2) = f(a) - (h/2) f^{(1)}(a) + (h^2/8) f^{(2)}(a) - (h^3/48) f^{(3)}(c_2)$$

unde $a - h/2 < c_1, c_2 < a + h/2$.

Impartind ultimele doua relatii prin h si apoi scazându-le se obtine în final

$$[\, f(a + h/2) - f(a - h/2) \,] / h - f^{(1)}(a) = (h^2/48) [\, f^{(3)}(c_1) + f^{(3)}(c_2) \,]$$

egalitate ce ne conduce în final, dupa efectuarea unor operatii cu module, la inegalitatea (9) .

Observatia 3. La estimarea derivatei $f^{(1)}(a)$, daca pasul de discretizare h ia valori foarte mici este recomandabil a se folosi aproximatia (7) în locul aproximatiei (6). In adevar, în aceasta situatie cantitatea h^2 devine "cu mult mai mica" decât h (a se compara maximele erorilor de aproximare facute, erori ce satisfac inegalitatile (8) si (9)) .

Mentionam faptul ca eroarea rezultata în aproximarea derivatei $f^{(1)}(a)$ depinde în mare masura atât de forma functiei f(x) cât si de pozitia punctului a . Erorile de rotunjire datorate operatiilor aritmetice specifice calculatorului se adauga erorilor de aproximare ce tin direct de metoda numerica folosita Neglijarea acestor aspecte poate conduce la aparitia unor rezultate aparent contradictorii din punct de vedere teoretic

Aplicatia 1. Vom exemplifica metodele folosite pentru functia $f(x) = 1 / (x^2 + 1)$ a carei derivata este data de formula $f^{(1)}(x) = (- 2 x) / (x^2 + 1)^2$. Aproximarea derivatei $f^{(1)}(x)$ se va realiza în punctul $a = 0.2$ cu ajutorul programului urmator :

```
REM  Aproximarea derivatei unei functii  f(x)  in punctul  a
DECLARE FUNCTION f1! (x!) :          DECLARE FUNCTION f! (x!)
h = .5 :          rh = .1 :          a = .2
FOR j = 1 TO 6
vd = f1(a) :          v1 = (f(a + h) - f(a)) / h :          v2 = (f(a + h / 2) - f(a - h / 2)) / h
er1 = vd - v1 :          er2 = vd - v2                    ' erori de aproximare
c$ = "h = #.######    vd = ##.######    v1 = ##.######    v2 = ##.######    er1 = #.######    er2
= ##.######"
PRINT USING c$; h; vd; v1; v2; er1; er2 :          h = h * rh :          NEXT j
FUNCTION f (x) :          f = 1 / (x * x + 1) :          END FUNCTION
FUNCTION f1 (x) :          ' f1(x)  este derivata functiei  f(x)
f1 = (-2 * x) / (x * x + 1) ^ 2 :          END FUNCTION
```

Exemplul 1. In Tabelul 1 sunt listate rezultatele rularii programului din Aplicatia 1 pentru $f(x) = 1 / (1 + x^2)$. Derivata $f^{(1)}(a)$ este aproximata de cantitatea $v_1 = (f(a+h) - f(a)) / h$ (formula (6)) sau de valoarea $v_2 = (f(a+h/2) - f(a-h/2)) / h$ (formula (7)) rezultând erorile ε_1 , respectiv ε_2 .

Analizarea datelor Tabelului 1 confirma aspectele discutate (Observatia 3). Indiferent de metoda folosita, modulul erorii de aproximare descreste odata cu micsorarea pasului de discretizare h . Trebuie avut însa în vedere si cumularea unor erori de rotunjire ce pot afecta aceasta caracteristica de monotonie a erorilor, fenomen ce se manifeata în cazul unor valori foarte mici ale lui h . In acest context vor fi interpretate rezultatele din Tabelul 1 pentru $h = 0.000050$ si $h = 0.000005$.

Tabelul 1. Erorile ε_1 , ε_2 de estimare a derivatei $f^{(1)}(a)$ pentru $f(x) = 1 / (1 + x^2)$ si $a = 0.2$ considerând aproximatiile $v_1 = (f(a+h) - f(a))/ h$ si $v_2 = (f(a+h/2) - f(a-h/2))/ h$

h	$f^{(1)}(0.2)$	v_1	v_2	ε_1	ε_2
0.500000	-0.369823	-0.580795	-0.331811	0.210972	-0.038012
0.050000	-0.369823	-0.407239	-0.369412	0.037417	-0.000411
0.005000	-0.369823	-0.373709	-0.369823	0.003887	0.000000
0.000500	-0.369823	-0.370145	-0.369906	0.000322	0.000084
0.000050	-0.369823	-0.369549	-0.370741	-.000274	0.000918
0.000005	-0.369823	-0.369549	-0.369549	-.000274	-0.000274

Metodele utilizate se pot extinde cu usurinta si pentru derivate de ordin superior. Astfel aplicând succesiv aproximatii de forma (7) obtinem

$$f^{(2)}(a) \approx [f^{(1)}(a+h/2) - f^{(1)}(a-h/2)] / h \approx [(f(a+h) - f(a)) / h - (f(a) - f(a-h)) / h] / h$$

adica

$$f^{(2)}(a) \approx [f(a+h) - 2 f(a) + f(a-h)] / h^2 \qquad (10)$$

Urmatoarea propozitie specifica marimea erorii atunci când se utilizeaza o aproximatie de tipul (10) pentru estimarea derivatei $f^{(2)}(a)$.

Propozitia 5. Daca $f \in C^{(4)}([a-h, a+h])$ atunci

$$|f^{(2)}(a) - [f(a+h) - 2f(a) + f(a-h)]/h^2| \le (h^2/12) \sup\{|f^{(4)}(x)|, a-h \le x \le a+h\} \quad (11)$$

Demonstratie. Vom utiliza dezvoltarea în serie Taylor a functiei $f(x)$, cu restul de ordinul 4,

$$f(a+h) = f(a) + h f^{(1)}(a) + (h^2/2) f^{(2)}(a) + (h^3/6) f^{(3)}(a) + (h^4/24) f^{(4)}(c_1)$$

$$f(a-h) = f(a) - h f^{(1)}(a) + (h^2/2) f^{(2)}(a) - (h^3/6) f^{(3)}(a) + (h^4/24) f^{(4)}(c_2)$$

unde $c_1 \in (a, a+h)$, $c_2 \in (a-h, a)$.

Adunând ultimile doua egalitati si împartind relatia rezultata cu h^2 se obtine

$$f^{(2)}(a) - [f(a+h) - 2f(a) + f(a-h)]/h^2 = -(h^4/24)[f^{(4)}(c_1) + f^{(4)}(c_1)]$$

Prin aplicarea modulului în aceasta ultima egalitate deducem inegalitatea (11).

3. Polinoame de interpolare

Fie o functie f definita pe un domeniu $D \subset R$ si care ia valori reale. Consideram intervalul $[a, b]$, $[a, b] \subset D$, si o diviziune μ a acestuia caracterizata de cele $n+1$ puncte de diviziune, $a = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = b$. Vom presupune ca functia $f(x)$ este de clasa $C^{(n+1)}([a,b])$ adica admite derivate de ordinul $n+1$ ce sunt continue pe întreg intervalul $[a, b]$.

Sa determinam un polinom $L_n(x)$ de gradul n astfel încât functia $f(x)$ si polinomul $L_n(x)$ sa coincida în toate punctele diviziunii μ, adica

$$f(x_j) = y_j = L_n(x_j), \quad j = 0, 1, 2, 3, \ldots, n-1, n \quad (12)$$

Polinomul $L_n(x)$ astfel construir se numeste polinomul de interpolare al functiei $f(x)$.

Propozitia 6. Pentru orice functie f si diviziune μ exista si este unic un polinom de interpolare $L_n(x)$.

Demonstratie. Definim polinoamele $Q_j(x)$ de gradul n, $0 \le j \le n$,

$$Q_j(x) = \frac{(x-x_0)(x-x_1)(x-x_2)\ldots(x-x_{j-1})(x-x_{j+1})\ldots(x-x_{n-1})(x-x_n)}{(x_j-x_0)(x_j-x_1)\ldots(x_j-x_{j-1})(x_j-x_{j+1})\ldots(x_j-x_{n-1})(x_j-x_n)} \quad (13)$$

Din modul în care au fost construite polinoamele $Q_j(x)$ rezulta $Q_j(x_k) = 0$ pentru orice $0 \le j \ne k \le n$, iar $Q_j(x_j) = 1$ daca $0 \le j \le n$.

Tinând seama de aceste relatii deducem ca polinomul de gradul n dat de egalitatea

$$L_n(x) = \sum_{j=0}^{n} f(x_j) Q_j(x) = \sum_{j=0}^{n} y_j Q_j(x) \quad (14)$$

este un polinom de interpolare deoarece $L_j(x_j) = y_j$, $0 \le j \le n$.

Fie $P_1(x)$ un alt polinom de interpolare de gradul n pentru functia $f(x)$ si diviziunea μ, adica $P_1(x_j) = y_j$, $0 \le j \le n$. Atunci polinomul $P_2(x) = L_n(x) - P_1(x)$ de grad cel mult egal cu n are $n+1$ radacini $x_0, x_1, x_2, \ldots, x_{n-1}, x_n$ deoarece

$$P_2(x_j) = L_n(x_j) - P_1(x_j) = y_j - y_j = 0, \qquad 0 \le j \le n$$

Asadar polinomul $P_2(x)$ este polinomul identic nul având mai multe radacini distincte decât gradul sau. Prin urmare avem $L_n(x) = P_1(x)$ pentru orice $x \in R$ adica polinomul de interpolare este unic determinat.

Observatia 4. Coeficientii a_j, $0 \le j \le n$, ai polinomului de interpolare $P_1(x)$ de grad n,

$$P_1(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \dots + a_{n-2} x^2 + a_{n-1} x + a_n$$

pot fi determinati direct prin rezolvarea unui sistem liniar.

In adevar, din conditiile (12), cum $P_1(x_j) = f(x_j)$, $0 \le j \le n$, rezulta urmatoarele relatii

$$\begin{array}{l} a_0 x_0^n + a_1 x_0^{n-1} + a_2 x_0^{n-2} + \dots + a_{n-2} x_0^2 + a_{n-1} x_0 + a_n = f(x_0) \\ a_0 x_1^n + a_1 x_1^{n-1} + a_2 x_1^{n-2} + \dots + a_{n-2} x_1^2 + a_{n-1} x_1 + a_n = f(x_1) \\ a_0 x_2^n + a_1 x_2^{n-1} + a_2 x_2^{n-2} + \quad + a_{n-2} x_2^2 + a_{n-1} x_2 + a_n = f(x_2) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ a_0 x_n^n + a_1 x_n^{n-1} + a_2 x_n^{n-2} + \dots + a_{n-2} x_n^2 + a_{n-1} x_n + a_n = f(x_n) \end{array} \qquad (15)$$

Precizarea polinomului de interpolare $P_1(x)$ se reduce astfel la gasirea valorilor celor $n + 1$ componente $a_0, a_1, a_2, \dots, a_n$ ale solutiei sistemului liniar (15) de $n + 1$ ecuatii. Sistemul (15) va admite întotdeauna o solutie unica deoarece determinantul sau Δ, de tip Vandermonde, este nenul.

$$\Delta = \det \begin{pmatrix} 1 & 1 & 1 & & 1 \\ x_0 & x_1 & x_2 & & x_n \\ x_0^2 & x_1^2 & x_2^2 & & x_n^2 \\ \dots & \dots & \dots & & \dots \\ x_0^n & x_1^n & x_2^n & & x_n^n \end{pmatrix} = \prod_{j>i} (x_j - x_i) \qquad (16)$$

Reamintim ca polinomul de interpolare de grad n este unic determinat. Avantajul formei Lagrange $L_n(x)$ a polinomului de interpolare $P_1(x)$ consta în faptul ca nu mai necesita rezolvarea sistemului liniar (15) de $n + 1$ ecuatii.

Observatia 5. Exista si alte moduri de reprezentarea polinomului de interpolare $L_n(x)$. De exemplu un polinom de interpolare în una dintre variantele Newton are urmatoarea forma

$$N_n(x) = a_0 + a_1 (x - x_0) + a_2 (x - x_0)(x - x_1) + \dots + a_n (x - x_0)(x - x_1)\dots(x - x_{n-1})$$

Coeficientii polinomului Newton $N_n(x)$ de gradul n vor fi determinati utilizând tehnica "puterilor generalizate" si a diferentelor de diferite ordine.

In aplicatii este necesara estimarea erorii $R_n(x)$ rezultata în urma aproximarii functiei $f(x)$ prin polinomul de interpolare $L_n(x)$ de gradul n.

Asadar avem relatia

$$f(x) = L_n(x) + R_n(x) \qquad (17)$$

Vom scrie restul $R_n(x)$ în forma

$$R_n(x) = h(x)\, W_{n+1}(x) \qquad (18)$$

unde $W_n(x)$ este un polinom de gradul $n + 1$ ale carui radacini sunt chiar nodurile diviziunii de interpolare, adica

$$W_{n+1}(x) = (x - x_0)(x - x_1)(x - x_2)\dots(x - x_{n-1})(x - x_n) \qquad (19)$$

Fixam o valoare $t \in [a, b]$ astfel încât $t \ne x_j$, $0 \le j \le n$. Utilizând notatia $\gamma = h(t)$ construim functia

$$g(y) = f(y) - L_n(y) - \gamma\, W_{n+1}(y)$$

Din modul în care a fost precizata valoarea γ rezulta $g(t) = 0$. In plus avem $g(x_j) = 0$ pentru orice $j = 0, 1, 2, \dots, n$. Asadar functia $g(y)$ admite cel putin $n + 2$ zerouri, anume $t, x_0, x_1, x_2, \dots, x_n$. Vom presupune ca functia $g(y)$ este de clasa $C^{(n+1)}([a, b])$.

Aplicând succesiv o consecinta a teoremei lui Rolle ("între doua zerouri ale unei functii exista cel putin un zerou al derivatei sale") rezulta ca exista un punct $\alpha \in (a, b)$ în care derivata de ordinul n+1 a functiei $g(y)$ se anuleaza, adica

$$g^{(n+1)}(\alpha) = 0$$

Cum $L_n(y)$, $W_{n+1}(y)$ sunt polinoame de gradul n, respectiv $n + 1$ rezulta

$$L_n^{(n+1)}(y) = 0 \quad , \quad W_{n+1}^{(n+1)}(y) = (n + 1)!$$

si deci

$$0 = g^{(n+1)}(\alpha) = f^{(n+1)}(\alpha) - L_n^{(n+1)}(\alpha) - \gamma\, W_{n+1}^{(n+1)}(\alpha) = f^{(n+1)}(\alpha) - \gamma\,(n + 1)!$$

Asadar constanta γ este data de expresia

$$\gamma = f^{(n+1)}(\alpha) / (n + 1)!$$

si cum

$$f(t) = L_n(t) + h(t)\, W_{n+1}(t) = L_n(t) + \gamma\, W_{n+1}(t)$$

pentru orice $t \in (a, b)$ cu $t \neq x_j$, $j = 0, 1, 2, \dots, n$, rezulta ca

$$f(t) = L_n(t) + [\, f^{(n+1)}(\alpha) / (n + 1)! \,]\, W_{n+1}(t) \qquad (20)$$

pentru $t \in (a, b)$ si $t \neq x_j$, $0 \leq j \leq n$.

Datorita formei particulare a polinomului $W_{n+1}(t)$, cum $W_{n+1}(x_j) = 0$ pentru $j = 0, 1, 2, \dots, n$, din formula (20) deducem :

Propozitia 7. Pentru orice $x \in [a, b]$ exista $\alpha \in (a, b)$ astfel încât

$$f(x) = L_n(x) + [\, f^{(n+1)}(\alpha) / (n + 1)! \,]\, W_{n+1}(x) \qquad (21)$$

Relatia (21) exprima eroarea pe care o facem atunci când aproximam comportamentul functiei $f(x)$ prin intermediul polinomului de interpolare $L_n(x)$. Din Propozitia 7 rezulta :

Propozitia 8. Eroarea de interpolare $R_n(x)$ într-un punct $x \in [a, b]$ satisface inegalitatea

$$|\, R_n(x) \,| \leq [\, M_{n+1} / (n + 1)! \,]\, |\, W_{n+1}(x) \,| \qquad (22)$$

unde polinomul $W_{n+1}(x)$ este definit de (19) si

$$M_{n+1} = \sup \{\, |\, f^{(n+1)}(\alpha) \,| \;\; ; \;\; \alpha \in (a, b) \,\} \qquad (23)$$

Aplicatia 2. Fie functia $f(x) = x^{1/3}$, $x \in [1, 27]$. Considerând diviziunea $1 < 8 < 27$ vom preciza polinomul de interpolare de gradul 2 corespunzator si vom estima marimea eroarii $R_n(x)$ pentru orice $x \in [1, 27]$.

Concret, derivatele functiei $f(x)$ sunt date de expresiile

$$f^{(1)}(x) = x^{-2/3} / 3 \qquad f^{(2)}(x) = -2\, x^{-5/3} / 9 \qquad f^{(3)}(x) = 10\, x^{-8/3} / 27$$

Asadar, pentru $x \in [1, 27]$ avem

$$|\, f^{(3)}(x) \,| \leq M_3 = 10 / 27$$

si deci din Propozitia 8 obtinem

$$|\, R_2(x) \,| \leq [\, M_3 / 3! \,]\, |\, W_3(x) \,|$$

Polinomul de interpolare Lagrange $L_2(x)$ de gradul 2 definit de (14) are forma

$$L_2(x) = y_0 \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} + y_1 \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} + y_2 \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} =$$

$$= 1\frac{(x-8)(x-27)}{(1-8)(1-27)} + 2\frac{(x-1)(x-27)}{(8-1)(8-27)} + 3\frac{(x-1)(x-8)}{(27-1)(27-8)} = \frac{-6x^2 + 301x + 1434}{1729} \qquad (24)$$

Prin urmare pentru orice $x \in [\ 1\ ,\ 27\ ]$ eroarea de interpolare satisface inegalitatea

$$| f(x) - L_2(x) | \le [\ 5 / 81\ ] | (x - 1)(x - 8)(x - 27) | \qquad (25)$$

Exemplul 2. In conditiile Aplicatiei 2, programul urmator va edita pentru diferite valori $x \in [\ 1\ ,\ 27\ ]$ atât eroarea de aproximare ce apare în procesul de interpolare data de diferenta $E_r = f(x) - L_2(x) = x^{1/3} + (6x^2 - 301x - 1434) / 1729$, cât si un majorant E_e al modulului estimatiei erorii E_r, anume $E_e = [\ 5 / 81\ ] | (x - 1) (x - 8) (x - 27) |$, $| E_r | \le E_e$ (formula (25))

Rezultatele rularii programului sunt sistematizate în Tabelul 2.

```
REM  Eroarea de interpolare cu un polinom de gradul 2 pentru functia  f(x) = x^(1/3)
FOR x = 1 TO 27
er = x ^ (1 / 3) - (-6 * x ^ 2 + 301 * x + 1434) / 1729
ee = (5 / 81) * ABS((x - 1) * (x - 8) * (x - 27))
PRINT "x = "; x; "    eroare reala = "; er; "    eroare estimata = "; ee :          NEXT x
```

Tabelul 2. Erorile de interpolare E_r, E_e precizate în Exemplul 2.

x	E_r	E_e	x	E_r	E_e
1	0.00000	0.00000	2	0.09624	9.25926
3	0.12183	14.81481	4	0.11719	17.03704
5	0.09690	16.29630	6	0.06813	12.96296
7	0.03497	7.40741	8	0.00000	0.00000
9	-0.03501	8.88889	10	-0.06882	18.88889
11	-0.10048	29.62963	12	-0.12931	40.74074
13	-0.15474	51.85185	14	-0.17632	62.59259
15	-0.19371	72.59259	16	-0.20659	81.48148
17	-0.21472	88.88889	18	-0.21789	94.44444
19	-0.21592	97.77778	20	-0.20866	98.51852
21	-0.19596	96.29630	22	-0.17772	90.74074
23	-0.15382	81.48148	24	-0.12418	68.14815
25	-0.08871	50.37037	26	-0.04734	27.77778
27	0.00000	0.00000			

Valoarea polinomul de interpolare $L_n(x)$ coincide cu valoarea functiei $f(x)$ în punctele $x_0 = 0$, $x_1 = 8$, $x_2 = 27$ ale diviziunii intervalului $[\ 0\ ,\ 27\ ]$. In aceste puncte atât eroarea de aproximare reala E_r cât si eroarea estimata E_e sunt nule.

Aplicatia 3. Urmatorul program va calcula efectiv valoarea polinomului de interpolare Lagrange de gradul n într-un punct arbitrar t , $t \in [a\ ,\ b]$, cunoscând valorile $y_j = f(x_j)$ ale functiei $f(x)$ în punctele de diviziune x_j , $0 \le j \le n$.

Avându-se în vederea anumite comparatii ulterioare, punctele x_0 , x_1 , x_2 , ... , x_n au fost alese astfel încât ele sa împarta intervalul [a , b] în subintervale de lungimi egale.

```
REM  Polinom de interpolare Lagrange  Lag(t,n,x,y)  de gradul  n  pentru punctele
REM  de coordonate  ( x(j) , y(j) ) ,  0 <= j <= n
DECLARE FUNCTION Lag! (t!, n!, x!(), y()) :          DECLARE FUNCTION f! (t!)
OPTION BASE 0 :          DIM x(15), y(15) :          DATA - 1.4, 1.0, 12
READ a, b      ' [ a , b ] este domeniul pe care se face interpolarea functiei  f :
READ m     ' m+1 = numarul de puncte  t  in care se listeaza valorile functiei  f
hm = (b - a) / m
FOR im = 0 TO m :          t = a + im * hm
PRINT USING "###.##  ,  ###.#####  ,  "; t; f(t);
FOR n = 2 TO 4 :          hn = (b - a) / n
FOR j = 0 TO n :       x(j) = a + j * hn :       y(j) = f(x(j)) :       NEXT j
va = Lag(t, n, x(), y())      ' va = valoare aproximativa calculata prin interpolare
PRINT USING "###.#####  ,  ##.#####  ,  "; va; f(t) - va; :          NEXT n
PRINT :          NEXT im
FUNCTION f (t) :          f = 1 / ( 2 * t + 3 ) :          END FUNCTION
FUNCTION Lag (t, n, x(), y())      ' Polinom de interpolare Lagrange
s = 0
FOR j = 0 TO n :          q = 1 :          xj = x(j)
FOR k = 0 TO n
IF k <> j THEN q = q * (t - x(k)) / (xj - x(k))
NEXT k
s = s + y(j) * q :          NEXT j
Lag = s :          END FUNCTION
```

Exemplul 3. In Aplicatia 3 consideram a = -1.4 , b = 1.0 , n ∈ { 2 , 3 , 4 } , si functia f(x) = 1 / (2 x + 3) .

Rezultatele rularii programului BASIC din Aplicatia 3 sunt prezentate în Tabelul 3 precizându-se pentru un argument t dat valoarea functiei f(t) cât si aproximatiile $L_2(t)$, $L_3(t)$, $L_4(t)$ ce au fost obtinute prin utilizarea unui polinom de interpolare Lagrange $L_n(t)$ de grad 2 , 3 , respectiv 4 . De asemenea este calculata si eroarea $R_n(t) = f(t) - L_n(t)$ aparuta în procesul de interpolare.

Tabelul 3. Valorile polinomului de interpolare Lagrange $L_n(t)$ si restul $R_n(t)$ corespunzator pentru functia $f(x) = 1 / (2x + 3)$, $-1.4 \le x \le 1.0$, $n \in \{ 2 , 3 , 4 \}$.

t	f(t)	$L_2(t)$	$R_2(t)$	$L_3(t)$	$R_3(t)$	$L_4(t)$	$R_4(t)$
-1.40	5.00000	5.00000	0.00000	5.00000	0.00000	5.00000	0.00000
-1.20	1.66667	3.92308	-2.25641	3.27712	-1.61046	2.75246	-1.08579
-1.00	1.00000	2.96923	-1.96923	2.00392	-1.00392	1.41457	-0.41457
-0.80	0.71429	2.13846	-1.42418	1.11765	-0.40336	0.71429	0.00000
-0.60	0.55556	1.43077	-0.87521	0.55556	0.00000	0.42394	0.13161
-0.40	0.45455	0.84615	-0.39161	0.25490	0.19964	0.36032	0.09422

-0.20	0.38462	0.38462	-0.00000	0.15294	0.23167	0.38462	0.00000
0.00	0.33333	0.04615	0.28718	0.18693	0.14641	0.40243	-0.06910
0.20	0.29412	-0.16923	0.46335	0.29412	0.00000	0.36379	-0.06968
0.40	0.26316	-0.26154	0.52470	0.41176	-0.14861	0.26316	0.00000
0.60	0.23810	-0.23077	0.46886	0.47712	-0.23903	0.13939	0.09871
0.80	0.21739	-0.07692	0.29431	0.42745	-0.21006	0.07577	0.14162
1.00	0.20000	0.20000	0.00000	0.20000	0.00000	0.20000	0.00000

Analizând Tabelul 3 se verifica relatia cunoscuta $R_n(x_j) = f(x_j) - L_n(x_j) = 0$, $n \in \{ 2, 3, 4 \}$. Se constata ca pentru setul de valori t utilizat avem $| R_2(t) | \leq 2.25641$, $| R_3(t) | \leq 1.61046$, $| R_4(t) | \leq 1.08579$, maximul erorii de aproximare micsorându-se odata cu cresterea gradului polinomului de interpolare .

Exemplul 4. Efectuând unele modificari vom rula programul prezentat în Aplicatia 3 pentru functia $f(t) = t^4 - 2t^3 + 3t^2 - 4t + 5$ definita pe intervalul $[0 , 2.4]$, polinomul de interpolare Lagrange având gradul n , $n \in \{ 2, 3, 4 \}$. Cum functia f (t) ce urmeaza a fi interpolata este un polinom de gradul 4 rezulta ca polinomul de aproximare Lagrange de gradul al patrulea va coincide cu aceasta functie, adica $L_4(t) \equiv f(t)$. Deci eroarea facuta în procesul de interpolare pentru $n = 4$ este nula ($R_4(t) = f(t) - L_4(t) = 0$, $\forall t \in [0 , 2.4]$) fapt ce poate fi verificat experimental (a se vedea Tabelul 4).

Tabelul 4. Valorile polinomului de interpolare Lagrange $L_n(t)$ si restul $R_n(t)$ corespunzator pentru functia $f(x) = x^4 - 2x^3 + 3x^2 - 4x + 5$, $n \in \{ 2, 3, 4 \}$.

t	f(t)	$L_2(t)$	$R_2(t)$	$L_3(t)$	$R_3(t)$	$L_4(t)$	$R_4(t)$
0.00	5.00000	5.00000	0.00000	5.00000	0.00000	5.00000	0.00000
0.20	4.30560	3.51360	0.79200	4.67520	-0.36960	4.30560	0.00000
0.40	3.77760	2.49760	1.28000	4.16160	-0.38400	3.77760	0.00000
0.60	3.37760	1.95200	1.42560	3.59360	-0.21600	3.37760	0.00000
0.80	3.10560	1.87680	1.22880	3.10560	0.00000	3.10560	0.00000
1.00	3.00000	2.27200	0.72800	2.83200	0.16800	3.00000	0.00000
1.20	3.13760	3.13760	0.00000	2.90720	0.23040	3.13760	0.00000
1.40	3.63360	4.47360	-0.84000	3.46560	0.16800	3.63360	-0.00000
1.60	4.64160	6.28000	-1.63840	4.64160	0.00000	4.64160	-0.00000
1.80	6.35360	8.55680	-2.20320	6.56960	-0.21600	6.35360	0.00000
2.00	9.00000	11.30400	-2.30400	9.38400	-0.38400	9.00000	0.00000
2.20	12.84960	14.52160	-1.67200	13.21920	-0.36960	12.84960	0.00000
2.40	18.20960	18.20960	0.00000	18.20960	0.00000	18.20960	0.00000

Mentionam ca pentru valorile t listate în Tabelul 4 avem $| R_2(t) | \leq 2.30400$ si

$|R_3(t)| \le 0.38400$, polinomul de aproximare Lagrange $L_3(t)$ de gradul al treilea fiind preferabil polinomului de interpolare $L_2(t)$ de gradul al doilea.

4. Aproximari polinomiale pe subintervale

Fie functia $f : [a, b] \to \mathbf{R}$ derivabila cu derivata continua pe $[a, b]$ si $a = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = b$ o diviziune a intervalului $[a, b]$.

Prin aproximarea functiei $f(x)$ cu polinomul de interpolare Lagrange $L_n(x)$ de grad n (dat de formula (14)) sunt respectate conditiile (12), anume $L_n(x_j) = f(x_j) = y_j$, $0 \le j \le n$. Eroarea de aproximare $R_n(x) = f(x) - L_n(x)$ este nula în toate nodurile x_j ale diviziunii, $R_n(x_j) = f(x_j) - L_n(x_j) = 0$.

Utilizarea polinoamelor de interpolare $L_n(x)$ nu conduce însa la aproximatii care sa se "muleze" cât mai bine pe functia $f(x)$. O dovada în acest sens o constitue faptul ca derivata $R_n^{(1)}(x)$ a erorii de aproximare nu se anuleaza neaparat în punctele x_j, $0 \le j \le n$.

In practica este necesara determinarea unor aproximatii $S(x)$ ale functiei $f(x)$ astfel încât sa fie respectate restrictiile suplimentare

$$S^{(k)}(x_j) = f^{(k)}(x_j), \quad 0 \le j \le n, \ 0 \le k \le q \tag{26}$$

Pentru $q = 0$ relatiile (26) se reduc la conditiile de interpolare de tipul (12).

Asadar eroarea de aproximare $R_n(x)$ împreuna cu toate derivatele sale ce nu depasesc ordinul q se anuleaza în punctele de diviziune x_j, $0 \le j \le n$. Prin impunerea cerintelor (26) eroarea $R_n(x)$ este o functie mult mai "neteda".

Vom studia cazul aproximatiilor (spline) $S(x)$ "cubice" ce sunt polinoame de gradul al treilea definite pe subintervalele $[x_{j-1}, x_j]$, $1 \le j \le n$, adica

$$S(x) = a_0^{(j)} x^3 + a_1^{(j)} x^2 + a_2^{(j)} x + a_3^{(j)} \quad \text{daca } x \in [x_{j-1}, x_j], \ 1 \le j \le n \tag{27}$$

Vom determina forma aproximatiei $S(x)$ astfel încât sa fie satisfacute relatiile

$$S(x_j) = f(x_j) = y_j \qquad S^{(1)}(x_j) = m_j \qquad \forall\ 0 \le j \le n \tag{28}$$

Propozitia 9. Pentru orice $x \in [x_{j-1}, x_j]$, $1 \le j \le n$, functia

$$S(x) = \frac{(x - x_{j+1})^2 (2x - 3x_j + x_{j+1}) y_j + (x - x_j)^2 (3x_{j+1} - x_j - 2x) y_{j+1}}{(x_{j+1} - x_j)^3} +$$

$$+ \frac{(x - x_{j+1})^2 (x - x_j) m_j + (x - x_j)^2 (x - x_{j+1}) m_{j+1}}{(x_{j+1} - x_j)^2} \tag{29}$$

verifica conditiile (28).

Demonstratie. Printr-un calcul direct deducem

$$S(x_j) = y_j = f(x_j) \qquad S(x_{j+1}) = y_{j+1} = f(x_{j+1}) \qquad 1 \le j \le n \tag{30}$$

Derivând functia $S(x)$ pe intervalul $[x_{j-1}, x_j]$ obtinem

$$S^{(1)}(j) = \frac{6(x_{j+1} - x)(x - x_j)(y_{j+1} - y_j)}{(x_{j+1} - x_j)^3} +$$

$$+\frac{(x_{j+1}-x)(x_{j+1}+2x_j-3x)m_j+(x-x_j)(3x-x_j-2x_{j+1})m_{j+1}}{(x_{j+1}-x_j)^2} \qquad (31)$$

si deci sunt adevarate relatiile

$$S^{(1)}(x_j)=m_j \qquad S^{(1)}(x_{j+1})=m_{j+1} \qquad 1\le j\le n \qquad (32)$$

Prin urmare aproximatia S(x) satisface conditiile (26) pentru q = 1 (egalitatile (30) si (32)).

Propozitia 10. Aproximatia S(x) definita de expresia (29) este singura functie polinomiala de gradul al treilea ce respecta restrictiile (26) pentru q = 1 .

Demonstratie. Cum S(x) are forma (27) rezulta $S^{(1)}(x)=3a_0^{(j)}x^2+2a_1^{(j)}x+a_2^{(j)}$ pentru orice $x\in[x_{j-1},x_j]$. Retranscriind conditiile (28) rezulta urmatorul sistem liniar de 4 ecuatii în necunoscutele $a_0^{(j)}, a_1^{(j)}, a_2^{(j)}, a_3^{(j)}$,

$$\begin{aligned}
&a_0^{(j)}x_j^3+a_1^{(j)}x_j^2+a_2^{(j)}x_j+a_3^{(j)}=y_j\\
&a_0^{(j)}x_{j+1}^3+a_1^{(j)}x_{j+1}^2+a_2^{(j)}x_{j+1}+a_3^{(j)}=y_{j+1}\\
&3a_0^{(j)}x_j^2+2a_1^{(j)}x_j+a_2^{(j)}=m_j\\
&3a_0^{(j)}x_{j+1}^2+2a_1^{(j)}x_{j+1}+a_2^{(j)}=m_{j+1}
\end{aligned} \qquad (33)$$

Sistemul liniar (33) admite solutie unica deoarece determinantul principal Δ este nenul,

$$\Delta=\begin{vmatrix} x_j^3 & x_j^2 & x_j & 1\\ x_{j+1}^3 & x_{j+1}^2 & x_{j+1} & 1\\ 3x_j^2 & 2x_j & 1 & 0\\ 3x_{j+1}^2 & 2x_{j+1} & 1 & 0\end{vmatrix}=\begin{vmatrix} x_j^3-x_{j+1}^3 & x_j^2-x_{j+1}^2 & x_j-x_{j+1}\\ 3x_j^2 & 2x_j & 1\\ 3x_{j+1}^2 & 2x_{j+1} & 1\end{vmatrix}=-(x_{j+1}-x_j)^4\neq 0$$

Observatia 6. Vom considera $m_j=f^{(1)}(x_j)$, $0\le j\le n$, în situatia în care se cunoaste efectiv prima derivata $f^{(1)}(x)$ a functiei f(x) .

Precizarea valorilor m_j, $0\le j\le n$, permite constructia, dupa formula (29), a aproximatiei cubice S(x) . In imposibilitatea obtinerii derivatei $f^{(1)}(x)$ pot fi folosite urmatoarele estimatii pentru cantitatile m_j, $0\le j\le n$,

$$f^{(1)}(a)=f^{(1)}(x_0)\approx\frac{4f(a+h)-f(a+2h)-3f(a)}{2h}=m_0$$

$$f^{(1)}(b)=f^{(1)}(x_n)\approx\frac{3f(b)-4f(b-h)+f(b-2h)}{2h}=m_n$$

$$f^{(1)}(x_j)\approx\frac{f(x_{j+1})-f(x_{j-1})}{2h}=m_j \qquad \forall\ 1\le j\le n-1 \qquad (34)$$

unde h este o cantitate pozitiva foarte mica.

5. Polinomul Taylor

Fie $f \in C^{(n+1)}([a, b])$ si $c \in (a, b)$. Vom determina un polinom $T_n(x)$ de grad n astfel încât sa fie verificate restrictiile

$$T_n^{(k)}(c) = f^{(k)}(c) = f_k \qquad \forall\ 0 \le k \le n \tag{35}$$

Polinomul $T_n(x)$ construit în acest mod se numeste polinomul Taylor de gradul n atasat functiei f(x) în punctul c.

Propozitia 11. Pentru orice functie $f \in C^{(n+1)}([a, b])$ exista un polinom Taylor $T_n(x)$ unic determinat.

Demonstratie. Fie $T_n(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_{n-2} x^2 + a_{n-1} x + a_n$ un polinom oarecare de gradul n ($a_0 \neq 0$). Vom determina coeficientii a_j, $0 \le j \le n$, astfel încât sa fie satisfacute conditiile (35). Totodata vom arata ca polinomul $T_n(x)$ astfel obtinut este unic determinat.

Impunând ca derivata de ordin k a polinomului $T_n(x)$ sa fie în punctul c egala cu $f_k = f^{(k)}(c)$, $0 \le k \le n$, rezulta urmatorul sistem liniar, superior triunghiular, de n+1 ecuatii, în necunoscutele $a_0, a_1, a_2, \ldots, a_{n-1}, a_n$,

$$\begin{aligned}
&A_n^0 a_0 c^n + A_{n-1}^0 a_1 c^{n-1} + A_{n-2}^0 a_2 c^{n-2} + A_{n-3}^0 a_3 c^{n-3} + \ldots + A_2^0 a_{n-2} c^2 + A_1^0 a_{n-1} c^1 + A_0^0 a_n c^0 = f_0 \\
&A_n^1 a_0 c^{n-1} + A_{n-1}^1 a_1 c^{n-2} + A_{n-2}^1 a_2 c^{n-3} + \ldots + A_3^1 a_{n-3} c^2 + A_2^1 a_{n-2} c^1 + A_1^1 a_{n-1} c^0 = f_1 \\
&A_n^2 a_0 c^{n-2} + A_{n-1}^2 a_1 c^{n-3} + \ldots + A_4^2 a_{n-4} c^2 + A_3^2 a_{n-3} c^1 + A_2^2 a_{n-2} c^0 = f_2 \\
&\qquad\qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \qquad (36) \\
&A_n^{n-2} a_0 c^2 + A_{n-1}^{n-2} a_1 c^1 + A_{n-2}^{n-2} a_2 c^0 = f_{n-2} \\
&A_n^{n-1} a_0 c^1 + A_{n-1}^{n-1} a_1 c^0 = f_{n-1} \\
&A_n^n a_0 c^0 = f_n
\end{aligned}$$

unde $A_m^k = m\,(m-1)\,(m-2)\ldots(m-k+2)\,(m-k+1)$.

Cum determinantul det (Δ) al sistemului liniar (36) este nenul,

$$\det(\Delta) = A_n^n A_{n-1}^{n-1} A_{n-2}^{n-2} \ldots A_2^2 A_1^1 A_0^0 \neq 0$$

rezulta ca acest sistem admite o solutie unica $a_0, a_1, a_2, \ldots, a_{n-1}, a_n$. In aceasta situatie polinomul $T_n(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_{n-2} x^2 + a_{n-1} x + a_n$ este unicul polinom de grad n ce satisface conditiile (35).

Observatia 7. Coeficientii a_k, $0 \le k \le n$, ai polinomului Taylor $T_n(x)$ pot fi obtinuti prin rezolvarea sistemului triunghiular (36). Ecuatiile sistemului vor fi utilizate în sens invers, începând cu ultima ecuatie (din care se determina coeficientul a_0) si terminând cu prima ecuatie (din care se deduce valoarea a_n).

Propozitia 12. Polinomul Taylor $T_n(x)$ este de forma

$$T_n(x) = f(c) + \frac{f^{(1)}(c)}{1!}(x-c) + \frac{f^{(2)}(c)}{2!}(x-c)^2 + \ldots + \frac{f^{(n-1)}(c)}{(n-1)!}(x-c)^{n-1} + \frac{f^{(n)}(c)}{n!}(x-c)^n \tag{37}$$

Demonstratie. Cum polinomul Taylor este unic determinat va fi suficient sa aratam ca polinomul $T_n(x)$ definit prin formula (37) satisface conditiile (35). In adevar, prin calcul direct, derivând succesiv polinomul (37), deducem $T_n^{(k)}(c) = f^{(k)}(c)$, pentru orice $0 \le k \le n$.

Este foarte important de a evalua restul $E_n(x) = f(x) - T_n(x)$ rezultat în urma aproximarii functiei f(x) prin polinomul Taylor $T_n(x)$.

Propozitia 13. Exista $\alpha \in (a, b)$ astfel încât

$$E_n(x) = f(x) - T_n(x) = \frac{(x-c)^{n+1}}{(n+1)!} f^{(n+1)}(\alpha) \qquad (38)$$

Demonstratie. Pentru o valoare x fixata fie $\lambda = [\, f(x) - T_n(x) \,] / (\, x - c \,)^{n+1}$

Pentru functia h(z) definita prin formula

$$h(z) = f(z) + \frac{f^{(1)}(z)}{1!}(x-z) + \frac{f^{(2)}(z)}{2!}(x-z)^2 + \ldots + \frac{f^{(n)}(z)}{n!}(x-z)^n + \lambda (x-z)^{n+1}$$

avem $h(x) = f(x)$ si $h(c) = f(x)$. Presupunând, de exemplu, $c < x$, atunci conform teoremei lui Rolle rezulta $\alpha \in (c, x)$ astfel încât $h^{(1)}(\alpha) = 0$.

Prin calcul direct deducem $h^{(1)}(z) = [\, (x-z)^n / n! \,] f^{(n+1)}(z) - \lambda (n+1)(x-z)^n$. Cum $h^{(1)}(\alpha) = 0$ si $\lambda = R_n(x) / (x-c)^{n+1}$ obtinem $f^{(n+1)}(\alpha) / n! - (n+1) R_n(x) / (x-c)^{n+1} = 0$ adica restul aproximarii Taylor este dat de expresia (38) (forma Lagrange a restului Taylor).

Observatia 8. Datorita conditiilor (35) polinomul Taylor $T_n(x)$ aproximeaza foarte bine functia f(x) într-o vecinatate a punctului c. In schimb polinomul de interpolare Lagrange $L_n(x)$ ia chiar valorile functiei f(x) în n+1 puncte fixate x_j, $0 \le j \le n$. In jurul unui punct c ales arbitrar polinomul de interpolare $L_n(x)$ este în general mai putin precis decât polinomul de aproximare Taylor asociat punctului c.

Aplicatia 4. Fie $f(x) = e^x$ cu $x \in [-1, 1]$ Pastrând notatiile anterioare, pentru $x_0 = -1$, $x_1 = c = 0$, $x_2 = 1$, $n = 2$ deducem expresia polinomului de interpolare $L_2(x)$ (formula (14)) si a polinomului de aproximare Taylor $T_2(x)$ (formula (37)), anume

$$L_2(x) = (x^2 - x)/(2e) + (1 - x^2) + (x^2 + x)\, e/2 \qquad T_2(x) = 1 + x + x^2/2 \qquad (39)$$

Urmatorul program determina valorile polinoamelor de aproximare $L_2(x)$, $T_2(x)$, rezultatele obtinute fiind centralizate în Tabelul 5.

```
REM  Compararea aproximarilor Lagrange si Taylor
DECLARE FUNCTION T2! (x!) :          DECLARE FUNCTION L2! (x!)
FOR x = -1.0 TO 1.0 STEP 0.1 :       PRINT EXP(x) , L2(x) , T2(x) :      NEXT x
FUNCTION L2 (x) :          e = 2.718282
L2 = (x * x - x) / (2! * e) + (1 - x * x) + (x * x + x) * e / 2! :          END FUNCTION
FUNCTION T2 (x) :          T2 = 1 + x + x * x / 2! :                       END FUNCTION
```

Datele listate în Tabelul 5 confirma Observatia 8.

In adevar polinomul de interpolare $L_2(x)$ definit de punctele $x_0 = -1$, $x_1 = 0$, $x_2 = 1$ este "exact" în aceste puncte, anume $L_2(x_j) = f(x_j)$, $j \in \{0, 1, 2\}$ (conditia (12)). Aproximatiile date de polinomul Taylor $T_2(x)$ în punctele x_0 si x_2 au erori mari (Tabelul 5). In jurul punctelor x_0 si x_2 se va prefera aproximarea functiei f(x) prin polinomul de interpolare Lagrange $L_2(x)$ si nu utilizarea

polinomului Taylor $T_2(x)$ (a se vedea Tabelul 5). In schimb polinomul Taylor $T_2(x)$ definit de punctul $c = 0$ aproximeaza mai precis valorile functiei $f(x)$ atunci când x este într-o vecinatate a lui c. Din Tabelul 5 reiese clar ca pentru $-0.6 \le x \le 0.6$ se va prefera aproximatia Taylor $T_2(x)$ în locul aproximarii Lagrange $L_2(x)$.

Tabelul 5. Compararea valorilor polinoamelor de aproximare $L_2(x)$ si $T_2(x)$ (expresiile (39)) pentru functia $f(x) = e^x$, $x \in [-1, 1]$.

x	**-1.0**	**-0.9**	**-0.8**	**-0.7**	**-0.6**	**-0.5**	**-0.4**
f(x)	0.368	0.407	0.449	0.497	0.549	0.607	0.670
$L_2(x)$	0.368	0.382	0.407	0.443	0.490	0.548	0.617
$T_2(x)$	0.500	0.505	0.520	0.545	0.580	0.625	0.680
x	**-0.3**	**-0.2**	**-0.1**	**0.0**	**0.1**	**0.2**	**0.3**
f(x)	0.741	0.819	0.905	1.000	1.105	1.221	1.350
$L_2(x)$	0.696	0.787	0.888	1.000	1.123	1.257	1.401
$T_2(x)$	0.745	0.820	0.905	1.000	1.105	1.220	1.345
x	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1.0**
f(x)	1.492	1.649	1.822	2.014	2.226	2.460	2.718
$L_2(x)$	1.557	1.723	1.901	2.089	2.288	2.498	2.718
$T_2(x)$	1.480	1.625	1.780	1.945	2.120	2.305	2.500

6. Aproximari în medie patratica

Definitie. Un **spatiu euclidian** $\mathbf{E}$ peste corpul $\mathbf{R}$ este un $\mathbf{R}$-spatiu vectorial pe care s-a definit un **produs scalar** $< . , . > : \mathbf{E} \times \mathbf{E} \to \mathbf{R}$. Functia $< . , . >$ are proprietatile :

1). $< v , w > = < w , v >$, $\forall\ v, w \in \mathbf{E}$ (comutativitate)

2). $< v , v > \ge 0$, $\forall\ v \in \mathbf{E}$ (pozitiv definita)

$\forall\ v \in \mathbf{E}$, $< v , v > = 0_R \Leftrightarrow v = 0_E$

3). $\forall\ \alpha \in \mathbf{R}$, $\forall\ v, w \in \mathbf{E} \Rightarrow < \alpha v , w > = \alpha < v , w >$ (biliniara)

$\forall\ u, v, w \in \mathbf{E} \Rightarrow < u + v , w > = < u , w > + < v , w >$ (40)

Elementele spatiului euclidian $\mathbf{E}$ se numesc vectori iar $\alpha \in \mathbf{R}$ este scalar.

Definitie. Un sistem de vectori $w_0 , w_1 , w_2 , \dots , w_m \in \mathbf{E}$ sunt **liniar independenti** daca pentru orice $\alpha_0 , \alpha_1 , \alpha_2 , \dots , \alpha_m \in \mathbf{R}$ astfel încât $\alpha_0 w_0 + \alpha_1 w_1 + \alpha_2 w_2 + \dots + \alpha_m w_m = 0_E$ rezulta obligatoriu $\alpha_0 = \alpha_1 = \alpha_2 = \dots = \alpha_m = 0_R$.

Pe un spatiu euclidian $\mathbf{E}$ definim functia "**norma**" $\| . \| : \mathbf{E} \to \mathbf{R}$ prin $\| v \| = (< v , v >)^{1/2}$, $\forall\ v \in \mathbf{E}$, ce are urmatoarele proprietati derivate din caracteristicile (40) ale produsului scalar :

1). $\| v \| \ge 0$, $\forall\ v \in \mathbf{E}$

$\forall\ v \in \mathbf{E},\ \| v \| = 0_R \Leftrightarrow v = 0_E$

2). $\forall\ \alpha \in \mathbf{R},\ \forall\ v \in \mathbf{E} \Rightarrow \| \alpha\, v \| = | \alpha |\ \| v \|$

3). $\forall\ u, v \in \mathbf{E} \Rightarrow \| u + v \| \le \| u \| + \| v \|$ (inegalitatea triunghiului) (41)

Propozitia 14. Fiind dati în spatiul euclidian $\mathbf{E}$ vectorii liniar independenti w_0, w_1, w_2, w_m atunci pentru orice $v \in \mathbf{E}$ exista $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_m$ unic determinati în $\mathbf{R}$ astfel încât

$$h(v; \alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_m) = \inf \{ h(v; \beta_0, \beta_1, \beta_2, \ldots, \beta_m) \mid \beta_j \in \mathbf{R},\ 0 \le j \le m \} \quad (42)$$

functia h fiind precizata de expresia

$$h(v; \alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_m) = \| v - \beta_0 w_0 - \beta_1 w_1 - \beta_2 w_2 - \ldots - \beta_m w_m \| \quad (43)$$

Demonstratie. Functia h definita de (43) poate fi adusa la forma

$$h(v;\beta_0,\beta_1,\beta_2,\ldots,\beta_m) = \| v - \sum_{j=0}^{m} \beta_j w_j \| = < v - \sum_{j=0}^{m} \beta_j w_j\ ,\ v - \sum_{k=0}^{m} \beta_k w_k > =$$

$$= < v\ , v > + \sum_{j=0}^{m} \sum_{k=0}^{m} \beta_j \beta_k < w_j\ , w_k > - 2 \sum_{j=0}^{m} \beta_j < w_j\ , v > \quad (44)$$

Forma patratica (44) este nenegativa si infinit derivabila în argumentele α_j, $0 \le j \le m$. Pentru un vector $v \in \mathbf{E}$ fixat, minimul functiei h este efectiv atins pentru valorile reale $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_m$ ce anuleaza derivatele partiale ale lui f, adica

$$\frac{\partial h(v;\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_m)}{\partial \alpha_j} = 0 \qquad \forall\ 0 \le j \le m \quad (45)$$

Tinând seama de forma (44) a functiei h, cele $m + 1$ relatii (45) ne conduc la un sistem liniar de $m + 1$ ecuatii în necunoscutele $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_m$, anume

$$\begin{aligned}
&\alpha_0 < w_0, w_0 > + \alpha_1 < w_0, w_1 > + \alpha_2 < w_0, w_2 > + \quad + \alpha_m < w_0, w_m > = < w_0, v > \\
&\alpha_0 < w_1, w_0 > + \alpha_1 < w_1, w_1 > + \alpha_2 < w_1, w_2 > + \ldots + \alpha_m < w_1, w_m > = < w_1, v > \\
&\alpha_0 < w_2, w_0 > + \alpha_1 < w_2, w_1 > + \alpha_2 < w_2, w_2 > + \quad + \alpha_m < w_2, w_m > = < w_2, v > \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\alpha_0 < w_m, w_0 > + \alpha_1 < w_m, w_1 > + \alpha_2 < w_m, w_2 > + \quad + \alpha_m < w_m, w_m > = < w_m, v >
\end{aligned} \quad (46)$$

Propozitia 14 este astfel demonstrata deoarece sistemul liniar (46) admite solutie unica, determinantul sau principal Δ fiind nenul.

In adevar, daca vom presupune prin absurd ca determinantul Δ este nul atunci exista $c_0, c_1, c_2, \ldots, c_m$ numere reale, nu toate nule, astfel încât pentru orice $0 \le k \le m$ sa avem

$$c_0 < w_k, w_0 > + c_1 < w_k, w_1 > + c_2 < w_k, w_2 > + \ldots + c_m < w_k, w_m > = 0 \quad (47)$$

Utilizând cele $m + 1$ relatii (47) obtinem

$$\| \sum_{j=0}^{m} c_j w_j \|^2 = < \sum_{k=0}^{m} c_k w_k\ , \sum_{j=0}^{m} c_j w_j > = \sum_{k=0}^{m} c_k \left(\sum_{j=0}^{m} c_j < w_k\ , w_j > \right) = \sum_{k=0}^{m} c_k \cdot 0 = 0 \quad (48)$$

Tinând seama de proprietatea 1.b a normei, din (48) rezulta $\sum_{j=0}^{m} c_j w_j = 0$ fara ca toti coeficientii c_j sa fie nuli. Prin urmare vectorii $w_0, w_1, w_2, \ldots, w_m$ sunt liniar dependenti fapt ce contrazice ipoteza propozitiei.

Observatia 9. Vectorul $z_m = \alpha_0 w_0 + \alpha_1 w_1 + \alpha_2 w_2 + \ldots + \alpha_m w_m$, unde coeficientii α_j , $0 \le j \le m$, sunt solutiile sistemului (46) constitue "cea mai buna aproximatie în medie patratica" a vectorului v exprimata în raport cu vectorii liniar independenti $w_0 , w_1 , w_2 , \ldots , w_m$.

Exemplul 5. Pentru $w_0 \in \mathbf{E}$ fixat vom determina valoarea parametrului $\alpha_0 \in \mathbf{R}$ astfel încât vectorul $z_0 = \alpha_0 w_0$ sa fie cea mai buna aproximatie în medie patratica a vectorului $v \in \mathbf{E}$.

Cum $m = 0$, sistemul (46) se reduce la ecuatia $\alpha_0 < w_0 , w_0 > = < w_0 , v >$ si deci

$$z_0 = (< w_0 , v > / < w_0 , w_0 >) w_0 \qquad (49)$$

Exemplul 6. Considerând vectorii w_0 si w_1 liniar independenti sistemul (46) devine

$$\begin{aligned} &\alpha_0 < w_0 , w_0 > + \alpha_1 < w_0 , w_1 > = < w_0 , v > \\ &\alpha_0 < w_1 , w_0 > + \alpha_1 < w_1 , w_1 > = < w_1 , v > \end{aligned} \qquad (50)$$

a carui solutie este

$$\alpha_0 = \frac{< w_0 , v > < w_1 , w_1 > - < w_1 , v > < w_0 , w_1 >}{< w_0 , w_0 > < w_1 , w_1 > - \left(< w_0 , w_1 > \right)^2}$$

$$\alpha_1 = \frac{< w_1 , v > < w_0 , w_0 > - < w_0 , v > < w_0 , w_1 >}{< w_0 , w_0 > < w_1 , w_1 > - \left(< w_0 , w_1 > \right)^2} \qquad (51)$$

Vectorul $z_1 = \alpha_0 w_0 + \alpha_1 w_1$, unde coeficientii α_0 si α_1 sunt dati de formulele (51), constitue cea mai buna aproximatie în medie patratica pentru elementul $v \in \mathbf{E}$. Aproximarea z_1 se bazeaza pe vectorii liniar independenti w_0 si w_1 .

Aplicatia 5. Fie $\mathbf{E}_1$ spatiul euclidian al functiilor continue pe intervalul [a , b] ,

$\mathbf{E}_1 = \{ f : [a , b] \to \mathbf{R} \mid f \text{ continua} \}$

cu produsul scalar $< . , . >_1 : \mathbf{E}_1 \times \mathbf{E}_1 \to \mathbf{R}$ definit de formula

$$< f , g > = \frac{1}{b-a} \int_a^b f(t)\, g(t)\, dt \qquad \forall\ f, g \in \mathbf{E}_1 \qquad (52)$$

Norma $\| . \|_1 : \mathbf{E}_1 \to \mathbf{R}$ introdusa de produsul scalar $< . , . >_1$ este data de expresia

$$\| f \|_1 = \sqrt{\frac{1}{b-a} \int_a^b f^2(t)\, dt} \qquad \forall\ f \in \mathbf{E}_1 \qquad (53)$$

Consideram elementele $w_0 , w_1 \in \mathbf{E}_1$ definite prin $w_0 (x) = 1$, $w_1 (x) = x$, $\forall\ a \le x \le b$. Se constata ca functiile w_0 si w_1 astfel definite sunt liniar independente.

Pentru o functie arbitrara $f \in \mathbf{E}_1$ se obtin aproximatiile $z_0 (x) = \alpha_0 w_0 (x)$ (formula (49)) si $z_1 (x) = \alpha_0 w_0 (x) + \alpha_1 w_1 (x)$ unde coeficientii α_0 , α_1 au forma (51). Prin calcul direct deducem

$$z_0 (x) = \frac{1}{b-a} \int_a^b f(t)\, dt \qquad z_1 (x) = \alpha_0 + \alpha_1 x \qquad \forall\ x \in [a , b] \qquad (54)$$

unde

$$\alpha_0 = \frac{1}{(b-a)^3} \left(4(a^2 + ab + b^2) \int_a^b f(t)\, dt \ - \ 6(a+b) \int_a^b t\, f(t)\, dt \right)$$

$$\alpha_1 = \frac{1}{(b-a)^3}\left(12 \int_a^b t\, f(t)\, dt \; - \; 6(a+b) \int_a^b f(t)\, dt \right) \qquad (55)$$

Exemplul 7. Fie $f(x) = e^x$, $w_0(x) = 1$, $w_1(x) = x$, $a = 0 \le x \le 1 = b$. Cum

$\int_0^1 e^t\, dt = e - 1$, $\int_0^1 t e^t\, dt = 1$, din (54) si (55) deducem

$$z_0(x) = e - 1 \qquad z_1(x) = (4e - 10) + (18 - 6e)\, x \qquad \forall\ 0 \le x \le 1 \qquad (56)$$

functiile $z_0(x)$ si $z_1(x)$ fiind cele mai bune aproximatii în medie patratica pentru functia e^x

Aplicatia 6. Pentru $a \le x_0 < x_1 < x_2 < \ldots < x_{m-1} < x_m \le b$, consideram spatiul euclidian $\mathbf{E}_2$, $\mathbf{E}_2 = \{ f : \{x_0, x_1, x_2, \ldots, x_m\} \to \mathbf{R} \}$, cu produsul scalar $< . , . >_2 : \mathbf{E}_2 \times \mathbf{E}_2 \to \mathbf{R}$,

$$< f , g > = \frac{1}{m-1} \sum_{j=0}^{m} f(x_j)\, g(x_j) \qquad \forall\ f, g \in \mathbf{E}_2 \qquad (57)$$

Produsul scalar $< . , . >_2$ induce pe spatiul $\mathbf{E}_2$ norma $\| . \|_2$ având forma

$$\| f \|_2 = \sqrt{\frac{1}{m+1} \sum_{j=0}^{m} f^2(x_j)} \qquad \forall\ f \in \mathbf{E}_2 \qquad (58)$$

Pentru orice $0 \le k \le m$ vom folosi notatiile $w_{0k} = w_0(x_k)$, $w_{1k} = w_1(x_k)$, $y_k = f(x_k)$. Conform relatiilor (49) si (51) rezulta aproximatiile $z_0(x)$ si $z_1(x)$ ale functiei $f(x)$, unde

$$z_{0k} = z_0(x_k) = w_{0k} \left(\sum_{j=0}^{m} w_{0j}\, y_j \right) \Big/ \left(\sum_{j=0}^{m} w_{0j}^2 \right)$$

$$z_{1k} = z_1(x_k) = \alpha_0\, w_{0k} + \alpha_1\, w_{1k} \qquad \forall\ 0 \le k \le m \qquad (59)$$

Aplicând formulele (51) sunt determinati coeficientii α_0, α_1 ce caracterizeaza expresiile (59), anume

$$\alpha_0 = \frac{\left(\sum_{j=0}^{m} w_{0j}\, y_j \right)\left(\sum_{j=0}^{m} w_{1j}^2 \right) - \left(\sum_{j=0}^{m} w_{1j}\, y_j \right)\left(\sum_{j=0}^{m} w_{0j}\, w_{1j} \right)}{\left(\sum_{j=0}^{m} w_{0j}^2 \right)\left(\sum_{j=0}^{m} w_{1j}^2 \right) - \left(\sum_{j=0}^{m} w_{0j}\, w_{1j} \right)^2}$$

$$\alpha_1 = \frac{\left(\sum_{j=0}^{m} w_{1j}\, y_j \right)\left(\sum_{j=0}^{m} w_{0j}^2 \right) - \left(\sum_{j=0}^{m} w_{0j}\, y_j \right)\left(\sum_{j=0}^{m} w_{0j}\, w_{1j} \right)}{\left(\sum_{j=0}^{m} w_{0j}^2 \right)\left(\sum_{j=0}^{m} w_{1j}^2 \right) - \left(\sum_{j=0}^{m} w_{0j}\, w_{1j} \right)^2} \qquad (60)$$

Exemplul 8 (Ecuatia de regresie liniara). Daca în Aplicatia 6 vom considera cazurile particulare $w_0(x) = 1$ si $w_1(x) = x$ atunci aproximatiile $z_0(x)$, $z_1(x)$ (relatiile (59)) pentru functia f(x) sunt de forma

$$z_{0k} = z_0(x_k) = \left(\sum_{j=0}^{m} w_{0j}\, y_j \right) / n$$

$$z_{1k} = z_1(x_k) = \alpha_0 + \alpha_1 x_k \qquad \forall\ 0 \le k \le m \tag{61}$$

unde parametrii α_0 si α_1 sunt obtinuti dupa formulele (60), adica

$$\alpha_1 = \frac{(m+1)\left(\sum_{j=0}^{m} x_j y_j \right) - \left(\sum_{j=0}^{m} x_j \right)\left(\sum_{j=0}^{m} y_j \right)}{(m+1)\left(\sum_{j=0}^{m} x_j^2 \right) - \left(\sum_{j=0}^{m} x_j \right)^2}$$

$$\alpha_0 = \frac{\left(\sum_{j=0}^{m} y_j \right)\left(\sum_{j=0}^{m} x_j^2 \right) - \left(\sum_{j=0}^{m} x_j y_j \right)\left(\sum_{j=0}^{m} x_j \right)}{(m+1)\left(\sum_{j=0}^{m} x_j^2 \right) - \left(\sum_{j=0}^{m} x_j \right)^2} = \frac{\sum_{j=0}^{m} y_j}{m+1} - \frac{\sum_{j=0}^{m} x_j}{m+1}\,\alpha_1 \tag{62}$$

Relatia $z_{1k} = \alpha_0 + \alpha_1 x_k$, cu coeficientii α_0, α_1 precizati de (62), defineste ecuatia de regresie liniara pentru punctele $P_k(x_k, y_k)$ cu $y_k = f(x_k)$, $0 \le k \le m$.

Dreapta de regresie $y = \alpha_0 + \alpha_1 x$ exprima de fapt tendinta norului de puncte $\{ P_k(x_k, y_k) \mid 0 \le k \le m \}$.

POLINOAME CEBISEV

1. Proprietati

Datorita unor proprietati specifice, polinoamele Cebîsev joaca un rol deosebit la gasirea unor bune aproximatii . In continuare vom trece în revista câteva caracteristici ale acestei clase de polinoame.

Definitia 1. Notam cu $T_n(x)$ functia $T_n : [-1, 1] \to \mathbf{R}$, $n \in \mathbf{N}$, definita prin formula

$$T_n(x) = \cos(n \arccos(x)) \; ; \qquad -1 \le x \le 1 \tag{1}$$

In urma unui calcul elementar deducem $T_0(x) = 1$, $T_1(x) = x$. Functiile $T_n(x)$ sunt cunoscute în literatura de specialitate sub denumirea de polinoame Cebîsev.

In egalitatea trigonometrica $\cos((n+1)t) + \cos((n-1)t) = 2\cos(nt)\cos(t)$ facând $t = \arccos(x)$ rezulta :

Propozitia 1. Functiile $T_n(x)$ pot fi obtinute utilizând relatia de recurenta

$$T_{n+1}(x) = 2x\,T_n(x) - T_{n-1}(x) \qquad \text{unde} \quad T_0(x) = 1 \; , \quad T_1(x) = x \tag{2}$$

Astfel

$T_2(x) = 2x\,T_1(x) - T_0(x) = 2x^2 - 1$ $\qquad$ $T_4(x) = 2x\,T_3(x) - T_2(x) = 8x^4 - 8x^2 + 1$

$T_3(x) = 2x\,T_2(x) - T_1(x) = 4x^3 - 3x$ $\qquad$ $T_5(x) = 2x\,T_4(x) - T_3(x) = 16x^5 - 20x^3 + 5x$

Propozitia 2. Functiile $T_n(x)$ sunt polinoame de gradul n în care coeficientul lui x^n este 2^{n-1} pentru orice $n \ge 1$. In plus sunt satisfacute relatiile

$$T_n(-x) = (-1)^n T_n(x) \qquad |T_n(x)| \le 1 \qquad \forall x, \; -1 \le x \le 1 \tag{3}$$

$$T_n(x_j) = 0 \quad \text{daca} \quad x_j = \cos((2j+1)\pi / (2n)) \; , \; j = 0, 1, 2, \ldots, n-1 \tag{4}$$

Justificarea faptului ca functiile $T_n(x)$ sunt polinoame rezulta imediat aplicând metoda inductiei matematice si tinând seama de formula recurenta (2).

Folosind relatia (2) deducem coeficientul lui x^{n+1} din $T_{n+1}(x)$ ca fiind dublul coeficientului lui x^n din polinomul $T_n(x)$. Dupa n iteratii se obtine în final valoarea 2^n pentru coeficientul monomului x^{n+1} al polinomului $T_{n+1}(x)$.

Cum $\cos(n(\pi - x)) = (-1)^n \cos(nx)$, $|\cos(y)| \le 1$, aplicând Definitia 1 rezulta relatiile (3)

Pentru $x_j = \cos((2j+1)\pi / (2n))$, $0 \le j \le n-1$, avem

$$T_n(x_j) = \cos(n \arccos(x_j)) = \cos((2j+1)\pi / 2) = 0$$

In plus cantitatile $x_j = \cos((2j+1)\pi / (2n))$ sunt toate distincte. Asadar toate cele n zerouri x_j ale polinomului Cebîsev $T_n(x)$ sunt numere reale si distincte în intervalul $[-1, 1]$.

Propozitia 3. Punctele $t_k = \cos(k\pi / n)$, $-1 \le t_k \le 1$, $k = 0, 1, 2, \ldots, n$, verifica egalitatea

$$T_n(t_k) = (-1)^k \tag{5}$$

t_k fiind punct de extrem pentru polinomul Cebîsev $T_n(x)$ de gradul n .

Demonstratie. $T_n(t_k) = \cos(n \arccos(\cos(k\pi / n))) = \cos(k\pi) = (-1)^k$, $\forall k \in \mathbf{Z}$.

Fie $P_n(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_{n-2} x^2 + a_{n-1} x + a_n$ un polinom oarecare de grad n având coeficienti reali. Notam cu $D_n(x)$ polinomul de grad n obtinut prin multiplicarea

polinomului Cebîsev $T_n(x)$ cu factorul $a_0 / 2^{n-1}$, adica $D_n(x) = (a_0 / 2^{n-1}) T_n(x)$. Prin constructia efectuata ambele polinoame $P_n(x)$ si $D_n(x)$ au acelasi coeficient a_0 al monomului x^n.

Propozitia 4. Pentru orice polinom P (x) este adevarata inegalitatea

$$\sup \{ |P_n(x)| ; \ -1 \le x \le 1 \} \ge \sup \{ |D_n(x)| ; \ -1 \le x \le 1 \} \qquad (6)$$

Demonstratie. Deoarece polinoamele $P_n(x)$ si $D_n(x)$ de grad n au acelasi coeficient a_0 pentru monomul de grad maxim rezulta ca polinomul $Q(x) = P_n(x) - D_n(x)$ are gradul cel mult n-1.

Vom demonstra relatia (6) prin procedeul reducerii la absurd. Astfel presupunând adevarata inegalitatea contrara lui (6), adica $\sup \{ |P_n(x)| ; -1 \le x \le 1 \} < \sup \{ |D_n(x)| ; -1 \le x \le 1 \}$, vom ajunge în final la o contradictie.

Deoarece $\sup \{ |P_n(x)| \} < \sup \{ |D_n(x)| \}$ unde $x \in [-1 , 1]$ deducem ca polinomul Q(x) nu este polinomul identic nul (caci în caz contrar $P_n(x) = D_n(x)$, $\forall x \in [-1 , 1]$ si deci nu putem avea $\sup \{ |P_n(x)| \} < \sup \{ |D_n(x)| \}$, $-1 \le x \le 1$).

Din (3) si (5) obtinem $\sup \{ |D_n(x)| ; -1 \le x \le 1 \} = |D_n(t_k)|$, unde $t_k = \cos (k \pi / n)$, $-1 \le t_k \le 1$, $k = 0, 1, 2, \dots , n$. Tinând seama de ipoteza facuta rezulta $|D_n(t_k)| > |P_n(t_k)|$, oricare ar fi $0 \le k \le n$. Aplicând Propozitia 3 deducem $D_n(t_{j-1}) D_n(t_j) < 0$, $1 \le j \le n$. Asadar pentru orice $1 \le j \le n$ este adevarata inegalitatea $Q(t_{j-1}) Q(t_j) = [P_n(t_{j-1}) - D_n(t_{j-1})] [P_n(t_j) - D_n(t_j)] < 0$. Prin urmare functia continua Q(x) are cel putin n zerouri în intervalul [-1 , 1] deoarece s-au pus în evidenta n+1 variatii de semn în punctele t_k, $-1 \le t_k \le 1$, $0 \le k \le n$. Cum Q(x) este un polinom de grad strict mai mic decât n ce are însa mai mult de n-1 zerouri pe intervalul [-1 , 1] rezulta ca polinomul Q(x) este identic nul, fapt ce contrazice deductia anterioara (Q(x) nu este polinomul identic nul).

Asadar s-a ajuns la o contradictie fapt ce demonstreaza ca presupunerea "prin absurd" facuta este falsa, Propozitia 4 fiind astfel adevarata.

Observatia 1. Propozitia 4 prezinta o importanta deosebita. Astfel dintre toate polinoamele P(x) de acelasi grad n ce sunt normate corespunzator pentru a avea acelasi coeficient al monomului x^n de grad maxim, valorile polinomului Cebîsev asociat au cea mai mica fluctuatie în jurul lui zero. Asadar polinoamele Cebîsev pot fi cu succes utilizate în operatiuni de aproximare prezentând o fluctuatie minima în raport cu toate celelalte polinoame de acelasi grad.

Observatia 2. Functia $h(x) = (2x - b - a) / (b - a)$ realizeaza o bijectie între intervalele [a , b] si [-1 , 1]. Acest lucru face posibila extinderea rezultatelor anterioare pentru polinoame definite pe un interval oarecare [a , b]. In aceasta situatie polinoamele Cebîsev $C_n(x)$ definite pe [a , b] vor fi date de relatia $C_n(x) = T_n(h(x)) = T_n((2x - b - a)/ (b - a))$, $\forall x$, $a \le x \le b$.

In cele ce urmeaza vom pune în evidenta si o alta proprietate importanta a polinoamelor de tip Cebîsev. Astfel la studierea formulelor de aproximare într-un spatiu (liniar) euclidian **E** este de dorit ca "vectorii" de referinta sa fie ortogonali. Utilizarea într-un spatiu euclidian a unei baze de referinta ortogonala simplifica calculul coeficientilor ce dau "ponderile" elementelor unei baze la explicitarea unui "vector" arbitrar al spatiului liniar. In plus acuratetea estimarilor este mai mare în cazul în care în acest proces de aproximare baza de referinta este ortogonala. Mentionam faptul ca polinoamele Cebîsev sunt ortogonale într-un sens ce va fi precizat în continuare.

Caracteristic unui spatiu euclidian **E** (Silov, 1983, p. 243) este prezenta unei forme biliniare pozitiv definita (denumita produs scalar, $< . , . > : \mathbf{E} \times \mathbf{E} \to \mathbf{R}$) cu ajutorul careia se poate construi o norma $\| . \|$, $\| . \| : \mathbf{E} \to \mathbf{R}_+$, anume $\| v \| = < v , v >^{1/2}$, $\forall v \in \mathbf{E}$. Existenta unei norme pe un spatiu liniar **E** oarecare prezinta avantajul ca orice doua elemente v, w ale spatiului **E** pot fi

comparate prin stabilirea raportului dintre "valorile" $\| v \|$, $\| w \|$ atasate "vectorilor" respectivi.

Concret, considerând spatiul $\mathbf{E} = \{ f : [-1, 1] \to \mathbf{R}$, f continua $\}$ vom introduce produsul scalar (Silov, 1983, p. 243), $< . , . > : \mathbf{E} \times \mathbf{E} \to \mathbf{R}$, ce este definit prin formula

$$< f , g > = \int_{-1}^{1} f(x) g(x) / \sqrt{1 - x^2}\, dx \qquad \forall\, f, g \in \mathbf{E} \tag{7}$$

Pentru produsul scalar (7) construim norma $\| . \| : \mathbf{E} \to \mathbf{R}$,

$$\| f \|^2 = < f , f > = \int_{-1}^{1} f^2(x) / \sqrt{1 - x^2}\, dx \qquad \forall\, f \in \mathbf{E} \tag{8}$$

Propozitia 5. Polinoamele Cebîsev $T_n(x)$, $n \geq 0$, satisfac urmatoarele relatii :

$$< T_k , T_n > = \int_{-1}^{1} \frac{T_k(x)\, T_n(x)\, dx}{\sqrt{1 - x^2}} = \begin{cases} 0 & ; \quad k \neq n ,\ k, n \in N \\ \pi & ; \quad pentru\ \ k = n = 0 \\ \pi / 2 & ; \quad pentru\ \ k = n \neq 0 \end{cases} \tag{9}$$

Demonstratie. Relatiile (9) vor fi justificate prin calcul direct. La estimarea valorilor unor integrale se va folosi transformarea de variabila $t = \arccos(x)$ caz în care $dt = - dx / (1 - x^2)^{1/2}$

In adevar

$$< T_0 , T_0 > = \int_{-1}^{1} \frac{T_0(x)\, T_0(x)\, dx}{\sqrt{1 - x^2}} = \int_{-1}^{1} \frac{dx}{\sqrt{1 - x^2}} = \arcsin(1) - \arcsin(-1) = \pi$$

Pentru orice $n \in \mathbf{N}^*$ avem

$$< T_n , T_n > = \int_{-1}^{1} \frac{T_n(x)\, T_n(x)\, dx}{\sqrt{1 - x^2}} = \int_{-1}^{1} \frac{\cos^2(n \arccos(x))\, dx}{\sqrt{1 - x^2}} =$$

$$= \int_{-1}^{1} \frac{1 + \cos(2n \arccos(x))\, dx}{2\sqrt{1 - x^2}} = \int_{-1}^{1} \frac{dx}{2\sqrt{1 - x^2}} + \int_{-1}^{1} \frac{\cos(2n \arccos(x))\, dx}{2\sqrt{1 - x^2}} =$$

$$= \frac{\arcsin(1) - \arcsin(-1)}{2} - \int_{\pi}^{0} \frac{\cos(2nt)\, dt}{2} = \frac{\pi}{2} + \frac{\sin(2n\pi) - \sin(0)}{4n} = \frac{\pi}{2}$$

Daca $n, k \in \mathbf{N}$, $k \neq n$, atunci

$$< T_k , T_n > = \int_{-1}^{1} \frac{T_k(x)\, T_n(x)\, dx}{\sqrt{1 - x^2}} = \int_{-1}^{1} \frac{\cos(k \arccos(x)) \cos(n \arccos(x))\, dx}{\sqrt{1 - x^2}} =$$

$$= - \int_{\pi}^{0} \cos(kt) \cos(nt)\, dt = \int_{0}^{\pi} \frac{\cos((n+k)t) + \cos((n-k)t)}{2}\, dt =$$

$$= \frac{\sin((n+k)\pi) - \sin((n+k)0)}{2(n+k)} + \frac{\sin((n-k)\pi) - \sin((n-k)0)}{2(n-k)} = 0$$

Definitia 2. (Silov, 1983, p.246). Functiile $f, g \in \mathbf{E}$ sunt ortogonale daca $< f , g > = 0$

Din Propozitia 5 (formulele (9)) rezulta :

Corolarul 1. Polinoamele Cebîsev $T_n(x)$, $n \geq 0$, sunt ortogonale doua câte doua în sensul Definitiei 2, considerând produsul scalar (7) .

Observatia 3. Pe spatiul liniar $\mathbf{E} = \{ f : [-1 , 1] \rightarrow \mathbf{R} , f \text{ continua} \}$ produsul scalar nu este unic. Astfel putem defini functia $< . , . >_0 : \mathbf{E} \times \mathbf{E} \rightarrow \mathbf{R}$ prin formula

$$< f , g >_0 = \int_{-1}^{1} f(x)\, g(x)\, dx \qquad \forall f, g \in \mathbf{E} \tag{10}$$

Propozitia 6. Pentru orice $k, n \in \mathbf{N}$, $k \neq n$, avem

$$< T_k , T_n >_0 = \int_{-1}^{1} T_k(x)\, T_n(x)\, dx = \begin{cases} 0 \ ; & \text{pentru } k+n \text{ impar} \\ \dfrac{1}{1-(n+m)^2} + \dfrac{1}{1-(n-m)^2} \ ; & \text{pentru } k+n \text{ par} \end{cases} \tag{11}$$

Demonstratie. Facând substitutia $t = \arccos(x)$, adica $x = \cos(t)$, $t \in [0 , \pi]$, obtinem $dx = -(1 - x^2)^{1/2}\, dt = -\sin(t)\, dt$. Asadar pentru $k \neq n$ rezulta

$$< T_k , T_n >_0 = \int_{-1}^{1} T_k(x)\, T_n(x)\, dx = -\int_{\pi}^{0} \big(-\sin(t)\big) \cos(k\,t) \cos(n\,t)\, dt =$$

$$= \frac{1}{4} \int_{0}^{\pi} \Big[\sin((1+n+m)t) + \sin((1+n-m)t) + \sin((1-n+m)t) + \sin((1-n-m)t) \Big] dt$$

Dezvoltând aceasta expresie ca o suma de integrale si folosind relatia

$$\int_{0}^{\pi} \sin(\alpha\, t)\, dt = \frac{1 - (-1)^{\alpha}}{\alpha} = \begin{cases} 0 \ ; & \text{pentru } \alpha \text{ par} \\ 2/\alpha \ ; & \text{pentru } \alpha \text{ impar} \end{cases}$$

se obtine în final formula (11).

Corolarul 2. In raport cu produsul scalar (10), polinoamele Cebîsev de grad par sunt ortogonale cu polinoamele Cebîsev de grad impar. In cazul folosirii produsului scalar $< . , . >_0$ polinoamele Cebîsev numai de grad par (sau numai de grad impar) nu sunt ortogonale între ele .

Reamintim faptul ca în raport cu produsul scalar $< . , . >$ dat de relatia (7) toate polinoamele Cebîsev distincte, de grad par sau impar, erau ortogonale (Corolarul 1).

2. Aplicatii

2.1. Obtinerea coeficientilor polinoamelor Cebîsev

Aplicatia 1. Bazat pe formula recurenta (2), programul BASIC urmator permite calcularea coeficientilor unui polinom Cebîsev $T_n(x)$ având un grad n specificat, $n \in \mathbf{N}$.

```
REM  Calculul coeficientilor unui polinom Cebisev de grad n
INPUT "n = "; n:        OPTION BASE 0:        DIM v1(n + 1), v2(n + 1), v3(n + 1)
v1(0) = 1:      v2(0) = 1:      v2(1) = 0:      v3(0) = 1:      v2(1) = 0
IF ((n = 0) OR (n = 1)) THEN GOTO 1
FOR k = 2 TO n:         v3(k) = 0
  FOR j = 0 TO k - 1:       v3(j) = 2 * v2(j):        NEXT j
```

```
  FOR j = 0 TO k - 2:        v3(j + 2) = v3(j + 2) - v1(j):        NEXT j
  FOR j = 0 TO k - 1:        v1(j) = v2(j):                        NEXT j
  FOR j = 0 TO k:            v2(j) = v3(j):                        NEXT j
NEXT k
1 : PRINT "Coeficientii polinomului Cebisev incepind cu monomul de grad maxim :"
FOR j = 0 TO n:        PRINT v3(j); " ; "; :        NEXT j:        PRINT
```

Prezentul program, implementat ca o subrutina, poate fi cu succes utilizat în probleme de interpolare si prognoza.

Exemplul 1. Rulând programul descris în Aplicatia 1 s-au obtinut rezultatele de test

```
n = ?  10
Coeficientii polinomului Cebisev incepind cu monomul de grad maxim
 512 ;  0 ; -1280 ;  0 ;  1120 ;  0 ; -400 ;  0 ;  50 ;  0 ; -1 ;
```

In literatura de specialitate este mentionat polinomul Cebîsev $T_{10}(x)$ ca fiind de forma (Rade, Westergren, 1990, p. 232)

$$T_{10}(x) = 512\,x^{10} - 1280\,x^{8} + 1120\,x^{6} - 400\,x^{4} + 50\,x^{2} - 1$$

fapt ce este în concordanta cu rezultatele listate de programul BASIC precedent (Exemplul 1).

2.2. Determinarea valorii unui polinom Cebîsev într-un punct dat

Un polinom Cebîsev $T_n(x)$ ia valori relativ mici, numai în intervalul [-1 , 1], pentru orice $-1 \le x \le 1$. In schimb coeficientii polinomului T_n pot lua valori deosebit de mari (de exemplu coeficientul lui x^n din $T_n(x)$ este 2^{n-1} , Propozitia 2).

In aplicatiile practice suntem adesea interesati în a evalua $T_n(a)$ pentru o valoare fixata a , $a \in [-1 , 1]$. Concret, în vederea obtinerii lui $T_n(a)$ vom opta pentru urmatoarele variante : folosirea relatiei de recurenta (2) sau aplicarea schemei lui Horner pentru calculul valorii unui polinom ai carui coeficienti sunt cunoscuti.

In prima varianta vom considera $T_n(a) = s_n$, termenii sirului $\{ s_j \}_j$ fiind dedusi recurent dupa formula $s_{j+1} = 2a\,s_j - s_{j-1}$, respectând conditiile initiale $s_0 = 1$ si $s_1 = a$.

In cea de a doua varianta, conform schemei lui Horner pentru polinomul

$$T_n(x) = b_0\,x^n + b_1\,x^{n-1} + b_2\,x^{n-2} + \ldots + b_{n-2}\,x^2 + b_{n-1}\,x + b_n$$

avem $T_n(a) = v_n$ unde $v_0 = b_0$ iar $v_j = a\,v_{j-1} + b_j$, $j = 1, 2, \ldots , n$.

Aplicatia 2. Programul BASIC urmator estimeaza valoarea $T_{10}(a)$, $a \in \{ -1.0 , -0.9 , -0.8 , \ldots , 1.0 \}$ urmând cele doua variante mentionate. Pentru calculul termenilor s_n si v_n este nevoie de a se opera la fiecare iteratie j , $1 \le j \le n$, numai cu cel mult doua valori anterioare (adica s_{j-1} , s_{j-2} , respectiv v_{j-1}) fapt de care s-a tinut seama la proiectarea programului (nu s-a retinut memorie pentru toti termenii sirurilor $\{ s_j \}_j$, $\{ v_j \}_j$, variabilele s si v fiind scalari si nu vectori).

```
REM  Calculul valorii unui polinom Cebisev in punctul x
DATA 10,512,0,-1280,0,1120,0,-400,0,50,0,-1
READ n :              OPTION BASE 0:              DIM b(n)
```

```
FOR j = 0 TO n:          READ b(j):          NEXT j
PRINT "Polinom Cebisev de grad "; n
FOR ja = 0 TO 20:          a = -1! + ja / 10! :          s1 = 1:          s2 = a
FOR j = 2 TO n:       s3 = 2! * a * s2 - s1:       s1 = s2:       s2 = s3:       NEXT j
v = b(0):       FOR j = 1 TO n:       v = v * a + b(j):       NEXT j
PRINT USING "##.# ;   ##.###### ;   ##.###### ;   ##.######"; a; s3; v; s3 - v
NEXT ja
```

Exemplul 2. In Tabelul 1 sunt precizate estimatiile s_{10} , v_{10} ale valorii polinomului Cebâsev $T_{10}(a)$ folosind programul BASIC proiectat în Aplicatia 2. Constatam cu surprindere o diferenta între valorile estimate datorata cumularii erorilor de rotunjire. Reamintim faptul ca $T_n(x) \in [-1 , 1]$, iar coeficientii b_j , $0 \le j \le n$, ai polinomului Cebâsev $T_n(x)$ iau valori absolute foarte mari ce cresc proportional cu gradul n al polinomului.

Tabelul 1. Estimatiile s_{10} , v_{10} pentru $T_{10}(a)$ prin iteratie directa sau folosind schema lui Horner (Aplicatia 2)

a	s_{10}	v_{10}	$s_{10} - v_{10}$	a	s_{10}	v_{10}	$s_{10} - v_{10}$
-1.0	1.000000	1.000000	0.000000	-0.9	-0.200747	-0.200754	0.000007
-0.8	0.988497	0.988490	0.000007	-0.7	-0.099840	-0.099840	-0.000000
-0.6	-0.988497	-0.988494	-0.000003	-0.5	-0.500000	-0.500000	0.000000
-0.4	0.562346	0.562346	0.000000	-0.3	0.995523	0.995523	-0.000000
-0.2	0.428456	0.428456	0.000000	-0.1	-0.538893	-0.538893	0.000000
0.0	-1.000000	-1.000000	0.000000	0.1	-0.538893	-0.538893	0.000000
0.2	0.428456	0.428456	0.000000	0.3	0.995523	0.995523	-0.000000
0.4	0.562346	0.562346	0.000000	0.5	-0.500000	-0.500000	0.000000
0.6	-0.988497	-0.988494	-0.000003	0.7	-0.099840	-0.099840	-0.000000
0.8	0.988497	0.988490	0.000007	0.9	-0.200747	-0.200754	0.000007
1.0	1.000000	1.000000	0.000000				

Observatia 4. Pentru marirea acuratetii în evaluarea lui $T_n(a)$ se vor evita pe cât posibil operatiile aritmetice directe cu coeficientii b_j , $0 \le j \le n$, ai polinomului . In acest context vom prefera varianta $T_n(a) = s_n$, unde $s_{j+1} = 2a\, s_j - s_{j-1}$, aceasta deoarece valorile termenilor a, s_{j-1} , s_j sunt comparabile (a , s_{j-1} , $s_j \in [-1 , 1]$). In aceasta situatie erorile de rotunjire sunt relativ mici (pentru detalii a se vedea : Donald E. Knuth, Tratat de programare a calculatoarelor - Algoritmi seminumerici. Editura Tehnica, Bucuresti, 1983).

In nici un caz nu vom aplica schemei lui Horner pentru estimarea lui $T_n(a) = v_n$. Intr-o astfel de situatie vom asista la o cumulare substantiala a erorile aritmetice de rotunjire prin efectuarea unor calcule de tipul $v_j = a\, v_{j-1} + b_j$, unde $a . v_{j-1} , v_j \in [-1 , 1]$ iar $| b_j |$ poate fi deosebit de mare (de exemplu $b_0 = 2^{n-1}$) . Mentionam totodata si o posibila sursa suplimentara de erori, în faza de

initializare a procedurii de calcul, prin retinerea unui numar insuficient de cifre semnificative la memorarea valorilor coeficientilor b_j, $0 \le j \le n$, ai polinomului $T_n(x)$.

Tabelul 2. Estimatiile s_{15}, v_{15} pentru $T_{15}(a)$ prin iteratie directa sau aplicând schema lui Horner (Aplicatia 2)

a	s_{15}	v_{15}	$s_{15} - v_{15}$	a	s_{15}	v_{15}	$s_{15} - v_{15}$
-1.0	-1.000000	-1.000000	0.000000	-0.9	-0.885969	-0.885550	-0.000418
-0.8	0.974179	0.973928	0.000251	-0.7	-0.804842	-0.804838	-0.000004
-0.6	-0.225776	-0.225788	0.000012	-0.5	1.000000	1.000000	0.000000
-0.4	-0.110208	-0.110206	-0.000002	-0.3	-0.989935	-0.989936	0.000001
-0.2	0.120927	0.120927	0.000000	-0.1	0.997669	0.997670	-0.000000
0.0	0.000000	0.000000	0.000000	0.1	-0.997669	-0.997670	0.000000
0.2	-0.120927	-0.120927	-0.000000	0.3	0.989935	0.989936	-0.000001
0.4	0.110208	0.110206	0.000002	0.5	-1.000000	-1.000000	0.000000
0.6	0.225776	0.225788	-0.000012	0.7	0.804842	0.804838	0.000004
0.8	-0.974179	-0.973928	-0.000251	0.9	0.885969	0.885550	0.000418
1.0	1.000000	1.000000	0.000000				

Tabelul 3. Estimatiile s_{20}, v_{20} pentru $T_{20}(a)$ prin iteratie directa sau folosind schema lui Horner (Aplicatia 2)

a	s_{20}	v_{20}	$s_{20} - v_{20}$	a	s_{20}	v_{20}	$s_{20} - v_{20}$
-1.0	1.000000	1.000000	0.000000	-0.9	-0.919401	-0.967986	0.048584
-0.8	0.954251	0.964127	-0.009876	-0.7	-0.980064	-0.981753	0.001689
-0.6	0.954251	0.954555	-0.000304	-0.5	-0.500000	-0.500000	0.000000
-0.4	-0.367533	-0.367552	0.000019	-0.3	0.982130	0.982129	0.000001
-0.2	-0.632851	-0.632851	-0.000000	-0.1	-0.419189	-0 419189	0.000000
0.0	1.000000	1.000000	0.000000	0.1	-0.419189	-0.419189	0.000000
0.2	-0.632851	-0.632851	-0.000000	0.3	0.982130	0.982129	0.000001
0.4	-0.367533	-0.367552	0.000019	0.5	-0.500000	-0.500000	0.000000
0.6	0.954251	0.954555	-0.000304	0.7	-0.980064	-0.981753	0.001689
0.8	0.954251	0.964127	-0.009876	0.9	-0.919401	-0.967986	0.048584
1.0	1.000000	1.000000	0.000000				

Observatia 5. O crestere a gradului n al polinomului $T_n(x)$ va amplifica diferenta dintre estimatiile s_n si v_n obtinute în varianta iterativa, respectiv prin schema lui Horner. Tabelele 2 si 3 exemplifica experimental aceasta afirmatie, diferenta $s_n - v_n$ crescând surprinzator pentru $T_{20}(x)$

In toate cazurile studiate ($n \in \{ 10, 15, 20 \}$) eroarea maxima a rezultat pentru $a = 0.9$, respectiv $a = -0.9$.

Constatam experimental ca relatiile de simetrie $T_n(x) = T_n(-x)$ pentru n par, respectiv $T_n(x) = -T_n(-x)$ pentru n impar, se mentin si în cadrul diferentei $s_n - v_n$, dintre cele doua estimatii ale lui $T_n(x)$, $n \in \{ 10, 15, 20 \}$.

2.3. Alegerea punctelor de interpolare pentru polinomul Lagrange

Fie $f : [a, b] \rightarrow \mathbf{R}$ o functie derivabila de n+1 ori ce se doreste a fi aproximata pe intervalul $[a, b]$ cu un polinom $L_n(x)$ de gradul n, astfel încât în n+1 puncte date x_j, $a \le x_j \le b$, sa avem egalitatea $f(x_j) = P_n(x_j)$, $0 \le j \le n$. In capitolul dedicat aproximarii functiilor a fost discutata forma polinomului de interpolare Lagrange $L_n(x)$ precizându-se o evaluare a restului $R_n(x)$ rezultat în urma operatiunii de aproximare a functiei f(x),

$$R_n(x) = f(x) - L_n(x) = [f^{(n+1)}(\alpha) / (n + 1)!] W_{n+1}(x) \qquad (12)$$

unde $\alpha \in (a, b)$ iar polinomul $W_{n+1}(x)$ depinde de cele n+1 puncte de interpolare x_j specificate,

$$W_{n+1}(x) = (x - x_0) (x - x_1) (x - x_2) \dots (x - x_{n-1}) (x - x_n) \qquad (13)$$

Daca $M_{n+1} = \sup \{ | f^{(n+1)}(x) | , a \le x \le b \}$ atunci

$$\sup \{ | R_n(x) | , a \le x \le b \} \le [M_{n+1} / (n+1)!] \sup \{ | W_{n+1}(x) | , a \le x \le b \} \qquad (14)$$

Eroarea facuta în procesul de interpolare se va micsora daca fluctuatiile polinomului $W_{n+1}(x)$ în jurul lui zero vor fi cât mai mici. Acest lucru se va realiza în situatia în care punctele de interpolare x_j sunt chiar cele n+1 zerouri ale polinomului Cebîsev $C_{n+1}(x)$ ce este definit pe intervalul $[a, b]$ (Propozitiile 2-4, Observatia 2). In acest caz, cum polinoamele $W_{n+1}(x)$ si $C_{n+1}(x)$ de grad n+1 au aceleasi zerouri rezulta ca ele difera numai printr-un factor de proportionalitate λ, adica $W_{n+1}(x) = \lambda C_{n+1}(x)$, $\forall x \in [a, b]$. Deoarece $C_{n+1}(x) = T_{n+1}((2x - b - a) / (b - a))$ iar coeficientul monomului x^{n+1} din polinomul $T_{n+1}(x)$ este 2^n (Propozitia 2) deducem ca $2^{2n+1} / (b - a)^{n+1}$ este factorul monomului de grad maxim a polinomului Cebâsev $C_{n+1}(x)$. Asadar pentru $a \le x \le b$ avem

$$W_{n+1}(x) = [(b - a)^{n+1} / 2^{2n+1}] C_{n+1}(x) = [(b - a)^{n+1} / 2^{2n+1}] T_{n+1}((2x - b - a) / (b - a))$$

Tinând seama ca $| T_{n+1}(t) | \le 1$, $\forall t \in [-1, 1]$ (Propozitia 2), egalitatea anterioara conduce la urmatorul majorant pentru restul de interpolare dat de (14) :

Propozitia 7. In cazul folosirii zerourilor polinoamelor Cebîsev drept noduri de interpolare atunci valoarea maxima a restului aproximarii Lagrange satisface inegalitatea

$$\sup \{ | f(x) - L_n(x) | , a \le x \le b \} \le [(b - a)^{n+1} M_{n+1}] / [2^{2n+1} (n+1) !] \qquad (15)$$

Observatia 6. Utilizarea zerourilor polinoamelor Cebîsev ca puncte de interpolare mareste substantial acuratetea aproximarilor. S-ar putea însa ca alegând alte puncte de interpolare sa se obtina rezultate mai precise. In adevar, considerând o functie oarecare f(x) si un polinom arbitrar $P_n(x)$ de grad n, maximul diferentei $| f(x) - P_n(x) |$, $x \in [a, b]$, nu se realizeaza neaparat în punctele y_j de extrem pentru polinomul Cebîsev $C_{n+1}(x)$. Vom discuta acest aspect pe un exemplu concret.

Aplicatia 3. Fie $f(x) = e^x$, $-1 \le x \le 1$. Vom aproxima functia f(x) cu un polinom de interpolare Lagrange $L_2(x)$ de gradul 2, având nodurile de interpolare x_0, x_1, x_2 situate în intervalul [-1, 1].

Asadar $L_2(x)$ este de forma

$$L_2(x) = f(x_0)\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + f(x_1)\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + f(x_2)\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

restul $R_2(x)$ al aproximarii având expresia (12).

Polinomul Cebîsev $T_3(x) = W_3(x)$ are zerourile $x_0 = \cos(\pi / 6) = 3^{1/2} / 2$, $x_1 = \cos(3\pi / 6) = 0$, $x_2 = \cos(5\pi / 6) = -x_0 = -3^{1/2} / 2$ (Propozitia 2, formula (4)), adica $x_0 = -\alpha$, $x_1 = 0$, $x_2 = \alpha$, unde $\alpha = 3^{1/2} / 2$.

In vederea justificarii Observatiei 6 vom studia clasa polinoamelor de interpolare Lagrange $L_2(x)$, $-1 \le x \le 1$, cu noduri simetrice, adica

$$P_2(x; \alpha) = [\, f(-\alpha)\, x\, (x-\alpha) \;-\; 2\, f(0)\, (x-\alpha)\, (x+\alpha) \;+\; f(\alpha)\, x\, (x+\alpha)\,] / (\, 2\, \alpha^2\,) \qquad (16)$$

unde $0 < \alpha \le 1$. Un element al acestei clase este polinomul $P_2(x; 3^{1/2}/2)$ ce aproximeaza functia $f(x)$ si ale carui noduri de interpolare sunt chiar zerourile polinomului Cebîsev $T_3(x)$.

Considerând functia $f(x) = e^x$, $-1 \le x \le 1$, vom arata experimental ca exista $\beta \in (0, 1]$, $\beta \ne 3^{1/2} / 2$, astfel încât

$$\sup \{\, |\, f(x) - P_2(x; \beta)\, |,\ -1 \le x \le 1 \,\} = \inf \{\, \sup \{\, |\, f(x) - P_2(x; \alpha)\, |,\ -1 \le x \le 1 \,\},\ 0 < \alpha \le 1 \,\}$$

adica polinomul de aproximare cu "noduri Cebîsev" nu da totusi "eroarea absoluta maxima" cea mai mica.

Urmatorul program BASIC determina valoarea optima $\beta \in \{\, k / 1000 \mid 1 \le k \le 1000 \,\}$ atunci când $x \in \{\, -1 + 2j / 2000 \mid 0 \le j \le 2000 \,\}$

```
DECLARE FUNCTION p! (x!, a!)
DECLARE FUNCTION f! (x!)
REM  Compararea polinoamelor de aproximare Lagrange de grad 3 cu noduri simetrice
mx = 2000 :        ma = 1000 :        xmm = -2 :        amm = -2 :        eamin = 999999
FOR ja = 1 TO ma:        a = ja / ma :        xmax = -2 :        exmax = 0
FOR jx = 0 TO mx :        x = -1 + 2 * jx / mx :        ex = f(x) - p(x, a)
IF ABS(ex) > ABS(exmax) THEN        xmax = x :        exmax = ex
NEXT jx
IF ABS(exmax) < ABS(eamin) THEN        xmm = xmax :        amm = a :        eamin = exmax
NEXT ja
PRINT USING "xmm = ##.###  ;  amm = ##.###  ;  mina maxx | F(x) - P(x,a) | = | ##.#######  |";
            xmm; amm; eamin
FUNCTION p (x, a)
  p = (f(-a) * x * (x - a) - 2 * f(0) * (x - a) * (x + a) + f(a) * x * (x + a)) / (2 * a * a)
END FUNCTION
FUNCTION f (x) :        f = EXP(x) :        END FUNCTION
```

Exemplul 3. Prin rularea programului BASIC definit în Aplicatia 3 s-au obtinut relatiile

$$\inf \{\, \sup \{\, |\, f(-1+2j/2000) - P_2(-1+2j/2000;\ k/1000)\, |,\ 0 \le j \le 2000 \,\},\ 1 \le \alpha \le 1000 \,\} =$$

$$= |\, f(0.530) - P_2(0.530\ ;\ 0.879)\, | = |\ -0.0517325\ | < |\ 0.0564778\ | = |\, f(1.0) - P_2(1.0\ ;\ 0.866)\, | =$$

$$= \sup \{\, |\, f(x) - P_2(x; 0.866)\, |,\ -1 \le x \le 1 \,\} < |\ 0.2182071\ | = |\, f(1.0 - P_2(1.0\ ;\ 0.02)\, | =$$

$$= \sup \{\, |\, f(x) - P_2(x; 0.02)\, |,\ -1 \le x \le 1 \,\}$$

Asadar $\beta \approx 0.879 \neq 0.866 = 3^{1/2} / 2$ confirmându-se astfel Observatia 6. In acest caz concret, optând pentru noduri de interpolare Cebîsev, eroarea de aproximare descreste de circa 4 ori (pornind de la o eroare maxima de 0.2182071 pentru $\alpha = 0.02$ se ajunge la o eroare de cel mult 0.0564778 daca $\alpha = 0.866 \approx 3^{1/2} / 2$).

Observatia 7. Dezvoltarea în serie Taylor în jurul punctului fixat $c \in [a, b]$ poate fi de asemenea folosita la aproximarea unei functii date $f : [a, b] \rightarrow \mathbf{R}$ derivabila de n+1 ori.

Astfel prin utilizarea polinomului Taylor $Q_n(x)$, de gradul n,

$$Q_n(x) = f(c) + \frac{f^{(1)}(c)}{1!}(x-c) + \frac{f^{(2)}(c)}{2!}(x-c)^2 + \ldots + \frac{f^{(n)}(c)}{n!}(x-c)^n \qquad (17)$$

rezulta o eroare $E_n(x) = [\, f^{(n+1)}(\tau) / (n+1)\, !\,]\, (x - c)^{n+1}$.

Un majorant al maximului erorii $E_n(x)$ este dat de expresia

$$\sup \{\, |f(x) - Q_n(x)|\, ,\ a \le x \le b \,\} \le [\, \max \{\, (c-a)^{n+1}, (b-c)^{n+1} \,\}\, M_{n+1}\,] / [(n+1)\, !\,] \qquad (18)$$

Din formula (18) deducem ca cea mai mica valoare pentru maximul erorii $|E_n(x)|$ se realizeaza daca $c = (b - a) / 2$, adica

$$\sup \{\, |f(x) - Q_n(x)|\, ,\ a \le x \le b \,\} \le [\, (b - a)^{n+1} M_{n+1}\,] / [\, 2^{n+1} (n+1)\, !\,] \qquad (19)$$

Folosirea polinoamelor de interpolare Lagrange având drept puncte de interpolare zerourile polinoamelor Cebîsev conduce la rezultate mai precise comparativ cu utilizarea aproximarilor Taylor. Inegalitatile (15) si (19) precizeaza o posibila micsorare a erorii de aproximare de cel mult 2^n ori.

Exemplul 4. In cazul functiei $f(x) = e^x$, $x \in [-1, 1]$, considerând $c = 0$ si $n = 2$ rezulta polinomul Taylor $Q_2(x) = 1 + x + x^2 / 2$. Cum functia $h(x) = e^x - 1 - x - x^2 / 2$ este crescatoare (derivata sa fiind nenegativa) deducem ca

$$\sup \{\, |f(x) - Q_2(x)|\, ,\ -1 \le x \le 1 \,\} = \sup \{\, |h(x)|\, ,\ -1 \le x \le 1 \,\} = |h(1)| = 0.2182817$$

Asadar, se constata efectiv ca eroarea maxima absoluta de $|0.2182817|$ rezultata în cazul aproximarii Taylor $Q_2(x)$ pentru functia $f(x) = e^x$ este de circa $4 = 2^2$ ori mai mare decât eroarea maxima absoluta de $|0.0564778|$ rezultata în cazul utilizarii polinoamului de interpolare Lagrange $P_2(x, 0.866)$ cu noduri de tip Cebîsev (formula (16)).

SCHEME CU DIFERENTE

1. Estimarea derivatei unei functii

Fie functia $f : [a, b] \to R$ derivabila de un numar n de ori pe intervalul [a, b]. Sa estimam derivata $f^{(k)}(c)$ de ordin k a functiei f(x) într-un punct $c \in (a, b)$.

Cunoscând valorile f(x), $x \in [a, b]$, vom evalua derivatele $f^{(1)}(c)$ si $f^{(2)}(c)$ de primul si al doilea ordin calculate în punctul c.

Propunem urmatoarea aproximare a derivatei $f^{(1)}(c)$,

$$f^{(1)}(c) = \lim_{x \to c} \frac{f(x) - f(c)}{x - c} \approx \frac{f(c+h) - f(c)}{h} = \Delta^{(1)}(c) \qquad (1)$$

unde valoarea parametrului h, $h > 0$, este fixata.

In mod evident exista si alte formule care sa estimeze valoarea derivatei $f^{(1)}(c)$ calculata într-un punct c precizat. Pornind de la expresia (1) sugeram si aproximatia

$$\Delta_0^{(1)}(c) = [\, f(c + h/2) - f(c - h/2) \,] / h \qquad (2)$$

O formula de aproximare devine operationala în momentul în care este previzibila eroarea rezultata în urma aplicarii sale. In cele ce urmeaza vom stabili acuratetea aproximatiilor $\Delta^{(1)}(c)$ si $\Delta_0^{(1)}(c)$.

Din dezvoltarea în serie Taylor a functiei f(x) rezulta $\alpha \in (c, c+h)$, $\beta \in (c, c+h/2)$, $\gamma \in (c-h/2, c)$ astfel încât

$$f(c+h) = f(c) + \frac{f^{(1)}(c)}{1!} h + \frac{f^{(2)}(\alpha)}{2!} h^2$$

$$f(c+h/2) = f(c) + \frac{f^{(1)}(c)}{1!} \frac{h}{2} + \frac{f^{(2)}(c)}{2!} \frac{h^2}{4} + \frac{f^{(3)}(\beta)}{3!} \frac{h^3}{8}$$

$$f(c-h/2) = f(c) - \frac{f^{(1)}(c)}{1!} \frac{h}{2} + \frac{f^{(2)}(c)}{2!} \frac{h^2}{4} - \frac{f^{(3)}(\gamma)}{3!} \frac{h^3}{8}$$

si deci

$$\frac{f(c+h) - f(c)}{h} = f^{(1)}(c) + \frac{f^{(2)}(\alpha)}{2} h$$

$$\frac{f(c+h/2) - f(c-h/2)}{h} = f^{(1)}(c) + \frac{f^{(3)}(\beta) + f^{(3)}(\gamma)}{48} h^2$$

Daca M_k = maximum { $|f^{(k)}(x)|$, $| a \le x \le b$ } din relatiile de mai sus deducem

Propozitia 1. Pastrând notatiile anterioare, erorile în valoare absoluta rezultate din aproximarea derivatei $f^{(1)}(c)$ prin cantitatile $\Delta^{(1)}(c)$, respectiv $\Delta_0^{(1)}(c)$, satisfac inegalitatile

$$\varepsilon^{(1)} = | f^{(1)}(c) - \Delta^{(1)}(c) | = | h f^{(2)}(\alpha) / 2 | \le h M_2 / 2 \qquad (3)$$

$$\varepsilon_0^{(1)} = | f^{(1)}(c) - \Delta_0^{(1)}(c) | = | h^2 [f^{(3)}(\beta) + f^{(3)}(\gamma)] / 48 | \le h^2 M_3 / 24 \qquad (4)$$

Observatia 1. Din formulele (3) si (4) rezulta ca pentru valori h foarte mici, cantitatea $\Delta_0^{(1)}(c)$ ar putea fi mai apropiata de $f^{(1)}(c)$ decât expresia $\Delta^{(1)}(c)$. Motivam aceasta afirmatie prin faptul ca eroarea absoluta $\varepsilon_0^{(1)}$ este în general mai mica decât eroarea absoluta $\varepsilon^{(1)}$.

Tehnica de aproximare prezentata se extinde fara dificultate si pentru derivatele de ordin superior ale unei functii date f(x), $x \in [a, b]$. In acest sens definim

$$\Delta^{(2)}(c) = [\Delta^{(1)}(c+h) - \Delta^{(1)}(c)] / h = [f(c+2h) - 2f(c+h) + f(c)] / h^2 \quad (5)$$

$$\Delta_0^{(2)}(c) = [\Delta_0^{(1)}(c+h/2) - \Delta_0^{(1)}(c-h/2)] / h = [f(c+h) - 2f(c) + f(c-h)] / h^2 \quad (6)$$

Va trebui sa alegem una dintre aceste doua aproximatii ale derivatei $f^{(2)}(c)$. Vom apela la dezvoltari în serie Taylor în jurul punctului c pentru functia f(x), adica

$$f(c+h) = f(c) + \frac{f^{(1)}(c)}{1!} h + \frac{f^{(2)}(c)}{2!} h^2 + \frac{f^{(3)}(\lambda)}{3!} h^3$$

$$f(c-h) = f(c) - \frac{f^{(1)}(c)}{1!} h + \frac{f^{(2)}(c)}{2!} h^2 - \frac{f^{(3)}(\mu)}{3!} h^3$$

$$f(c+2h) = f(c) + \frac{f^{(1)}(c)}{1!} (2h) + \frac{f^{(2)}(c)}{2!} (2h)^2 + \frac{f^{(3)}(\varphi)}{3!} (2h)^3$$

unde $\lambda \in (c, c+h)$, $\mu \in (c-h, c)$, $\varphi \in (c, c+2h)$.

Tinând seama de ultimele relatii, în urma unui calcul elementar deducem

$$\Delta^{(2)}(c) = \frac{f(c+2h) - 2f(c+h) + f(c)}{h^2} = f^{(2)}(c) + \frac{8f^{(3)}(\varphi) - 2f^{(3)}(\lambda)}{6} h$$

$$\Delta_0^{(2)}(c) = \frac{f(c+h) - 2f(c) + f(c-h)}{h^2} = f^{(2)}(c) + \frac{f^{(3)}(\lambda) - f^{(3)}(\mu)}{6} h$$

Prelucrând aceste egalitati se obtine :

Propozitia 2. Estimarea valorii derivatei $f^{(2)}(c)$ prin cantitatile $\Delta^{(2)}(c)$, respectiv $\Delta_0^{(2)}(c)$ conduce la urmatoarele evaluari ale modulului erorii de aproximare

$$\varepsilon^{(2)} = |f^{(2)}(c) - \Delta^{(2)}(c)| = h|[8f^{(3)}(\varphi) - 2f^{(3)}(\lambda)] / 6| \le 5hM_3 / 3 \quad (7)$$

$$\varepsilon_0^{(2)} = |f^{(2)}(c) - \Delta_0^{(2)}(c)| = h|[f^{(3)}(\lambda) - f^{(3)}(\mu)] / 6| \le hM_3 / 3 \quad (8)$$

Observatia 2. Comparativ cu $\Delta^{(2)}(c)$, folosirea aproximarii $\Delta_0^{(2)}(c)$ pentru estimarea derivatei de ordin doi $f^{(2)}(c)$ este, de regula, mai precisa, fapt evidentiat si de inegalitatile (7) si (8). In general se obtine $\varepsilon_0^{(2)} < \varepsilon^{(2)}$.

Aplicatia 1. Rezultatele teoretice prezentate (Propozitiile 1-2, Observatiile 1-2) au fost confirmate si din punct de vedere experimental. A se vedea în acest sens Tabelul 1 unde s-a considerat functia infinit derivabila $f(x) = x \exp(x^2)$ definita pe multimea **R**. Se constata experimental satisfacerea inegalitatilor $\varepsilon_0^{(k)} < \varepsilon^{(k)}$, aproximarea $\Delta_0^{(k)}(c)$ fiind preferata în locul utilizarii variantei $\Delta^{(k)}(c)$, $k = 1, 2$. In plus o micsorare a valorii pasului de discretizare h antreneaza o îmbunatatire a acuratetii aproximarilor propuse (Tabelul 1) fapt ce este în concordanta cu formulele (3)-(4), (7)-(8)

Rezultatele afisate în Tabelul 1 au fost obtinute în urma rularii urmatorului program BASIC

```
REM  Evaluarea erorilor la estimarea derivatelor unei functii
DECLARE FUNCTION f! (x!)
```

```
DECLARE FUNCTION f1! (x!) :          DECLARE FUNCTION f2! (x!)
FOR c = 1 TO 3 :          FOR j = 1 TO 3 :          h = 10 ^ (-j)
d1 = (f(c + h) - f(c)) / h :          d01 = (f(c + h / 2) - f(c - h / 2)) / h
d2 = (f(c + 2 * h) - 2 * f(c + h) + f(c)) / (h * h)
d02 = (f(c + h) - 2 * f(c) + f(c - h)) / (h * h)
e1 = ABS(f1(c) - d1) :          e01 = ABS(f1(c) - d01)
e2 = ABS(f2(c) - d2) :          e02 = ABS(f2(c) - d02)
PRINT USING "## ; #.### ; #.###^^^^ ; #.###^^^^ ; #.###^^^^ ;  #.###^^^^";
                c; h; e1; e01; e2; e02
NEXT j :          NEXT c

FUNCTION f (x) :          f = x * EXP(x * x) :          END FUNCTION
REM  Derivata f1(x) a functiei f(x)
FUNCTION f1 (x) :          f1 = (1 + 2 * x * x) * EXP(x * x) :          END FUNCTION
REM  Derivata f2(x) de ordinul al doilea a functiei f(x)
FUNCTION f2 (x) :          f2 = (6 * x + 4 * x ^ 3) * EXP(x * x) :          END FUNCTION
```

Tabelul 1. Erorile de aproximare $\varepsilon^{(k)}$ si $\varepsilon_0^{(k)}$ ale derivatei $f^{(k)}(c)$ prin $\Delta^{(k)}(c)$, respectiv $\Delta_0^{(k)}(c)$, $k = 1, 2$, unde $f(x) = x \exp(x^2)$.

c	h	$\varepsilon^{(1)}$	$\varepsilon_0^{(1)}$	$\varepsilon^{(2)}$	$\varepsilon_0^{(2)}$
1	0.100	0.155E+01	0.431E-01	0.134E+02	0.356E+00
1	0.010	0.138E+00	0.429E-03	0.106E+01	0.644E-02
1	0.001	0.141E-01	0.215E-03	0.957E+00	0.712E+00
2	0.100	0.144E+03	0.525E+01	0.179E+04	0.586E+02
2	0.010	0.122E+02	0.572E-01	0.130E+03	0.635E+00
2	0.001	0.117E+01	0.351E-01	0.131E+03	0.525E+02
3	0.100	0.653E+05	0.296E+04	0.112E+07	0.429E+05
3	0.010	0.522E+04	0.328E+02	0.735E+05	0.418E+03
3	0.001	0.500E+03	0.113E+02	0.435E+05	0.496E+03

2. Rezolvarea ecuatiilor diferentiale

Schemele cu diferente sunt adesea folosite cu succes la obtinerea unei aproximatii $\{ y_k \}_k$ ce exprima discretizarea solutiei $y(x)$ a unei ecuatii diferentiale în puncte $x_k \in [a, b]$ precizate. Vom exemplifica o astfel de procedura în cazul unei ecuatii diferentiale ordinare simple a carei solutie $y(x)$ este definita pe un întreg interval $[a, b]$. Functia $y(x)$ verifica egalitatea

$$y^{(1)}(x) + p(x)\, y(x) = f(x) \qquad (9)$$

pentru orice $a \le x \le b$, respectând în plus conditia initiala $y(a) = c$, valoarea c fiind data.

Pentru $n \in \mathbf{N}$ fixat, consideram pasul $h = (b - a) / n$ de discretizare a solutiei $y(x)$ ce satisface ecuatia (9). In loc de a evalua functia $y(x)$ în fiecare punct x din intervalul $[a , b]$ ne vom multumi sa estimam numai sirul finit de valori $\{ y_k \}_k$, $1 \le k \le n$, unde $y_k = y (x_k)$ cu $x_k = a + k h$.

Tinând seama de rezultatele anterioare derivata $y^{(1)}(x)$ a solutiei $y(x)$ va fi aproximata prin diferenta $[y_{k+1} - y_k] / h$, deoarece $[y_{k+1} - y_k] / h \approx y^{(1)}(a + k h)$ (formula (1)).

Asadar în locul egalitatii $y^{(1)}(a + k h) + p(a + k h) y(a + k h) = f(a + k h)$ rezultata din ecuatia (9) pentru $x = x_k = a + k h$, putem considera relatia $[y_{k+1} - y_k] / h + p_k y_k = f_k$ unde $p_k = p(x_k)$ si $f_k = f(x_k)$, pentru orice k , $1 \le k \le n$

Prin urmare sirul $y_1 , y_2 , \dots , y_{n-1} , y_n$ ce estimeaza valorile $y(a + k h)$, $1 \le k \le n$, ale solutiei $y(x)$ a ecuatiei diferentiale (9), poate fi determinat aplicând succesiv formula

$$y_{k+1} = (1 - h p_k) y_k + h f_k \qquad (10)$$

si tinând seama de conditia initiala $y_0 = c = y(a)$.

Aplicatia 2. Programul BASIC urmator determina sirul $\{ y_k \}_k$, $1 \le k \le n$, pentru functii $f(x)$ si $p(x)$ date, $a \le x \le b$.

Exemplificam procesul de discretizare mentionat considerând $f(x) = \exp(x^2)$, $p(x) = - 2x$, $a \le x \le b$. In acest caz solutia "exacta" a ecuatiei diferentiale (9), justificata prin calcul direct, este $y(x) = x \exp(x^2)$. Sirul $\{ y_k \}_k$, $1 \le k \le n$, este obtinut prin aplicarea formulei (10) pentru pasul de discretizare $h = (b - a)/n$, unde $y_0 = c = a \exp(a^2)$.

```
REM  Rezolvarea ecuatiilor diferentiale utilizând scheme cu diferente
REM  Solutia y(x) a ecuatiei diferentiale      y'(x) + p(x) y(x) = f(x)
REM  unde   a <= x <= b ,   y(a) = g(a)   si  n  puncte de discretizare
DECLARE FUNCTION g! (x!) :          DECLARE FUNCTION f! (x!)
DECLARE FUNCTION p! (x!)
INPUT "a, b, n ="; a, b, n :          y# = g(a) :          h# = (b - a) / n
FOR k = 1 TO n :          x = a + k * h#
y# = (1# - h# * p(x)) * y# + h# * f(x) :                NEXT k
PRINT USING "###### ; #.##### ; ##.# ; ##.# ; ####.### ; ####.### ; ####.###";
   n; h#; a; b; g(a); g(b); y#
FUNCTION f (x) :          f = EXP(x * x) :          END FUNCTION
FUNCTION g (x) :          g = x * EXP(x * x) :          END FUNCTION
FUNCTION p (x) :          p = -2 * x :          END FUNCTION
```

Functia $g(x)$ din acest program este chiar solutia $y(x) = x \exp(x^2)$ a ecuatiei diferentiale (9). Verificarea corectitudinii metodei expuse se realizeaza comparând valoarea exacta $y(b) = g(b)$ cu valoarea aproximativa y_n rezultata în urma unui proces de discretizare cu pasul $h = (b - a)/n$.

Micsorarea pasului de discretizare h conduce la o acuratete sporita a aproximatiei y_n a valorii exacte $y(b)$ fapt evidentiat si de rezultatele din Tabelul 2. Aproximatiile y_n sunt relativ stabile în cazul în care parametrii b si h sunt mentinuti constanti dar în schimb marginea din stânga a intervalului $[a , b]$ variaza.

Observatia 3. In cadrul a doua sectiuni din prezenta lucrare vor fi sugerate si alte tehnici de rezolvarea ecuatiilor diferentiale, metode ce nu sunt bazate pe scheme cu diferente. In plus utilizând

schemele cu diferente vor fi indicate proceduri de solutionare a ecuatilor diferentiale liniare si pentru alte tipuri de conditii la limita (nu neaparat de forma y(a) = c ; formula (9)).

Tabelul 2. Obtinerea aproximatiei y_n (relatia recurenta (10)) a valorii y(b) atunci când pasul de discretizare h variaza (y(x) este solutia ecuatiei diferentiale (9)).

n	h	a	b	y(a)	y(b)	y_n
10	0.10000	1.0	2.0	2.718	109.196	85.976
100	0.01000	1.0	2.0	2.718	109.196	106.045
1000	0.00100	1.0	2.0	2.718	109.196	108.870
10000	0.00010	1.0	2.0	2.718	109.196	109.164
100000	0.00001	1.0	2.0	2.718	109.196	109.193
10000	0.00020	0.0	2.0	0.000	109.196	109.131
20000	0.00010	0.0	2.0	0.000	109.196	109.164
100000	0.00002	0.0	2.0	0.000	109.196	109.190
200000	0.00001	0.0	2.0	0.000	109.196	109.193
10	0.30000	0.5	3.5	0.642	731434.500	121189.366
100	0.03000	0.5	3.5	0.642	731434.500	439469.334
1000	0.00300	0.5	3.5	0.642	731434.500	690052.794
10000	0.00030	0.5	3.5	0.642	731434.500	727128.002
100000	0.00003	0.5	3.5	0.642	731434.500	731002.116
300000	0.00001	0.5	3.5	0.642	731434.500	731290.337

APROXIMAREA INTEGRALEI UNEI FUNCTII

1. Metoda exacta de rezolvare

Fie o functie f(x) definita pe un interval [a , b] ce ia valori reale. Daca F(x) este o primitiva a functiei f(x) , adica derivata $F^{(1)}(x)$ este chiar f(x) , atunci pentru calculul integralei I a functiei f(x) , pe intervalul [a , b] , se poate folosi formula

$$I = \int_a^b f(x)\,dx = F(b) - F(a) \qquad (1)$$

Mentionam faptul ca primitiva F(x) nu este unic determinata deoarece orice expresie $F_1(x) = F(x) + c$ este de asemenea o primitiva a functiei f(x) , unde c este o constanta reala arbitrara. Folosirea unei alte primitive $F_1(x)$ a functiei f(x) pentru estimarea integralei I data de formula (1) nu afecteaza însa valoarea acestei integrale fiind adevarata egalitatea $F(b) - F(a) = F_1(b) - F_1(a)$

Exemplul 1. In cazul în care se poate preciza o primitiva F(x) a lui f(x) atunci vom putea utiliza formula (1) în vederea estimarii cu ajutorul calculatorului a integralei I . Astfel pentru o functie f(x) de tip polinomial

$$f(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_{n-2} x^2 + a_{n-1} x + a_n \qquad (2)$$

o primitiva a sa este de forma

$$F(x) = a_0 x^{n+1}/(n+1) + a_1 x^n/n + a_2 x^{n-1}/(n-1) + \ldots + a_{n-1} x^2/2 + a_n x \qquad (3)$$

Determinarea valorilor polinomului F(x) în punctele a si b se poate realiza cu ajutorul schemei lui Horner. Astfel deducerea valorii F(c) se va efectua iterativ, evaluând dupa procedura Horner urmatoarea expresie polinomiala

$$F(c) = [\, a_0 c^n/(n+1) + a_1 x^{n-1}/n + a_2 x^{n-2}/(n-1) + \ldots + a_{n-1} x/2 + a_n \,] \cdot c \qquad (4)$$

Aplicatia 1. Procedura BASIC **intp** va calcula integrala I pe intervalul [c1, c2] din polinomul f(x) de gradul n (formula (2)), cu ajutorul formulei (4). Polinomul f(x) va fi precizat specificându-se efectiv valorile celor n + 1 coeficienti ai sai, $a_j \equiv a(j)$, $0 \le j \le n$.

Listam în continuare programul sursa.

```
DECLARE FUNCTION intp! (n!, a!(), c1!, c2!)
REM  Calculul integralei unui polinom  P(x)  de coeficienti  a(j)  pe intervalul  [ c1 , c2 ]
CLS :          INPUT "Gradul polinomului  = "; k
OPTION BASE 0 :    DIM a(k)
PRINT "Definiti coeficientii polinomului de integrat :"
FOR j = 0 TO k :    PRINT "a("; j; " ) = ";  :    INPUT a(j) :    NEXT j
INPUT "Specificati limitele de integrare :      c1 , c2 = "; c1, c2
PRINT "Integrala pe domeniul  ["; c1; " , "; c2; " ] din polinomul  P(x) ="; intp(k, a(), c1, c2)

FUNCTION intp (n, a(), c1, c2)
REM  Calculul integralei unui polinom de gradul  n  cu coeficienti  a(j)
REM  Intervalul de integrare este  [ c1 , c2 ]
REM  Calculul valorii  s1  a primitivei in punctul  c1
s1 = a(0) / (n + 1) :    FOR j = 1 TO n :    s1 = c1 * s1 + a(j) / (n + 1 - j) :    NEXT j
s1 = s1 * c1
```

```
REM  Calculul valorii  s2  a primitivei in punctul  c2
s2 = a(0) / (n + 1) :    FOR j = 1 TO n :    s2 = c2 * s2 + a(j) / (n + 1 - j)      NEXT j
s2 = s2 * c2
REM  Estimarea integralei
intp = s2 - s1 :                    END FUNCTION
```

Indicii elementelor vectorului a ce caracterizeaza coeficientii polinomului f(x) iau valori cuprinse între 0 si n . Având în vedere acest aspect s-a introdus în program specificatia OPTION BASE 0 ce va permite referirea si la elementul $a_0 \equiv a(0)$.

Exemplul 2. Utilizând programul prezentat în Aplicatia 1 vom evalua integralele I_1 si I_2 ,

$$I_1 = \int_0^2 3\,dx \qquad\qquad I_2 = \int_{-1}^2 (2x^2 - 3x + 1)\,dx$$

Polinoamele $F_1(x) = 3x$, $F_2(x) = 2x^3 / 3 - 3x^2 / 2 + x$ sunt primitive ale functiilor $f_1(x) = 3$ si respectiv $f_2(x) = 2x^2 - 3x + 1$. Printr-un calcul direct se obtin valorile integralelor I_1 si I_2 , anume $I_1 = 3 . 2 - 3 . 0 = 6$ si $I_2 = (16 / 3 - 6 + 2) - (-2 / 3 - 3 / 2 - 1) = 9 / 2$

Aceste rezultate sunt confirmate practic prin utilizarea functiei BASIC **intp** (n, a, c1, c2) . Listam în continuare varianta de test.

```
Gradul polinomului  = ?  0
Definiti coeficientii polinomului de integrat :    a( 0 ) = ?  3
Specificati limitele de integrare :      c1 , c2 = ?  0 , 2
Integrala pe domeniul [ 0 , 2 ] din polinomul  P(x) = 6
Gradul polinomului  = ?  2
Definiti coeficientii polinomului de integrat :   a( 0 ) = ?  2    a( 1 ) = ? -3    a( 2 ) = ?  1
Specificati limitele de integrare :      c1 , c2 = ? -1 , 2
Integrala pe domeniul [-1 , 2 ] din polinomul  P(x) = 4.5
```

2. Metode aproximative de calculul integralei

2.1. O procedura generala

Algoritmul anterior necesita precizarea unei primitive **F(x) a functiei f(x)** , conditie ce practic nu poate fi îndeplinita în cele mai multe situatii. In fapt nu ne intereseaza sa determinam chiar valoarea exacta a integralei I ci sa precizam o aproximatie I* a sa cu o eroare ε impusa, diferenta | I - I* | putând fi micsorata oricât de mult se doreste, adica | I - I* | < ε cu ε > 0 arbitrar, oricât de mic

Urmând Exemplul 1 ce permite evaluarea integralei unui polinom arbitrar P(x) , intentionam sa aproximam integrala I dintr-o functie oarecare f(x) (formula (1)) printr-o integrala I_0 a unui polinom de "aproximare" P(x) ce va trebui însa a fi precizat în raport cu fiecare functie f(x) Asadar

$$I = \int_a^b f(x)\,dx \approx I_0 = \int_a^b P(x)\,dx \qquad (5)$$

Luând în considerare rezultatele privind aproximarea functiilor, optam în a alege drept

polinom $P(x)$ polinomul de interpolare Lagrange $L_m(x)$ al functiei $f(x)$. Mai precis, daca $a \le x_0 < x_1 < x_2 < \ldots < x_{m-1} < x_m \le b$ sunt $m+1$ noduri fixate în intervalul $[a, b]$, atunci

$$L_m(x) = \sum_{j=0}^{m} f(x_j)\, Q_j(x) = \sum_{j=0}^{m} y_j\, Q_j(x) \tag{6}$$

unde $y_j = f(x_j) = L_n(y_j)$, $0 \le j \le m$, polinoamele $Q_j(x)$ de grad m având forma

$$Q_j(x) = \frac{(x-x_0)(x-x_1)(x-x_2)\ldots(x-x_{j-1})(x-x_{j+1})\ldots(x-x_{m-1})(x-x_m)}{(x_j-x_0)(x_j-x_1)\ldots(x_j-x_{j-1})(x_j-x_{j+1})\ldots(x_j-x_{m-1})(x_j-x_m)} \tag{7}$$

In aceste conditii aproximarea I_0 a integralei I este data de expresia

$$I_0 = \lambda_0 f(x_0) + \lambda_1 f(x_1) + \lambda_2 f(x_2) + \ldots + \lambda_m f(x_m) \tag{8}$$

unde coeficientii λ_j, $0 \le j \le m$, nu depind de functia $f(x)$,

$$\lambda_j = \int_a^b Q_j(x)\, dx \quad , \quad 0 \le j \le m \tag{9}$$

Aproximarea I_0 depinde numai de valorile y_j, $0 \le j \le m$, ale functiei $f(x)$ în cele $m+1$ puncte $x_0, x_1, x_2, \ldots, x_m$ precizate în intervalul $[a, b]$, unde $y_j = f(x_j)$.

Observatia 1. Formula de aproximare (5) este exacta (adica $I = I_0$) pentru orice polinom $P(x)$ al carui grad nu depaseste m.

In adevar pentru $f(x) = P(x)$, $P(x)$ fiind un polinom de grad k, $k \le n$, din unicitatea polinomului de interpolare $L_m(x)$ ce verifica conditiile $P(x_j) = f(x_j) = L_m(x_j)$, $0 \le j \le m$, rezulta $P(x) \equiv L_m(x)$, $\forall x \in \mathbb{R}$, si deci $I = I_0$ (pentru grad $(P) \le n$ relatia de aproximare (5) devine egalitate).

Observatia 1 ar putea fi folosita la determinarea ponderilor λ_j, $0 \le j \le m$, fara a fi necesara estimarea integralelor (9). Astfel vom deduce valorile celor $m+1$ coeficienti λ_j impunând egalitatea $I = I_0$ pentru orice polinom $P(x)$ de grad $k \le m$, adica

$$\sum_{j=0}^{m} \lambda_j P(x_j) = \int_a^b P(x)\, dx \tag{10}$$

Considerând $P(x) = x^k$, $0 \le k \le m$, se obtine urmatorul sistem liniar de $m+1$ ecuatii

$$\begin{aligned}
&\lambda_0 + \lambda_1 + \lambda_2 + \lambda_3 + \ldots + \lambda_m = b - a \\
&\lambda_0 x_0 + \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 + \ldots + \lambda_m x_m = (b^2 - a^2)/2 \\
&\lambda_0 x_0^2 + \lambda_1 x_1^2 + \lambda_2 x_2^2 + \lambda_3 x_3^2 + \ldots + \lambda_m x_m^2 = (b^3 - a^3)/3 \\
&\lambda_0 x_0^3 + \lambda_1 x_1^3 + \lambda_2 x_2^3 + \lambda_3 x_3^3 + \ldots + \lambda_m x_m^3 = (b^4 - a^4)/4 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\lambda_0 x_0^k + \lambda_1 x_1^k + \lambda_2 x_2^k + \lambda_3 x_3^k + \ldots + \lambda_m x_m^k = (b^{k+1} - a^{k+1})/(k+1) \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\lambda_0 x_0^m + \lambda_1 x_1^m + \lambda_2 x_2^m + \lambda_3 x_3^m + \ldots + \lambda_m x_m^m = (b^{m+1} - a^{m+1})/(m+1)
\end{aligned} \tag{11}$$

Sistemul (11) are întotdeauna solutie deoarece determinantul principal al acestui sistem este nenul fiind de tip Vandermonde, cu $x_i \ne x_j$, $0 \le i \ne j \le m$,

$$\det\begin{pmatrix} 1 & 1 & 1 & 1 \\ x_0 & x_1 & x_2 & x_m \\ x_0^2 & x_1^2 & x_2^2 & x_m^2 \\ \dots & \dots & \dots & \dots \\ x_0^m & x_1^m & x_2^m & x_m^m \end{pmatrix} = \prod_{j>i}(x_j - x_i) \qquad (12)$$

In concluzie :

Propozitia 1. Daca λ_j, $0 \le j \le m$, sunt solutiile sistemului liniar (11) atunci

$$\int_a^b f(x)\,dx \approx \sum_{j=0}^{m} \lambda_j f(x_j) \qquad (13)$$

Exemplul 3. Fie $m = 2$, $x_0 = (3a + b)/4$, $x_1 = (a + b)/2$, $x_0 = (a + 3b)/4$

Relatiile (11) devin

$$\lambda_0 + \lambda_1 + \lambda_2 = b - a \qquad (14)$$

$$[(3a+b)/4]\lambda_0 + [(a+b)/2]\lambda_1 + [(a+3b)/4]\lambda_2 = (b^2 - a^2)/2$$

$$[(3a+b)/4]^2\lambda_0 + [(a+b)/2]^2\lambda_1 + [(a+3b)/4]^2\lambda_2 = (b^3 - a^3)/3$$

Sistemul liniar (14) are solutia

$$\lambda_0 = \lambda_2 = 2(b-a)/3 \qquad \lambda_1 = -(b-a)/3 \qquad (15)$$

si deci

$$\int_a^b f(x)\,dx \approx \frac{b-a}{3}\left[2f(\frac{3a+b}{4}) - f(\frac{a+b}{2}) + 2f(\frac{a+3b}{4})\right] \qquad (16)$$

Exemplul 4 (formula trapezelor). Pentru $m = 2$, $x_0 = a$, $x_1 = b$, din (11) se obtine un sistem liniat de doua ecuatii

$$\lambda_0 + \lambda_1 = b - a \qquad (17)$$

$$a\lambda_0 + b\lambda_1 = (b^2 - a^2)/2$$

a carui solutie este $\lambda_0 = \lambda_1 = (b-a)/2$. Asadar

$$\int_a^b f(x)\,dx \approx \frac{b-a}{2}\left[f(a) + f(b)\right] \qquad (18)$$

integrala I fiind aproximata cu aria trapezului de înaltime b - a având laturile paralele de lungime f(a), respectiv f(b).

Exemplul 5 (formula Simpson). Pentru $m = 3$, $x_0 = a$, $x_1 = (a+b)/2$, $x_2 = b$, relatiile (11) devin

$$\lambda_0 + \lambda_1 + \lambda_2 = b - a \qquad (19)$$

$$a\lambda_0 + [(a+b)/2]\lambda_1 + b\lambda_3 = (b^2 - a^2)/2$$

$$a^2\lambda_0 + [(a+b)/2]^2\lambda_1 + b^2\lambda_3 = (b^3 - a^3)/3$$

Sistemul (19) are solutia $\lambda_0 = \lambda_2 = (b-a)/6$, $\lambda_1 = 2(b-a)/3$.

Aplicând Propozitia 1 obtinem aproximatia Simpson a integralei I, adica

$$\int_a^b f(x)\,dx \approx \frac{b-a}{6}\left[f(a) + 4f(\frac{a+b}{2}) + f(b) \right] \tag{20}$$

2.2. Estimarea erorii de aproximare

Orice formula de aproximare este practic inoperabila în cazul în care nu se poate aprecia marimea erorii rezultate prin aplicarea ei. In acest sens vom încerca sa evaluam expresia restului în cazul folosirii metodei trapezului sau a aproximarii Simpson.

Adesea vom utiliza urmatoarea formula de medie dedusa din proprietatea lui Darboux pentru functii continue.

Propozitia 2. (Prima formula de medie). Daca $g_1(x)$ si $g_2(x)$ sunt doua functii continue pe intervalul [a , b] si în plus functia $g_2(x)$ pastreaza acelasi semn pe [a , b] , atunci exista $c \in [a , b]$ astfel încât

$$\int_a^b g_1(t)g_2(t)\,dt = g_1(c)\int_a^b g_2(t)\,dt \tag{21}$$

Observatia 2. In cazul în care functia $g_2(x)$ nu pastreaza semn constant pe intervalul [a , b] s-ar putea ca formula de medie (21) sa nu fie adevarata. Astfel pentru $a = -1$, $b = 1$, $g_1(x) = g_2(x) = x$ relatia (21) nu este satisfacuta, aceasta deoarece $2/3 \neq 0$.

Tinând seama de faptul ca functiile continue au proprietatea lui Darboux rezulta :

Propozitia 3. (A doua formula de medie). Daca g(x) este o functie continua pe intervalul [a , b] iar $c_1 , c_2 , c_3 , \dots , c_n$ sunt n puncte arbitrare în acest interval atunci exista un punct $c \in [a , b]$ astfel încât

$$g(c_1) + g(c_2) + g(c_3) = \dots + g(c_{n-1}) + g(c_n) = n\,g(c) \tag{22}$$

Propozitia 3 nu impune ca punctele c_j , $1 \le j \le n$, sa fie neaparat distincte.

2.3. Metoda trapezului

2.3.1. Restul în formula trapezului

Cu notatia $h = b - a$, restul $R_2(h)$ al aproximarii date prin aplicarea formulei (18), a trapezului , este dat de expresia

$$R_2(h) = \int_a^b f(x)\,dx - \frac{h}{2}\left[f(a) + f(b) \right] \tag{23}$$

Vom evalua derivatele $R_2^{(1)}(h)$, $R_2^{(2)}(h)$ de primul si al doilea ordin ale functiei $R_2(h)$. Prin calcul direct se obtine

$$\begin{aligned} R_2^{(1)}(h) &= f(a+h) - [\,f(a) + f(a+h)\,]/2 - (h/2)\,f^{(1)}(a+h) = \\ &= [\,f(a+h) - f(a)\,]/2 - (h/2)\,f^{(1)}(a+h) \end{aligned} \tag{24}$$

$$R_2^{(2)}(h) = (1/2)\,f^{(1)}(a+h) - (1/2)\,f^{(1)}(a+h) - (h/2)\,f^{(2)}(a+h) = -(h/2)\,f^{(2)}(a+h)$$

In mod evident $R_2(0) = R_2^{(1)}(0) = R_2^{(2)}(0) = 0$ si deci

$$R_2^{(1)}(h) = R_2^{(1)}(0) + \int_0^h R_2^{(2)}(t)\,dt = \int_0^h -(t/2) f^{(2)}(a+t)\,dt =$$

$$= -\left[\left(f^{(2)}(c_0(h)\right)/2\right]\int_0^h t\,dt = -\left(h^2/4\right) f^{(2)}(c_0(h)) \qquad (25)$$

unde valoarea $c_0(h)$, $c_0(h) \in [\,a\,,\,a+h\,]$ este data de formula de medie (21). Asadar

$$R_2(h) = R_2(0) + \int_0^h R_2^{(1)}(t)\,dt = \int_0^h -\left(t^2/4\right) f^{(2)}(c_0(t))\,dt =$$

$$= -\left(f^{(2)}(c)\right)\int_0^h t^2\,dt = -\left(h^3/12\right) f^{(2)}(c) \qquad (26)$$

valoarea c, $c \in [\,a\,,\,a+h\,]$, rezultând în urma aplicarii formulei de medie (21) (Propozitia 2).

Asadar prin aplicarea metodei trapezului avem egalitatea $I = I_0 + R_2(b\text{-}a)$ si deci

Propozitia 4. Exista $c \in [\,a\,,\,b\,]$ ce verifica egalitatea

$$\int_a^b f(x)\,dx = \frac{b-a}{2}\left[f(a) + f(b)\right] - \left((b-a)^3/12\right) f^{(2)}(c) \qquad (27)$$

Vom nota prin M_2 maximul functiei $|f^{(2)}(x)|$ pe intervalul $[\,a\,,\,b\,]$. Prin urmare

$$|f^{(2)}(x)| \le M_2 \qquad \forall\, x \in [\,a\,,\,b\,] \qquad (28)$$

In aceste conditii din formula (26) rezulta $|R_2(b\text{-}a)| \le (b-a)^3 M_2/12$ adica

Corolar 1. Aplicarea metodei trapezului conduce la urmatoarea estimare a restului aproximarii integralei I (formula (1)) prin expresia $I_0 = (b-a)[f(a) + f(b)]/2$,

$$\left|\int_a^b f(x)\,dx - \frac{b-a}{2}\left[f(a) + f(b)\right]\right| \le \frac{(b-a)^3 M_2}{12} \qquad (29)$$

Intentionam sa extindem metoda trapezului împartind intervalul $[\,a\,,\,b\,]$ în n subintervale. Fiecarui astfel subinterval i se va aplica în mod independent procedura trapezului de aproximare a integralei I ce este restrictionata la subintervalul selectat .

2.3.2. Formula generala de aproximare în metoda trapezului

Fie $a = t_0 < t_1 < t_2 < \ldots < t_{n-1} < t_n = b$ o diviziune a intervalului $[\,a\,,\,b\,]$ cu noduri echidistante, adica $t_j = a + j\,(b-a)/n$, $0 \le j \le n$. Aplicând formula (18) în cazul fiecarui subinterval $[\,t_{j-1}\,,\,t_j\,]$, $1 \le j \le n$, cu notatia $h = (b-a)/n$ vom obtine urmatoarea aproximare I_0 a integralei I

$$I = \int_a^b f(t)\,dt = \sum_{j=1}^{n} \int_{t_{j-1}}^{t_j} f(t)\,dt \approx \sum_{j=1}^{n} \frac{h}{2}\left(f(t_{j-1}) + f(t_j)\right) = I_0 \qquad (30)$$

Conform relatiei (26) , restul R_{2*} al aproximarii I_0 definita de formula (30) are expresia

$$R_{2*} = (- h^3 / 12) [f^{(2)}(c_1) + f^{(2)}(c_2) + f^{(2)}(c_3) + \ldots + f^{(2)}(c_n)] \quad (31)$$

Aplicând cea de a doua formula de medie (Propozitia 3) se obtine

$R_{2*} = (- h^3 / 12) n f^{(2)}(c)$ adica

Propozitia 5. Exista $c \in [a , b]$ ce verifica egalitatea

$$\int_a^b f(x)\,dx = \frac{b-a}{n}\left[\frac{f(t_0)}{2} + f(t_1) + f(t_2) + \ldots + f(t_{n-1}) + \frac{f(t_n)}{2}\right] - \frac{(b-a)^3}{12n^2} f^{(2)}(c) \quad (32)$$

Corolar 2. Daca $| f^{(2)}(x) | \le M_2$, $\forall x \in [a , b]$ atunci restul R_{2*} al aproximarii (30) satisface inegalitatea

$$R_{2*} \le (b - a)^3 M_2 / (12 n^2) \quad (33)$$

Exemplul 6. In practica se impune gasirea unei aproximatii I_0 a integralei I astfel încât evaluarea (30) sa se realizeze cu o eroare care sa nu depaseasca un prag ε dat, $\varepsilon > 0$.

In aceasta situatie, din conditia

$$(b - a)^3 M_2 / (12 n^2) \le \varepsilon$$

deducem inegalitatea

$$n \ge [(b - a)^3 M_2 / (12 \varepsilon)]^{1/2} \quad (34)$$

ce exprima numarul minim n de puncte de diviziune ce vor trebui folosite pentru a fi siguri ca eroarea R_{2*} rezultata prin aplicarea metodei trapezului este sub valoarea ε

2.3.3. Utilizarea practica a metodei trapezelor

Aplicatia 2. Vom determina experimental o estimatie I_0 a integralei I pentru functia $f_k(x) = x^k$, $a \le x \le b$, $a = 0$, $b = 2$, $2 \le k \le 10$, acceptând o eroare $\varepsilon = 0.001$

Valoarea exacta a integralei I data de formula (1) este $I = 2^{k+1} / (k + 1)$ Aproximatia I_0 a integralei I se va face cu o precizie ε , în aceasta situatie luând în considerare n+1 puncte de diviziune $a = t_0 , t_1 , t_2 , \ldots , t_{n-1} , t_n = b$ unde n satisface inegalitatea (34) iar $t_j - t_{j-1} = (b - a) / n$, $1 \le j \le n$, si

$$I_0 = [(b - a) / n] \{ [f(t_0) + f(t_n)] / 2 + f(t_1) + f(t_2) + \ldots + f(t_{n-2}) + f(t_{n-1}) \}$$

Un majorant al functiei $f_k^{(2)}$ pe intervalul [a , b] , $a \ge 0$, este dat de expresia $M_2 = k (k - 1) b^{k-2}$ pentru $k \ge 2$. Pentru $k = 1$ se obtine $M_2 = 0$ (formula de aproximare (32) este în acest caz exacta pentru orice n , adica $R_{2*} = 0$).

In programul BASIC urmator functia **int2** estimeaza integrala I prin metoda trapezului.

```
DECLARE FUNCTION f! (x!, k!) :          DECLARE FUNCTION int2! (a!, b!, n!, k!)
REM  Calculul integralelor prin metoda trapezelor
eps = .001          ' eps = eroarea de aproximare admisa
a = 0 :          b = 2
FOR k = 2 TO 10 :          PRINT "Cazul  k = "; k
REM  Calculul numarului  n  de intervale pentru a se asigura precizia  eps
```

```
m2 = k * (k - 1) * (b ^ (k - 2))
n = FIX(SQR(m2 * ((b - a) ^ 3) / (12! * eps))) + 1! :     PRINT "Numarul de intervale = "; n
ie = (b ^ (k + 1) - a ^ (k + 1)) / (k + 1)
PRINT "Valoarea exacta a integralei pe intervalul ["; a; " , "; b; " ] este "; ie
ia = int2(a, b, n, k) :          PRINT "Valoarea aproximativa a integralei = "; ia
er = ie - ia :          PRINT "Eroare = "; er :          NEXT k

FUNCTION f (x, k) :     f = x ^ k :     END FUNCTION

FUNCTION int2 (a, b, n, k)
s# = (f(a, k) + f(b, k)) / 2! :          h = (b - a) / n
FOR j = 1 TO n - 1 :          s# = s# + f(a + j * h, k) :          NEXT j
int2 = h * s# :          END FUNCTION
```

Exemplul 7. Rezultatele rulării programului proiectat în Aplicatia 2 pentru functiile $f(x) = x^k$, $0 \le x \le 2$, sunt sintetizate în Tabelul 1. Acuratetea aproximarii I_0 a integralei I este dictata de diferenta $I - I_0$. Inegalitatea (34) precizeaza numarul minim n de subintervale ce trebuiesc folosite pentru atingerea unei erori $\varepsilon = 0.001$.

Rezultatele experimentale din Tabelul 1 confirma obtinerea preciziei ε dorite, verificându-se inegalitatea $|I - I_0| < \varepsilon$.

Tabelul 1. Calculul prin metoda trapezului a aproximatiei I_0 a integralei I utilizând n subintervale (eroarea admisa $\varepsilon = 0.001$, $a = 0$, $b = 2$, $f(x) = x^k$)

$f(x) = x^k$	I	I_0	n	$I - I_0$
$f(x) = x^2$	2.666667	2.667641	37	-0.0009739
$f(x) = x^3$	4.000000	4.000494	90	-0.0004940
$f(x) = x^4$	6.400000	6.400332	179	-0.0003319
$f(x) = x^5$	10.666667	10.666918	327	-0.0002508
$f(x) = x^6$	18.285715	18.285915	566	0.0002003
$f(x) = x^7$	32.000000	32.000175	947	-0.0001755
$f(x) = x^8$	56.888889	56.889038	1546	-0.0001488
$f(x) = x^9$	102.400002	102.400124	2479	-0.0001221
$f(x) = x^{10}$	186.181824	186.181839	3920	-0.0000153

2.4. Metoda Simpson

2.4.1. Restul în formula Simpson

Considerând $h = (b - a) / 2$, $d = (a + b) / 2$, restul $R_3(h)$ al aproximarii Simpson (a se vedea formula (20)) este dat de expresia

$$R_3(h) = \int_{d-h}^{d+h} f(x)\,dx - \frac{h}{3}\left[f(d-h) + 4f(d) + f(d+h) \right] \tag{35}$$

Ca si în cazul metodei trapezului, dupa un calcul elementar deducem valorile derivatelor $R_3^{(1)}(h)$, $R_3^{(2)}(h)$, $R_3^{(3)}(h)$ de primele trei ordine ale functiei $R_2(h)$, anume

$$\begin{aligned} R_3^{(1)}(h) &= [\,2f(d+h) + 2f(d-h) - 4f(d)\,]/3 - h\,[\,f^{(1)}(d+h) - f^{(1)}(d-h)\,]/3 \\ R_3^{(2)}(h) &= [\,f^{(1)}(d+h) - f^{(1)}(d-h)\,]/3 - h\,[\,f^{(2)}(d+h) - f^{(2)}(d-h)\,]/3 \\ R_3^{(3)}(h) &= -h\,[\,f^{(3)}(d+h) - f^{(3)}(d-h)\,]/3 \end{aligned} \tag{36}$$

Aplicând în ultima egalitate teorema lui Lagrange rezulta un punct $c_3(h) \in (\,d-h\,,\,d+h\,)$ astfel încât sa se verifice relatia

$$R_3^{(2)}(h) = -(\,2h^2/3\,)\,f^{(4)}(c_3(h)) \tag{37}$$

Din formulele (35) si (36) deducem $R_3(0) = R_3^{(1)}(0) = R_3^{(2)}(0) = R_3^{(3)}(0)$ si deci

$$\begin{aligned} R_3^{(2)}(h) &= R_3^{(2)}(0) + \int_0^h R_3^{(3)}(t)\,dt = -\int_0^h \frac{2t^2}{3} f^{(4)}(c_3(t))\,dt = \\ &= -\frac{2f^{(4)}(c_2(h))}{3}\int_0^h t^2\,dt = -\frac{2h^3 f^{(4)}(c_2(h))}{9} \end{aligned} \tag{38}$$

valoarea $c_2(h) \in [\,d-h\,,\,d+h\,]$ fiind data de prima formula de medie (21). In mod analog

$$\begin{aligned} R_3^{(1)}(h) &= R_3^{(1)}(0) + \int_0^h R_3^{(2)}\,dt = -\int_0^h \frac{2t^3 f^{(4)}(c_2(t))}{9}\,dt = \\ &= -\frac{2f^{(4)}(c_1(h))}{9}\int_0^h t^3\,dt = -\frac{h^4 f^{(4)}(c_1(h))}{18} \end{aligned} \tag{39}$$

punctul $c_1(h) \in [\,d-h\,,\,d+h\,]$ rezultând din Propozitia 2. In urma unui calcul similar se obtine

$$\begin{aligned} R_3(h) &= R_3(0) + \int_0^h R_3^{(1)}(t)\,dt = -\int_0^h \frac{t^4 f^{(4)}(c_1(t))}{18}\,dt = \\ &= -\frac{f^{(4)}(c_0(h))}{18}\int_0^h t^4\,dt = -\frac{h^5 f^{(4)}(c_0(h))}{90} \end{aligned} \tag{40}$$

valoarea $c_0(h)$, $c_0(h) \in [\,d-h\,,\,d+h\,]$, fiind dedusa prin utilizarea formulei de medie (21).

Pentru $h = (b - a) / 2$, prin aplicarea metodei Simpson (relatia (20)) rezulta egalitatea $I = I_0 + R_3(b-a)$. Tinând seama de expresia (40) a restului de aproximare $R_3(h)$ deducem în final

Propozitia 6. Exista $c \in [a , b]$ ce verifica egalitatea

$$\int_a^b f(x)\,dx = \frac{b-a}{6}\left[f(a) + 4f(\frac{a+b}{2}) + f(b) \right] - \frac{(b-a)^5 f^{(4)}(c)}{2880} \qquad (41)$$

Vom nota prin M_4 maximul functiei $| f^{(4)}(x) |$ pe intervalul $[a , b]$. Prin urmare

$$| f^{(4)}(x) | \le M_4 \qquad \forall x \in [a , b] \qquad (42)$$

In aceste conditii din formula (40) deducem $| R_3 (b-a) | \le (b - a)^5 M_4 / 2880$ adica

Corolar 3. Procedura Simpson conduce la urmatoarea estimare a restului aproximarii integralei I (formula (1)) prin expresia $I_0 = (b - a) [f(a) + 4 f((a+b)/2) + f(b)] / 6$,

$$\left| \int_a^b f(x)\,dx - \frac{b-a}{6}\left[f(a) + 4f(\frac{a+b}{2}) + f(b) \right] \right| \le \frac{(b-a)^5 M_4}{2880} \qquad (43)$$

Vom extinde metoda lui Simpson divizând intervalul $[a , b]$ în $2n$ subintervale si aplicând procedura Simpson pentru fiecare interval astfel rezultat.

2.4.2. Formula generala de aproximare în metoda Simpson

Fie $a = t_0 < t_1 < t_2 < \ldots < t_{2n-1} < t_{2n} = b$ o diviziune a intervalului $[a , b]$ cu noduri echidistante, adica $t_j = a + j (b - a) / (2n)$, $0 \le j \le 2n$. Utilizând formula (41) în cazul fiecarui subinterval $[t_{2j-2} , t_{2j}]$, $1 \le j \le n$, obtinem urmatoarea aproximare I_0 a integralei I :

$$I = \int_a^b f(t)\,dt = \sum_{j=1}^{n} \int_{t_{2j-2}}^{t_{2j}} f(t)\,dt \approx \sum_{j=1}^{n} \frac{b-a}{6n}\left(f(t_{2j-2}) + 4f(t_{2j-1}) + f(t_{2j}) \right) = I_0 \qquad (44)$$

Conform relatiei (40) , restul R_{3*} al aproximarii I_0 definite de formula (44) are expresia

$$R_{3*} = (- h^5 / 90) [f^{(4)}(c_1) + f^{(4)}(c_2) + f^{(4)}(c_3) + \ldots + f^{(4)}(c_n)] \qquad (45)$$

Folosind cea de a doua formula de medie (Propozitia 3) se obtine

$R_{3*} = (- h^5 / 90) n f^{(4)}(c)$ adica

Propozitia 7. Exista $c \in [a , b]$ ce verifica egalitatea

$$\int_a^b f(x)\,dx = \frac{b-a}{6n}\left[f(t_0) + f(t_n) + 2\sum_{j=1}^{n-1} f(t_{2j}) + 4\sum_{j=1}^{n} f(t_{2j-1}) \right] - \frac{(b-a)^5}{2880 n^4} f^{(4)}(c) \qquad (46)$$

Corolar 4. Daca $| f^{(4)}(x) | \le M_4$, $\forall x \in [a , b]$ atunci restul R_{3*} rezultat în urma aproximarii (44) satisface inegalitatea

$$R_{3*} \le (b - a)^5 M_4 / (2880 n^4) \qquad (47)$$

Exemplul 8. Vom deduce aproximarea I_0 data de expresia (44) astfel încât eroarea facuta sa nu depaseasca un prag ε dat, $\varepsilon > 0$. In acest caz, din conditia

$$(b - a)^5 M_4 / (2880 n^4) \le \varepsilon$$

obtinem inegalitatea

$$n \ge [(b - a)^5 M_4 / (2880 \varepsilon)]^{1/4} \qquad (48)$$

ce exprima numarul minim n de puncte de diviziune ce vor trebui folosite pentru a fi siguri ca eroarea R_{3*} rezultata prin aplicarea metodei Simpson este sub pragul ε .

2.4.3. Utilizarea practica a metodei Simpson

Aplicatia 3. Vom studia cazul $a = 0$, $b = 2$, $f_k(x) = x^k$, $2 \le k \le 10$, cu $\varepsilon = 0.001$. Valoarea exacta a integralei I data de formula (1) este $2^{k+1} / (k + 1)$. Vom impune ca aproximatia I_0 a integralei I se se obtina cu o precizie ε . In aceasta situatie vom lua în considerare 2n+1 puncte de diviziune $a = t_0 , t_1 , t_2 , \dots , t_{2n-1} , t_{2n} = b$ unde n satisface inegalitatea (48) iar $t_j - t_{j-1} = (b - a) / (2 n)$, $1 \le j \le 2n$, cu

$$I_0 = \frac{b-a}{6n}\left[f(t_0) + f(t_n) + 2\sum_{j=1}^{n-1} f(t_{2j}) + 4\sum_{j=1}^{n} f(t_{2j-1}) \right] \qquad (49)$$

Un majorant al functiei $f_k^{(4)}$ pe intervalul [a , b] este $M_4 = k (k - 1) (k - 2) (k - 3) b^{k-4}$ pentru $k \ge 4$. Pentru $k < 4$ obtinem $M_4 = 0$ (adica formula de aproximare (44) este exacta).

In programul BASIC urmator functia **int3** estimeaza integrala I prin metoda Simpson

```
DECLARE FUNCTION f! (x!, k!) :              DECLARE FUNCTION int3! (a!, b!, n!, k!)
REM  Calculul integralelor prin metoda Simpson
eps = .001              ' eps = eroarea de aproximare admisa
a = 0 :             b = 2
FOR k = 2 TO 10 :          PRINT "Cazul  k = "; k
REM  Calculul numarului  n  de intervale pentru a se asigura precizia  eps
m4 = k * (k - 1) * (k - 2) * (k - 3) * (b ^ (k - 4))
n = FIX((m4 * ((b - a) ^ 5) / (2880! * eps)) ^ .25) + 1! :     PRINT "Numarul de intervale = "; n
ie = (b ^ (k + 1) - a ^ (k + 1)) / (k + 1)
PRINT "Valoarea exacta a integralei pe intervalul  ["; a; " , "; b; " ]  este "; ie
ia = int3(a, b, n, k) :          PRINT "Valoarea aproximativa a integralei = "; ia
er = ie - ia :          PRINT "Eroare = "; er :          NEXT k

FUNCTION f (x, k) :    f = x ^ k :    END FUNCTION

FUNCTION int3 (a, b, n, k)
s# = f(a, k) + f(b, k) :          h = (b - a) / (2 * n) :          s1# = 0 :    s2# = 0
FOR j = 1 TO n :          s1# = s1# + f(a + (2 * j - 1) * h, k) :          NEXT j
FOR j = 1 TO n - 1 :          s2# = s2# + f(a + 2 * j * h, k) :          NEXT j
int3 = h * (s# + 4 * s1# + 2 * s2#) / 3 :          END FUNCTION
```

Exemplul 9. Rezultatele rularii programului proiectat în Aplicatia 3 pentru $f(x) = x^k$, $0 \le x \le 2$, sunt sintetizate în Tabelul 2. Folosind inegalitatea (48) se determina numarul minim n de subintervale pentru care se va aplica metoda Simpson astfel încât eroarea de aproximare sa nu depaseasca pragul $\varepsilon = 0.001$ fixat. Acest aspect este confirmat de rezultatele experimentale din Tabelul 2 deoarece $| I - I_0 | \le \varepsilon = 0.001$.

Observatia 3. Impunând o eroare ε la obtinerea unei aproximatii I_0 a integralei I, din Tabelele 1 si 2 se constata ca numarul n al punctelor de diviziune este cu mult mai mic în cazul metodei Simpson comparativ cu metoda trapezelor. Asadar prin aplicarea procedurii Simpson se poate realiza aceeasi precizie ε a aproximatiei I_0 ca si în cazul metodei trapezului dar cu un volum de calcule sensibil mai redus.

Tabelul 2. Calculul aproximatiei I_0 a integralei I utilizând metoda Simpson cu $2n$ subintervale (eroarea admisa $\varepsilon = 0.001$, $a = 0$, $b = 2$, $f(x) = x^k$)

$f(x) = x^k$	I	I_0	n	$I - I_0$
$f(x) = x^2$	2.666667	2.666667	1	0.0000000
$f(x) = x^3$	4.000000	4.000000	1	0.0000000
$f(x) = x^4$	6.400000	6.400427	5	-0.0004272
$f(x) = x^5$	10.666667	10.666992	8	-0.0003252
$f(x) = x^6$	18.285715	18.285975	12	-0.0002594
$f(x) = x^7$	32.000000	32.000225	17	-0.0002251
$f(x) = x^8$	56.888889	56.889080	24	-0.0001907
$f(x) = x^9$	102.400002	102.400177	33	-0.0001755
$f(x) = x^{10}$	186.181824	186.182007	44	-0.0001831

2.5. O metoda probabilista

In paragrafele anterioare obtinerea aproximarii I_0 a integralei I s-a realizat prin aplicarea unor metode deterministe, estimându-se totodata si un interval de variatie al erorii de aproximare respective.

Din punct de vedere teoretic integrala I este limita sumelor Riemann corespunzatoare, adica

$$I = \lim_{n \to \infty} \sum_{j=1}^{n} f(u_j)\,(x_j - x_{j-1}) \tag{50}$$

unde $a = x_0 < x_1 < x_2 , \ldots < x_{n-1} < x_n = b$ este o diviziune fixata a intervalului $[a , b]$ cu $n+1$ puncte de diviziune iar u_j sunt puncte arbitrare, $u_j \in [x_{j-1} , x_j]$, $1 \le j \le n$.

Asadar putem considera aproximarea I_0 data de expresia

$$I_0 = \sum_{j=1}^{n} f(u_j)\,(x_j - x_{j-1}) \tag{51}$$

unde numarul de subintervale n ale domeniului $[a , b]$ este relativ mare.

In aplicatii putem opera cu o partitie de subintervale $[x_{k-1} , x_k]$, $1 \le k \le n$, ce au aceeasi lungime $h = (b - a) / n$, $x_j = a + j\,h$, $0 \le j \le n$, considerând în formula (51) $u_j \equiv x_j$, $0 \le j \le n$. In aceasta situatie aproximatia I_0 a integralei I este data de expresia

$$I_0 = [(b - a) / n] [f(x_1) + f(x_2) + f(x_3) + \quad + f(x_{n-1}) + f(x_n)] \qquad (52)$$

Expresia (52) este asemanatoare cu cea specificata de metoda trapezului (a se compara cu formulele (30) si (32)).

Aplicatia 4. Programul urmator, bazat pe formula (52), estimeaza valoarea integralei I pentru functia $f(x) = x^3$

```
DECLARE FUNCTION f! (x!)
REM  Calculul integrale printr-o suma Riemann
INPUT "n, a, b = "; n, a, b :          s = 0 :          h = (b - a) / n
FOR j = 1 TO n :          u = a + j * h :          s = s + f(u) :          NEXT j
s = h * s :          PRINT "Aproximatia Riemann a integralei : "; s

FUNCTION f (x) :    f = x ^ 3 :    END FUNCTION
```

Exemplul 10. Valoarea exacta a integralei I pentru $f(x) = x^3$, cu $a = 0$, $b = 2$ este $I = (b^4 - a^4) / 4 = 2^4 / 4 = 4$. Rulând programul BASIC descris în Aplicatia 4 pentru $n = 100$ se obtine valoarea aproximatiei I_0 a integralei I, anume $I_0 = 4.0804$.

Observatia 4. In cazul în care punctele u_j se aleg în mod determinist am putea fi expusi unor eventuale erori sistematice de estimare. Astfel în cazul precedent, daca functiile f(x) sunt crescatoare (descrescatoare) atunci optând pentru $u_j = x_j$ rezulta întotdeauna o supraevaluare (respectiv subevaluare) I_0 a integralei I.

Exemplul 11. Considerând functia $f(x) = x^3$, $0 \le x \le 2$, vom executa programul propus în Aplicatia 4 pentru un numar variabil n de puncte de diviziune. Valorile sumelor Riemann I_0 astfel rezultate (formula (52)), listate în Tabelul 3, aproximeaza integrala I din functia f(x), $0 \le x \le 2$.

Tabelul 3. Sumele Riemann I_0 date de formula (52) pentru $f(x) = x^3$, $0 \le x \le 2$.

n	I_0	n	I_0	n	I_0	n	I_0	n	I_0
50	4.16160	100	4.08040	150	4.05351	200	4.04010	250	4.03206
300	4.02671	350	4.02289	400	4.02002	450	4.01780	500	4.01602

Mentionam faptul ca toate evaluarile I_0 astfel rezultate sunt din ce în ce mai precise dar depasesc sistematic valoarea exacta $I = 4$ a integralei functiei $f(x) = x^3$ pe intervalul [0 , 2] (Observatia 4 este confirmata, functia f(x) fiind crescatoare).

Aplicatia 5. Vom sugera o procedura Monte Carlo bruta în vederea determinarii aproximatiei I_0 a integralei I. Metoda se bazeaza pe formula (51) unde cantitatile u_j sunt valori aleatoare uniform repartizate respectiv în intervalele $[x_{j-1} , x_j]$, $1 \le j \le n$. Obtinerea valorii întâmplatoare $u_j \in [x_{j-1} , x_j]$ se realizeaza practic utilizând formula $u_j = x_{j-1} + (x_j - x_{j-1}) V$ unde V este un numar pseudoaleator uniform repartizat în intervalul [0 , 1] ce este produs cu ajutorul calculatorului.

Generarea valorilor aleatoare V, $V \in [0 , 1]$, se realizeaza în limbajul BASIC prin apelarea functiei RND.

In program s-a considerat $x_k = a + k\,h$, $0 \le k \le n$, unde $h = (b - a) / n$

```
REM  Calculul unei integrale prin metoda Monte Carlo
DECLARE FUNCTION f! (x!) :                RANDOMIZE TIMER
INPUT "n, a, b = "; n, a, b :        PRINT "Estimatii ale integralei : "        h = (b - a) / n
FOR k = 1 TO 24 :          s = 0
FOR j = 1 TO n :        u = a + (j - 1 + RND) * h :        s = s + f(u) .        NEXT j
s = (b - a) * s / n :        PRINT s, " , ";:        NEXT k

FUNCTION f (x) :    f = x ^ 3 :    END FUNCTION
```

Exemplul 12. Prin rularea programului prezentat în Aplicatia 5 , pentru $n = 100$, $a = 0$, $b = 2$, $f(x) = x^3$, s-au obtinut 24 de estimatii distincte I_0 ale integralei I (Tabelul 4).

Tabelul 4. Evaluari I_0 ale integralei I obtinute prin aplicarea metodei Monte Carlo bruta (formula (51) , $f(x) = x^3$, $0 \le x \le 2$, $n = 100$).

4.000461	4.006549	3.994599	4.004309	4.007972	3.998848	4.016898	3.989934
4.001083	3.990132	3.997615	3.997539	4.000218	4.002399	3.989283	3.992018
3.988111	4.008351	4.002393	4.006297	3.999034	3.987329	3.993988	3.995304

Observatia 5. Toate estimatiile I_0 din Tabelul 4 sunt foarte apropiate de valoarea $I = 4$ Analizând aproximatiile I_0 rezultate prin aplicarea tehnicii Monte Carlo evidentiem atât valori mai mari cât si mai mici de valoarea exacta I fapt ce infirma producerea unor erori sistematice în evaluarea integralei I .

Observatia 6. Este normal, dupa cum s-a procedat si în paragrafele anterioare, sa ne punem problema estimarii marimii erorii de aproximare $| I - I_0 |$ corespunzatoare metodei probabiliste mentionate.

Trebuie remarcat faptul ca prin aplicarea acestei tehnici de tip Monte Carlo, aproximarea I_0 reprezinta una dintre multiplele realizari posibile ale unei variabile aleatoare W , anume

$$W = [(b - a) / n] [f(U_1) + f(U_2) + f(U_3) + \ldots + f(U_{n-1}) + f(U_n)] \qquad (53)$$

unde $U_1 , U_2 , \ldots , U_n$ sunt variabile aleatoare independente uniform repartizate în intervalele $[x_{j-1} , x_j]$, $1 \le j \le n$.

Teoretic se poate demonstra ca media variabilei aleatoare W data de formula (53) este chiar I In acest context eroarea de aproximare rezultata prin aplicarea unei proceduri de tip Monte Carlo este privita din punct de vedere probabilist, fiind interpretata drept dispersie a variabilei aleatoare W

Remarcam si posibilitatea definirii variabilei aleatoare W printr-o relatie asemanatoare formulei (53) dar în care cele n variabile aleatoare independente $U_1 , U_2 , \ldots , U_n$ au o repartitie oarecare pe intervalul de integrare, nu neaparat o repartitie uniforma pe $[a , b]$. In aceasta situatie se impune efectuarea unui studiu comparativ al erorilor rezultate prin aplicarea diverselor metode probabiliste (a se vedea Ermakov, 1976)

REZOLVAREA ECUATIILOR DIFERENTIALE ORDINARE

In cele ce urmeaza vom studia doua metode pentru aflarea solutiei ecuatiei diferentiale ordinare

$$\partial v(x) / \partial x = f(x, v(x)) \qquad (1)$$

fapt ce revine la a preciza functia $v(x)$ ce verifica relatia (1).

Presupunem ca functia $f(x,y)$ este derivabila de un numar suficient de mare de ori astfel încât calculele ce vor fi prezentate în continuare sa se poata desfasura corect. Putem astfel accepta existenta derivatelor partiale ale functiei $f(x,y)$ pâna la ordinul 4, pentru $a_1 \le x \le b_1$, $a_2 \le y \le b_2$.

Vom impune ca solutia $v(x)$ a ecuatiei diferentiale (1), $a_1 \le x \le b_1$, sa verifice în plus conditia initiala

$$v(a_1) = v_0 \qquad (2)$$

In locul precizarii solutiei reale $v(x)$ vom determina aproximatii w_j ale sale în puncte $x_j \in [a_1, b_1]$ fixate, adica $w_j \approx v(x_j)$, $0 \le j \le m$. In aceasta situatie eroarea punctuala ε_j, $0 \le j \le m$, este diferenta dintre valoarea reala $v(x_j)$ si aproximatia w_j ce a fost utilizata,

$$\varepsilon_j = v(x_j) - w_j \qquad (3)$$

O masura globala $|\varepsilon|$ a erorilor punctuale ε_j, $0 \le j \le m$, este data de relatia

$$|\varepsilon| = \max \{ |\varepsilon_j| , \ 0 \le j \le m \} \qquad (4)$$

1. Metoda Euler

Conform teoremei lui Lagrange avem

$$v(x_{j+1}) = v(x_j) + (\partial v(\alpha) / \partial x) . [x_{j+1} - x_j]$$

unde $\alpha \in (x_j, x_{j+1})$. Vom accepta aproximatia

$$v(x_{j+1}) \approx v(x_j) + (\partial v(x_j) / \partial x) . [x_{j+1} - x_j]$$

Cum $v(x)$ este solutia ecuatiei diferentiale (1), din relatia anterioara rezulta

$$v(x_{j+1}) \approx v(x_j) + f(x_j, v(x_j)) . [x_{j+1} - x_j]$$

Constructia sirului de aproximatii $\{ w_j \}_j$, $0 \le j \le m$, pentru valorile discrete $v(x_j)$ se realizeaza iterativ prin utilizarea repetata a formulei

$$w_{j+1} = w_j + f(x_j, w_j) . [x_{j+1} - x_j] , \qquad 0 \le j \le m$$

In cele ce urmeaza vom alege nodurile x_j, $0 \le j \le m$, echidistante. Expresia $f(x_j, v(x_j))$ va fi aproximata de valoarea $f(x_j, w_j)$ cu o eroare δ_j, $\forall\, 0 \le j \le m$. Prin urmare

$$w_{j+1} = w_j + h f(x_j, w_j) + \delta_j , \qquad 0 \le j < m \qquad (5)$$

unde

$$w_0 = v_0 , \qquad h = (b_1 - a_1) / m , \qquad x_j = a_1 + j h , \qquad 0 \le j \le m \qquad (6)$$

Folosind formula (4) vom estima eroarea $|\varepsilon|$ rezultata prin aproximarea solutiei $v(x)$ a ecuatiei (1) cu sirul de valori $\{ w_j \}_j$, $0 \le j \le m$.

In acest context acceptam ipoteza

$$|\delta_j| \le c_f h^2 , \qquad 0 \le j \le m \qquad (7)$$

unde c_f este o constanta ce depinde de forma functiei f.

Pentru simplificarea exprimarii vom utiliza notatiile

$$c_0 = \sup \{ | f(x, y) | , \forall (x, y) \in D_f \} , \quad c_1 = \sup \{ | \partial f(x, y) / \partial x | , \forall (x, y) \in D_f \}$$

$$c_2 = \sup \{ | \partial f(x, y) / \partial y | , \forall (x, y) \in D_f \} , \quad D_f \equiv [a_1 , b_1] \times [a_2 , b_2] \tag{8}$$

Derivând relatia (1) în raport cu variabila x obtinem

$$\partial^2 v(x) / \partial x^2 = f_x (x , v(x)) + f(x , v(x)) \cdot f_y (x , v(x)) \tag{9}$$

unde $f_x (x , y) = \partial f(x , y) / \partial x$, $f_y (x , y) = \partial f(x , y) / \partial y$.

Pentru orice $t \in [a_1 , b_1]$, din (8) si (9) rezulta egalitatile :

$$| \partial^2 v(t) / \partial x^2 | \le | f_x (t , v(t)) | + | f(t , v(t)) | \cdot | f_y (t , v(t)) | \le c_1 + c_0 \cdot c_2 \tag{10}$$

Urmatoarea identitate reprezinta dezvoltarea functiei v(x) în serie Taylor în jurul punctului x_j

$$v(x_{j+1}) = v(x_j) + [\partial v(x_j) / \partial x] (x_{j+1} - x_j) + [\partial^2 v(t_j) / \partial x^2] (x_{j+1} - x_j)^2 / 2 \tag{11}$$

unde $t_j \in (x_j , x_{j+1})$.

Aplicând teorema lui Lagrange pentru functia $g(y) = f(x, y)$ avem

$$f(x_j , v_j) - f(x_j , w_j) = f_y (x_j , z_j) \cdot (v_j - w_j) \tag{12}$$

pentru orice $v_j , w_j \in [a_2 , b_2]$, unde $z_j \in (v_j , w_j)$ (sau $z_j \in (w_j , v_j)$) .

Cu notatiile anterioare, din relatiile (1), (3), (5), (11) si (12) deducem

$$\varepsilon_{j+1} = v(x_{j+1}) - w_{j+1} = v(x_j) - w_j + h [f(x_j , v(x_j)) - f(x_j , w_j)] - \delta_j +$$

$$+ (h^2 / 2) [\partial^2 v(t_j) / \partial x^2] = \varepsilon_j + h \cdot f_y (x_j , z_j) \varepsilon_j - \delta_j + (h^2 / 2) [\partial^2 v(t_j) / \partial x^2] \tag{13}$$

Aplicând modulul în expresia (13) si folosind inegalitatile (7) si (10) se obtine în final

$$| \varepsilon_{j+1} | \le | \varepsilon_j | (1 + h c_2) + (h^2 / 2) (2 c_f + c_1 + c_0 c_2) = (1 + c) | \varepsilon_j | + d$$

$$c = h c_2 , \quad d = (h^2 / 2) (2 c_f + c_1 + c_0 c_2) \tag{14}$$

unde $\varepsilon_0 = 0$.

Propozitia 1. Pentru orice $j \in \mathbf{N}$ este satisfacuta inegalitatea

$$| \varepsilon_j | \le d \cdot [\exp (j \cdot c) - 1] / c \tag{15}$$

In adevar, relatia (15) este adevarata pentru $j = 0$ aceasta deoarece $\varepsilon_0 = 0$.

Presupunând adevarata relatia (15) si tinând seama în expresia (14) de inegalitatea cunoscuta $1 + c \le \exp (c)$, inegalitate ce este satisfacuta pentru orice $c \ge 0$, rezulta

$$| \varepsilon_{j+1} | \le (1 + c) | \varepsilon_j | + d \le (1 + c) d [\exp (j c) - 1] / c + d =$$

$$= d [(1 + c) \exp (j c) - 1] / c \le d [\exp (c) \exp (j c) - 1] / c$$

adica

$$| \varepsilon_{j+1} | \le d [\exp ((j + 1) c) - 1] / c$$

fapt ce încheie demonstratia prin inductie a inegalitatii (15).

Din (14) si (15) deducem

$$| \varepsilon_j | \le h [\exp (j h c_2) - 1] [2 c_f + c_1 + c_0 c_2] / [2 c2] \tag{16}$$

In concluzie, cum $j h \le b_1 - a_1$, $c_2 \ge 0$ iar functia exp (t) este crescatoare, rezulta :

Propozitia 2. Sirul $\{ w_j \}_j$, $0 \le j \le m$, definit recurent de relatia

$$w_{j+1} = w_j + h f(a_1 + j h , w_j) , \quad 0 \le j < m \tag{17}$$

cu $w_0 = v_0$, $h = (b_1 - a_1) / m$, constituie o aproximatie în punctele $x_j = a_1 + h j$ pentru solutia v (x) a ecuatiei diferentiale (1). In plus, eroarea $| \varepsilon |$ de forma (4) verifica inegalitatea

$$| \varepsilon | \le h [\exp ((b_1 - a_1) c_2) - 1] [2 c_f + c_1 + c_0 c_2] / (2 c_2) \tag{18}$$

Observatia 1. Constructia iterativa dupa formula (17) a aproximatiei discrete $\{ w_j \}_{j \in N}$ a

solutiei ecuatiei diferentiale (1) este cunoscuta în literatura sub denumirea de metoda Euler. Eroarea de aproximare $|\varepsilon|$ data de formula (4) depinde de pasul de discretizare h si se micsoreaza direct proportional cu descresterea acestuia.

Exemplul 1. Fie $v_0 = e^{-2}$, $a_1 = 0$, $b_1 = 2$, $m \in \{ 100, 400, 1600, 6400 \}$, $f(x, y) = (2x + 1) y - 10 x^2 - 5x + 5$

Solutia ecuatiei diferentiale $\partial v(x) / \partial x = f(x, v(x))$ ce respecta conditia initiala $v(0) = e^2$, este data de expresia $v(x) = 5x + \exp(x^2 + x - 2)$.

Aplicatia 1. Programul urmator va determina sirul $\{ w_j \}_j$, definit recurent de relatia $w_{j+1} = w_j + h \cdot f(2j/m, w_j)$, $0 \le j \le m - 1$, $w_0 = e^{-2}$ si va estima eroarea $|\varepsilon| = \max \{ |\varepsilon_j| ; \forall j, 0 \le j \le m \}$ unde $\varepsilon_j = v(2j/m) - w_j$

```
REM  Metoda Euler pentru rezolvarea ecuatiilor diferentiale ordinare ,   a1 ≤ x ≤ b1 ,  a2 ≤ y ≤ b2
DECLARE FUNCTION f! (x!, y!) :          DECLARE FUNCTION v! (x!)
a1 = 0 :        b1 = 2 :        PRINT "Metoda Euler        a1 = "; a1; "      b1 = "; b1
v0 = v(a1)    ' Valoarea initiala a solutiei
DATA 100,400,1600,6400        ' Numarul  m  de pasi utilizati
FOR im = 1 TO 4 :       READ m :       h = (b1 - a1) / m :       w1 = v0
vmax = v0 :     vmin = v0 :     wmax = v0 :     wmin = v0 :     erm = 0    ' erm = maximul erorii
FOR j = 1 TO m :       x = a1 + j * h :       v2 = v(x)              ' Solutia teoretica
IF v2 < vmin THEN vmin = v2
IF v2 > vmax THEN vmax = v2
w2 = w1 + h * f(x, w1)          ' Solutia empirica
IF w2 < wmin THEN wmin = w2
IF w2 > wmax THEN wmax = w2
er = ABS(v2 - w2)        ' er = diferenta dintre valoarea empirica si teoretica a solutiei in punctul x
IF erm < er THEN erm = er        ' Obtinerea maximului erorii
w1 = w2 :          NEXT j
PRINT "Numarul punctelor de discretizare = "; m; "    ( h = "; h; " )    Eroarea maxima = "; erm
PRINT "   Valoarea solutiei teoretice ( discretizata )  maxim = "; vmax; "      minim = "; vmin
PRINT "   Valoarea solutiei empirice ( discretizata )  maxim = "; wmax; "      minim = "; wmin
NEXT im

FUNCTION f (x, y) :      f = (2 * x + 1) * y - 10 * x ^ 2 - 5 * x + 5 :      END FUNCTION
FUNCTION v (x) :      v = EXP(x ^ 2 + x - 2) + 5 * x :      END FUNCTION
```

Exemplul 2. Vom executa programul prezentat în Aplicatia 1 în vederea solutionarii ecuatiei (1), unde functia f(x, y) este data în Exemplul 1. Rezultatele rularii programului sunt sintetizate în Tabelul 1, unde $h = (b - a) / m$,

$v_{min} = \min \{ v(a + j h) \mid 0 \le j \le m \}$ $\quad$ $w_{min} = \min \{ w_j \mid 0 \le j \le m \}$

$v_{max} = \max \{ v(a + j h) \mid 0 \le j \le m \}$ $\quad$ $w_{max} = \max \{ w_j \mid 0 \le j \le m \}$

Urmarind datele din Tabelul 1 se constata ca prin cresterea numarului m al punctelor de discretizare (fapt echivalent cu micsorarea pasului discretizarii) aproximarea $\{ w_j \}_{j \in N}$ a solutiei $v(x)$ se va îmbunatati (se reduce eroarea $|\varepsilon|$). Acest rezultat experimental este justificat din punct de vedere teoretic de Propozitia 2.

Tabelul 1. Eroarea maxima ε_{max} data de formula (4) pentru metoda Euler
(Exemplul 1 ; $a_1 = 0$; $b_1 = 2$; $v_0 = e^{-2}$)

m	v_{min}	w_{min}	v_{max}	w_{max}	ε_{max}
100	.1353353	.1353353	64.59815	22.25848	42.33967
400	.1353353	.1353353	64.59815	52.74207	11.85609
1600	.1353353	.1353353	64.59815	61.54557	3.052582
6400	.1353353	.1353353	64.59815	63.82863	.7695198

Exemplul 3. Functia $v(x) = (x - 1)(x - 3)(x - 4) = x^3 - 8x^2 + 19x - 12$ este solutie a ecuatiei diferentiale (1) pentru $f(x,y) = 3y / x + 8x - 38 + 36 / x$ cu conditia initiala $v_0 = v(a_1) = -4.375$ si $a_1 = 0.5$, $b_1 = 4.5$.

Rezultatele rularii programului sunt precizate în Tabelul 2. Ele confirma experimental Propozitia 2, aproximatia discreta $\{ w_j \}_j$ fiind cu atât mai precisa cu cât se micsoreaza pasul h de discretizare.

Tabelul 2. Eroarea maxima ε_{max} data de formula (4) pentru metoda Euler
(Exemplul 3 ; $a_1 = 0.5$; $b_1 = 4.5$; $v_0 = -4.375$)

m	v_{min}	w_{min}	v_{max}	w_{max}	ε_{max}
100	-4.375	-198.11750	2.625	-2.397526	200.7425
400	-4.375	-65.42783	2.625	-.2836739	68.05283
1600	-4.375	-15.85589	2.625	1.195017	18.48089
6400	-4.375	-4.375	2.625	1.838008	4.719050
12800	-4.375	-4.375	2.625	1.970218	2.365378
25600	-4.375	-4.375	2.625	2.039885	1.186433

Observatia 2. Datorita cumularii unor erori de aproximare valoarea maxima (minima) a solutiei teoretice $v(x)$ în punctele de discretizare x_j nu coincide neaparat cu valoarea maxima (minima) a solutiei empirice $\{ w_j \}_j$, $0 \le j \le m$.

2. Metoda Runge-Kutta.

Tinând seama ca functia $v(x)$ este solutie a ecuatiei diferentiale (1), conform teoremei lui Lagrange avem egalitatea

$$v(x_{j+1}) = v(x_j) + (\partial v(\alpha) / \partial x)[x_{j+1} - x_j] = v(x_j) + f(\alpha , v(\alpha))[x_{j+1} - x_j]$$

unde α este o valoare în intervalul (x_j , x_{j+1}) .

Daca w_j este o aproximatie a valorii $v(x_j)$ atunci egalitatea anterioara capata forma

$w_{j+1} = w_j + h\, f(\alpha , v(\alpha))$, $\alpha \in (a_1 + j\,h , a_1 + (j+1)\,h)$ (19)

La baza constructiei sirului de aproximatii punctuale $\{ w_j \}_j$ rezultate din metoda Euler s-a aflat aplicarea unei formule de tipul (19) în care, în locul valorii $f(\alpha , v(\alpha))$, s-a folosit aproximatia $f(x_j , w_j)$, fapt ce revine la a-l înlocui pe α cu x_j iar pe $v(\alpha)$ cu w_j, $w_j \approx v(x_j)$.

O aproximare "mai buna" a valorilor $f(\alpha , v(\alpha))$, pentru $\alpha \in (x_j , x_{j+1})$ este data de expresia

$[f(x_j , v(x_j)) + f(x_{j+1} , v(x_{j+1}))] / 2 \approx [f(x_j , w_j) + f(x_{j+1} , w_j + h\, f(x_j , w_j))] / 2$

Un asemenea tip de aproximare este utilizat într-o varianta a metodei Runge-Kutta. In acest caz aproximatia discreta $\{ w_j \}_j$ va fi obtinuta iterativ dupa formula

$w_{j+1} = w_j + h [f(a_1 + j\,h , w_j) + f(a_1 + (j+1)\,h , w_j + h\, f(a_1 + j\,h , w_j))] / 2$ (20)

Efectuând un calcul complex, asemanator aceluia ce a condus la Propozitia 2, deducem în final

Propozitia 3. Sirul $\{ w_j \}_j$, $0 \le j \le m$, definit recurent de expresia (20), unde $w_0 = v_0$ iar $h = (b_1 - a_1) / m$, aproximeaza solutia $v(x)$ a ecuatiei diferentiale (1) în punctele $x_j = a_1 + h\,j$. Eroarea $|\varepsilon|$ data de relatia (4) satisface o inegalitate de tipul

$$|\varepsilon| \le c_{RK} \cdot h^2 \qquad (21)$$

Valoarea c_{RK} este independenta de pasul de discretizare h, fiind însa dependenta de parametrii $a_1 , b_1 , c_f , c_0 , c_1 , c_2$ a caror semnificatie a fost deja precizata.

Observatia 3. Metoda Runge-Kutta este în general mai precisa decât metoda Euler deoarece maximul erorii de aproximare este proportional cu h^2 si nu cu h (formulele (18) si (21)). Aceasta afirmatie a fost verificata experimental pe mai multe exemple. Programul anterior a fost adaptat procedurii Runge-Kutta, instructiunea de atribuire `w2 = w1 + h * f(x, w1)` fiind înlocuita de urmatoarele doua comenzi

`w2a = w1 + h * f(x, w1)` `w2 = w1 + h * (f(x, w1) + f(x + h, w2a)) / 2`

Exemplul 4. Vom aplica metoda Runge-Kutta pentru ecuatia diferentiala (1) cu parametrii ce au fost definiti în Exemplul 1, adica $v_0 = e^{-2}$, $a_1 = 0$, $b_1 = 2$, $f(x , y) = (2\,x + 1)\,y - 10\,x^2 - 5\,x + 5$.

Rezultatele obtinute prin utilizarea metodei Runge-Kutta sunt mai precise (eroarea maxima de aproximare este mai mica iar maximul w_{max} al "solutiei empirice" se apropie mai mult de maximul v_{max} al "solutiei teoretice").

Observatia 3 este astfel confirmata si de analiza comparativa a Tabelelor 1 si 3.

Tabelul 3. Eroarea maxima ε_{max} data de formula (4) pentru metoda Runge-Kutta (Exemplul 1 ; $a_1 = 0$; $b_1 = 2$; $v_0 = e^{-2}$)

m	v_{min}	w_{min}	v_{max}	w_{max}	ε_{max}
100	.1353353	.1353353	64.59815	25.46464	39.13351
400	.1353353	.1353353	64.59815	55.42178	9.176369
1600	.1353353	.1353353	64.59815	62.34271	2.255444
6400	.1353353	.1353353	64.59815	64.03604	.5621109

Observatia 4. Deosebirea esentiala dintre metoda Euler si metoda Runge-Kutta consta în utilizarea valorilor $f(x_j , w_j)$, respectiv $[f(x_j , w_j) + f(x_j + h , w_j + h\, f(x_j , w_j))] / 2$ pentru a

aproxima expresia $f(\alpha, v(\alpha))$, unde $\alpha \in (x_j, x_j + h)$ rezulta din formula lui Lagrange (a se vedea si relatia (19)).

In raport cu forma concreta a functiei $f(x, y)$, s-ar putea ca $f(x_j, w_j)$ sa fie o mai buna aproximare pentru $f(\alpha, v(\alpha))$ decât $[f(x_j, w_j) + f(x_j + h, w_j + h f(x_j, w_j))] / 2$. Intr-o asemenea situatie metoda Euler este "mai precisa" decât metoda Runge-Kutta, fapt evidentiat si de exemplul urmator.

Exemplul 5. Vom aplica metoda Runge-Kutta pentru ecuatia diferentiala (1) unde atât functia $f(x, y)$ cât si conditia initiala v_0 sunt cele mentionate în Exemplul 2 ($v_0 = v(a_1) = -4.375$, $a_1 = 0.5$, $b_1 = 4.5$, $f(x,y) = 3y / x + 8x - 38 + 36 / x$).

Comparând rezultatele experimentale obtinute prin utilizarea metodelor Runge-Kutta si Euler, s-a constatat ca de aceasta data metoda Euler este "mai precisa" (Tabelele 2 si 4 ; criterii de comparare : eroarea maxima de aproximare ε_{max} si gradul de apropiere al maximului w_{max} al "solutiei empirice" de maximul v_{max} pentru "solutia teoretica"). In acest caz, indiferent de alegerea pasului h de discretizare, rezultatele aplicarii metodei Runge-Kutta (Tabelul 4) sunt mai putin precise decât acelea obtinute prin folosirea metodei Euler (Tabelul 2).

Tabelul 4. Eroarea maxima ε_{max} data de formula (4) pentru metoda Runge-Kutta (Exemplul 2 ; $a_1 = 0.5$; $b_1 = 4.5$; $v_0 = -4.375$)

m	v_{min}	w_{min}	v_{max}	w_{max}	ε_{max}
100	-4.375	-265.1158	2.625	-2.750065	267.7408
400	-4.375	-77.84393	2.625	-.5061191	80.46893
1600	-4.375	-18.44422	2.625	1.097229	21.06922
6400	-4.375	-4.375	2.625	1.805698	5.328175
12800	-4.375	-4.375	2.625	1.952854	2.669217
25600	-4.375	-4.375	2.625	2.031033	1.335167

REZOLVAREA ECUATIILOR DIFERENTIALE LINIARE

In aceasta sectiune vom discuta mai multe metode pentru rezolvarea ecuatiilor diferentiale liniare cu conditii la limita.

Suntem astfel interesati în a afla solutia $u(x)$ a unei ecuatii diferentiale liniare de ordin m, având forma

$$\frac{\partial^m u(x)}{\partial x^m} + \sum_{j=1}^{m} s_j(x) \frac{\partial^{n-j} u(x)}{\partial x^j} = h(x)$$

In vederea simplificarii expunerii vom exemplifica modul de solutionare a ecuatiilor diferentiale ordinare liniare de gradul al doilea.

Cu notatiile

$$L(u(x)) = u^{(2)}(x) + p(x)\,u^{(1)}(x) + q(x)\,u(x) \qquad u^{(k)}(x) = \frac{\partial^k u(x)}{\partial x^k}$$

$$R_1(u) = \alpha_1\,u(a) + \beta_1\,u^{(1)}(a) \qquad R_2(u) = \alpha_2\,u(b) + \beta_2\,u^{(1)}(b) \tag{1}$$

sa determinam solutia ecuatiei diferentiale liniare

$$L(u(x)) = h(x) \tag{2}$$

ce satisface conditiile la limita

$$R_1(u) = \gamma_1 \qquad R_2(u) = \gamma_2 \tag{3}$$

unde $\alpha_1^2 + \beta_1^2 > 0$, $\alpha_2^2 + \beta_2^2 > 0$ iar $p(x)$, $q(x)$, $h(x)$ sunt functii continue pe intervalul $[a, b]$

Observatia 1. Daca ecuatia omogena $L(u(x)) = 0$ admite numai solutia banala $u(x) = 0$, $x \in [a, b]$, atunci ecuatia neomogena (2) va admite solutie unica. In adevar, presupunând doua solutii $u_1(x)$, $u_2(x)$ ale ecuatiei diferentiale (2), adica $L(u_1(x)) = h(x)$, $L(u_2(x)) = h(x)$, rezulta $L(u(x)) = 0$, unde $u(x) = u_1(x) - u_2(x)$, $\forall x \in [a, b]$. Deoarece singura solutie a ecuatiei omogene este solutia nula deducem $u(x) = 0$, $a \le x \le b$, adica $u_1(x) = u_2(x)$ $\forall x \in [a, b]$.

Intentionam sa gasim o aproximare $w(x)$ pentru solutia $u(x)$ a ecuatiei diferentiale liniare neomogene (2) ce respecta restrictiile la limita (3). Vom exprima aproximatia $w(x)$ în raport cu un sistem de $n+1$ functii (liniar independente) $f_j(x)$, $0 \le j \le n$.

Vom alege functiile $f_j(x)$ cu conditia sa verifice relatiile

$$R_1(f_0) = \gamma_1 \qquad R_2(f_0) = \gamma_2 \qquad R_1(f_k) = 0 = R_2(f_k)\ ,\ 1 \le k \le n \tag{4}$$

In aceasta situatie aproximatia $w(x)$ exprimata ca o combinatie liniara a functiilor $f_j(x)$,

$$w(x) = f_0(x) + c_1 f_1(x) + c_2 f_2(x) + c_3 f_3(x) + \ldots + c_n f_n(x) \tag{5}$$

respecta conditiile la limita (3) si anume pentru $i = 1, 2$ avem

$$R_i(w) = R_i(f_0 + \sum_{j=1}^{n} c_j f_j) = R_i(f_0) + \sum_{j=1}^{n} c_j R_i(f_j) = \gamma_i \tag{6}$$

In mod evident egalitatea $L(w(x)) = h(x)$, $\forall x \in [a, b]$, nu este în general satisfacuta pentru aproximatia $w(x)$ a solutiei $u(x)$ a ecuatiei diferentiale (2) ce respecta restrictiile (3). Putem defini eroarea $E(x; c_1, c_2, \ldots, c_n)$ ce depinde de fiecare valoare x din intervalul $[a, b]$, drept diferenta

$$E(x; c_1, c_2, \ldots, c_n) = L(w(x)) - h(x) = L(f_0 + \sum_{j=1}^{n} c_j f_j(x)) - h(x) =$$

$$= L(f_0(x)) - h(x) + \sum_{j=1}^{n} c_j L(f_j(x)) \qquad (7)$$

Coeficientii c_j, $1 \le j \le n$, ce caracterizeaza aproximatia $w(x)$ vor fi determinati astfel încât eroarea "globala" E_0 sa fie minima. Termenul de "eroare globala" poate avea diferite acceptiuni.

La calculul erorii globale E_0 vor trebui a fi luate în considerare toate valorile erorilor "locale" $E(x ; c_1, c_2, \ldots, c_n)$ în puncte x din intervalul $[a , b]$. Acceptând diverse interpretari pentru eroarea "globala" E_0 putem imagina criterii diferite de "minimizare" a expresiei E_0.

Exemplul 1. Fie ecuatia diferentiala $u^{(2)}(x) + p(x) u^{(1)}(x) + q(x) u(x) = h(x)$, unde

$$p(x) = 2x^2 + 1 \qquad q(x) = 3 - x \qquad h(x) = 6x^4 - 30x^3 + 21x^2 + 24x - 19 \qquad (8)$$

$$\alpha_1 = \alpha_2 = 1 \qquad \beta_1 = \beta_2 = 0 \qquad \gamma_1 = -10 \qquad \gamma_2 = 20 \qquad a = 0 \qquad b = 6$$

a carei solutie $u(x)$ ce satisface conditiile la limita (3) este de forma

$$u(x) = x^3 - 8x^2 + 17x - 10 \qquad (9)$$

Având functiile $f_0(x)$, $f_1(x)$, $f_2(x)$ ce verifica restrictii de tip (4), unde

$$f_0(x) = 5x - 10 \qquad f_1(x) = x (x - 6) \qquad f_2(x) = x^2 (x - 6) \qquad (10)$$

Vom construi astfel aproximarea $w(x)$ a solutiei (9), anume (formula (5)),

$$w(x) = (5x - 10) + c_1 x (x-6) + c_2 x^2 (x - 6) = c_2 x^3 + (c_1 - 6c_2) x^2 + (5 - 6c_1) x - 10 \qquad (11)$$

Observatia 2. In vederea estimarii valorilor coeficientilor $c_1, c_2, \ldots, c_n$ ce definesc o aproximatie $w(x)$ a solutiei ecuatiei diferentiale liniare (2) cu conditiile la limita (3) pot fi utilizate mai multe metode. In final este de dorit de a se compara rezultatele astfel obtinute alegându-se varianta optima.

1. Metoda punctelor fixe

De multe ori, din lipsa de informatie, nu avem o imagine de ansamblu asupra comportarii erorilor $E(x ; c_1, c_2, \ldots, c_n)$ în orice punct x din intervalul $[a , b]$. Suntem adesea constrânsi în a masura erorile $E(x ; c_1, c_2, \ldots, c_n)$ numai în puncte izolate $x_1, x_2, \ldots, x_n$ din $[a , b]$. In aceste conditii coeficientii c_j, $1 \le j \le n$, vor fi determinati impunând ca erorile "locale" sa fie nule în punctele fixate $x_1, x_2, \ldots, x_n$, adica

$$E(x_j ; c_1, c_2, \ldots, c_n) = 0 \quad , \quad 1 \le j \le n \qquad (12)$$

Utilizând formula (7) ce defineste eroarea E rezulta urmatorul sistem liniar de n ecuatii în necunoscutele $c_1, c_2, \ldots, c_n$, adica

$$\begin{aligned}
&c_1 L(f_1(x_1)) + c_2 L(f_2(x_1)) + \ldots + c_n L(f_n(x_1)) = h(x_1) - L(f_0(x_1)) \\
&c_1 L(f_1(x_2)) + c_2 L(f_2(x_2)) + \ldots + c_n L(f_n(x_2)) = h(x_2) - L(f_0(x_2)) \\
&c_1 L(f_1(x_3)) + c_2 L(f_2(x_3)) + \ldots + c_n L(f_n(x_3)) = h(x_3) - L(f_0(x_3)) \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \qquad (13) \\
&c_1 L(f_1(x_n)) + c_2 L(f_2(x_n)) + \ldots + c_n L(f_n(x_n)) = h(x_n) - L(f_0(x_n))
\end{aligned}$$

Prin aplicarea formulei (5) solutia $(c_1, c_2, \ldots, c_n)$ a sistemului liniar (13) va fi utilizata la construirea aproximarii $w(x)$ a solutiei ecuatiei diferentiale (2), aproximarea $w(x)$ respectând în plus conditiile la limita (3).

Pentru simplificarea scrierii în sistemul (13) vom introduce functiile $g_k(x) = L(f_k(x))$,

$0 \le k \le n$, ce vor fi utilizate în continuare.

Exemplul 2. Considerând $n = 2$, sa aplicam metoda punctelor fixe pentru ecuatia diferentiala (8) precizata în Exemplul 1. Asadar vom opta pentru doua puncte fixe $a \le x_1 , x_2 \le b$

In aceste conditii, cum $L(f(x)) = f^{(2)}(x) + (2x^2 + 1) f^{(1)}(x) + (3 - x) f(x)$ deducem

$$L(f_0(x)) = (2x^2 + 1) 5 + (3 - x)(5x - 10) = 5x^2 + 25x - 25 \qquad (14)$$

$$L(f_1(x)) = 2 + (2x^2 + 1)(2x - 6) + (3 - x)(x^2 - 6x) = 3x^3 - 3x^2 - 16x - 4$$

$$L(f_2(x)) = (6x - 12) + (2x^2 + 1)(3x^2 - 12x) + (3 - x)(x^3 - 6x^2) = 5x^4 - 15x^3 - 15x^2 - 6x - 12$$

$$h(x) - L(f_0(x)) = 6x^4 - 30x^3 + 16x^2 - x + 6$$

Pentru $n = 2$ sistemul (13) devine

$$c_1 L(f_1(x_1)) + c_2 L(f_2(x_1)) = h(x_1) - L(f_0(x_1))$$
$$c_1 L(f_1(x_2)) + c_2 L(f_2(x_2)) = h(x_2) - L(f_0(x_2)) \qquad (15)$$

Asadar rezulta un sistem liniar de doua ecuatii cu necunoscutele c_1 , c_2

$$a_{11} c_1 + a_{12} c_2 = b_1$$
$$a_{21} c_1 + a_{22} c_2 = b_2 \qquad (16)$$

având solutia

$$c_1 = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11} a_{22} - a_{12} a_{21}} \qquad c_2 = \frac{b_2 a_{11} - b_1 a_{21}}{a_{11} a_{22} - a_{12} a_{21}} \qquad (17)$$

Aplicatia 1. Respectând notatiile anterioare, urmatorul program determina solutia (c_1 , c_2) a sistemului (16) pentru

$a_{11} = g_1(x_1)$ $a_{12} = g_2(x_1)$ $b_1 = h(x_1) - g_0(x_1)$

$a_{21} = g_1(x_2)$ $a_{22} = g_2(x_2)$ $b_2 = h(x_2) - g_0(x_2)$

unde $g_0(x) = L(f_0(x)) = 5x^2 + 25x - 25$, $g_1(x) = L(f_1(x)) = 3x^3 - 3x^2 - 16x - 4$,

$g_2(x) = L(f_2(x)) = 5x^4 - 15x^3 - 15x^2 - 6x - 12$

```
REM  Metoda punctelor fixe
DECLARE FUNCTION f0! (x!) :        DECLARE FUNCTION f1! (x!)
DECLARE FUNCTION f2! (x!) :        DECLARE FUNCTION h! (x!)
DECLARE FUNCTION u! (x!) :         DECLARE FUNCTION g0! (x!)
DECLARE FUNCTION g1! (x!) :        DECLARE FUNCTION g2! (x!)
CLS :              INPUT "x1, x2 ="; x1, x2
a11 = g1(x1):      a12 = g2(x1):      b1 = h(x1) - g0(x1)
a21 = g1(x2):      a22 = g2(x2):      b2 = h(x2) - g0(x2)
d = a11 * a22 - a12 * a21:     c1 = (b1 * a22 - b2 * a12) / d:     c2 = (b2 * a11 - b1 * a21) / d
PRINT USING "V = ( x1 = ##.# , x2 = ##.# )"; x1; x2
PRINT USING "u :   ######.## ; ######.## ; ######.## ; ######.## ; "; 1; -8; 17; -10
PRINT USING "w :   ######.## ; ######.## ; ######.## ; ######.## ; ";
c2; c1 - 6 * c2; 5 - 6 * c1; -10
PRINT "Valorile functiei  u  in punctele :";
FOR j = 0 TO 6: PRINT j; " , "; :          NEXT j:        PRINT
FOR j = 0 TO 6: PRINT USING "#####.## ; "; u(j); :        NEXT j:        PRINT
PRINT "Valorile functiei  w  in punctele specificate"
FOR j = 0 TO 6:      w = f0(j) + c1 * f1(j) + c2 * f2(j)
```

```
PRINT USING "#####.## ; "; w; :          NEXT j.          PRINT

FUNCTION f0 (x) :     f0 = 5 * x - 10:          END FUNCTION
FUNCTION f1 (x) :     f1 = x * (x - 6):         END FUNCTION
FUNCTION f2 (x) :     f2 = x * x * (x - 6):     END FUNCTION
FUNCTION g0 (x) :     g0 = 5 * x ^ 2 + 25 * x - 25:          END FUNCTION
FUNCTION g1 (x) :     g1 = 3 * x ^ 3 - 3 * x ^ 2 - 16 * x - 4:       END FUNCTION
FUNCTION g2 (x) :     g2 = 5 * x ^ 4 - 15 * x ^ 3 - 15 * x ^ 2 - 6 * x - 12:     END FUNCTION
FUNCTION h (x) :      h = 6 * x ^ 4 - 30 * x ^ 3 + 21 * x ^ 2 + 24 * x - 19:     END FUNCTION
FUNCTION u (x) :      u = x ^ 3 - 8 * x ^ 2 + 17 * x - 10:       END FUNCTION
```

Exemplul 3. Vom compara solutia "exacta" u(x) a ecuatiei diferentiale din Exemplul 1 (formula (9)) cu aproximatia sa w(x) obtinuta prin metoda punctelor fixe (forma (11)). In expresia (5) ce defineste functia w(x) coeficientii c_1 , c_2 , c_3 sunt dati de (17) .

Notam cu w(x; V) valoarea aproximatiei w(x) în varianta V. In cadrul fiecarei variante V caracterizata de perechea (x_1 , x_2) sunt specificate valorile punctelor fixe x_1 si x_2 în care erorile $E(x ; c_1, c_2)$ sunt presupuse a fi nule (conditia (12)).

Rularea programului prezentat în Aplicatia 1 se va efectua în variantele : V1 ≡ (0 , 1) ; V2 ≡ (0 , 2) ; V3 ≡ (0 , 3) ; V4 ≡ (0 , 4) ; V5 ≡ (0 , 5) ; V6 ≡ (0 , 6) ; V7 ≡ (1 , 2) ; V8 ≡ (2 , 3) ; V9 ≡ (3 , 4) ; V10 ≡ (4 , 5) ; V11 ≡ (5 , 6) .

Considerând variantele V1-V11, Tabelul 1a prezinta valorile luate de solutia u(x) a ecuatiei diferentiale (2) cât si de aproximatia sa (formula(11)) în punctele "fixe" x ∈ { 0, 1, 2, 3, 4, 5, 6 }. In Tabelul 1b sunt dati coeficientii ce definesc polinomul de aproximatie w(x) rezultat prin aplicarea metodei punctelor fixe.

Tabelul 1a. Valorile solutiei u(x) si a aproximatiei sale w(x;V) obtinuta prin metoda punctelor fixe în variantele V1-V11 (n = 2).

	x=0	x=1	x=2	x=3	x=4	x=5	x=6
u(x)	-10.00	0.00	0.00	-4.00	-6.00	-0.00	20.00
w(x; V1)	-10.00	-16.91	-3.53	18.50	37.53	41.91	20.00
w(x; V2)	-10.00	24.04	29.23	18.50	4.77	0.96	20.00
w(x; V3)	-10.00	12.68	20.14	18.50	13.86	12.32	20.00
w(x; V4)	-10.00	3.15	12.52	18.50	21.48	21.85	20.00
w(x; V5)	-10.00	41.69	43.35	18.50	-9.35	-16.69	20.00
w(x; V6)	-10.00	21.51	27.21	18.50	6.79	3.49	20.00
w(x; V7)	-10.00	0.00	0.00	-4.00	-6.00	0.00	20.00
w(x; V8)	-10.00	0.95	1.16	-3.11	-5.57	0.04	20.00
w(x; V9)	-10.00	1.09	1.38	-2.86	-5.34	0.18	20.00
w(x; V10)	-10.00	1.03	1.07	-3.44	-6.08	-0.41	20.00
w(x; V11)	-10.00	1.50	1.55	-3.19	-6.12	-0.60	20.00

Observatia 3. Cum era si normal, precizia aproximarii w(x) depinde de pozitia punctelor fixe x_j , $1 \le j \le n$, ce au fost selectate. In varianta V7 , prin rezolvarea sistemului (16) se obtine $c_1 = -2$

si $c_2 = 1$. In acest caz aproximatia w(x) calculata dupa expresia (11) coincide chiar cu solutia exacta u(x) data de formula (9) (a se vedea Tabelele 1a si 1b).

Tabelul 1b. Coeficientii monoamelor x^k, $0 \le k \le 3$, ce definesc polinoamele u(x) si w(x;V) în variantele V1-V11 (metoda punctelor fixe, n = 2).

	x^3	x^2	x^1	x^0
u(x)	1.00	-8.00	17.00	-10.00
w(x; V1)	-1.94	15.97	-20.94	-10.00
w(x; V2)	2.15	-20.88	52.77	-10.00
w(x; V3)	1.02	-10.66	32.32	-10.00
w(x; V4)	0.07	-2.09	15.17	-10.00
w(x; V5)	3.92	-36.77	84.54	-10.00
w(x; V6)	1.90	-18.61	48.22	-10.00
w(x; V7)	1.00	-8.00	17.00	-10.00
w(x; V8)	1.05	-8.51	18.41	-10.00
w(x; V9)	1.05	-8.53	18.58	-10.00
w(x; V10)	1.07	-8.71	18.67	-10.00
w(x; V11)	1.10	-9.03	19.42	-10.00

2. Metoda subdomeniilor

Fie o diviziune $a = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = b$ a intervalui [a , b]. Coeficientii c_1 , c_2 , ..., c_n vor fi determinati astfel încât media erorilor $E(x ; c_1,c_2,\ldots,c_n)$ pe fiecare interval $[x_{k-1} , x_k]$, $1 \le k \le n$, sa fie nula, adica

$$\int_{x_{k-1}}^{x_k} E(x;c_1,c_2,\ldots,c_n)\,dx = 0 \qquad (18)$$

Tinând seama de expresia (7) ce da eroarea de aproximare E se obtine urmatorul sistem liniar de n ecuatii cu necunoscutele c_1 , c_2 , ... , c_n ,

$$\sum_{j=1}^{n} c_j \left(\int_{x_{k-1}}^{x_k} L(f_j(x))\,dx \right) = \int_{x_{k-1}}^{x_k} \Big(h(x) - L(f_0(x))\Big)\,dx \qquad (19)$$

Coeficientii c_j , $1 \le j \le n$, astfel rezultati vor determina aproximatia w(x) precizata de formula (5).

Exemplul 4. Vom aplica metoda subdomeniilor în cazul precizat de Exemplul 1, cu n = 2. Atunci : $x_0 = a = 0$; $x_1 = \lambda$, $a < \lambda < b$; $x_2 = b = 6$. Cu notatiile

$$a_{11}(\lambda) = \int_0^{\lambda} L(f_1(x))\,dx = \int_0^{\lambda} \left(3x^3 - 3x^2 - 16x - 4\right) dx = g_1(\lambda) - g_1(0) \qquad (20)$$

$$a_{12}(\lambda) = \int_0^{\lambda} L(f_2(x))\,dx = \int_0^{\lambda} \left(5x^4 - 15x^3 - 15x^2 - 6x - 12\right) dx = g_2(\lambda) - g_2(0)$$

$$b_1(\lambda) = \int_0^{\lambda} \left(L(f_0(x)) - h(x)\right) dx = \int_0^{\lambda} \left(6x^4 - 30x^3 + 16x^2 - x + 6\right) dx = g_0(\lambda) - g_0(0)$$

$$a_{21}(\lambda) = \int_{\lambda}^{6} L(f_1(x))\,dx = \int_{\lambda}^{6} \left(3x^3 - 3x^2 - 16x - 4\right) dx = g_1(6) - g_1(\lambda)$$

$$a_{22}(\lambda) = \int_{\lambda}^{6} L(f_2(x))\,dx = \int_0^{\lambda} \left(5x^4 - 15x^3 - 15x^2 - 6x - 12\right) dx = g_2(6) - g_2(\lambda)$$

$$b_2(\lambda) = \int_{\lambda}^{6} \left(L(f_0(x)) - h(x)\right) dx = \int_{\lambda}^{6} \left(6x^4 - 30x^3 + 16x^2 - x + 6\right) dx = g_0(6) - g_0(\lambda)$$

$g_0(x) = 6x^5/5 - 15x^4/2 + 16x^3/3 - x^2/2 + 6x$

$g_1(x) = 3x^4/4 - x^3 - 8x^2 - 4x$ $\qquad$ $g_2(x) = x^5 - 15x^4/4 - 5x^3 - 3x^2 - 12x$

se obtine sistemul liniar

$$a_{11}(\lambda)\, c_1 + a_{12}(\lambda)\, c_2 = b_1(\lambda)$$
$$a_{21}(\lambda)\, c_1 + a_{22}(\lambda)\, c_2 = b_2(\lambda) \qquad (21)$$

cu solutia

$$c_1(\lambda) = \frac{b_1(\lambda)\, a_{22}(\lambda) - b_2(\lambda)\, a_{12}(\lambda)}{a_{11}(\lambda)\, a_{22}(\lambda) - a_{12}(\lambda)\, a_{21}(\lambda)}$$

$$c_2(\lambda) = \frac{b_2(\lambda)\, a_{11}(\lambda) - b_1(\lambda)\, a_{21}(\lambda)}{a_{11}(\lambda)\, a_{22}(\lambda) - a_{12}(\lambda)\, a_{21}(\lambda)} \qquad (22)$$

Asadar pentru o valoare fixata a parametrului λ, $0 < \lambda < 6$, aproximatia $w(x ; \lambda)$ a solutiei $u(x)$ a ecuatiei diferentiale liniare (8) cu conditiile la limita respective este data de expresia

$$w(x ; \lambda) = c_2(\lambda)\, x^3 + (c_1(\lambda) - 6c_2(\lambda))\, x^2 + (5 - 6c_1(\lambda))\, x - 10 \qquad (23)$$

Aplicatia 2. Programul urmator calculeaza aproximatia $w(x ; \lambda)$ data de (23) unde coeficientii $c_1(\lambda)$, $c_2(\lambda)$ depind de parametrul λ, $0 = a < a < b = 6$, si au forma (22).

```
REM  Metoda subdomeniilor
DECLARE FUNCTION u! (x!) :  DECLARE FUNCTION g0! (x!)
DECLARE FUNCTION g1! (x!) :  DECLARE FUNCTION g2! (x!)
CLS :      s = .5
WHILE s < 6!
a11 = g1(s) - g1(0) :        a12 = g2(s) - g2(0) :        b1 = g0(s) - g0(0)
a21 = g1(6) - g1(s) :        a22 = g2(6) - g2(s) :        b2 = g0(6) - g0(s)
d   a11 * a22 - a12 * a21
```

```
c1 = (b1 * a22 - b2 * a12) / d :          c2 = (b2 * a11 - b1 * a21) / d
PRINT USING "s = ##.##"; s
PRINT USING "u :    ######.## ; ######.## ; ######.## ; ######.## , ", 1; -8, 17; -10
PRINT USING "w :    ###### ## , ######.## ; ######.## ; ######.## , ";
c2; c1 - 6 * c2; 5 - 6 * c1; -10
PRINT "Valorile functiei  u  in punctele :";
FOR x = 0 TO 6 :        PRINT x; " , "; :        NEXT x :        PRINT
FOR x = 0 TO 6 :      PRINT USING "#####.## ; "; u(x); :      NEXT x :      PRINT
PRINT "Valorile functiei  w  in punctele specificate"
FOR x = 0 TO 6 :        w = c2 * x * x * x + (c1 - 6 * c2) * x * x + (5 - 6 * c1) * x - 10
PRINT USING "#####.## ; "; w; :          NEXT x :          PRINT
s = s + .5 :              WEND

FUNCTION g0 (x) :       g0 = 1.2 * x ^ 5 - 7.5 * x ^ 4 + (16! / 3!) * x ^ 3 - .5 * x ^ 2 + 6 * x
                        END FUNCTION
FUNCTION g1 (x) :     g1 = .75 * x ^ 4 - x ^ 3 - 8 * x ^ 2 - 4 * x :      END FUNCTION
FUNCTION g2 (x) :     g2 = x ^ 5 - 3.75 * x ^ 4 - 5 * x ^ 3 - 3 * x ^ 2 - 12 * x
                      END FUNCTION
FUNCTION u (x) :      u = x ^ 3 - 8 * x ^ 2 + 17 * x - 10 :       END FUNCTION
```

Exemplul 5. Se va compara solutia exacta u(x) data de formula (9) pentru ecuatia diferentiala (8) cu aproximatia $w(x ; \lambda)$ obtinuta prin metoda subdomeniilor (expresia (23)). Valorile $w(x ; \lambda)$ sunt deduse prin apelarea procedurii din Aplicatia 2.

Tabelul 2a. Valorile solutiei u(x) si a aproximatiei sale $w(x; \lambda)$ obtinuta prin metoda subdomeniilor (formula (23)).

	x=0	x=1	x=2	x=3	x=4	x=5	x=6
u(x)	-10.00	0.00	0.00	-4.00	-6.00	-0.00	20.00
w(x; 0.5)	-10.00	4.60	4.57	-1.98	-6.99	-2.36	20.00
w(x; 1.0)	-10.00	1.72	1.66	-3.37	-6.53	-1.02	20.00
w(x; 1.5)	-10.00	1.25	1.18	-3.59	-6.46	-0.80	20.00
w(x; 2.0)	-10.00	1.72	1.65	-3.37	-6.53	-1.02	20.00
w(x; 2.5)	-10.00	-18.20	-18.54	-12.95	-3.38	8.25	20.00
w(x; 3.0)	-10.00	-0.41	-0.50	-4.39	-6.20	-0.03	20.00
w(x; 3.5)	-10.00	0.20	0.12	-4.10	-6.29	-0.31	20.00
w(x; 4.0)	-10.00	0.48	0.40	-3.97	-6.34	-0.44	20.00
w(x; 4.5)	-10.00	0.66	0.58	-3.88	-6.37	-0.53	20.00
w(x; 5.0)	-10.00	0.80	0.73	-3.81	-6.39	-0.59	20.00
w(x; 5.5)	-10.00	0.92	0.85	-3.75	-6.41	-0.65	20.00

Tabelul 2a prezinta valorile luate de solutia u(x) a ecuatiei diferentiale (2) si de aproximatia sa $w(x ; \lambda)$ (formul (23)) în punctele de diviziune $x \in \{ 0, 1, 2, 3, 4, 5, 6 \}$. In Tabelul 2b sunt dati coeficientii polinomului $w(x ; \lambda)$ rezultat din aplicarea metodei subdomeniilor. Urmarind aceste

tabele constatam ca functia w(x; 3.5) rezultata prin aplicarea metodei subdomeniilor aproximeaza foarte bine solutia u(x) a ecuatiei diferentiale liniare din Exemplul 1.

Tabelul 2b. Coeficientii monoamelor x^k, $0 \le k \le 3$, pentru polinoamele u(x) si w(x; λ) (metoda subdomeniilor, n = 2).

	x^3	x^2	x^1	x^0
u(x)	1.00	-8.00	17.00	-10.00
w(x; 0.5)	1.35	-11.35	24.60	-10.00
w(x; 1.0)	1.14	-9.31	19.89	-10.00
w(x; 1.5)	1.10	-8.97	19.11	-10.00
w(x; 2.0)	1.14	-9.30	19.88	-10.00
w(x; 2.5)	-0.32	4.90	-12.78	-10.00
w(x; 3.0)	0.98	-7.79	16.40	-10.00
w(x; 3.5)	1.03	-8.22	17.39	-10.00
w(x; 4.0)	1.05	-8.42	17.85	-10.00
w(x; 4.5)	1.06	-8.55	18.15	-10.00
w(x; 5.0)	1.07	-8.65	18.39	-10.00
w(x; 5.5)	1.08	-8.73	18.58	-10.00

3. Metoda celor mai mici patrate în cazul discret

Prin metoda punctelor fixe se cerea ca erorile E sa fie nule în puncte fixate. Aceasta metoda presupune alegerea a n puncte "de referinta" $x_1, x_2, \dots, x_n$, tot atâtea cât numarul parametrilor $c_1, c_2, \dots, c_n$ utilizati ca ponderi pentru functiilor $f_j(x)$, $1 \le j \le n$, ce caracterizeaza aproximatia w(x) (a se vedea formula (5)). In cazul în care initial avem un numar m de puncte fixe, cu $m > n$, numai n dintre aceste puncte vor fi utilizate efectiv pierzîndu-se astfel informatie utila.

In practica este de dorit folosirea tuturor celor m puncte "fixe" $x_1, x_2, \dots, x_m$ disponibile pentru care a fost evaluata eroarea E. Acest lucru se va realiza definind eroarea globala E_0 drept suma patratelor erorilor punctuale $E(x_t; c_1, c_2, \dots, c_n)$, $1 \le t \le m$,

$$E_0 = \sum_{t=1}^{m} \left(E(x_k; c_1, c_2, \dots, c_n) \right)^2 \qquad (24)$$

Nu este de dorit de a defini eroarea globala E_0 ca o suma simpla a erorilor locale deoarece în acest caz poate rezulta o eroare globala E_0 chiar nula în situatia unor erori locale E foarte mari dar de semne contrare.

Coeficientii $c_1, c_2, \dots, c_n$ vor fi determinati astfel încât sa minimizeze eroarea globala E_0.

Tinând seama de expresia (7) ce da eroarea E în punctele x_t, $1 \le t \le m$, rezulta

$$E_0(c_1,c_2,\ldots,c_n) = \sum_{t=1}^{m}\left(L(f_0(x_t)) - h(x) + \sum_{j=1}^{n} c_j\, L(f_j(x_t))\right)^2 \qquad (25)$$

In vederea minimizarea erorii globale E_0 se impun conditiile

$$\frac{\partial E_0(c_1,c_2,\ldots,c_n)}{\partial c_k} = 0 \quad , \qquad 1 \le k \le n \qquad (26)$$

Rezulta astfel un sistem liniar de n ecuatii în necunoscutele c_1 , c_2 , ... , c_n , unde linia k , $1 \le k \le n$, a acestui sistem este de forma,

$$\sum_{j=1}^{n} c_j \left(\sum_{t=1}^{m} L(f_j(x_t))\, L(f_k(x_t))\right) = \sum_{t=1}^{m} L(f_k(x_t))\big(h(x_t)) - L(f_0(x_t))\big) \qquad (27)$$

Odata determinati coeficientii c_j , $1 \le j \le n$, utilizând formula (5) se obtine aproximatia w(x) a ecuatiei diferentiale (2), aproximatie ce respecta conditii la limita de tip (3).

Exemplul 6. Pentru ecuatia diferentiala precizata în Exemplul 1 vom considera m=4 , punctele fixe fiind $x_1 = 0$, $x_2 = 2$, $x_3 = 4$, $x_4 = 6$. Aplicând în aceasta situatie metoda celor mai mici patrate, sistemul liniar (27) are forma particulara (16). Componentele c_1 , c_2 ale solutiei acestui sistem liniar de doua ecuatii sunt date de expresiile (17) pentru

$$a_{jk} = \sum_{t=1}^{4} L(f_j(x_t))\, L(f_k(x_t)) \qquad j , k \in \{1 , 2\}$$

$$b_k = \sum_{t=1}^{4} L(f_k(x_t))\big(h(x_t) - L(f_0(x_t))\big) \qquad k \in \{1 , 2\} \qquad (28)$$

Functiile $h(x) - L(f_0(x))$, $L(f_j(x))$, $j \in \{1 , 2\}$, sunt definite prin relatiile (14).

Aplicatia 3. Fie ecuatia diferentiala (2) în conditiile (8). Programul urmator calculeaza aproximatia w(x) a solutiei u(x) prin metoda celor mai mici patrate în cazul discret (cu m = 4).

Functia w(x) are expresia (5), coeficientii c_j , $1 \le j \le n$, fiind o solutie a sistemului liniar (27) cu n = 2.

```
REM  Metoda celor mai mici patrate - cazul discret -
DECLARE FUNCTION h! (x!) :          DECLARE FUNCTION u! (x!)
DECLARE FUNCTION g0! (x!) :         DECLARE FUNCTION g1! (x!)
DECLARE FUNCTION g2! (x!)
CLS :       x1 = 0 :        x2 = 2 :        x3 = 4 :        x4 = 6
a11 = g1(x1) * g1(x1) + g1(x2) * g1(x2) + g1(x3) * g1(x3) + g1(x4) * g1(x4)
a12 = g1(x1) * g2(x1) + g1(x2) * g2(x2) + g1(x3) * g2(x3) + g1(x4) * g2(x4):     a21 = a12
a22 = g2(x1) * g2(x1) + g2(x2) * g2(x2) + g2(x3) * g2(x3) + g2(x4) * g2(x4)
b1 = g1(x1) * (h(x1) - g0(x1)) + g1(x2) * (h(x2) - g0(x2)) + g1(x3) * (h(x3) - g0(x3)) +
     g1(x4) * (h(x4) - g0(x4))
b2 = g2(x1) * (h(x1) - g0(x1)) + g2(x2) * (h(x2) - g0(x2)) + g2(x3) * (h(x3) - g0(x3)) +
     g2(x4) * (h(x4) - g0(x4))
d = a11 * a22 - a12 * a21
c1 = (b1 * a22 - b2 * a12) / d :                c2 = (b2 * a11 - b1 * a21) / d
PRINT USING "u :    ######.## ; ######.## ; ######.## ; ######.## ; "; 1; -8; 17; -10
```

```
PRINT USING "w :    ######.## ; ######.## ; ######.## ; ######.## ; ";
c2; c1 - 6 * c2; 5 - 6 * c1; -10
PRINT "Valorile functiei u  in punctele :";
FOR j = 0 TO 6: PRINT j; " , "; : NEXT j: PRINT
FOR j = 0 TO 6:       PRINT USING "#####.## ; "; u(j); :       NEXT j :       PRINT
PRINT "Valorile functiei  w  in punctele specificate"
FOR x = 0 TO 6 :        w = c2 * x * x * x + (c1 - 6 * c2) * x * x + (5 - 6 * c1) * x - 10
PRINT USING "#####.## ; "; w; :          NEXT x :          PRINT

FUNCTION g0 (x) :      g0 = 5 * x ^ 2 + 25 * x - 25 :       END FUNCTION
FUNCTION g1 (x) :      g1 = 3 * x ^ 3 - 3 * x ^ 2 - 16 * x - 4 :       END FUNCTION
FUNCTION g2 (x) :      g2 = 5 * x ^ 4 - 15 * x ^ 3 - 15 * x ^ 2 - 6 * x - 12 :   END FUNCTION
FUNCTION h (x) :       h = 6 * x ^ 4 - 30 * x ^ 3 + 21 * x ^ 2 + 24 * x - 19 :   END FUNCTION
FUNCTION u (x) :       u = x ^ 3 - 8 * x ^ 2 + 17 * x - 10 :          END FUNCTION
```

Exemplul 7. In urma rularii programului proiectat în Aplicatia 3, pentru solutia u(x) a ecuatiei diferentiale din Exemplul 1 s-a obtinut aproximatia w(x) , unde

$u(x) = x^3 - 8x^2 + 17x - 10$ $\qquad$ $w(x) = 1.09\,x^3 - 8.81\,x^2 + 18.76\,x - 10.00$

Comparatii între valorile solutiei u(x) si valorile w(x) ale aproximatiei sale pot fi facute urmarindu-se datele din Tabelul 3 ($0 \le x \le 6$).

Tabelul 3. Valorile solutiei u(x) si ale aproximatiei sale w(x) obtinuta prin metoda celor mai mici patrate (varianta discreta, m = 4).

	x=0	x=1	x=2	x=3	x=4	x=5	x=6
u(x)	-10.00	0.00	0.00	-4.00	-6.00	-0.00	20.00
w(x)	-10.00	1.03	0.96	-3.69	-6.42	-0.69	20.00

4. Metoda celor mai mici patrate în cazul continuu

In cazul în care eroarea $E(x ; c_1,c_2,\ldots,c_n)$ poate fi evaluata în orice punct $a \le x \le b$, vom defini eroarea globala E_0 prin relatia

$$E_0 = \int_a^b \left(E(x ; c_1,c_2,\ldots,c_n) \right)^2 dx \qquad (29)$$

Expresia (29) este transpunerea relatiei (24) în cazul continuu.

Explicitând eroarea $E(x ; c_1, c_2, \ldots, c_n)$ dupa formula (7) rezulta o expresie de tipul (25),

$$E_0(c_1,c_2,\ldots,c_n) = \int_a^b \left(L(f_0(c)) - h(x) + \sum_{j=1}^{n} c_j\, L(f_j(x)) \right)^2 dx \qquad (30)$$

Minimul expresiei $E_0(c_1, c_2, \ldots, c_n)$ se realizeaza impunând conditiile (26) de anulare a derivatelor partiale ale functiei E_0. Obtinem în final un sistem liniar în necunoscutele $c_1, c_2, \ldots, c_n$ unde linia k, $1 \le k \le n$, a acestui sistem are expresia

$$\sum_{j=1}^{n} c_j \left(\int_a^b L(f_j(x))\, L(f_k(x))\, dx \right) = \int_a^b L(f_k(x)) \big(h(x) - L(f_0(x)) \big)\, dx \qquad (31)$$

Cu ajutorul coeficientilor c_j, $1 \le j \le n$, astfel determinati se construieste dupa formula (5) aproximatia $w(x)$ a solutiei ecuatiei diferentiale liniare (2), aproximatie ce verifica în plus si conditii la limita de tip (3).

Exemplul 8. Vom aplica metoda celor mai mici patrate în varianta continua considerând ecuatia diferentiala particulara din Exemplul 1, cu $n = 2$, în conditiile (8). In aceasta varianta sistemul liniar (31) este de tipul (16), în necunoscutele c_1, c_2 date de expresiile (17) pentru

$$a_{jk} = \int_a^b L(f_j(x))\, L(f_k(x))\, dx \qquad \forall\ j, k \in \{1, 2\}$$

$$b_k = \int_a^b L(f_k(x)) \big(h(x) - L(f_0(x)) \big)\, dx \qquad k \in \{1, 2\} \qquad (32)$$

unde $a = 0$ si $b = 6$.

Precizam ca functiile $h(x) - L(f_0(x))$, $L(f_j(x))$, $j \in \{1, 2\}$, au fost definite prin relatiile (14).

Aplicatia 4. Urmatorul program BASIC evalueaza aproximatia $w(x)$ a solutiei $u(x)$ prin metoda celor mai mici patrate în varianta continua.

Integralele ce definesc coeficientii a_{jk}, b_k sunt aplicate unor polinoame fapt ce permite o evaluare directa a lor. In program s-a preferat aproximarea acestor integrale prin sume de tip Riemann considerând o retea echidistanta de $m = 100$ puncte ce divide intervalul de integrare $[0, 6]$.

Asadar programul ce a fost propus ne conduce la aproximatia $w(x)$ si în cazul în care functiile $h(x)$, $f_k(x)$, $L(f_k(x))$ nu sunt neaparat polinoame.

```
REM  Metoda celor mai mici patrate in varianta continua
DECLARE FUNCTION h! (x!) :          DECLARE FUNCTION u! (x!)
DECLARE FUNCTION g0! (x!) :         DECLARE FUNCTION g1! (x!)
DECLARE FUNCTION g2! (x!)
CLS :        m = 100 :        p = 6 / m
a11 = 0 :   FOR i = 1 TO m :   x = p * i :   a11 = a11 + g1(x) * g1(x) :   NEXT i
a12 = 0 :   FOR i = 1 TO m :   x = p * i :   a12 = a12 + g1(x) * g2(x) :   NEXT i :   a21 = a12
a22 = 0 :   FOR i = 1 TO m :   x = p * i :   a22 = a22 + g2(x) * g2(x) :   NEXT i
b1 = 0 :   FOR i = 1 TO m :   x = p * i :   b1 = b1 + g1(x) * (h(x) - g0(x)) :   NEXT i
b2 = 0 :   FOR i = 1 TO m :   x = p * i :   b2 = b2 + g2(x) * (h(x) - g0(x)) :   NEXT i
d = a11 * a22 - a12 * a21
c1 = (b1 * a22 - b2 * a12) / d :          c2 = (b2 * a11 - b1 * a21) / d
PRINT USING "u :   ######.## ; ######.## ; ######.## ; ######.## ; "; 1; -8; 17; -10
PRINT USING "w :   ######.## ; ######.## ; ######.## ; ######.## ; ";
c2; c1 - 6 * c2; 5 - 6 * c1; -10
PRINT "Valorile functiei  u  in punctele :";
FOR j = 0 TO 6: PRINT j; " , "; : NEXT j: PRINT
```

```
FOR j = 0 TO 6: PRINT USING "#####.## ; "; u(j); : NEXT j: PRINT
PRINT "Valorile functiei w in punctele specificate"
FOR x = 0 TO 6: w = c2 * x * x * x + (c1 - 6 * c2) * x * x + (5 - 6 * c1) * x - 10
PRINT USING "#####.## ; "; w; : NEXT x: PRINT

FUNCTION g0 (x) :   g0 = 5 * x ^ 2 + 25 * x - 25 :          END FUNCTION
FUNCTION g1 (x) :   g1 = 3 * x ^ 3 - 3 * x ^ 2 - 16 * x - 4 :   END FUNCTION
FUNCTION g2 (x) :   g2 = 5 * x ^ 4 - 15 * x ^ 3 - 15 * x ^ 2 - 6 * x - 12 :   END FUNCTION
FUNCTION h (x) :    h = 6 * x ^ 4 - 30 * x ^ 3 + 21 * x ^ 2 + 24 * x - 19 :   END FUNCTION
FUNCTION u (x) :    u = x ^ 3 - 8 * x ^ 2 + 17 * x - 10 :   END FUNCTION
```

Exemplul 9. In urma rularii programului propus în Aplicatia 4, pentru solutia u(x) a ecuatiei diferentiale liniare definita în Exemplul 1 cu conditiile la limita (8) s-a obtinut aproximatia w(x) , unde

$u(x) = x^3 - 8x^2 + 17x - 10$ $\qquad$ $w(x) = 1.08\,x^3 - 8.81\,x^2 + 18.82\,x - 10.00$

Acuratetea aproximarii obtinute poate fi apreciata urmarindu-se datele din Tabelul 4.

Tabelul 4. Valorile solutiei u(x) si ale aproximatiei sale w(x) obtinuta prin metoda celor mai mici patrate (varianta continua).

	x=0	x=1	x=2	x=3	x=4	x=5	x=6
u(x)	-10.00	0.00	0.00	-4.00	-6.00	-0.00	20.00
w(x)	-10.00	1.10	1.08	-3.54	-6.27	-0.59	20.00

5. Metoda Galerkin

B.G. Galerkin determina valorile parametrilor c_1 , c_2 , ... , c_n ce definesc aproximatia w(x) (formula (5)) impunând ca erorile $E(x ; c_1,c_2,\ldots,c_n)$ (formula (7)) sa fie "ortogonale" pe functiile de referinta $f_j(x)$, $1 \le j \le n$. Precizam faptul ca functiile integrabile ce sunt folosite în procesul de aproximare sunt în acest caz interpretate drept "vectori" într-un spatiu euclidian, unde produsul scalar $< h_1 , h_2 >$ între oricare doi "vectori" h_1 , h_2 este definit prin

$$< h_1 , h_2 > = \int_a^b h_1(x) h_2(x)\, dx \qquad (33)$$

Intuitiv, conditia de "ortogonalitate" a vectorilor E(x) si $f_j(x)$, adica $< E , f_j > = 0$, $1 \le j \le n$, deriva din "principiul independentei erorilor" $E(x ; c_1,c_2,\ldots,c_n)$ în raport cu functiile $f_j(x)$ utilizate la obtinerea aproximatiei w(x) . Asadar

$$\int_a^b E(x;c_1,c_2,\ldots,c_n) f_j(x)\, dx = 0 \qquad \forall\ 1 \le j \le n \qquad (34)$$

Explicitând eroarea punctuala E dupa formula (7), din (32) se obtine în final un sistem liniar de n ecuatii, a k-a ecuatie a sistemului, $1 \le k \le n$, fiind de forma

$$c_1 \int_a^b L(f_1(x)) f_k(x)\,dx + c_2 \int_a^b L(f_2(x)) f_k(x)\,dx + \ldots + c_n \int_a^b L(f_n(x)) f_k(x)\,dx =$$

$$= \int_a^b \Big(h(x) - L(f_0(x))\Big) f_k(x)\,dx \qquad (35)$$

In urma rezolvarii sistemului liniar (35) se deduc valorile parametrilor $c_1, c_2, \ldots, c_n$ facând posibila precizarea aproximatiei w(x) dupa formula (5), functiile de referinta $f_j(x)$, $0 \le j \le n$, fiind cunoscute.

Exemplul 10. Vom determina prin metoda Galerkin aproximarea w(x) pentru solutia u(x) a ecuatiei diferentiale liniare definita în Exemplul 1. Astfel,pentru $n = 2$, sistemul liniar (35) este de forma (16), cu solutia (c_1, c_2) data de formulele (17), unde

$$a_{11} = \int_a^b f_1(x) L(f_1(x))\,dx \qquad a_{12} = \int_a^b f_1(x) L(f_2(x))\,dx$$

$$a_{21} = \int_a^b f_2(x) L(f_1(x))\,dx \qquad a_{22} = \int_a^b f_2(x) L(f_2(x))\,dx \qquad (36)$$

$$b_1 = \int_a^b f_1(x)\Big(h(x) - L(f_0(x))\Big)\,dx \qquad b_2 = \int_a^b f_2(x)\Big(h(x) - L(f_0(x))\Big)\,dx$$

cu $a = 0$, $b = 6$, functiile $h(x) - L(f_0(x))$, $L(f_j(x))$, $j = 1, 2$ având expresiile (14).

Aplicatia 5. Urmatorul program BASIC determina efectiv prin metoda Galerkin aproximatia w(x) (formula (5)) pentru solutiei u(x) a ecuatiei diferentiale liniare (2) cu conditiile la limita (3). Integralele ce definesc coeficientii a_{jk}, b_k sunt aproximate prin sume Riemann considerând o diviziune a intervalului [a , b] cu $m = 100$ puncte de diviziune echidistante ($a = 0$, $b = 6$).

```
REM  Metoda Galerkin
DECLARE FUNCTION h! (x!) :          DECLARE FUNCTION u! (x!)
DECLARE FUNCTION g0! (x!) :         DECLARE FUNCTION g1! (x!)
DECLARE FUNCTION g2! (x!) :         DECLARE FUNCTION f0! (x!)
DECLARE FUNCTION f1! (x!) :         DECLARE FUNCTION f2! (x!)
CLS :       m = 100 :       p = 6 / m
a11 = 0 :   FOR i = 1 TO m :   x = p * i :   a11 = a11 + f1(x) * g1(x) :   NEXT i
a12 = 0 :   FOR i = 1 TO m :   x = p * i :   a12 = a12 + f1(x) * g2(x) :   NEXT i
a21 = 0 :   FOR i = 1 TO m :   x = p * i :   a21 = a21 + f2(x) * g1(x) :   NEXT i
a22 = 0 :   FOR i = 1 TO m :   x = p * i :   a22 = a22 + f2(x) * g2(x) :   NEXT i
b1 = 0 :   FOR i = 1 TO m :   x = p * i :   b1 = b1 + f1(x) * (h(x) - g0(x)) :   NEXT i
b2 = 0 :   FOR i = 1 TO m :   x = p * i :   b2 = b2 + f2(x) * (h(x) - g0(x)) :   NEXT i
d = a11 * a22 - a12 * a21
c1 = (b1 * a22 - b2 * a12) / d :          c2 = (b2 * a11 - b1 * a21) / d
PRINT USING "u :   ######.## ; ######.## ; ######.## ; ######.## ; "; 1; -8; 17; -10
PRINT USING "w :   ######.## ; ######.## ; ######.## ; ######.## ; ";
```

```
c2; c1 - 6 * c2; 5 - 6 * c1; -10
PRINT "Valorile functiei u in punctele :";
FOR j = 0 TO 6 :        PRINT j; " , ";  :        NEXT j :        PRINT
FOR j = 0 TO 6 :        PRINT USING "#####.## ; "; u(j); :        NEXT j :        PRINT
PRINT "Valorile functiei w in punctele specificate"
FOR x = 0 TO 6 :        w = c2 * x * x * x + (c1 - 6 * c2) * x * x + (5 - 6 * c1) * x - 10
PRINT USING "#####.## ; "; w; :        NEXT x :        PRINT

FUNCTION f0 (x) :        f0 = 5 * x - 10 :        END FUNCTION
FUNCTION f1 (x) :        f1 = x * x - 6 * x :        END FUNCTION
FUNCTION f2 (x) :        f2 = x * x * x - 6 * x * x :        END FUNCTION
FUNCTION g0 (x) :        g0 = 5 * x ^ 2 + 25 * x - 25 :        END FUNCTION
FUNCTION g1 (x) :        g1 = 3 * x ^ 3 - 3 * x ^ 2 - 16 * x - 4 :        END FUNCTION
FUNCTION g2 (x) :        g2 = 5 * x ^ 4 - 15 * x ^ 3 - 15 * x ^ 2 - 6 * x - 12 :    END FUNCTION
FUNCTION h (x) :        h = 6 * x ^ 4 - 30 * x ^ 3 + 21 * x ^ 2 + 24 * x - 19 :    END FUNCTION
FUNCTION u (x) :        u = x ^ 3 - 8 * x ^ 2 + 17 * x - 10 :        END FUNCTION
```

Exemplul 11. In urma executarii procedurii Galerkin prezentata în Aplicatia 5 solutia u(x) a ecuatiei diferentiale liniare definita în Exemplul 1 cu conditiile la limita (8) a fost aproximata prin functia w(x) , unde

$u(x) = x^3 - 8x^2 + 17x - 10$ $\qquad$ $w(x) = 1.06\,x^3 - 8.54\,x^2 + 18.25\,x - 10.00$

Acuratetea aproximarii Galerkin poate fi apreciata urmarindu-se valorile din Tabelul 5.

Observatia 4. Prin compararea rezultatelor din Tabelele 3-5 se poate constata ca prin aplicarea metodei Galerkin rezultatele sunt mai precise decât în cazul folosirii metodei celor mai mici patrate (atât cazul discret cât si cazul continuu).

Tabelul 5. Valorile solutiei u(x) si ale aproximatiei sale w(x) obtinuta prin metoda Galerkin.

	x=0	x=1	x=2	x=3	x=4	x=5	x=6
u(x)	-10.00	0.00	0.00	-4.00	-6.00	-0.00	20.00
w(x)	-10.00	0.76	0.78	-3.62	-6.10	-0.34	20.00

6. Metoda schemelor cu diferente

Vom rezolva ecuatia diferentiala liniara (2) impunând diverse conditii la limita.

6.1. Conditii la limita - Varianta 1

Vom determina o aproximare w(x) a solutiei u(x) pentru ecuatia diferentiala (2) ce respecta conditiile la limita

$$u(a) = \gamma_1 \qquad u^{(1)}(a) = \gamma_2 \tag{37}$$

Conditiile initiale (37) difera de cele impuse în Exemplul 1.

In acest sens vom împarti intervalul [a , b] în m parti egale considerând punctele de diviziune echidistanta $x_j = a + \lambda j$, unde $0 \le j \le m$, $\lambda = (b - a) / m$. Vom utiliza notatia simplificatoare g_j , $0 \le j \le m$, pentru a desemna valoarea unei functii g , g : [a , b] → **R** , în punctul x_j , adica $g_j = g(x_j) = g(a+\lambda j)$

In capitolul referitor la estimarea derivatelor unei functii cu ajutorul schemelor cu diferente sunt sugerate urmatoarele aproximatii pentru derivatele $u^{(1)}(x)$ si $u^{(2)}(x)$,

$$u^{(2)}(x) \approx \frac{u(x-\lambda) - 2u(x) + u(x+\lambda)}{\lambda^2} \qquad (38)$$

$$u^{(1)}(x) \approx \frac{u(x+\lambda) - u(x-\lambda)}{2\lambda} \qquad u^{(1)}(x) \approx \frac{u(x+\lambda) - u(x)}{\lambda}$$

Utilizând aceste aproximari în ecuatia diferentiala liniara (2) pentru punctele x_j fixate, $1 \le j \le m-1$, deducem urmatoarea schema cu diferente

$$\frac{u_{j-1} - 2u_j + u_{j+1}}{\lambda^2} + p_j \frac{u_{j+1} - u_{j-1}}{2\lambda} + q_j u_j = h_j \qquad (39)$$

conditiile la limita (37) devenind

$$u_0 = \gamma_1 \qquad \frac{u_1 - u_0}{\lambda} = \gamma_2 \qquad (40)$$

Asadar

$u_0 = \gamma_1$ $\qquad u_1 = u_0 + \lambda \gamma_2$

$$u_{j+1} = \frac{4 - 2\lambda^2 q_j}{\lambda p_j + 2} u_j + \frac{\lambda p_j - 2}{\lambda p_j + 2} u_{j-1} + \frac{2\lambda^2 h_j}{\lambda p_j + 2} \qquad (41)$$

Relatiile recurente (41) permit construirea pas cu pas a sirului $u_0 , u_1 , u_2 , u_3 , \dots , u_{m-1} , u_m$ ce defineste aproximarea discreta w(x ; m) a solutiei u(x) calculata în punctele de diviziune $x_0 = a$, $x_1 , x_2 , x_3 , \dots , x_{m-1}$, respectiv $x_m = b$. Astfel $w(x_j ; m) = u_j$, $0 \le j \le m$.

Micsorând pasul de discretizare λ se asigura o mai mare precizie aproximarii w(x ; m).

Exemplul 12. Fie ecuatia diferentiala $u^{(2)}(x) + p(x) u^{(1)}(x) + q(x) u(x) = h(x)$, unde $p(x) = 2x^2 + 1$ $\quad q(x) = 3 - x$ $\quad h(x) = 6x^4 - 30x^3 + 21x^2 + 24x - 19$ $\quad a = 0$ $\quad b = 6$ cu conditiile la limita de tipul (37) , adica

$$u(0) = -10 \qquad u^{(1)}(0) = 17 \qquad (42)$$

Prin calcul direct se verifica faptul ca solutia acestei ecuatii cu conditiile la limita (42) este $u(x) = x^3 - 8x^2 + 17x - 10$, $x \in [0 , 6]$.

Aplicatia 6. Notam prin w(x ; m) valoarea aproximatiei în punctul x a solutiei u(x) , aproximatie obtinuta prin metoda schemelor cu diferente pentru m puncte de diviziune ale intervalului [0 , 6]. Urmatorul programul determina estimatia w(x ; m) în punctele $x_j = 6 j / m$, $0 \le j \le m$.

```
REM  Metoda schemelor cu diferente.  Varianta 1
DECLARE FUNCTION p! (x!) :          DECLARE FUNCTION q! (x!)
DECLARE FUNCTION h! (x!) : DECLARE FUNCTION u! (x!)
CLS :      INPUT "m = "; m :      d = 1 / m
u0 = -10:      u1 = u0 + 17 * d:      x = 0
PRINT "Numarul de intervale de diviziune = "; 6 * m
PRINT USING "x = ##.##      u = #####.##      w = #####.##"; x; u(x); u0
```

```
FOR k = 1 TO 6
FOR j = 1 TO m :      x = k - 1 + j * d :      px = p(x) :      qx = q(x) :      hx = h(x)
u2 = (4 - 2 * d * d * qx) * u1 + (d * px - 2) * u0 + 2 * d * d * hx
u2 = u2 / (d * px + 2) :      u0 = u1 :      u1 = u2 :      NEXT j
PRINT USING "x = ##.##      u = #####.##      w = #####.##"; x; u(x); u0
NEXT k

FUNCTION h (x) :    h = 6 * x ^ 4 - 30 * x ^ 3 + 21 * x ^ 2 + 24 * x - 19 :    END FUNCTION
FUNCTION p (x) :    p = 2 * x * x + 1 :      END FUNCTION
FUNCTION q (x) :    q = 3 - x :      END FUNCTION
FUNCTION u (x) :    u = x ^ 3 - 8 * x ^ 2 + 17 * x - 10 :      END FUNCTION
```

Exemplul 13. Vom rula programul proiectat în Aplicatia 6 pentru ecuatia diferentiala din Exemplul 12, impunând conditiile la limita (42).

In Tabelul 6 sunt listate valorile în punctele $x = 0, 1, \ldots, 6$ ale aproximatiilor $w(x ; m)$ ale lui $u(x)$ gasite prin metoda schemelor cu diferente pentru m puncte de diviziune ale intervalului $[0 , 6]$. Prin marirea numarului m de puncte ce divid intervalul $[0 , 6]$ se observa o îmbunatatire a acuratetii aproximarii $w(x ; m)$.

Utilizarea în programul BASIC a unei precizii duble nu modifica valorile din Tabelul 6 (ce au fost afisate cu doua zecimale).

Tabelul 6. Valorile solutiei $u(x)$ si aproximatia sa discreta $w(x ; m)$ obtinuta prin metoda schemelor cu diferente (conditii initiale : $u(0) = -10$, $u^{(1)}(0) = 17$)

	x=0	x=1	x=2	x=3	x=4	x=5	x=6
u(x)	-10.00	0.00	0.00	-4.00	-6.00	-0.00	20.00
w(x ; 60)	-10.00	1.12	0.89	-3.34	-5.87	-0.23	20.34
w(x ; 120)	-10.00	1.05	0.85	-3.37	-5.89	-0.25	20.34
w(x ; 180)	-10.00	1.03	0.84	-3.38	-5.90	-0.26	20.33
w(x ; 300)	-10.00	1.01	0.83	-3.39	-5.91	-0.27	20.32
w(x ; 600)	-10.00	1.00	0.82	-3.40	-5.92	-0.27	20.31
w(x ; 1200)	-10.00	0.99	0.81	-3.41	-5.92	-0.28	20.31
w(x ; 3000)	-10.00	0.99	0.81	-3.41	-5.92	-0.28	20.31

6.2. Conditii la limita - Varianta 2

Vom modifica conditiilor la limita (37) considerând restrictiile rezultate din (3) pentru $\alpha_1 = \beta_2 = 0$ si $\alpha_2 = \beta_1 = 1$, adica

$$u^{(1)}(a) = \gamma_1 \qquad u(b) = \gamma_2 \tag{43}$$

Conditiile la limita (43) sunt diferite de conditiile initiale (37) precum si de restrictiile initiale impuse solutiei ecuatiei diferentiale din Exemplul 1.

Din formula (39) cu conditiile initiale (43) deducem relatiile

$$(u_1 - u_0) / \lambda = \gamma_1 \qquad u_m = \gamma_m \qquad A_j u_{j-1} + B_j u_j + C_j u_{j+1} = h_j \ , \ 1 \le j \le m-1 \tag{44}$$

unde

$$A_j = \frac{1}{\lambda^2} - \frac{p_j}{2\lambda} \qquad B_j = q_j - \frac{2}{\lambda^2} \qquad C_j = \frac{1}{\lambda^2} + \frac{p_j}{2\lambda} \tag{45}$$

Vom determina coeficientii y_j , z_j astfel încât sa avem

$$u_j = y_j u_{j+1} + z_j \quad , \quad 0 \le j \le m-1 \tag{46}$$

Cum $u_0 = u_1 - \lambda \gamma_1$ rezulta $y_0 = 1$ si $z_0 = -\lambda \gamma_1$. Utilizând egalitatea $u_{j-1} = y_{j-1} u_j + z_{j-1}$ în relatia recurenta (44) obtinem $A_j (y_{j-1} u_j + z_{j-1}) + B_j u_j + C_j u_{j+1} = h_j$ adica

$$u_j = -\frac{C_j}{A_j y_{j-1} + B_j} u_{j+1} + \frac{h_j - A_j z_{j-1}}{A_j y_{j-1} + B_j}$$

Prin urmare

$$y_j = -\frac{C_j}{A_j y_{j-1} + B_j} \qquad z_j = \frac{h_j - A_j z_{j-1}}{A_j y_{j-1} + B_j} \qquad 1 \le j \le m-1 \tag{47}$$

Cum $y_0 = 1$ si $z_0 = -\lambda \gamma_1$, din egalitatile (47) se determina iterativ coeficientii $y_1 , z_1 , y_2 , z_2 , \dots , y_{m-1} , z_{m-1}$. Cunoscând $u_m = \gamma_2$ si utilizând formula (46) deducem sirul de valori $u_{m-1} , u_{m-2} , \dots , u_2 , u_1 , u_0$ ce caracterizeaza aproximatia discreta $w(x ; m)$ a lui $u(x)$ în punctele $x_j = a + j \lambda$, $0 \le j \le m$, unde $\lambda = (b - a) / m$. Avem $w(x_j ; m) = u_j$, $0 \le j \le m$.

Observatia 5. Metoda de determinare efectiva a elementelor sirului $\{ u_j \}_j$ aplicând tehnica schemelor cu diferente difera în cadrul celor doua variante analizate. In prima varianta termenii sirului $\{u_j\}_j$ rezultau direct din relatia iterativa (41).In varianta a doua aceasta operatie este mai complicata, precizându-se succesiv într-o prima etapa coeficientii y_j , z_j , $0 \le j < m$ (formula (47)). Ulterior se vor obtine recursiv si valorile termenilor u_j (relatia (46)).

Exemplul 14. Fie ecuatia diferentiala definita în Exemplul 12 cu conditiile la limita de tip (43),

$$u^{(1)}(0) = 17 \qquad u(6) = 20 \tag{48}$$

Prin calcul direct se verifica faptul ca solutia ecuatiei mentionate cu conditiile la limita (48) este $u(x) = x^3 - 8x^2 + 17x - 10$, $x \in [0 , 6]$.

Aplicatia 7. Vom desemna prin $w(x ; m)$ aproximatia discreta a solutiei $u(x)$ considerând $m+1$ puncte de diviziune $x_0 , x_1 , x_2 , \dots , x_m$. Asadar $w(x_j ; m) = u_j$, $0 \le j \le m$, sirul $\{ u_j \}_j$ fiind obtinut recursiv prin aplicarea formulei (46) unde coeficientii y_j , z_j au expresiile (47).

Prezentam în continuare programul pentru calculul aproximatiei $w(x_j ; m)$, $0 \le j \le m$, pentru $x_j = a + (b - a) j / m$.

```
REM  Metoda schemelor cu diferente.   Varianta 2
DECLARE FUNCTION p! (x!) :          DECLARE FUNCTION q! (x!)
DECLARE FUNCTION h! (x!) :          DECLARE FUNCTION u! (x!)
a = 0!:        b = 6!:        v1 = 17:        v2 = 20      ' Conditii initiale
CLS : INPUT "m = "; m :  OPTION BASE 0 :   DIM w(m), y(m), z(m) :   d = (b - a) / m
y(0) = 1 :        z(0) = -d * v1 :        w(m) = v2
PRINT "Numarul de intervale de diviziune = "; m
FOR j = 1 TO m - 1 :    x = a + j * d :    px = p(x) :    qx = q(x) :    hx = h(x)
aj = 1! / (d * d) - px / (2! * d) :    bj = -2! / (d * d) + qx :    cj = 1! / (d * d) + px / (2! * d)
y(j) = -cj / (aj * y(j - 1) + bj) :    z(j) = (hx - aj * z(j - 1)) / (aj * y(j - 1) + bj) :    NEXT j
FOR j = m - 1 TO 0 STEP -1 :    w(j) = y(j) * w(j + 1) + z(j) :    NEXT j
```

```
REM  Tiparirea rezultatelor
FOR x = 0 TO 6 :                j = (x - a) / d
PRINT USING "x = ##.##     u = #####.##     w = #####.##"; x; u(x); w(j) :     NEXT x

FUNCTION h (x) :  h = 6 * x ^ 4 - 30 * x ^ 3 + 21 * x ^ 2 + 24 * x - 19 :  END FUNCTION
FUNCTION p (x) :          p = 2 * x * x + 1 :          END FUNCTION
FUNCTION q (x) :          q = 3 - x :          END FUNCTION
FUNCTION u (x) :          u = x ^ 3 - 8 * x ^ 2 + 17 * x - 10 :          END FUNCTION
```

Observatia 6. In programul BASIC prezentat în Aplicatia 7 nu era nevoie retinerea întregului set de valori y_j , z_j , $0 \le j \le m-1$, ca elemente ale unor vectori. Astfel începând cu y_0 si z_0 , aplicând succesiv formulele (47) se determina în final valorile y_{m-1} si z_{m-1} . Pornind de la aceste ultime valori si utilizând din nou relatiile (47) poate fi refacut pas cu pas, în sens invers, întregul sir de coeficienti y_{m-2} , z_{m-2} , y_{m-3} , z_{m-3} , ... , y_2 , z_2 , y_1 , z_1 , y_0 , z_0 fara a fi nevoie retinerea lor în vectori. Totodata, prin aplicarea formulele (46), vor fi evaluate si aproximatiile u_j , j = m-1, m-2, ..., 3, 2, 1, 0 , ale solutiei u(x) .

Exemplul 15. Consideram ecuatia diferentiala liniara din Exemplul 12 cu conditiile la limita (48). Solutia u(x) a acestei ecuatii diferentiale, precizata în Exemplul 14, va fi aproximata prin functia discreta $w(x_j ; m)$ obtinuta prin rularea programului din Aplicatia 7.

In Tabelul 7 sunt prezentate comparativ, în functie de numarul m al punctelor de diviziune $x_j = a + (b - a)\, j / m$, $0 \le j \le m$, valorile u(x) si $w(x ; m)$ pentru $0 \le x \le 6$.

Efectuarea calculelor în dubla precizie nu a modificat rezultatele listate cu doua zecimale în Tabelul 7.

Tabelul 7. Valorile solutiei u(x) si aproximatia sa discreta $w(x ; m)$ obtinuta prin metoda schemelor cu diferente (conditii initiale : $u^{(1)}(0) = 17$, $u(6) = 20$)

	x=0	x=1	x=2	x=3	x=4	x=5	x=6
u(x)	-10.00	0.00	0.00	-4.00	-6.00	-0.00	20.00
w(x ; 50)	-12.05	0.16	0.45	-3.64	-6.19	-0.07	20.00
w(x ; 100)	-12.40	0.34	0.59	-3.68	-6.19	-0.81	20.00
w(x ; 200)	-12.53	0.14	0.49	-3.68	-6.21	-0.45	20.00
w(x ; 300)	-12.56	0.18	0.52	-3.69	-6.21	-0.57	20.00
w(x ; 500)	-12.59	0.16	0.50	-3.69	-6.21	-0.53	20.00
w(x ; 1000)	-12.60	0.19	0.52	-3.69	-6.21	-0.60	20.00

TEHNICI MONTE CARLO

Pentru studierea din punct de vedere teoretic a diverselor tehnici Monte Carlo este nevoie de cunostinte elementare de probabilitati si statistica matematica. In aceasta sectiune vom trata însa aceste aspecte din punct de vedere experimental, prezentând justificari intuitive si evitând, pe cât posibil, o abordare teoretica.

1. Siruri de valori aleatoare

Orice limbaj de programare (si nu numai) are o procedura specializata în generarea sirurilor de valori aleatoare ce au o repartitie uniforma pe intervalul [0 , 1]. In cazul limbajului BASIC generarea numerelor întâmplatoare uniform repartizate pe [0 , 1] este realizata de functia RND .

Procedura RND produce sirul $x_0 , x_1 , x_2 , x_3 , \dots , x_n , x_{n+1} , \dots$ folosind o metoda congruentiala de tipul $x_{n+1} = a\, x_n$ (modulo M). Asadar x_{n+1} este restul împartirii produsului $a\, x_n$ la M. Parametrii a si M ai generatorului au valori alese astfel încât termenii sirului $\{ x_n \}_{n \in N}$ sa fie "cât mai aleatori". De fapt sirul $\{ x_n \}_n$ astfel obtinut este un sir pseudoaleator.

Exemplul 1. Urmatorul program genereaza, la fiecare rulare , mereu aceeasi secventa de n valori aleatoare $x_1 , x_2 , \dots , x_n$ uniform repartizate pe intervalul [0 , 1].

```
REM  Generarea unei secvente de n valori aleatoare fara a se initializa generatorul  RND
INPUT "n = "; n
FOR j = 1 TO n :      x = RND :      PRINT "x("; j; ")="; x ; :      NEXT j
```

Rulând succesiv acest program pentru n = 3 , respectiv n = 5 , se observa repetarea secventei de valori aleatoare generata, primele trei numere pseudoaleatoare din secventele rezultate în urma celor doua rulari fiind identice,

```
n = ? 3
x(1)= .7055475      x(2)= .5334240      x(3)= .5795186
n = ? 5
x(1)= .7055475      x(2)= .5334240      x(3)= .5795186      x(4)= .2895625      x(5)= .301948
```

Datorita metodei deterministe folosite, sirul $\{ x_n \}_n$ este construit în mod unic atunci când este precizata valoarea initiala x_0 , valoare cunoscuta sub numele de "samânta generatorului".

Valoarea initiala x_0 poate ramâne mereu aceeasi la fiecare apelare a programului RND , aceasta în situatia în care nu se doreste o modificare a lui x_0 la fiecare noua rulare a modulului de generare. Initializarea "samântei" generatorului RND este realizata de procedura RANDOMIZE .

De obicei se utilizareaza varianta RANDOMIZE TIMER fapt ce antreneaza initializarea generatorului RND de numere aleatoare uniforme cu o valoare precizata de "ceasul calculatorului" în momentul apelarii procedurii RANDOMIZE .

In Exemplul 1 este produsa mereu aceeasi secventa de numere pseudoaleatoare datorita neinitializarii de catre programator a "samântei" generatorului RND. In acest caz initializarea functiei RND va fi realizata de produsul BASIC cu o valoare standard.

Exemplul 2. Spre deosebire de programul anterior în urmatoarea variantă s a inițializat generatorul de numere aleatoare cu "timpul curent al calculatorului " (prin comanda RANDOMIZE TIMER).

```
REM  Generarea unei secvente de n valori aleatoare
RANDOMIZE TIMER          ' S-a initializat generatorul  RND
INPUT "n = "; n
FOR j = 1 TO n :      x = RND :      PRINT "x("; j; ")="; x :      NEXT j
```

Rularile succesive ale acestui program conduc la siruri distincte de valori aleatoare De aceasta data primele trei numere ale fiecarei secvente de rezultate difera de la o rulare la alta

```
n = ? 3
x(1)= .3421289 :    x(2)= .2330548 :    x(3)= .6525014
n = ? 5
x(1)= .4273340 :    x(2)= .2274395 :    x(3)= .9193471 :    x(4)= .0044520 :    x(5)= .9541544
```

Observatia 1. De obicei, neinitializarea de catre programator a generatorului RND este practicata în faza de testare a unui program, atunci când se doreste de fapt repetarea identica a unor simulari în vederea urmaririi anumitor caracteristici.

Propozitia 1. Daca evenimentele elementare sunt egal probabile atunci putem estima probabilitatea realizarii unui eveniment oarecare A prin raportul dintre numarul cazurilor favorabile realizarii lui A si numarul cazurilor posibile (numarul tuturor cazurilor).

Propozitia 2. Daca U este o variabila aleatoare uniform repartizata în intervalul [0 , 1] atunci, pentru orice $0 \le a < b \le 1$, probabilitatea ca realizarile variabilei U sa fie în intervalul [a , b] este egala cu lungimea acestui interval, adica

$$Pr (a \le U \le b) = b - a \qquad 0 \le a < b \le 1 \qquad (1)$$

Aplicatia 1. Bazându-ne pe Propozitia 1 vom verifica din punct de vedere experimental calitatile statistice ale unui sir $\{ u_n \}_n$ produs de generatorul RND .

Putem evalua probabilitatea teoretica $P_t(a,b) = Pr (a \le u_i \le b , i \in \mathbf{N})$ generând cu algoritmul RND n valori aleatoare independente u_j , $1 \le j \le n$, si determinând numarul m al acelor termeni u_j cu valori în intervalul [a , b] .

Conform Propozitiei 1 valoarea $P_e(n,a,b) = Pr (a \le u_i \le b , 1 \le i \le n) = m / n$ obtinuta experimental estimeaza probabilitatea teoretica $P_t(a,b)$.

In cazul în care valorile u_j , $j \ge 1$, generate de RND ar fi realizari independente ale unei variabile aleatoare U uniform repartizata în intervalul [0 , 1] atunci este adevarata egalitatea $P_t(a,b) = Pr (a \le U \le b)$. Asadar, aplicând Propozitia 2 pentru $0 \le a < b \le 1$, rezulta $m / n \approx b - a$ fapt ce ar putea confirma indirect repartitia uniforma a sirului de valori $\{ u_n \}_n$ obtinut cu ajutorul functiei RND .

Vom exemplifica procedura de test descrisa considerând k intervale $[a_j , b_j]$, $1 \le j \le k$ nu neaparat disjuncte, unde $0 \le a_j \le b_j \le 1$. Pentru fiecare interval $[a_j , b_j]$ se va compara estimatia experimentala $P_e(n,a_j,b_j)$ cu probabilitatea teoretica $Pr (a_j \le U \le b_j)$

Urmatorul program BASIC utilizeaza n termeni ai sirului $\{ u_n \}_n$, generat cu functia RND pentru determinarea estimatiilor $P_e(n,a_j,b_j)$.

```
REM  Compararea probabilitatilor  Pr ( a ≤ U ≤ b ),   Pr ( a ≤ u_i ≤ b , i ∈ N )
INPUT "Numarul de intervale = "; k :                DIM a(k), b(k), m(k)
FOR j = 1 TO k
1 : PRINT "Intervalul "; j; :          INPUT "      a , b = "; a(j), b(j)
IF (a(j) < 0) OR (a(j) >= b(j)) OR (b(j) > 1) THEN GOTO 1
m(j) = 0 :          NEXT j
INPUT "Numarul de generari ="; n :          RANDOMIZE TIMER
FOR i = 1 TO n :          u = RND
FOR j = 1 TO k
IF (u >= a(j)) AND (u <= b(j)) THEN m(j) = m(j) + 1
NEXT j :          NEXT i
FOR j = 1 TO k
PRINT "Interval  [ "; a(j); " , "; b(j); " ]"; TAB(35); "n ="; n; TAB(50); "m ="; m(j); TAB(65); "b - a
="; b(j) - a(j); TAB(80); "m / n ="; m(j) / n :          NEXT j
```

Exemplul 3. In Tabelul 1 sunt prezentate rezultatele rularii programului prezentat în Aplicatia 1 pentru k = 6 intervale : [0.0 , 0.3] ; [0.1 , 0.6] ; [0.2 , 0.4] ; [0.3 , 0.8] ; [0.75, 0.85] ; [0.6 , 1.0] .

Cum, pentru intervalele [a , b] mentionate, estimatiile P_e (n,a,b) "aproximeaza bine" probabilitatile Pr (a ≤ U ≤ b) , nu putem confirma existenta unei repartitii neuniforme pentru sirul de valori $\{ u_n \}_n$ produs de functia RND .

Atunci când volumul selectiei creste de la n = 500 la n = 30000 estimatiile experimentale P_e (n,a,b) sunt mai precise, valorile lor fiind mai apropiate de probabilitatea Pr (a ≤ U ≤ b) . Acest rezultat este justificat din punct de vedere teoretic.

Tabelul 1. Verificarea repartitiei uniforme a sirului $\{ u_n \}_n$ generat cu RND.

a	b	Pr(a≤U≤b)	n	m	P_e (n,a,b)	n	m	P_e (n,a,b)
0.00	0.30	0.30	500	127	0.254	30000	9095	0.3032
0.10	0.60	0.50	500	260	0.520	30000	14963	0.4988
0.20	0.40	0.20	500	99	0.198	30000	5990	0.1997
0.30	0.80	0.50	500	269	0.538	30000	14833	0.4944
0.75	0.85	0.10	500	42	0.084	30000	2904	0.0968
0.60	1.00	0.40	500	193	0.386	30000	12003	0.4001

2. Evaluarea integralelor

Vom estima printr-o tehnica de tip Monte Carlo valoarea integralei I_1 ,

$$I_1 = \int_0^1 f_1(t)\, dt \qquad (2)$$

Utilizând cunostinte elementare de probabilitati si statistica matematica se poate demonsra :

Propozitia 3. Daca $u_1, u_2, u_3, \ldots, u_n$ sunt n realizari independente ale variabilei aleatoare U uniform repartizata peintervalul [0 , 1], atunci

$$\int_0^1 f_1(t)\,dt \approx A_1(f_1,n) = \frac{f_1(u_1) + f_1(u_2) + f_1(u_3) + \ldots + f_1(u_n)}{n} \tag{3}$$

Pentru a evalua integrala I_2 cu limite a , b finite , $I_2 = \int_a^b f_2(t)\,dt$, facem transformarea de variabile $x = a + (b - a)\,t$ si deci $I_2 = \int_0^1 (b-a) f_2(a+(b-a)t)\,dt$. In aceasta ultima forma integrala I_2 este de tipul (2) pentru $f_1(t) = (b - a)\, f_2(a + (b-a)\, t)$. Din Propozitia 3 rezulta

Corolarul 1. Daca $u_1, u_2, u_3, \ldots, u_n$ sunt n realizari independente ale variabilei aleatoare U uniform repartizata pe intervalul [0 , 1] atunci

$$\int_a^b f_2(x)\,dx \approx A_2(f_2,n,a,b) = \frac{b-a}{n}\sum_{j=1}^{n} f_2(a+(b-a)u_j) \tag{4}$$

Integrala $I_3 = \int_0^\infty f_3(x)\,dx$ este adusa la forma (2) prin utilizarea transformarii $x = 1 / t - 1$

Asadar $I_3 = \int_0^\infty f_3(x)\,dx = \int_0^1 \frac{f_3(1/t-1)}{t^2}\,dt$

Aplicând Propozitia 3 pentru functia $f_1(t) = \frac{f_3(1/t-1)}{t^2}$ deducem

Corolarul 2. Daca $u_1, u_2, u_3, \ldots, u_n$ sunt n realizari independente ale variabilei aleatoare U ce are o repartitie uniforma pe intervalul [0 , 1] atunci

$$\int_0^\infty f_3(x)\,dx \approx A_3(f_3,n) = \frac{1}{n}\sum_{j=1}^{n} \frac{f_3(1/u_j - 1)}{u_j^2} \tag{5}$$

Folosind transformarea de variabile $y = - x$ deducem $I_4 = \int_{-\infty}^0 f_4(y)\,dy = \int_0^\infty f_4(-x)\,dx$, expresia I_4 fiind o integrala de tipul I_3 pentru functia $f_3(x) = - f_4(-x)$. Asadar Corolarul 2 devine

Corolarul 3. Daca $u_1, u_2, u_3, \ldots, u_n$ sunt realizari independente ale variabilei aleatoare U uniform repartizata în domeniul [0 , 1] atunci

$$\int_{-\infty}^0 f_4(x)\,dx \approx A_4(f_4,n) = \frac{1}{n}\sum_{j=1}^{n} \frac{f_4(1-1/u_j)}{u_j^2} \tag{6}$$

Observatia 2. Folosind rezultatele anterioare pot fi evaluate integrale unidimensionale si pentru alte domenii de integrare. Spre exemplificare, pentru $a < 0$ avem

$$\int_{-\infty}^{a} f_5(x)\,dx = \int_{-\infty}^{0} f_5(x)\,dx - \int_{a}^{0} f_5(x)\,dx \approx A_4(f_5,n) - A_2(f_5,n,a,0)$$

$$\int_{a}^{\infty} f_6(x)\,dx = \int_{0}^{\infty} f_6(x)\,dx + \int_{a}^{0} f_6(x)\,dx \approx A_3(f_6,n) + A_2(f_6,n,a,0) \qquad (7a)$$

Daca a > 0 rezulta

$$\int_{-\infty}^{a} f_5(x)\,dx = \int_{-\infty}^{0} f_5(x)\,dx + \int_{0}^{a} f_5(x)\,dx \approx A_4(f_5,n) + A_2(f_5,n,0,a)$$

$$\int_{a}^{\infty} f_6(x)\,dx = \int_{0}^{\infty} f_6(x)\,dx - \int_{0}^{a} f_6(x)\,dx \approx A_3(f_6,n) - A_2(f_6,n,0,a) \qquad (7b)$$

Aplicatia 2. Urmatorul program BASIC estimeaza valorile integralelor I_1, I_2, I_3, I_4 considerând n realizari independente ale unei variabile aleatoare U ce are o repartitie uniforma pe intervalul [0 , 1].

```
DECLARE FUNCTION f1! (x!) :          DECLARE FUNCTION f2! (x!)
DECLARE FUNCTION f3! (x!) :          DECLARE FUNCTION f4! (x!)
REM  Estimarea integralelor prin metoda Monte Carlo
RANDOMIZE TIMER :      a = -2 :      b = 3 :      c = b - a
INPUT "n = "; n :      a1 = 0 :      a2 = 0 :      a3 = 0 :      a4 = 0
FOR j = 1 TO n :            u = RND
a1 = a1 + f1(u) :           a2 = a2 + f2(a + c * u)
a3 = a3 + f3(1! / u - 1) / (u * u) :          a4 = a4 + f4(1 - 1! / u) / (u * u)
NEXT j :      a1 = a1 / n :      a2 = c * (a2 / n) :      a3 = a3 / n :      a4 = a4 / n
PRINT USING "##### ; ##.### ; ##.### ; ##.### ; ##.###"; n; a1; a2; a3; a4
FUNCTION f1 (x) :           f1 = x ^ 4 :          END FUNCTION
FUNCTION f2 (x) :           f2 = x * x :          END FUNCTION
FUNCTION f3 (x) :           f3 = EXP(-2 * x) :          END FUNCTION
FUNCTION f4 (x) :           f4 = 1 / (1 - 2 * x) ^ 3 :          END FUNCTION
```

Exemplul 4. Fie functiile : $f_1(x) = x^4$, $0 \le x \le 1$; $f_2(x) = x^2$, $-2 \le x \le 3$; $f_3(x) = e^{-2x}$, $x \ge 0$; $f_4(x) = 1 / (1 - 2x)^3$, $x \le 0$, ale caror integrale I_1, I_2, I_3, I_4 le vom evalua pe întreg domeniul lor de definitie. Notând prin $g_k(x)$ o primitiva a functiei $f_k(x)$, pentru $1 \le k \le 4$ avem respectiv :

$g_1(x) = x^5 / 5$, $0 \le x \le 1$; $g_2(x) = x^3 / 3$, $-2 \le x \le 3$; $g_3(x) = -0.5\, e^{-2x}$, $x \ge 0$;

$g_4(x) = 1 / [\, 4 (1 - 2x)^2 \,]$, $x \le 0$. Asadar valoarea exacta a integralelor I_k, $1 \le k \le 4$, este egala cu

$I_1 = g_1(1) - g_1(0) = 1 / 5$; $I_2 = g_2(3) - g_2(-2) = 35 / 3 \approx 11.6667$;

$I_3 = g_3(\infty) - g_3(0) = 1 / 2$; $I_4 = g_4(0) - g_4(-\infty) = 1 / 4$

In Tabelul 2 sunt listate aproximatiile $A_1(f_1,n)$, $A_2(f_2,n,-2,3)$, $A_3(f_3,n)$, $A_4(f_4,n)$ date de expresiile (3)-(6) pentru integralele I_k, $1 \le k \le 4$ utilizând un numar variabil n de realizari u_j, $1 \le j \le n$, ale unei variabile U uniform distribuita pe [0 , 1].

Precizia de estimare se îmbunatateste odata cu utilizarea mai multor valori aleatoare u_j pentru evaluarea integralelor I_k. Acest lucru este demonstrat teoretic si justificat experimental în Tabelul 2 Prin cresterea volumului n al esantionului $\{ u_j \}_j$ valorile experimentale A_1, A_2, A_3, A_4 se apropie mai mult de valorile teoretice $I_1 = 1 / 5$, $I_2 = 35 / 3 \approx 11.6667$, $I_3 = 1 / 2$, respectiv $I_4 = 1 / 4$

Procedura Monte Carlo descrisa poate fi cu usurinta adaptata si pentru evaluarea integralelor în cazul multidimensional.

Tabelul 2. Aproximarea integralelor I_k prin expresiile A_k, $1 \le k \le 4$, date de formulele (3)-(6) în raport cu numarul n de realizari u_j (Exemplul 4)

n	A_1	A_2	A_3	A_4	n	A_1	A_2	A_3	A_4
50	0.288	15.597	0.611	0.333	100	0.213	11.416	0.545	0.266
200	0.228	12.587	0.540	0.278	300	0.203	11.773	0.505	0.253
500	0.205	12.084	0.501	0.254	1000	0.191	11.286	0.487	0.241
2000	0.201	11.715	0.504	0.251	3000	0.204	11.859	0.500	0.254
5000	0.196	11.525	0.494	0.246	10000	0.200	11.712	0.496	0.249

3. Estimarea volumelor

Tehnicile Monte Carlo pot fi aplicate cu succes la evaluarea ariilor si volumelor. Vom exemplifica o astfel de procedura în cele ce urmeaza.

Fie paralelipipedul $D_1 [a_1 , b_1] \times [a_2 , b_2] \times [a_3 , b_3] \subset \mathbf{R}^3$ ce include corpul D_2 ($D_2 \subset D_1$) al carui volum intentionam sa-l estimam.

In acest sens vom genera n puncte aleatoare $P_j (x_j , y_j , z_j)$, $1 \le j \le n$, uniform repartizate în paralelipipedul D_1 si vom contoriza cele $m \le n$ puncte ce cad în domeniul D_2. Datorita repartitiei uniforme a punctelor P_j, se poate demonstra teoretic ca m / n estimeaza raportul volumelor $vol (D_2) / vol (D_1)$. Asadar

Propozitia 4. O estimatie E_0 a volumului corpului D_2 este data de formula

$$vol (D_2) \approx E_0 (n) = (m / n) \, vol (D_1) = m (b_1 - a_1) (b_2 - a_2) (b_3 - a_3) / n \qquad (9)$$

Propozitia 5. Daca U_1, U_2, U_3 sunt variabile aleatoare independente uniform repartizate în intervalul [0 , 1] atunci $P (X, Y, Z)$ este un punct aleator uniform distribuit în paralelipipedul D_1 unde $X = a_1 + (b_1 - a_1) U_1$, $Y = a_2 + (b_2 - a_2) U_2$, $Z = a_3 + (b_3 - a_3) U_3$

Algoritmul **AMCEV** bazat pe formula (9) si Propozitia 5 propune o estimatie $E_0 (n)$ a volumului subdomeniului D_2 inclus în paralelipipedul D_1.

Algoritmul AMCEV (Algoritm Monte Carlo de Estimarea Volumului D_2)

Pas 0. Precizeaza numarul n de generari si paralelipipedul $D_1 = [a_1 , b_1] \times [a_2 , b_2] \times [a_3 , b_3]$

$m = 0$ $\quad$ $j = 0$

Pas 1. $j = j + 1$

Genereaza u_1 , u_2 , u_3 realizari independente ale unei variabile aleatoare U ce are o repartitie uniforma în intervalul [0 , 1]

$x = a_1 + (b_1 - a_1)\, u_1$; $y = a_2 + (b_2 - a_2)\, u_2$; $z = a_3 + (b_3 - a_3)\, u_3$

Pas 2. Daca $(x, y, z) \in D_2$ atunci $m = m + 1$

Daca $j < n$ atunci Mergi la Pasul 1

Pas 3. $E_0 = m\,(b_1 - a_1)\,(b_2 - a_2)\,(b_3 - a_3) / n$ STOP

Aplicatia 3. Fie $D_1 \equiv [\,-a\,,\,a\,] \times [\,-b\,,\,b\,] \times [\,-c\,,\,c\,]$, domeniul D_2 fiind elipsoidul de ecuatie $x^2 / a^2 + y^2 / b^2 + z^2 / c^2 - 1 = 0$, adica $D_2 = \{\, (x, y, z) \mid x^2 / a^2 + y^2 / b^2 + z^2 / c^2 \le 1 \,\}$ Evident $D_2 \subset D_1$.

Urmatorul program constituie o implementare în limbajul BASIC a algoritmului **AMCEV** pentru $a_1 = -a$, $b_1 = a$, $a_2 = -b$, $b_2 = b$, $a_3 = -c$, $b_3 = c$.

```
REM  Estimarea volumului D2 a unui elipsoid
INPUT "n, a, b, c = "; n, a, b, c :          pi = 3.1415965# :          m = 0
FOR j = 1 TO n
x = a * (2 * RND - 1) :          y = b * (2 * RND - 1) :          z = c * (2 * RND - 1)
IF  ((x * x) / (a * a) + (y * y) / (b * b) + (z * z) / (c * c)) <= 1  THEN  m = m + 1
NEXT j :                    v = 8 * a * b * c * m / n
PRINT USING "Volum real = ####.###       Volum estimat ####.### = "; 4 * pi * a * b * c / 3; v
```

Vom calcula valoarea exacta a volumului elipsoidului D_2 .

Cu notatiile $\lambda(x) = (\,1 - x^2 / a^2\,)^{1/2}$, $\mu(x,y) = (\,1 - x^2 / a^2 - y^2 / b^2\,)^{1/2}$ avem

$$vol(D_2) = \int_{-a}^{a} \left(\int_{-b\,\lambda(x)}^{b\,\lambda(x)} \left(\int_{-c\,\mu(x,y)}^{\mu(x,y)} dz \right) dy \right) dx = 2c \int_{-a}^{a} \left(\int_{-b\,\lambda(x)}^{b\,\lambda(x)} \mu(x,y)\, dy \right) dx =$$

$$= 2bc \left[\int_{-a}^{a} \left(1 - \frac{x^2}{a^2} \right) dx \right] \left[\int_{-1}^{1} \sqrt{1 - t^2}\, dt \right] = \frac{4\pi abc}{3} \qquad (10)$$

Din formula (9) deducem estimatia $E_0(n)$ a volumului D_2 inclus în paralelipipedul $D_1 \equiv [\,-a\,,\,a\,] \times [\,-b\,,\,b\,] \times [\,-c\,,\,c\,]$, anume

$$E_0(n) = 8\, a\, b\, c\, m / n \qquad (11)$$

Exemplul 5. Fie domeniile D_1 , D_2 definite în Aplicatia 3 , unde $a = 1$, $b = 2$, $c = 3$. Din (10) se obtine valoarea exacta a volumului elipsoidului D_2 , anume $vol\,(D_2) = 8\,\pi = 25.133$. Formula (11) defineste estimatia $E_0(n) = 48\, m / n$ pentru volumul $vol\,(D_2)$, numarul m fiind determinat prin simulare Monte Carlo (executând algoritmul **AMCEV**).

Teoretic se demonstreaza ca odata cu cresterea numarului n de simulari creste si precizia $E_0(n)$ de evaluare a volumului elipsoidului D_2 . Acest aspect este evidentiat si de rezultatele experimentale $E_0(n)$ sintetizate în Tabelul 3 pentru diferite valori ale numarului n de simulari.

Se vor compara valorile experimentale $E_0(n)$ cu valoarea teoretica $vol\,(D_2) = 25.133$.

Tabelul 3. Estimatiile $E_0(n)$ date de formula (11) pentru vol (D_2) (Exemplul 5)

n	$E_0(n)$	n	$E_0(n)$	n	$E_0(n)$	n	$E_0(n)$	n	$E_0(n)$
1000	25.344	2000	24.840	3000	25.232	4000	24.492	5000	24.941
6000	25.120	7000	25.481	8000	25.062	9000	24.933	10000	24.509
11000	25.370	12000	24.944	13000	25.606	14000	25.275	15000	25.162
16000	25.422	17000	24.788	18000	25.363	19000	25.281	20000	25.066
21000	25.058	22000	25.241	23000	24.858	24000	24.986	25000	25.279
26000	25.098	27000	25.141	28000	25.399	29000	25.210	30000	25.138

4. Simulare stohastica

Vom sugera o posibila utilizare a simularii stohastice la solutionarea unei probleme importante din geologie, anume estimarea rezervei de minereu util dintr-un zacamânt.

In practica sunt cunoscute mai multe procedee de evaluarea cantitatii medii q_0 de minereu prezent într-un domeniu $D_2 \subset \mathbf{R}^3$, atunci când, prin masuratori, s-au obtinut continuturile q_j de minereu în n puncte date $P_j(x_j, y_j, z_j)$, $1 \le j \le n$. De fapt procedeele de evaluarea rezervei de minereu din zona D_2 precizeaza, în functie de structura zacamântului, ponderile λ_j, $\lambda_j \ge 0$, $1 \le j \le n$, de influenta a continuturilor q_j asupra continutului mediu q_0 din domeniul D_2, unde

$$q_0 = (\lambda_1 q_1 + \lambda_2 q_2 + \lambda_3 q_3 + \ldots + \lambda_n q_n) / (\lambda_1 + \lambda_2 + \lambda_3 + \ldots + \lambda_n)$$

Este dificil de a se evalua teoretic precizia acestor metode de estimare a rezervei de substanta minerala utila prin utilizarea unor procedee clasice. Acest aspect poate fi însa solutionat experimental prin simularea pe calculator a unor date de intrare q_j cu caracteristici statistice precizate. Valorile q_j vor fi atasate unor puncte $P_j(x_j, y_j, z_j)$, $1 \le j \le n$, aleatoare în domeniul D_2. Pentru datele create artificial prin simulare stohastica se vor aplica metodele traditionale de determinarea continutului mediu q_0. In urma unui studiu statistic comparativ privind rezultatele q_0 deduse în cadrul unui numar mare de simulari se va aprecia în final eficienta si precizia fiecarui procedeu de estimare a continutului mediu de substanta minerala utila din domeniul D_2.

In vederea producerii unor puncte aleatoare P(x,y,z) uniform repartizate în domeniul D_2 putem utiliza metoda clasica de "respingere". Astfel vom genera puncte aleatoare uniform distribuite într-un paralelipiped D_1 ce-l include pe D_2, $D_2 \subset D_1$, si vom retine numai punctele P ce apartin lui D_2, respingând toate celelalte puncte (a se vedea si Propozitia 5) .

Simularea trebuie sa puna în evidenta si caracteristicile zacamântului respectiv.

Astfel continutul q de minereu într-un punct arbitrar P(x,y,z) are o componenta determinista f(x,y,z) atasata "tendintei" de concentrare spatiala a minereului în zacamânt precum si o componenta întâmplatoare h(U), unde, de exemplu, variabila aleatoare U are o repartitie uniforma pe intervalul [0 , 1]. Asadar continutul q de minereu din punctul P(x,y,z) va fi simulat aleator dupa formula q = f(x,y,z) + h(U), valorile întâmplatoare U fiind produse cu calculatorul prin apelarea functiei RND .

BIBLIOGRAFIE

1. Milton Abramowitz, Irene A Stegun (Editors) : Handbook of mathematical functions with formulas, graphs and mathematical tables. Dover Publications, New York, 1964.

2. Kendall Atkinson : Elementary numerical analysis. John Wiley & Sons, New York, 1985

3. Sara Baase : Computer algorithms - Introduction to design and analysis. Addison Wesley, New York, 1988 (second edition)

4. N. Bakhvalov : Méthodes numeriques : Analyse, algèbre, équations différentielles ordinaires. Editions MIR, Moscou, 1976.

5. Paul Bratley, Bennett L. Fox, Linus E. Schrage : A guide to simulation. Springer Verlag, New York, 1983.

6. Richard I. Burden, J Douglas Faires : Numerical analysis. PWS - KENT Publishing Company, Boston, 1985 (third edition)

7. A. M. Cohen, J.F. Cutts, R. Fielder, D.E. Jones, J. Ribbans, E. Stuart : Numerical analysis McGraw Hill, ondon, 1973.

8. Ioan Cuculescu : Analiza numerica. Editura Tehnica, Bucuresti, 1967.

9. B.P. Demidovich, I.A. Maron : Computational mathematics. MIR Publishers, Moscow, 1981.

10. Tim Denvir : Introduction to discrete mathematics for software engineering. MacMillan Publishing Company, New York, 1987.

11. Gheorghe Dodescu : Metode numerice în algebra. Editura Tehnica, Bucuresti, 1979

12. Horia Dumitrascu : Sa învatam Basic. Editura Albatros, Bucuresti, 1987.

13. Liviu Dumitrascu : Invatam totul despre BASIC. Editura Tehnica, Bucuresti, 1989 (vol 1+2)

14. S.M. Ermakov : Metoda Monte Carlo si probleme înrudite. Editura Tehnica, Bucuresti 1976.

15. Valeriu Iorga, Boris Jora, Cristina Nicolescu, Ionut Lopatan, Ion Fatu : Programare numerica. Editura Teora, Bucuresti, 1996.

16. Donald E. Knuth : Tratat de programare a calculatoarelor - Sortare si cautare. Editura Tehnica, Bucuresti, 1976.

17. Donald E. Knuth : Tratat de programare a calculatoarelor - Algoritmi seminumerici. Editura Tehnica, Bucuresti, 1983.

18. Horia Georgescu, Octavian Bâsca : Programe în limbajul FORTRAN. Editura Albatros, Bucuresti, 1975.

19. Mark Steven Heyman : Bazele Visual Basic 4 . Teora, Bucuresti, 1996.

20. D.V. Ionescu : Quadraturi numerice. Editura Tehnica, Bucuresti, 1957.

21. Eugene Isaacson, Herbert Keller : Analysis of numerical methods. John Wiley and Sons, New York, 1966.

22. Dan Larionescu : Metode numerice, Editura Tehnica, Bucuresti, 1989.

23. Leon Livovschi, Horia Georgescu : Sinteza si analiza algoritmilor. Editura Stiintifica si Enciclopedica, Bucuresti, 1986.

24. G.I. Marciuk : Metode de analiza numerica. Editura Academiei, Bucuresti, 1983.

25. Gheorghe Marinescu : Analiza numerica. Editura Academiei, Bucuresti, 1974.

26. Miron Nicolescu, Nicolae Dinculeanu, Solomon Marcus : Analiza matematica. Editura Didactica si Pedagogica, Bucuresti 1966 (vol.1), 1977 (vol. 1 + 2), 2 vol. (editia III, 1980 ; editia V , 1989).

27. William Press, Brian Flannery, Saul Teukolsky, William Vetterling : Numerical recipes. Cambridge University Press, Cambridge. 1986.

28. N. Racoveanu, Gh. Dodescu, I. Mincu : Metode numerice pentru ecuatii cu derivate partiale de tip parabolic. Editura Tehnica, Bucuresti, 1977.

29. Lennart Rade, Bertil Westergren : Beta, Mathematics Handbook : Concepts, theorems, methods, algorithms, formulas, graphs, tables. Sudentlitteratur, Chartwell-Bratt, Lund, 1990.

30. Anthony Ralston, Philip Rabinowitz : A first course in numerical analysis. McGraw Hill, New York, 1978 (second edition).

31. Mario G. Salvadori, Melvin L. Baron : Metode numerice în tehnica. Editura Tehnica, Bucuresti, 1972.

32. Francis Scheid : Analyse numérique - Cours et problèmes . Serie Schaum, Groupe McGraw Hill, Paris, 1986.

33. G.E. Silov : Analiza matematica - Spatii finit dimensionale -. Editura Stiintifica si Enciclopedica, Bucuresti, 1983.

34. Stefan Stefanescu : Erori de calcul la evaluarea unei balante. Studii si Cercetari de Calcul Economic si Cibernetica Economica, nr. 2, 1999.

35. Stefan Stefanescu : Asupra convergentei fractiilor continue. Studii si Cercetari de Calcul Economic si Cibernetica Economica, 1999.

36. Stefan Stefanescu : Stil de programare în proiectarea aplicatiilor. Contract CNCSU, preprint, 1998.

37. Stefan Stefanescu : Erori de programare si tratarea lor. Contract CNCSU, preprint, 1998.

38. Lucian Vasiu : Aplicatii în Q - BASIC. Editura Tehnica, Bucuresti, 1994.

39. Adrian I. Vlad : Programe de instruire BASIC. Editura Stiintifica si Enciclopedica, Bucuresti, 1989.

40. E.A. Volkov : Numerical methods. Mir Publishers, Moscow, 1982.

41. Camelia Zaharia, Marian Zaharia : Sa învatam sa programam. Editura Tehnica, Bucuresti, 1992

42. ***** Microsoft : GW - BASIC interpreter. User's guide and user's reference, Minta, 1990.

ISBN 973-575-435-5

Lei 31000