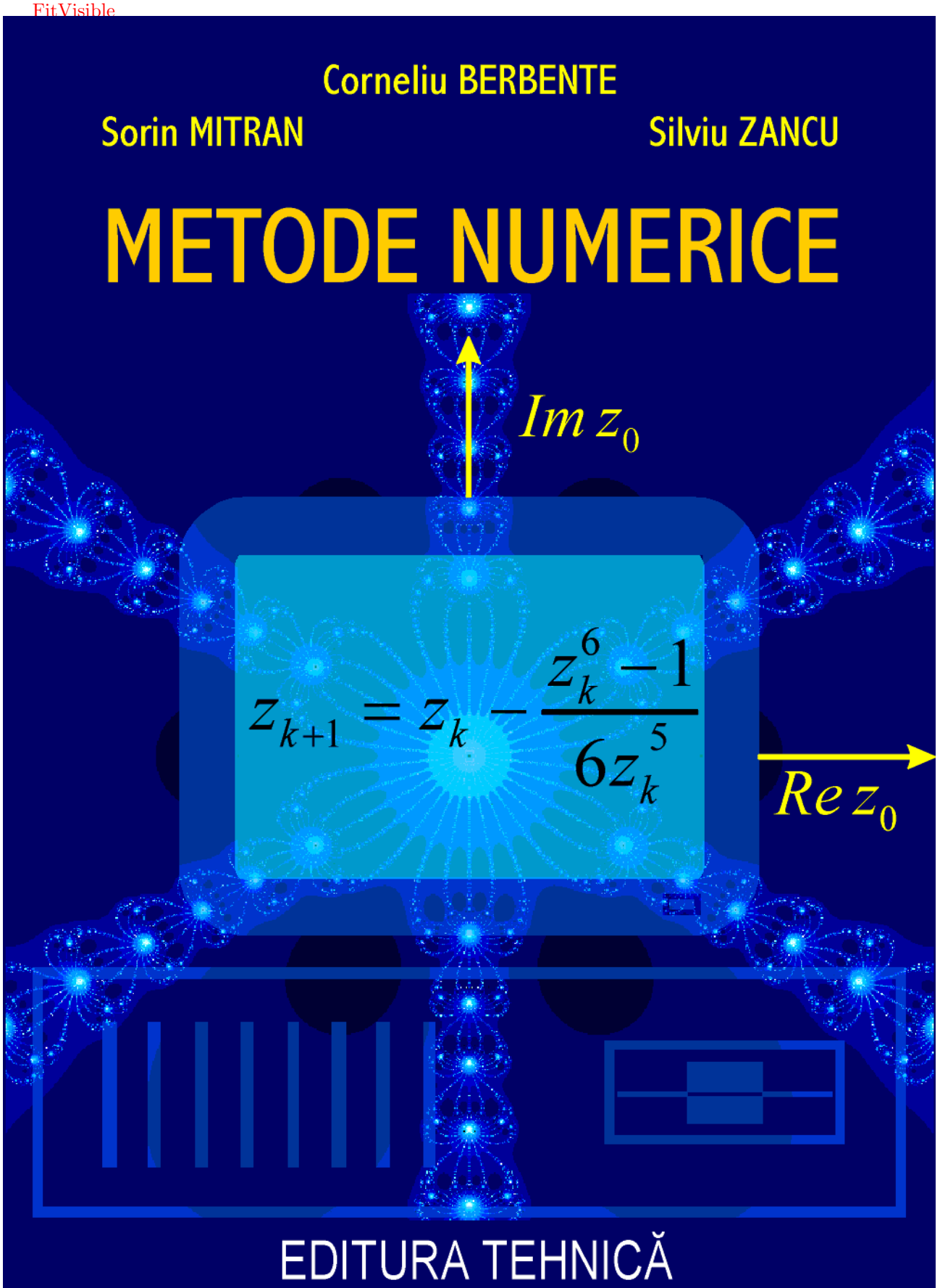
FitVisible

Corneliu BERBENTE
Sorin MITRAN
Silviu ZANCU

METODE NUMERICE



EDITURA TEHNICĂ

Aceasta este versiunea electronică a cărţii *Metode Numerice* publicată de Editura Tehnică. Cartea a fost culeasă folosind sistemul LaTeX a lui Leslie Lamport, o extindere a programului TeX scris de Donald Knuth. Versiunea electronică foloseşte formatul Portable Document Format (PDF) elaborat de Adobe Systems. Traducerea formatului LaTeX în PDF a fost realizată cu programul `pdflatex` elaborat de Han The Thanh. Hiperlegăturile din versiunea electronică au fost generate automat folosind pachetul `hyperref` al lui Sebastian Rahtz.

COPYRIGHT ©1998, Corneliu Berbente, Sorin Mitran, Silviu Zancu

Toate drepturile asupra ediţiei electronice sunt rezervate autorilor. Nu este permisă tipărirea conţinutului acestei ediţii fără consimţământul scris al autorilor.

COPYRIGHT ©1997, Editura Tehnică

Toate drepturile asupra ediţiei tipărite sunt rezervate editurii.

Adresa: EDITURA TEHNICĂ
Piaţa Presei Libere, 1
33 Bucureşti, România
cod 71341

Redactor: ing. Vasile Buzatu
Tehnoredactor: Diana Jilavu
Coperta: Sorin Mitran

Bun de tipar: 15.11.1997; Coli tipo: 17,75
CZU: 519.6
ISBN 973-31-1135-X

PREFAŢĂ

Extraordinara dezvoltare a tehnicii de calcul din ultima perioadă permite simularea unui tot mai mare număr de probleme fizice, inginereşti sau economice. În paralel, a avut loc o dezvoltare a programelor disponibile fizicianului, inginerului sau economistului, oferindu-le o bogată gamă de algoritmi pentru rezolvarea unor aplicaţii concrete. Utilizarea acestei bogăţii de tehnici şi informaţii necesită însă, o bază teoretică solidă pentru a fi efectiv folosite.

Reprezentarea printr-un număr finit de cifre semnificative a numerelor în calculator introduce dificultăţi extrem de mari în asigurarea condiţiilor pentru aplicarea unora din noţiunile fundamentale ale matematicilor moderne, legate de procesul de trecere la limită, amendând astfel utilizarea eficientă a unor teoreme de bază din analiză. În schimb, se introduc erorile de rotunjire a căror propagare, în interacţie cu alte tipuri de erori (inerente sau de metodă) este greu de urmărit. Prinre consecinţe, se poate întâmpla ca varainate echivalente teoretic (spre exemplu pe baza unor teoreme privind unicitatea soluţiei) să ducă, numeric, la rezultate foarte diferite. Ca urmare, este explicabilă tendinţa de a se descoperi noi şi noi formule de calcul numeric, chiar dacă în esenţă (matematic) acestea diferă foarte puţin.

Această carte prezintă o viziune detaliată asupra teoriei şi practicii metodelor numerice, rod al unei activităţi de aproape 20 de ani în acest domeniu. Algoritmii cei mai utilizaţi sunt prezentaţi integral. O serie de algoritmi avansaţi, de largă aplicabilitate sunt de asemenea incluşi. Autorii au încercat o prezentare intuitivă a teoriei ce stă la baza metodelor numerice considerate, urmărindu-se mai mult uşurinţa înţelegerii materialului. Locul demonstraţiilor riguroase – de altfel dificile şi nu întotdeauna eficiente didactic – e luat, într-o serie de cazuri, de observaţii critice şi de "bun simţ". O observaţie de "bun simţ" este şi aceea de a face apel la mai multă teorie atunci când modalităţile cunoscute au fost epuizate. Ca atare, se vor regăsi în carte şi o serie de cunoştinţe mai avansate necesare dezvoltării unor metode numerice performante.

Sunt incluse capitole privind: aproximarea funcţiilor, derivarea şi integrarea numerică, problemele algebrei liniare, ecuaţii şi sisteme de ecuaţii neliniare, optimizare, ecuaţii diferenţiale. În prezentarea algoritmilor s-a optat pentru folosirea unui meta-limbaj, asemănător celui din programul Matlab. Cititorul poate transcrie un algoritm în limbajul de programare preferat cu uşurinţă. Pentru a preîntimpina cererile unor utilizatori ce doresc programe sursă sau direct executabile, cartea este suplimentată de un bogat material oferit pe Internet la adresa `http://www.propulsion.pub.ro`. *La acest sit se pot regăsi implementări în Pascal, FORTRAN şi C++ ale celor mai utilizaţi algoritmi, exemple extinse, legături la alte situri de pe Internet de interes pentru analiza numerică. Cei cu acces la Internet pot beneficia de programele de instruire asistată de calculator ce sunt disponibile la acest sit, unde este disponibilă o versiune electronică a*

acestei cărţi, o serie de lucrări de laborator şi numeroase aplicaţii mici ce pot fi rulate direct din browser-ul preferat.

Pe tot parcursul prezentării, elementele teoretice sunt completate cu numeroase exemple detaliat rezolvate. Acestea provin din cele mai variate domenii: ingineria mecanică, ingineria electrică, fizică şi chimie. S-a încercat formularea unor exemple iniţiale simple, ce să se concentreze pe aspectele strict numerice, iar apoi, a unor exemple apropriate problemelor reale. Se speră ca această modalitate de prezentare să fie utilă atât studentului cât şi practicianului metodelor numerice.

1997 *Autorii*

Cuprins

1 Aproximarea funcţiilor de o variabilă **1**
1.1 Aproximarea prin interpolare 2
1.1.1 Interpolarea polinomială globală 3
1.1.2 Interpolare cu funcţii spline 10
1.1.3 Interpolare cu funcţii trigonometrice 16
1.1.4 Interpolare în planul complex 23
1.2 Aproximarea mini-max 30
1.2.1 Polinoamele Cebâşev 30
1.2.2 Minimizarea erorii la interpolarea polinomială 32
1.2.3 Aproximarea aproape mini-max a unei funcţii 34
1.3 Aproximarea în sensul celor mai mici pătrate 36
1.4 Elemente de teoria aproximării 40
1.4.1 Spaţii vectoriale 41
1.4.2 Produsul scalar şi ortogonalitate 42
1.4.3 Norme, operatori şi funcţionale 47
1.4.4 Problema generală a celei mai bune aproximări 49

2 Derivarea şi integrarea numerică **53**
2.1 Derivarea numerică 53
2.1.1 Derivate folosind polinoame de interpolare 54
2.1.2 Formularea operatorială 57
2.1.3 Polinoame de interpolare în funcţie şi derivată 59
2.1.4 Derivate folosind funcţii spline 61
2.1.5 Derivate folosind diverse aproximaţii 61
2.2 Integrarea numerică 62
2.2.1 Formule Newton-Cotes închise 63
2.2.2 Formule de integrare deschise 68
2.2.3 Tehnici de atingere a unei precizii impuse 76

3 Rezolvarea ecuaţiilor neliniare **85**
3.1 Metoda înjumătăţirii intervalelor 86
3.2 Procedee iterative 87
3.2.1 Iteraţia simplă 87
3.2.2 Metoda Newton-Raphson 89

3.2.3 Metoda secantei . . . 91
3.2.4 Metoda parabolelor tangente . . . 93
3.3 Determinarea rădăcinilor polinoamelor . . . 96
3.3.1 Metoda Lobacevschi-Graeffe . . . 96
3.3.2 Metode de factorizare a polinoamelor . . . 101

4 Erorile de calcul numeric 109
4.1 Surse de erori . . . 109
4.2 Propagarea erorilor în calcule . . . 111

5 Rezolvarea sistemelor liniare 115
5.1 Metode directe . . . 116
5.1.1 Metoda eliminării a lui Gauss . . . 116
5.1.2 Metoda Gauss-Jordan . . . 121
5.1.3 Propagarea erorilor la metodele de eliminare. Rafinarea soluţiei . . . 121
5.1.4 Interpretarea matriceală a metodelor de eliminare . . . 124
5.1.5 Calculul matricei inverse . . . 125
5.1.6 Relaţia Sherman-Morisson . . . 128
5.1.7 Rafinarea matricei inverse . . . 129
5.1.8 Efectele erorilor din datele iniţiale . . . 131
5.1.9 Factorizarea $L \cdot U$ . . . 132
5.1.10 Descompunerea SVD . . . 134
5.1.11 Sisteme cu matrice rare . . . 135
5.2 Metode iterative . . . 138
5.2.1 Metoda iterativă Jacobi . . . 138
5.2.2 Metoda iterativă Gauss-Seidel . . . 140
5.2.3 Accelerarea convergenţei metodelor iterative . . . 143
5.3 Comparaţii între metode . . . 145
5.4 Elemente de calcul matriceal . . . 146

6 Vectori şi valori proprii 151
6.1 Elemente introductive . . . 151
6.2 Metode pentru câteva valori proprii . . . 152
6.2.1 Metoda puterii directe . . . 152
6.2.2 Metoda puterii inverse . . . 155
6.2.3 Metoda deplasării . . . 156
6.3 Determinarea tuturor valorilor şi vectorilor proprii . . . 156
6.4 Metoda Danilevschi . . . 157
6.5 Metodele QR şi LR . . . 162
6.5.1 Rezultate teoretice preliminarii . . . 163
6.5.2 Algoritmi auxiliari . . . 169
6.5.3 Formularea metodelor QR şi LR . . . 171
6.5.4 Reducerea numărului de operaţii la factorizare . . . 172
6.5.5 Accelerarea metodelor QR şi LR . . . 175
6.5.6 Calculul vectorilor proprii . . . 176

7 Metode de optimizare **181**
7.1 Minimizarea în lungul unei direcţii . . . 183
7.2 Metode de minimizare fără calculul derivatelor . . . 187
7.3 Metoda gradientului . . . 190
7.4 Metoda Newton . . . 193
7.5 Metode cvasi-Newton . . . 196
7.6 Metode de gradient conjugat . . . 198
7.6.1 Rezolvarea sistemelor de ecuaţii liniare folosind metode de optimizare . . . 200
7.7 Metode specifice de optimizare . . . 204
7.8 Probleme de optimizare cu restricţii . . . 205

8 Rezolvarea sistemelor neliniare **213**
8.1 Iteraţia simplă . . . 214
8.2 Metoda iterativă Newton . . . 216
8.3 Metode cvasi-Newton . . . 219
8.4 Metoda gradientului . . . 221
8.5 Metoda hibridă . . . 223

9 Rezolvarea ecuaţiilor diferenţiale **229**
9.1 Consideraţii generale . . . 229
9.2 Metode cu paşi separaţi . . . 230
9.2.1 Formule Euler . . . 230
9.2.2 Formule Runge-Kutta . . . 233
9.2.3 Formule Runge-Kutta-Gill . . . 234
9.2.4 Alegerea pasului la rezolvarea ecuaţiei diferenţiale . . . 235
9.3 Extrapolare Richardson. Metoda Bulirsch-Stoer . . . 238
9.4 Metode cu paşi legaţi . . . 239
9.4.1 Formule explicite . . . 239
9.4.2 Formule implicite . . . 241
9.5 Propagarea erorilor. Stabilitate. . . . 243
9.6 Sisteme de ecuaţii diferenţiale. Ecuaţii de ordin superior . . . 247
9.6.1 Probleme cu valori iniţiale . . . 248
9.6.2 Probleme cu valori la limite . . . 249
9.6.3 Ecuaţii diferenţiale de ordin superior . . . 254
9.7 Sisteme cu scări disparate . . . 255

10 Ecuaţii diferenţiale cu derivate parţiale **263**
10.1 Ecuaţii cu derivate parţiale de ordinul I . . . 264
10.2 Ecuaţii cu derivate parţiale de ordinul II . . . 270
10.2.1 Ecuaţii cu derivate parţiale de tip parabolic . . . 273
10.2.2 Ecuaţii cu derivate parţiale de tip eliptic . . . 287
10.2.3 Ecuaţii cu derivate parţiale de tip hiperbolic . . . 295
10.2.4 Metoda caracteristicilor . . . 297
10.2.5 Scheme cu diferenţe finite . . . 300

Capitolul 1

Aproximarea funcţiilor de o variabilă

Problema aproximării unei funcţii de o variabilă se poate pune în situaţii diverse, următoarele două fiind mai frecvente:

1. funcţia este cunoscută, dar are o formă complicată, dificil de manipulat în calcule (spre exemplu pentru operaţii de derivare, integrare, etc.);

2. funcţia nu este complet cunoscută, fiind date numai valorile ei pe o mulţime discretă şi finită de puncte.

În primul caz, aproximarea se poate face, în principiu, oricât de exact, restricţiile fiind legate de condiţia ca funcţia care aproximează să fie cât mai simplă. În al doilea caz informaţiile sunt reduse şi se completează cu presupuneri suplimentare, privind gradul de regularitate al funcţiei (continuitatea funcţiei şi a derivatelor sale, etc.). În ambele cazuri, este importantă alegerea unui *criteriu de aproximare*.

Fie $[a, b] \subset \mathbf{R}$ un interval pe dreapta reală şi x_i, $i \in \{1, 2, \ldots, N\}, N \in \mathbf{N}$, o reţea de puncte de diviziune ale acestui interval, $x_i \in [a, b]$, $x_1 = a, x_N = b$. Punctele de diviziune se numesc *noduri*. Presupunem date valorile în noduri ale

unei funcţii reale f

$$y_i = f(x_i),\ i \in \overline{1, N}\,. \tag{1.1}$$

Notăm cu $g(x)$ funcţia cu care vrem să aproximăm pe $f(x)$ pe intervalul dat. Iată câteva criterii de aproximare:

a) *Interpolare.* În acest caz, funcţia "mai simplă" $g(x)$ este determinată din condiţia ca să ia aceleaşi valori în noduri

$$g(x_i) = y_i,\ i \in \overline{1, N}\ . \tag{1.2}$$

Criteriul de aproximare prin interpolare presupune tacit că nodurile (x_i, y_i) sunt cunoscute exact. Dacă din diverse motive – cel mai adesea datorită unui procedeu de măsurare – nodurile sunt afectate de erori atunci criteriul de interpolare este inadecvat.

b) *Minimizarea abaterii maxime.* Se impune condiţia ca abaterea maximă să fie minimă pe intervalul ales, adică

$$\max_{x\in[a,b]} |f(x) - g(x)| = \text{minim}. \tag{1.3}$$

Relaţia (1.3) are analogul discret

$$\max_{i\in\overline{1,N}} |y_i - g(x_i)| = \text{minim}. \tag{1.4}$$

Aproximarea făcută pe baza criteriului de mai sus se numeşte *aproximare mini-max.*

c) *Minimizarea sumei pătratelor abaterilor în noduri.* În acest caz se impune ca

$$S = \sum_{i=1}^{n} \left(y_i - g(x_i)\right)^2 = \text{minim}\ . \tag{1.5}$$

Se observă că, în cazul interpolării, această sumă este chiar nulă, adică are cea mai mică valoare posibilă. Totuşi, această observaţie nu face superfluu criteriul (1.5) care este mai general şi permite tratarea datelor cunoscute incert, aşa cum se va vedea mai departe. Metoda care foloseşte acest criteriu este întâlnită sub numele de *metoda celor mai mici pătrate.*

1.1 Aproximarea prin interpolare

Presupunând că nodurile x_i sunt *distincte*, condiţia de interpolare (1.1) reprezintă un sistem de N condiţii şi va duce în general la un sistem de N ecuaţii cu N necunoscute. Considerentele de simplitate amintite mai sus ne sugerează

c-alegerea formei funcţiei de aproximare să fie făcută astfel încât sistemul de condiţii să conducă la ecuaţii liniare. O posibilă alegere este următoarea: se ia un set de N funcţii simple, cunoscute, $g_k(x)$, $k \in \{1, 2, ..., N\}$ şi un set de N parametrii nedeterminaţi (scalari) a_k, $k \in \{1, 2, ..., N\}$, în funcţie de care se scrie aproximanta $g(x)$

$$g(x) = \sum_{k=1}^{N} a_k g_k(x) \ . \tag{1.6}$$

Deoarece nu ne-am înscris într-un formalism riguros, vom face unele observaţii de "bun simţ". Astfel, am ales N parametri nedeterminaţi a_k, deoarece avem N condiţii. Pe de altă parte, setul de funcţii $g_k(x)$ trebuie să conţină elemente distincte, astfel încât introduse în forma (1.6), numărul de parametri să nu se reducă[1]. Într-un limbaj mai riguros, se spune că cele N funcţii cunoscute g_k trebuie să fie *liniar independente*. În lipsa altei analize, ne putem limita la funcţii despre care ştim că au această proprietate. Un astfel de set îl reprezintă monoamele x^{k-1}, $k \in \{1, 2, ..., N\}$, în care caz funcţia de interpolare este un polinom de gradul $N - 1$

$$g(x) = \sum_{k=1}^{N} a_k x^{k-1}. \tag{1.7}$$

Alte seturi de funcţii pot fi funcţiile trigonometrice, exponenţiale, etc., pe care le vom trata ulterior.

1.1.1 Interpolarea polinomială globală

Revenind la forma (1.7) se pune problema găsirii coeficienţilor a_k din condiţia de interpolare, adică a rezolvării sistemului de ecuaţii liniare

$$\sum_{k=1}^{N} a_k x^{k-1} = y_i, \ i \in \overline{1, N} \ . \tag{1.8}$$

Dacă N este mare, rezolvarea sistemului (1.8) este dificilă sau cel puţin neconvenabilă. În orice caz, nodurile x_i fiind distincte, sistemul de ecuaţii (1.8) este un sistem cu determinant Vandermonde diferit de zero şi are o soluţie unică, bine determinată pentru coeficienţii a_k. În consecinţă, oricare ar fi calea pe care se construieşte efectiv polinomul de interpolare (1.7), acesta este *unic* pentru o funcţie şi o diviziune dată. Aproximarea efectuată este o interpolare *globală* în sensul că se foloseşte un singur polinom pe tot intervalul $[a, b]$.

Forma Newton a polinomului de interpolare.

O modalitate convenabilă de construcţie a polinomului de interpolare îl constituie *polinomul lui Newton cu diferenţe divizate*. Fiind date perechile de puncte

[1] Spre exemplu, dacă $g_2 = \alpha g_1, \alpha \neq 0$ atunci $a_1 g_1 + a_2 g_2 = (a_1 + \alpha a_2) g_1$, deci în loc de doi parametri independenţi a_1 şi a_2 ar apare doar o combinaţie $a_1' = a_1 + \alpha a_2$.

(x_i, y_i), se introduc următoarele rapoarte denumite *diferenţe divizate* (DD)

$$\mathrm{DD}(x_2, x_1) = \frac{y_2 - y_1}{x_2 - x_1}\,,$$

$$\mathrm{DD}(x_3, x_2, x_1) = \frac{\mathrm{DD}(x_3, x_2) - \mathrm{DD}(x_2, x_1)}{x_3 - x_1}\,,$$

$$\ldots$$

$$\mathrm{DD}(x_N, x_{N-1}, \ldots, x_1) = \frac{\mathrm{DD}(x_N, \ldots, x_2) - \mathrm{DD}(x_{N-1}, \ldots, x_1)}{x_N - x_1}\,. \tag{1.9}$$

Diferenţele divizate care se construiesc folosind $k+1$ puncte se numesc *diferenţe divizate de ordinul* k. Se poate demonstra prin inducţie matematică, următoarea expresie a diferenţei divizate de ordinul $N-1$:

$$\mathrm{DD}(x_N, x_{N-1}, \ldots, x_1) = \sum_{i=1}^{N} y_i \text{Œ} \prod_{j=1}^{N}{}' (x_i - x_j)\,. \tag{1.10}$$

Semnul $'$ denotă omiterea factorului $j = i$ din produs. Relaţia (1.10) fiind simetrică, rezultă că valoarea diferenţei divizate nu depinde de ordinea în care luăm punctele x_i. Din punct de vedere practic, este mai comod ca diferenţele divizate să nu se calculeze cu formula (1.10) ci recursiv, alcătuindu-se un tabel (Tab. 1.1).

Funcţia de aproximare $g(x)$ este un polinom de gradul $N-1$, pe care îl vom nota cu $p_{N-1}(x)$. Vom scrie

$$f(x) = p_{N-1}(x) + R_{N-1}(x)\,, \tag{1.11}$$

unde $R_{N-1}(x)$ este *restul* sau *eroarea de aproximare* la interpolarea polinomială. Pe de altă parte, ţinând seama de definiţia diferenţelor divizate, se poate scrie

$$\begin{gathered} f(x) = y_1 + (x - x_1)\,\mathrm{DD}(x, x_1) = \\ y_1 + (x - x_1)\,\mathrm{DD}(x_2, x_1) + (x - x_1)(x - x_2)\,\mathrm{DD}(x, x_2, x_1)\,. \end{gathered} \tag{1.12}$$

În relaţia (1.12) ultima diferenţă divizată este formată cu punctul curent x. Continuând procedeul (1.12) până se iau în consideraţie toate nodurile, rezultă

$$\begin{gathered} p_{N-1}(x) = y_1 + (x - x_1)\,\mathrm{DD}(x_2, x_1) + (x - x_1)(x - x_2)\,\mathrm{DD}(x, x_2, x_1) \; + \\ \ldots + (x - x_1)(x - x_2) \cdot \ldots \cdot (x - x_{N-1})\,\mathrm{DD}(x_N, x_{N-1}, \ldots, x_1)\,, \end{gathered} \tag{1.13}$$

$$R_{N-1} = \prod_{i=1}^{N}(x - x_i)\;\; \mathrm{DD}(x, x_N, x_{N-1}, \ldots, x_1)\,. \tag{1.14}$$

Se verifică direct din (1.14) că restul se anulează în noduri ($R_{N-1}(x_i) = 0,\ i = 1, 2, ..., N$) şi deci $p_{N-1}(x)$ dat de (1.13) este polinomul de interpolare corespunzător celor N puncte date. Forma (1.13) se numeşte *polinomul lui Newton cu diferenţe divizate.*

Pentru scrierea polinomului (1.13) se alcătuieşte mai intâi tabloul diferenţelor divizate de diverse ordine. Nu este necesar ca tabloul să cuprindă toate diferenţele divizate posibile. Este suficient să procedăm în ordinea nodurilor, din aproape în aproape, conform definiţiilor (1.9). Un exemplu de calcul este prezentat în tabelul 1.1 unde s-a notat cu DD_i diferenţa divizată de ordinul $i\,(i = 1, 2, 3)$. Polinomul obţinut este

$$p_3(x) = 2 + (x-1)\cdot(1) + (x-1)(x-2)\cdot(-2) + (x-1)(x-2)(x-3)\cdot(1)\,.$$

În ceea ce priveşte restul $R_{N-1}(x)$ (eroarea la interpolare), se poate face o evaluare dacă avem informaţii suplimentare referitoare la funcţia aproximată $f(x)$ şi la derivatele sale. În acest scop, considerăm funcţia ajutătoare $Q(t)$ definită prin relaţia

$$Q(t) = f(t) - p_{N-1}(t) - \prod_{i=1}^{N}(t - x_i)\ \ \mathrm{DD}(x, x_N, x_{N-1}, \ldots, x_1)\ . \tag{1.15}$$

Se observă că funcţia $Q(t)$ se anulează pentru $t = x$ şi $t = x_i, i = 1, 2, ...N$, adică are $N+1$ zerouri. Presupunând că $f(t)$ este derivabilă de un număr convenabil de ori, putem aplica funcţiei $Q(t)$ şi derivatelor sale teorema lui Rolle. Rezultă, succesiv, că derivata $Q'(t)$ are cel puţin N zerouri, derivata $Q''(t)$ are cel puţin $N-1$ zerouri ş.a.m.d., astfel că derivata de ordinul N are cel puţin un zero pe intervalul (a, b). Fie $t = \xi$ acest zero. Derivând relaţia (1.15) de N ori, şi făcând $t = \xi$, se obţine

$$f^{(N)}(\xi) = N! \cdot \mathrm{DD}(x, x_N, x_{N-1}, \ldots, x_1)\ ,$$

relaţie din care putem deduce expresia diferenţei divizate de ordinul N în funcţie de derivata de ordinul N. În acest fel, restul la interpolare (1.14) capătă forma

$$R_{N-1}(x) = \prod_{i=1}^{N}(x - x_i)\ f^{(N)}(\xi)/N!\,. \tag{1.16}$$

Prezenţa produselor $\prod(x - x_i)$, sugerează că restul este mai mic (în modul) când punctul curent x este centrat pe intervalul care conţine diviziunea şi mai mare când x este luat spre marginile intervalului sau în afara lui – acest ultim caz este denumit *extrapolare*. Deoarece derivata de ordinul N a funcţiei în punctul ξ nu este accesibilă (din diverse motive), evaluări ale restului se pot face presupunând că, la schimbarea diviziunii, punctul ξ (necunoscut) nu se deplasează mult, astfel încât derivata respectivă să fie aproximativ constantă, şi refăcând calculul pentru o nouă diviziune a intervalului se poate testa astfel şi sensibilitatea erorii la includerea de noi puncte de interpolare.

Tabelul 1.1: Un tabel de diferenţe divizate

x_i	y_i	DD_1	DD_2	DD_3	DD_4
1	2	–	–	–	–
2	3	1	–	–	–
3	0	-3	-2	–	–
5	6	3	2	1	–
4	4	2	1	-3/2	-5/6

De regulă, este *nerecomandabilă utilizarea unui număr mare de noduri* la interpolare, mai ales dacă se intenţionează calcularea unor derivate cu ajutorul acestui polinom. Uzual, nu se folosesc toate cele N noduri, ci doar 3-5 noduri cele mai apropriate de punctul în care se cere valoarea funcţiei. În consecinţă, există posibilitatea unor variante chiar când nu putem îndesi reţeaua prin alegerea altor noduri.

Interpolarea polinomială apare deseori ca o componentă a altor algoritmi numerici cum ar fi integrarea sau derivarea numerică a unor funcţii. În aceste aplicaţii se consideră deseori cazul *diviziunilor egale*

$$x_{i+1} - x_i = h,\ i = 1, 2, \ldots, N-1\,,$$

h fiind *pasul reţelei*. Se introduc operatorii Δ şi ∇ denumiţi *diferenţă la dreapta* respectiv, *diferenţă la stânga*, prin relaţiile

$$\Delta f(x) = f(x+h) - f(x)\,, \tag{1.17}$$

$$\nabla f(x) = f(x) - f(x-h)\ . \tag{1.18}$$

Rezultatul aplicării operatorului Δ sau ∇ asupra lui $f(x)$ se numeşte *diferenţă finită* (de ordinul I). Pentru n întreg, se defineşte un operator de translaţie, E, prin relaţia

$$E^n f(x) = f(x+nh), \quad n \in \mathbf{Z}\,. \tag{1.19}$$

Avem $E^1 f(x) = f(x+h)$, $E^0 f(x) = f(x)$, $E^{-1} f(x) = f(x-h)$. Se observă că între operatorii Δ, ∇ şi E există relaţiile

$$\Delta = E - E^0,\ \nabla = E^0 - E^{-1}\ . \tag{1.20}$$

Diferenţele divizate se pot exprima în funcţie de diferenţele finite şi de pasul h

$$\mathrm{DD}(x_2, x_1) = \left[f(x_1+h) - f(x_1)\right]/h = \left[\Delta f(x_1)\right]/h\,, \tag{1.21}$$

$$\mathrm{DD}(x_N, x_{N-1}) = \left[f(x_N) - f(x_N - h)\right]/h = \left[\nabla f(x_N)\right]/h\ . \tag{1.22}$$

Prin inducţie, se demonstrează uşor că

$$\mathrm{DD}(x_N, x_{N-1}, \ldots, x_1) = \frac{\Delta^{N-1} f(x_1)}{(N-1)! h^{N-1}} = \frac{\nabla^{N-1} f(x_N)}{(N-1)! h^{N-1}}\ , \tag{1.23}$$

unde exponentul indică aplicarea repetată a operatorului.

Punând variabila curentă x sub forma

$$x = x_i + \alpha h,\ \alpha \in [0, N-1]\,, \tag{1.24}$$

se poate obţine *expresia polinomul de interpolare Newton cu diferenţe finite la dreapta*

$$p_{N-1}(x) = y_1 + \alpha\Delta y_1 + \tfrac{1}{2}\alpha(\alpha-1)\Delta^2 y_1 + \ldots + C_\alpha^{N-1}\Delta^{N-1}y_1, \tag{1.25}$$

unde C_α^k, $k = 0, 1, \ldots, N-1$ sunt coeficienţii binomiali. Restul $R_{N-1}(x)$ capătă forma

$$R_{N-1}(x_1 + \alpha h) = h^N C_\alpha^N\, f^{(N)}(\xi)\ . \tag{1.26}$$

Calculul se face alcătuind un tablou al diferenţelor finite, similar cu tabloul diferenţelor divizate.

În mod asemănător, notând

$$x = x_N + \beta h,\ \beta \in [-N+1, 0]\,, \tag{1.27}$$

se obţin expresii cu *diferenţe la stânga*

$$p_{N-1}(x) = y_N + \beta\nabla y_N + \tfrac{1}{2}\beta(\beta+1)\nabla^2 y_N + \ldots + (-1)^{N-1}C_{-\beta}^{N-1}\nabla^{N-1}y_N$$

$$R_{N-1}(x_N + \beta h) = (-1)^N h^N C_\beta^N\, f^{(N)}(\xi)\ . \tag{1.28}$$

Forma Lagrange a polinomului de interpolare.

Polinomul de interpolare Lagrange se scrie alegând funcţiile $g_k(x)$ din relaţia (1.6) sub forma unor polinoame denumite *polinoame Lagrange* şi notate cu $\mathcal{L}_k(x)$, $k = 1, 2, ..., N$. Aceste polinoame au expresiile

$$\mathcal{L}_k(x) = \prod_{j=1}^{N}{}' \frac{x - x_j}{x_k - x_j},\ k \in \overline{1, N}\,, \tag{1.29}$$

unde produsul $\prod'$ se ia pentru $j \neq k$. Se observă direct din (1.29) că

$$\mathcal{L}_k(x_j) = 0 \text{ dacă } x_j \neq x_k;\ \mathcal{L}_k(x_j) = 1 \text{ dacă } x_j = x_k\ . \tag{1.30}$$

Polinomul de interpolare Lagrange se scrie

$$p_{N-1}(x) = \sum_{k=1}^{N} y_k \mathcal{L}_k(x)\ , \tag{1.31}$$

deci coeficienţii a_k din expresia (1.6) sunt chiar valorile funcţiei $f(x)$ în noduri, $a_k = y_k$. Se verifică direct, ţinând cont de proprietăţile (1.30) că $p_{N-1}(x_i) = y_i$, $i = 1, 2, ..., N$.

EXEMPLU. Utilizând primele patru noduri din tabelul 1, polinoamele Lagrange sunt

$$\mathcal{L}_1(x) = \frac{(x-2)(x-3)(x-5)}{(1-2)(1-3)(1-5)} = -\frac{1}{8}(x-2)(x-3)(x-5),$$

$$\mathcal{L}_2(x) = \frac{(x-1)(x-3)(x-5)}{(2-1)(2-3)(2-5)} = \frac{1}{3}(x-1)(x-3)(x-5),$$

$$\mathcal{L}_3(x) = \frac{(x-1)(x-2)(x-5)}{(3-1)(3-2)(3-5)} = -\frac{1}{4}(x-1)(x-2)(x-5),$$

$$\mathcal{L}_4(x) = \frac{(x-1)(x-2)(x-3)}{(5-1)(5-2)(5-3)} = \frac{1}{24}(x-1)(x-2)(x-3),$$

iar polinomul de interpolare este

$$p_3(x) = 2\mathcal{L}_1(x) + 3\mathcal{L}_2(x) + 0\mathcal{L}_3(x) + 6\mathcal{L}_4(x) = x^3 - 8x^2 + 18x - 9\,,$$

identic cu polinomul obţinut prin metoda diferenţelor divizate. Aducerea polinomului la forma canonică a fost făcută numai pentru compararea celor două metode, nefiind în general necesară în calculul numeric.

Convergenţa interpolării polinomiale globale.

Expresiile restului $R_{N-1}(x)$ obţinute la interpolarea polinomială sugerează o creştere a preciziei atunci când numărul N de noduri creşte. Spre exemplu, în cazul diviziunilor egale, expresiile (1.26) şi (1.28) indică proporţionalitatea abaterii cu h^N (h fiind pasul diviziunii) dar şi cu alţi factori cum ar fi derivata de ordinul N. Interpolarea ar fi *convergentă* dacă atunci când numărul de puncte de interpolare creşte indefinit $N \to \infty$, restul scade oricât de mult $R_{N-1} \to 0$. Se pune întrebarea: este interpolarea polinomială întotdeauna convergentă? Răspunsul la această întrebare este *negativ*. Încă din 1901, Runge a dat exemplul funcţiei

$$f(x) = 1/(1+x^2),\ x \in [-5,5]\ .$$

Se poate verifica faptul că $|R_{N-1}(x)| \to \infty$ când $N \to \infty$, diviziunile intervalului $[-5,5]$ fiind luate egale. Comportarea interpolării pentru $N = 6$ şi $N = 11$ este redată în figura 1.1. Mai general, acest rezultat negativ se exprimă prin *teorema lui Faber* care spune că pentru orice diviziune a intervalului $[a,b]$ există o funcţie, chiar continuă, faţă de care abaterea polinomului de interpolare creşte oricât de mult când $N \to \infty$. Faptul că există sigur cel puţin o funcţie pentru care interpolarea polinomială globală nu converge reduce aplicabilitatea practică a procedeului, acesta folosindu-se îndeosebi ca o componentă a altor algoritmi numerici, pentru valori mici ale lui N.

Figura 1.1: Polinoamele de interpolare $p_{N-1}(x)$ cu $N = 6, 11$ suprapuse peste $f(x)$.

Aspecte practice ale interpolării polinomiale.

Evident, polinoamele Newton şi Lagrange diferă numai prin formă, pentru aceeaşi reţea restul fiind acelaşi în ambele cazuri. Din punct de vedere al calculului numeric, este preferată folosirea polinomului Newton ce necesită un număr de operaţii aritmetice mai mic, de $O(3N^2/2)$ faţă de $O(4N^2)$ pentru polinomul Lagrange. Necesarul de memorie este acelaşi pentru ambii algoritmi. Pentru polinomul Newton ar părea că este nevoie de o matrice suplimentară pentru tabelul de diferenţe divizate. Însă din tabelul de diferenţe divizate se folosesc efectiv doar N coeficienţi existând posibilitatea refolosirii celulelor de memorie precum în algoritmul[2]

$$\begin{array}{l} d \leftarrow y \\ j = 2 : N \\ \quad \left[\begin{array}{l} k = N : -1 : j \\ \quad [d_k \leftarrow (d_k - d_{k-1})/(x_k - x_{k-j-1}) \end{array} \right. \end{array} \tag{1.32}$$

în urma căruia diferenţele divizate de pe diagonala tabelului se obţin în vectorul d ce a fost iniţializat cu ordonatele y. Aceasta este partea cea mai costisitoare a interpolării Newton necesitând $O(3N^2/2)$ operaţii aritmetice. Evaluarea polinomului într-un punct u se face eficient prin schema lui Horner

$$\begin{array}{l} S \leftarrow d_N \\ j = (N-1) : -1 : 1 \\ \quad [S \leftarrow d_j + (u - x_j) \cdot S \end{array} \tag{1.33}$$

[2] Am folosit notaţia Matlab $j = j_{ini} : pas : j_{fin}$ pentru bucle: variabila j este iniţializată la valoarea j_{ini} şi apoi incrementată cu pas. Instrucţiunile din buclă, delimitate de [se execută repetat până când $j > j_{fin}$ Dacă pas nu este precizat, precum în $j = j_{ini} : j_{fin}$, se subînţelege $pas = 1$.

Figura 1.2: Aproximarea funcţiei lui Runge $f(x)$ printr-o linie poligonală $s(x)$.

necesitând numai $O(3N)$ operaţii. Evaluarea polinomului Lagrange într-un punct u necesită $O(4N^2)$ prin

$$
\begin{array}{l}
S \leftarrow 0 \\
k = 1 : N \\
\quad \left\lceil \begin{array}{l} P \leftarrow 1 \\ j = 1 : k-1 \\ \quad [P \leftarrow P \cdot (u - x_j)/(x_k - x_j) \end{array} \right. \\
\quad \left\lfloor \begin{array}{l} j = k+1 : N \\ \quad [P \leftarrow P \cdot (u - x_j)/(x_k - x_j) \\ S \leftarrow S + y_k \cdot P \end{array} \right.
\end{array}
$$

1.1.2 Interpolare cu funcţii spline

Am văzut că interpolarea polinomială *globală*, pe tot intervalul $[a, b]$, nu converge întotdeauna. Desigur, dacă am reuşi să micşorăm diviziunea fără a modifica gradul polinomului de interpolare, rezultatul ar putea fi modificat. Spre exemplu, aproximarea unei funcţii derivabile cu o linie poligonală se poate face oricât de bine când numărul de laturi ale poligonului creşte infinit (fig. 1.2). Evident, funcţia poligonală nu se identifică cu o funcţie de gradul 1 deoarece depinde şi de diviziunea aleasă. Acest exemplu conduce la ideea de interpolare polinomială pe *porţiuni,* la care pe fiecare subdiviziune a intervalului $[a, b]$ definim un alt polinom de interpolare. Funcţia poligonală este unul dintre exemplele cele mai simple ale acestui tip de interpolare prin *funcţii spline* [3]. Aceste funcţii sunt caracterizate prin formele lor pe subintervalele dintre două noduri (care pot fi diverse funcţii cunoscute) şi prin anumite condiţii de racordare în noduri. În cele ce urmează, vom considera doar cazul *funcţiilor spline polinomiale fără deficienţă.*

[3] Se citeşte "splain".

Figura 1.3: Subintervalele de definire a unei funcţii spline.

Definiţie. Fie $[a,b] \subset \mathbf{R}$ un interval pe dreapta reală şi $x_i, i = 1,2,...,N$ o reţea de puncte de diviziune ($x_1 = a, x_N = b$). Notăm cu I_i subintervalele $[x_i, x_{i+1})$. Funcţia $s : [a,b] \to \mathbf{R}$ se numeşte *funcţie spline polinomială de ordinul* m dacă

1. restricţiile ei pe subintervalele I_i sunt polinoame de gradul m, $s|_{I_i} = p_{m,i}$;
2. s este derivabilă de $m-1$ ori pe intervalul $[a,b]$, $s \in C^{(m-1)}[a,b]$.

A doua condiţie conţine în sine condiţia de racordare în noduri

$$p^{(k)}_{m,i}(x_{i+1}) = p^{(k)}_{m,i+1}(x_{i+1}),\ k = 0,1,\dots,m-1\,, \tag{1.34}$$

adică la frontiera x_{i+1} dintre două subintervale, polinomul din stânga $p_{m,i}$ şi primele sale $m-1$ derivate trebuie să aibe aceleaşi valori cu ale polinomului din dreapta, $p_{m,i+1}$. În afara intervalului $[a,b]$ funcţia s se poate prelungi prin polinoame de grad $\leq m$. Condiţiile de racordare în noduri pot fi slăbite, astfel încât funcţia s să nu mai fie de clasă $C^{(m-1)}$ pe tot intervalul $[a,b]$, ci să fie derivabilă de mai puţine ori pe diverse subintervale. În acest caz, obţinem *funcţii spline cu deficienţă.*

Funcţia spline de ordinul întâi (linia poligonală).

Funcţia spline este formată din segmente de ecuaţie

$$p_{1,i}(x) = y_i + m_i(x - x_i),\ x \in [x_i, x_{i+1}), \tag{1.35}$$

$$m_i = (y_{i+1} - y_i)/h_i,\ h_i \equiv x_{i+1} - x_i\,, \tag{1.36}$$

m_i reprezentând panta pe intervalul I_i (vezi fig. 1.3). Funcţia spline de ordinul întâi este simplă, dar nu furnizează derivata funcţiei interpolate.

Funcţia spline de ordinul doi.

Funcţia este formată din segmente de parabolă, racordate în noduri până la derivata de ordinul 1, inclusiv

$$p_{2,i}(x) = y_i + m_i(x - x_i) + a_i(x - x_i)^2,\ x \in [x_i, x_{i+1}),\ i \in \overline{1, N-1}\ . \quad (1.37)$$

Forma polinomială (1.37) satisface automat condiţia $p_{2,i}(x_i) = y_i$ prin modul de scriere. Condiţiile de racordare

$$p_{2,i}(x_{i+1}) = y_{i+1} \quad (1.38)$$

conduc la următoarele ecuaţii pentru coeficienţii a_i

$$a_i = (y_{i+1} - y_i)/h_i^2 - m_i/h_i, i \in \overline{1, N-1}\,. \quad (1.39)$$

Eliminarea lui a_i din condiţiile de racordare

$$\begin{array}{c} p_{2,i}(x_{i+1}) = y_{i+1}\,, \\ p'_{2,i}(x_{i+1}) = p'_{2,i+1}(x_{i+1})\,, \end{array} \quad (1.40)$$

care se pot scrie în nodurile x_i, $i = 1, 2, 3, \ldots, N-2$ conduce la sistemul

$$m_i + m_{i+1} = 2(y_{i+1} - y_i)/h_i,\ i \in \overline{2, N-1}\ , \quad (1.41)$$

ce trebuie completat cu o singură condiţie. Spre exemplu, se poate da panta la unul din capetele intervalului (m_1 sau m_N). Necesitatea condiţiei suplimentare provine din faptul că nu mai putem impune condiţia de racordare în derivată în nodul x_N. În ambele cazuri, sistemul devine determinat:

1. m_1 dat duce la substituirea

$$m_{i+1} = 2(y_{i+1} - y_i)/h_i - m_i,\ i \in \overline{1, N-1}\ ; \quad (1.42)$$

2. m_N dat permite retrosubstituirea

$$m_i = 2(y_{i+1} - y_i)/h_i - m_{i+1},\ i \in \overline{N-1, 1}\ . \quad (1.43)$$

Funcţia spline de ordinul trei sau cubică.

Este una din cele mai utilizate funcţii spline, având derivate continue până la ordinul doi inclusiv, ceea ce permite calculul razei de curbură. Din condiţiile de continuitate pe noduri până la derivata de ordinul 2 inclusiv

$$\begin{array}{c} p_{3,i}(x_{i+1}) = y_{i+1}\,, \\ p'_{3,i}(x_{i+1}) = p'_{3,i+1}(x_{i+1})\,, \\ p''_{3,i}(x_{i+1}) = p''_{3,i+1}(x_{i+1})\,, \end{array} \quad (1.44)$$

pentru $i = 1, 2, 3, \ldots, N-2$, se deduc coeficienţii polinomului de gradul 3

$$p_{3,i}(x) = y_i + m_i(x - x_i) + b_i(x - x_i)^2 + a_i(x - x_i)^3 \tag{1.45}$$

care reprezintă comportarea funcţiei spline pe fiecare subinterval (x_i, x_{i+1}), $i \in \overline{1, N-1}$

$$a_i = (m_{i+1} + m_i)/h_i^2 - 2(y_{i+1} - y_i)/h_i^3\,, \tag{1.46}$$

$$b_i = 3(y_{i+1} - y_i)/h_i^2 - (m_{i+1} + 2m_i)/h_i\ . \tag{1.47}$$

Pantele pe noduri, m_i, sunt date în acest caz de sistemul

$$\rho_i m_{i-1} + 2m_i + \lambda_i m_i = d_i,\ i \in \overline{2, N-2} \tag{1.48}$$

ce trebuie completat cu două condiţii, pentru a suplini condiţiile de racordare în prima şi a doua derivată ce nu mai pot fi scrise în x_N. S-au făcut notaţiile

$$\rho_i \equiv h_i/(h_{i-1} + h_i),\ \lambda_i \equiv 1 - \rho_i,\ h_i \equiv x_{i+1} - x_i, \tag{1.49}$$

$$d_i \equiv 3\left[\lambda_i(y_{i+1} - y_i)/h_i + \rho_i(y_i - y_{i-1})/h_{i-1}\right]\ .$$

Sistemul de condiţii de racordare impuse lasă de data aceasta două grade de libertate, ce pot fi precizarea pantelor la capete, m_1 şi m_N, sau, mai general, precizarea unei relaţii, în general liniară, ale acestor pante cu pantele vecine, de forma

$$\begin{cases} 2m_1 + \lambda_1 m_2 = d_1 \\ \rho_N m_{N-1} + 2m_N = d_N\,. \end{cases}\ . \tag{1.50}$$

În relaţiile (1.50), coeficienţii $\lambda_1, d_1, \rho_N, d_N$ sunt daţi prin natura condiţiilor puse la capetele intervalului, deci nu sunt deduşi din relaţiile (1.49) care *nu* sunt definite pentru $i = 1$ şi $i = N$. Spre exemplu, a da pantele m_1 şi m_N revine la a impune

$$\lambda_1 = 0,\ d_1 = 2m_1,\ \rho_N = 0,\ d_N = 2m_N\ .$$

Sistemul de N ecuaţii cu N necunoscute $Am = d$, obţinut prin reuniunea egalităţilor (1.48) şi (1.50), are matrice tridiagonală. Ne punem mai întâi problema existenţei unei soluţii. Elementele din matricea A rezultate din condiţiile de racordare (1.48) sunt *diagonal dominante* pe linii, adică $2 > |\rho_i| + |\lambda_i| = 1$. Cum o matrice diagonal dominantă este, în general, inversabilă, este suficient ca şi condiţiile suplimentare (1.50) să păstreze această proprietate. Practic, problema este rezolvabilă dacă impunem condiţii necontradictorii şi distincte.

Forma tridiagonală a matricei A permite o rezolvare foarte eficientă prin descompunerea matricei într-un produs de două matrice bidiagonale[4] $A = L \cdot R$

[4]Un caz particular al factorizării Doolittle ce va fi prezentată în capitolul 5, cunoscut ca algoritmul lui *Thomas*.

sau explicit

$$A = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ l_2 & 1 & \cdots & 0 & 0 \\ 0 & l_3 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 1 & 0 \\ 0 & 0 & \cdots & l_N & 1 \end{bmatrix} \cdot \begin{bmatrix} r_1 & \lambda_1 & \cdots & 0 & 0 \\ 0 & r_2 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & \lambda_{N-2} & 0 \\ \vdots & \vdots & \vdots & r_{N-1} & \lambda_{N-1} \\ 0 & 0 & \cdots & 0 & r_N \end{bmatrix} .$$

Coeficienţii necunoscuţi r_i, $i = 1, 2, ..., N$ şi l_i, $i = 2, 3, ..., N$ se determină prin identificarea elementelor din A şi din matricea produs. Elementele supradiagonalei din matricea R au fost deja identificate. Determinarea pantelor se bazează pe asociativitatea produsului de matrice

$$(L \cdot R) \cdot m = L \cdot (R \cdot m) = d \ .$$

Introducem vectorul $z = R \cdot m$. Etapele algoritmului sunt

1. factorizarea $A = L \cdot R$

$$\begin{array}{l} r_1 \leftarrow 2; \\ i = 2 : N \\ \quad [l_i \leftarrow \rho_i / r_{i-1}; \quad r_i \leftarrow 2 - l_i \lambda_{i-1} \end{array}$$

2. rezolvarea sistemului bidiagonal $L \cdot z = d$

$$\begin{array}{l} z_1 \leftarrow d_1 \\ i = 2 : N \\ \quad [z_i \leftarrow d_i - l_i z_{i-1} \end{array}$$

3. rezolvarea sistemului bidiagonal $R \cdot m = z$

$$\begin{array}{l} m_N \leftarrow z_N / r_N \\ i = (N-1) : -1 : 1 \\ \quad [m_i \leftarrow (z_i - \lambda_i m_{i+1}) / r_i \end{array}$$

Observaţii. 1. Un caz particular important de funcţie spline de ordinul trei este cel al *interpolării spline cubice naturale* definit prin condiţiile în capete

$$s''(x_1) = s''(x_N) = 0$$

ceea ce conduce la

$$2m_1 + m_2 = 3(y_2 - y_1)/h_1 \, ,$$

$$m_{N-1} + 2m_N = 3(y_N - y_{N-1})/h_{N-1} \, , \tag{1.51}$$

adică

$$\lambda_1 = 1,\ d_1 = 3(y_2 - y_1)/h_1,\ \rho_N = 1,\ d_N = 3(y_N - y_{N-1})/h_{N-1}\,, \tag{1.52}$$

cu notaţiile anterioare. Se poate demonstra că impunerea acestor condiţii de capăt minimizează integrala

$$I = \int_{x_1}^{x_N} \left[f''(x)\right]^2 dx\,, \tag{1.53}$$

unde $f(x)$ este funcţia exactă, necunoscută, de clasă $C^{(2)}[a,b]$ ce este aproximată de interpolarea spline. Minimizarea integralei (1.53) prin impunerea *condiţiilor naturale* (1.51) conduce la cea mai netedă interpolare spline cubică. În absenţa unor informaţii precise asupra pantelor în capete m_1, m_N, se recomandă folosirea condiţiilor naturale ce conduc de regulă la minizarea erorii de interpolare.

2. Folosită pentru reprezentarea unor curbe date (traiectorii ale unei scule, profile aerodinamice, etc.), funcţia spline cubică poate avea abateri în zonele cu pante mari. De aceea, se recomandă verificări mai atente în vecinătatea nodurilor cu pante mari, $m_i \geq 5$.

3. Restricţiile unei funcţii spline $s(x)$ pe intervalele I_i nu trebuie să fie obligatoriu polinoame. Se pot racorda alte funcţii, sau polinoame cu alte funcţii – spre exemplu, cercuri cu polinoame. Aceste combinaţii pot fi avantajoase în cazul când pantele pe noduri sunt mari.

4. Pentru interpolarea spline nu s-a dedus o expresie a erorii de aproximare, de o manieră directă ca la interpolarea polinomială. S-a reţinut doar afirmaţia că, o funcţie continuă poate fi aproximată oricât de bine pe tot intervalul $[x_1, x_N]$ atunci când numărul de diviziuni creşte, adică interpolarea spline este *întotdeauna convergentă*. Deoarece derivata de ordinul m, $s^{(m)}$, a unei funcţii spline polinomiale este o funcţie treaptă (constantă pe porţiuni), iar o funcţie treaptă aproximează oricât de bine o funcţie continuă pe interval când numărul de diviziuni creşte, se poate da o evaluare a erorii în funcţie de abaterea maximă între derivatele $f^{(m)}(x)$ şi $s^{(m)}(x)$, presupunând că $f^{(m)}(x)$ există şi este continuă

$$\max|f(x) - s(x)| \leq (b-a)^m \max\left|f^{(m)}(x) - s^{(m)}(x)\right| / m!\ .$$

Pentru $m = 1$, marginea erorii reprezintă tocmai produsul dintre mărimea intervalului şi diferenţa maximă de pante între funcţia f şi linia poligonală. Această evaluare *nu* corespunde însă neapărat funcţiei spline de interpolare, dar sugerează o anumită relaţie cu modul în care funcţia treapta $s^{(m)}(x)$ aproximează derivata de ordinul m a funcţiei $f(x)$.

Exemplu. Vom considera problema interpolării spline a funcţiei lui Runge $f(x) = 1/(1+x^2)$ pe intervalul $[-5,5]$ pentru care am văzut că interpolarea polinomială globală eşuează. Se adoptă o diviziune echidistantă a intervalului cu $h = [5-(-5)]/(N-1)$, $x_k = x_1 + (k-1)h$, $k \in \overline{1,N}$. Se vor considera trei tipuri de condiţii în capete:

1. precizarea valorilor exacte ale pantei $m_1 = f'(-5)$, $m_N = f'(5)$ ceea ce conduce la $\lambda_1 = 0$, $d_1 = 2f'(-5)$, $\rho_N = 0$, $d_N = 2f'(5)$;

Figura 1.4: Logaritmul zecimal al erorii relative la interpolarea spline a f, f', f''.

2. extrapolarea liniară a pantelor adiacente $m_1 = m_2$, $m_{N-1} = m_N$ ceea ce conduce la $\lambda_1 = -2$, $d_1 = 0$, $\rho_N = -2$, $d_N = 0$;

3. condiţii naturale (1.52).

Calitatea aproximării se apreciază prin evaluarea erorilor relative

$$\varepsilon_k = \sum_{i=1}^{1000} \left| f^{(k)}(u_i) - s^{(k)}(u_i) \right| / \left| f^{(k)}(u_i) \right|, \; k = 0, 1, 2$$

pentru diverse valori ale lui N unde $\{u_i, i \in \overline{1, 1000}\}$ este o diviziune echidistantă fină a intervalului $[-5, 5]$. Rezultatele sunt prezentate în figura (1.4). Se observă că interpolarea este convergentă, eroarea relativă scăzând rapid cu creşterea lui N. Pentru un număr mic de puncte toate condiţiile de capăt dau rezultate comparabile. Pe măsură ce diviziunea devine suficient de fină pentru a descrie precis variaţiile funcţiei, condiţiile exacte dau eroarea minimă după cum era de aşteptat, urmate de condiţiile naturale şi apoi de cele de extrapolare a pantei. Se remarcă pierderea a 1,2 ordine de precizie a aproximării în urma derivării.

1.1.3 Interpolare cu funcţii trigonometrice

Interpolarea polinomială nu este adecvată aproximării tuturor tipurilor de funcţii. Vom considera acum o altă alegere a funcţiilor liniar independente din (1.6), anume funcţiile trigonometrice (fig. 1.5)

$$\cos(2\pi k x), \; k \in \overline{0, N}; \; \sin(2\pi m x), \; m \in \overline{1, N-1} \; . \tag{1.54}$$

Această bază este deosebit de eficace în aproximarea funcţiilor periodice $f(x) = f(x+1)$. Funcţiile periodice cu o altă perioadă $f(z) = f(z+T)$ pot fi aduse la forma anterioară prin transformarea $x = z/T$. În cazul interpolării cu funcţii

Figura 1.5: Primele 11 funcţii din baza trigonometrică.

trigonometrice, funcţiile sinus şi cosinus *împreună* formează o bază[5]. Avem un număr de $2N$ funcţii în această bază. Ca atare, vom considera un număr par $2N$ de puncte de diviziune echidistante pe intervalul $[0,1]$

$$x_j = j/2N, j \in \overline{0, 2N-1} \; . \tag{1.55}$$

Se verifică direct că setul (1.54) prezintă următoarele proprietăţi de ortogonalitate pe mulţimea discretă de puncte $\{x_i\} = \{0, 1/2N, 2/2N, ..., (2N-1)/2N\}$

$$\sum_{j=0}^{2N-1} \cos 2\pi k x_j \cos 2\pi m x_j = \left\{ \begin{array}{cc} 0, & k \neq m \\ N, & k = m \neq 0, N \\ 2N, & k = m = 0, N \end{array} \right. \tag{1.56}$$

$$\sum_{j=0}^{2N-1} \sin 2\pi k x_j \sin 2\pi m x_j = \left\{ \begin{array}{cc} 0, & k \neq m \\ N, & k = m \end{array} \right. \; ; \quad \sum_{j=0}^{2N-1} \cos 2\pi k x_j \sin 2\pi m x_j = 0 \, ,$$

$\forall k \in \overline{0, N}$, $m \in \overline{1, N-1}$. Demonstraţia se construieşte prin transformarea produselor de funcţii trigonometrice în sume de sinus şi cosinus. Acestea se pot înlocui cu funcţii exponenţiale, $\sin x = (e^{ix} - e^{-ix})/2i$, $\cos x = (e^{ix} + e^{-ix})/2$, rezultând progresii geometrice simplu de însumat (vezi şi 1.1.4).

[5]Strict vorbind doar mulţimea infinită $\{1, \cos x, \sin x, \cos 2x, \sin 2x, \dots\}$ formează o bază a spaţiului de funcţii. Păstrarea unui număr finit de funcţii conduce la apariţia unor erori ce vor fi considerate mai jos.

Utilizând setul de funcţii de bază (1.54), aproximanta prin interpolare se scrie sub forma *polinomului Fourier*

$$g(x) = \tfrac{1}{2}a_0 + \sum_{k=1}^{N-1} [a_k \cos(2\pi kx) + b_k \sin(2\pi kx)] + \tfrac{1}{2}a_N \cos 2\pi Nx \ , \qquad (1.57)$$

ce satisface $2N$ condiţii de interpolare

$$g(x_j) = f(x_j) \equiv y_j,\ j \in \overline{0, 2N-1} \ .$$

Coeficienţii a_k, b_k se determină prin utilizarea proprietăţilor de ortogonalitate (1.56). Prezentăm calculul doar pentru coeficienţii b_k. Polinomul (1.57) se evaluează în punctele x_j, relaţia obţinută se amplifică cu $\sin 2\pi mx_j$ iar apoi se calculează suma de la $j = 0$ la $j = 2N-1$

$$\sum_{j=0}^{2N-1} y_j \sin 2\pi mx_j = \frac{a_0}{2} \sum_{j=0}^{2N-1} \sin 2\pi mx_j + \sum_{k=1}^{N-1} \left[a_k \sum_{j=0}^{2N-1} (\cos 2\pi kx_j \sin 2\pi mx_j) + b_k \sum_{j=0}^{2N-1} (\sin 2\pi kx_j \sin 2\pi mx_j) \right] + \frac{a_N}{2} \sum_{j=0}^{2N-1} (\cos 2\pi Nx_j \ \sin 2\pi mx_j) \ .$$

Se poate observa schimbarea ordinii de însumare din relaţia de mai sus ce permite aplicarea (1.56) obţinându-se

$$\sum_{j=0}^{2N-1} g(x_j) \sin(2\pi mx_j) = \sum_{j=0}^{2N-1} y_j \sin(2\pi mx_j) = b_m N \ .$$

Un calcul analog pentru ceilalţi coeficienţi conduce la relaţiile

$$a_k = \frac{1}{N} \sum_{j=0}^{2N-1} y_j \cos 2\pi kx_j,\ b_m = \frac{1}{N} \sum_{j=0}^{2N-1} y_j \sin 2\pi mx_j \qquad (1.58)$$

cu $k \in \overline{0, N}$, $m \in \overline{1, N-1}$.

În aplicaţii, coeficienţii a_k, b_k se evaluează *mult* mai economic decât prin calculul direct al sumelor de mai sus prin folosirea transformării Fourier rapide prezentate în 1.1.4. Se poate lesne observa din (1.58) că vom avea toţi $a_k = 0$ pentru funcţii impare $f(-x) = -f(x)$ şi toţi $b_m = 0$ pentru funcţii pare $f(-x) = f(x)$.

Apariţia unei oarecare asimetrii – termenii în cos sunt mai numeroşi decât cei în sin – este legată de alegerea unui număr par de $2N$ intervale în care se divide perioada funcţiei. Dacă se aleg $2N+1$ intervale, forma funcţiei de interpolare este

$$g(x) = \tfrac{1}{2}a_0 + \sum_{k=1}^{N} [a_k \cos(2\pi kx) + b_k \sin(2\pi kx)] \ ,$$

coeficienţii a_k şi b_k fiind daţi de

$$a_k = \frac{2}{2N+1}\sum_{j=0}^{2N} y_j \cos 2\pi k x_j,\; b_m = \frac{2}{2N+1}\sum_{j=0}^{2N} y_j \sin 2\pi m x_j\,,$$

cu $k \in \overline{0,N}$, $m \in \overline{1,N}$, $x_j = j/(2N+1)$.

Convergenţa interpolării trigonometrice.

Se pot determina exprimări ale erorii de interpolare în genul restului $R_{N-1}(x)$ de la interpolarea polinomială şi pentru interpolarea trigonometrică. Expresiile obţinute sunt însă sub o formă integrală greu utilizabilă practic. Vom prefera o discuţie mai puţin formală a erorii şi convergenţei procedeului. Dacă funcţia f admite o dezvoltare în serie Fourier mărirea numărului de noduri conduce la aproximaţii din ce în ce mai bune. Într-adevăr, ridicând expresia (1.57) la pătrat membru cu membru, însumând valorile pe noduri şi ţinând seama de relaţiile de ortogonalitate (1.56), se obţine

$$\frac{1}{4}a_0^2 + \frac{1}{2}\sum_{k=1}^{N-1}(a_k^2+b_k^2) + \frac{1}{4}a_N^2 = \frac{1}{2N}\sum_{j=0}^{2N-1} y_j^2 \tag{1.59}$$

relaţie denumită *egalitatea lui Parseval discretă*. Când N creşte, suma din membrul drept se apropie de integrala $\int_0^1 y^2 dx$. Dacă integrala este mărginită[6] suma este de asemenea mărginită. Ca urmare, seria pătratelor coeficienţilor este convergentă, ceea ce arată că a_k^2, b_k^2 devin din ce în ce mai mici când N creşte. *Interpolarea trigonometrică este aşadar convergentă* pentru funcţii f continue sau cu un număr finit de discontinuităţi – acestea fiind condiţii suficiente pentru a asigura existenţa integralei anterioare. Stabilirea convergenţei este importantă pentru validarea interpolării trigonometrice. În aplicaţii însă mai apar şi alte aspecte ale comportării erorii la interpolare. Vom considera doar două mai importante: rapiditatea convergenţei şi efectul considerării doar a unui număr finit de funcţii trigonometrice în dezvoltarea (1.57).

Fără a încerca o definiţie formală, vom spune că o interpolare trigonometrică este rapid convergentă dacă numărul de termeni N necesar realizării unei precizii impuse a aproximării este "mic". Înţelesul cuvântului "mic" depinde de aplicaţie, dar un domeniu orientativ ar fi $2 \le N \le 128$. Urmărirea fig. 1.5 sugerează că includerea mai multor termeni în polinomul Fourier (1.57) permite descrierea unei funcţii cu variaţii mai rapide pe intervalul $[0,1]$. Fie Δx cea mai mică distanţă de pe abscisă pe care funcţia f are o variaţie semnificativă. Deoarece f are perioada 1, spunem că $\nu = 1/\Delta x$ este frecvenţa variaţiilor celor mai rapide ale funcţiei. Pentru a descrie variaţiile date de frecvenţa cea mai rapidă a funcţiei f polinomul (1.57) trebuie să conţină un număr de termeni $N \ge \nu$. Acest rezultat este cunoscut sub numele de *criteriul Nyquist* ce rezultă dintr-un rezultat mai general denumit *teorema de eşantionare Shannon*. Observaţi că

[6] Într-o formulare riguroasă dacă $y = f(x)$ este pătrat integrabilă pe $[0,1]$, ceea ce se scrie $f \in L_2[0,1]$.

Figura 1.6: Comportarea polinoamelor trigonometrice cu $N = 8, 16, 32, 64$ la interpolarea funcţiei treaptă. Pe măsură ce N creşte, interpolarea se aproprie mai mult de funcţia exactă fără însă a elimina oscilaţiile în zona punctelor de discontinuitate.

pentru a avea N frecvenţe în polinomul (1.57) este nevoie de $2N$ puncte (x_j, y_j). De aici o formulare echivalentă a criteriului Nyquist este ca *numărul de noduri să fie minim de două ori frecvenţa cea mai rapidă*.

Dacă f are variaţii lente atunci ν este mic şi numărul de termeni din polinomul (1.57) este de asemenea mic. Funcţiile cu variaţii rapide au însă ν mare necesitând un număr mare de termeni în polinomul (1.57). Cea mai rapidă frecvenţă de variaţie posibilă a unei funcţii ar fi $\nu \to \infty$ ceea ce corespunde la $\Delta x = 0$, adică f să prezinte discontinuităţi. Cum nu vom putea îndeplini niciodată criteriul Nyquist $N \geq \infty$ pentru funcţii discontinue, interpolarea trigonometrică va avea erori mai mari în asemenea cazuri. Erorile ce apar sunt concentrate în jurul discontinuităţilor, comportare cunoscută sub denumirea de *fenomenul Gibbs*. Un exemplu faimos al fenomenul Gibbs este aproximarea unui semnal dreptunghiular $y(x) = 1$ pentru $n < x < n + 1/2$, $y(x) = -1$ pentru $n + 1/2 < x < n + 1$ şi $y(n + 1/2) = 0$ cu $n \in \mathbf{N}$, exemplu prezentat în fig. 1.6. Ne aşteptăm aşadar ca interpolarea trigonometrică să fie lent convergentă pentru funcţii discontinue şi rapid convergentă pentru funcţii netede, cu variaţii lente.

Să presupunem acum că *nu* este îndeplinit criteriul Nyquist[7] şi am luat un număr prea mic $N < \nu$ de termeni în dezvoltarea (1.57). Ne punem problema dacă coeficienţii a_k, b_k $k \leq N$ determinaţi prin relaţiile (1.58) sunt corecţi, adică au aceleaşi valori ca în cazul în care criteriul Nyquist ar fi satisfăcut. Răspunsul este *negativ*. Să refacem calculul anterior ce a furnizat valorile coeficienţilor b_k, de data aceasta pentru dezvoltarea Fourier completă a funcţiei f ce are

[7] Deoarece funcţia f este în general necunoscută şi ca atare nu cunoaştem frecvenţa ν.

coeficienţii exacţi α_k, β_k

$$f(x) = \tfrac{1}{2}\alpha_0 + \sum_{k=1}^{\infty} \left[\alpha_k \cos(2\pi kx) + \beta_k \sin(2\pi kx)\right] \ . \tag{1.60}$$

Ca mai înainte, evaluăm (1.60) în x_j, înmulţim cu $\sin 2\pi m x_j$ şi însumăm de la $j = 0$ la $j = 2N - 1$. Urmărim doar termenii cu produse de funcţii sinus – ceilalţi dau contribuţii nule conform (1.56) –

$$\begin{aligned}\sum_{j=0}^{2N-1} f(x_j) \sin 2\pi m x_j = &\sum_{k=1}^{2N-1} \left[\beta_k \sum_{j=0}^{2N-1} (\sin 2\pi k x_j \sin 2\pi m x_j)\right] + \\ &\sum_{k=2N}^{4N-1} \left[\beta_k \sum_{j=0}^{2N-1} (\sin 2\pi k x_j \sin 2\pi m x_j)\right] + \quad \ldots + \\ &\sum_{k=2pN}^{2(p+1)N-1} \left[\beta_k \sum_{j=0}^{2N-1} (\sin 2\pi k x_j \sin 2\pi m x_j)\right] + \quad \ldots \ .\end{aligned}$$

Însă $\sin 2\pi(2pN + k)x_j = \sin 2\pi k x_j$ pentru $x_j = j/(2N)$. Se obţine aşadar

$$b_m = \beta_m + \beta_{m+2N} + \beta_{m+4N} + \ldots \ ,$$

altfel spus contribuţiile frecvenţelor mari $m + 2N, m + 4N, \ldots$ apar *mascate* în coeficientul b_m. *Fenomenul de mascare*[8] impune să urmărim în aplicaţii variaţia coeficienţilor a_k, b_k la dublarea lui N. Fie $a_k^{(2N)}$, $b_k^{(2N)}$ coeficienţii determinaţi folosindu-se $2N$ puncte şi $a_k^{(4N)}$, $b_k^{(4N)}$ coeficienţii determinaţi folosindu-se $4N$ puncte. Dacă $a_k^{(2N)} \cong a_k^{(4N)}$, $b_k^{(2N)} \cong b_k^{(4N)}$ pentru $k \leq N$ atunci numărul de puncte a fost considerat suficient de mare pentru a elimina fenomenul de mascare.

EXEMPLU. Folosind interpolarea trigonometrică să se aproximeze poziţia punctelor situate pe elipsa

$$x^2/a^2 + y^2/b^2 = 1 \ .$$

Rezolvare. Ca în orice problemă de aproximare, alegerea variabilelor este importantă. Reprezentarea în coordonate carteziene x, y este dezavantajoasă, conducând la două funcţii

$$y = \pm b\sqrt{1 - x^2/a^2} \ .$$

De aceea se preferă coordonatele polare r, ϕ sau coordonatele paramametrice r, t. În coordonatele polare $x = r\cos\phi$, $y = r\sin\phi$, elipsa este dată de funcţia

$$r(\phi) = ab\left[a^2 \sin^2\phi + b^2 \cos^2\phi\right]^{-1/2} \ ,$$

cu $\phi \in [0, 2\pi]$. Funcţia $r(\phi)$ are perioada 2π. O aducem la perioada 1 prin transformarea $s = \phi/2\pi$,

$$r(s) = ab\left[a^2 \sin^2 2\pi s + b^2 \cos^2 2\pi s\right]^{-1/2} \ .$$

[8] Comportarea este descrisă deseori prin denumirea din engleză de *aliasing*.

Figura 1.7: Variaţia erorii relative la interpolarea trigonometrică a unor elipse. Rezultatele pentru reprezentarea parametrică sunt unite cu linii.

În reprezentarea parametrică, elipsa este descrisă de $x = a\cos t$, $y = b\sin t$, astfel încât obţinem

$$r(t) = \left[a^2 \sin^2 t + b^2 \cos^2 t\right]^{1/2}$$

cu $t \in [0, 2\pi]$. Aducem funcţia la perioada 1 prin $s = t/2\pi$ şi avem o a doua reprezentare

$$r(s) = \left[a^2 \sin^2 2\pi s + b^2 \cos^2 2\pi s\right]^{1/2}$$

Vom nota prin $g_N(s)$ polinomul trigonometric ce interpolează $r(s)$ în $2N$ puncte echidistant repartizate în intervalul $[0, 1]$. Evaluăm calitatea interpolării pentru diverse valori ale lui $N = 4, 8, \ldots, 256$ prin calculul erorii relative pe o diviziune mai deasă $\{\sigma_j = j/2048, j \in \overline{0, 2048}\}$

$$\varepsilon_N = \sum_{j=0}^{4N} |r(\sigma_j) - g_N(\sigma_j)| \,/\, |r(\sigma_j)| \ .$$

Presupunem că $a = 1$ şi vom studia comportarea erorii relative pentru mai multe valori ale lui b, în cele două reprezentări adoptate.

Rezultatele sunt prezentate în fig. 1.7. În toate cazurile, interpolarea este convergentă: pe măsură ce N creşte eroarea se apropie de zero. Cea mai rapidă convergenţă se obţine pentru $a/b = 1/2$ deoarece funcţia ce descrie elipsa are variaţii lente pe intervalul $[0, 1]$. Odată cu scăderea raportului a/b, observăm că este nevoie de un număr mai mare de termeni în polinomul trigonometric pentru a se obţine o precizie dată. Elipsa are variaţii rapide în punctele $(\pm 1, 0)$ şi este nevoie de mai mulţi termeni pentru a satisface criteriul Nyquist. În plus, este de aşteptat ca aproximarea să fie afectată şi de fenomenul Gibbs în zona punctelor $(\pm 1, 0)$ – variaţiile funcţiilor sunt într-atât de

rapide încât nesatisfacerea criteriului Nyquist le face să apară ca nişte discontinuităţi. Se poate observa că, pentru un acelaşi raport a/b reprezentarea parametrică (puncte unite cu linii în fig. 1.7) conduce la erori mai mici în toate cazurile studiate, aratând importanţa unei alegeri judicioase a reprezentării.

Să mai remarcăm că, o dată atinsă o precizie de circa 12 cifre, în cazul unei funcţii $r(s)$ netede, $a/b = 1/2$, creşterea lui N nu mai îmbunătăţeşte calitatea aproximării. Calculele au fost efectuate în virgulă mobilă cu o precizie de circa $11,12$ cifre semnificative. Odată atins acest prag, creşterea lui N produce efecte *detrimentale* deoarece prin creşterea numărului de operaţii aritmetice are loc o acumulare a erorilor de rotunjire. Dacă într-adevăr este nevoie de o acurateţe mai mare trebuie lucrat în precizie sporită.

1.1.4 Interpolare în planul complex

Anumite aplicaţii importante impun interpolarea în planul complex. Printre acestea se numără transformarea conformă şi transformata Fourier discretă care au multe aspecte asemănătoare. *Transformarea conformă* a unui contur (C) de formă oarecare din planul complex $z = x + iy$, pe un cerc (K) de rază egală cu unitatea plasat în planul $\zeta = \xi + i\eta$, astfel încât exteriorul conturului să se transforme pe exteriorul cercului (fig. 1.8), este o problemă des întâlnită în mecanica fluidelor incompresibile sau în electricitate. Conform teoremei de reprezentare Riemann, odată precizate contururile, se mai pot alege trei parametrii reali. Aceştia se determină de regulă din impunerea unor corespondenţe ce au semnificaţie fizică. Vom încerca mai întâi determinarea formei generale a transformării conforme, lăsând la urmă stabilirea celor trei parametrii. Forma generală a transformării conforme este o serie care se limitează la un număr convenabil de termeni M[9]

$$z = \sum_{n=-1}^{M-2} C_{-n}\zeta^{-n} \,. \tag{1.61}$$

Cei M coeficienţi C_{-n}, $n = -1, 0, \ldots, M-2$ se determină din condiţii de corespondenţă între punctele $P_k(z_k)$ de pe contur şi punctele $\Pi_k(\zeta_k)$, $k = \overline{0, M-1}$ de pe cerc. Prin urmare se obţine sistemul de ecuaţii

$$z_k = \sum_{n=-1}^{M-2} C_{-n}\zeta_k^{-n} \,. \tag{1.62}$$

Este convenabil ca împărţirea cercului să se facă în părţi egale, adică să luăm

$$\zeta_k = \exp(ik\delta),\ i \equiv \sqrt{-1},\ \delta \equiv 2\pi/M \,. \tag{1.63}$$

Coeficienţii C_{-n} sunt soluţia sistemului

$$\sum_{n=-1}^{M-2} C_{-n}e^{-ik\delta n} = z_k \,, \tag{1.64}$$

[9]Forma reprezentată a fost astfel aleasă încât punctele de la infinit ale celor două plane complexe să corespundă.

Figura 1.8: Corespondenţa punctelor din planul fizic şi cel de calcul.

punctele z_k fiind date. Sistemul (1.64) se inversează uşor, ţinând seama că funcţiile $\exp(\pm ik\delta n)$ formează un sistem ortogonal. Într-adevăr, înmulţind (1.64) membru cu membru, cu $\exp(ijk\delta)$, $j = 1, 2, ..., M-2$ şi sumând după k, se obţine succesiv

$$\sum_{k=0}^{M-1} z_k e^{ijk\delta} = \sum_{k=0}^{M-1} \sum_{n=-1}^{M-2} C_{-n} e^{ik\delta(j-n)} = \sum_{n=-1}^{M-2} C_{-n} \sum_{k=0}^{M-1} e^{ik\delta(j-n)} \ . \tag{1.65}$$

Dar progresia geometrică de raţie $\exp i\delta(j-n)$ se poate suma obţinându-se

$$\sum_{k=0}^{M-1} e^{ik\delta(j-n)} = \frac{1-e^{i\delta(j-n)M}}{1-e^{i\delta(j-n)}} = \begin{cases} 0 & \text{dacă } j \neq n \\ M & \text{dacă } j = n \end{cases} \ .$$

În consecinţă, din (1.65) se obţine

$$C_{-j} = \frac{1}{M} \sum_{k=0}^{M-1} e^{ijk\delta}, \ j \in \overline{-1, M-2} \ . \tag{1.66}$$

Trebuie menţionat că problema nu s-a încheiat odată cu găsirea coeficienţilor C_{-j} pentru un M ales. Anume, mai trebuie verificată atingerea preciziei dorite prin considerarea a unor puncte intermediare ca, de exemplu,

$$\zeta'_k = \exp\left[i\delta(k+1/2)\right], \ k \in \overline{0, M-1} \ .$$

Dacă afixele z'_k sunt prea îndepărtate de conturul (C) atunci trebuie îndesită reţeaua de puncte de calcul şi, în consecinţă, mărit numărul de termeni în seria (1.61).

Rezultatele obţinute se înscriu într-o formulare mai generală, importantă pentru multe aplicaţii[10]. Pentru reţeaua de puncte $x_j = jh$, $j = 0, 1, \ldots, N-1$,

[10] Algoritmul TFR prezentat în continuare este, conform unor studii statistice de utilizare a calculatoarelor, al doilea cel mai des utilizat algoritm, fiind devansat doar de rezolvarea directă a sistemelor liniare. Exemple de aplicaţii: modelări meteorologice, analize de vibraţii mecanice, prelucrare de imagini.

având pasul $h = 1/N$, setul de valori ale unei funcţii u în noduri se notează $u^{(N)}$ şi se numeşte *funcţie reţea*. Notând mai departe cu ω rădăcina de ordinul N a unităţii

$$\omega = \exp(2\pi i/N), \tag{1.67}$$

se defineşte *transformata Fourier discretă directă (TFDD)* a funcţiei $u^{(N)}$ ca fiind o altă funcţie reţea, notată $U^{(N)}$, ale cărei valori în noduri sunt date de relaţiile

$$U_k^{(N)} = \sum_{j=0}^{N-1} u_j^{(N)} \omega^{jk},\ k \in \overline{0, N-1}\ . \tag{1.68}$$

Se observă că (1.66) este un caz particular de TFD. Prescurtăm notaţia prin adoptarea semnului $\Longrightarrow$ pentru TFD dată de (1.68)

$$u^{(N)} \Longrightarrow U^{(N)}\ .$$

Sistemul (1.68) se poate inversa, prin aplicarea relaţiilor de ortogonalitate obţinându-se

$$u_j^{(N)} = \frac{1}{N} \sum_{k=0}^{N-1} U_k^{(N)} \omega^{-jk},\ j \in \overline{0, N-1}\ . \tag{1.69}$$

denumită *transformata Fourier discretă inversă (TFDI)* ce va fi notată

$$u^{(N)} \Longleftarrow U^{(N)}\ .$$

Calculul direct al sumelor din (1.68) revine la înmulţirea vectorului $u^{(N)}$ cu matricea $\Omega = [\omega^{jk}]$, $j, k \in \overline{0, N-1}$ şi ar necesita $O(N^2)$ operaţii aritmetice. Însă matricea Ω are proprietăţi remarcabile, provenite din structura sa ciclică, ce permit o evaluare în numai $O(N \log N)$ operaţii. Câştigul este enorm, iar algoritmul ce realizează acest câştig se numeşte *transformata Fourier rapidă* sau prescurtat *TFR*[11]. Vom prezenta cea mai simplă deducere a algoritmului, o exemplificare a tehnicii generale *divide et impera* – o problemă dificilă se poate rezolva uneori mai uşor prin descompunerea în două probleme mai simple.

Vom presupune că funcţia reţea $u^{(N)}$ este definită pe un număr par de puncte $N = 2P$. Din funcţia reţea $u^{(2P)}$ construim două noi funcţii reţea $v^{(P)}, w^{(P)}$ definite în punctele de indice par, respectiv impar

$$v_m^{(P)} = u_{2m}^{(2P)},\ w_m^{(P)} = u_{2m+1}^{(2P)},\ m \in \overline{0, P-1}\ .$$

[11] O aplicaţie tipică provine din domeniul previziunilor meteo pe termen scurt pe care se bazează navigaţia aeriană şi marină. Tipic, pentru o previziune pe 3 zile, se calculează $\sim 10^6$ TFD-uri de lungime $N = 2^{14}$. Un calculator performant, de viteză 10^9 operaţii aritmetice pe secundă, n-ar reuşi calculul produselor matrice-vector decât în 74 ore – prea târziu ca să mai fie de folos. Cu folosirea TFR timpul se reduce la circa 4 minute!

Suma (1.68) se poate scrie

$$U_k^{(2P)} = \sum_{j=0}^{2P-1} u_j^{(2P)} \omega^{jk} = \sum_{m=0}^{P-1} \left[v_m^{(P)} \omega^{2mk} + w_m^{(P)} \omega^{(2m+1)k} \right] =$$

$$\sum_{m=0}^{P-1} v_m^{(P)} \omega^{2mk} + \omega^k \sum_{m=0}^{P-1} w_m^{(P)} \omega^{2mk} \ . \tag{1.70}$$

Însă ω^2 este rădăcina de ordinul $N/2 = P$ a unităţii astfel încât se observă apariţia în (1.70) a transformatelor

$$v^{(P)} \Longrightarrow V^{(P)},\ w^{(P)} \Longrightarrow W^{(P)} \ ,$$

relaţiile de legătură între cele trei transformate fiind deosebit de simple

$$U_k^{(2P)} = V_k^{(P)} + \omega^k W_k^{(P)},\ U_{k+P}^{(2P)} = V_k^{(P)} - \omega^k W_k^{(P)},\ k \in \overline{0, P-1} \tag{1.71}$$

unde s-a folosit identitatea $\omega^{k+P} = -\omega^k$. Acest rezultat este cunoscut ca *lema lui Danielson şi Lanczos*. Evaluarea directă a lui $U^{(2P)}$ ar fi costat $O(4P^2)$ operaţii aritmetice. Acum avem de evaluat două transformate, $V^{(P)}, W^{(P)}$ ce fiecare necesită $O(P^2)$ operaţii pentru un total de $O(2P^2)$ operaţii. Reducerea la jumătate a calculelor este benefică, însă se poate câştiga şi mai mult dacă N este o putere a lui 2, $N = 2^q$. În acest caz separarea pe indici pari şi impari poate continua până când se ajunge la transformarea unui vector de lungime 1 ce nu necesită nici o operaţie aritmetică deoarece se reduce la transformarea de identitate, $U^{(1)} = u^{(1)}$. Singurele operaţii artimetice ce mai rămân sunt înmulţirile cu ω^k din (1.71). Calculul se poate vizualiza ca parcurgerea arborelui cu $q = \log_2 N$ nivele din fig. 1.9. Pe fiecare nivel sunt necesare doar N operaţii aritmetice astfel încât obţinem costul menţionat anterior de $O(N \log_2 N)$.

Separarea repetată în indici pari şi impari conduce la o altă ordonare a componentelor vectorului $u^{(N)}$. Exemplificăm pentru $N = 2^3 = 8$. Vectorul $u^{(N)}$ are componente de indici de la 0 la 7. Procedeul de separare par-impar se aplică de două ori după cum este arătat în următoarea schemă

etapa 0:	0	1	2	3	4	5	6	7
etapa 1:	0	2	4	6	1	3	5	7
etapa 2:	0	4	2	6	1	5	3	7

Înmulţirile cu diversele puteri ale lui ω trebuie să le începem asupra vectorului permutat $\{u_0, u_4, u_2, u_6, u_1, u_5, u_3, u_7\}$. Permutarea se poate construi însă foarte uşor dacă scriem indicii iniţiali şi finali în binar

etapa 0:	000_2	001_2	010_2	011_2	100_2	101_2	110_2	111_2
etapa 2:	000_2	100_2	010_2	110_2	001_2	101_2	011_2	111_2

.

Observaţi că inversarea ordinei de citire a indicilor din ultima etapă corespunde chiar la numerotarea naturală $0, 1, \ldots, 7$ anume: 000_2 citit de la dreapta la

Figura 1.9: Arborele de recurenţă de la TFR.

stânga este $000_2 = 0$, 100_2 citit de la dreapta la stânga este $001_2 = 1$,...,011_2 citit de la dreapta la stânga este $110_2 = 6$, 111_2 citit de la dreapta la stânga este $111_2 = 7$. Rezultatul este valabil pentru orice $N = 2^q$ iar permutarea finală a indicilor se zice în ordine *bit-inversată*. Putem da acum algoritmul TFR

$j = 1 : N-2$
  $k \leftarrow invbit(j, q)$
  dacă $k > j$ $permută(u_j, u_k)$
$rad \leftarrow \exp(semn \cdot 2\pi i/N)$;
$\omega_0 \leftarrow 1$
$j = 1 : N/2$
  $\omega_j \leftarrow rad \cdot \omega_{j-1}$
$t \leftarrow 1$; $p \leftarrow 2$; $s \leftarrow N/2$

cât timp $p \leq N$
  $j = 0 : p : N-1$
    $r \leftarrow 0$
    $k = 0 : t-1$
      $tmp \leftarrow u_{j+k}$
      $u_{j+k} \leftarrow tmp + \omega_r u_{j+k+t}$
      $u_{j+k+t} \leftarrow tmp - \omega_r u_{j+k+t}$
      $r \leftarrow r + s$
  $t \leftarrow 2t$; $p \leftarrow 2p$; $s \leftarrow s/2$.

Vectorul iniţial u este înlocuit de transformata sa discretă – ca atare algoritmul nu necesită memorie suplimentară. Variabila *semn* ia valoarea 1 pentru TFR directă şi -1 pentru TFR inversă; în acest ultim caz vectorul final mai trebuie împărţit la N. Funcţia $invbit(j, q)$ întoarce indicele ce se obţine prin inversarea ordinei biţilor de la 0 la $q-1$ din j. Operaţia se poate exprima într-un limbaj de nivel înalt prin înmulţiri şi împărţiri cu 2, dar de regulă este implementată eficient la nivel de cod maşină. Verificarea condiţiei $k > j$ are rolul de a nu strica permutările deja efectuate în cadrul buclei.

Algoritmul TFR poate fi folosit pentru calculul coeficienţilor polinomului trigonometric (1.57) precizaţi de relaţiile (1.58). O cale evidentă este să se scrie

$$c_k \equiv a_k + ib_k = \frac{1}{N} \sum_{j=0}^{2N-1} y_j \exp(2\pi ijk/2N),\ k \in \overline{0, N}$$

şi să se aplice TFR asupra vectorului y_j, $j \in \overline{0, 2N-1}$ ce are $2N$ componente. Însă, dacă vectorul $\{y_j\}$ este real, se poate obţine rezultatul cu doar jumătate

Figura 1.10: O măsurătoare experimentală a unei vibraţii compuse complexe.

din efortul de calcul printr-o separare în componente pare şi impare

$$z_m = y_{2m} + iy_{2m+1},\ m \in \overline{0, N-1}\ .$$

Introducem transformatele directe $z \Longrightarrow Z$, $(y_{2m}) \Longrightarrow P$, $(y_{2m+1}) \Longrightarrow I$. Avem

$$Z_k = P_k + iI_k, \quad Nc_k = P_k + I_k \exp(\pi ik/N)\,.$$

Se verifică imediat că TFD, F a unei funcţii reale f satisface $F_{N-k} = F_k^*$, unde F_k^* este conjugata complexă a lui F_k. Ca atare

$$Z_{N-k} = P_{N-k} + iI_{N-k} = P_k^* + iI_k^* \Rightarrow Z_{N-k}^* = P_k - iI_k$$

şi putem deduce expresiile $P_k = (Z_k + Z_{N-k}^*)/2, I_k = -i(Z_k - Z_{N-k}^*)/2$ astfel încât

$$c_k = \tfrac{1}{2N}(Z_k + Z_{N-k}^*) - \tfrac{i}{2N}(Z_k - Z_{N-k}^*)\exp(\pi ik/N),\ k \in \overline{0, N}\,. \tag{1.72}$$

Există multe astfel de combinaţii posibile ce furnizează moduri economice de a calcula transformata Fourier a unui vector ce are proprietăţi de simetrie suplimentare.

Exemplu. Prezentăm o aplicaţie reală tipică. Un senzor de vibraţii a măsurat deplasările unei componente mecanice rezultând datele din fig. 1.10. Vibraţiile de frecvenţă mai joasă de 1 Hz nu prezentau interes astfel încât fereastra de timp în care s-au măsurat datele a fost de $\Delta t = 1$ sec. Se cere identificarea frecvenţelor de amplitudine maximă astfel încât să se poată identifica sursele de excitaţie ce produc vibraţia componentei.

Rezolvare. Măsurătorile au fost efectuate cu patru rate de eşantionare $N = 256$, 512, 1024 rezultând datele $u^{(256)}, u^{(512)}, u^{(1024)}$. Pentru identificarea frecvenţelor dominante se calculează *spectrul de putere* al semnalului definit de

$$P_u(f) = 2U(f)\overline{U}(f)\,,$$

unde $u \Longrightarrow U$. Spectrul se calculează prin aplicarea TFR asupra datelor din fig. 1.10. Deoarece semnalul este real se aplică relaţia (1.72), lungimea transformatelor fiind

Figura 1.11: Spectrele de putere ale semnalului anterior pentru $N = 256, 512, 1024$.

$N/2$. Rezultatele sunt prezentate în fig. 1.11. Maximele locale sunt denumite *picuri* şi corespund la vibraţiile predominante. Se poate observa că la trecerea la o rată de eşantionare mai mare uneori apar noi picuri în domeniul frecvenţelor mari iar unele picuri din domeniul frecvenţelor mici dispar. De asemenea, amplitudinile asociate unui pic se modifică uneori. Alte picuri nu par afectate de creşterea ratei de eşantionare.

Comportarea rezultă din fenomenul de mascare discutat în 1.1.3. De exemplu, picurile α, β sunt stabile la trecerea de la spectrul 2 la 3. Rezultă că pentru aceste frecevenţe mici criteriul Nyquist este statisfăcut şi în plus nu are loc mascharea unor frecvenţe mai înalte. Picul γ din spectrul 2, construit cu $N = 512$ dispare complet însă la trecerea la spectrul 3, construit cu $N = 1024$. În spectrul 2 picul γ era fals, în sensul că nu reprezenta o vibraţie reală cu frecvenţa f_γ. De fapt, rata de eşantionare folosită era prea mică astfel încât se înregistra contribuţia unei frecvenţe mai înalte $f_\delta \simeq 2f_\gamma$, frecvenţă a cărei contribuţie devine discernabilă atunci când se dublează rata de eşantionare în spectrul 3. Într-adevăr pentru a discerne frecvenţa $f_\delta = 350$ Hz criteriul Nyquist indică necesitatea a cel puţin 700 puncte, condiţie realizată doar de măsurarea $u^{(1024)}$. Exemplul arată importanţa *studiilor de rezoluţie* în aplicaţii de aproximare spectrală, studii în care se urmăreşte stabilitatea spectrului la modificarea ratelor de eşantionare. În aplicaţii practice, fenomenul de mascare se elimină prin folosirea unor filtre 'trece-jos' cu preţul pierderii porţiunii de frecvenţe înalte a spectrului.

1.2 Aproximarea mini-max

Trecem acum la considerarea celui de-al doilea criteriu de aproximare (1.3). Exemplul lui Runge din 1.1.1 a arătat că aproximarea prin interpolare, chiar dacă trece prin noduri, poate prezenta erori mari (chiar infinit de mari!) între noduri. De aici provine ideea de a încerca găsirea unei aproximări *optimale* pe *tot* intervalul $[a, b]$, nu numai în noduri. O exprimare matematică a acestei idei este să se minimizeze eroarea maximă între funcţia $f(x)$ şi aproximanta $g(x)$, adică criteriul (1.3) pe care-l repetăm aici

$$\max_{x\in[a,b]} |f(x) - g(x)| = \text{minim} \ . \tag{1.73}$$

Astfel formulată problema este momentan incomplet definită deoarece nu am precizat din ce mulţime de funcţii luăm pe $g(x)$.

1.2.1 Polinoamele Cebâşev

Vom începe prin a presupune iarăşi că funcţia aproximantă $g(x)$ aparţine mulţimii polinoamelor. Criteriul de minimizare a erorii maxime (1.73) face referire la un anume interval $[a, b]$. Deoarece rezultatele de mai jos depind de interval, adoptăm un interval canonic $[-1, 1]$. O transformare de scară $x = z(b-a)/2 + (b+a)/2$, $x \in [a, b]$, $z \in [-1, 1]$ poate aduce orice alt interval finit pe cel canonic.

De data aceasta nu vom scrie funcţia aproximantă $g(x)$ ca o combinaţie liniară de monoame. Examinînd comportarea monoamelor x^k pe intervalul $[-1, 1]$ se constată că toate iau valori absolute maxime la capetele intervalului; prin urmare este de aşteptat ca erorile de aproximare să nu fie distribuite uniform pe interval. Aceasta sugerează căutarea unor polinoame care să nu varieze monoton pe $[-1, 1]$. Intuitiv, variaţia monotonă face dificilă descrierea unor variaţii rapide ale funcţiei f în interiorul intervalului $[-1, 1]$. Ceea ce ne trebuie este o mulţime de polinoame definite pe $[-1, 1]$ care să poată descrie astfel de variaţii într-un mod cât mai economic, adică folosind un număr mic de polinoame. Funcţiile trigonometrice considerate anterior prezentau parţial o astfel de comportare: $\cos 2\pi x$ şi $\sin 2\pi x$ descriau variaţiile de perioadă 1, $\cos 4\pi x$ şi $\sin 4\pi x$ descriau variaţiile de perioadă $1/2$, ş.a.m.d. Spunem parţial deoarece nu a fost îndeplinită şi dorinţa de economicitate – avem două funcţii, sin şi cos pentru descrierea variaţiilor de o anumită perioadă. Pe baza funcţiilor trigonometrice putem însă introduce o clasă remarcabilă de polinoame descoperite de Cebâşev a căror proprietăţi sunt deosebit de favorabile aproximării optimale căutate în această secţiune.

Vom lua $z \in [-1, 1]$. Introducem variabila $\theta \in [0, \pi]$ prin relaţiile

$$z = \cos\theta, \ \theta = \arccos z \, .$$

Să considerăm acum funcţia $T_n(z) = \cos n\theta$. Funcţia are n rădăcini pe $(-1, 1)$

$$T_n(z) = \cos n\theta = 0 \Rightarrow \theta_k = (2k-1)\pi/2n, \ k \in \overline{1, n} \ ,$$

Figura 1.12: Primele 5 polinoame Cebâşev.

Tabelul 1.2: Polinoame Cebâşev

n	$T_n(z)$	z^n
0	1	T_0
1	z	T_1
2	$-1+2z^2$	$(T_0+T_2)/2$
3	$-3z+4z^3$	$(3T_1+T_3)/4$
4	$1-8z^2+8z^4$	$(3T_0+4T_2+T_4)/8$
5	$5z-20z^3+16z^5$	$10T_1+5T_3+T_5$

$$z_k = \cos\theta_k = \cos[(2k-1)\pi/2n],\ k \in \overline{1,n}\ .$$

Din identitatea trigonometrică $\cos n\theta + \cos(n-2)\theta = 2\cos(n-1)\theta\cos\theta$ deducem o relaţie de recurenţă pentru $T_n(z)$

$$T_n(z) = 2zT_{n-1}(z) - T_{n-2}(z)\ . \tag{1.74}$$

Deoarece $T_0(z) = 1$, $T_1(z) = z$ iar $T_2(z), T_3(z), \ldots$ rezultă din (1.74) prin operaţii de adunare şi înmulţire, putem observa că $T_n(z)$ sunt polinoame de gradul n, denumite *polinoame Cebâşev* relative la intervalul compact $[-1,1]$. Funcţia $\cos n\theta$ se numeşte *funcţia generatoare* a acestor polinoame.

Graficele primelor câteva polinoame Cebâşev sunt prezentate în fig. 1.12 iar expresiile lor în tabelul 1.2 împreună cu exprimările monoamelor z^n în funcţie de $T_k(z)$, $k \in \overline{0,n}$. Se observă din fig. 1.12 că fiecare polinom Cebâşev de grad mai înalt poate descrie variaţii mai rapide ale unei funcţii pe intervalul $[-1,1]$.

Din formula de recurenţă, se observă că în $T_n(z)$ coeficientul lui z^n este 2^{n-1}, astfel încât polinomul $\bar{T}_n(z) \equiv 2^{1-n}T_n(z)$ are coeficientul lui z^n egal

cu unitatea. Polinoamele cu un coeficient 1 al termenului de grad maxim se numesc *polinoame monice*. Proprietatea esenţială a polinoamelor Cebâşev ce le face adecvate aproximării mini-max este:

Teoremă. Dintre toate polinoamele monice de grad n fixat, $\bar{T}_n(z)$ are cea mai mică margine în valoare absolută pe intervalul $[-1,1]$.

Demonstraţie. Procedăm prin reducere la absurd. În acest scop să observăm mai întâi că funcţia $\left|\bar{T}_n(z)\right| = 2^{1-n}\left|\cos n\theta\right|$ ia de $n+1$ ori valorea maximă 2^{1-n} pe $[-1,1]$, anume în punctele distincte $z'_k = \cos(k\pi/n)$, $k \in \overline{0,n}$. Să considerăm apoi un alt polinom monic de acelaşi grad $\bar{p}_n(z)$, şi să presupunem prin absurd că acesta ar avea o margine superioară mai mică în modul decît $\bar{T}_n(z)$ pe $[-1,1]$, adică

$$\sup_{z\in[-1,1]} |\bar{p}_n(z)| < \sup_{z\in[-1,1]} \left|\bar{T}_n(z)\right| = 2^{1-n}\ . \tag{1.75}$$

În consecinţă, chiar în punctele z'_k

$$|\bar{p}_n(z'_k)| < 2^{1-n}\,. \tag{1.76}$$

Considerînd diferenţa $d_{n-1}(z)$ a polinoamelor $\bar{T}_n(z)$ şi $\bar{p}_n(z)$, $d_{n-1}(z) \equiv \bar{T}_n(z) - \bar{p}_n(z)$ vom obţine, evident, un polinom de gradul $n-1$. Din (1.76) rezultă

$$(-1)^k\left[\bar{T}_n(z'_k) - \overline{p}_n(z'_k)\right] = (-1)^{k+n} d_{n-1}(z_k) > 0, \quad k \in \overline{n,0}$$

adică $d_{n-1}(z)$ are n schimbări de semn pe $(-1,1)$, deci polinomul de gradul $n-1$, $d_{n-1}(z)$, ar avea n rădăcini, ceea ce este absurd. Rezultă că presupunerea (1.75) nu este adevărată, deci oricare ar fi polinomul monic $\bar{p}_n(z)$ avem

$$\sup_{z\in[-1,1]} |\bar{p}_n(z)| \geq 2^{1-n}\ . \quad \mathbf{2} \tag{1.77}$$

În continuare se dau două aplicaţii importante ale polinoamelor mini-max.

1.2.2 Minimizarea erorii la interpolarea polinomială

Trecem la un prim exemplu concret de aplicare a criteriului mini-max, anume în formularea generală (1.73) vom lua funcţia aproximantă un polinom de gradul $N-1$

$$\max_{x\in[a,b]} |f(x) - p_{N-1}(x)| = \text{minim}.$$

Dorim să determinăm, dintre toate polinoamele de grad $N-1$, pe cel ce minimizează abaterea maximă faţă de funcţia $f(x)$. Polinomul obţinut va fi numit *polinom mini-max* de grad $N-1$ al funcţiei $f(x)$ pe intervalul $[a,b]$. Este convenabil să privim polinomul p_{N-1} ca fiind definit de faptul că trece prin nodurile $\{(x_i,y_i), i \in \overline{1,N}\}$. Nodurile însă, spre deosebire de problema anterioară de

interpolare, sunt acuma *necunoscute*. Vom minimiza eroarea maximă printr-o alegere adecvată a nodurilor. Eroarea este în acest caz chiar restul interpolării ce are forma (1.16)

$$R_{N-1}(x) = \prod_{i=1}^{N}(x - x_i)\ f^{(N)}(\xi)/N! \quad .$$

În general derivata $f^{(N)}(\xi)$ nu este cunoscută, astfel încât se poate pune doar problema minimizării produsului. Acest este un polinom de gradul N. Utilizând schimbarea de variabilă

$$x = z(b-a)/2 + (b+a)/2 \tag{1.78}$$

trecem de la intervalul $[a, b]$ pe care dorim să minimizăm eroarea la intervalul $[-1, 1]$. Se obţine, considerând şi (1.77)

$$\left|\prod_{i=1}^{N}(x - x_i)\right| = \left(\frac{b-a}{2}\right)^N \left|\prod_{i=1}^{N}(z - z_i)\right| \geq \left(\frac{b-a}{2}\right)^N 2^{1-N} \ .$$

Rezultă că optimul, corespunzând egalităţii, se realizează dacă punctele z_i sunt rădăcinile polinomului Cebâşev de gradul n. Vom aranja indicii astfel încât rădăcinile să apară în ordine crescătoare

$$z_{N-i+1} = \cos[(2i-1)\pi/2N],\ i \in \overline{1, N} \ . \tag{1.79}$$

În aceste condiţii se obţine cea mai mică margine superioară pentru restul la interpolare

$$|R_{N-1}(x)| \leq 2^{1-2N}(b-a)^N \max_{\xi\in[a,b]} \left|f^{(N)}(\xi)\right| /N! \, .$$

Marginea depinde de mărimea intervalului (a, b), de numărul de noduri N şi de derivata $f^{(N)}$. Rezultatul obţinut se poate enunţa: dintre toate polinoamele de interpolare de grad $N-1$, cel ce minimizează eroarea maximă $\max|f(x) - p_{N-1}(x)|$ cu $x \in [-1, 1]$ este cel construit cu abscisele nodurilor date de rădăcinile polinomului Cebâşev de grad N.

Exemplu. Am văzut că interpolarea funcţiei lui Runge $f(x) = 1/(1+x^2)$ pe $[-5, 5]$ cu o repartiţie echidistantă a nodurilor nu converge. Vom determina acum un alt polinom ce aproximează pe $f(x)$ dar la care nodurile nu mai sunt luate echidistant, ci sunt determinate de criteriul mini-max, anume vom lua $x_k = 5z_k$, $k \in \overline{1, N}$, cu z_k determinate de (1.79). Ca în exemplele precedente calitatea aproximării este apreciată prin evaluarea erorii relative $\epsilon = \sum_{i=1}^{500} |f(u_i) - p_{N-1}(u_i)| \,/\, |f(u_i)|$ pe o diviziune mai fină a intervalului $[-5, 5]$. Variaţia erorii ϵ cu N este

N	5	10	15	20	25	30	35	40	45	50
$\lg\epsilon$	-0.4	-0.8	-1.2	-1.6	-2.1	-2.4	-3.0	-3.3	-3.8	-4.2

obervându-se că aproximarea converge. Observăm acum clar importanţa alegerii nodurilor pe care se bazează o interpolare. Aproximarea prezentă este tot o interpolare, bazată însă pe alte noduri decât cele echidistante considerate anterior. Comportarea aproximării este redată în fig. 1.13. Se observă că, pentru N mic, apar oscilaţii între noduri dar, spre deosebire de cazul redat în fig. 1.1, acestea nu mai cresc la infinit.

Figura 1.13: Polinoamele de interpolare cu noduri alese optimal pentru $N = 11, 21, 31$ suprapuse peste $f(x)$. Nodurile sunt reprezentate doar pentru $N = 31$.

1.2.3 Aproximarea aproape mini-max a unei funcţii

Vom considera acum o relaxare a criteriului (1.73) anume

$$\max_{x\in[a,b]} |f(x) - p_N(x)| \leq e \ . \tag{1.80}$$

Criteriul (1.80) este mai larg decât cel anterior (1.73) deoarece nu mai impunem determinarea polinomului aproximant $p_N(x)$ astfel încât să realizăm un minim al erorii, ci doar ca eroarea să devină mai mică decât un prag dat e. De data aceasta gradul polinomului nu se mai consideră fixat, ci căutăm polinomul de gradul cel mai mic ce satisface condiţia (1.80). Asemenea probleme apar tipic în situaţii în care dorim o aproximare cât mai economică în operaţii aritmetice[12].

Vom lua $[-1, 1]$ pentru intervalul din (1.80) şi vom presupune că funcţia $f(x)$ are o dezvoltare în serie de puteri

$$f(x) = \sum_{k=0}^{M} b_k x^k \ , \tag{1.81}$$

unde eventual putem avea $M \to \infty$. Puterile x^k se pot exprima în funcţie de polinoamele Cebâşev (vezi tabel 1.2) astfel încât f se poate scrie

$$f(x) = \sum_{k=0}^{M} a_k T_k(x) \ . \tag{1.82}$$

Am văzut că polinoamele Cebâşev sunt mai eficiente în descrierea variaţiilor unei funcţii pe $[-1, 1]$ decât monoamele x^k. Este aşadar de aşteptat ca şirul

[12] Procedeul ce urmează mai este cunoscut sub denumirea de *economizare de serii*.

$\{a_k\}$ de coeficienţi să scadă mai repede în modul decât şirul $\{b_k\}$. Dorim să păstrăm un număr de termeni cât mai mic posibil în dezvoltarea (1.82) pentru a reduce numărul de operaţii aritmetice. Ca atare, trunchiem dezvoltarea la $k = N$, aproximarea fiind

$$p_N(x) = \sum_{k=0}^{N} a_k T_k(x).$$

Deoarece $|T_k(x)| \leq 1$, vom putea trunchia seria (1.82) la cel mai mic N pentru care avem

$$|f(x) - p_N(x)| = \sum_{k=N+1}^{M} |a_k| < e \ .$$

Polinomul $p_N(x)$ obţinut poate fi diferit de polinomul mini-max de grad N al funcţiei $f(x)$ şi este uzual denumit *polinom aproape mini-max*. Apare imediat întrebarea: de ce mai e nevoie de un procedeu diferit de cel din 1.2.2? De ce nu construim direct polinomul mini-max de grad N? Observaţi însă că în procedeul din 1.2.2 gradul polinomului se considera cunoscut. De asemenea, cu toate că se ştie că polinomul mini-max realizează o eroare minimă, *nu este cunoscută valoarea efectivă* a acestei erori. Urmând algoritmul din această secţiune, avem certitudinea îndeplinirii unei condiţii de eroare impusă.

Exemplu. Într-un calculator de navigaţie aeriană, este nevoie de un algoritm de aproximare a funcţiei $\cos x$. Abaterea maximă admisă este $e \leq 10^{-5}$ pe intervalul $[-\pi/2, \pi/2]$. Calculatorul lucrează în timp real. Ca atare este esenţial ca algoritmul să fie cât mai economic posibil. Pentru a rezolva problema se trece la intervalul $[-1, 1]$, cu schimbarea de variabilă $x = \pi z/2$. Funcţia $\cos(\pi z/2)$ are dezvoltarea în serie

$$\cos\frac{\pi}{2}z = 1 - \frac{1}{2!}\left(\frac{\pi}{2}\right)^2 z^2 + \frac{1}{4!}\left(\frac{\pi}{2}\right)^4 z^4 + \ldots + (-1)^n \frac{1}{(2n)!}\left(\frac{\pi}{2}\right)^{2n} z^{2n} + R_{2n+1}.$$

Restul dezvoltării poate fi mărginit după cum urmează

$$|R_{2n+1}| = \frac{\left|\cos^{(2n+1)}\xi\right|}{(2n+1)!}\left(\frac{\pi}{2}\right)^{2n} |z|^{2n+1} \leq \frac{1}{(2n+1)!}\left(\frac{\pi}{2}\right)^{2n}.$$

Restul devine mai mic decât eroarea impusă pentru $n = 5$. Deci suma Taylor

$$S_{10}(z) = 1 - \frac{1}{2!}\left(\frac{\pi}{2}\right)^2 z^2 + \frac{1}{4!}\left(\frac{\pi}{2}\right)^4 z^4 - \frac{1}{6!}\left(\frac{\pi}{2}\right)^6 z^6 + \frac{1}{8!}\left(\frac{\pi}{2}\right)^8 z^8 - \frac{1}{10!}\left(\frac{\pi}{2}\right)^{10} z^{10}$$

realizează condiţia de eroare impusă şi avem $M = 10$ în (1.81). Înlocuim monoamele z^k din $S_{10}(z)$ prin expresiile lor în funcţie de polinoamele Cebâşev folosind relaţiile din tabelul 1.2. Se obţin următoarele valori ale coeficienţilor a_k din (1.82)

a_0	a_2	a_4	a_6	a_8	a_{10}
0.472	-0.499	$2.79 \cdot 10^{-2}$	$-5.96 \cdot 10^{-4}$	$6.69 \cdot 10^{-6}$	$-4.92 \cdot 10^{-8}$

Coeficienţii de indici impari sunt nuli. Avem $|a_8| + |a_{10}| < e = 10^{-5}$ astfel încât putem să trunchiem la $N = 6$ şi obţinem aproximarea

$$p_6(x) = 0.999993 - 0.499912x^2 + 0.0414876x^4 - 1.27122 \cdot 10^{-3}x^6$$

Figura 1.14: Variaţia erorii $e(z) = \cos z - p_6(z)$.

ce realizează eroarea impusă dar cu 40% mai puţine operaţii aritmetice decât S_{10}. Graficul erorii $\cos x - p_6(x)$ prezentat în figura (1.14) confirmă atingerea preciziei impuse.

1.3 Aproximarea în sensul celor mai mici pătrate

Reamintim că, în acest caz, criteriul de aproximare îl reprezintă minimizarea sumei (1.5)

$$S = \sum_{k=1}^{N} [y_k - g(x_k)]^2 = \text{minim}, \tag{1.83}$$

valorile (x_k, y_k) fiind date. Este convenabil ca aproximanta $g(x)$ să se pună sub forma (1.6)

$$g(x) = \sum_{j=1}^{n} a_j g_j(x)\,, \tag{1.84}$$

$g_j(x)$ fiind funcţii cunoscute, liniar independente, iar a_j, $j = 1, 2, \ldots, n$ parametri nedeterminaţi. Ca şi în 1.2, criteriul (1.83) pune o problemă de minimizare a erorii. Vom vedea cum câteva noţiuni mai avansate, considerate în 1.4, ne vor conduce la concluzia că atât criteriul mini-max cât şi cel al celor mai mici pătrate sunt doar formulări diferite ale unui acelaşi criteriu de aproximare optimală. Deocamdată vom considera însă doar aspectele simple ale teoriei.

Aproximarea în sensul celor mai mici pătrate este utilizată mai ales în cazul prelucrării datelor experimentale. În acest caz, nu se recomandă folosirea interpolării deoarece valorile măsurate conţin erori inerente, repartizate probabilistic

având caracterul unor perturbaţii care trebuie, dimpotrivă, eliminate[13]. Este deci evident că numărul de parametri n trebuie să fie mai mic decât numărul de noduri N, $n < N$.

Criteriul (1.83), ca şi 1.73, minimizează o eroare. În acest caz însă eroarea este scrisă ca o sumă de pătrate. Consecinţa esenţială a acestei alegeri este că $S(a_1, \ldots, a_n)$ este derivabilă, fapt ce poate fi exploatat pentru a determina coeficienţii a_i. Pentru ca $S(a_1, \ldots, a_n)$ să aibe un extrem trebuie ca derivatele parţiale în raport cu a_i să se anuleze

$$\partial S/\partial a_i = 0,\ i \in \overline{1,n}. \tag{1.85}$$

Observaţi introducerea unui indice i diferit de cel de însumare j din (1.84). Relaţiile (1.85) reprezintă un sistem de n ecuaţii cu n necunoscute. Deoarece S este o sumă de pătrate extremul dat de condiţiile (1.86) există şi reprezintă chiar un minim. Folosind expresiile (1.83) şi aranjând termenii se obţine

$$\sum_{j=1}^{n} a_j \left(\sum_{k=1}^{N} g_i(x_k) g_j(x_k) \right) = \sum_{k=1}^{N} y_k g_i(x_k),\ i \in \overline{1,n}\,, \tag{1.86}$$

adică un sistem de n ecuaţii pentru cei n parametrii a_j, $j \in \overline{1,n}$.

Forma sumei S se poate generaliza întrucâtva introducând o funcţie pondere $w(x)$, pozitivă, continuă, cu valori în intervalul $[0,1]$, care să ia în considerare unele distincţii privind importanţa valorilor luate în noduri. Se va scrie atunci

$$S = \sum_{k=1}^{N} w(x_k)[y_k - g(x_k)]^2 = \text{minim}, \tag{1.87}$$

iar sistemul (1.86) se va înlocui cu

$$\sum_{j=1}^{n} a_j \left(\sum_{k=1}^{N} w(x_k) g_i(x_k) g_j(x_k) \right) = \sum_{k=1}^{N} y_k w(x_k) g_i(x_k),\ i \in \overline{1,n}\ . \tag{1.88}$$

Sistemele (1.86) sau (1.88) se pot rezolva prin metodele din capitolul 5, de rezolvare a sistemelor de ecuaţii liniare. O observaţie importantă este legată de faptul că aceste sisteme pot ridica probleme legate de introducerea unor erori mari în calculul numeric, mai ales când diviziunile sunt egale[14]. Funcţiile $g_j(x)$ se pot alege din baza canonică

$$g_j(x) = x^{j-1}, \quad j \in \overline{1,n}\,, \tag{1.89}$$

sau baze formate din diverse polinoame ortogonale pe mulţimea discretă de puncte x_k, $k \in \overline{1,N}$ (vezi 1.4.2). Alegerea de polinoame ortogonale are avantajul considerabil al reducerii erorilor ce pot apare la rezolvarea sistemelor (1.86) sau

[13] Un exemplu ar fi eliminarea zgomotului de fond de la aparatura radio.

[14] În limbajul din capitolul 5, matricea sistemului (1.86) sau (1.88) este *rău condiţionată*, iar rezolvarea cere tehnici speciale cum ar fi descompunerea în valori singulare.

Figura 1.15: Abaterile geometrice e (în microni) înregistrate la prelucrarea mecanică a 1000 de piese.

(1.88). Deseori, din informaţii suplimentare, cunoaştem forma cea mai adecvată a funcţiilor $g_j(x)$. Un exemplu este prezentat la sfârşitul secţiunii. Folosirea unei combinaţii liniare (1.84) pentru funcţia aproximantă $g(x)$ este convenabilă deoarece conduce la un sistem liniar de ecuaţii pentru coeficienţii a_i. Se pot adopta însă şi alte forme, cu complicaţia posibilei apariţii a unui sistem neliniar de ecuaţii, mult mai dificil de rezolvat. Încheiem cu observaţia că nu s-a precizat încă ce grad al polinomului ar conduce la cea mai mică eroare, adică la cea mai mică valoare a lui S în (1.83). Un criteriu util pentru alegerea gradului este minimizarea expresiei

$$\bar{U} = S/(N-n) \tag{1.90}$$

prin varierea lui n.

Exemplul 1. În cadrul unei operaţii de prelucrare mecanică se înregistrează abaterile din fig. 1.15 de la cotele nominale. Sunt reprezentate şi marginile ce determină rebuturi. Se pune întrebarea dacă are loc vreo creştere sistematică a abaterilor datorită uzurii utilajului.

Rezolvare. Construim o aproximare liniară prin cele mai mici pătrate. Procedeul este denumit *regresie liniară*. Expresia pentru abaterile e funcţie de numărul de piese prelucrate p este

$$e = ap + b$$

iar coeficienţii a, b rezultă din rezolvarea sistemului

$$\begin{cases} \left(\sum_{k=1}^{N} p_k^2\right) a + \left(\sum_{k=1}^{N} p_k\right) b &= \left(\sum_{k=1}^{N} e_k p_k\right) \\ \left(\sum_{k=1}^{N} p_k\right) a + Nb &= \left(\sum_{k=1}^{N} e_k\right) \end{cases}$$

Se obţine $a = 1.11 \cdot 10^{-2}$ μ/piesă şi $b = -4.78$ μ ceea ce indică o creştere cu ~ 1 μ a abaterii la fiecare 100 de piese prelucrate, creştere datorată probabil uzurii utilajului. De asemenea din faptul că reglajul iniţial a fost efectuat astfel încât să se producă o abatere negativă de circa 5 μ se poate deduce că apariţia uzurii era prevăzută.

EXEMPLUL 2. O serie de măsurători chimice au furnizat ratele de reacţie din tabelul de mai jos. Din teoria reacţiilor chimice se cunoaşte dependenţa ratei de reacţie de temperatură

$$k = cT^{\eta} \exp(-K/\mathcal{R}T)\,.$$

Se cere estimarea parametrilor c, η, K pe baza măsurătorilor efectuate. Constanta gazelor $\mathcal{R}$ =8314 J/mol/K este cunoscută.

T	k	T	k	T	k
K	mol/cm^3	K	mol/cm^3	K	mol/cm^3
2000	8.4765e12	2600	5.0189e12	3200	3.3146e12
2100	7.6895e12	2700	4.6544e12	3300	3.1169e12
2200	7.0071e12	2800	4.3282e12	3400	2.9364e12
2300	6.4118e12	2900	4.0351e12	3500	2.7711e12
2400	5.8892e12	3000	3.7708e12	3600	2.6194e12
2500	5.4280e12	3100	3.5317e12	3700	2.4799e12

Rezolvare. Dependenţa $k(c, \eta, K)$ este neliniară. Putem însă logaritma relaţia de mai sus

$$\ln k = \ln c + \eta \ln T - K/\mathcal{R}T$$

şi reobţinem o dependenţă liniară de parametrii c, η, K. Se construieşte suma

$$S = \sum_{j=1}^{N} \left(\ln c + \eta \ln T_j - K/\mathcal{R}T_j - \ln k_j\right)^2 \ .$$

Condiţiile de extrem $\partial S/\partial(\ln c) = 0$, $\partial S/\partial\eta = 0$, $\partial S/\partial K = 0$ conduc la sistemul

$$\left\{\begin{array}{rl} N \ln c + \left(\sum_{j=1}^N \ln T_j\right)\eta - \left(\sum_{j=1}^N 1/T_j\right) K/\mathcal{R} & = \left(\sum_{j=1}^N \ln k_j\right) \\ \left(\sum_{j=1}^N \ln T_j\right)\ln c + \left(\sum_{j=1}^N \ln^2 T_j\right)\eta - \left(\sum_{j=1}^N \ln T_j/T_j\right) K/\mathcal{R} & = \left(\sum_{j=1}^N \ln k_j \ln T_j\right) \\ \left(\sum_{j=1}^N 1/T_j\right)\ln c + \left(\sum_{j=1}^N \ln T_j/T_j\right)\eta - \left(\sum_{j=1}^N 1/T_j^2\right) K/\mathcal{R} & = \left(\sum_{j=1}^N \ln k_j/T_j\right) \end{array}\right.$$

a cărui soluţie este $\ln c = 44.97$, $c = 3.39 \cdot 10^{19}$ mol/K^2/cm^3/s, $\eta = -2$, $K = 46000$ J/mol.

EXEMPLUL 3. Să presupunem date măsurători fizice ale energiei de vibraţie ale unei molecule de O_2. Se cunoaşte dependenţa energiei de temperatură

$$e = \frac{hv/kT}{\exp(hv/kT) - 1} RT$$

Se cere ν, frecvenţa fotonilor emişi la saltul între două nivele energetice.

Rezolvare. În acest caz nu avem nici o posibilitate de a liniariza problema. Notăm $x = hv/k$. Suma patratelor este

$$S = \sum_{j=1}^{N} \left[\frac{x/T_j}{\exp(x/T_j) - 1} RT_j - e_j\right]^2 ,$$

Tabelul 1.3: Coeficienţii polinomului aproximant din exemplul 4.

	a_1	a_2	a_3	a_4	a_5	a_6	a_7
Simplă	-526.0	4769	4059	1242	-1683	11.79	0.7434
Dublă	40.00	10.00	5.000	3.000	2.000	1.000	1.000

iar din $\partial S/\partial v = 0$ se obţine o ecuaţie neliniară de forma $f(x) = 0$, sau explicit

$$\sum_{j=1}^{N} \left\{ \left[\frac{x/T_j}{\exp(x/T_j) - 1} RT_j - e_j \right] \frac{[\exp(x/T_j) - 1] - x/T_j \exp(x/T_j)}{[\exp(x/T_j) - 1]^2} \right\} = 0$$

Determinarea soluţiei se poate efectua prin metodele din capitolul 4.

Exemplul 4. (Dorn) Dăm un exemplu al erorilor mari ce pot apărea atunci când nu lucrăm cu funcţii ortogonale. Generăm un set de noduri $\{x_k, y_k, k \in \overline{1, 15}\}$ unde luăm $x_k = k - 1$, $y_k = f(x_k)$ cu $f(x) = 40 + 10x + 5x^2 + 3x^3 + 2x^4 + x^5 + x^6$. Vom încerca reobţinerea coeficienţilor polinomului f prin metoda celor mai mici pătrate folosind alegerea (1.89), $g_j = x^{j-1}$, $j \in \overline{1, 7}$. Coeficienţii obţinuţi în urma rezolvării sistemului liniar[15] în simplă (32 biţi) şi dublă precizie (64 biţi) sunt prezentaţi în tabel 1.3. Acumularea rezultatelor parţiale s-a făcut în precizie extinsă (80 biţi) în ambele cazuri. Se poate observa că obţinem abateri mari de la valorile exacte ale coeficienţilor în simplă precizie. Mărirea preciziei furnizează rezultatul exact, astfel încât suntem asiguraţi că modul de calcul este corect şi abaterile din calculul în simpla precizie ţin de însăşi natura problemei. Exemplul este reluat la sfârşitul secţiunii 1.4 cu folosirea unor funcţii ortogonale pentru g_j pentru a vedea efectul benefic al acestora.

Să presupunem acum că nu am cunoaşte gradul polinomului ce ar conduce la o cea mai mică eroare. Rezolvând problema pentru diverse valori ale lui n se obţin următoarele valori pentru $S/(N - n)$

$n-1$	2	3	4	5	6	7	8	9
$S/(N-n)$	5.3E11	5.1E10	1.8E09	1.3E07	9.4E05	3.3E05	3.2E04	5.2E04

Se observă că cea mai mică valoare a sumei se obţine aproape de gradul polinomului f, $n = 8$.

1.4 Elemente de teoria aproximării

Din cele prezentate până acum, s-ar părea că problemele de aproximare prezintă multe aspecte *ad hoc*, lăsate la alegerea celui care abordează o problemă concretă sau alta. Pentru a putea vedea modul natural în care apar criteriile de aproximare discutate până acum avem nevoie de un cadru teoretic mai general. Efortul depus în asimilarea acestuia este pe deplin răsplătit în aprecierea unităţii problemelor de aproximare deja parcurse. În plus, deoarece aproximarea stă la baza multor altor algoritmi, cadrul teoretic va avea aplicaţii directe şi în alte

[15] Prin eliminare Gauss cu pivotare completă – vezi capitolul 5.

procedee numerice. Vom prefera o prezentare mai degrabă intuitivă decât riguroasă a noţiunilor de analiză funcţională utile teoriei aproximării. De asemenea, vom încerca formularea unei analogii geometrice utile asimilării acestor noţiuni.

1.4.1 Spaţii vectoriale

Astfel, forma (1.6) a funcţiei de aproximare $g(x)$ a fost aleasă astfel încât determinarea coeficienţilor a_k, $k \in \overline{1,N}$ să conducă la sisteme de ecuaţii liniare. S-a remarcat apoi că nici setul de funcţii $g_k(x)$ nu poate fi complet arbitrar, ajungându-se la sugerarea proprietăţii de independenţă liniară. De fapt, am ales elemente dintr-o structură bine cunoscută din algebră, aceea de *spaţiu vectorial* sau *spaţiu liniar*. Notăm un spaţiu vectorial oarecare prin $\mathcal{V}$. Elementele spaţiului se numesc *vectori*. Denumirea sugerează originea geometrică a noţiunilor, însă vom putea interpreta ca "vectori" şi funcţiile $g_k(x)$ sau funcţia $g(x)$. În definiţia acestei structuri algebrice intră operaţii de adunare între vectori şi de înmulţire a vectorilor cu numere reale sau complexe, denumite *scalari,* cum ar fi parametrii a_k. Mulţimea din care se iau scalarii o notăm cu S. De regulă avem scalari din mulţimea numerelor reale, $S = \mathbf{R}$, sau din mulţimea numerelor complexe, $S = \mathbf{C}$. Fără a repeta toate proprietăţile din definiţia unui spaţiu vectorial, vom reţine că prin înmulţirea unui scalar cu un vector, rezultatul este un vector, iar la adunarea vectorilor se obţin tot vectori.

Vectorii nenuli u_k, $k = \overline{1,N}$ se numesc *liniar dependenţi* dacă unul dintre ei se poate exprima ca o combinaţie liniară a celorlalţi, adică să existe scalarii a_k, *nu toţi nuli*, astfel încât să aibe loc egalitatea

$$\sum_{k=1}^{N} a_k u_k = 0 \ . \tag{1.91}$$

Dacă a_M este unul din coeficienţii nenuli, vectorul u_M se poate exprima ca o combinaţie liniară a celorlalţi

$$u_M = -\frac{1}{a_M}\left(\sum_{k=1}^{M-1} a_k u_k + \sum_{k=M+1}^{N} a_k u_k\right). \tag{1.92}$$

În egalitatea (1.91), 0 din membrul drept este vectorul nul al spaţiului, diferit de scalarul zero. Din context se poate înţelege întotdeauna semnificaţia lui 0. Dacă vectorii u_k nu pot fi puşi în relaţii de forma (1.91) decât atunci când scalarii a_k sunt toţi nuli, vectorii se vor numi *liniar independenţi*. Numărul maxim de vectori liniar independenţi dintr-un spaţiu vectorial se numeşte *dimensiunea spaţiului*. Această dimensiune poate fi finită sau infinită.

O dată stabilit cât de mare este un spaţiu vectorial $\mathcal{V}$, ne punem problema cum putem să descriem eficient un element oarecare $u \in \mathcal{V}$. Instrumentul adecvat este noţiunea de *bază* a unui spaţiu vectorial ce este orice mulţime de vectori liniar independenţi în număr egal cu dimensiunea spaţiului $\mathcal{V}$. Orice alt element al spaţiului poate fi exprimat ca o combinaţie liniară a elementelor din bază.

Într-adevăr, dacă $\{u_k, k \in \overline{1,N}\}$ este o bază a unui spaţiu vectorial de dimensiune N, orice alt vector u_{N+1} formează cu $\{u_k\}$ un set de vectori liniari dependenţi – altfel am avea $N+1$ vectori liniari independenţi şi dimensiunea spaţiului ar fi $N+1$ în loc de N. Vectorul u_{N+1} se poate exprima sub forma (1.92), ca o combinaţie liniară de vectorii din bază[16]. Imediat ce baza este stabilită un element oarecare din $\mathcal{V}$ poate fi descris doar prin cei N scalari $(a_1, a_2, \ldots, a_N)$.

Dăm câteva exemple utile de spaţii vectoriale.

1. Spaţiul euclidian de dimensiune 3. Avem 3 vectori $\{\vec{i}, \vec{j}, \vec{k}\}$ ce sunt liniar independenţi iar orice alt vector se poate exprima ca o combinaţie liniară a acestora, de exemplu $\vec{V} = \vec{i} + \vec{j} + \vec{k}/2$, sau pe scurt $\vec{V} \equiv (1, 1, 1/2)$.

2. Polinoamele $p_{N-1}(x)$ de grad cel mult egal cu $N-1$ formează un spaţiu vectorial de dimensiune N, pe care-l vom nota Π_{N-1}. Într-adevăr monoamele x^{k-1}, $k \in \overline{1,N}$ sunt liniar independente. Un alt element al spaţiului vectorial, de exemplu $p_2(x) = 1 + x + x^2/2$, se poate exprima ca o combinaţie liniară de monoame. Notaţia prin coeficienţii scalari este $p_2 \equiv (1, 1, 1/2)$.

3. Mulţimea $C^{(\infty)}(-\infty, \infty)$ a funcţiilor infinit derivabile pe axa reală formează un spaţiu de dimensiune infinită. Într-adevăr monoamele x^k, $k \in \mathbf{N}$ sunt liniar independente, iar k poate lua valori oricât de mari. Orice funcţie infinit derivabilă se poate scrie ca o combinaţie liniară de monoame conform teoremei de dezvoltare în serie Taylor. De exemplu $e^x = 1 + x + x^2/2 + x^3/6 + \ldots$ a cărei notaţie prin coeficienţi este $e^x \equiv (1, 1, 1/2, 1/6, \ldots)$.

4. Mulţimea $C^{(0)}[0, 1]$ a funcţiilor continue pe intervalul $[0, 1]$. Sunt liniar independente funcţiile $\sin 2k\pi x$, $\cos 2k\pi x$, $k \in \mathbf{Z}$. Conform teoremei de dezvoltare în serie Fourier, orice funcţie continuă se poate scrie ca o combinaţie liniară de sin şi cos, în particular $f(x) = (2\cos \pi x + \sin \pi x)\cos \pi x = 1 + \cos 2\pi x + \frac{1}{2}\sin 2\pi x$, sau notând doar coeficienţii $f \equiv (1, 1, 1/2)$.

Observaţi cum în toate exemplele de mai sus a apărut aceeaşi notaţie prin coeficienţi $(1, 1, 1/2)$ pentru elemente foarte diferite între ele. Notaţia ne permite stabilirea unor analogii între spaţiile mai abstracte de funcţii şi cel geometric familiar. Analogiile nu sunt demonstraţii matematice, dar vom prefera să fim oneşti asupra modului în care se ajunge la o demonstraţie – calea se bazează de cele mai multe ori pe o intuiţie ce este doar *apoi* verificată riguros.

1.4.2 Produsul scalar şi ortogonalitate

Să folosim analogia pentru a introduce o clasificare a bazelor. Începem cu considerarea spaţiului vectorial plan familiar. În fig. 1.16 sunt prezentate două

[16] Mai trebuie demonstrat că avem $a_{N+1} \neq 0$ astfel încât să putem scrie $u_{N+1} = -a_{N+1}^{-1} \sum_{k=1}^{N} a_k u_k$. Dacă prin absurd $a_{N+1} = 0$, am avea $\sum_{k=1}^{N} a_k u_k = 0$ cu unul dintre coeficienţii $a_k, k \in \overline{1,N}$ nenul astfel încât $\{u_k\}$ nu ar fi o bază în contradicţie cu ipoteza.

Figura 1.16: O baza neortogonală şi una ortonormată.

baze. Ne punem problema descrierii schimbării poziţiei unui element din A în B, traiectoria fiind paralelă cu primul vector din bază. În baza (a) descrierea este complicată: se modifică ambele coordonate şi în plus unităţile de măsură sunt diferite pe cele două direcţii. În baza (b) deplasarea se descrie simplu. Din geometrie, ştim că simplitatea provine din faptul că baza (b) este ortonormată. Am dori ca şi în cazul spaţiilor vectoriale abstracte să avem posibilitatea folosirii unor baze în care deplasările să poată fi descrise simplu. Pentru aceasta trebuie să generalizăm noţiunea geometrică de ortogonalitate prin introducerea unei funcţii care să ne arate cât de aproape de a fi ortogonali se află doi vectori. Funcţia căutată este denumită *produs scalar*. Vom nota produsul scalar a doi vectori u, v prin (u, v). Definiţia formală a produsului scalar este precizată de proprietăţile următoare, unde $u, v, w \in \mathcal{V}$, $a, b \in S$:

1. $(\underline{\circ}, \underline{\circ}) : \mathcal{V} \times \mathcal{V} \to S$;
2. $(u, u) \geq 0$ şi $(u, u) = 0 \Rightarrow u = 0$;
3. $(u, v) = (v, u)^*$;
4. $(au + bv, w) = a(u, w) + b(v, w)$.

Ortogonalitatea a doi vectori nenuli se poate acum defini: $u, v \neq 0$ sunt *ortogonali* dacă $(u, v) = 0$. O bază de vectori $\{u_k, k \in \overline{1, N}\}$ este *ortogonală* dacă $\forall j \neq k,\ j, k \in \overline{1, N}\ (u_j, u_k) = 0$ şi *ortonormată* dacă este ortogonală şi în plus $\forall k,\ k \in \overline{1, N}\ (u_k, u_k) = 1$. Ortogonalitatea unor vectori implică şi independenţa liniară. Prin reducere la absurd, dacă vectorii nenuli $\{u_k, k \in \overline{1, N}\}$ ar fi ortogonali doi câte doi, dar liniar dependenţi atunci în $\sum_{k=1}^{N} a_k u_k = 0$ am avea un coeficient nenul, fie a_M acesta. Efectuând produsul scalar al sumei cu u_M rezultă $a_M(u_M, u_M) = 0$ datorită ortogonalităţii. Dar $u_M \neq 0$ şi deci $(u_M, u_M) > 0$ şi rezultă contradicţia $a_M = 0$.

Avem acum definită general noţiunea de ortogonalitate. Am văzut că bazele ortonormate permit descrieri mai simple ale elementelor unui spaţiu vectorial. Ne punem întrebarea: putem dintr-o bază oarecare $\{u_k, k \in \overline{1, N}\}$ să obţinem

una ortonormată $\{g_k, k \in \overline{1,N}\}$? Răspunsul este pozitiv şi furnizat de următorul algoritm denumit *procedeul Gram-Schmidt*

$$\begin{array}{l} g_1 = u_1/(u_1,u_1) \\ k = 2:N \\ \quad \left[\begin{array}{l} v_k = u_k - \sum_{j=1}^{k-1}(u_k,g_j)g_j \\ g_k = v_k/(v_k,v_k) \end{array} \right. . \end{array}$$

Geometric, în etapa k, se scad din vectorul u_k componentele sale în direcţiile $g_1, g_2, \ldots, g_{k-1}$ ce sunt deja ortogonale. Vectorul v_k ce rămâne din diferenţă este sigur ortogonal pe $g_1, g_2, \ldots, g_{k-1}$. Mai rămâne doar de adus la lungime unitară prin împărţirea $g_k = v_k/(v_k, v_k)$. Să considerăm acum câteva exemple de produse scalare şi baze ortonormate utile în aplicaţii.

Polinoame ortogonale.

Ortonormarea monoamelor $\{1, x, x^2, \ldots\}$ produce o bază $\{p_0(x), p_1(x), p_2(x), \ldots\}$ ale cărei elemente sunt evident polinoame. Aceste *polinoame ortogonale* au multe aplicaţii iar expresiile lor rezultă din procedeul Gram-Schmidt ca fiind

$$p_0(x) = 1,\ p_1(x) = x - \alpha_1, \tag{1.93}$$

pentru primele două elemente, celelalte elemente ale bazei fiind definite de relaţia recursivă

$$p_n(x) = (x - \alpha_n)p_{n-1}(x) - \beta_n p_{n-2}(x)\ ,\ n \geq 2\ . \tag{1.94}$$

Coeficienţii α_n, β_n sunt dependenţi de produsul scalar ales fiind precizaţi de relaţiile

$$\alpha_n = (xp_{n-1}, p_{n-1})/(p_{n-1}, p_{n-1}), \quad \beta_n = (xp_{n-1}, p_{n-2})/(p_{n-2}, p_{n-2})\ . \tag{1.95}$$

Relaţiile (1.93)-(1.95) se pot verifica uşor prin inducţie. Urmărind relaţiile de mai sus se poate observa că vom avea totdeauna un coeficient unitar al termenului de grad maxim, adică polinoamele obţinute vor fi sub forma lor *monică* (vezi 1.2.1). De regulă asupra formei monice se impune o *condiţie de standardizare* ce revine la înmulţirea cu o constantă. Să considerăm cazurile cele mai des întâlnite.

1. *Polinoamele Legendre* $P_n(x)$. Se consideră $\mathcal{V} = C^{(0)}[-1,1]$ şi se adoptă produsul scalar

$$(u,v) = \int_{-1}^{1} u(x)v(x)dx\ . \tag{1.96}$$

Condiţia de standardizare este $P_n(1) = 1$. Deci $P_0(x) = 1$, $\alpha_1 = (x,1)/(1,1) = 0$ şi deci $P_1(x) = x$. Apoi $\alpha_2 = (x^2,x)/(x,x) = 0$, $\beta_2 = (x^2,1)/(1,1) = 1/3$ şi deci $P_2(x) = x^2 - 1/3$. Impunerea condiţiei de standardizare conduce la $P_2(x) = 3x^2/2 - 1/2$. Primele câteva polinoame Legendre sunt date în tabelul 1.4.

2. *Polinoamele Cebâşev* $T_n(x)$. Se consideră tot $\mathcal{V} = C^{(0)}[-1,1]$ dar se adoptă produsul scalar

$$(u,v) = \int_{-1}^{1} u(x)v(x)(1-x^2)^{-1/2}\,dx\ . \tag{1.97}$$

Observaţi că diferenţa faţă de produsul scalar anterior (1.96) este apariţia unei *funcţii de pondere* $w(x) = (1-x^2)^{-1/2}$. Intuitiv, ponderea adoptată acordă o mai mare importanţă punctelor de la capetele de interval ± 1. Condiţia de standardizare este $T_n(1) = 1$. Expresiile lor au fost deja prezentate în tabelul 1.2. Să aplicăm însă (1.93)-(1.95) pentru a vedea că într-adevăr reobţinem acele expresii. Avem $T_0(x) = 1$, $\alpha_1 = (x,1)/(1,1)$. Apar integrale ce se calculează prin substituţia trigonometrică $x = \cos\theta$

$$(x,1) = \int_{-1}^{1} x(1-x^2)^{-1/2}\,dx = \int_{0}^{\pi} \cos\theta d\theta = 0$$

şi deci $T_1(x) = x$. Apoi $\alpha_2 = (x^2,x)/(x,x) = 0$, $\beta_2 = (x^2,1)/(1,1) = 1/2$ deci $T_2(x) = x^2 - 1/2$. Am reobţinut forma monică a polinomului Cebâşev pentru $n = 2$. Aplicarea condiţiei de standardizare conduce la $T_2(x) = 2x^2 - 1$, adică exact forma din tabelul 1.2.

3. *Polinoamele Laguerre* $L_n(x)$. Se consideră $\mathcal{V} = C^{(0)}[0,\infty)$ şi se adoptă produsul scalar

$$(u,v) = \int_{0}^{\infty} u(x)v(x)\,\exp(-x)\,dx\ . \tag{1.98}$$

Domeniul de definiţie al elementelor din acest spaţiu vectorial este acum infinit la dreapta. Observaţi că un astfel de spaţiu permite tratarea problemelor de aproximare a unor funcţii definite pe toată semiaxa reală pozitivă. Ponderea $w(x) = \exp(-x)$ atenuează comportarea la infinit a funcţiilor $u(x), v(x)$ astfel încât integrala (1.98) să conveargă pentru $u(x), v(x)$ ce nu cresc mai repede decât un polinom. Condiţia de standardizare este: coeficientul termenului de grad maxim să fie 2^n. Expresiile primelor câteva polinoame sunt date în tabelul 1.4.

4. *Polinoamele Hermite* $H_n(x)$. Se consideră $\mathcal{V} = C^{(0)}(-\infty,\infty)$ şi se adoptă produsul scalar

$$(u,v) = \int_{-\infty}^{\infty} u(x)v(x)\,\exp(-x^2)\,dx\ . \tag{1.99}$$

Evident acest spaţiu vectorial reprezintă extinderea celui anterior la toată axa reală. Condiţia de standardizare este: coeficientul termenului de grad maxim să fie $(-1)^n/n!$.

Tabelul 1.4: Primele câteva polinoame ortogonale

n	P_n	H_n	L_n
0	1	1	1
1	x	$2x$	$-x$
2	$(-1+3x^2)/2$	$-2+4x^2$	$(2-4x+x^2)/2$
3	$(-3x+5x^3)/2$	$-12x+8x^3$	$(6-18x+9x^2-x^3)/6$
4	$(3-30x^2+35x^4)/8$	$12-48x^2+16x^4$	$(24-96x+72x^2-16x^3+x^4)/24$

Ortogonalitate pe mulţimi discrete.

În paragraful anterior s-au introdus produse scalare definite prin integrale, adecvate situaţiilor în care căutăm aproximări pe intervale continue. Dacă însă ne interesează o aproximare punctuală, în nodurile $\{x_k, k \in \overline{1,N}\}$, atunci este mai util să introducem produse scalare definite prin sume. Vom introduce acum *polinoame ortogonale discrete*, rezultate din aplicarea procedeului Gram-Schmidt asupra bazei $\{1, x, x^2, \dots\}$ utilizând aceste noi produse scalare. O primă răsplată a abordării abstracte a noţiunii de ortogonalitate este că relaţiile (1.93)-(1.95) rămân valabile deoarece ele nu fac referire la forma particulară a produsului scalar. Polinoamele ortogonale obţinute de data aceasta vor fi dependente de nodurile alese $\{x_k, k \in \overline{1,N}\}$ şi vom include mulţimea nodurilor ca un argument pentru a sublinia acest fapt.

1. *Polinoamele Legendre discrete* $P_n(x;\{x_k\})$. Se consideră $\mathcal{V} = C^{(0)}[-1,1]$, nodurile se consideră în acest interval $-1 \le x_k \le 1$ şi se adoptă produsul scalar

$$(u,v) = \sum_{k=1}^{N} u(x_k)v(x_k) \ . \tag{1.100}$$

2. *Polinoamele Cebâşev discrete* $T_n(x;\{x_k\})$. Se consideră $\mathcal{V} = C^{(0)}(-1,1)$, nodurile se iau $-1 < x_k < 1$ şi se adoptă produsul scalar

$$(u,v) = \sum_{k=1}^{N} u(x_k)v(x_k)(1-x_k^2)^{-1/2} \ . \tag{1.101}$$

3. *Polinoamele Hermite discrete* $H_n(x;\{x_k\})$. Se consideră $\mathcal{V} = C^{(0)}[0,\infty)$ şi se adoptă produsul scalar

$$(u,v) = \sum_{k=1}^{N} u(x_k)v(x_k)\exp(-x_k) \ . \tag{1.102}$$

4. *Polinoamele Laguerre discrete* $L_n(x;\{x_k\})$. Se consideră tot $\mathcal{V} = C^{(0)}(-\infty,\infty)$ şi se adoptă produsul scalar

$$(u,v) = \sum_{k=1}^{N} u(x_k)v(x_k)\exp(-x_k^2) \ . \tag{1.103}$$

1.4.3 Norme, operatori şi funcţionale

Noţiunea de ortogonalitate din secţiunea anterioară ne-a permis să descriem orientarea relativă a doi vectori. Mai avem nevoie de un instrument ce să poată măsura mărimea unui vector, instrument furnizat de noţiunea matematică de *normă*. Norma unui vector $u \in \mathcal{V}$ se notează $\|u\|$ şi este un număr real pozitiv. Cu $a \in S$, $u, v \in \mathcal{V}$, definiţia formală este precizată de

1. $\|\underline{\circ}\| : \mathcal{V} \to \mathbf{R}_+$;
2. $\|u\| = 0 \Leftrightarrow u = 0$;
3. $\|u+v\| \le \|u\| + \|v\|$; (inegalitatea triunghiului)
4. $\|au\| = |a|\,\|u\|$.

O clasă importantă de norme este definită prin

$$\|u\|_p = \left(\int_a^b |u(x)|^p\,dx\right)^{1/p} \tag{1.104}$$

pe cazul continuu, cu analogul discret pe mulţimea de noduri $\{x_k, k \in \overline{1,N}\}$

$$\|u\|_p = \left(\sum_{k=1}^{N} |u(x_k)|^p\right)^{1/p} \ . \tag{1.105}$$

Presupunem ca aceste integrale şi sume există. Acestea se numesc *norme p* continue sau discrete, respectiv. Se observă imediat că norma 2 discretă a unui vector obişnuit $u \in \mathbf{R}^N$ corespunde definiţiei clasice a modulului unui vector $\|u\|_2 = (\sum_{k=1}^{N} u_k^2)^{1/2}$, astfel încât ne putem întări intuiţia perceperii normei ca "mărime" a unui element dintr-un spaţiu vectorial. Se poate demonstra că atunci când $p \to \infty$ avem

$$\|u\|_\infty = \sup_{x\in[a,b]} |u(x)| \tag{1.106}$$

în cazul continuu şi

$$\|u\|_\infty = \max_{k\in\overline{1,N}} |u(x_k)| \tag{1.107}$$

Figura 1.17: Discurile $\|x\|_p \leq 1$.

în cazul discret. Un alt exemplu important de normă este norma indusă de produsul scalar, în cazul în care acesta din urmă este deja definit

$$\|u\| = (u, u)^{1/2}. \tag{1.108}$$

Se poate verifica satisfacerea proprietăţilor unei norme. În particular inegalitatea triunghiului rezultă din forma generală a inegalităţii Cauchy-Buniacovski

$$|(u, v)| \leq \|u\| \cdot \|v\| \ . \tag{1.109}$$

Este important să se formeze o intuiţie a semnificaţiei adoptării unei norme sau alta. Apelăm la spaţiul vectorial cel mai familiar, al vectorilor din planul $\mathbf{R}^2$ şi desenăm în fig. 1.17 discurile ce satisfac $\|x\|_p \leq 1$, pentru $p = 1, 2, 3$ şi $p = \infty$.

Distanţa dintre doi vectori se poate acum introduce imediat ca fiind o funcţie $d : \mathcal{V} \times \mathcal{V} \rightarrow \mathbf{R}_+$ definită prin

$$d(u, v) = \|u - v\| \ .$$

Altfel spus distanţa dintre doi vectori este mărimea vectorului diferenţă. Definirea unei distanţe permite introducerea vecinătăţilor şi a noţiunilor conexe, în particular continuitatea.

Norma şi produsul scalar sunt exemple de aplicaţii sau funcţii prin care unor elemente ale unui spaţiu vectorial li se face să corespundă un număr. Se pot defini şi alte funcţii ale căror valori pot fi scalari sau chiar vectori. O aplicaţie definită pe un spaţiu vectorial care ia valori scalare se numeşte *funcţională*. Spre exemplu, pe spaţiul vectorial al funcţiilor continue $C^{(0)}[a, b]$, $\int_a^b u(x)dx$ este o funcţională deoarece integrarea face să corespundă un număr fiecărei funcţii u . O funcţie definită pe un spaţiu vectorial, cu valori în alt spaţiu vectorial (eventual acelaşi) e denumită *operator*. Spre exemplu, dacă $\mathbf{R}^n$ este spaţiul vectorilor coloană cu n componente $(x_1, x_2, ..., x_n)^T$, atunci înmulţirea vectorului cu o matrice pătrată cu $n \times n$ elemente poate fi considerată un operator.

Cu ajutorul normelor, putem compara şi aplicaţiile definite pe spaţii vectoriale. Să considerăm că aplicaţia $f : \mathcal{V}_1 \rightarrow \mathcal{V}_2$ este liniară, adică $\forall a, b \in S$, $u, v \in \mathcal{V}_1$ avem

$$f(au + bv) = af(u) + bf(v) \ .$$

O aplicaţie liniară f este *mărginită* dacă există un număr real pozitiv M astfel încât $\|f(u)\| \leq M\,\|u\|$. Cu alte cuvinte, norma valorilor aplicaţiei liniare este raportată la norma vectorului. Cel mai mic număr M pentru care are loc inegalitatea se numeşte *norma aplicaţiei* f şi se notează $\|f\|$. Prin urmare, putem scrie

$$\|f(u)\| \leq \|f\| \cdot \|u\| \tag{1.110}$$

pentru orice aplicaţie liniară mărginită. Mai mult, o aplicaţie liniară mărginită este şi continuă, deci (1.110) implică şi continuitatea.

1.4.4 Problema generală a celei mai bune aproximări

Să vedem acum modul în care noţiunile introduse permit o abordare generală a problemei de aproximare. Reamintin problema de aproximare: avem o funcţie f complicată sau incomplet cunoscută şi dorim găsirea unei aproximante g. Am dori ca diferenţa dintre f şi g să fie cât mai mică posibilă, adică să minimizăm distanţa

$$d(f,g) = \|f-g\| \ . \tag{1.111}$$

Din această formulare generală, putem reobţine criteriile de aproximare menţionate la începutul capitolului. Dacă adoptăm oricare dintre normele discrete şi impunem realizarea unei erori nule, $d(f,g)=0$, obţinem

$$\|f-g\|_p = \left(\sum_{k=1}^{N} |f(x_k)-g(x_k)|^p\right)^{1/p} = 0\,,$$

ceea ce nu se poate realiza decât dacă

$$g(x_k) = y_k,\ k \in \overline{1,N}\,,$$

cu $y_k = f(x_k)$ sau exact condiţia de interpolare (1.1). Dacă adoptăm norma ∞ pe cazul continuu, problema revine la a minimiza

$$\max_{x\in[a,b]} |f(x)-g(x)|\ ,$$

sau criteriul mini-max (1.3). Alegerea normei ∞ discrete conduce la criteriul mini-max discret (1.4). Alegerea normei 2 pe cazul discret conduce la minimizarea sumei

$$S = \sum_{k=1}^{N} [y_k - g(x_k)]^2\,,$$

adică criteriul celor mai mici pătrate (1.5).

Teoria generală ne furnizează cadrul de a pune anumite întrebări peste care am sărit în prezentarea elementară anterioară. Există o cea mai bună aproximare? Dacă există, cărui spaţiu vectorial aparţine? Putem construi un şir de

Figura 1.18: Cea mai bună aproximare este ortogonală pe subspaţiul aproximantei.

aproximaţii care să conveargă la cea mai bună aproximare? Nu vom parcurge aici în detaliu aceste chestiuni. Vom face mai degrabă o discuţie calitativă, apelând la intuiţia geometrică din spaţiul vectorial euclidian.

Începem printr-un exemplu simplu din geometria plană. Să presupunem că dorim să aproximăm cât mai bine punctul $X(a,b)$ din plan folosind elemente doar de pe axa x_1, adică de forma $(x_1,0)$ (fig. 1.18). Scriem

$$d(X,P) = \|X-P\| = \text{minim},$$

ceea ce conduce la $d(X,P) = \left[(a-\xi)^2+b^2\right]^{1/2}$ =minim, pentru norma 2 discretă. Este clar că cea mai mică abatere dintre toate punctele pe axa x_1 se obţine pentru punctul P ce are proprietatea că XP este *ortogonal* pe axa x_1. Axa x_1 este un subspaţiu al planului, iar P parcurge doar acest subspaţiu. Cea mai bună "aproximare" a lui X de către P se obţine atunci când *diferenţa* $X-P$ *este ortogonală pe subspaţiul parcurs de* P.

Concluzia de mai este general valabilă într-un spaţiu în care avem definit un produs scalar, iar norma rezultă din acest produs scalar conform (1.108). De exemplu, să adoptăm spaţiul $C^{(\infty)}(-\infty,\infty)$ ce are baza $\{1,x,x^2,\dots\}$. Problema analoagă celei anterioare este să se aproximeze funcţia $X = a+bx$ cât mai bine cu o constantă $P=\xi$. Reprezentarea prin coordonate $X \equiv (a,b,0,\dots)$, $P \equiv (\xi,0,0,\dots)$ ne convinge imediat că avem aceeaşi problemă ca în cazul anterior, iar soluţia este dată şi în acest caz de condiţia de ortogonalitate. Este educativă demonstraţia generală a acestui rezultat.

Teoremă. Fie $\mathcal{V}$ un spaţiu vectorial în care avem definit un produs scalar, şi $\mathcal{S}$ un subspaţiu al lui $\mathcal{V}$. Dacă $v-u$ este ortogonal pe orice w din $\mathcal{S}$ atunci u este cea mai bună aproximare a lui v prin elemente din $\mathcal{S}$, anume u realizează minimul distanţei $d(u,v)$.

Demonstraţie. Să vedem dacă vreun alt $w \in \mathcal{S}$ realizează o distanţă mai mică:

$$d^2(v,w) = (v-w,v-w) = (v-u+u-w,v-u+u-w) = (v-u,v-u)+$$
$$2(u-w,v-u)+(u-w,u-w) = \|v-u\|^2+\|u-w\|^2+2(u-w,v-u).$$

Dar $v-u$ este ortogonal pe orice element din $\mathcal{S}$, în particular pe $u-w$ deci $(u-w, v-u)=0$ şi avem $d^2(v,w)=\|v-u\|^2+\|u-w\|^2 \geq \|v-u\|^2$, adică distanţa de la orice alt element w la v este mai mare decât cea de la u la v. **2**

Ortogonalitatea pe un subspaţiu de dimensiune n se verifică prin ortogonalitatea pe elementele unei baze a acelui subspaţiu, cel mai eficient chiar pe una ortonormată $\{g_1, g_2, \ldots, g_n\}$. Condiţia de cea mai bună aproximantă g a unei funcţii f dintr-un spaţiu $\mathcal{V}$ cu produs scalar se scrie aşadar

$$(f-g, g_k)=0,\ k \in \overline{1,n}\,. \tag{1.112}$$

Observaţi că (1.86) sau (1.88) se pot obţine din condiţia generală (1.112) prin alegerea produsului scalar obişnuit între vectori din $\mathbf{R}^n$ cu ponderea w. Scriind aproximanta sub forma unei combinaţii liniare $g=\sum_{j=1}^{n} a_j g_j$ sistemul (1.112) se poate rezolva pentru coeficienţii a_k

$$a_k=(f,g_k)/(g_k,g_k),\ k \in \overline{1,n}, \tag{1.113}$$

ce sunt numiţi *coeficienţi Fourier generalizaţi*. De exemplu alegerea bazei trigonometrice $\cos(2\pi kx)$, $k \in \overline{0,N}$; $\sin(2\pi mx)$, $m \in \overline{1,N-1}$ definită pe punctele discrete $\{x_j=j/2N, j \in \overline{0,2N-1}\}$ şi a produsului scalar obişnuit permite obţinerea relaţiilor (1.58) folosind rezultatul general (1.113).

Rezultate ca cele de mai sus arată utilitatea spaţiilor vectoriale în care se introduce un produs scalar şi norma indusă prin relaţia (1.108). Un astfel de spaţiu se numeşte *spaţiu prehilbertian*. Într-un spaţiu prehilbertian putem obţine pentru un n fixat coeficienţii aproximantei $g=\sum_{j=1}^{n} a_j g_j$ prin relaţiile (1.113). Pe măsură ce n creşte, am dori ca g să rămână în spaţiul considerat. Acest deziderat conduce natural la spaţii Hilbert $\mathcal{H}$, definite prin faptul că limitele şirurilor fundamentale aparţin şi ele lui $\mathcal{H}$. Spaţiile vectoriale din exemplele din 1.4.1 sunt spaţii de acest tip, ele numindu-se *complete*. Aceste noţiuni sunt deosebit de importante pentru procedeele numerice şi merită să zăbovim puţin.

Dacă revedeţi exemplele prezentate până acum, veţi observa că deseori s-a calculat o eroare faţă de un rezultat exact *cunoscut*. În exemplificarea unui procedeu numeric alegerea unei probleme la care cunoaştem deja răspunsul este perfect acceptabilă. Dar în aplicaţiile reale *nu* cunoaştem răspunsul exact. Cum putem, în asemenea condiţii, să ne dăm seama de convergenţa unui algoritm numeric? Noţiunile de mai sus furnizează cadrul matematic în care putem da un răspuns la această întrebare. Reamintim definiţia convergenţei unui şir către o limită: a este limită a şirului $\{a_n\}$ cu $n \in \mathbf{N}$ dacă $\forall \varepsilon>0$ există un N_ε astfel încât $|a_n-a|<\varepsilon$ pentru $n>N_\varepsilon$. Observaţi că în această definiţie a convergenţei *valoarea limitei se presupune cunoscută*. În afară de această definiţie se mai introduce în analiză noţiunea de *şir fundamental* sau Cauchy $\{a_n\}$ caracterizat prin faptul că şirul este convergent dacă $\forall \varepsilon>0$, $p \in \mathbf{N}$ există un N_ε astfel încât $|a_{n+p}-a_n|<\varepsilon$ pentru $n>N_\varepsilon$. Observaţi elementul esenţial al definiţiei unui şir fundamental: convergenţa este definită prin aproprierea *termenilor succesivi*. Pentru verificarea convergenţei *nu* este nevoie să cunoaştem limita şirului. În

aplicaţii, aceasta este situaţia uzuală. Putem obţine aproximaţii succesive, dar nu, de regulă, şi limita exactă a şirului de aproximaţii. Lucrând în spaţii Hilbert vom avea un bun indiciu al convergenţei, atunci când aproximaţiile succesive devin apropriate între ele.

Spaţiile Hilbert au multe alte proprietăţile utile aplicaţiilor numerice. Două exemple ar fi: (1) *teorema de reprezentare Fréchet-Riesz* ce ne asigură că rezultatul aplicării unei funcţionale f (e.g. integrala definită) asupra unui element $u \in \mathcal{H}$ este el însuşi exprimabil ca un produs scalar $f(u) = (u, v_f)$, unde v_f este un element al spaţiului care depinde de f ; (2) teoreme generale asupra proprietăţilor operatorilor printre care se remarcă cele de descompunere spectrală.

EXEMPLU. Reluăm exemplul 4 din 1.3, de data aceasta folosind ca bază un set de polinoame ortogonale definite pe mulţimea discretă $\{x_k = k-1, k \in \overline{1,N}\}$, cu $N = 15$. Funcţia aproximată este polinomul $f(x) = 40 + 10x + 5x^2 + 3x^3 + 2x^4 + x^5 + x^6$. Cum nu avem vreun motiv să acordăm vreunui nod o importanţă deosebită luăm ponderea $w = 1$ şi adoptăm polinoamele Legendre discrete. Acestea sunt definite pe $[-1, 1]$ astfel încât este nevoie să introducem transformarea $z = 2x/(N-1)-1$. Funcţia aproximantă se scrie $g(z) = \sum_{j=0}^{n} a_j P_j(z)$ cu $n = 6$. Conform relaţiilor de recurenţă (1.93)-(1.95) primele 7 polinoame Legendre discrete sunt $P_0(z) = 1$, $P_1(z) = z$, $P_2(z) = -\frac{8}{21} + z^2$, $P_3(z) = -\frac{167}{245}z + z^3$, $P_4(z) = \frac{1296}{12005} - \frac{331}{343}z^2 + z^4$, $P_5(z) = \frac{44252}{151263}z - \frac{545}{441}z^3 + z^5$, $P_6(z) = -\frac{36000}{1294139} + \frac{2042}{3773}z^2 - \frac{115}{77}z^4 + z^6$.

Coeficienţii a_k rezultă imediat din (1.113)

$$a_j = \left[\sum_{k=1}^{N} f(z_k)P_j(z_k)\right] / \left[\sum_{k=1}^{N} P_j^2(z_k)\right] \ .$$

Efectuând calculul în numere raţionale a produselor scalare rezultă coeficienţii

$$a_0 = \tfrac{75236936}{1764735} \quad a_1 = \tfrac{1058398}{84035} \quad a_2 = \tfrac{18800}{2401} \quad a_3 = \tfrac{1868}{441} \quad a_4 = \tfrac{269}{77} \quad a_5 = 1 \quad a_6 = 1$$

ce introduşi în combinaţia liniară de mai sus dau exact funcţia f. Calculul în simplă precizie furnizează valorile coeficienţilor a_j

$$\begin{array}{llllllll} a_0 = & 42.633556 & a_1 = & 12.594727 & a_2 = & 7.8300747 & a_3 = & 4.2358241 \\ a_4 = & 3.4935030 & a_5 = & 0.99998158 & a_6 = & .99996691 & & \end{array}$$

cu o eroare relativă maximă de $\varepsilon = 3 \cdot 10^{-5}$ faţă de valorile exacte. Se observă că folosirea unei baze ortogonale a permis obţinerea unei erori mici lucrând chiar în simplă precizie.

Capitolul 2

Derivarea şi integrarea numerică

2.1 Derivarea numerică

Pentru derivarea numerică, punctul de plecare îl constituie tot găsirea unei funcţii aproximante. Ideea constă în aproximarea derivatei funcţiei date cu derivata aproximantei. Cele mai uzuale formule de calcul se obţin plecând de la aproximarea prin interpolare. Utilizând aceeaşi diviziune a intervalului pe care se află punctele în care se doreşte calculul derivatei, x_i, $i = 1, 2, \ldots, N$, vom scrie derivata sub forma

$$y' \equiv \frac{df}{dx} = \frac{dg}{dx} + \frac{dR}{dx}\,, \tag{2.1}$$

unde g este aproximanta iar R restul (eroarea).

În formulele de aproximare numerică a derivatei, se utilizează de regulă valorile funcţiei f în câteva din punctele x_i. Fie h distanţa tipică între aceste abscise. Un aspect important al aproximării numerice este stabilirea *ordinului* erorii. Eroarea se poate scrie sub forma

$$e = \frac{df}{dx} - \frac{dg}{dx} = \frac{dR}{dx} = O(h^k)\,. \tag{2.2}$$

Simbolul O este folosit pentru a elimina constantele ce înmulţesc expresia de interes de sub paranteze, în cazul de faţă h^k. Ordinul erorii este exponentul k. Se observă ca la micşorarea pasului h, eroarea e va scădea mai repede cu cât ordinul erorii k este mai mare. Spre exemplu, pentru $k = 1$ o înjumătăţire a pasului conduce tipic la o eroare de două ori mai mică, pe când pentru $k = 2$ eroarea scade de patru ori.

2.1.1 Derivate folosind polinoame de interpolare

În cazul interpolării polinomiale, cu diviziuni egale, se poate scrie

$$x = x_{i-1} + \alpha h, \quad \alpha \in [0, n], \quad n \in \overline{1, N-1}\,, \tag{2.3}$$

$$g(x) = y_{i-1} + \alpha \Delta y_{i-1} + C_\alpha^2 \Delta^2 y_{i-1} + \ldots + C_\alpha^n \Delta^n y_{i-1} \equiv p_n(x) \tag{2.4}$$

$$R = R_n(x) = h^{n+1} C_\alpha^{n+1} f^{(n+1)}(\xi)\,. \tag{2.5}$$

Se observă că există doi parametri la dispoziţie:

1. punctul x_{i-1} faţă de care calculăm diferenţele la dreapta;
2. gradul n al polinomului de interpolare.

Derivata (2.1) se scrie

$$y' = \frac{df}{dx} = \frac{1}{h}\frac{df}{d\alpha} = \frac{1}{h}\frac{dp_n}{d\alpha} + \frac{1}{h}\frac{dR_n}{d\alpha}\,. \tag{2.6}$$

Prezenţa pasului h la numitor în relaţia (2.6) sugerează tendinţa creşterii erorii la derivarea aproximativă. Astfel, dacă eroarea la interpolare era de ordinul $n+1$, $R = O(h^{n+1})$, eroarea la derivare este ordinul n, $R' = O(h^n)$, datorită împărţirii la h. De aceea, se iau o serie de precauţiuni ca:

1. utilizarea unor polinoame de aproximare de grad nu prea mare ($n < 7$) (pentru a preveni apariţia vreunei comportări prezise de teorema lui Faber, vezi 1.1.1);
2. centrarea punctului în care se face calculul pe intervalul $(x_{i-1}, x_{i-1} + nh)$;
3. calculul derivatei folosind polinoame de grade diferite – dacă diferenţele sunt mari, se iau precauţiuni suplimentare (micşorarea pasului, aproximarea spline sau prin alte metode de aproximare, etc.).

Considerăm acum câteva cazuri particulare utile.

Cazul $n = 1$.

Din relaţiile (2.4)-(2.6) se obţine

$$y' = \frac{1}{h}\Delta y_{i-1} + \frac{h}{2}\frac{d}{d\alpha}\left[\alpha(\alpha-1)f''(\xi)\right] . \tag{2.7}$$

Pentru valorile $\alpha = 0$ şi $\alpha = 1$, ce corespund la capetele intervalului (x_{i-1}, x_i) se obţine

$$y'_{i-1} = \frac{y_i - y_{i-1}}{h} - \frac{h}{2}f''(\xi), \tag{2.8}$$

$$y'_i = \frac{y_i - y_{i-1}}{h} + \frac{h}{2}f''(\xi) . \tag{2.9}$$

Aproximaţia (2.8) utilizează punctul x_i ce se află la dreapta punctului x_{i-1} în care se evaluează derivata şi se numeşte, uzual, *formulă la dreapta*. Analog (2.9) se numeşte *formulă la stânga*. Ambele formule au o eroare de ordinul întâi. Spunem pe scurt că sunt *formule de ordinul I*.

Cazul $n = 2$.

În acest caz, din (2.4)-(2.6), se obţine

$$y' = \frac{1}{h}\Delta y_{i-1} + \frac{2\alpha-1}{2h}\Delta^2 y_{i-1} + \frac{h^2}{6}\frac{d}{d\alpha}\left[\alpha(\alpha-1)(\alpha-2)f'''(\xi)\right] . \tag{2.10}$$

Pentru $\alpha = 1$, din (2.10) rezultă

$$y'_i = \frac{y_{i+1} - y_{i-1}}{2h} - \frac{h^2}{6}f'''(\xi) , \tag{2.11}$$

în timp ce, pentru $\alpha = 0$, se deduce

$$y'_{i-1} = \frac{-y_{i+1} + 4y_i - 3y_{i-1}}{2h} + \frac{h^2}{3}f'''(\xi) . \tag{2.12}$$

Comparând formulele (2.11) şi (2.12), se observă că, la acelaşi grad al polinomului de interpolare, plasarea punctului de calcul spre centrul diviziunii conduce la reducerea de două ori a erorii. Mai importantă este însă comparaţia cu cazul anterior, $n = 1$. Se observă că formulele (2.11), (2.12) sunt de ordinul doi faţă de formulele (2.8), (2.9) care sunt de ordinul întâi. Câştigul de precizie este obţinut cu preţul considerării variaţiei funcţiei peste un interval mai mare, $2h$, pentru $n = 2$, faţă de h, pentru $n = 1$.

Cu ajutorul expresiei (2.10) se poate calcula şi derivata de ordinul doi

$$y'' = \frac{\Delta^2 y_{i-1}}{h^2} + \frac{h}{6}\frac{d^2}{d\alpha^2}\left[\alpha(\alpha-1)(\alpha-2)f'''(\xi)\right] . \tag{2.13}$$

Pentru $\alpha = 1$, se obţine

$$y_i'' = \frac{\Delta^2 y_{i-1}}{h^2} - \frac{h^2}{3}\left(\frac{d\xi}{d\alpha}\right)_{\alpha=1} f^{(4)}(\xi) = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} - \frac{h^2}{12} f^{(4)}(\xi)\,, \tag{2.14}$$

eroarea fiind de ordinul doi. A doua egalitate (2.14) se obţine cu ajutorul unor formule Taylor pentru y_{i-1}, sau prin utilizarea unui polinom de gradul trei. Se constată o precizie superioară, datorată centrării punctului, faţă de cazul $\alpha = 0$, pentru care din (2.13) rezultă formula

$$y_{i-1}'' = \frac{\Delta^2 y_{i-1}}{h^2} - h f'''(\xi) + \frac{2h^2}{3}\left(\frac{d\xi}{d\alpha}\right)_{\alpha=0} f^{(4)}(\xi) \tag{2.15}$$

la care eroarea este de ordinul întâi.

EXEMPLU. (Efectul reprezentării finite în numere maşină a numerelor reale) Se calculează derivatele y', y'' ale funcţiei $y(x) = e^{x/2}$ în punctul $x_i = 2.5$ folosind formulele de mai sus. Valorile exacte sunt $y_i' = 1.74517$, $y_i'' = 0.87259$. În formulele de derivare apare pasul h. Valoarea acestuia este necunoscută. Din definiţia derivatei

$$y'(x_0) = \lim_{x \to x_0} \frac{y(x) - y(x_0)}{x - x_0} \tag{2.16}$$

ne-am putea aştepta ca valorile cele mai exacte să fie obţinute pentru $h \to 0$. Vom considera valorile $h_k = 10^{-k}$, $k = \overline{1,16}$. Aproximările derivatelor pentru câteva valori ale lui h sunt prezentate în tabelul de mai jos. S-au considerat următoarele cazuri: (1) $y_i' \cong (y_i - y_{i-1})/h$, (2) $y_i' \cong (y_{i+1} - y_i)/h$, (3) $y_i' = (y_{i+1} - y_{i-1})/2h$, (4) $y_i'' \cong (y_{i+2} - 2y_{i+1} + y_i)/h^2$, (5) $y_i'' \cong (y_{i+1} - 2y_i + y_{i-1})/h^2$. S-a utilizat o precizie de lucru de 80 biţi. Se poate observa că micşorarea pasului conduce iniţial la îmbunătăţirea preciziei, dar, după un anumit prag, apar abateri mari. Acestea sunt rezultatul pierderii numărului de cifre semnificative din diferenţa $y_i - y_{i-1}$. De exemplu, dacă în calculator se memorează 7 cifre zecimale, iar y_i, y_{i-1} au primele 6 cifre zecimale identice, diferenţa are o singură cifră zecimală exactă.

$\lg h$ Cazul	1	2	3	4	5
-1	1.70226	1.78954	1.74590	0.91752	0.87277
-4	1.74513	1.74522	1.74517	0.87202	0.87239
-7	1.74517	1.74517	1.74517	0.00000	0.00000
-10	1.70985	1.70985	1.70985	$7.3 \cdot 10^8$	0.00000

Mărirea preciziei de lucru nu elimină fenomenul de pierdere de cifre semnificative. Acesta se va produce la valori mai mici ale pasului h. O prezentare sugestivă a acestei comportări universale a procedeului de aproximare numerică a derivatei este dată în fig. 2.1 unde se reprezintă grafic logaritmul zecimal al erorii

$$\lg e_1 = \lg \left| y_i' - \tilde{y}_i' \right|, \quad \lg e_2 = \lg \left| y_i'' - \tilde{y}_i'' \right| \tag{2.17}$$

funcţie de logaritmul pasului. Reprezentarea logaritmică este convenabilă pentru verificarea practică a ordinului erorii prezis teoretic. Dacă din teorie se prezice că $e = O(h^k)$, atunci dependenţa dintre eroare şi pas se exprimă logaritmic prin $\lg e = C + k \lg h$,

Figura 2.1: Variaţia erorii absolute în estimarea derivatelor funcţie de pasul h.

unde C este o constantă. Putem construi regresia lineară (vezi EXEMPLUL 1 din 1.3) a valorilor erorii calculate în experimente numerice şi să verificăm obţinerea ordinului de eroare. Trebuie să eliminăm din datele de intrare cele ce sunt afectate de pierderea catastrofală de cifre semnificative. Pentru cazurile 1, 3 şi 5 se obţin dreptele $-0.364134 + 0.999353 \lg h$, $-1.00836 + 2.05568 \lg h$, $-1.72727 + 2.00789 \lg h$ respectiv. Aceste drepte sunt reprezentate cu linie groasă în fig. 2.1. Se confirmă şi prin experiment numeric ordinele $O(h)$ pentru cazul 1, $O(h^2)$ pentru cazurile 3, 5, ce au fost prezise teoretic.

2.1.2 Formularea operatorială

Folosind operatorul de translaţie, se pot obţine operatorii de derivare. Astfel, plecând de la expresia (1.19)

$$f(x_0 + \alpha h) = (E^0 + \Delta)^\alpha y_0, \tag{2.18}$$

prin derivare, se obţine

$$\frac{df}{dx} = \frac{1}{h}\frac{d}{d\alpha}\left((E^0 + \Delta)^\alpha y_0\right) = \frac{1}{h}\left[(E^0 + \Delta)^\alpha \ln(E^0 + \Delta)\right]\ y_0 = \tag{2.19}$$

$$\frac{1}{h}\left[\ln(E^0 + \Delta)\right]\ \left(\left[(E^0 + \Delta)^\alpha\right] y_0\right) = \frac{1}{h}\left[\ln(E^0 + \Delta)\right] f(x) \tag{2.20}$$

unde s-a evidenţiat prin $[\,]$ operatorul ce trebuie aplicat. Prin identificare, rezultă operatorul de derivare cu diferenţa la dreapta (Δ)

$$\frac{d}{dx} = \frac{1}{h}\ln(E^0 + \Delta)\ . \tag{2.21}$$

În mod similar, folosind operatorul diferenţa la stânga (∇), se obţine

$$\frac{d}{dx} = \frac{1}{h}\ln(E^0 - \nabla)\,. \tag{2.22}$$

Extinderea operaţiei de derivare a funcţiei exponenţiale la derivarea operatorilor este justificată prin analogia expresiilor utilizate cu seria binomială. Practic, expresiile (2.21) şi (2.22) se utilizează sub forma obţinută prin dezvoltarea în serie a logaritmului

$$\frac{d}{dx} = \frac{1}{h}\left(\Delta - \frac{\Delta^2}{2} + \frac{\Delta^3}{3} - \frac{\Delta^4}{4} + \dots\right) \tag{2.23}$$

$$= \frac{1}{h}\left(\nabla + \frac{\nabla^2}{2} + \frac{\nabla^3}{3} + \frac{\nabla^4}{4} + \dots\right)\,. \tag{2.24}$$

Operatorii (2.23) şi (2.24) pot fi aplicaţi oricărui polinom de interpolare cu diferenţe la dreapta sau la stânga care aproximează funcţia $f(x)$. În acest fel, se obţin, pe cale formală, relaţii similare cu cele precedente. Prin aplicarea repetată a operatorilor (2.23) se obţin operatorii derivatelor de ordin superior. Spre exemplu, pentru derivata de ordinul doi rezultă

$$\frac{d^2}{dx^2} = \frac{1}{h^2}\left(\Delta^2 - \Delta^3 + \frac{11}{2}\Delta^4 + \dots\right) \tag{2.25}$$

$$= \frac{1}{h^2}\left(\nabla^2 + \nabla^3 + \frac{11}{2}\nabla^4 + \dots\right)\,. \tag{2.26}$$

Exemplu. Fie $f : [0,1] \to R$ o funcţie a cărei valori în nodurile $x_j = j/n$, $j = \overline{0,n}$ sunt $f_j = f(x_j)$ şi sunt cunoscute. Se cere o estimare de $O(h^4)$ a derivatei $f_0' = f'(x_0 = 0)$ unde $h = 1/n$.

Rezolvare. Fiind disponibile doar punctele la dreapta, vom aplica (2.23). Numărul de termeni luaţi din dezvoltarea în serie determină ordinul formulei. În cazul de faţă, va trebui să considerăm patru termeni

$$f_0' \cong \frac{1}{h}\left(\Delta - \frac{\Delta^2}{2} + \frac{\Delta^3}{3} - \frac{\Delta^4}{4}\right) f_0\,. \tag{2.27}$$

Aplicarea repetată a operatorului de diferenţă la dreapta conduce la $\Delta f_0 = f_1 - f_0$, $\Delta^2 f_0 = \Delta(\Delta f_0) = \Delta(f_1 - f_0) = \Delta f_1 - \Delta f_0 = f_2 - 2f_1 + f_0$, $\Delta^3 f_0 = \Delta(\Delta^2 f_0) = f_3 - 3f_2 + 3f_1 - f_0$, $\Delta^4 f_0 = \Delta(\Delta^3 f_0) = f_4 - 4f_3 + 6f_2 - 4f_1 + f_0$. Se remarcă apariţia coeficienţilor din dezvoltarea binomială $(1-a)^n$. Rezultă formula

$$f_0' \cong \frac{1}{h}\left(-\frac{1}{4}f_4 + \frac{4}{3}f_3 - 3f_2 + 4f_1 - \frac{25}{12}f_0\right)\,. \tag{2.28}$$

Pentru a verifica ordinul formulei, folosim primii 5 termeni din dezvoltarea în serie Taylor a funcţiei f, $T_5(x) = f_0 + f_0'x + f_0''x/2 + f_0^{(3)}x/6 + f_0^{(4)}x/24 + f_0^{(5)}x/120$ şi se obţine

$$\frac{1}{h}\left(-\frac{1}{4}T_5(4h) + \frac{4}{3}T_5(3h) - 3T_5(2h) + 4T_5(h) - \frac{25}{12}T_5(0)\right) = f_0' - h^4 f_0^{(5)}/5\,, \tag{2.29}$$

deci eroarea este într-adevăr de ordinul IV.

2.1.3 Polinoame de interpolare în funcţie şi derivată

Să presupunem că sunt date valorile unei funcţii y_i şi ale derivatei sale y_i' în nodurile $x_i \in [a,b]$, $i = \overline{1,N}$. Pentru evaluarea derivatei în alte puncte din intervalul $[a,b]$, se poate urma procedeul anterior de derivare a polinomului de interpolare ce satisface $p_{N-1}(x_i) = y_i$. O metodă mai precisă, ce foloseşte şi informaţiile asupra derivatei în noduri, este însă să se construiască polinomul de grad $2N-1$ ce satisface

$$p_{2N-1}(x_i) = y_i, \quad p'_{2N-1}(x_i) = y'_i\,, \quad i = \overline{1,N}. \tag{2.30}$$

O astfel de interpolare ce foloseşte şi valorile unor derivate în noduri se numeşte de tip *Hermite*. Polinomul de interpolare Hermite este unic definit pentru abscise distincte, $i \neq j \Rightarrow x_i \neq x_j$. Ca şi polinomul de interpolare uzual, polinomul de interpolare Hermite are o formă Newton, ce utilizează diferenţe divizate, şi o formă Lagrange.

Forma Newton a polinomului de interpolare Hermite utilizează o generalizare diferenţelor divizate, denumită *diferenţe divizate cu repetiţie* în care se defineşte

$$\mathrm{DD}(x_i, x_i) \equiv f'(x_i) = y'_i\,. \tag{2.31}$$

Regula recursivă (1.9), de obţinere a diferenţelor divizate superioare, rămâne valabilă. Explicit, forma Newton a polinomului Hermite este

$$p_{2N-1}(x) = y_1 + (x-x_1)\,\mathrm{DD}(x_1,x_1) + (x-x_1)^2\,\mathrm{DD}(x_1,x_1,x_2) + \tag{2.32}$$
$$(x-x_1)^2(x-x_2)\,\mathrm{DD}(x_1,x_1,x_2,x_2) + \ldots \tag{2.33}$$

ce se poate compara cu (1.13). Coeficienţii formei Newton se pot calcula convenabil tabelar precum în exemplul de mai jos.

$$\begin{array}{cccccc}
x_1 & y_1 & - & - & \cdots \\
x_1 & y_1 & \mathrm{DD}(x_1,x_1) = y'_1 & - & \cdots \\
x_2 & y_2 & \mathrm{DD}(x_1,x_2) = \dfrac{y_2-y_1}{x_2-x_1} & \mathrm{DD}(x_1,x_1,x_2) = \dfrac{\mathrm{DD}(x_2,x_1)-\mathrm{DD}(x_1,x_1)}{x_2-x_1} & \cdots \\
x_2 & y_2 & \mathrm{DD}(x_2,x_2) = y'_2 & \mathrm{DD}(x_1,x_2,x_2) = \dfrac{\mathrm{DD}(x_2,x_2)-\mathrm{DD}(x_2,x_1)}{x_2-x_1} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

Forma Lagrange a polinomului de interpolare Hermite este

$$p_{2N-1}(x) = \sum_{k=1}^{N} \mathcal{A}_k(x) y_k + \sum_{k=1}^{N} \mathcal{B}_k(x) y'_k\,, \tag{2.34}$$

unde impunem proprietăţile

$$\mathcal{A}_k(x_j) = \delta_{kj}, \quad \mathcal{A}'_k(x_j) = 0, \quad \mathcal{B}_k(x_j) = 0, \quad \mathcal{B}'_k(x) = \delta_{kj} \tag{2.35}$$

prin analogie cu procedeul din 1.1. S-a utilizat simbolul Kronecker: $\delta_{kj} = 0$, dacă $j \neq k$; $\delta_{kj} = 1$, dacă $j = k$. Polinoamele $\mathcal{A}_k$, $\mathcal{B}_k$ se pot exprima funcţie de polinoamele Lagrange (1.29) $\mathcal{L}_k$

$$\mathcal{A}_k(x) = [1 - 2(x - x_k)\mathcal{L}_k'(x_k)]\,\mathcal{L}_k^2(x)\,, \tag{2.36}$$

$$\mathcal{B}_k(x) = (x - x_k)\mathcal{L}_k^2(x)\,. \tag{2.37}$$

Printr-un procedeu asemănător celui folosit pentru determinarea formulei (1.16), se poate obţine expresia restului la interpolarea Hermite

$$R_{2N-1}(x) \equiv f(x) - p_{2N-1}(x) = \prod_{i=1}^{N}(x - x_i)^2\; f^{(2N)}(\xi)/(2N)!\,. \tag{2.38}$$

EXEMPLU. Vom calcula derivata funcţiei $f(x) = \sin \pi x$ în $x = 1/8$ folosind forma Newton a polinomul Hermite. Utilizăm nodurile $x_j = (j-1)/4$, $j = 1, 2$. Tabelul diferenţelor divizate este

x_j	y_j			
0	0	–	–	–
0	0	$\boxed{\pi}$	–	–
$\frac{1}{4}$	$\frac{\sqrt{2}}{2}$	$2\sqrt{2}$	$4(2\sqrt{2} - \pi)$	–
$\frac{1}{4}$	$\frac{\sqrt{2}}{2}$	$\boxed{\frac{\pi\sqrt{2}}{2}}$	$2\sqrt{2}(\pi - 4)$	$4\pi(2\sqrt{2} + 4) - 64\sqrt{2}$

unde valorile obţinute din derivata exactă $f'(x) = \pi \cos \pi x$ au fost încadrate. Forma Newton a polinomului Hermite este

$$p_2(x) = x\pi + x^2 4(2\sqrt{2} - \pi) + x^2(x - 1/4)\left[4\pi(2\sqrt{2} + 4) - 64\sqrt{2}\right]\,, \tag{2.39}$$

ceea ce conduce la următoarea formă canonică a polinomului

$$p_2(x) = \pi + (48\sqrt{2} - 16\pi - 4\sqrt{2}\pi)\,x + (-192\sqrt{2} + 48\pi + 24\sqrt{2}\pi)\,x^2\,. \tag{2.40}$$

Derivata cerută este $p_2'(1/8) = 2.90188$ foarte apropriată de cea exactă $f'(1/8) = 2.90245$ şi cu mult mai precisă decât estimarea centrată de ordinul II $(f(1/4) - f(0))/(1/4) = 2.82843$. Graficul logaritmului erorii relative $\lg \varepsilon = \lg |(p_2'(x) - f'(x))\,/f'(x)|$ de mai jos arată prezicerea corectă a 3 cifre semnificative în evaluarea derivatei pe intervalul $[0, 1/4]$.

2.1.4 Derivate folosind funcţii spline

Funcţia spline polinomială de ordinul 3 poate fi utilizată pentru calculul aproximativ al derivatelor de ordinul 1 şi 2. În acest scop, se determină coeficienţii m_i, a_i, b_i, restricţia funcţiei spline pe intervalul (x_i, x_{i+1}) fiind

$$p_{3,i}(x) = y_i + m_i(x - x_i) + b_i(x - x_i)^2 + a_i(x - x_i)^3, \tag{2.41}$$

determinată în paragraful 1.1.2. Derivatele de ordinul 1 şi 2 se aproximează prin derivatele de acelaşi ordin ale polinomului $p_{3,i}$

$$y' = m_i + 2b_i(x - x_i) + 3a_i(x - x_i)^2, \quad y'' = 2b_i + 6a_i(x - x_i)\,. \tag{2.42}$$

Eroarea la aproximarea derivatei de ordinul k poate fi evaluată cu ajutorul relaţiei (vezi Observaţia 4 din 1.1.2)

$$\max_{x\in[a,b]} \left| f^{(k)}(x) - s_m^{(k)}(x) \right| \le \frac{(b-a)^{m-k}}{(m-k)!} \max_{x\in[a,b]} \left| f^{(m)}(x) - s_m^{(m)}(x) \right|, \tag{2.43}$$

unde m este ordinul funcţiei spline ($m = 3$ pentru funcţia spline cubică). Deoarece funcţia spline este derivabilă de $m - 1$ ori pe (a, b), am notat prin $s_m^{(m)}$ o funcţie treaptă, obţinută prin derivarea restricţiilor funcţiei s pe subintervale. Exemplul din paragraful 1.1.2 prezintă şi comportarea numerică a derivatelor.

2.1.5 Derivate folosind diverse aproximaţii

Formule de derivare se pot obţine folosind oricare dintre procedeele de aproximare prezentate în capitolul 1. Considerăm pe scurt câteva alte procedee inidicând domeniile tipice de aplicare.

Interpolare trigonometrică.

Prin derivarea expresiei (1.57), se obţine

$$\frac{df}{dx} = y' \cong \frac{dg}{dx} =$$
$$2\pi \sum_{k=1}^{N-1} [-ka_k \sin(2\pi kx) + kb_k \cos(2\pi kx)] - \pi N\, a_N \sin 2\pi Nx\,. \tag{2.44}$$

Rezultatul este o nouă funcţie ce are coeficienţii Fourier $-2\pi ka_k$, $-2\pi kb_k$. Consideraţiile din 1.1.3 asupra convergenţei interpolării trigonometrice se aplică acum noii funcţii. În particular, vom urmări dacă $c_k = 2\pi k(a_k^2 + b_k^2)^{1/2}$ devin suficient de mici pe măsură ce N creşte. O proprietate importantă a aproximărilor trigonometrice este că, pentru funcţii netede, coeficienţii c_k scad mai repede decât orice putere a lui $h = 1/N$ începând de la un anumit rang k. Prin analogie cu analiza de ordin de eroare de la formulările anterioare, spunem că aproximarea trigonometrică a derivatei este de ordin infinit. Această

proprietate stă la baza *metodelor spectrale* de rezolvare a ecuaţiilor diferenţiale. Aplicabilitatea generală a procedeului este însă limitată de cerinţa de netezime. De exemplu, dacă funcţia f prezintă discontinuităţi izolate, vor apărea erori importante de mascare (vezi 1.1.3).

Aproximarea mini-max.

Se derivează polinomul aproape mini-max (vezi 1.2.3). Gradul acestui polinom se ia, în general, mai mic decât gradul polinomului de interpolare corespunzător unei diviziuni alese. Astfel de aproximări se utilizează când se doreşte minimizarea numărului de operaţii aritmetice necesare estimării derivatei.

Aproximarea prin metoda celor mai mici pătrate.

Se derivează aproximanta (1.83). În general, derivatele astfel obţinute sunt mai netede decât în cazul unei interpolari care foloseşte toate punctele din reţea. Se foloseşte tipic pentru date cunoscute incert (măsurători experimentale).

2.2 Integrarea numerică

Integrarea numerică a funcţiilor de o variabilă reprezintă o aplicaţie imediată a aproximării. În general, prin integrare numerică erorile de calcul se reduc, variaţiile se netezesc. Cazurile mai dificile sunt legate de integrarea funcţiilor care prezintă variaţii rapide de semn pe intervalul de integrare, astfel încât termenii însumaţi se compensează, iar rezultatul, un număr mic în modul, apare ca diferenţa a unor numere mari în modul[1]. Asemenea funcţii sunt indicate pentru testarea diferitelor formule de integrare propuse. Schema generală pentru a obţine o formula de integrare numerică este următoarea:

1. Se introduce o diviziune a intervalului de calcul $[a, b]$ prin punctele $\{x_i, i \in \overline{1, N}\}$.

2. Se scrie funcţia de integrat $f(x)$, punându-se în evidenţă aproximanta $g(x)$ şi eroarea (restul) $R(x)$

$$f(x) = g(x) + R(x) \ . \tag{2.45}$$

3. Se integrează relaţia (2.45) termen cu termen, obţinându-se o valoare aproximativă şi o eroare pentru integrare

$$\int_a^b f(x)dx = \int_a^b g(x)dx + \int_a^b R(x)dx \ . \tag{2.46}$$

[1] Vezi capitolul 4.

Integrala aproximantei $g(x)$ se evaluează numeric pe fiecare subinterval al diviziunii

$$I = \int_a^b g(x)dx = \sum_{k=1}^{N} a_k I_k,\ I_k \equiv \int_a^b g_k(x)dx\ . \tag{2.47}$$

Uzual aproximanta $g(x)$ se alege de forma (1.6), astfel încât integralele I_k să poată fi evaluate exact. Eroarea de integrare este în acest caz integrala restului

$$\delta = \int_a^b R(x)dx\ . \tag{2.48}$$

4. Se caută o posibilitate de minimizare a erorii.

În efectuarea diverselor calcule menţionate mai sus, sunt utile câteva rezultate din analiză pe care le reamintim fără demonstraţie.

Teorema de valoare medie a unei integrale. Pentru f, g continue pe $[a, b]$ şi $g(x) \geq 0$ pe $[a, b]$, există un $\xi \in [a, b]$ astfel încât

$$\int_a^b f(x)g(x)dx = f(\xi)\int_a^b g(x)dx.$$

Teorema valorii intermediare. Fie f continuă pe $[a, b]$ şi m valoarea sa minimă şi M valoarea sa maximă pe acest interval. Pentru orice $r \in [m, M]$ există un $\xi \in [a, b]$ astfel încât $f(\xi) = r$.

Teorema fundamentală a analizei. Dacă f este continuă pe $[a, b]$, atunci funcţia $\varphi(x) = \int_a^x f(t)dt$ este derivabilă în orice punct $c \in [a, b]$ şi derivata sa este $\varphi'(c) = f(c)$.

2.2.1 Formule Newton-Cotes închise

Formulele de integrare care utilizează valorile funcţiei la capetele intervalului de integrare, $y_1 = f(a)$, $y_N = f(b)$ sunt denumite formule *închise*. Foarte uzuale sunt metodele care utilizează interpolarea polinomială pe o diviziune echidistantă a intervalului de integrare $\{a = x_1, x_2, \ldots, x_N = b\}$ cu $x_{i+1} - x_i \equiv h = (b - a)/(N - 1)$, formulele obţinute fiind denumite de tip *Newton-Cotes*. În obţinerea formulelor de integrare, este convenabilă forma Newton cu diferenţe finite a polinomului de interpolare (1.25). Vom remarca însă că se preferă utilizarea unor polinoame de grad mic (unu, doi sau trei), pe subintervale, ceea ce revine de fapt, la integrarea unor funcţii spline, racordate pe noduri numai prin valorile funcţiei $f(x)$, dar nu şi prin derivate. În cele ce urmează vom folosi Π_n pentru a nota familia polinoamelor de grad cel mult n.

Formula trapezelor.

Funcţia aproximantă este o linie poligonală. Pe fiecare subinterval (x_i, x_{i+1}), funcţia $f(x)$ se înlocuieşte cu aproximarea sa printr-un polinom de gradul întâi plus un rest. Conform (1.25), polinomul se scrie

$$p_1(x) = y_i + \alpha \Delta y_i \,, \tag{2.49}$$

cu $\alpha \in [0, 1]$, unde restul (1.26) este

$$R_1(x) = h^2 C_\alpha^2 f''(\xi_i),\ \xi_i \in (x_i, x_{i+1}) \,. \tag{2.50}$$

Pe intervalul (x_i, x_{i+1}), variabila x se poate exprima

$$x = x_i + \alpha h,$$

de unde $dx = h d\alpha$. Integrala exactă a aproximantei (2.49) este

$$I_i = \int_0^1 (y_i + \alpha \Delta y_i) d\alpha = (y_i + y_{i+1})/2 \ , \tag{2.51}$$

iar eroarea pe interval rezultă din integrarea restului (2.50)

$$\delta_i = (h^3/2) \int_0^1 \alpha(\alpha - 1) f''[\xi_i(\alpha)] \, d\alpha,\ \xi_i \in (x_i, x_{i+1}) \ .$$

Expresia erorii poate fi transformată aplicând teorema de medie pentru integrale. Avem $-\alpha(\alpha - 1) \geq 0$ şi vom presupune că $-f''[\xi_i(\alpha)]$ este continuă pe $[x_i, x_{i+1}]$, astfel încât condiţiile teoremei să fie satisfăcute. Se obţine

$$\delta_i = [h^3 f''(\xi_i')/2] \int_0^1 \alpha(\alpha - 1) d\alpha = -h^3 f''(\xi_i')/12 \,, \tag{2.52}$$

cu $\xi_i' \in [x_i, x_{i+1}]$, în general diferit de ξ_i din (2.50). Formula (2.51) se aplică pe fiecare subinterval şi vom obţine, prin sumare, valoarea aproximativă a integralei

$$I = \sum_{i=1}^{N-1} I_i = \frac{h}{2}(y_1 + 2y_2 + \ldots + 2y_{N-1} + y_N) \ . \tag{2.53}$$

Eroarea la integrare este suma erorilor pe subintervale, $\delta = \sum_{i=1}^{N-1} \delta_i = -(h^3/12) \cdot \sum_{i=1}^{N-1} f''(\xi_i')$. Însă $(N-1)^{-1} \sum_{i=1}^{N-1} f''(\xi_i')$ este o valoare medie a valorilor funcţiei f'' în punctele ξ_i' şi am presupus f'' continuă. Conform teoremei valorii intermediare, există un $\xi \in [x_1, x_N]$ astfel încât

$$\frac{1}{N-1} \sum_{i=1}^{N-1} f''(\xi_i') = f''(\xi) \tag{2.54}$$

şi eroarea se poate scrie

$$\delta = -\frac{h^2}{12}(b-a)f''(\xi) = -\frac{(b-a)^3}{12(N-1)^2}f''(\xi)\,. \tag{2.55}$$

Expresia (2.55) sugerează că, pentru funcţii cu derivata de ordinul doi continuă, eroarea la integrare scade aproximativ cu pătratul numărului de intervale $N-1$. Spunem astfel că eroarea la formula trapezelor este de ordinul doi, $\delta = O(h^2)$. Se observă că, faţă de eroarea pe un subinterval $\delta_i = O(h^3)$, eroarea pe tot intervalul δ este cu un ordin mai mic datorită acumulării erorii din însumare. Se remarcă, de asemenea, că formula trapezelor este *exactă* pentru toate polinoamele de gradul 1, adică pentru $\forall f \in \Pi_1$.

Formulele Simpson.

Urmărim obţinerea unor formule mai precise pe subintervalele de calcul. Din procedeul general, observăm că acest deziderat se poate realiza prin folosirea unor polinoame de interpolare de grad mai mare pe subintervale. Deoarece, pentru polinoame de grad mai mare de unu, avem nevoie de mai mult de două noduri, vom grupa convenabil subintervalele (x_i, x_{i+1}). Din punctul de vedere al aproximării funcţiei $f(x)$ pe *tot* intervalul de calcul $[a, b]$ procedeul revine la folosirea unei interpolări spline cu deficienţă, ale cărei restricţii pe subintervale sunt polinoame.

Formula Simpson "1/3". Începem cu polinoame de gradul doi pe subinterval. Avem nevoie de trei noduri şi ca atare vom obţine o formulă de integrare pe (x_{i-1}, x_{i+1}) [2]. Polinomul aproximant este

$$p_2(x) = y_{i-1} + \alpha\Delta y_{i-1} + \alpha(\alpha-1)\Delta^2 y_{i-1}/2$$

conform (1.25) iar restul are forma

$$R_2(x) = h^3 C_\alpha^3 f^{(3)}(\xi_i),\ \xi_i \in (x_{i-1}, x_{i+1})\ .$$

Formula obţinută prin integrarea lui $p_2(x)$ este

$$I_i = h\int_0^2 p_2(x_{i-1}+\alpha h)d\alpha = \frac{h}{3}(y_{i-1}+4y_i+y_{i+1})\ . \tag{2.56}$$

Aplicarea teoremei de medie la integrarea restului trebuie efectuată pe subintervalele pe care C_α^3 nu schimbă semnul

$$h^3\int_0^2 C_\alpha^3 f^{(3)}(\xi_i)d\alpha = \frac{h^3}{3!}\left[f^{(3)}(\xi_i^{(1)})\left(\frac{1}{4}\right) + f^{(3)}(\xi_i^{(2)})\left(-\frac{1}{4}\right)\right]\ .$$

Rezultatul obţinut este însă o supraestimare. Faptul că s-au obţinut coeficienţi numerici de semne contrare sugerează că avem un efect de compensare a erorilor.

[2] Aici şi în cele ce urmează vom alege intervalele astfel încât formulele să fie centrate pe punctul x_i.

Ne convingem de aceasta prin intermediul unui alt procedeu de calcul al erorii ce are aplicabilitate mai largă. Considerăm valoarea aproximativă a integralei

$$I_i = \frac{h}{3}[f(x_{i-1}) + 4f(x_i) + f(x_{i+1})]$$

şi vom presupune f derivabilă de minim 5 ori. Dezvoltăm în serie Taylor în jurul punctului x_{i-1} expresia de mai sus

$$I_i = 2hf_{i-1} + 2h^2 f'_{i-1} + 4h^3 f''_{i-1}/3 + 2h^4 f^{(3)}_{i-1}/3 + 5h^5 f^{(4)}_{i-1}/18 + h^6 f^{(5)}_{i-1}/10 + \dots$$

unde am notat $f_i^{(k)} = f^{(k)}(x_i)$. Pe de altă parte, introducem

$$F(x) = \int_{x_{i-1}}^{x} f(x)dx\,,$$

valoarea exactă a integralei pe intervalul (x_{i-1}, x). Avem, în particular, $F(x_{i-1}) = 0$, şi $F(x_{i-1} + 2h)$ este valoarea exactă pe intervalul (x_{i-1}, x_{i+1}). Conform teoremei fundamentale a analizei, $F'(x) = f(x)$. Ca atare, dezvoltarea în serie Taylor în jurul punctului x_{i-1} a valorii exacte $F(x_{i-1} + 2h)$ este

$$2hf_{i-1} + 2h^2 f'_{i-1} + 4h^3 f''_{i-1}/3 + 2h^4 f^{(3)}_{i-1}/3 + 4h^5 f^{(4)}_{i-1}/15 + 4h^6 f^{(5)}_{i-1}/45 + \dots$$

Diferenţa $\delta = F(x_{i-1} + 2h) - I_i$ este eroarea căutată şi are expresia

$$\delta = -h^5 f^{(4)}_{i-1}/90 - h^6 f^{(5)}_{i-1}/90 + \dots\,.$$

Astfel, chiar dacă am pornit cu un polinom aproximant de gradul doi, formula obţinută este exactă şi pentru polinoame de gradul trei. Acest rezultat face ca formula lui Simpson (2.56) să fie *economică* în numărul de evaluări ale funcţiei f necesare realizării unei anumite precizii, de unde larga sa folosire în aplicaţii.

Considerând acum problema integrării pe tot intervalul $[a, b]$ vom introduce un număr impar de $2N+1$ noduri, pentru a avea un număr întreg de perechi de intervale, pe care sumăm relaţia (2.56) şi obţinem formula de integrare Simpson

$$I = \frac{h}{3}(y_1 + 4y_2 + 2y_3 + \dots + 2y_{2N-1} + 4y_{2N} + y_{2N+1})\,. \tag{2.57}$$

Estimarea erorii la integrare este

$$\delta = -(b-a)^5 f^{(4)}(\xi)\,/\,2880N^4, \quad \xi \in (a, b)\,. \tag{2.58}$$

Din (2.58), se observă că eroarea δ este invers proporţională cu puterea a patra a numărului de perechi de subintervale N, adică $\delta = O(h^4)$. Spunem că formula Simpson "1/3" este de ordinul IV.

Fomula Simpson "3/8". Trecem acum la următorul polinom de interpolare, cel de gradul 3. Considerând 4 puncte de diviziune pe intervalul (x_{i-1}, x_{i+2}), polinomul de interpolare este

$$p_3(x) = y_{i-1} + \alpha\Delta y_{i-1} + \alpha(\alpha-1)\Delta^2 y_{i-1}/2 + \alpha(\alpha-1)(\alpha-2)\Delta^3 y_{i-1}/3!\,,$$

iar, prin integrare, se obţine formula lui Simpson "3/8"

$$I_i = h\int_0^3 p_3(x_{i-1}+\alpha h)d\alpha = \frac{3h}{8}(y_{i-1}+3y_i+3y_{i+1}+y_{i+2})\ , \tag{2.59}$$

cu eroarea

$$\delta_i = h^5\int_0^3 C_\alpha^4 f^{(4)}(\xi_i)\,d\alpha = -3h^5 f^{(4)}(\xi_i)/80, \quad \xi_i\in(x_{i-1},x_{i+2})\ . \tag{2.60}$$

Considerând $3N+1$ puncte de diviziune, cu pasul $h=(b-a)/3N$, formula de integrare pe (a,b) se va scrie

$$I = \frac{3h}{8}(y_1+3y_2+3y_2+2y_4+\ldots+2y_{3N-2}+3y_{3N-1}+3y_{3N}+y_{3N+1})\,, \tag{2.61}$$

iar eroarea de integrare pe (a,b) este

$$\delta = -(b-a)^5 f^{(4)}(\xi)\,/\,6480N^4 = O(h^4), \quad \xi\in(a,b)\,. \tag{2.62}$$

La acelaşi interval (a,b) şi acelaşi N, eroarea este mai mică decât în cazul formulei Simpson "1/3" datorită factorului numeric de la numitor. Cum însă o precizie sporită la integrare este obţinută îndeosebi prin mărirea numărului de subintervale, observăm că mărirea gradului polinomului de interpolare $p_3(x)$ nu conduce la un ordin al erorii mai mare, formula "3/8" fiind tot de ordinul IV ca şi formula "1/3". Datorită numărului mai mic de evaluări de funcţie cerute este preferabilă folosirea formulei Simpson "1/3" celei "3/8".

Informativ, dăm şi formula obţinută prin interpolare cu un polinom de gradul 4, pe intervalul (x_{i-2},x_{i+2})

$$I_i = \frac{2h}{45}(7y_{i-2}+32y_{i-1}+12y_i+32y_{i+1}+7y_{i+2})\,, \tag{2.63}$$

cu eroarea

$$\delta_i = -8h^7 f^{(7)}(\xi_i)/945, \quad \xi_i\in(x_{i-1},x_{i+2})\,. \tag{2.64}$$

În acest caz, creşterea gradului polinomului de interpolare a condus la o scădere importantă a erorii. Pe tot interalul $[a,b]$ avem $\delta=O(h^6)$ faţă de $\delta=O(h^4)$ din (2.62).

Formulele de mai sus se pot obţine şi pe două alte căi ce permit extinderi importante. Să considerăm că integrala pe un interval $[a,b]$ se poate aproxima ca o medie ponderată a valorilor funcţiei în punctele $x_i, i=1,\ldots,N$

$$I=\int_a^b f(x)dx \cong \sum_{i=1}^N A_i y_i\,, \tag{2.65}$$

unde A_i sunt ponderi necunoscute. Putem impune ca integrala să fie evaluată exact pentru funcţii polinomiale până la gradul $N-1$. De exemplu, pentru $N=3$, obţinem sistemul de ecuaţii

$$\left\{\begin{array}{lll} \int_a^b dx & = b-a & = A_1+A_2+A_3 \\ \int_a^b xdx & = (b^2-a^2)/2 & = A_1x_1+A_2x_2+A_3x_3 \\ \int_a^b x^2dx & = (b^2-a^2)/3 & = A_1x_1^2+A_2x_2^2+A_3x_3^2 \end{array}\right. \tag{2.66}$$

unde am înlocuit rând pe rând funcţia $f(x)$ cu un polinom de gradul 0,1,2. Se obţin coeficienţii $A_1=A_3=h/3$, $A_2=4h/3$, adică exact cei din formula Simpson "1/3" (2.56). Procedeul de mai sus este cunoscut sub denumirea de *identificare a coeficienţilor de integrare*.

Un al doilea procedeu este dat de înlocuirea funcţiei f cu forma Lagrange a polinomului de interpolare (1.31)

$$\int_a^b f(x)dx=\int_a^b\sum_{i=1}^N y_i\mathcal{L}_i(x)dx+\int_a^b\frac{f^{(N)}(\xi)}{N!}\prod_{i=1}^N(x-x_i)\,dx\,, \tag{2.67}$$

unde s-a utilizat forma (1.16) a restului. Comparând cu (2.65) se observă că

$$A_i=\int_a^b\mathcal{L}_i(x)dx\,. \tag{2.68}$$

În final, să adăugăm, că în numeroase situaţii practice, este convenabil să se evidenţieze o funcţie pondere $w(x)$ în operaţiile de integrare. Se generalizează astfel operaţia de medie ponderată din aritmetică. Utilizând (2.67) avem

$$\int_a^b w(x)f(x)dx=\int_a^b w(x)\left[\sum_{i=1}^N\mathcal{L}_i(x)y_i+\frac{f^{(N)}(\xi)}{N!}\prod_{i=1}^N(x-x_i)\right]dx=\sum_{i=1}^N A_iy_i+\delta,$$

$$A_i=\int_a^b w(x)\mathcal{L}_i(x)dx,\quad \delta=\int_a^b w(x)\frac{f^{(N)}(\xi)}{N!}\prod_{i=1}^N(x-x_i)\,dx,\quad \xi\in(a,b).$$

2.2.2 Formule de integrare deschise

O întrebare imediată asociată formulei generale de integrare (2.65) este dacă alegerea altor puncte x_i şi ponderi A_i ar putea aduce avantaje de calcul. Vom considera în cele ce urmează câteva asemenea cazuri. În general, în obţinerea acestor formule nu se utilizează valorile funcţiei f din capetele intervalului. Astfel de formule se numesc *formule deschise*.

Formule de integrare deschise de tip Newton Cotes

Aceste formule se obţin utilizând un polinom de interpolare care nu trece prin capetele intervalului, deci cu un polinom cu două grade mai mic decât în cazul

formulelor închise. Revenim la obţinerea formulelor de integrare prin integrarea polinomului de aproximare. De data aceasta folosim o formă în care nu apar valorile în capete

$$p_{N-3}(x_1+\alpha h) = y_2 + (\alpha-1)\Delta y_2 + C_{\alpha-1}^2\Delta^2 y_2 + \ldots + C_{\alpha-1}^{N-3}\Delta^{N-3}y_2, \quad (2.69)$$

$\alpha \in [0, N-1]$, eroarea fiind

$$R_{N-1}(x_1+\alpha h) = h^{N-2}C_{\alpha-1}^{N-2} f^{(N-2)}(\xi)\,. \quad (2.70)$$

Evident, formulele de mai sus se pot utiliza pentru $N > 3$. Putem obţine diverse formule prin alegerea intervalului de integrare şi a gradului polinomului de aproximare.

Pentru $N = 4$, calculăm integrala polinomului aproximant de la $\alpha = 0$ la $\alpha = 2$ şi se obţine

$$I_i = 2hy_i, \quad \delta_i = h^3 f''(\xi_i)/3\,. \quad (2.71)$$

În acest caz, intervalul de integrare a fost (x_{i-1}, x_{i+1}), dar termenul y_{i+1} s-a redus, astfel încât formula (2.71) nu conţine nici una din valorile de la capetele intervalului. Observăm că se obţine acelaşi ordin al erorii $\delta \sim O(h^3)$ ca la formula trapezelor (2.53), folosind însă o singură evaluare a funcţiei faţă de două în (2.53). Factorul numeric din expresia erorii de mai sus este însă mai mare decât cel din (2.55). Ca atare, avantajul de calcul obţinut poate fi nesemnificativ în practică.

Pentru $N = 6$ şi $x \in (x_{i-1}, x_{i+3})$ se obţine o formulă cu acelaşi ordin de eroare ca formula Simpson

$$I_i = \frac{4h}{3}(2y_i - y_{i+1} + 2y_{i+2}), \quad \delta_i = \frac{14h^5}{45} f^{(4)}(\xi_i)\,. \quad (2.72)$$

Coeficientul numeric din formula erorii este iarăşi mai mare decât cel din (2.58) şi numărul de evaluări de funcţie este acelaşi. Ca atare, nu se obţine vreun avantaj de calcul.

Formule deschise de tip Gauss.

În formula generală de integrare (2.65), observăm însă că avem la dispoziţie $2N$ parametrii, nu numai ponderile A_i ci şi abscisele în care evaluăm funcţia x_i. Până acum am utilizat abscise echidistante şi am obţinut formule exacte pentru polinoame de grad până la $N-1$, prin alegerea a N ponderi. Putem spera ca, prin utilizarea tuturor celor $2N$ parametri, să obţinem formule exacte pentru polinoame până la gradul $2N-1$. Calea de a obţine astfel de formule este dată de un rezultat teoretic important formulat de Gauss.

Punctul de plecare este formula de aproximare a valorii unei integrale

$$\int_a^b w(x)f(x)dx \cong \sum_{i=1}^{N} A_i f(x_i)\,, \quad (2.73)$$

unde $w(x)$ este o funcţie pondere pozitivă, iar coeficienţii A_i sunt aleşi astfel încât formula este exactă pentru toate polinomale de grad până în $N-1$.

Teoremă. Dacă abscisele x_i, din (2.73) se aleg ca fiind zerourile unui polinom $q(x)$ de gradul N ce satisface relaţia de ortogonalitate

$$\int_a^b w(x)p(x)q(x)dx = 0 \tag{2.74}$$

pentru orice polinom $p(x)$ de grad până la $N-1$, atunci formula (2.73) este exactă pentru toate polinoamele de grad până la $2N-1$.

Demonstraţie. Fie f un polinom de grad cel mult $2N-1$. Acesta se poate scrie $f = pq + r$ unde p, r sunt polinoame de grad cel mult $N-1$. Deoarece $x_i, i = 1, \ldots, N$ sunt zerourile polinomului q, avem $f(x_i) = r(x_i)$ şi

$$\int_a^b w(x)f(x)dx = \int_a^b w(x)\left[p(x)q(x) + r(x)\right]dx = \int_a^b w(x)r(x)dx = \sum_{i=1}^{N} A_i r(x_i) = \sum_{i=1}^{N} A_i f(x_i)\,, \tag{2.75}$$

unde ultima egalitate rezultă din alegerea coeficienţilor A_i enunţată. **2**

Se poate demonstra că zerourile unui polinom ce satisface (2.74) sunt simple şi sunt în interiorul intervalului $[a, b]$, adică $x_i \in (a, b)$. Poziţiile zerourilor depind, în general, de intervalul $[a, b]$. Pentru a obţine valori standard ale abscişelor, se folosesc domenii de integrare canonice, specifice fiecărui tip de ponderi $w(x)$. Apariţia funcţiei de pondere în formula (2.73) facilitează tratarea cazurilor cu singularităţi integrabile după cum vom vedea mai jos. Odată stabilită familia de polinoame ortogonale, prin alegerea ponderii $w(x)$, se află rădăcinile acestora, iar coeficienţii A_i rezultă prin integrarea polinoamelor Lagrange asociaţi rădăcinilor conform relaţiei (2.68).

Am văzut însă în 1.1.1, că interpolarea polinomială globală nu converge întotdeauna către funcţia f. O întrebare naturală este dacă o asemenea comportare nu ar apărea şi în formulele de integrare Gauss. Condiţia de ortogonalitate (2.74) joacă un rol esenţial aici şi se poate demonstra (Stieltjes) că, pentru funcţii f continue, aproximarea Gauss converge către valoarea exactă a integralei pe măsură ce $N \to \infty$.

Fie acum f o funcţie de clasă $C^{2N}[a, b]$ oarecare, nu neapărat un polinom. Dorim să determinăm dacă se obţine o eroare mai mică la integrare prin folosirea unei formule Gauss cu N noduri, prin comparaţie cu formulările anterioare. Teorema de mai sus sugerează compararea cu un polinom de grad $2N-1$. Un polinom de grad $2N-1$ poate fi definit de valorile funcţiei şi derivatei în cele N noduri, după cum am văzut la interpolarea Hermite 2.1.3,

$$p_{2N-1}(x_i) = f(x_i), \quad p'_{2N-1}(x_i) = f'(x_i), \quad i = \overline{1, N}\,. \tag{2.76}$$

Eroarea la interpolare este dată de (2.38), de unde rezultă

$$\int_a^b w(x)f(x)dx - \int_a^b w(x)p_{2N-1}(x)dx = \int_a^b w(x)\prod_{i=1}^{N}(x-x_i)^2\frac{f^{(2N)}(\zeta(x))}{(2N)!}\,dx\,. \tag{2.77}$$

Formula de integrare Gauss, fiind exactă pentru polinoame de grad maxim $2N-1$, rezultă

$$\int_a^b w(x)p_{2N-1}(x)dx = \sum_{i=1}^{N}A_i p_{2N-1}(x_i) = \sum_{i=1}^{N}A_i f(x_i)\,, \tag{2.78}$$

de unde eroarea la integrarea funcţiei f este

$$\delta = \int_a^b w(x)f(x)dx - \sum_{i=1}^{N}A_i f(x_i) = \frac{1}{(2N)!}\int_a^b w(x)\prod_{i=1}^{N}(x-x_i)^2\ f^{(2N)}(\zeta(x))\,dx\,. \tag{2.79}$$

Nodurile x_i se aleg ca fiind rădăcinile polinomului $q(x)$ de gradul N şi ortogonal pe orice polinom $p(x)$ de grad până la $N-1$. Conform teoremei de medie, avem

$$\delta = \frac{f^{(2N)}(\xi)}{(2N)!}\int_a^b w(x)q^2(x)\ \ dx\,,\quad \xi\in(a,b). \tag{2.80}$$

Recunoaştem apariţia produsului scalar

$$(q,q) = \int_a^b w(x)q^2(x)\ \ dx \tag{2.81}$$

(vezi 1.4.2). Ca atare, eroarea la integrarea Gauss este

$$\delta = f^{(2N)}(\xi)\,(q,q)/(2N)!\,. \tag{2.82}$$

Dacă $f\in C^{2N}[a,b]$, atunci $f^{(2N)}(\xi)$ este mărginită. Produsul (q,q) se poate evalua folosind relaţiile de recurenţă (1.94) şi este de asemenea mărginit. În plus, factorul $(q,q)/(2N)!$ scade rapid cu creşterea lui N. Să considerăm acum cele mai des întâlnite cazuri.

Formule Gauss-Legendre.

Mai întâi luăm ponderea $w(x)=1$, adică vom considera integrala obişnuită. Intervalul canonic uzual este $[-1,1]$. Orice alt interval de integrare $[a,b]$ poate fi uşor transformat în $[-1,1]$ prin substituţia $z=2(x-a)/(b-a)-1$

$$\int_a^b f(x)dx = \frac{b-a}{2}\int_{-1}^{+1}F(z)dz,\quad F(z)\equiv f\left(\frac{b-a}{2}\,z+\frac{b+a}{2}\right). \tag{2.83}$$

În cazul de faţă condiţia de ortogonalitate este

$$\int_{-1}^{+1} p(x)q(x)dx = 0\,, \tag{2.84}$$

identică cu (1.96), aşa încât polinoamele $q(x)$ sunt polinoamele Legendre. Poziţiile zerourilor z_i şi ponderile aferente A_i sunt redate, pentru câteva valori ale lui N, în tabelul (2.1). Integrala pe $[-1,1]$ se aproximează prin

$$\int_{-1}^{+1} F(z)dz \cong \sum_{i=1}^{N} A_i F(z_i)\,, \tag{2.85}$$

iar cea pe $[a,b]$ prin

$$\int_a^b f(x)dx \cong \frac{b-a}{2}\sum_{i=1}^{N} A_i f(x_i)\,, \tag{2.86}$$

cu $x_i = (b-a)z_i/2+(a+b)/2$. Din relaţiile de recurenţă (1.94) şi (2.82) rezultă eroarea la integrare

$$\delta = 2^{2N+1}(N!)^4 f^{(2N)}(\xi)/(2N+1)[(2N)!]^3 \tag{2.87}$$

pe intervalul $[-1,1]$ sau

$$\delta = (b-a)^{2N+1}(N!)^4 f^{(2N)}(\xi)/(2N+1)[(2N)!]^3 \tag{2.88}$$

pe intervalul $[a,b]$ folosind substituţia de mai sus. Abscisele x_i în care se evaluează funcţia F nu sunt de regulă echidistante. Putem însă pune în evidenţă un pas mediu $h = (b-a)/N$, pentru a determina ordinul erorii în vederea comparării cu metodele anterioare. Se obţine

$$\delta = \left(\frac{b-a}{N}\right)^{2N+1} \frac{N^{2N+1} f^{(2N)}(\xi)}{(N!)^4(2N+1)[(2N)!]^3} < \frac{h^{2N+1} f^{(2N)}(\xi)}{(N!)^4(2N)!} \tag{2.89}$$

şi se observă obţinerea unui ordin al erorii de $O(h^{2N+1})$, folosind doar N evaluări ale funcţiei f. Mai mult, coeficientul numeric din (2.89) scade exponenţial cu creşterea lui N (vezi tabelul 2.1).

Formule Gauss-Cebâşev.

Considerăm acum ponderea $w(z) = (1-z^2)^{-1/2}$ pe intervalul $[-1,1]$. Se poate observa că funcţia pondere prezintă singularităţi în ± 1. Utilitatea practică a unei astfel de ponderi este integrarea numerică a funcţiilor ce prezintă acelaşi gen de singularitate integrabilă în capetele domeniului de integrare. Fie $f(z)$ o

Tabelul 2.1: Rădăcinile z_i, ponderile A_i şi eroarea la integrarea Gauss-Legendre

N	z_i	A_i	δ
2	$\mp 1/\sqrt{3}$	1	$\dfrac{h^5 f^{(4)}(\xi)}{2^8 \cdot 3^3 \cdot 5}$
3	$\mp\sqrt{3/5}$ 0	5/9 8/9	$\dfrac{h^7 f^{(6)}(\xi)}{2^{16} \cdot 3^3 \cdot 5^3 \cdot 7}$
4	$\mp$0.86113 63115 94053 $\mp$0.33998 10435 84856	0.34785 48451 37454 0.65214 51548 62546	$\dfrac{h^9 f^{(8)}(\xi)}{2^{15} \cdot 3^{12} \cdot 5^3 \cdot 7^3}$
5	$\mp$0.90617 98459 38664 $\mp$0.53846 93101 05683 0	0.23692 68850 56189 0.47862 86704 99366 0.56888 88888 88889	$\dfrac{5h^{11} f^{(10)}(\xi)}{2^{36} \cdot 3^{16} \cdot 7^3 \cdot 11}$

astfel de funcţie şi să presupunem că avem de evaluat integrala acestei funcţii pe $[-1, 1]$. Putem da factor comun forţat termenul singular

$$\int_{-1}^{+1} f(z)\,dz = \int_{-1}^{+1} w(z)F(z)\,dz$$

unde $F(z)$ este acum o funcţie continuă. Cea de-a doua integrală se poate evalua prin folosirea formulelor Gauss-Cebâşev. Avantajul unui astfel de procedeu constă în izolarea comportării singulare din capete în funcţia pondere. Rădăcinile şi ponderile sunt (vezi şi 1.2.1)

$$z_i = \cos\left[(2i-1)\pi/2N\right], \quad A_i = \pi/N, \quad i = \overline{1,n}\,. \tag{2.90}$$

Integrala pe $[-1, 1]$ se evaluează ca

$$\int_{-1}^{+1} \frac{F(z)}{\sqrt{1-z^2}}\,dz = \frac{\pi}{N}\sum_{i=1}^{N} F(z_i) + \frac{2\pi F^{(2N)}(\zeta)}{2^N(2N)!}, \quad \zeta \in (-1,1) \tag{2.91}$$

unde ultimul termen este eroarea la integrare. Deoarece toate ponderile A_i de la formulele Gauss-Cebâşev au aceeaşi valoare, aceste formule necesită un număr de operaţii mai mic pentru atingerea unui anumit ordin de precizie, în comparaţie cu formulele Gauss-Legendre. De aceea, în situaţii precum efectuarea unui mare număr de integrale din interiorul unei bucle, se preferă utilizarea unei formule Gauss-Cebâşev.

Formule Gauss-Laguerre.

Un alt gen de comportare singulară apare la integrarea pe un interval semi-infinit. Ponderea adecvată este în acest caz $w(x) = \exp(-x)$ şi familia de polinoame ortogonale indusă de această funcţie pondere este cea a polinoamelor Laguerre din 1.4.2. Formula de calcul este în acest caz

$$\int_0^{\infty} e^{-x} f(x)\,dx = \sum_{i=1}^{N} A_i f(x_i) + \frac{(N!)^2}{(2N)!} f^{(2N)}(\xi), \quad \xi \in (0,\infty) \tag{2.92}$$

cu rădăcinile şi ponderile din tabelul 2.2.

Tabelul 2.2: Rădăcinile x_i, ponderile A_i şi eroarea la integrarea Gauss-Laguerre

N	x_i	A_i
2	0.58578 64376	8.53553 39059 E-1
	3.41421 35623	1.46446 60941 E-1
3	0.41577 45568	7.11093 00993 E-1
	2.29428 03603	2.78517 73357 E-1
	6.28994 50829	1.03892 56502 E-2
4	0.32254 76896	6.03154 10434 E-1
	1.74576 11012	3.57418 69244 E-1
	4.53662 02969	3.88879 08515 E-2
	9.39507 09123	5.39294 70556 E-1
5	0.26356 03197	5.21755 61058 E-1
	1.41340 30591	3.98666 81108 E-1
	3.59642 57710	7.59424 49682 E-2
	7.08581 00058	3.61175 86799 E-3
	12.64080 08443	2.33699 72386 E-5

Tabelul 2.3: Rădăcinile z_i, ponderile A_i şi eroarea la integrarea Gauss-Hermite

N	$\mp x_i$	A_i
2	0.70710 67812	8.86226 92545 E-1
3	0	1.18163 59006
	1.22474 48714	2.95408 97515 E-1
4	0.52464 76233	8.04914 09001 E-1
	1.65068 01239	8.13128 35447 E-2
5	0	9.45308 72048 E-1
	0.95857 24646	3.93619 32315 E-1
	2.02018 28705	1.99532 42059 E-2

Formule Gauss-Hermite.

În fine, pentru domeniile de integrare infinite se poate folosi ponderea $w(x) = \exp(-x^2)$ şi polinoamele Hermite asociate. Formula de integrare numerică este

$$\int_{-\infty}^{\infty} e^{-x^2} f(x)\,dx = \sum_{i=1}^{N} A_i f(x_i) + \frac{(N!)\sqrt{\pi}}{2^N (2N)!} f^{(2N)}(\xi), \quad \xi \in (-\infty, \infty) \tag{2.93}$$

cu rădăcinile şi ponderile din tabelul 2.3

Exemplu. (Calculul lungimii de arc al unui sfert de elipsă) Punctele de pe o elipsă satisfac ecuaţia $(x/a)^2 + (y/b)^2 = 1$ şi pot fi descrise parametric prin $x = a\cos\theta$, $y = b\sin\theta$. Presupunem $a > b$. Lungimea de arc cerută este

$$L = \int dl = \int \sqrt{dx^2 + dy^2} = \int_0^{\pi/2} \sqrt{a^2 \sin^2\theta + b^2 \cos^2\theta}\,d\theta\,. \tag{2.94}$$

Figura 2.2: Eroarea relativă ε funcţie de pasul convenţional $h = 2/n$ la integrare Gauss-Cebâşev.

Prin substituţia $z = \cos\theta$ se obţine

$$L = b\int_0^1 \frac{\sqrt{1+k^2z^2}}{\sqrt{1-z^2}}\,dz = \frac{b}{2}\int_{-1}^1 \frac{\sqrt{1+k^2z^2}}{\sqrt{1-z^2}}\,dz\,, \tag{2.95}$$

unde $k^2 \equiv (a^2-b^2)/b^2$ iar ultima egalitate s-a obţinut observând că funcţia de sub integrală este pară. Integralele de tipul de mai sus se numesc *integrale eliptice* şi formează un capitol important al analizei matematice. Se observă că am obţinut o integrală de forma

$$L = \frac{b}{2}\int_{-1}^1 \frac{f(z)}{\sqrt{1-z^2}}\,dz = \frac{b}{2}I\,, \tag{2.96}$$

în care apare ponderea Cebâşev $w(z) = 1/\sqrt{1-z^2}$. Pentru evaluarea integralei folosim formula (2.91) cu coeficienţii din (2.90). Pentru $a = 5$, $b = 4$ rezultă $k = 3/4$, iar şirul de aproximaţii succesive ale integralei ce se obţine mărind ordinul formulei Gauss-Cebâşev utilizate este

Ordin	2	3	4	5	6	7
I	3.556041	3.544605	3.545250	3.545205	3.545209	3.545208

(2.97)

Teoria funcţiilor eliptice furnizează valoarea $I = 3.54520849$. Se observă convergenţa rapidă către valoarea de referinţă. Este instructiv în acest sens graficul de convergenţă în coordonate logaritmice (fig. 2.2). Se observă pe acest grafic cum panta curbei scade continuu, denotând o scădere a erorii mai rapidă decât cea polinomială (pentru care graficul ar fi fost o linie dreaptă în coordonate logaritmice, vezi Exemplul din 2.1.1 şi fig. 2.1). O astfel de comportare a erorii se numeşte *spectrală* şi, în comparaţie cu comportările erorii $O(h^k)$ de la integrarea polinomului de interpolare, la o aproximare spectrală avem $\delta = O(h^\infty)$. Ordinul infinit se interpretează în sensul că eroarea scade mai repede decât orice putere a lui h. În cazul de faţă de exemplu între $n = 2$ şi $n = 3$ eroarea a avut o comportare $\delta = O(h^{0.7})$ pe când între $n = 16$ şi $n = 17$ comportarea a fost $\delta = O(h^{640})$.

2.2.3 Tehnici de atingere a unei precizii impuse

Pentru toate formulele de integrare prezentate anterior s-a determinat şi o expresie a erorii de integrare. Tipic, forma acestei expresii este

$$\delta = C\, f^{(k)}(\xi)\, h^m\,, \tag{2.98}$$

unde C este o constantă. Sunt foarte rare însă aplicaţiile în care derivata este cunoscută. După cum s-a menţionat, partea importantă a expresiei (2.98) este dependenţa de pasul h. Întrebarea ce apare totdeauna într-o aplicaţie este: cum se poate stabili pasul h necesar atingerii unei precizii impuse ε? Vom considera acum câteva tehnici ce permit determinarea preciziei de integrare. O trăsătură importantă a tehnicilor considerate este că ele pot fi aplicate împreună cu oricare dintre formulele de integrare considerate mai sus.

Reguli de integrare recursive.

Singura modalitate practică de verificare a convergenţei aproximărilor numerice ale unei integrale este repetarea calculelor cu un pas mai mic. O tehnică des utilizată este înjumătăţirea pasului. Considerăm intervalul $[a, b]$ şi o diviziune echidistantă a sa în N părţi egale. Pasul diviziunii este $h_N = (b-a)/N$ iar nodurile sunt $x_j^{(N)} = a + jh$, $j = \overline{0, N}$ [3]. Prin înjumătăţirea pasului obţinem $h_{2N} = (b-a)/2N$, $x_j^{(2N)} = a + jh$, $j = \overline{0, 2N}$. Deoarece $x_{2j}^{(2N)} = x_j^{(N)}$, $j = \overline{0, N}$, rezultă $y_{2j}^{(2N)} = y_j^{(N)}$ unde $y_{2j}^{(2N)} = f(x_{2j}^{(2N)})$, $y_j^{(N)} = f(x_j^{(N)})$. Deci valorile funcţiei cunoscute din calculul efectuat cu pasul h_N pot fi refolosite în calculul efectuat cu pasul h_{2N}. Noua aproximare a integralei se poate exprima recursiv utilizând vechea aproximare şi valorile funcţiei f în noile noduri $x_{2j+1}^{(2N)}$, $j = \overline{0, 2N-1}$. Să stabilim efectiv aceste reguli pentru două dintre formulele cele mai utilizate, cea a trapezelor (2.53) şi formula Simpson "1/3" (2.56).

Vom nota prin $T(h_N)$ aproximarea integralei I obţinută prin formula trapezelor (2.53) cu N intervale şi prin $T(h_{2N})$ aproximarea obţinută cu $2N$ intervale. $2T(h_N)/h_N$ şi $2T(h_{2N})/h_{2N}$ sunt

$$\begin{bmatrix} y_0^{(N)} & + & & + & 2y_1^{(N)} & + & \dots & + & 2y_{N-1}^{(N)} & + & & + & y_N^{(N)} \\ y_0^{(2N)} & + & 2y_1^{(2N)} & + & 2y_2^{(2N)} & + & \dots & + & 2y_{2N-2}^{(2N)} & + & 2y_{2N-1}^{(2N)} & + & y_{2N}^{(2N)} \end{bmatrix},$$

respectiv. Valorile egale au fost aliniate pe verticală. Ţinând cont că avem $h_{2N} = h_N/2$ se obţine

$$T(h_{2N}) = \frac{1}{2} T(h_N) + h_{2N} \sum_{j=1}^{N} y_{2j-1}^{(2N)}\,. \tag{2.99}$$

[3] Datorită operaţiilor succesive de înjumătăţire este preferabilă începerii numerotării nodurilor de la 0.

Formula trapezelor furnizează o regulă recursivă deosebit de simplă datorită faptului că avem aceeaşi coeficienţi numerici la formulele din etapele N şi $2N$. Pentru alte formule, regulile recursive sunt mai complicate datorită alternanţei coeficienţilor. Exemplificăm modalitatea de abordare pentru formula Simpson "1/3". Vom nota prin $S(h_{2N})$ suma din (2.56). Deoarece formula lucrează cu un număr par de intervale, vom face trecerea de la etapa $2N$ la etapa $4N$. Coeficienţii asociaţi valorilor nodale sunt reprezentaţi grafic pe schema de mai jos.

Valorile funcţiei ce trebuiesc calculate în etapa $4N$ sunt cele de indice impar. Pentru cele de indice par se pot refolosi rezultatele etapei $2N$. Datorită alternanţei coeficienţilor este convenabil să separăm suma $S(h_{2N})$ în suma componentelor pare şi a celor impare

$$S(h_{2N}) = S_{par}(h_{2N}) + S_{impar}(h_{2N})\,, \tag{2.100}$$

$$S_{par}(h_{2N}) = \frac{h_{2N}}{3}\left[y_0^{(2N)} + 2y_2^{(2N)} + \ldots + 2y_{2N-2}^{(2N)} + y_{2N}^{(2N)}\right]\,, \tag{2.101}$$

$$S_{impar}(h_{2N}) = \frac{4h_{2N}}{3}\left[y_1^{(2N)} + y_3^{(2N)} + \ldots + y_{2N-3}^{(2N)} + y_{2N-1}^{(2N)}\right]\,. \tag{2.102}$$

Regula recursivă de trecere la noua etapă este

$$S_{par}(h_{4N}) = \frac{1}{2}S_{par}(h_{2N}) + \frac{1}{4}S_{impar}(h_{2N}), \tag{2.103}$$

$$S(h_{4N}) = S_{par}(h_{4N}) + S_{impar}(h_{4N})\,. \tag{2.104}$$

Regulile recursive se folosesc în cadrul unei bucle de verificare a convergenţei aproximărilor. Exemplificăm, pentru regula recursivă Simpson "1/3", în algoritmul de mai jos. În urma aplicării algoritmului, se obţine un şir de aproximaţii ale integralei $S_1, S_2, \ldots, S_M$ calculate cu $2^1, 2^2, \ldots, 2^M$ subintervale.

$M \leftarrow 1, \quad N \leftarrow 2, \quad h \leftarrow (b-a)/2$
$S_{par} \leftarrow h(f(a)+f(b))/3, \quad S_{impar} \leftarrow 4hf(a+h)/3$
$S_1 \leftarrow S_{par} + S_{impar}$

$$
\begin{array}{l}
\text{repetă} \\
\quad \left[\begin{array}{l}
N \leftarrow 2N, \quad M \leftarrow M+1, \quad doih \leftarrow h, \quad h \leftarrow h/2 \\
S_{par} \leftarrow S_{par}/2 + S_{impar}/4 \\
x \leftarrow a+h, \quad S_{impar} \leftarrow 0 \\
k = 1:2:N \\
\quad \left[\begin{array}{l} S_{impar} \leftarrow S_{impar} + f(x) \\ x \leftarrow x + doih \end{array} \right. \\
S_{impar} \leftarrow 4hS_{impar}/3 \\
S_M \leftarrow S_{par} + S_{impar}
\end{array} \right. \\
\text{până când } |S_M - S_{M-1}| \le \varepsilon\,|S_M| + \varepsilon_S \text{ sau } M > M_{\max} \quad .
\end{array}
\tag{2.105}
$$

Condiţia de oprire a iteraţiei este cea de atingere a unei erori relative impuse ε

$$|(S_M - S_{M-1})/S_M| \le \varepsilon\,, \tag{2.106}$$

transcrisă convenabil pentru a permite şi $S_M = 0$[4] (sau foarte mic). În plus, se impune un număr maximal de subdivizări $M_{\max}$ ce nu trebuie depăşit.

Extrapolarea Richardson.

Sunt multe situaţii în care evaluarea funcţiei f este costisitoare şi este de dorit obţinerea unei precizii sporite cu un număr minim de evaluări ale funcţiei. Vom considera acum un procedeu de îmbunătăţire a aproximaţiilor ce se bazează pe cunoaşterea formei erorii de integrare (2.98). Considerăm mai întâi aproximarea prin formula trapezelor. Valoarea exactă a integralei este

$$I = T(h) + Cf^{(2)}(\xi_1)h^2\,. \tag{2.107}$$

Înjumătăţind pasul avem

$$I = T(h/2) + Cf^{(2)}(\xi_2)h^2/4\,, \tag{2.108}$$

cu $\xi_1 \neq \xi_2$ în general. Dacă se poate presupune însă că $f^{(2)}(\xi_1) \cong f^{(2)}(\xi_2)$[5] se poate elimina $Cf^{(2)}(\xi_1)$ din cele două ecuaţii de mai sus şi obţine

$$I \cong T(h/2) + [T(h/2) - T(h)]/3\,. \tag{2.109}$$

Tipic, noua aproximare este de un ordin de precizie mai ridicat

$$I = T(h/2) + [T(h/2) - T(h)]/3 + O(h^k)\,, \tag{2.110}$$

cu $k > 2$. Acelaşi procedeu aplicat pentru formula Simpson, la care avem

$$I = S(h) + Cf^{(4)}(\xi)h^4\,, \tag{2.111}$$

conduce la

$$I = S(h/2) + [S(h/2) - S(h)]/15 + O(h^k), \quad k > 4\,. \tag{2.112}$$

[4] ε_S este denumit coeficient de siguranţă şi permite oprirea iteraţiilor şi când $S_M = 0$. Uzual se ia $\varepsilon_S = \varepsilon$ şi condiţia de oprire a iteraţiilor se scrie $|S_M - S_{M-1}| \le \varepsilon(1 + |S_M|)$.

[5] Presupunerea $f^{(2)}(\xi_1) \cong f^{(2)}(\xi_2)$ implică fie că ξ_1 nu diferă de ξ_2 fie că $f^{(2)}(x)$ nu are variaţii importante pe intervalul considerat.

Algoritmul Romberg.

Punctul slab al tehnicii de mai sus este presupunerea $f^{(2)}(\xi_1) \cong f^{(2)}(\xi_2)$. Există însă o cale de a justifica acest pas pentru funcţii netede, $f \in C^{(2m)}[a,b]$. Punctul de pornire îl constituie formula Euler-MacLaurin[6]

$$\int_0^1 F(t)\,dt = \frac{F(0)+F(1)}{2} + \sum_{k=1}^{m-1} A_{2k}\left[F^{(2k-1)}(0) - F^{(2k-1)}(1)\right] - A_{2m}F^{(2m)}(\xi_0),$$

cu $\xi_0 \in [0,1]$ şi A_{2k} constante numerice. Aplicăm formula pentru $F(t) = f(x_j + t\,h)$ şi obţinem

$$\begin{aligned}\int_{x_j}^{x_{j+1}} f(x)\,dx &= \frac{h\,[f(x_j)+f(x_{j+1})]}{2} + \sum_{k=1}^{m-1} A_{2k}h^{2k}\left[f^{(2k-1)}(x_j) - f^{(2k-1)}(x_{j+1})\right] \\ &\quad -A_{2m}h^{2m+1}f^{(2m)}(\xi_i)\,.\end{aligned}$$

Prin sumarea $\sum_{j=0}^{2^N-1}$, se obţine

$$\begin{aligned}\int_a^b f(x)\,dx &= \sum_{j=0}^{2^N-1} \frac{h\,[f(x_j)+f(x_{j+1})]}{2} + \sum_{k=1}^{m-1} A_{2k}h^{2k}\left[f^{(2k-1)}(a) - f^{(2k-1)}(b)\right] \\ &\quad -A_{2m}(b-a)h^{2m}f^{(2m)}(\xi)\,.\end{aligned}$$

Prima sumă este chiar aproximarea prin formula trapezelor $T(h)$, astfel încât avem

$$I = \int_a^b f(x)\,dx = T(h) + a_2h^2 + a_4h^4 + \ldots + a_{2m}h^{2m}f^{(2m)}(\xi)\,. \qquad (2.113)$$

De data aceasta, observăm că s-a obţinut un coeficient a_2 al primului termen din eroare, ce este constant. Ca atare, extrapolarea Richardson este justificată. De fapt procedeul de extrapolare poate fi repetat de $m-1$ ori. Această combinaţie dintre regula trapezelor recursivă şi extrapolare Richardson este cunoscută ca *metoda Romberg*. Fie $R(n,m)$ estimarea integralei obţinută cu 2^n subintervale şi m etape de extrapolare Richardson. Avem $R(n,0) = T((b-a)/2^n)$ şi

$$R(n,m) = R(n,m-1) + \frac{1}{4^m-1}\left[R(n,m-1) - R(n-1,m-1)\right]\,. \qquad (2.114)$$

În scrierea programelor de calcul, este recomandată utilizarea formei de mai sus în care se adaugă o corecţie $R(n,m-1) - R(n-1,m-1)$, la o aproximaţie deja efectuată $R(n,m-1)$, spre deosebire de forma

$$R(n,m) = \frac{4^m}{4^m-1}R(n,m-1) - \frac{1}{4^m-1}R(n-1,m-1)\,. \qquad (2.115)$$

[6]Formula rezultă din integrarea prin părţi repetată şi alegerea convenabilă a constantelor de integrare: $\int_0^1 F(t)\,dt = \left[(t-\frac{1}{2})F(t)\right]_0^1 - \int_0^1 (t-\frac{1}{2})\,F'(t)\,dt = \ldots$

Calculul se organizează convenabil într-un tabel

$$\begin{array}{lllll} R(0,0) & & & & \\ R(1,0) & R(1,1) & & & \\ R(2,0) & R(2,1) & R(2,2) & & \\ \vdots & \vdots & \vdots & \ddots & \\ R(M,0) & R(M,1) & R(M,2) & \cdots & R(M,M) \end{array} \tag{2.116}$$

ce se parcurge în sensul umplerii triunghiurilor inferioare: $R(0,0)$, $R(1,0)$, $R(1,1)$, $R(2,0)$, $R(2,1)$, $R(2,2)$, ... Noi evaluări ale funcţiei f sunt necesare doar pentru prima coloană ce conţine $R(0,0)$, $R(1,0)$, ... Algoritmul este

$n \leftarrow 0; \quad h \leftarrow b-a; \quad N \leftarrow 1$
$R(0,0) \leftarrow h(f(a)+f(b))/2$
repetă
  $n \leftarrow n+1; \quad N \leftarrow 2N; \quad doih \leftarrow h; \quad h \leftarrow h/2$
  $R(n,0) = 0; \quad x \leftarrow a+h$
  $k = 1:2:N$
    $R(n,0) \leftarrow R(n,0) + f(x)$
    $x \leftarrow x + doih$
  $R(n,0) \leftarrow R(n-1,0)/2 + h\,R(n,0)$
  $AtinsPrecizie \leftarrow fals; \quad m \leftarrow 0; \quad putere4 \leftarrow 1$
  repetă
    $putere4 \leftarrow 4\,putere4; \quad m \leftarrow m+1$
    $dif \leftarrow R(n,m-1) - R(n-1,m-1)$
    $R(n,m) \leftarrow R(n,m-1) + dif\,/\,(putere4-1)$
    $AtinsPrecizie \leftarrow |R(n,m) - R(n,m-1)| \le \varepsilon\,(1+|R(n,m-1)|)$
  până când $m > n$ sau *AtinsPrecizie*
până când *AtinsPrecizie* sau $n > n_{\max}$.

Partea cea mai sensibilă a algoritmului este stabilirea unei condiţii de oprire. Cea de mai sus este simplă dar funcţionează bine doar pentru funcţii netede. În aplicaţii ale procedeului trebuie atent eşantionată funcţia de sub integrală pentru a determina prezenţa unor singularităţi sau zone de variaţii rapide ce ar împiedica ameliorarea aproximaţiilor prin extrapolarea Richardson.

Integrarea adaptivă.

Pe lângă atingerea unei precizii impuse, în practică, se doreşte atingerea acestei precizii cu un număr minim de evaluări ale funcţiei f. Algoritmii consideraţi până acum prezintă dezavantajul că realizează o diviziune a întregului interval de integrare. Ori, este adesea cazul că subdivizarea este necesară numai pe zone restrânse, de variaţie mai rapidă a funcţiei f (fig. 2.3). Strategia adecvată este de verificare a îndeplinirii unei condiţii de eroare pe fiecare subinterval în parte şi de subdivizare numai a acelor subintervale pe care această condiţie nu este îndeplinită.

Algoritmul poate fi exprimat elegant sub formă recursivă. Fie F o formulă de integrare pe care o vom aplica pentru a calcula $\int_a^b f(x)\,dx$, cu eroarea relativă ε. Notăm prin $P(F,a,b,y_a,y_b,\varepsilon)$ funcţia de integrare adaptivă. S-au introdus ca argumente şi $y_a \equiv f(a)$, $y_b \equiv f(b)$, pentru a se putea refolosi evaluările de funcţie în capetele intervalului, atunci când se face o subdivizare. Algoritmul este

$$
\begin{array}{l}
P(F,a,b,y_a,y_b,\varepsilon) \\
\quad \left[\begin{array}{l}
I_{ab} \leftarrow F(a,b,y_a,y_b) \\
c \leftarrow (a+b)/2, \quad y_c \leftarrow f(c) \\
I_{ac} \leftarrow F(a,c,y_a,y_c), \quad I_{cb} \leftarrow F(c,b,y_c,y_b) \\
I_{nou} \leftarrow I_{ac} + I_{cb} \\
\text{dacă } |I_{nou} - I_{ab}| \le \varepsilon(1+|I_{nou}|) \text{ atunci } P \leftarrow I_{nou} \\
\text{altfel } P \leftarrow P(F,a,c,y_a,y_c,\varepsilon) + P(F,c,b,y_c,y_b,\varepsilon)
\end{array} \right.
\end{array}
\tag{2.117}
$$

Pentru intervalul $[a,b]$, se efectuează două evaluări ale integralei, folosind formula F. Prima I_{ab} utilizează tot intervalul, cealaltă I_{nou} utilizează o divizare în două a intervalului. Dacă, între cele două evaluări condiţia de eroarea relativă este îndeplinită, se acceptă evaluarea mai precisă I_{nou}. Altfel se repetă aplicarea procedeului pe cele două subintervale $[a,c]$, $[c,b]$. Procedeul se zice *recursiv* deoarece funcţia P se autoapelează.

Integrarea adaptivă poate fi cuplată cu oricare dintre formulele de integrare prezentate anterior. Pentru formula Simpson "1/3" avem

$$
\begin{array}{l}
F(a,b,y_a,y_b) \\
\quad \left[\begin{array}{l}
h \leftarrow (b-a)/2, \quad y_m \leftarrow f(a+h) \\
F \leftarrow h\,(y_a + 4y_m + y_b)/3
\end{array} \right.
\end{array}
\quad , \tag{2.118}
$$

iar pentru formula Gauss-Legendre de ordinul III (vezi tabel 2.1):

$$
\begin{array}{l}
F(a,b,y_a,y_b) \\
\quad \left[\begin{array}{l}
z_1 = -\sqrt{3/5}, \quad z_2 = 0, \quad z_3 = \sqrt{3/5} \\
A_1 = 5/9, \quad A_2 = 8/9, \quad A_3 = 5/9 \\
m \leftarrow (b-a)/2, \quad n \leftarrow (a+b)/2 \\
j = 1:3 \\
\quad x_j = m\,z_j + n \\
F \leftarrow m\,(A_1 f(x_1) + A_2 f(x_2) + A_3 f(x_3))
\end{array} \right.
\end{array}
\quad . \tag{2.119}
$$

Exemplu 1. Considerăm integrala

$$
I = \int_0^{10} f(x)dx, \quad f(x) = \frac{1}{(x-1)^2+0.01} + \frac{1}{(x-3)^2+0.04} \tag{2.120}
$$

ce se poate evalua analitic, $I = 45.54044$. Ne propunem să comparăm diversele metode de integrare pe acest caz. Impunem atingerea unei erori relative de $\varepsilon = 10^{-6}$ (6 cifre semnificative exacte). Prin regulile recursive, pentru formula trapezelor şi formula Simpson "1/3", se obţin rezultatele din tabelul următor.

Figura 2.3: O funcţie cu variaţii rapide în jurul absciselor $x = 1$, $x = 3$ şi lente în rest. Sunt figurate pe axa absciselor şi pe graficul funcţiei nodurile folosite la integrarea adaptivă cu formula Simpson "1/3".

M	trapeze	Simpson "1/3"	M	trapeze	Simpson "1/3"
1	4.38356	14.67353	7	45.54563	45.53835
2	12.10104	30.84147	8	45.54017	45.54044
3	26.15636	32.09992	9	45.54037	45.54044
4	30.61403	51.99130	10	45.54042	
5	46.64699	43.96342	11	45.54043	

(2.121)

cu 2049 de apeluri ale funcţiei f pentru formula trapezelor şi 1025 pentru formula Simpson până la atingerea erorii impuse.

Aplicăm acum extrapolarea Richardson asupra rezultatelor de la metoda trapezelor (metoda Romberg). Se obţine tabelul de mai jos.

nØm	0	1	2	3	4
0	5.667271				
1	4.383564	3.955662			
2	12.101037	14.673528	15.388053		
3	26.156360	30.841468	31.919330	32.181731	
4	30.614030	32.099919	32.183816	32.188014	32.188039

(2.122)

Algoritmul se opreşte la $n = 4$, $m = 4$, deoarece corecţia adusă de extrapolare este mică, lăsând primele 6 cifre semnificative nemodificate. Dar rezultatul este foarte departe de cel exact şi de fapt nu avem nici o cifră semnficativă corectă. Funcţia f prezintă două puncte de variaţie foarte rapidă, la $x = 1$ şi $x = 3$. Acestea apar ca nişte singularităţi dacă rata de eşantionare a funcţiei f nu este suficient de fină.

În fine, comparăm diversele formule de integrare în interiorul procedurii adaptive $P(F, a, b, y_a, y_b, \varepsilon)$. Se obţin rezultatele

Formulă	Trapeze	Simpson "1/3"	GL II	GL III	GL IV
I	45.5419	45.5405	45.5404	45.5405	45.5404
evaluări f	839	557	877	573	485

(2.123)

unde prin GLn s-a notat formula Gauss-Legendre de ordinul n. Procedeul adaptiv determină în general corect cele 6 cifre semnificative cerute. Excepţie face doar metoda trapezelor. Nesatisfacerea preciziei în acest caz este rezultatul condiţiei de oprire a iteraţiilor. S-a presupus necunoscută valoarea exactă şi s-au comparat două aproximaţii succesive pentru a se estima eroarea. În termenii din capitolul 1 s-a aplicat criteriul de convergenţă pentru şiruri Cauchy. Criteriul este însă strict valabil doar când $n \to \infty$. Exemplul de aici ne arată pericolul identificării aproprierii dintre două aproximaţii cu aproprierea aproximaţiilor de soluţia exactă. Pentru a preveni o estimare eronată a preciziei cu care s-a determinat un rezultat trebuie impusă o condiţie de eroare mai strictă decât cea dorită efectiv în rezultatul final. Refăcând calculul cu $\varepsilon = 10^{-7}$ se obţine $I = 45.5407$ pentru metoda trapezelor cu 1801 evaluări ale funcţiei f.

Numărul de evaluări de funcţie scade cu creşterea ordinului de integrare. Câştigul devine din ce în ce mai mic însă. Diferenţa de număr de evaluări între Simpson "1/3" şi GLII ce sunt procedee cu comportări comparabile ale erorii provine de la posibilitatea reutilizării nodurilor în formula Simpson. La formulele Gauss-Legendre modificarea intervalului conduce şi la modificarea poziţiei nodurilor.

Prin comparaţie cu regula recursivă Simpson se observă reducerea semnificativă a numărului de evaluări ale funcţiei f în cazul procedurii adaptive ce utilizează formula Simpson "1/3". Nodurile folosite pentru atingerea unei erori relative de $\varepsilon = 10^{-3}$ sunt reprezentate în fig. 2.3. Se observă îndesirea nodurilor în zonele de variaţie rapidă a funcţiei. Pentru obţinerea aceleaşi precizii regula recursivă Simpson împarte tot intervalul cu aceeşi fineţe rezultând multe evaluări ale funcţiei inutile, în zonele de variaţie lentă.

Exemplul 2. Fie acum integrala

$$I = \int_0^{10} f(x)dx, \quad f(x) = \frac{1}{(x-1)^2+1} + \frac{1}{(x-3)^2+4} \tag{2.124}$$

ce are valoarea exactă $I = 3.38318$. Funcţia este asemănătoare celei din exemplu precedent dar are un aspect mult mai neted. Procedeul Romberg furnizează tabelul:

nØm	0	1	2	3	4	5
0	3.039931					
1	2.439083	2.238800				
2	2.737904	2.837511	2.877425			
3	3.326170	3.522259	3.567908	3.578868		
4	3.365313	3.378361	3.368768	3.365607	3.364771	
5	3.378763	3.383246	3.383572	3.383807	3.383878	3.383897
6	3.382077	3.383182	3.383178	3.383172	3.383169	

Valoarea la care s-a îndeplinit condiţia de eroare este $I = 3.38317$ după 65 de evaluări ale funcţiei f. Funcţia fiind netedă extrapolarea Richardson funcţionează corect obţinându-se şi un efort de calcul mult inferior celor 1025 de evaluări de funcţie necesare atingerii aceleaşi precizii prin regula trapezelor recursivă (de 16 ori mai puţine calcule!).

Capitolul 3

Rezolvarea ecuaţiilor neliniare

Vom aborda, la început problema găsirii rădăcinilor *reale* ale unei funcţii $F(x)$, definită pe un interval $[a, b]$, adică a valorilor variabilei x pentru care are loc relaţia

$$F(x) = 0, \quad x \in [a, b]. \tag{3.1}$$

Dacă $F(x)$ este o funcţie liniară problema este banală şi admite o soluţie analitică. Ca atare vom presupune că $F(x)$ este neliniară, caz în care problema devine dificilă datorită absenţei vreunei teoreme matematice ce să furnizeze numărul rădăcinilor sau vecinătăţile în care acestea se află în cazul general. De aceea o primă etapă în rezolvarea ecuaţiei (3.1) este *căutarea rădăcinilor* pentru care se folosesc rezultate din analiza matematică, precum teorema lui Rolle, sau eşantionarea funcţiei $F(x)$. După determinarea intervalelor pe care se află rădăcini se trece la restrângerea acestor intervale, proces cunoscut şi sub denumirea de *localizare a rădăcinilor*. Localizarea rădăcinilor se efectuează de regulă prin procedee sigure dar nu deosebit de eficiente. Pentru a obţine rapid o precizie ridicată a aproximaţiei unei rădăcini se efectuează o ultimă etapă de *rafinare a rădăcinilor*.

Uzual, procedeele de rezolvare a ecuaţiei (3.1) furnizează un şir de aproximaţii $\{x_n\}_{n\in\mathbf{N}}$ ce tinde către o rădăcină α. În analiza acestor procedee este util să se

stabilească o modalitate de cuantificare a rapidităţii cu care şirul tinde către limită. Spunem că şirul $\{x_n\}_{n\in\mathbf{N}}$ *converge liniar* către α dacă există $c<1, N\in\mathbf{N}$ astfel încât

$$|x_{n+1}-\alpha| \le c\,|x_n-\alpha|\,, \quad n>N\,. \tag{3.2}$$

Sirul $\{x_n\}_{n\in\mathbf{N}}$ *converge pătratic* către α dacă există C, N astfel încât

$$|x_{n+1}-a| \le C\,|x_n-a|^2\,, \quad n>N\,. \tag{3.3}$$

Analog se pot defini ordine de convergenţă superioare, existând însă, un număr mic de algoritmi ce au ordine de convergenţă mai mari de 2. Multe metode prezintă un ordin de convergenţă intermediar între cel liniar şi cel pătratic. Spunem că şirul $\{x_n\}_{n\in\mathbf{N}}$ *converge superliniar* către α dacă există N şi un alt şir $\{\varepsilon_n\}_{n\in\mathbf{N}}$ cu $\lim_{n\to\infty}\varepsilon_n=0$ astfel încât

$$|x_{n+1}-a| \le |\varepsilon_n|\,|x_n-a|\,, \quad n>N \tag{3.4}$$

Ordinul de convergenţă va indica numărul de iteraţii ce trebuiesc efectuate pentru a atinge o anumită precizie. Metodele cu ordin de convergenţă mai ridicat tind să efectueze mai multe operaţii aritmetice pe iteraţie. De aceea este necesară şi o cuantificare a efortului de calcul într-o iteraţie pentru a avea o estimare realistă a timpului de calcul. Uzual, unitatea de lucru adoptată este numărul de evaluări ale funcţiei $F(x)$ într-o iteraţie.

3.1 Metoda înjumătăţirii intervalelor

Metoda este exemplul clasic de un procedeu de localizare. Vom presupune că din etapa de căutare s-a determinat un interval $[a,b]$ pentru care $F(a)F(b)<0$, ceea ce denotă existenţa a cel puţin unei rădăcini (eventual un număr impar) în acest interval. Metoda înjumătăţirii împarte intervalul în două părţi egale prin punctul

$$c=(a+b)/2\,. \tag{3.5}$$

Dacă $F(c)=0$, atunci c este o rădăcină. Dacă $F(c)\neq 0$, se calculează semnul produsului $F(a)F(c)$. Dacă acesta este negativ, $F(a)F(c)<0$, atunci rădăcina se află în intervalul $[a,c]$, altfel rădăcina se află în $[c,b]$. Calculele se reiau pe noul subinterval.

Metoda este simplă şi necesită doar o evaluare de funcţie per iteraţie. Convergenţa este însă doar liniară ($c=1/2$ în 3.2), algoritmul efectuând tipic un

mare număr de iteraţii. Algoritmul este

$$
\begin{array}{l}
F_a \leftarrow F(a); \quad F_b \leftarrow F(b); \quad m \leftarrow 0; \quad e \leftarrow b-a \\
\text{repetă} \\
\quad \left[\begin{array}{l}
e \leftarrow 0.5e; \quad c \leftarrow a+e; \quad F_c = F(c) \\
\text{dacă semn}(F_a) = \text{semn}(F_c) \text{ atunci} \\
\quad [b \leftarrow c; \quad F_b \leftarrow F_c \\
\text{altfel} \\
\quad [a \leftarrow c; \quad F_a \leftarrow F_c \\
m \leftarrow m+1
\end{array} \right. \\
\text{până când } m > m_{\max} \text{ sau } |b-a| < \delta \text{ sau } |F_c| < \varepsilon
\end{array}
\tag{3.6}
$$

Se poate observa utilizarea a unui cumul de trei condiţii de oprire: depăşirea numărului maxim de iteraţii, restrângerea intervalului sub un prag ales δ, sau obţinerea unei valori a funcţiei mai mici decât numărul mic ε. În algoritm valoarea c este actualizată prin adăgarea unei mici corecţii e, spre deosebire de transcrierea directă a formulei (3.5). Aceasta conduce tipic la un control mai bun al erorilor numerice pe parcursul iteraţiilor.

3.2 Procedee iterative

Metoda înjumătăţirii intervalului este simplă dar converge lent, deci nu este indicată atunci când se doreşte o precizie ridicată (un număr mare de cifre semnificative exacte). De obicei, metoda înjumătăţirii intervalului serveşte pentru stabilirea intervalului pe care există o rădăcină reală, după care se aplică procedee care converg mai rapid.

3.2.1 Iteraţia simplă

Se scrie ecuaţia (3.1) sub forma

$$F(x) = x - f(x) = 0\,, \tag{3.7}$$

unde $f(x)$ se numeşte *funcţie iterată*, iar $[a,b]$ este un interval pe care ştim că există o rădăcină $x = \alpha$ (separată, spre exemplu, cu metoda înjumătăţirii intervalelor). Prin urmare

$$F(\alpha) = \alpha - f(\alpha) = 0, \quad \alpha \in (a,b). \tag{3.8}$$

Se construieşte următorul şir, în mod recurent,

$$x_{k+1} = f(x_k), \quad k = 0,1,2,\ldots \quad . \tag{3.9}$$

Vom căuta să stabilim în ce condiţii şirul (3.9) converge spre valorea α, plecând de la un punct arbitrar $x_0 \in (a,b)$. În acest scop, vom calcula distanţa la rădăcină

$$|x_{k+1} - \alpha| = |f(x_k) - f(\alpha)| = |f'(\xi)|\,|x_k - \alpha|\,, \quad \xi \in (x_k, \alpha)\,. \tag{3.10}$$

Am ţinut seama că α este rădăcină, şi am folosit formula creşterilor finite (Lagrange) presupunând că $f(x)$ este derivabilă pe intervalul (a,b). Notând cu m cea mai mare valoare a modulului derivatei $f\prime(x)$

$$m = \sup_{x\in(a,b)} |f'(x)| \,, \tag{3.11}$$

din (3.10) se obţine

$$|x_{k+1} - \alpha| \le m\,|x_k - \alpha| \,. \tag{3.12}$$

Rezultă că, pentru ca şirul x_{k+1} să aibă ca limită pe α este suficient ca

$$m < 1 \,, \tag{3.13}$$

convergenţa procedeului fiind liniară. Condiţia (3.13) este şi necesară dacă dorim ca şirul (3.9) să conveargă spre α, oricare ar fi punctul de plecare x_0 . Astfel, valorile derivatei, în modul, trebuie să fie subunitare într-o vecinătate a rădăcinii α, inclusă în intervalul pe care căutăm rădăcina. Şirul (3.9) defineşte un procedeu de determinare a unei rădăcini reale denumit *iteraţie simplă*. Forma (3.7) a ecuaţiei are o interpretare geometrică simplă, ilustrată în fig. 3.1. Se observă că, rădăcinile reale, dacă există, se găsesc la intersecţia funcţiei $y = f(x)$ cu prima bisectoare. Plecând dintr-un punct x_0, este accesibilă doar rădăcina $x = \alpha_2$ în vecinătatea căreia $|f'(x)| < 1$. Pe de altă parte $|f'(x_0)| > 1$ deci condiţia (3.13) nu este întodeauna necesară. Pentru oprirea iteraţiilor un criteriu simplu este ca distanţa între doua aproximaţii succesive sa fie suficient de mică. Fie ε precizia dorită şi $n+1$ rangul ultimei iteraţii astfel încât

$$|x_{n+1} - x_n| \le \varepsilon \,. \tag{3.14}$$

Mai departe, se obţine succesiv

$$|x_{n+1} - x_n| = |f(x_n) - x_n| = |F(x_n)| = |F(x_n) - F(\alpha)| = \tag{3.15}$$

$$|x_n - \alpha|\,|F'(\xi_n)| = |x_n - \alpha|\,|1 - f'(\xi_n)| \ge |x_n - \alpha|\,|1 - m| \tag{3.16}$$

Ţinând, mai departe cont de (vezi şi 3.10)

$$|x_{n+1} - \alpha| \le m\,|x_n - \alpha| \,, \quad |x_n - \alpha| \le m\varepsilon/(1-m) \tag{3.17}$$

rezultă că distanţa la rădăcina ultimei valori calculate, x_{n+1} satisface inegalitatea

$$|x_{n+1} - \alpha| \le \varepsilon\, m/(1-m) \,. \tag{3.18}$$

În consecinţă, dacă panta m este foarte apropiată de unitate, distanţa la rădăcină poate fi de alt ordin de mărime decât ε ales. În particular, dacă nu s-a verificat condiţia de existenţă a rădăcinii, s-ar putea ca aceasta să nici nu existe (fig. 3.1). De aceea, în programul de calcul, se recomandă testarea valorii $|f'(x)|$.

Figura 3.1: (a) Exemplu de convergenţă a iteraţiei simple. (b) Exemplu de divergenţă a iteraţiei simple.

3.2.2 Metoda Newton-Raphson

Condiţia (3.13) este destul de restrictivă, anumite rădăcini neputând fi atinse (cum ar fi rădăcina corespunzătoare punctului A din fig. 3.1). De aceea, ne propunem construirea unui alt şir iterativ ce să tindă spre rădăcina $x = \alpha$,

$$x_{k+1} = x_k + \beta\,(f(x_k) - x_k)\,, \quad k = 0, 1, 2, \ldots\,, \tag{3.19}$$

unde s-a introdus parametrul de ajustare β. Evident, pentru $\beta = 1$ se obţine iteraţia simplă. Pe de altă parte, distanţa la rădăcină este

$$|x_{k+1} - \alpha| \le |x_k - \alpha + \beta\,(f(x_k) - f(\alpha) + \alpha - x_k)| = \tag{3.20}$$

$$|x_k - \alpha|\,|1 - \beta\,(1 - f'(\xi_k))|\,, \quad \xi_k \in (x_k, \alpha)\,. \tag{3.21}$$

Pentru a realiza convergenţa şirului (4.15) este suficient să avem

$$|1 - \beta\,(1 - f'(x))| < 1, \quad x \in (a, b) \tag{3.22}$$

pe intervalul unde căutăm rădăcina. Se observă că am înlocuit ξ_k cu x, deoarece ξ_k este, în general, necunoscut. În particular, dacă am cunoaşte ξ_k, luând

$$\beta = 1/\left[1 - f'(\xi_k)\right] \tag{3.23}$$

am avea $x_{k+1} = \alpha$, adică rădăcina ar fi determinată.

Pentru $\xi_k = x_k$, din (3.23) şi (3.19), se obţine formula iterativă *Newton-Raphson*

$$x_{k+1} = x_k - F(x_k)/F'(x_k), \quad k = 0, 1, \ldots\,, F'(x_k) \neq 0\,, \tag{3.24}$$

ale cărei condiţii de convergenţă le vom preciza mai jos. Din punct de vedere geometric, procedeul iterativ (3.24) corespunde apropierii de rădăcină prin punctele de intersecţie ale tangentelor duse la curba $F(x)$ în punctele $(x_k, F(x_k))$ (fig. 3.2). Altfel spus, următoarea aproximaţie a rădăcinii x_{k+1} este dată de rădăcina unei *liniarizări* a funcţiei $F(x)$. Aceeaşi idee, de aproximare locală liniară, rezultă şi pe cale analitică din formula lui Taylor

$$F(x) = F(x_k) + (x - x_k)F'(x_k) + (x - x_k)^2 F''(\xi_k)/2, \quad \xi_k \in (x, x_k)\,. \tag{3.25}$$

Aproximarea liniară a funcţiei $F(x)$ va fi

$$L(x) = F(x_k) + (x - x_k)F'(x_k)\,, \tag{3.26}$$

ce are rădăcina dată de formula (3.24).

Stabilim acum condiţiile de convergenţă a metodei Newton-Raphson. Pentru $x = \alpha$, din (3.25) rezultă

$$0 = F(x_k) + (\alpha - x_k)F'(x_k) + (\alpha - x_k)^2 F''(\xi_k)/2\,. \tag{3.27}$$

sau, considerând $F'(x_k)$ diferit de zero,

$$\alpha = x_k - \frac{F(x_k)}{F'(x_k)} - \frac{(x_k - \alpha)^2}{2F'(x_k)} F''(\xi_k)\,. \tag{3.28}$$

Scazând relaţiile (3.24) şi (3.28) se obţine distanţa la rădăcină în aproximaţia $k+1$

$$|x_{k+1} - \alpha| = (x_k - \alpha)^2 \left|F''(\xi_k)/F'(x_k)\right|/2 \le M(x_k - \alpha)^2\,, \tag{3.29}$$

unde am notat cu M marginea

$$M = \sup_{x', x'' \in (a,b)} \left|F''(x'')/F'(x')\right|, \quad F'(x') \neq 0\,. \tag{3.30}$$

Relaţia (3.29) arată că de la o etapă k la o etapă $k+1$ distanţa scade aproximativ cu pătratul distanţei la rădăcina din etapa precedentă, adică metoda Newton-Raphson are o *viteză de convergenţă pătratică*, superioară aproximaţiei simple. Efortul de calcul este de două unităţi de lucru deoarece la fiecare iteraţie se evaluează atât funcţia cât şi derivata acesteia. Astfel, notând cu x_0 punctul de start, se obţine

$$|x_{k+1} - \alpha| \le M^{-1} \left[M(x_0 - \alpha)\right]^{2^{k+1}}\,. \tag{3.31}$$

În consecinţă, o condiţie suficientă de convergenţă este

$$M\,|x_0 - \alpha| < 1\,. \tag{3.32}$$

Dacă M este finit, condiţia (3.32) este satisfăcută întotdeauna *dacă plecăm suficient de aproape de rădăcină*. Altă condiţie de convergenţă pentru metoda

Figura 3.2: Reprezentarea geometrică a iteraţiilor din metoda Newton-Raphson (a) şi secantei (b).

Newton-Raphson este următoarea: *metoda converge întotdeauna dacă în vecinătatea rădăcinii derivatele $F'(x)$ şi $F''(x)$ păstrează un semn constant.*

Dacă s-a stabilit că intervalul pe care se găseşte rădăcina este (a, b), se poate pleca cu aproximaţia iniţială

$$x_0 = (a + b)/2\,. \tag{3.33}$$

Se recomandă ca în orice etapă să controlăm între ce valori se găseşte rădăcina, tangenta putând fi dusă dintr-un capăt al intervalului sau din celălalt (dacă noua valoare calculată cu (3.24) iese în afara intervalului în care se găseşte rădăcina). Dacă se iese din interval indiferent de capătul unde se duce tangenta, se recomandă o nouă înjumătăţire a intervalului.

3.2.3 Metoda secantei

Metoda Newton-Raphson prezintă dezavantajul necesităţii cunoaşterii derivatei funcţiei $F(x)$. Putem însă înlocui linearizarea ce foloseşte tangenta la graficul funcţiei cu cea ce foloseşte secanta construită cu două iteraţii succesive (fig. 3.2). Formula de iterare este

$$x_{k+1} = x_k - (x_k - x_{k-1})\, F(x_k) / \left[F(x_k) - F(x_{k-1})\right]\,, \tag{3.34}$$

ceea ce revine la a înlocui

$$F'(x_k) \cong \left[F(x_k) - F(x_{k-1})\right] / \left(x_k - x_{k-1}\right) \tag{3.35}$$

în (3.24), adică o aproximare cu diferenţe finite a derivatei. Pentru pornirea metodei este nevoie de două valori x_0, x_1. Uzual, dacă se cunoaşte intervalul $[a, b]$ pe care se află o rădăcină, se ia $x_0 = a$, $x_1 = b$. În acest caz metoda poate fi interpretată ca o îmbunătăţire a metodei înjumătăţirii intervalului prin faptul

că împarte intervalul $[a, b]$ în segmente proporţionale, dintre care cel mai mic conţine rădăcina (fig. 3.3).

În vederea analizei convergenţei metodei notăm $F_k = F(x_k)$ şi $e_k = x_k - \alpha$, eroarea în iteraţia k. Avem

$$e_{k+1} = x_{k+1} - \alpha = x_k - \frac{(x_k - x_{k-1})\, F_k}{F_k - F_{k-1}} - \alpha = \frac{F_k e_{k-1} - F_{k-1} e_k}{F_k - F_{k-1}}\,. \tag{3.36}$$

Dând factor comun forţat pe $e_k e_{k-1}$ şi inserând raportul unitar $(x_k - x_{k-1})/(x_k - x_{k-1})$ se obţine

$$e_{k+1} = e_k e_{k-1} \left[\frac{x_k - x_{k-1}}{F_k - F_{k-1}}\right] \left[\frac{F_k/e_k - F_{k-1}/e_{k-1}}{x_k - x_{k-1}}\right]\,. \tag{3.37}$$

Dezvoltarea în serie Taylor în jurul rădăcinii α

$$F(x_k) = F(\alpha + e_k) = F(\alpha) + e_k F'(\alpha) + e_k^2 F''(\alpha)/2 + O(e_k^3) \tag{3.38}$$

conduce la

$$F_k/e_k = F'(\alpha) + e_k F''(\alpha)/2 + O(e_k^2)\,. \tag{3.39}$$

Înlocuind în (3.37) se obţine relaţia de recurenţă în erori

$$e_{k+1} \cong [F''(\alpha)/2F'(\alpha)] \cdot e_k e_{k-1}, \tag{3.40}$$

asemănătoare relaţiei (3.29) de la procedeul Newton-Raphson. În deducerea relaţiei (3.40) s-a înlocuit aproximarea prin diferenţe finite a derivatei cu $F'(\alpha)$. Relaţia sugerează un ordin de convergenţă mai mare decât liniar dar sub cel pătratic

$$|e_{k+1}| = A\,|e_k|^p\,. \tag{3.41}$$

Avem $|e_k| = A\,|e_{k-1}|^p$ de unde $|e_{k-1}| = \left(A^{-1}\,|e_k|\right)^{1/p}$. Înlocuind în (3.40) vom obţine

$$2A^{1+1/p}\,|F'(\alpha)|\,/\,|F''(\alpha)| \cong |e_k|^{1-p+1/p}\,. \tag{3.42}$$

Membrul stâng este în general finit pe când în membrul drept $e_k \to 0$. Singura posibilitate ca relaţia să rămână valabilă pe măsură ce k creşte este ca

$$1 - p + 1/p = 0 \tag{3.43}$$

ceea conduce la $p = (1 + \sqrt{5})/2 \cong 1.62$, ordinul de convergenţă al metodei secantei. Valoarea obţinută ar indica o convergenţă mai lentă decât metoda Newton-Raphson. Însă, în fiecare iteraţie metoda secantei efectuează o singură evaluare a funcţiei F. Pentru *două* iteraţii ale metodei secantei (pentru a avea acelaşi efort de calcul ca şi metoda Newton-Raphson), ordinul de convergenţă ar fi $p^2 \cong 2.62$, metoda secantei fiind mai rapidă decât metoda Newton-Raphson.

Figura 3.3: (a) Metoda secantei restrânge intervalul pe care se află rădăcina la (x_2, x_1), mai mic decât intervalul (c, b) furnizat de metoda înjumătăţirii. (b) Metoda parabolelor tangente restrânge rapid, din ambele capete, intervalul pe care se află rădăcina de la (a_0, b_0) la (a_1, b_1).

3.2.4 Metoda parabolelor tangente

În cadrul metodei Newton-Raphson procesul iterativ se bazează pe aproximarea liniară locală a funcţiei $F(x)$. Imediat este sugerată ideea construirii unei metode mai rapid convergente prin folosirea unei aproximări locale mai exacte, de exemplu prin parabole. O parabolă ce aproximează local funcţia $F(x)$ s-ar putea obţine din trunchierea dezvoltării în serie Taylor

$$F(x) \cong F(x_k) + (x - x_k)F'(x_k) + (x - x_k)^2 F''(\xi_k)/2\,. \tag{3.44}$$

Apariţia derivatei de ordinul doi este dezavantajoasă însă, datorită pierderii de precizie.

Mai convenabil este să se construiască o parabolă ce foloseşte informaţii din două puncte ale funcţiei. Dacă acestea se aleg ca fiind capetele intervalului $[a, b]$ pe care ştim că se află o rădăcină se obţine o metodă foarte rapid convergentă[1]. Vom construi două parabole care trec prin $(a, F(a))$, $(b, F(b))$, una fiind tangentă la $F(x)$ în $x = a$, cealaltă fiind tangentă la $F(x)$ în $x = b$ (fig. 3.3). Se consideră că $F(x)$ are derivate continui până la ordinul 3 inclusiv ($F(x) \in C^3[a, b]$). Parabolele se exprimă ca polinoamele de gradul doi $p_i(x)$, $i \in \{1, 2\}$, de forma

$$p_i(x) = A_i x^2 + B_i x + C_i\,. \tag{3.45}$$

Coeficienţii A_i, B_i,C_i, $i \in \{1, 2\}$, sunt determinaţi din condiţiile

$$p_1(a) = F(a), \quad p_1(b) = F(b), \quad p_1'(a) = F'(a)\,. \tag{3.46}$$

$$p_2(a) = F(a), \quad p_2(b) = F(b), \quad p_2'(b) = F'(b)\,. \tag{3.47}$$

[1] Metoda a fost elaborată de prof. M. Blumenfeld, 1977.

Notând cu δ_i abaterea polinomului $p_i(x)$ faţă de funcţia $F(x)$, $\delta_i(x) = F(x) - p_i(x)$, se obţin relaţiile

$$\delta_1(x) = (x-a)^2\delta_1''(\xi_1)/2, \quad \delta_2(x) = (x-b)^2\delta_2''(\xi_2)/2, \tag{3.48}$$

cu $\xi_1 \in (a,x)$, $\xi_2 \in (x,b)$. Dacă derivata $\delta_i'''(x) = F'''(x)$ este diferită de zero pe (a,b), rezultă că derivatele $\delta_i''(x)$ sunt monotone pe (a,b) şi se pot anula în cel mult un punct. Să considerăm, spre exemplu, abaterea $\delta_1(x)$. Deoarece $\delta_1(b) = 0$, rezultă că $\delta_1''(\xi_{1b}) = 0$, adică $\delta_1'' = 0$ pentru valoarea ξ_{1b} care corespunde lui $x = b$. În mod similar se obţine o singură valoare ξ_{2a} care anuleaza pe δ_2''. Ţinând cont de (3.48), se deduc inegalităţile

$$F'''(x)\delta_1(x) < 0, \quad F'''(x)\delta_2(x) > 0, \tag{3.49}$$

pentru $F'''(x) \neq 0$ de unde rezultă că pentru $x \in (a,b)$

$$\delta_1(x)\delta_2(x) < 0\,. \tag{3.50}$$

În consecinţă, dacă derivata $F'''(x)$ nu se anulează pe (a,b), $F(x)$ se găseşte *între* cele două parabole tangente definite de (3.46,3.47).

Soluţia ecuaţiei $p_1(x) = 0$, care se găseşte în intervalul (a,b) este

$$x_0 = a + 2l(b-a)/\left[\sqrt{k^2 + 4l(k+1)} - k\right], \quad k^2 + 4l(k+1) \geq 0, \tag{3.51}$$

$$k \equiv (b-a)F'(a)/[F(a) - F(b)], \quad l = F(a)/[F(a) - F(b)]\,. \tag{3.52}$$

La aplicarea algoritmului, se utilizează o singură formulă, schimbând capătul de tangenţă, pentru a obţine un şir care converge spre rădăcină cu valori din ambele parţi (în măsura în care $F'''(x)$ păstreaza un semn constant).

EXEMPLUL 1. Să se găsească rădăcinile reale ale ecuaţiei

$$F(x) \equiv x - \ln|x| - 1.2 = 0 \tag{3.53}$$

situate în intervalul $(-1,2)$.

Rezolvare. Pentru etapa iniţială de *căutare* folosim şirul lui Rolle sintetizat în următorul tabel

x	-1		0		1		2
$F(x)$	-2.2	$\nearrow$	$+\infty \mid +\infty$	$\searrow$	-0.2	$\nearrow$	0.107
$F'(x)$	2	$+$	$+\infty \mid -\infty$	-	0	$+$	0.5
$F''(x)$	$+$	$+$	$+\infty \mid +\infty$		$+$		$+$

Se deduce prezenţa a trei rădăcini, câte una în fiecare din intervalele $(-1,0)$, $(0,1)$, $(1,2)$. E preferabil să se evite singularitatea din $x = 0$, luând primele două intervale ca $(-1,-\varepsilon)$, $(\varepsilon,1)$ cu ε apropriat de zero şi care păstrează semnul funcţiei F. Cum $F(-0.1) = 1.00259 > 0$ şi $F(-0.1) = 1.20259 > 0$, vom considera intervalele $(-1,-0.1)$, $(0.1,1)$ şi $(1,2)$. Am încheiat etapa de căutare a rădăcinilor. Trecem la restrângerea intervalelor sau localizarea acestora. Pentru această etapă utilizăm înjumătăţirea intervalelor. Nu se recomandă efectuarea unui mare număr de iteraţii. Patru iteraţii ale metodei conduc la intervalele precizate în tabelul următor pentru prima rădăcină

k	a_k	c_k	b_k	$f(a_k)$	$f(c_k)$	$f(b_k)$
0	-1.000000	-0.550000	-0.100000	-2.200000	-1.152163	1.002585
1	-0.550000	-0.325000	-0.100000	-1.152163	-0.401070	1.002585
2	-0.325000	-0.212500	-0.100000	-0.401070	0.136313	1.002585
3	-0.325000	-0.268750	-0.212500	-0.401070	-0.154776	0.136313
4	-0.268750	-0.240625	-0.212500	-0.154776	-0.016109	0.136313

Un calcul asemănător furnizează intervalele $(0.4375, 0.49375)$ şi $(1.75, 1.78125)$ pentru celelalte două rădăcini. Valorile funcţiei la mijloacele acestor două intervale sunt $f(0.465625) = 0.0299$ şi $f(1.765625) = -0.00288$, suficient de mici. S-a încheiat etapa de localizare a rădăcinilor.

Pentru rafinarea rădăcinilor vom compara mai multe metode. Iteraţia simplă conduce la relaţia

$$x_{k+1} = f(x_k) = \ln|x_k| + 1.2 \tag{3.54}$$

cu $m = |f'(x)| = 1/|x| < 1$ doar pentru intervalul $(1.75, 1.78125)$. Pentru celelalte două intervale se poate folosi modificarea iteraţiei simple (3.19) pentru care β trebuie să satisfacă (3.22), ce în cazul de faţă devine

$$|1 - \beta(1 - 1/x)| < 1\,. \tag{3.55}$$

Alegem $\beta = 0.1$ pentru intervalul $(-0.26875, -0.2125)$ şi $\beta = 0.5$ pentru intervalul $(0.4375, 0.49375)$. Aceste alegeri nu sunt unicele posibile. Vom compara aceste trei variante ale iteraţiei simple cu metoda Newton-Raphson şi a secantei. Ne propunem determinarea rădăcinilor cu şase cifre semnificative exacte şi astfel încât valoarea funcţiei să fie mai mică de 10^{-7}. Primele câteva iteraţii sunt redate în tabelul următor.

k	$\beta = 0.1$	secantă	Newton	$\beta = 0.5$	secantă	Newton
0	-0.240625	-0.240625	-0.240625	0.465625	0.465625	0.465625
1	-0.239014	-0.268750	-0.237501	0.480625	0.437500	0.491765
2	-0.238236	-0.238841	-0.237517	0.487271	0.493294	0.493235
3	-0.237862	-0.237449	–	0.490374	0.493245	0.493239
4	-0.237682	-0.237517	–	0.491855	0.493239	–
5	-0.237596	–	–	0.492568	–	–
6	-0.237555	–	–	0.492913	–	–

$\beta = 1$	secantă	Newton
1.765625	1.765625	1.765625
1.768505	1.772176	1.772266
1.770134	1.772250	1.772250
1.771055	1.772250	–
1.771869	–	–
1.772228	–	–
1.772248	–	–

Iteraţiile au fost oprite în momentul atingerii condiţiilor impuse. Metodele de iteraţie simplă au necesitat 18, 18 şi 20 de iteraţii până la atingerea acestor condiţii.

Se observă convergenţa mult mai rapidă a metodelor Newton şi secantă faţă de iteraţia simplă. Metoda Newton converge în general mai repede, însă metoda secantei realizează numărul minim de evaluări de funcţie până la atingerea preciziei impuse.

Exemplu 2. Să se determine rădăcina ecuaţiei

$$F(x) = e^x - \cos x = 0$$

cuprinsă în intervalul (-2,-1) cu 7 cifre semnificative exacte.

Rezolvare. Vom compara de data aceasta metoda Newton-Raphson cu metoda parabolelor tangente. Valoarea de start furnizată de metoda parabolelor tangente (3.51) este $x_0 = -0.17271412$. Din acest punct de start se obţine rădăcina evidentă din $x = 0$. Efectuăm totuşi calculul pentru a compara cele două metode. Iteraţiile succesive sunt

k	x_k Newton	x_k Parabole tangente
1	0.0419823	0.0376656
2	0.0016474	-0.0001885
3	0.0000027	0.0000000

Se observă convergenţa mai rapida către soluţie a metodei parabolelor tangente pe acest exemplu.

3.3 Determinarea rădăcinilor polinoamelor

Metodele prezentate anterior permit determinarea rădăcinilor unei funcţii $F(x)$ oarecare. Dacă funcţia $F(x)$ are o formă particulară se pot elabora algoritmi mai performanţi ce exploatează acest fapt. Spre exemplu, metodele anterioare necesită o aproximare iniţială a unei rădăcini şi nu pot garanta obţinerea tuturor rădăcinilor funcţiei $F(x)$. Dacă $F(x)$ este o funcţie polinomială cu coeficienţi reali se pot elabora metode care permit determinarea *tuturor* rădăcinilor fără a fi nevoie de furnizarea unor aproximaţii iniţiale.

3.3.1 Metoda Lobacevschi-Graeffe

Se consideră polinomul $P_n(x)$ de gradul n, cu coeficienţi reali

$$P_n(x) = \sum_{j=0}^{n} a_{n-j} x^j = a_0 \prod_{j=1}^{n} (x - x_j), \quad a_0 \neq 0\,, \tag{3.56}$$

având rădăcinile x_j, $j = 1, 2, \ldots, n$, reale sau complex conjugate (coeficienţii a_j fiind numere reale). Reamintim prima relaţie Vieta care furnizează o legătură dintre coeficienţii unui polinom şi rădăcinile acestuia

$$x_1 + x_2 + \ldots + x_n = -a_1/a_0\,. \tag{3.57}$$

Dacă una din rădăcini, să zicem x_1, ar fi mult mai mare în modul decât toate celelalte, $|x_1| \gg |x_j|$, $j = \overline{2, n}$, relaţia (3.57) ar conduce la

$$x_1 \cong -a_1/a_0\,, \tag{3.58}$$

iar rădăcina x_1 se zice *preponderentă*. În general, nu vom putea presupune că un polinom oarecare are o rădăcină preponderentă. Dacă însă una dintre rădăcini este mai mare în modul decât toate celelalte $|x_1| > |x_j|$, $j = \overline{2,n}$, prin ridicarea la o putere mare m ea va deveni preponderentă $|x_1^m| \gg \left|x_j^m\right|$, $j = \overline{2,n}$. Procedeul Lobacevschi-Graeffe furnizează o metodă de ridicare succesivă la pătrat a rădăcinilor ce permite separarea rădăcinilor preponderente.

Se observă imediat că, schimbând x în $-x$, se obţine

$$P_n(-x) = (-1)^n a_0 \prod_{j=1}^{n}(x + x_j)\,. \tag{3.59}$$

Făcând apoi produsul

$$(-1)^n P_n(x)P_n(-x) = a_0^2 \prod_{j=1}^{n}(x^2 - x_j^2)\,, \tag{3.60}$$

se obţine un polinom de gradul n, în variabila x^2. Repetând procedeul pentru polinomul în variabila x^2 se obţine un polinom de gradul n în variabila x^4 şi aşa mai departe. Într-o etapa s vom nota

$$z = x^m, \quad m = 2^s, \quad s \in \mathbf{N}, \tag{3.61}$$

iar polinomul în variabila z se va scrie

$$Q_n(z) = a_0^m \prod_{j=1}^{n}(z - x_j^m) = \sum_{j=0}^{n}(-1)^j A_{n-j}^{(s)} z^j, \quad m = 2^0, 2^1, \ldots, 2^s\,. \tag{3.62}$$

Evident, coeficienţii $A_j(s)$ nu se obţin cu ajutorul rădăcinilor x_j deoarece acestea sunt necunoscutele problemei, ci prin înmultirea directă a polinoamelor $P_n(z)$ şi $P_n(-z)$. Notând cu $A_j^{(s+1)}$ valorile acestor coeficienţi în etapa $s+1$, se obţine următoarea relaţie de recurenţă (prin identificarea coeficienţilor):

$$A_j^{(s+1)} = \left[A_j^{(s)}\right]^2 + 2\sum_{k=0}^{n}(-1)^k A_{j-k}^{(s)} A_{j+k}^{(s)}, \quad j = \overline{0,n}\,, \tag{3.63}$$

unde s-a făcut convenţia că termenii de indice $j + k > n$ sau de indice $j - i < 0$ sunt consideraţi nuli.

Găsirea rădăcinilor în metoda Lobacevschi-Graeffe se bazează pe interpretarea semnului şi valorilor coeficienţilor $A_j^{(s)}$ în diverse etape s. Pentru a arăta cum se procedează, vom considera trei cazuri, discutate mai jos.

Cazul rădăcinilor reale şi distincte.

Să presupunem că toate rădăcinile x_j, $j = \overline{1,n}$, ale polinomului $P_n(x)$ sunt distincte. Rădăcinile vor fi ordonate după indici astfel, în ordinea descrescătoare a modulelor,

$$|x_1| > |x_2| > \ldots > |x_n|\,. \tag{3.64}$$

Relaţiile Vieta, dintre rădăcini şi coeficienţi, pentru polinomul $Q_n(z)$ din (3.62) sunt

$$x_1^m + x_2^m + \ldots + x_n^m = A_1^{(s)}/A_0^{(s)} \tag{3.65}$$

$$x_1^m x_2^m + x_1^m x_3^m + \ldots + x_{n-1}^m x_n^m = A_2^{(s)}/A_0^{(s)} \tag{3.66}$$

$$x_1^m x_2^m x_3^m + x_1^m x_2^m x_4^m + \ldots + x_{n-2}^m x_{n-1}^m x_n^m = A_3^{(s)}/A_0^{(s)} \tag{3.67}$$

$$\ldots$$

$$x_1^m x_2^m \cdot \ldots \cdot x_n^m = A_n^{(s)}/A_0^{(s)} \tag{3.68}$$

Tinând cont de inegalităţile (3.64) rezultă că, la un anumit m, termenul x_1^m va deveni preponderent în relaţia (3.65); la fel se va întîmpla cu termenul $x_1^m x_2^m$ în relaţia (3.66) şi aşa mai departe. În consecinţă, pentru m suficient de mare, sistemul (3.65)-(3.68) se aproximează prin cantităţi care reprezintă pătrate perfecte

$$x_1^m \cong A_1^{(s)}/A_0^{(s)} \tag{3.69}$$

$$x_1^m x_2^m \cong A_2^{(s)}/A_0^{(s)} \tag{3.70}$$

$$\ldots$$

$$x_1^m x_2^m \cdot \ldots \cdot x_n^m \cong A_n^{(s)}/A_0^{(s)}\,, \tag{3.71}$$

de unde se obţin imediat modulele rădăcinilor

$$x_j = \pm \left(A_j^{(s)}/A_{j-1}^{(s)} \right)^{1/m}, \quad j = \overline{1,n}\,. \tag{3.72}$$

Determinarea semnului se face prin introducerea în ecuaţia iniţiala $P_n(x) = 0$. O problemă care se pune este aceea de a şti când termenii reţinuţi în sistemul (3.69-3.71) au devenit într-adevăr preponderenţi, astfel încât să se producă separarea rădăcinilor. Pentru aceasta se calculează rapoartele

$$r_j^{(s)} = \left[A_j^{(s-1)} \right]^2 / A_j^{(s)}, \quad j = \overline{1,n-1}\,, \tag{3.73}$$

în fiecare etapă s. Cu $m = 2^s$, avem $m/2 = 2^{s-1}$. Dacă separarea s-ar fi produs în iteraţia $s-1$ atunci ar trebui ca

$$x_1^{m/2} \cong A_1^{(s-1)}/A_0^{(s-1)}\,.$$

Cu atât mai mult în iteraţia următoare

$$x_1^m \cong A_1^{(s)}/A_0^{(s)}$$

Ca atare, raportul $r_1^{(s)}$ ar trebui să fie aproape de unu, $r_{n-1}^{(s)} \cong 1$. Raţionamentul se repetă pentru ceilalţi coeficienţi astfel că ajungem la următorul criteriu de separare: dacă *toate* rapoartele $r_j^{(s)}$, $j = \overline{1,n-1}$ tind către unu atunci s-a produs separarea rădăcinilor polinomului ce au fost presupuse *distincte*.

Cazul rădăcinilor reale, multiple în modul.

Putem trata şi cazul în care polinomul $P_n(x)$ are mai multe rădăcini reale, de acelaşi modul. Să presupunem

$$|x_1| = |x_2| = \ldots = |x_M| > |x_{M+1}| > |x_{M+1}| > \ldots > |x_n|\,, \tag{3.74}$$

adică cea mai mare rădăcină în modul apare de M ori. Atunci partea preponderentă a primei relaţii Vieta (3.65) este

$$x_1^m + x_2^m + \ldots + x_M^m = M\, x_1^m \cong A_1^{(s)}/A_0^{(s)}, \tag{3.75}$$

iar cea a relaţiei Vieta M este

$$x_1^m x_2^m \ldots x_M^m = x_1^{mM} \cong A_M^{(s)}/A_0^{(s)}\,. \tag{3.76}$$

Să deducem criteriul de separare. Dacă separarea s-ar fi produs în iteraţia $s-1$ atunci ar trebui ca

$$x_1^{m/2} \cong A_1^{(s-1)}/A_0^{(s-1)}/M\,,$$

iar în iteraţia următoare

$$x_1^m \cong A_1^{(s)}/A_0^{(s)}/M\,.$$

Ca atare, raportul $r_1^{(s)}$ ar trebui să fie aproape de M, $r_1^{(s)} \cong M$. Deci, când un raport r_j tinde către un număr întreg M, atunci s-a produs separarea unei rădăcini M-multiple în modul. Calculul rădăcinii se efectuează folosind relaţia Vieta M

$$x_1 = \pm\left[A_M^{(s)}/A_0^{(s)}\right]^{1/mM} \tag{3.77}$$

deoarece se obţine o aproximaţie mai bună datorită exponentului mai mare. Se observă că pentru calculul unei rădăcini M-multiple în modul se folosesc coeficienţi cu diferenţa între indici egală cu M.

Dacă rădăcina multiplă în modul nu este şi cea mai mare în modul,

$$|x_1| > \ldots > |x_k| = \ldots = |x_M| > \ldots > |x_n|\,, \tag{3.78}$$

atunci formulele de mai sus se aplică cu o deplasare cu k a indicilor. Anume raportul $r_k^{(s)}$ va tinde către M la separarea rădăcinii x_k^m, iar calculul acesteia se efectuează conform formulei

$$x_k = \pm\left[A_{k+M}^{(s)}/A_k^{(s)}\right]^{1/mM}\,. \tag{3.79}$$

Cazul rădăcinilor complexe.

Să presupunem acum că x_1, x_2 sunt rădăcini complex conjugate

$$|x_1| = |x_2| > \ldots > |x_n| \ . \tag{3.80}$$

Vom scrie

$$x_1 = \rho e^{i\theta}, \quad x_2 = \rho e^{-i\theta} \tag{3.81}$$

cu $\rho = |x_1| = |x_2|$ modulul rădăcinilor şi θ argumentul acestora. Prima relaţie Vieta (3.65) devine

$$2\rho^m \cos m\theta + x_3^m + \ldots + x_n^m \cong A_1^{(s)}/A_0^{(s)} \ . \tag{3.82}$$

Termenul $2\rho^m \cos m\theta$ nu poate fi considerat preponderent datorită factorului oscilant $\cos m\theta$. A doua relaţie Vieta (3.66) are partea preponderentă

$$\rho^{2m} \cong A_2^{(s)}/A_0^{(s)} \ . \tag{3.83}$$

Modulul ρ se determină analog determinării unei rădăcini reale, 2-multiplă în modul

$$\rho = \left[A_2^{(s)}/A_0^{(s)}\right]^{\frac{1}{2m}} \ . \tag{3.84}$$

Mai rămâne de determinat criteriul de separare. Dacă separarea s-ar fi produs în iteraţia $s-1$ atunci ar trebui ca

$$\rho^{m/2} \cong A_2^{(s-1)}/A_0^{(s-1)} \ ,$$

iar în iteraţia următoare

$$\rho^m \cong A_2^{(s)}/A_0^{(s)}.$$

Ca atare, raportul $r_2^{(s)}$ ar trebui să fie aproape de unu, $r_2^{(s)} \cong 1$. Pe de altă parte, raportul

$$r_1^{(s)} = \left[A_1^{(s-1)}\right]^2 /A_1^{(s)} \tag{3.85}$$

este oscilant datorită schimbării semnului termenului $2\rho^m \cos m\theta$. Aşadar, vom recunoaşte prezenţa rădăcinilor complex conjugate prin apariţia unor rapoarte oscilante. Separarea modulului unei perechi de rădăcini complex conjugate se produce atunci când raportul imediat vecin celui oscilant tinde către unu.

Regulile de determinare a rădăcinilor.

Sintetizând rezultatele discuţiilor de mai sus vom stabili următoarele concluzii.

1. Modulele rădăcinilor, reale sau complexe, sunt date de rapoartele coeficientilor $A_j^{(s)}$, $j = \overline{1, n-1}$ care reprezintă pătrate perfecte, în ordinea indicilor. Coeficienţii $A_0^{(s)}$ şi $A_n^{(s)}$ sunt întodeauna pătrate perfecte.

2. Verificarea că un anumit $A_j^{(s)}$ este un pătrat perfect se stabileşte prin raportul $r_j^{(s)}$ dintre valorile coeficientului $A_j^{(s)}$ în etape succesive, care tinde către unitate,

$$r_j^{(s)} = \left[A_1^{(s-1)}\right]^2 / A_1^{(s)} \to 1 \tag{3.86}$$

când pătratul perfect se realizează.

3. Dacă rapoartele $r_j^{(s)}$ care tind către unitate au indici consecutivi, atunci se obţin rădăcini simple.

4. Dacă $A_j^{(s)}$ şi $A_{j+M}^{(s)}$ sunt pătrate perfecte consecutive, atunci modulul unor rădăcini are ordinul de multiplicitate M. Dacă între $r_j^{(s)}$ şi $r_{j+M}^{(s)}$ există M' rapoarte oscilante, atunci vor apare M' perechi de rădăcini complex conjugate. Dacă între $r_j^{(s)}$ şi $r_{j+M}^{(s)}$ nu există rapoarte oscilante, ci $r_j^{(s)} \to M$, atunci se va produce o rădăcina reală cu ordinul de multiplicitate M.

5. Metoda prezentată furnizează doar modulele rădăcinilor. În cazul rădăcinilor reale, semnul se stabileşte prin înlocuirea în ecuaţia originală, $P_n(x) = 0$. În cazul rădăcinilor complexe, modulul odată determinat, se determină partea reală şi imaginară folosind relaţiile între rădăcini şi coeficienţi pentru ecuaţia $P_n(x) = 0$. Nu se recomandă determinarea argumentului din relaţii de forma (3.82) deoarece numărul m este mare şi se obţin multe soluţii ale ecuaţiilor de forma $\cos m\theta = \lambda$, $\lambda \in \mathbf{R}$, care trebuie verificate apoi în ecuaţia iniţială.

6. Principalul avantaj al metodei Graeffe-Lobacevski este furnizarea tuturor rădăcinilor, reale şi complexe, ale unui polinom cu coeficienţi reali. Printre dezavantaje se numară: posibilitatea depăşirii superioare de registru, necesitatea examinării de către utilizator a tabloului coeficienţilor precum şi necesitatea rezolvării unor sisteme de ecuaţii neliniare în cazul existenţei rădăcinilor complexe.

7. Datorită lucrului cu numere mari, aproximaţiile rădăcinilor furnizate de metoda Lobacevschi-Graeffe sunt destul de imprecise. Ca atare metoda va fi folosită pentru *localizarea tuturor rădăcinilor*. După localizare, se va trece la *rafinarea* acestora prin procedeele de factorizare a polinoamelor prezentate mai jos.

3.3.2 Metode de factorizare a polinoamelor

Se cunoaşte faptul că, dacă un polinom are ca rădăcină simplă pe $x = \alpha$, atunci el este divizibil cu $x-\alpha$. Prin procedee de localizare a rădăcinilor precum metoda Lobacevschi-Graeffe, sau teoreme asupra vecinătăţii în care se află rădăcina unui polinom, se poate obţine o aproximaţie iniţială a rădăcinii α_0. Procedeele

uzuale de rafinare, precum metoda Newton-Raphson, pot fi apoi folosite pentru creşterea preciziei aproximaţiei iniţiale. În aplicarea metodei Newton-Raphson se foloseşte schema lui Horner pentru a evalua valorile unui polinom cu un număr minim de operaţii aritmetice.

Schema lui Horner.

Împărţind polinomul $P_n(x)$ prin monomul $x-\alpha$ se obţine

$$P_n(x)=\sum_{j=0}^{n} a_{n-j}\,x^j=(x-\alpha)\sum_{j=0}^{n-1} b_{n-j-1}\,x^j+b_n\,, \tag{3.87}$$

unde b_j, $j=\overline{0,n}$ sunt coeficienţi care se determină prin identificarea termenilor având aceeaşi putere a lui x,

$$b_0=a_0,\quad b_k=a_k+\alpha b_{k-1},\quad k=\overline{1,n}\,. \tag{3.88}$$

Din (3.87) se observă că valoarea polinomului în $x=\alpha$ este chiar b_n. Relaţia de recurenţă (3.88) defineşte *schema lui Horner*, ce permite determinarea valorii $P_n(\alpha)$ cu doar n înmulţiri şi $n-1$ adunări.

Schema lui Horner poate fi adaptată şi pentru evaluarea derivatelor unui polinom. Deoarece valoarea polinomului în $x=\alpha$ este b_n putem considera $P_n(\alpha)=b_n(\alpha)$ şi deriva relaţiile de recurenţă după (3.88) α

$$\frac{db_k}{d\alpha}=b_{k-1}+\alpha\frac{db_{k-1}}{d\alpha},\quad k=\overline{1,n}\,. \tag{3.89}$$

S-au obţinut relaţii de recurenţă pentru noii coeficienţi

$$c_{k-1}\equiv\frac{db_k}{d\alpha},\quad k=\overline{1,n}\,, \tag{3.90}$$

ce satisfac relaţiile de recurenţă

$$c_0=b_0,\quad c_k=b_k+\alpha c_{k-1},\quad k=\overline{1,n-1}. \tag{3.91}$$

Valoarea derivatei polinomului în $x=\alpha$ este $P_n(\alpha)=c_{n-1}$.

Determinarea rădăcinilor simple.

Dată fiind aproximaţia iniţială a unei rădăcini simple α_0, rafinarea iterativă prin procedeul Newton-Raphson se exprimă ca

$$\alpha_{m+1}=\alpha_m-b_n(\alpha_m)/c_{n-1}(\alpha_m) \tag{3.92}$$

cu b_n, c_{n-1} calculaţi folosind recurenţele (3.88, 3.91), respectiv. Algoritmul complet este

$$\begin{array}{l}\alpha \leftarrow \alpha_0; \quad m \leftarrow 0; \quad M \leftarrow 1; \texttt{EvalPolinom}(\alpha, b, c);\\ \text{repetă}\\ \left[\begin{array}{l}\text{dacă } |c| < \delta_1 \text{ atunci Determină multiplicitatea } M\\ \alpha_{nou} \leftarrow \alpha - Mb/c; \texttt{EvalPolinom}(\alpha, b, c)\\ \mathit{AtinsPrecizie} \leftarrow |\alpha_{nou} - \alpha| \le \varepsilon\,(1 + |\alpha_{nou}|) \text{ sau } |b| < \delta\\ \alpha \leftarrow \alpha_{nou}; \quad m \leftarrow m + 1\end{array}\right.\\ \text{până când } \mathit{AtinsPrecizie} \text{ sau } m > m_{\max}\end{array} \tag{3.93}$$

unde ε este eroarea relativă maxim admisă pentru rădăcină, δ este valoarea maxim admisă pentru valoare polinomului, iar M este multiplicitatea rădăcinii căutate. Iniţial se presupune $M = 1$. Valoarea δ_1 este folosită şi pentru a stabili un prag pentru valoarea derivatei. Valori ale derivatei sub acest prag indică probabilitatea unor rădăcini multiple. Procedura de evaluare a polinomului este

$$\begin{array}{l}\texttt{EvalPolinom}(\alpha, b, c)\\ \left[\begin{array}{l}b \leftarrow a_0; \quad c \leftarrow b\\ k = 1 : (n-1)\\ \quad [b \leftarrow a_k + \alpha b, \quad c \leftarrow b + \alpha c\\ b \leftarrow a_n + \alpha b\end{array}\right.\end{array} \tag{3.94}$$

Metoda poate fi aplicată direct pentru calculul rădăcinilor complexe, deoarece polinomul este o funcţie analitică în tot planul **C**.

Determinarea rădăcinilor multiple.

Dacă α este o rădăcină multiplă, derivata polinomului în α va fi nulă, $P_n'(\alpha) = 0$. Numerele b_n, c_{n-1} vor tinde către zero simultan, iar convergenţa procedeului Newton-Raphson se înrăutăţeşte. Se poate efectua o modificare simplă a procedeului ce îmbunătăţeşte convergenţa. Fie M ordinul de multiplicitate a rădăcinii. Aplicăm procedeul Newton-Raphson funcţiei $f(x) = [P_n(x)]^{1/m}$ ce are derivata $f'(x) = [P_n(x)]^{1/m-1}/m$. Iteraţia Newton-Raphson se exprimă ca

$$\alpha_{m+1} = \alpha_m - Mb_n(\alpha_m)/c_{n-1}(\alpha_m)\,, \tag{3.95}$$

forma folosită în algoritmul de mai sus. Mai rămâne de stabilit o modalitate de determinare a multiplicităţii M. Se ştie că dacă rădăcina α are ordinul de multiplicitate M, avem $P^{(k)}(\alpha) = 0$ pentru $k = 1, 2, \ldots, M-1$ şi $P^{(M)}(\alpha) \neq 0$. Ca atare, la detectarea posibilităţii unei rădăcini multiple (prin faptul că prima derivată a devenit mică) vom evalua toate derivatele polinomului. Prima dintre derivatele polinomului ce este nenulă va furniza multiplicitatea rădăcinii.

Pentru evaluarea derivatelor se foloseşte *schema lui Horner completă*. Determinarea tuturor derivatelor unui polinom în $x = \alpha$ este echivalentă cu dezvoltarea în serie Taylor în jurul lui $x = \alpha$

$$P_n(x) = C_0 + C_1(x-\alpha) + C_2(x-\alpha)^2 + \ldots + C_n(x-\alpha)^n,\ C_k = P_n^{(k)}(\alpha)/k!\,. \tag{3.96}$$

Prin aplicarea schemei Horner (3.88) se observă (vezi 3.87) că se obţin coeficienţii polinomului $P_{n-1}(x)$ din factorizarea

$$P_n(x) = (x-\alpha)P_{n-1}(x) + C_0\,, \tag{3.97}$$

cu $C_0 = P_n(\alpha)$. Coeficienţii polinomului $P_{n-1}(x)$ sunt $b_0, b_1, \ldots, b_{n-1}$ din 3.87. Dacă se mai aplică o dată schema lui Horner asupra polinomului $P_{n-1}(x)$ vom obţine coeficienţii lui $P_{n-2}(x)$ din factorizarea

$$P_{n-1}(x) = (x-\alpha)P_{n-2}(x) + C_1\,, \tag{3.98}$$

şi aşa mai departe. Aplicarea de n ori a schemei Horner va furniza toţi coeficienţii C_k, $k = \overline{0, n-1}$, deci implicit toate derivatele $P_n^{(k)}(\alpha) = k!\,C_k$. Algoritmul este

$$\begin{array}{l} k = 0 : n \\ \quad [C_k = a_{n-k} \\ k = 0 : (n-1) \\ \quad \left[\begin{array}{l} j = (n-1) : k : -1 \\ \quad [C_j \leftarrow C_j + \alpha\, C_{j+1} \end{array} \right. \end{array} \tag{3.99}$$

Procedura de determinare a multiplicităţii unei rădăcini, ce a fost menţionată în algoritmul (3.93), va determina primul indice k pentru care $|k!\,C_k| > \delta$, iar acesta va fi luat ca multiplicitatea rădăcinii M.

Determinarea rădăcinilor complex conjugate (Metoda Bairstow).

În cazul când polinomul $P_n(x)$ are coeficienţii reali, rădăcinile complexe (dacă există) sunt conjugate, iar calculul cu numere complexe poate fi evitat. În acest scop, se utilizează o factorizare cu un polinom de gradul doi de forma x^2+px+q

$$P_n(x) = \sum_{j=0}^{n} a_{n-j}x^j = (x^2+px+q)\sum_{j=0}^{n-2} b_{n-j-2}x^j + Rx + S\,. \tag{3.100}$$

Identificarea termenilor conduce la relaţiile

$$b_k = a_k - pb_{k-1} - qb_{k-2}, \quad k = \overline{0, n-2}\,, \tag{3.101}$$

$$R = a_{n-1} - pb_{n-2} - qb_{n-3}, \quad S = a_n - qb_{n-2}\,. \tag{3.102}$$

unde s-a convenit $b_{-2} = b_{-1} = 0$. Dacă extindem relaţia de recurenţă pentru $k = \overline{0, n}$, vom obţine $R = b_{n-1}$, $S = b_n + pb_{n-1}$. Pentru ca factorul x^2+px+q să dividă pe $P_n(x)$ exact trebuie ca restul să fie nul $R(p,q) = 0$, $S(p,q) = 0$, ceea ce este echivalent cu

$$b_n(p,q) = 0, \quad b_{n-1}(p,q) = 0\,. \tag{3.103}$$

Relaţiile (3.103) formează un sistem de două ecuaţii cu două necunoscute. Ecuaţiile sunt nelineare şi pentru rezolvarea sistemului se aplică tot o metodă

de tip Newton-Raphson[2]. Vom aplica principiul liniarizării, păstrând numai termenii liniari din dezvoltările în serie Taylor a funcţiilor $b_n(p,q)$, $b_{n-1}(p,q)$ în jurul unei aproximaţii curente (p_m, q_m)

$$b_n(p,q) \cong b_n(p_m,q_m) + (p-p_m)\left[\frac{\partial b_n}{\partial p}\right]_m + (q-q_m)\left[\frac{\partial b_n}{\partial q}\right]_m = 0\,, \quad (3.104)$$

$$b_{n-1}(p,q) \cong b_{n-1}(p_m,q_m) + (p-p_m)\left[\frac{\partial b_{n-1}}{\partial p}\right]_m + (q-q_m)\left[\frac{\partial b_{n-1}}{\partial q}\right]_m = 0\,. \quad (3.105)$$

Următoarea aproximaţie (p_{m+1}, q_{m+1}), va fi dată de soluţia sistemului liniarizat de mai sus. Pentru obţinerea derivatelor parţiale diferenţiem relaţia de recurenţă (3.101)

$$\frac{\partial b_k}{\partial p} = -b_{k-1} - p\frac{\partial b_{k-1}}{\partial p} - q\frac{\partial b_{k-2}}{\partial p}, \quad \frac{\partial b_k}{\partial q} = -b_{k-2} - p\frac{\partial b_{k-1}}{\partial q} - q\frac{\partial b_{k-2}}{\partial q}\,. \quad (3.106)$$

Dacă notăm $c_k \equiv \partial b_k/\partial p$, $d_k \equiv \partial b_{k+1}/\partial q$, obţinem

$$c_k = -b_{k-1} - pc_{k-1} - qc_{k-2}, \quad d_k = -b_{k-1} - pd_{k-1} - qd_{k-2} \quad (3.107)$$

pentru $k = \overline{0,n}$ şi cu $c_{-1} = c_{-2} = d_{-1} = d_{-2} = 0$. Cum cele două relaţii de recurenţă sunt identice vom păstra numai una dintre ele pentru calcule. Sistemul liniarizat devine

$$\begin{cases} (p_{m+1}-p_m)c_n + (q_{m+1}-q_m)c_{n-1} = -b_n \\ (p_{m+1}-p_m)c_{n-1} + (q_{m+1}-q_m)c_{n-2} = -b_{n-1} \end{cases}. \quad (3.108)$$

Matricea sistemului se numeşte *matrice Jacobiană*, şi este nesingulară pentru rădăcini complexe simple. Soluţia sistemului este, în aceste condiţii,

$$\delta p \equiv p_{m+1} - p_m = (b_{n-1}c_{n-1} - b_nc_{n-2})/J\,, \quad (3.109)$$

$$\delta q \equiv q_{m+1} - q_m = (b_nc_{n-1} - b_{n-1}c_n)/J\,, \quad (3.110)$$

cu $J \equiv c_nc_{n-2} - c_{n-1}^2$.

Procedeul poate fi repetat pentru polinomul

$$P_{n-2}(x) = \sum_{j=0}^{n-2} b_{n-j-2}x^j \quad (3.111)$$

până la factorizarea polinomului $P_n(x)$ în produse de polinoame de grad cel mult doi. Se obţin perechi de soluţii de forma

$$x_{1,2} = \left(-p \pm \sqrt{p^2-4q}\right)/2\,. \quad (3.112)$$

[2]Vezi capitolul 8.

Aproximaţia iniţială (p_0, q_0) se poate obţine dintr-un procedeu de localizare. Este recomandabilă folosirea coordonatelor polare

$$x_1 = \rho e^{i\theta},\ x_2 = \rho e^{-i\theta}, \quad x_1 x_2 = \rho^2 = q, \quad x_1 + x_2 = 2\rho\cos\theta = -p\,. \tag{3.113}$$

Prin folosirea metodei Lobacevschi-Graeffe pentru localizarea rădăcinilor se obţine o aproximaţie iniţială foarte bună pentru ρ. Mai trebuie doar încercate diverse valori pentru θ pentru a vedea ce valori iniţiale conduc la convergenţa procedeului. Există şi alte rezultate teoretice ce pot fi folosite pentru localizarea rădăcinilor. Spre exemplu, dacă notăm

$$R_1 = 1 + |a_0|^{-1} \max_{0<k\le n} |a_k|\,, \quad R_2 = \left[1 + |a_n|^{-1} \max_{0\le k<n} |a_k|\right]^{-1}, \tag{3.114}$$

atunci toate rădăcinile nenule ale polinomului cu coeficienţii a_k se vor afla în coroana

$$R_2 < \rho \le R_1\,. \tag{3.115}$$

Algoritmul Bairstow este

$p \leftarrow p_0; \quad q \leftarrow q_0; \quad m \leftarrow 0;$
repetă
  $c_0 \leftarrow 0;\ c_1 \leftarrow 0;\ b_0 \leftarrow 0; b_1 \leftarrow 0$
  $k = 0 : n$
    $b_2 \leftarrow b_1;\ b_1 \leftarrow b_0;\ c_2 \leftarrow c_1;\ c_1 \leftarrow c_0$
    $b_0 \leftarrow a_k - pb_1 - qb_2$
    $c_0 \leftarrow -b_1 - pc_1 - qc_2$
  $J \leftarrow c_0 c_2 - c_1^2;$
  dacă $|J| < \delta_1$ atunci mesaj(*Rădăcini multiple*); stop
  $\delta p \leftarrow (b_1 c_1 - b_0 c_2)/J; \quad \delta q \leftarrow (b_0 c_1 - b_1 c_0)/J;$
  $p_{nou} \leftarrow p + \delta p; \quad q_{nou} \leftarrow q + \delta q$
  $AtinsPrecizie \leftarrow |\delta p| + |\delta q| \le \varepsilon\,(1 + |p_{nou}| + |q_{nou}|)$ sau $|b_0| + |b_1| < \delta$
  $p \leftarrow p_{nou}; \quad q \leftarrow q_{nou}; \quad m \leftarrow m + 1$
până când *AtinsPrecizie* sau $m > m_{\max}$.

(3.116)

EXEMPLU 1. Să se determine rădăcinile polinomului

$$p_5(x) = x^5 - 4x^4 + x^3 + 3x^2 - x + 6\,.$$

Rezolvare. Vom folosi metoda Lobacevschi-Graeffe pentru a determina aproximaţiile iniţiale ale rădăcinilor. Rezultatele aplicării relaţiei (3.63) sunt redate în următorul tabel.

s	$A_0^{(s)}$	$A_1^{(s)}$	$A_2^{(s)}$	$A_3^{(s)}$	$A_4^{(s)}$	$A_5^{(s)}$
0	1.00	4.000E+00	1.000E+00	-3.000E+00	-1.000E+00	-6.000E+00
1	1.00	1.400E+01	2.300E+01	-3.700E+01	-3.500E+01	3.600E+01
2	1.00	1.500E+02	1.495E+03	3.987E+03	3.889E+03	1.296E+03
3	1.00	1.951E+04	1.047E+06	4.657E+06	4.790E+06	1.680E+06
4	1.00	3.785E+08	9.139E+11	1.172E+13	7.301E+12	2.821E+12
5	1.00	1.433E+17	8.263E+23	1.241E+26	-1.285E+25	7.959E+24
6	1.00	2.053E+34	6.828E+47	1.543E+52	-1.811E+51	6.334E+49

Calculul s-a desfăşurat cu mai multe cifre semnificative decât sunt afişate în tabel. După 6 iteraţii valorile coeficienţilor sunt $A_0^{(6)} = 1$, $A_1^{(6)} = 2.05336782 \cdot 10^{34}$, $A_2^{(6)} = 6.82754862 \cdot 10^{47}$, $A_3^{(6)} = 1.54270278 \cdot 10^{52}$, $A_4^{(6)} = -1.81053152 \cdot 10^{51}$, $A_5^{(6)} = 6.33402866 \cdot 10^{49}$. Pentru stabilirea convergenţei metodei se calculează şi rapoartele din (3.73):

s	$r_1^{(s)}$	$r_2^{(s)}$	$r_3^{(s)}$	$r_4^{(s)}$
0	–	–	–	–
1	1.14	0.04	-0.24	-0.03
2	1.31	0.35	0.34	0.31
3	1.15	2.14	3.41	3.16
4	1.01	1.20	1.85	3.14
5	1.00	1.01	1.11	-4.15
6	1.00	1.00	1.00	-0.09

Se observă obţinerea izolării a trei rădăcine reale din rapoartele $r_1^{(s)}$, $r_2^{(2)}$, $r_3^{(s)}$ ce tind toate către 1. Raportul $r_4^{(s)}$ oscilează denotând prezenţa unei perechi de rădăcini complexe, conjugate. Procedeul furnizează două valori posibile pentru rădăcina de modul maxim

$$x_1 = \pm \left(A_1^{(6)} / A_0^{(6)} \right)^{1/64} = \pm 3.4366. \tag{3.117}$$

Cum $p_5(3.4366) = -0.00269734$ şi $p_5(3.4366) = -1032.99$ alegem $x_1 = 3.4366$. Analog obţinem

$$x_2 = \pm \left(A_2^{(6)} / A_1^{(6)} \right)^{1/64} = \pm 1.62659 \Rightarrow x_2 = 1.62659 \tag{3.118}$$

$$x_3 = \pm \left(A_3^{(6)} / A_2^{(6)} \right)^{1/64} = \pm 1.16958 \Rightarrow x_3 = -1.16958 \tag{3.119}$$

Modulul perechii de rădăcini complexe este

$$\rho = \left(A_5^{(6)} / A_3^{(6)} \right)^{1/(2 \cdot 64)} = 0.9579761 \tag{3.120}$$

Rafinarea soluţiilor reale conduce la rădăcinile

$$x_1 = 3.43663,\ x_2 = 1.62659,\ x_3 = -1.16958 \tag{3.121}$$

după o singură iteraţie Newton-Raphson. Se observă calitatea extrem de bună a aproximaţiilor iniţiale furnizate de procedeul Lobacevschi-Graeffe. Pentru rădăcinile

complexe am putea extrage factorii deja găsiţi şi să rezolvăm ecuaţia pătratică ce rezultă. Vom proceda însă la aplicarea procedeului Bairstow pentru exemplificarea modului de lucru. Luăm $q = \rho^2$ şi $p = 0$ ca aproximaţie iniţială ceea ce corespunde la $\theta = \pi/2$. Se obţin următoarele rezultate

Iteraţie	p	q	b_0	b_1	R	S
0	0.0000	0.9180	-1.25E-01	-1.08E+00	-1.0753	-0.1249
1	-0.1052	0.9028	1.43E-01	-3.60E-02	-0.0360	0.1468
2	-0.1064	0.9407	-7.04E-04	3.54E-04	0.0004	-0.0007
3	-0.1064	0.9178	-4.73E-09	1.76E-08	0.0000	0.0000

Radăcinile complexe satisfac ecuaţia

$$x^2 + px + q = 0 \tag{3.122}$$

cu $p = -0.1063682185$, $q = 0.91771811581$ de unde rădăcinile

$$x_{4,5} = 0.0531841 \pm 0.956499i\,. \tag{3.123}$$

Se observă o convergenţă rapidă a iteraţiilor. Plecarea din $\theta = 0$ însă nu converge către soluţie.

EXEMPLU 2. Să se determine rădăcinile polinomului

$$p_4(x) = x^4 - 6x^3 + 18x^2 - 30x + 25$$

Rezolvare. Cum nici unul din rapoartele r_1, r_2, r_3 nu se stabilizează deducem prezenţa a două perechi de rădăcini complexe, de acelaşi modul

$$\rho_1 = \rho_2 = \rho_3 = \rho_4 = (A_4/A_0)^{1/(464)} = \sqrt{5}\,.$$

Fie rădăcinile $x_{1,2} = a_1 + ib_1$, $x_{3,4} = a_2 + ib_2$. Rezultă

$$a_1^2 + b_1^2 = a_2^2 + b_2^2 = 5\,.$$

Utilizând primele două relaţii Vieta pentru polinomul original se obţine

$$a_1 + a_2 = 3,\ a_1 a_2 = 2$$

de unde $a_1 = 1$, $b_1 = 2$, $a_2 = 2$, $b_2 = 1$. Prin urmare rădăcinile sunt

$$x_{1,2} = 1 + 2i,\ x_{3,4} = 2 + i\,.$$

Exemplu 3. Un ultim exemplu cu implicaţii profunde pentru matematică provine din întrebarea relativ simplă: din ce puncte ale planului complex converge metoda Newton pentru aflarea rădăcinilor de ordinul p ale unităţii? Procesul iterativ este

$$z_{k+1} = z_k - (z_k^p - 1)/(pz_k^{p-1}), \quad p > 1. \tag{3.124}$$

Pentru aflarea răspunsului se efectuează o serie de experimente numerice, luând diverse puncte din $[-2, 2] \times [-2, 2]$ ca puncte de start. Se înregistrează numărul de iteraţii necesare atingerii unei rădăcini sau faptul că procesul diverge. Frontiera dintre domeniul "atractiv", pentru care procesul converge şi cel "repulsiv", pentru care procesul diverge este extrem de complicată. Cercetări matematice moderne au arătat că este vorba de o nouă structură matematică cu proprietăţi remarcabile, printre care şi o dimensiune a frontierei ce nu este un număr întreg. O asemenea mulţime Julia este reprezentată pe copertă pentru $p = 6$. Nuanţele de culori închise indică convergenţa rapidă către o rădăcină pe când cele deschise indică divergenţa.

Capitolul 4

Erorile de calcul numeric

În momentul de faţă, suntem în posesia unui bagaj de metode numerice suficient pentru a considera mai în detaliu problema erorilor de calcul numeric. Se poate observa că o formulă de calcul numeric se aplică de obicei în mod repetat. În consecinţă, prezintă importanţă nu numai eroarea introdusă într-o etapă, ci şi tendinţa de a amplifica sau, dimpotrivă, de a atenua erorile introduse anterior, adică stabilitatea metodei numerice. Studiul erorilor numerice rămâne deci o problemă deschisă, care trebuie luată în considerare pentru fiecare metodă de calcul în parte.

4.1 Surse de erori

Erorile inerente sunt erorile legate de cunoaşterea aproximativă a unor valori provenite din măsurători sau din faptul că avem de-a face cu numere iraţionale (algebrice sau transcendente: numerele π, e, $\sqrt{3}$ ş.a.). Evident, rezultatul oricăror calcule depinde şi de precizia datelor introduse iniţial. Ca erori inerente pot fi considerate şi erorile de conversie făcute la trecerea în baza 2 a unor numere care se introduc în memoria calculatoarelor numerice actuale. Spre

exemplu, numărul 0.1 reprezentat printr-un număr finit de zecimale în baza 10, devine o fracţie zecimală periodică în baza 2 ($0.1_{10} = 0.0(0011)_2$).

Erorile de metodă sau *erorile de trunchiere* sunt provenite din aproximaţiile făcute la deducerea formulelor de calcul. Exemple: restul $R_N(x)$ la interpolarea polinomială, distanţa $|x_{n+1} - \alpha|$ la rădăcină, din metodele iterative de calcul, eroarea $\delta_i = -h^3/12\, f''(\xi)$ introdusă de formula de integrare a trapezelor pe un interval egal cu pasul h, erorile introduse prin trunchierea seriilor la un anumit rang, etc. Spre deosebire de erorile inerente, erorile de metodă pot fi reduse, în principiu, oricât de mult.

Erorile de rotunjire sunt legate de posibilităţile limitate de reprezentare a numerelor în calculatoarele numerice. În general, orice calculator poate reprezenta numerele cu un număr redus de cifre semnificative, depinzând de lungimea cuvântului (numărul de biţi) utilizat la stocarea unui număr. În mod curent se lucrează cu un echivalent de circa 7 cifre semnificative în simplă precizie şi de circa 15 cifre semnificative în dublă precizie.

După cum se ştie, în memoria internă a calculatoarelor actuale se foloseşte reprezentarea în virgulă mobilă, în forma normalizată. Astfel, orice număr real x se scrie

$$x = f \cdot b^n, \quad |f| < 1 \tag{4.1}$$

unde f este un număr real denumit *mantisă*, $b > 0$ ($b \neq 1$) este *baza* sistemului de numeraţie utilizat, iar n (întreg) este *exponentul*. În forma normalizată, mantisa este cuprinsă în intervalul $[b^{-1}, 1)$

$$b^{-1} \leq |f| < 1\,. \tag{4.2}$$

Singura excepţie de la acestă regulă de reprezentare este numărul zero.

În consecinţă, un număr real cu mai multe cifre semnificative este "rotunjit" la numărul de cifre maxim. Acest lucru se realizează prin rotunjirea mantisei. Alte rotunjiri se efectuează în decursul operaţiilor.

În general, notând cu x valoarea exactă a numărului şi cu $\overline{x}$ valoarea calculată (aproximativă), *eroarea absolută* e_x se defineşte ca diferenţa dintre valoarea exactă şi cea aproximativă

$$e_x = x - \overline{x}\,. \tag{4.3}$$

Raportul $e_x/\bar{x}$ se numeşte *eroare relativă*, notată deseori cu ε_x

$$\varepsilon_x = e_x/\bar{x}\,. \tag{4.4}$$

Fie t numărul de cifre semnificative. Pentru comoditate, să presupunem că lucrăm în baza 10 ($b = 10$). Atunci, un număr x a cărui valoare o presupunem iniţial cunoscută cu exactitate, se va scrie

$$x = f \cdot 10^n + g \cdot 10^{n-t}, \quad |f|\,, |g| \in [0.1, 1), \tag{4.5}$$

unde g conţine cifrele care nu pot fi incluse în mantisa f. Rotunjirea se face de obicei simetric, adică se înlocuieşte

$$|g| = 1 \text{ dacă } |g| \geq 0.5,\ |g| = 0 \text{ dacă } |g| < 0.5\,. \tag{4.6}$$

În acest fel, marginea erorii relative este

$$|e_x| = |g| \cdot 10^{n-t} / |f| \cdot 10^n \leq 5 \cdot 10^{-t}\,. \tag{4.7}$$

Erorile cu marginea dată de (4.7) se fac la introducerea numerelor reale în memoria calculatorului numeric. Ele afectează rezultatele în funcţie de operaţiile la care sunt supuse valorile introduse.

4.2 Propagarea erorilor în calcule

Propagarea erorilor la înmulţire.

Vom considera două numere, x şi y, introduse cu erorile e_x, respectiv e_y

$$x = \bar{x} + e_x, \quad y = \bar{y} + e_y\,. \tag{4.8}$$

Presupunem că se efectuează produsul numerelor

$$x\,y = (\bar{x} + e_x)(\vec{y} + e_y) = \bar{x}\,\bar{y} + \bar{y}\,e_x + \bar{x}\,e_y, \tag{4.9}$$

unde s-a neglijat produsul $e_x e_y$ considerat ca având un ordin de mărime suficient de mic. Rezultă eroarea la înmulţire

$$e_{xy}/\bar{x}\,\bar{y} = e_x/\bar{x} + e_y/\bar{y} = \varepsilon_x + \varepsilon_y\,. \tag{4.10}$$

Rezultă că la înmulţire *erorile relative introduse iniţial se adună*. În afară de aceste erori, pot apărea însă noi erori, deoarece produsul xy poate avea, la rândul său, un număr de cifre semnificative mai mare decât cel admis (notat cu t), necesitând o nouă rotunjire (simetrică). Notând cu ε_p această nouă eroare, vom obţine eroarea relativă totală ε_{tp} la înmulţirea a două numere

$$\varepsilon_{tp} = \varepsilon_x + \varepsilon_y + \varepsilon_p \tag{4.11}$$

iar ca margine a erorii

$$|\varepsilon_{tp}| \leq |\varepsilon_x| + |\varepsilon_y| + |\varepsilon_p| < 15 \cdot 10^{-t}\,. \tag{4.12}$$

Desigur, evaluarea (4.12), bazată pe relaţia (4.7), este acoperitoare deoarece erorile nu se adună, ci se compun după legi mai complicate (probabilistice).

Propagarea erorilor la împărţire.

Cu notaţiile (4.8) vom scrie

$$x/y = (\bar{x} + e_x)/(\bar{y} + e_y) = \bar{x}/\bar{y}(1 + e_x/\bar{x} - e_y/\bar{y}) = \bar{x}/\bar{y}(1 + \varepsilon_x - \varepsilon_y), \quad (4.13)$$

unde am neglijat termenii de grad superior lui 2 în e_x şi e_y. S-a folosit dezvoltarea în serie Taylor pentru $1/(1+\varepsilon_y) \cong 1 - \varepsilon_y + \ldots$. Rezultă că eroarea relativă la împărţire, datorată erorilor de reprezentare în calculator (e_x şi e_y), este

$$e_{x/y}/(\bar{x}/\bar{y}) = \varepsilon_x - \varepsilon_y, \quad (4.14)$$

adică este egală cu diferenţa erorilor relative introduse iniţial în valorile numărătorului şi numitorului. La acestea se adaugă eroarea de rotunjire la reprezentarea raportului $\bar{x}/\bar{y}$, pe care o vom nota cu ε_d, astfel încât eroarea relativă totală la împărţirea a două numere x, y este

$$\varepsilon_{td} = \varepsilon_x - \varepsilon_y + \varepsilon_d \,. \quad (4.15)$$

Propagarea erorilor la adunare.

În acest caz, suma a două numere este

$$x + y = \bar{x} + \bar{y} + e_x + e_y \,, \quad (4.16)$$

astfel încât eroarea relativă la sumare este

$$e_{x+y}/(\bar{x} + \bar{y}) = (e_x + e_y)/(\bar{x} + \bar{y}) = \bar{x}/(\bar{x} + \bar{y})\varepsilon_x + \bar{y}/(\bar{x} + \bar{y})\varepsilon_y, \quad (4.17)$$

adică o sumă ponderată a erorilor introduse la reprezentarea în calculator a cantităţii sumate. Şi în acest caz se introduce o eroare suplimentară la reprezentarea sumei $\bar{x} + \bar{y}$, a cărei valoare relativă o vom nota cu ε_s. Ca urmare, eroarea relativă la sumare, ε_{ts}, va fi

$$\varepsilon_{ts} = \bar{x}/(\bar{x} + \bar{y})\varepsilon_x + \bar{y}/(\bar{x} + \bar{y})\varepsilon_y + \varepsilon_s \,. \quad (4.18)$$

Ca aplicaţie, să scriem eroarea totală la calculul expresiei $E = (x + y)z \cong (\bar{x} + \bar{y})\bar{z}$. Rezultă

$$\varepsilon_{tE} = \bar{x}/(\bar{x} + \bar{y})\varepsilon_x + \bar{y}/(\bar{x} + \bar{y})\varepsilon_y + \varepsilon_z + \varepsilon_s + \varepsilon_p \,, \quad (4.19)$$

cu marginea

$$|\varepsilon_{tE}| \leq 5 \cdot 10^{-t} \left[(|\bar{x}| + |\bar{y}|)/\,|\bar{x} + \bar{y}| + 3\right] \quad (4.20)$$

Ca recomandare generală, în calculul numeric trebuie studiată şi propagarea erorilor. În anumite cazuri, acumularea erorilor poate conduce la rezultate complet eronate. Pentru a ilustra această posibilitate, să considerăm calculul integralei

$$I_n = \int_0^1 x^n e^{x-1} dx \,. \quad (4.21)$$

Tabelul 4.1: Exemplu de amplificare a erorilor

n	I_n (4.22)	I_n (4.24)	n	I_n (4.22)	I_n (4.24)
0	0.632121	0.632121	9	0.091586	0.091612
1	0.367879	0.367879	13	-0.377925	0.066948
3	0.207277	0.207277	17	–	0.052778
7	0.112383	0.112384	20	–	0.000000

O modalitate de calcul o reprezintă utilizarea unei relaţii de recurenţă

$$I_n = 1 - nI_{n-1}, \quad n = 1, 2, \dots , \tag{4.22}$$

plecând de la valoarea $I_0 = 1 - e^{-1}$. Rezultatele calculelor pentru diferitele valori ale lui în sunt date în tabelul 4.1 Se observă că, pentru $n = 13$, se obţine o valoare negativă a integralei, ceea ce nu este posibil, funcţia integrată fiind pozitivă pe $[0, 1]$. Rezultă că valorile obţinute sunt eronate, începând cu o anumită valoare a lui n. Explicaţia o constituie faptul că valoarea integralei I_0 se calculează cu o eroare e_0 care este amplificată prin aplicarea formulei de recurenţă (4.22) astfel că, la calculul lui I_{13} eroarea este

$$e_{13} \cong 13!\, e_0 = 6.227 \cdot 10^9\, e_0\,. \tag{4.23}$$

Pentru a obţine o valoare corectă în cazul indicilor n mari se aplică formula (4.22) sub forma

$$I_{n-1} = (1 - I_n)/n, \quad n = N, N-1, \dots , \tag{4.24}$$

în care erorile sunt reduse în fiecare etapă. Prin urmare, plecând de la un N suficient de mare eroarea introdusă din necunoaştera valorii I_n va fi redusă, astfel încât să obţinem rezultate corecte. Observând că valoarea integralei scade cu n, vom alege $N = 20$ şi vom aproxima $I_{20} \cong 0$, calculând integralele de indice mai mic din ce în ce mai precis. Valorile obţinute sunt date în acelaşi tabel 4.1 (s-a utilizat un calculator HP-25 care lucrează cu zece zecimale). Se observă că rezultatele sunt mai bune la indici mari şi identice la indici mici ($n < 7$).

Capitolul 5

Rezolvarea sistemelor liniare

Rezolvarea sistemelor cu un mare număr de ecuaţii liniare reprezintă unul din domeniile în care calculatoarele numerice şi-au dovedit din plin eficienţa. Problema rezolvării sistemelor de ecuaţii liniare este foarte des întâlnită în simularea numerică. Enumerăm câteva situaţii: interpolare cu funcţii spline cubice, rezolvarea sistemelor de ecuaţii neliniare cu ajutorul metodelor iterative care au la bază liniarizarea ecuaţiilor, discretizarea ecuaţiilor diferenţiale ordinare cu condiţii la limită, discretizarea ecuaţiilor cu derivate parţiale. În mod corespunzător, a trebuit să fie puse la punct procedee numerice adecvate, atât pentru reducerea numărului mare de operaţii, cât şi pentru reducerea erorilor de calcul care cresc cu dimensiunile sistemului de ecuaţii.

În cazul general, problema care trebuie rezolvată poate fi scrisă sub forma

$$\sum_{j=1}^{n} a_{i,j} x_j = b_i,\ i \in \overline{1,m}\ , \tag{5.1}$$

unde $a_{i,j} \in \mathbf{R}$ sunt coeficienţi, x_j, $j \in \overline{1,n}$ sunt necunoscutele sistemului, iar b_i sunt termenii liberi. Vom distinge trei situaţii.

(a) Pentru $m < n$ sistemul este *subdeterminat*, avem mai puţine ecuaţii decât necunoscute. În general, vor trebui aleşi $n-m$ parametrii pentru a obţine o soluţie.

(b) Pentru $m = n$ şi $\det A \neq 0$ sistemul este *compatibil determinat.* Sistemul are o soluţie unică. Este cazul cel mai des întâlnit. Pentru $m = n$ şi $\det A = 0$ sistemul poate fi *compatibil nedeterminat*, cu o infinitate de soluţii posibile, sau *incompatibil*, cu nici o soluţie.

(c) Pentru $m > n$ sistemul este *supradeterminat*, caz în care se caută o soluţie care să verifice "cel mai bine" ecuaţiile (5.1) în sensul minimizării reziduului $R = \sum_{i=1}^{n}(b_i - \sum_{j=1}^{n} a_{i,j}x_j)^2$.

Nu trebuie uitat că, la reprezentarea numerelor în calculator nu putem reţine decât un număr finit de cifre, iar erorile de rotunjire se propagă după regulile discutate în capitolul 4. Se poate ajunge în situaţia ca din cauza acestor erori, determinantul să devină egal cu zero sau să aibă o valoare foarte mică în modul. Complexitatea implementărilor prezente în bibliotecile matematice se datorează în mare parte tehnicilor de evitare a acestei situaţii, dar totodată şi eforturilor de minimizare a memoriei utilizate şi a timpului de calcul.

Metodele de rezolvare a sistemelor de ecuaţii liniare sunt de două tipuri: (a) *metode directe* (sau *metode de eliminare* sau *metode exacte*), în care soluţia este obţinută în urma unui număr de operaţii dinainte cunoscut; (b) *metode iterative*, care se bazează pe folosirea unei aproximaţii iniţiale ce se îmbunătăţeşte de la o etapă la alta.

5.1 Metode directe

5.1.1 Metoda eliminării a lui Gauss

Metoda constă în eliminarea succesivă a necunoscutelor într-o manieră care conduce la un număr de operaţii mult mai redus decât procedeul care ar utiliza regula lui Cramer şi calculul determinanţilor corespunzători. Vom observa că, în operaţia de eliminare, nu este necesar să transcriem şi necunoscutele; este suficient să operăm doar cu coeficienţii matricei sistemului şi cu termenii liberi. Să considerăm un sistem de n ecuaţii liniare cu n necunoscute scris sub forma

$$Ax = b \ , \tag{5.2}$$

unde A este o matrice pătrată, nesingulară ($\det A \neq 0$), de dimensiuni $n \times n$, iar x şi b sunt vectori coloană de dimensiune n. Pentru fixarea ideilor, vom considera că A are coeficienţii a_{ij}, iar b are componentele b_i ($i, j \in \overline{1,n}$). Pe de altă parte, deoarece operaţiile efectuate cu coeficienţii a_{ij} şi cu termenii liberi b_i

sunt simultane, este avantajos în prezentarea teoretică a algoritmului să renotăm coeficienţii $b_i \equiv a_{i,n+1}$ şi să considerăm *matricea extinsă* A'

$$A' = \left[\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \dots & a_{1,n} & a_{1,n+1} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} & a_{2,n+1} \\ \dots & \dots & \dots & \dots & \dots \\ a_{i,1} & a_{i,2} & \dots & a_{i,n} & a_{i,n+1} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} & a_{n,n+1} \end{array}\right] . \tag{5.3}$$

Metoda eliminării a lui Gauss constă în a obţine zerouri succesiv, întâi pe prima coloană (sub coeficientul $a_{1,1}$), apoi pe a doua coloană (sub coeficientul $a_{2,2}$) ş.a.m.d., pe ultima linie a matricei A rămânând doar coeficientul $a_{n,n}$ (evident modificat de operaţiile de eliminare anterioare). Aceasta revine la a reduce matricea A la o *matrice superior triunghiulară*, iar sistemul (5.2) la forma

$$\left[\begin{array}{cccccc} a_{1,1} & a_{1,2} & \dots & \dots & \dots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & \dots & \dots & \dots & a_{2,n}^{(1)} \\ 0 & 0 & \ddots & \dots & \dots & \dots \\ 0 & 0 & 0 & a_{i,j}^{(i-1)} & \dots & a_{i,n}^{(i-1)} \\ 0 & 0 & 0 & 0 & \ddots & \dots \\ 0 & 0 & 0 & 0 & 0 & a_{n,n}^{(n-1)} \end{array}\right] \cdot \left[\begin{array}{c} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{array}\right] = \left[\begin{array}{c} a_{1,n+1} \\ a_{2,n+1}^{(1)} \\ \vdots \\ a_{i,n+1}^{(i-1)} \\ \vdots \\ a_{n,n+1}^{(n-1)} \end{array}\right] . \tag{5.4}$$

Indicii superiori indică etapa în care a fost obţinut elementul. Pentru a obţine zerourile de sub diagonala principală, se folosesc operaţii simple de înmulţire a unei linii cu un *multiplicator* şi de scădere din altă linie. Spre exemplu, pentru a obţine zerouri pe prima coloană, din linia i se scade prima linie înmulţită cu multiplicatorul $m_{i,1}$, obţinându-se

$$a_{i,1}^{(1)} = a_{i,1} - m_{i,1}a_{1,1} = 0 \Rightarrow m_{i,1} = a_{i,1}/a_{1,1} . \tag{5.5}$$

Repetând procedeul pentru $i = \overline{2,n}$, se obţin elemente nule pe coloana întâi în această primă etapă. Evident, pentru a putea opera, trebuie ca $a_{1,1} \neq 0$. Mai mult, după cum vom vedea la paragraful (5.1.3), se recomandă ca $a_{1,1}$ să fie în modul cât mai mare posibil, deoarece în acest mod, erorile de rotunjire sunt reduse. Elementul $a_{i,i}$ plasat pe diagonala principală se numeşte *pivot*. Obţinerea unui pivot cu modul cât mai mare este posibilă prin schimbări de linii şi coloane care nu afectează zerourile deja produse, adică pivotul se poate alege dintre elementele de sub şi/sau la dreapta poziţiei de pe diagonala principală a pivotului. În funcţie de numărul de elemente dintre care este selectat, pivotul poate fi *parţial*, când se alege cel mai mare în modul[1] dintre elementele de pe coloana pivotului, sub diagonala principală, sau *total*, când se alege cel mai mare în modul dintre toate elementele conţinute în linii şi coloane care, interschimbate, *nu modifică zerourile deja obţinute*. În cazul în care pentru obţinerea unui pivot

[1] Cu privire la alegerea elementului "cel mai mare în modul", vezi şi paragraful 5.1.3.

convenabil este necesară o schimbare de linii, aceasta poate fi făcută efectiv în matricea extinsă A', sau se poate memora doar indicele liniei pe care se află elementul dorit ca pivot, cu ajutorul unui vector l. Dacă în etapa s elementul dorit ca pivot se află pe linia i (evident $i \geq s$, pentru a păstra zerourile deja obţinute), atunci atribuim $l_s = i$. Acest mod de lucru complică algoritmul, fără a aduce avantaje privind timpul de calcul, astfel încât recomandăm prima variantă. La interschimbarea a două coloane (j cu k), vo m observa că trebuie interschimbate şi necunoscutele (x_j cu x_k). Acest lucru se poate realiza cu ajutorul unui vector c, care este iniţializat cu valorile $c = (1, 2, \ldots, k, \ldots, n)$. Dacă spre exemplu în prima etapă schimbăm coloanele 1 cu k, se schimbă corespunzător şi elementele c_1 cu c_k, iar vectorul c devine $c = (k, 2, \ldots, 1, \ldots, n)$.

În fiecare etapă $s \in \overline{1, n-1}$, elementele din matrice se modifică după regula

$$\begin{array}{l} s = 1 : (n-1) \\ \left[\begin{array}{l} i = (s+1) : n \\ \quad \left[\begin{array}{l} m_{i,s} \leftarrow a_{i,s}^{(s-1)} / a_{s,s}^{(s-1)} \\ j = (s+1) : (n+1) \\ \quad \left[a_{i,j}^{(s)} \leftarrow a_{i,j}^{(s-1)} - m_{i,s} a_{s,j}^{(s-1)} \right. \end{array} \right. \end{array} \right. \end{array} \quad . \tag{5.6}$$

Într-adevăr, în etapa (s) pivotul este $a_{s,s}^{(s-1)}$, iar elementele care se anulează sunt plasate în poziţia $a_{i,s}$, $i \in \overline{s+1, n}$. Practic, aceste elemente nu se mai calculează deoarece se ştie că ele sunt nule, iar pe poziţiile acestora se pot memora multiplicatorii $m_{i,s}$, eventual necesari pentru calcule ulterioare cum ar fi rafinarea soluţiei (paragraful 5.1.3). În final, dacă matricea A nu este singulară, se ajunge la forma (5.4) a sistemului, coeficienţii $a_{i,j}^{(i-1)}$, $i \in \overline{1, n}$, $j \in \overline{1, n+1}$ fiind obţinuţi prin operaţii asupra matricei extinse A'. Ca urmare, necunoscutele se obţin prin *retrosubstituire*, după schema

$$\begin{array}{l} x_{c_n} \leftarrow a_{n,n+1}^{(n-1)} / a_{n,n}^{(n-1)} \\ i = (n-1) : 1 \\ \left[x_{c_i} \leftarrow (a_{i,n+1}^{(i-1)} - \displaystyle\sum_{j=i+1}^{n} a_{i,j}^{(i-1)} x_{c_j}) \,/\, a_{i,i}^{(i-1)} \right. \end{array} \quad , \tag{5.7}$$

în care am ţinut cont şi de eventualele schimbări de coloane necesare în cazul folosirii pivotării totale şi memorate în vectorul c. Trebuie observat faptul că elementele matricei iniţiale A se pierd dacă se folosesc schemele (5.6) şi (5.7). În cazul în care matricea A este necesară pentru calcule ulterioare (cum ar fi rafinarea inversei, vezi. 5.1.7), trebuie păstrată o copie a acesteia.

Numărul de operaţii necesare pentru obţinerea soluţiei prin procedeul de eliminare gaussiană se calculează uşor. Evident, nu luăm în considerare şi operaţiile legate de permutarea elementelor în vederea găsirii pivotului parţial sau total, deoarece acestea depind de fiecare matrice în parte. De regulă, se consideră că împărţirea reprezintă o singură "operaţie", în timp ce o adunare şi o înmulţire formează împreună tot a "operaţie" (aproximativ acelaşi timp

de calcul pe calculator). Amintindu-ne că operaţiile se efectuează cu elementele matricei extinse A', de dimensiuni $n \cdot (n+1)$, vom avea de efectuat, după schema (5.6), într-o etapă s, $n-s$ împărţiri pentru a obţine multiplicatorii $m_{i,s}$. Numărul de adunări şi înmulţiri este egal cu produsul numărului de valori luate de indicii i, j, adică $(n-s)(n+1-s)$. Rezultă numărul de operaţii la eliminarea Gauss

$$n_G = \sum_{s=1}^{n-1}[(n-s)^2 + 2(n-s)] = \frac{n^3}{3} + \frac{n^2}{2} - \frac{5n}{6} \ . \tag{5.8}$$

Acest număr este foarte mic (începând cu $n \geq 3$) faţă de $n \cdot n!$ operaţii cerute de regula lui Cramer (cu determinanţii calculaţi după minori). La operaţiile (5.8) se adaugă încă aproximativ $1 + (n-1) + n(n-1)/2 = n^2/2 + n/2$ operaţii pentru retrosubstituirea (5.7). Rezultă numărul total de operaţii pentru eliminarea gaussiană

$$n_G = \frac{n^3}{3} + n^2 - \frac{n}{3} \approx \frac{n^3}{3} + n^2 \ . \tag{5.9}$$

aproximaţia fiind valabilă pentru $n \gg 1$.

Relaţiile (5.6) şi (5.7) se pot optimiza obţinând 1 pe diagonala principală, după algoritmul dat de (5.10), unde am inclus şi calculul determinantului matricei A, $d = \det A$. Numărul de operaţii se reduce (nesemnificativ) cu n împărţiri la etapa de retrosubstituire.

Algoritm Gauss

[Eliminare]
$d \leftarrow 1$
$s = 1 : (n-1)$
$\left[\begin{array}{l} j = (s+1) : (n+1) \\ \quad [a_{s,j} \leftarrow a_{s,j}/a_{s,s}] \\ d \leftarrow d \cdot a_{s,s};\ a_{s,s} \leftarrow 1 \\ i = (s+1) : n \\ \left[\begin{array}{l} j = (s+1) : n \\ \quad \left[a_{i,j}^{(s)} \leftarrow a_{i,j}^{(s-1)} - a_{i,s}^{(s-1)} a_{i,j}^{(s-1)} \right. \end{array} \right. \end{array} \right.$
[Retrosubstituire]
$x_n \leftarrow a_{n,n+1}^{(n-1)}$
$i = (n-1) : 1$
$\left[x_i \leftarrow a_{i,n+1}^{(i-1)} - \sum_{j=i+1}^{n} a_{i,j}^{(i-1)} x_j \right.$

Algoritm Gauss-Jordan

[Eliminare]
$d \leftarrow 1$
$s = 1 : n$
$\left[\begin{array}{l} j = (s+1) : (n+1) \\ \quad [a_{s,j} \leftarrow a_{s,j}/a_{s,s}] \\ d \leftarrow d \cdot a_{s,s};\ a_{s,s} \leftarrow 1 \\ i = 1 : n \\ \left[\begin{array}{l} \text{dacă } i \neq s \\ \left[\begin{array}{l} j = (s+1) : n \\ \quad \left[a_{i,j}^{(s)} \leftarrow a_{i,j}^{(s-1)} - a_{i,s}^{(s-1)} a_{i,j}^{(s-1)} \right. \end{array} \right. \end{array} \right. \end{array} \right.$
[Calculul necunoscutelor]
$i = 1 : n$
$\left[x_i \leftarrow a_{i,n+1}^{(i-1)} \right.$

(5.10)

EXEMPLU. Folosind metoda eliminării gaussiene, să se rezolve sistemul de ecuaţii

$$\left\{\begin{array}{rcrcrcr} x_1 & + & \frac{2}{9}x_2 & + & x_3 & = & 2 \\ 8x_1 & + & 2x_2 & - & 3x_3 & = & -1 \\ x_1 & + & 2x_2 & - & 5x_3 & = & 1 \end{array}\right. . \tag{5.11}$$

Scriem matricea extinsă şi efectuăm eliminările după regula (5.6). Obţinem succesiv

$$\left[\begin{array}{cccc} \boxed{1} & \frac{2}{9} & 1 & 2 \\ 8 & 2 & -3 & -1 \\ 1 & 2 & -5 & 1 \end{array}\right] \overset{s=1}{\to} \left[\begin{array}{cccc} 1 & \frac{2}{9} & 1 & 2 \\ 0 & \boxed{\frac{2}{9}} & -11 & -17 \\ 0 & \frac{16}{9} & -6 & -1 \end{array}\right] \overset{s=2}{\to} \left[\begin{array}{cccc} 1 & \frac{2}{9} & 1 & 2 \\ 0 & \frac{2}{9} & -11 & -17 \\ 0 & 0 & 82 & 135 \end{array}\right] .$$

Rezultă

$$x_3 = 135/82 \simeq 1.6463, \quad x_2 = (9/2)\left(17 + 11\tfrac{135}{82}\right) = 819/164 \simeq 4.9939, \\ x_1 = 2 - 135/82 - (2/9)(819/164) = -31/41 \simeq -0.7561 . \tag{5.12}$$

Valoarea din chenar reprezintă pivotul etapei respective. Modul de lucru nu este cel mai adecvat pentru transpunerea pe calculator deoarece: (a) eliminările s-au făcut fără a utiliza nici o tehnică de pivotare (parţială sau totală); (b) calculele s-au efectuat utilizând fracţii ordinare, ceea ce implică programe speciale.

Pentru evidenţierea erorilor şi ţinând cont că sistemul este foarte mic, reluăm calculele folosind pivotarea parţială şi lucrând doar cu trei cifre semnificative[2]. Se obţine

$$\left[\begin{array}{cccc} 1 & 0.222 & 1 & 2 \\ 8 & 2 & -3 & -1 \\ 1 & 2 & -5 & 1 \end{array}\right] \overset{l_1 \leftrightarrow l_2}{\to} \left[\begin{array}{cccc} \boxed{8} & 2 & -3 & -1 \\ 1 & 0.222 & 1 & 2 \\ 1 & 2 & -5 & 1 \end{array}\right] \overset{s=1}{\to}$$

$$\overset{s=1}{\to} \left[\begin{array}{cccc} 8 & 2 & -3 & -1 \\ 0 & -0.028 & 1.38 & 2.13 \\ 0 & 1.75 & -4.63 & 1.13 \end{array}\right] \overset{l_2 \leftrightarrow l_3}{\to} \left[\begin{array}{cccc} 8 & 2 & -3 & -1 \\ 0 & \boxed{1.75} & -4.63 & 1.13 \\ 0 & -0.028 & -1.38 & 2.13 \end{array}\right] \overset{s=2}{\to}$$

$$\overset{s=2}{\to} \left[\begin{array}{cccc} 8 & 2 & -3 & -1 \\ 0 & 1.75 & -4.63 & 1.13 \\ 0 & 0 & 1.31 & 2.15 \end{array}\right] .$$

Rezultă

$$\bar{x}_3 = 0.164 \cdot 10^1, \ \bar{x}_2 = 0.498 \cdot 10^1, \ \bar{x}_1 = -0.755 \cdot 10^0 . \tag{5.13}$$

Valorile necunoscutelor fiind aproximative, s-a folosit notaţia cu bară $\bar{x}$. Scrierea este cu virgulă mobilă normalizată în sistem zecimal. Deşi sistemul este mic şi relativ bine condiţionat (vezi paragraful 5.1.8, numărul de condiţionare calculat cu (5.64) şi norma euclidiană este $\mu(A) = 24$, iar calculat cu (5.85) este $\mu(A) = 21.6$, în timp ce numărul

[2]Calculele intermediare au fost efectuate însă cu şapte cifre semnificative.

de condiţionare al matricei Hilbert[3] 3×3 calculat cu (5.85) este 524), utilizarea a numai trei cifre semnificative a condus la erori resimţite la a treia cifră semnificativă.

Pentru a observa influenţa alegerii pivotului asupra preciziei rezultatelor, refacem calculele de mai sus fără a mai schimba liniile 2 cu 3 pentru a aduce valoarea 1.75 pe poziţia pivotului în etapa a doua. Se obţine

$$\left[\begin{array}{cccc} 8 & 2 & -3 & -1 \\ 0 & \boxed{-0.028} & 1.38 & 2.13 \\ 0 & 1.75 & -4.63 & 1.13 \end{array}\right] \overset{s=2'}{\rightarrow} \left[\begin{array}{cccc} 8 & 2 & -3 & -1 \\ 0 & -0.028 & 1.38 & 2.13 \\ 0 & 0 & 81.6 & 1.13 \end{array}\right],$$

cu soluţiile

$$\bar{x}'_3 = 0.164 \cdot 10^1,\ \bar{x}'_2 = 0.476 \cdot 10^1,\ \bar{x}'_1 = -0.7 \cdot 10^0\ . \tag{5.14}$$

Se observă apariţia unor erori mult mai mari, comparativ cu valorile (5.12). Normele maxime ale erorilor sunt

$$\|x - \bar{x}\|_\infty = 0.139 \cdot 10^{-1}, \|x - \bar{x}'\|_\infty = 0.234\ . \tag{5.15}$$

5.1.2 Metoda Gauss-Jordan

Este similară cu metoda precedentă, cu deosebirea că produce zerouri atât dedesubtul cât şi deasupra diagonalei principale, reducând astfel matricea iniţială la o matrice diagonală. În acest fel, numărul de operaţii este cu circa 50% mai mare (nu este de două ori mai mare cum ne-am fi aşteptat, unele reduceri apar la determinarea necunoscutelor care nu se mai obţin prin retrosubstituire, ci direct, printr-o singură împărţire). Ca urmare a numărului sporit de operaţii, utilizarea metodei *Gauss-Jordan* la rezolvarea sistemelor de ecuaţii nu este justificată. Această metodă este însă utilizată la calculul numeric al *matricei inverse*, operaţie pe care o vom prezenta la paragraful 5.1.5. Numărul de operaţii, n_{GJ}, pentru metoda de eliminare Gauss-Jordan se calculează observând că la fiecare etapă s, se calculează $n-1$ multiplicatori. Algoritmul eliminării Gauss-Jordan este dat de (5.10). Rezultă numărul de operaţii la eliminarea Gauss-Jordan

$$n_{GJ} = \sum_{s=1}^{n}[(n-1) + (n-1)(n+1-s)] = \frac{n^3}{2} + n^2 - \frac{3n}{2} \approx \frac{n^3}{2} + n^2\ , \tag{5.16}$$

aproximaţia fiind valabilă pentru $n \gg 1$.

5.1.3 Propagarea erorilor la metodele de eliminare. Rafinarea soluţiei

Formula de calcul a coeficienţilor matricei extinse este dată de relaţiile (5.6) pe care le retranscriem sub forma

$$a_{i,j}^{(s)} = a_{i,j}^{(s-1)} - a_{i,s}^{(s-1)} a_{s,j}^{(s-1)} / a_{s,s}^{(s-1)},\ j \in \overline{s+1, n+1}\ , \tag{5.17}$$

[3]Elementele matricei Hilbert sunt date de relaţia $h_{i,j} = 1/(i+j+1)$. Matricea Hilbert este un caz tipic de matrice rău condiţionată; spre exemplu numărul de condiţionare al matricei Hilbert de 5×5 este de ordinul 10^5 iar pentru 6×6 este de ordinul 10^7.

unde $i \in \overline{s+1,n}$, $s \in \overline{1,n-1}$ la procedeul Gauss şi $i = 1,2,\dots s-1,s+1,\dots n$, $s \in \overline{1,n}$ la procedeul Gauss-Jordan. Rezultă eroarea relativă totală

$$\begin{aligned} \varepsilon_{i,j}^{(s)} &= \frac{\overline{a_{i,j}^{(s-1)}}}{\overline{a_{i,j}^{(s)}}}\varepsilon_{i,j}^{(s-1)} - \frac{\overline{a_{i,s}^{(s-1)}}\,\overline{a_{s,j}^{(s-1)}}}{\overline{a_{s,s}^{(s-1)}}\,\overline{a_{i,j}^{(s)}}}\varepsilon' + \varepsilon_s \ , \\ \varepsilon' &\equiv \varepsilon_{i,s}^{(s-1)} + \varepsilon_{s,j}^{(s-1)} - \varepsilon_{s,s}^{(s-1)} + \varepsilon_p + \varepsilon_d \, . \end{aligned} \tag{5.18}$$

În expresia (5.18) s-a ţinut seama de propagarea erorilor la scădere (sau sumare), la produs şi la împărţire, după regulile date în capitolul 4. Marginea erorii relative totale este

$$\varepsilon_{i,j}^{(s)} \le 5 \cdot 10^{-t} \left(\left| \frac{\overline{a_{i,j}^{(s-1)}}}{\overline{a_{i,j}^{(s)}}} \right| + 5 \left| \frac{\overline{a_{i,s}^{(s-1)}}\,\overline{a_{s,j}^{(s-1)}}}{\overline{a_{s,s}^{(s-1)}}\,\overline{a_{i,j}^{(s)}}} \right| + 1 \right) , \tag{5.19}$$

unde t este numărul de cifre semnificative cu care se lucrează. Se observă că singurul factor asupra căruia se poate acţiona este pivotul $a_{s,s}^{(s-1)}$ a cărui valoare în modul se recomandă, în general, să fie cât mai mare. În legătură cu strategia pivotului maxim, se poate face însă şi următoarea remarcă: deoarece, conform relaţiei (5.35), produsul pivoţilor este în modul egal cu valoarea lui det A, rezultă că introducerea de pivoţi mari, în modul, în primele etape de eliminare, va implica în mod obligatoriu apariţia unor pivoţi mici în etapele finale, cu atât mai devreme cu cât valoarea absolută a determinantului matricei A este mai mică. Prin urmare, este posibilă o strategie optimă de alegere a pivoţilor.

Un aspect şi mai important legat de strategia alegerii pivotului îl constituie *scalarea ecuaţiilor* (sau *precondiţionarea prin echilibrare*), care poate fi făcută atât pe linii cât şi pe coloane. Pentru a înţelege mai bine care este problema, să presupunem că înmulţim una din liniile sistemului cu un număr foarte mare. Acest lucru nu modifică teoretic soluţia sistemului dar avem mari şanse ca elementul maxim în modul să se afle pe această linie, chiar dacă acest lucru nu s-ar fi întâmplat cu linia originală. Este necesară deci *scalarea* matricei care constă în înmulţirea fiecărei linii cu câte un multiplicator r_i astfel încât *elementele maxime în modul de pe fiecare linie să fie egale cu unitatea*. Sistemul (5.1) devine

$$\sum_{j=1}^{n} r_i a_{i,j} x_j = r_i b_i, \ r_i = 1/\max_{j\in\overline{1,n}} |a_{ij}|, \ i \in \overline{1,n} \ , \tag{5.20}$$

Pentru a reduce pe cât de mult posibil erorile de rotunjire, multiplicatorul se alege de forma $1/2^m$ astfel încât elementele maxime de pe fiecare linie să fie cât mai apropiate de unitate. În cazul în care se calculează inversa sau determinantul, aceşti multiplicatori trebuie memoraţi.

Similar, *precondiţionarea prin echilibrare pe coloane*, constă în transformarea sistemului (5.1) în sistemul echivalent

$$\sum_{j=1}^{n} p_j a_{i,j} (x_j/p_j) = b_i, \ p_j = 1/\max_{i\in\overline{1,n}} |a_{ij}|, \ j \in \overline{1,n} \ , \tag{5.21}$$

Tot pe linia obţinerii unei unei precizii cât mai bune se înscrie şi *rafinarea soluţiei*, care constă în îmbunătăţirea soluţiei printr-un proces iterativ. Notând cu e vectorul eroare şi cu $\bar{x}$ vectorul soluţiei aproximative găsite, înlocuind în sistemul (5.2) vom obţine o valoare $\bar{b} \neq b$

$$A\bar{x} = \bar{b},\ \bar{x} = x - e\ , \tag{5.22}$$

adică apare o diferenţă faţă de vectorul b, diferenţă pe care o vom nota cu β

$$\beta = b - \bar{b}\ . \tag{5.23}$$

Prin scăderea ecuaţiilor (5.2) şi (5.22), se obţine un sistemul

$$Ae = \beta\ , \tag{5.24}$$

care are aceeaşi matrice cu sistemul iniţial. Prin urmare, sistemul (5.24) se rezolvă cu acelaşi algoritm, nefiind necesare decât operaţiile de eliminare aplicate asupra termenilor liberi. Se obţine soluţia rafinată

$$\bar{x}_1 = \bar{x} + e\ . \tag{5.25}$$

Evident, nici această soluţie (5.25) nu este exactă, deoarece la determinarea vectorului eroare e se introduc aceleaşi erori ca la rezolvarea sistemului iniţial (5.2), prin transformările efectuate asupra matricei A. O nouă rafinare, plecând de la x_1 este posibilă, dar aceasta nu îmbunătăţeşte în mod necesar soluţia sistemului. Condiţii suficiente pentru corectarea soluţiei se vor da la paragraful 5.1.7.

EXEMPLU. Reluăm exemplul (5.11). Indiferent de metoda utilizată, soluţia trebuie verificată în sistem. În acest fel, se găseşte vectorul erorilor β (5.23)

Soluţia	β_1	β_2	β_3	$\|\beta\|_\infty / \|b\|_\infty$
(5.13)	$0.944 \cdot 10^{-2}$	0	$-0.5 \cdot 10^{-2}$	$0.47 \cdot 10^{-2}$
(5.14)	$0.328 \cdot 10^{-2}$	0	$0.38 \cdot 10^{0}$	$0.19 \cdot 10^{0}$

Se constată că, în cazul soluţiei (5.14), există o eroare relativă mare (circa 20%) în termenii liberi, deci se impune rafinarea soluţiei. Presupunînd că efectuăm asupra matricei A aceleaşi operaţii care ne-au condus la soluţia (5.14), sistemul (5.24) pentru determinarea vectorului eroare e conduce la matricea

$$\left[\begin{array}{ccc|c} 8 & 2 & -3 & 0 \\ 0 & \boxed{-0.028} & 1.38 & 0.00328 \\ 0 & 1.75 & -4.63 & 0.38 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 8 & 2 & -3 & 0 \\ 0 & -0.028 & 1.38 & 0.00328 \\ 0 & 0 & 81.6 & 0.585 \end{array}\right],$$

cu soluţia

$$e_3 = 0.717 \cdot 10^{-2},\ e_2 = 0.236 \cdot 10^{0},\ e_1 = -0.563 \cdot 10^{-1}$$

Soluţia corectată este

$$\bar{x}_3'' = 0.165 \cdot 10^{1},\ \bar{x}_2'' = 0.450 \cdot 10^{1},\ \bar{x}_1 = -0.756,\ \ (\bar{x}_i'' = \bar{x}_i' + e_i)\ ,$$

cu norma maximă a erorii

$$\|x - \bar{x}''\|_\infty = 0.4 \cdot 10^{-1}\ .$$

Se observă o reducere a normei erorii absolute în raport cu soluţia $\bar{x}'$, dar rezultatul este sub precizia realizată cu pivotare, chiar fără rafinare (5.12), care are norma maximă dată de (5.15).

5.1.4 Interpretarea matriceală a metodelor de eliminare

Eliminarea elementelor de pe coloane este echivalentă cu înmulţirea matricei A cu anumite matrice simple. Spre exemplu, în prima etapă, matricea A este înmulţită la stânga cu o matrice notată cu L_1, în a doua etapă cu o matrice L_2, ş.a.m.d. Se verifică direct că aceste matrice sunt de forma

$$L_1 = \begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ -m_{2,1} & 1 & \dots & \dots & 0 \\ -m_{3,1} & 0 & \ddots & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ -m_{n,1} & 0 & \dots & 0 & 1 \end{bmatrix}, \quad L_2 = \begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & \dots & \dots & 0 \\ 0 & -m_{3,2} & \ddots & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -m_{n,2} & \dots & \dots & 1 \end{bmatrix},$$

adică se obţin din matricea unitate de dimensiuni $n \times n$ prin introducerea multiplicatorilor, cu semn schimbat, în poziţiile specificate chiar prin indicii acestor multiplicatori. Ca probă, elementul $p_{i,1}$ din matricea produs $L_1 A$ este

$$p_{i,1} = -m_{i,1} \cdot a_{1,1} + 1 \cdot a_{i,1} = 0, \ i = 2, 3, \dots, n \ . \tag{5.26}$$

În consecinţă, în absenţa permutărilor de linii şi coloane (pe care de fapt le efectuăm pentru obţinerea pivoţilor de modul convenabil), se obţine produsul de matrice

$$U = L_{n-1} L_{n-2} \dots L_2 L_1 A \ , \tag{5.27}$$

U fiind matricea *superior triunghiulară*[4], obţinută în etapa $n-1$ a eliminării gaussiene (etapa finală). Vom observa, mai departe, că matricele L_s, $s \in \overline{1, n-1}$ sunt toate inversabile, având determinanţi egali cu unitatea. Mai mult, se verifică direct că matricele inverse L_s^{-1} se obţin din matricele L_s prin schimbarea semnelor elementelor $m_{i,s}$. Notând cu $L = (L_{n-1} L_{n-2} \dots L_2 L_1)^{-1}$ matricea inversă a transformărilor matricei A (5.27), rezultă că la eliminarea gaussiană (fără permutări de linii sau coloane), s-a realizat factorizarea

$$A = L \cdot U \ , \tag{5.28}$$

unde L este o matrice inferior triunghiulară având elemente egale cu unitatea pe diagonala principală, iar în rest multiplicatorii utilizaţi la eliminare

$$L = L_1^{-1} L_2^{-1} \dots L_{n-1}^{-1} = \begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ m_{2,1} & 1 & \dots & \dots & 0 \\ m_{3,1} & m_{3,2} & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ m_{n,1} & m_{n,2} & \dots & m_{n,n-1} & 1 \end{bmatrix}. \tag{5.29}$$

[4] Literele L şi U sunt preluate de la cuvintele din limba engleză *lower* şi *upper*.

Pe de altă parte, matricea U, superior triunghiulară, are expresia

$$U = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \dots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \dots & a_{2,n}^{(1)} \\ 0 & 0 & a_{3,3}^{(2)} & \dots & a_{3,n}^{(2)} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & a_{n,n}^{(n-1)} \end{bmatrix} . \tag{5.30}$$

Permutarea de linii sau coloane revine la înmulţirea matricei ale cărei linii sau coloane se permută, cu o *matrice de permutare*. Vom nota cu $P_{i,j}$ matricea obţinută prin permutarea în matricea unitate de aceleaşi dimensiuni, a liniilor sau coloanelor i şi j. Pentru a permuta *liniile* i şi j dintr-o matrice oarecare, se înmulţeşte matricea respectivă cu $P_{i,j}$ la *stânga*, iar pentru a permuta *coloanele* i şi j, se înmulţeşte cu $P_{i,j}$ la *dreapta*. Presupunem că, spre exemplu, pentru a ajunge la forma triunghiulară U' s-au folosit atât permutări de linii cât şi de coloane, astfel încât, înainte de a efectua eliminările în coloana a doua, s-au permutat liniile i, j şi coloanele k, l $(i, j, k, l \geq 2)$. Se obţine

$$U' = L_{n-1} \dots L_2 P_{i,j} L_1 A P_{k,l} \tag{5.31}$$

unde $P_{i,j}$ şi $P_{k,l}$ sunt matrice de permutare. Din (5.31) se poate trage concluzia că, datorită permutărilor, matricea A nu se mai descompune chiar într-un produs de matrice triughiulare. De fapt

$$A = (L_1^{-1} P_{i,j} L_2^{-1} \dots L_{n-1}^{-1}) U' P_{k,l} \ , \tag{5.32}$$

unde am ţinut cont că $P_{i,j}^{-1} = P_{i,j}$, sau

$$P_{i,j} A P_{k,l} = L U' \ , \tag{5.33}$$

unde am ţinut cont că operaţiile de permutare de linii şi de eliminare sunt comutative, adică

$$P_{i,j} L_1^{-1} P_{i,j} (L_2^{-1} \dots L_{n-1}^{-1} U' P_{k,l}) = P_{i,j} P_{i,j} L_1^{-1} (L_2^{-1} \dots L_{n-1}^{-1} U' P_{k,l}) \tag{5.34}$$

Eliminarea gaussiană poate fi utilizată şi pentru calculul valorii determinanţilor. Din (5.32) se obţine

$$\det A = (-1)^{n_p} \det U' = a_{1,1} a_{2,2}^{(1)} \dots a_{n,n}^{(n-1)} (-1)^{n_p} \ , \tag{5.35}$$

unde n_p reprezintă numărul total de permutări de linii şi coloane. Deoarece $\det A \neq 0$, rezultă că nici unul dintre pivoţii $a_{s,s}^{(s-1)}$, $(s \in \overline{1, n})$ nu poate fi nul.

5.1.5 Calculul matricei inverse

Matricea A este inversabilă dacă $\det A \neq 0$, condiţie pusă şi pentru existenţa soluţiei sistemului (5.2). Prin urmare, este de aşteptat ca rezolvarea sistemului

şi inversarea matricei A să fie probleme strâns legate. Într-adevăr, matricea B este inversa matricei A dacă

$$AB = BA = I \ . \tag{5.36}$$

Dacă luăm în considerare prima coloană a matricei B, obţinem

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \dots & \dots & \dots & \dots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} \end{bmatrix} \cdot \begin{bmatrix} b_{1,1} \\ b_{2,1} \\ \dots \\ b_{n,1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \dots \\ 0 \end{bmatrix}, \tag{5.37}$$

adică un sistem de n ecuaţii cu n necunoscute. Rezultă deci că problema determinării matricei inverse B este echivalentă cu rezolvarea a n sisteme de n ecuaţii cu n necunoscute. Vom considera matricea M de dimensiuni $n \times 2n$, formată din alăturarea matricei unitate I la matricea A, urmând a aplica transformări matricei A astfel încât să o aducem la matricea unitate I,

$$M = [A|I] \to [I|A] \ . \tag{5.38}$$

În acest scop, se aplică transformările de la eliminarea Gauss- Jordan. Rezultă o relaţie de forma

$$I = D^{(n)} T_n T_{n-1} \dots P_{i,j} T_1 A P_{k,l} \ , \tag{5.39}$$

unde $P_{i,j}$, $P_{k,l}$ reprezintă matricele de permutare, T_s, $s \in \overline{1,n}$ sunt matrice care conţin multiplicatorii, iar $D^{(n)}$ este o matrice diagonală

$$T_1 = L_1,$$

$$T_2 = \begin{bmatrix} 1 & -m_{1,2} & \dots & \dots & 0 \\ 0 & 1 & \dots & \dots & 0 \\ 0 & -m_{3,2} & \ddots & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -m_{n,2} & \dots & \dots & 1 \end{bmatrix}, \ D^{(n)} = \begin{bmatrix} 1/a_{1,1} & 0 & \dots & 0 \\ 0 & 1/a_{2,2}^{(1)} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1/a_{n,n}^{(n-1)} \end{bmatrix}.$$

Dacă aceleaşi transformări la stânga sunt aplicate şi matricei unitate, rezultă o matrice B având expresia

$$B = D^{(n)} T_n T_{n-1} \dots P_{i,j} T_1 I \ . \tag{5.40}$$

Din (5.39) şi (5.40) se obţine

$$I = BAP_{k,l} \ , \tag{5.41}$$

sau, observând că $P_{k,l}^2 = I$ şi înmulţind egalitatea (5.41) la dreapta şi la stînga cu $P_{k,l}$, se obţine

$$P_{k,l} I P_{k,l} = P_{k,l} B A P_{k,l} P_{k,l}, \text{ rezultă } (P_{k,l} B) A = I, \text{ deci } A^{-1} = P_{k,l} B \ , \tag{5.42}$$

adică matricea inversă se obţine din matricea B printr-o *permutare de linii, corespunzătoare permutărilor de coloane* efectuate la căutarea pivoţilor în operaţiile de eliminare. Dacă s-au efectuat mai multe permutări de coloane, spre exemplu

$$I = BAP_{k_1,l_1}P_{k_2,l_2}\dots P_{k_m,l_m} \ , \tag{5.43}$$

rezultă

$$A^{-1} = (P_{k_m,l_m}\dots(P_{k_2,l_2}(P_{k_1,l_1}B))\dots) \ , \tag{5.44}$$

adică permutările de linii în matricea B se efectuează în *ordine inversă* faţă de permutările de coloane efectuate în matricea A. Dacă la căutarea pivoţilor nu se folosesc schimbări de coloane, atunci $A^{-1} = B$.

Lăsând la o parte permutările, numărul de operaţii la inversare este

$$n_{inv} = 3n^3/2 - 2n^2 + 5n/2 + 1 \approx 3n^3/2 - 2n^2, \ (n \gg 1) \ . \tag{5.45}$$

Se observă că numărul de operaţii la inversarea matricelor mari ($n \geq 40$) este de circa 3 ori mai mare decât pentru eliminarea Gauss-Jordan şi de 4.5 ori mai mare decât la eliminarea Gauss. Numărul de operaţii se poate reduce însă ţinând seama de forma particulară a matricei unitate. Într-adevăr, în prima etapă a eliminării, matricea I (plasată în coloanele $n+1$, $2n$) sau orice matrice de permutare $P_{i,j}$ care ar putea apărea în locul ei în urma alegerii pivotului, conţine doar un element nenul pe fiecare linie. În urma eliminării din prima etapă, coloanele $n+1\dots 2n$ vor forma o matrice cu cel mult două elemente nenule pe fiecare linie cu care se va opera în etapa a doua, ş.a.m.d. În etapa s, se va opera cu cel mult s elemente diferite de zero pe linia pivotului, în coloanele $n+1\dots 2n$. Ca urmare, numărul de operaţii poate fi redus la expresia

$$n_{inv1} = n^3 - n^2 + 1 \approx n^3 - n^2 \ , \tag{5.46}$$

deci, aproape de 1.5 ori mai mic faţă de (5.45) în cazul matricelor mari. Această reducere se face pe seama ţinerii evidenţei poziţiilor elementelor nenule din linia pivotului. Această evidenţă nu este dificilă deoarece, la fiecare etapă se adaugă o coloană nouă la precedentele. În final, vom observa că permutările de linii (singurele care se aplică pe poziţia ocupată iniţial de matricea I) schimbă doar poziţiile elementelor nenule (nu şi numărul acestora).

Determinarea matricei inverse am văzut că este echivalentă cu rezolvarea a n sisteme de n ecuaţii simultane (cu aceeaşi matrice A), având ca termeni liberi coloanele matricei I, iar ca necunoscute elementele matricei inverse. Aceeaşi problemă poate fi rezolvată şi prin eliminare gaussiană, retrosubstituirea pentru găsirea celor n^2 necunoscute cerând însă $n^2(n+1)/2$ operaţii. În aceste condiţii, numărul de operaţii pentru inversare este

(a) fără considerarea zerourilor din matricea unitate

$$n_{inv2} = 4n^3/3 - n/3 \ ; \tag{5.47}$$

(b) cu considerarea zerourilor din matricea unitate

$$n_{inv3} = n^3 + n^2/2 - n/2 \ . \tag{5.48}$$

Se observă că $n_{inv2} < n_{inv}$, în timp ce $n_{inv3} > n_{inv1}$.

EXEMPLU. Ne propunem să calculăm inversa matricei A a sistemului (5.11). Vom folosi metoda Gauss-Jordan cu pivotare totală pentru matricea extinsă. Avem succesiv

$$\left[\begin{array}{ccc|ccc} 1 & 0.2222 & 1 & 1 & 0 & 0 \\ 8 & 2 & -3 & 0 & 1 & 0 \\ 1 & 2 & -5 & 0 & 0 & 1 \end{array}\right] \overset{l_1 \leftrightarrow l_2}{\longrightarrow} \left[\begin{array}{ccc|ccc} \boxed{8} & 2 & -3 & 0 & 1 & 0 \\ 1 & 0.2222 & 1 & 1 & 0 & 0 \\ 1 & 2 & -5 & 0 & 0 & 1 \end{array}\right]$$

$$\overset{s=1}{\longrightarrow} \left[\begin{array}{ccc|ccc} 8 & 2 & -3 & 0 & 1 & 0 \\ 0 & -0.0278 & 1.375 & 1 & -0.125 & 0 \\ 0 & 1.75 & -4.625 & 0 & -0.125 & 1 \end{array}\right]$$

$$\begin{array}{c} l_2 \leftrightarrow l_3 \\ \longrightarrow \\ c_2 \leftrightarrow c_3 \end{array} \left[\begin{array}{ccc|ccc} 8 & 2 & -3 & 0 & 1 & 0 \\ 0 & \boxed{\text{-4.625}} & 1.75 & 1 & -0.125 & 1 \\ 0 & 1.375 & -0.0278 & 0 & -0.125 & 0 \end{array}\right]$$

$$\overset{s=2}{\longrightarrow} \left[\begin{array}{ccc|ccc} 8 & 0 & 0.8649 & 0 & 1.081 & -0.6486 \\ 0 & -4.625 & 1.75 & 1 & -0.125 & 1 \\ 0 & 0 & \boxed{0.4925} & 0 & -0.1622 & 0.2973 \end{array}\right]$$

$$\overset{s=3}{\longrightarrow} \left[\begin{array}{ccc|ccc} 8 & 0 & 0 & -1.756 & 1.366 & -1.171 \\ 0 & -4.625 & 0 & -3.553 & 0.4513 & -0.0564 \\ 0 & 0 & 0.4925 & 1 & -0.1622 & 0.2973 \end{array}\right]$$

$$\overset{D}{\longrightarrow} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -0.2195 & 0.171 & -0.1463 \\ 0 & 1 & 0 & 0.7683 & -0.09758 & 0.0122 \\ 0 & 0 & 1 & 2.030 & -0.3293 & 0.06037 \end{array}\right] . \tag{5.49}$$

Calculele au fost efectuate cu 4 cifre semnificative. Deoarece am permutat coloanele 2 şi 3 pentru aducerea valorii -4.625 pe poziţia pivotului, matricea inversă B_0 se obţine din (5.49) prin permutarea liniilor 2 şi 3 în matricea formată cu ultimele trei coloane

$$B_0 = \left[\begin{array}{ccc} -0.2195 & 0.1707 & -0.1463 \\ 2.030 & -0.3293 & 0.6037 \\ 0.7683 & -0.09758 & 0.0122 \end{array}\right] . \tag{5.50}$$

5.1.6 Relaţia Sherman-Morisson

Fie matricea A pentru care am calculat, cu un efort apreciabil, o aproximaţie a inversei, $B \cong A^{-1}$. Să presupunem că în matricea A sunt câteva elemente care suferă mici modificări (se poate modifica numai un element, sau o linie, sau o coloană, etc.). Se pune problema de a folosi munca depusă anterior pentru calculul matricei inverse B. Acest lucru se poate realiza cu formula *Sherman-Morisson*

$$(A + u \cdot v^T)^{-1} = A^{-1} - (A^{-1}u)(vA^{-1})^T/(1 + vA^{-1}u) \ , \tag{5.51}$$

unde u şi v sunt doi vectori cu ajutorul cărora putem exprima schimbarea din matricea A de forma

$$A \to A + u \cdot v^T \ . \tag{5.52}$$

Spre exemplu, dacă se modifică linia i din matricea A, trebuie ca $u_j = 0$, $j \in \overline{1,n} \setminus \{i\}$, $u_i = 1$, iar vectorul v conţine valorile cu care se modifică elementele $a_{i,j}$, $j \in \overline{1,n}$. Analog, pentru a modifica coloana j, $v_i = 0$, $i \in \overline{1,n} \setminus \{j\}$, $v_j = 1$, iar vectorul u conţine valorile cu care se modifică elementele $a_{i,j}$, $i \in \overline{1,n}$.

5.1.7 Rafinarea matricei inverse

Deoarece în cursul calculelor se produc erori de rotunjire, matricea B, obţinută prin oricare din procedeele prezentate, nu este chiar matricea A^{-1}. Fie B_0 inversa matricei A, obţinută iniţial. Putem defini o matrice a erorilor, F_0, prin relaţia

$$F_0 = I - AB_0 \ . \tag{5.53}$$

Evident, dacă B_0 este exactă, atunci $F = 0$. Să presupunem că erorile sunt suficient de mici, astfel încât

$$\|F_0\| \ll 1 \ , \tag{5.54}$$

unde $\|F_0\|$ reprezintă o normă oarecare a matricei F_0 (vezi şi paragraful 5.4). Mai departe, vom construi un şir de iteraţii astfel încât

$$F_{k+1} = I - AB_{k+1},\ F_{k+1} = F_k^2,\ k = 0, 1, 2, \ldots \tag{5.55}$$

Ţinând cont de (5.54), atunci $F_k \to 0$, deoarece

$$\|F_k\| = \|F_0^{2^k}\| \le \|F_0\|^{2^k} \to 0, \text{ pentru } k \to \infty \ . \tag{5.56}$$

Din (5.55) se deduc succesiv

$$\begin{gathered} F_{k+1} = F_k F_k = (I - AB_k)F_k = F_k - AB_k F_k = \\ = I - AB_k - AB_k F_k = I - A(B_k + B_k F_k) = I - AB_{k+1} \ , \end{gathered} \tag{5.57}$$

sau comparând ultimele două egalităţi (5.57)

$$B_{k+1} = B_k + B_k F_k = B_k(I + F_k) = 2B_k - B_k A B_k,\ k = 0, 1, 2, \ldots \tag{5.58}$$

Relaţia (5.58) arată cum trebuie construite inversele cu o precizie din ce în ce mai bună, dacă este realizată condiţia (5.54). Din relaţia (5.55) se poate scrie

$$A^{-1}F_{k+1} = A^{-1} - B_{k+1} \ . \tag{5.59}$$

Rezultă eroarea relativă la inversare, în normă

$$\|\delta A^{-1}\|_{k+1} / \|A^{-1}\| = \|A^{-1} - B_{k+1}\| / \|A^{-1}\| \le \|F_{k+1}\| \ . \tag{5.60}$$

Odată găsită o inversă suficient de exactă, se poate obţine şi o soluţie rafinată a sistemului (5.2)

$$x = A^{-1}b = B_{k+1}b,\ k \geq 1\ . \tag{5.61}$$

Deoarece procesul de rafinare conţine înmulţiri de matrice care necesită circa n^3 operaţii, este de dorit ca $\|F_0\|$ să fie cât mai mică în raport cu unitatea, adică inversarea iniţială să fie cât mai bună.

EXEMPLU. Matricea erorilor pentru matricea inversă B_0 dată de (5.50) este

$$F_0 = 10^{-5} \cdot \begin{bmatrix} 13.4 & 7.046 & -4.214 \\ 90 & 20 & -40 \\ 100 & -10 & -10 \end{bmatrix}\ , \tag{5.62}$$

cu norma

$$\|F_0\|_\infty = 10^{-5}(90 + 20 + 40) = 1.5 \cdot 10^{-3} \ll 1\ .$$

Deoarece $\|F_0\|_\infty < 1$, putem rafina inversa B_0 folosind (5.58), unde apare corecţia

$$B_0F_0 = 10^{-5} \cdot \begin{bmatrix} -2.208 & 3.33 & -4.44 \\ 57.94 & 1.68 & -1.419 \\ 2.731 & 3.339 & 0.544 \end{bmatrix}\ .$$

Se obţine matricea rafinată B_1 şi matricea erorilor F_1 date de

$$B_1 = \begin{bmatrix} -0.2195 & 0.171 & -0.1463 \\ 2.0306 & -0.3293 & 0.6037 \\ 0.7683 & -0.09758 & 0.0122 \end{bmatrix},\ F_1 = 10^{-8} \begin{bmatrix} 3.923 & 2.775 & -2.962 \\ -9.94 & 14.34 & -7.793 \\ -5.6 & -6.046 & 0.786 \end{bmatrix}. \tag{5.63}$$

Folosind inversele B_0 şi B_1 pentru rezolvarea sistemului (5.11), se obţin soluţiile

$$\bar{x}^{(0)} = B_0b = (-0.756,\ 4.993,\ 1.6464)^T,\ \bar{x}^{(1)} = B_1b = (-0.7561,\ 4.9941,\ 1.6464)^T\ ,$$

cu vectorii eroare daţi de

$$e^{(0)} = 10^{-5} \cdot (-9.756,\ 90.024,\ -5.854)^T,\ e^{(1)} = 10^{-5} \cdot (0.244,\ -19.76,\ -5.854)^T\ .$$

Eroarea relativă la inversare dată de relaţia (5.60) în norma $\|\cdot\|_\infty$ este pentru matricea nerafinată B_0 dată de (5.50)

$$\|\delta_0 A^{-1}\|_\infty / \|A^{-1}\|_\infty \leq \|F_0\|_\infty = 1.5 \cdot 10^{-3}\ ,$$

iar pentru matricea rafinată B_1 dată de (5.63) este

$$\|\delta_1 A^{-1}\|_\infty / \|A^{-1}\|_\infty \leq \|F_1\|_\infty = 0.32 \cdot 10^{-6}\ .$$

5.1.8 Efectele erorilor din datele iniţiale

Dacă în datele iniţiale ale sistemului (5.1) există erori, acestea pot fi amplificate în calcul mai mult sau mai puţin, în funcţie de numărul de condiţionare al matricei sistemului. *Numărul de condiţionare* (sau *indicele de condiţionare*) al unei matrice A, nesingulare, este notat cu $\mu(A)$ şi are, prin definiţie, expresia

$$\mu(A) = \|A\| \cdot \|A^{-1}\| \geq 1 \ . \tag{5.64}$$

Cu cât $\mu(A)$ este mai mare, cu atât matricea este mai rău condiţionată, calculul matricei inverse fiind mai dificil de efectuat. De aceea, este mai utilă definiţia (5.85) pe care o vom da în paragraful 5.1.10.

Exemplu. Numărul de condiţionare al matricei A din exemplul (5.11) calculat cu relaţia (5.64), folosind norma $\|\cdot\|_\infty$ şi cu matricea inversă nerafinată B_0 dată de (5.50) este

$$\mu_\infty(A) = \|A\|_\infty \|B_0\|_\infty = 13 \cdot 2.962 = 38.5 \ ,$$

adică amplificarea erorilor este moderată.

Să presupunem că, în matricea iniţială A, s-a introdus perturbaţia δA astfel încât eroarea relativă, în normă, satisface inegalitatea

$$\|\delta A\| \cdot \|A^{-1}\| < 1 \ . \tag{5.65}$$

Să presupunem că şi vectorul b este dat cu eroarea δb. În consecinţă, va rezulta o imprecizie δx în soluţie, astfel încât

$$(A + \delta A)(x + \delta x) = b + \delta b \ , \tag{5.66}$$

sau, ţinând cont de (5.1)

$$\delta A \cdot x + (A + \delta A)\delta x = \delta b \ . \tag{5.67}$$

Din (5.67) se obţine succesiv

$$\begin{aligned} \delta x = (A + \delta A)^{-1}(\delta b - \delta A\, x) = [A(I + A^{-1}\delta A)]^{-1}(\delta b - \delta A\, x) = \\ = (I + A^{-1}\delta A)^{-1} A^{-1}(\delta b - \delta A\, x) \ , \end{aligned} \tag{5.68}$$

Inegalitatea (5.65) implică însă

$$\|(I + A^{-1}\delta A)^{-1}\| \leq (1 - \|A^{-1}\delta A\|)^{-1} \leq (1 - \|A^{-1}\| \cdot \|\delta A\|)^{-1} \ . \tag{5.69}$$

iar din (5.1) se obţine

$$\|x\|^{-1} \leq \|A\| \cdot \|b\|^{-1} \ . \tag{5.70}$$

Utilizând (5.68), rezultă

$$\|\delta x\| \leq (1 - \|A^{-1}\|/\|\delta A\|)^{-1} \|A^{-1}\|(\|\delta b\| + \|\delta A\| \cdot \|x\|) \tag{5.71}$$

sau, împărţind cu $\|x\| \neq 0$ şi utilizând (5.70)

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\mu(A)}{1-\mu(A)\|\delta A\|/\|A\|}\left(\frac{\|\delta b\|}{\|b\|}+\frac{\|\delta A\|}{\|A\|}\right) . \tag{5.72}$$

Semnul "egal" în (5.72) corespunde matricelor hermitice, cu norma spectrală. Se observă importanţa numărului de condiţionare $\mu(A)$ în amplificarea erorilor iniţiale. Legat de eroarea la inversarea matricei A, se stabileşte o expresie similară cu (5.72)

$$\frac{\|(A+\delta A)^{-1}-A^{-1}\|}{\|A^{-1}\|} \leq \frac{\mu(A)}{1-\mu(A)\|\delta A\|/\|A\|}\frac{\|\delta A\|}{\|A\|} . \tag{5.73}$$

5.1.9 Factorizarea $L \cdot U$

După cum am văzul la paragraful 5.1.4, metoda eliminării gaussiane fără permutări de linii sau coloane este echivalentă cu descompunerea matricei A într-un produs dintre o matrice superior triunghiulară L şi una inferior triunghiulară U

$$A = L \cdot U , \tag{5.74}$$

adică

$$A = \begin{bmatrix} l_{1,1} & 0 & \dots & 0 \\ l_{2,1} & l_{2,2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n,1} & l_{n,2} & \dots & l_{n,n} \end{bmatrix} \cdot \begin{bmatrix} u_{1,1} & u_{1,2} & \dots & u_{1,n} \\ 0 & u_{2,2} & \dots & u_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & u_{n,n} \end{bmatrix} . \tag{5.75}$$

Putem folosi descompunerea (5.74) pentru a rezolva sistemul (5.2) ţinând cont de asociativitatea produsului de matrice

$$Ax = (LU)x = L(Ux) = b , \tag{5.76}$$

rezolvând pe rând sistemele

$$Lz = b,\ Ux = z . \tag{5.77}$$

Sistemele (5.77) se rezovă cu uşurinţă datorită faptului că matricele L şi U sunt triunghiulare. Relaţia (5.75) reprezintă un sistem de n^2 ecuaţii cu n^2+n necunoscute, deci trebuie precizate n dintre necunoscute. Cele mai utilizate alegeri sunt date de *factorizarea Doolittle* când se aleg $l_{i,i}=1$, *factorizarea Crout* când se aleg $u_{i,i}=1$ şi *factorizarea Cholesky* când $U=L^T$, deci $l_{i,i}=u_{i,i}$. Factorizarea Cholesky este posibilă numai dacă matricea A este simetrică şi pozitiv definită[5].

[5] O matrice se numeşte *pozitiv definită* dacă pentru orice vector $x \neq 0$, avem $x^T Ax > 0$.

Algoritmul de calcul pentru factorizarea LU se determină relativ uşor dezvoltând relaţia (5.75) şi este dat de (5.78)-stânga.

$$
\begin{array}{l}
k = 1 : n \\
\left[\begin{array}{l}
\text{[Se alege } u_{k,k} \text{ sau } l_{k,k}, \\
\text{se determină } l_{k,k} \text{ sau } u_{k,k} \text{]} \\
l_{k,k}u_{k,k} \leftarrow a_{k,k} - \sum_{s=1}^{k-1} l_{k,s}u_{s,k} \\
i = (k+1) : n \\
\left[\begin{array}{l}
j \leftarrow i \\
u_{k,j} \leftarrow \dfrac{(a_{k,j} - \sum_{s=1}^{k-1} l_{k,s}u_{s,j})}{l_{k,k}} \\
l_{i,k} \leftarrow \dfrac{(a_{i,k} - \sum_{s=1}^{k-1} l_{i,s}u_{s,k})}{u_{k,k}}
\end{array} \right.
\end{array} \right.
\end{array}
\qquad
\begin{array}{l}
k = 1 : n \\
\left[\begin{array}{l}
\text{[Se alege } l_{k,k}, \text{ se determină} \\
u_{k,k} \text{ care se păstrează în} \\
\text{locaţia } a_{k,k}\text{]} \\
a_{k,k} \leftarrow \dfrac{(a_{k,k} - \sum_{s=1}^{k-1} a_{k,s}a_{s,k})}{l_{k,k}} \\
i = (k+1) : n \\
\left[\begin{array}{l}
j \leftarrow i \\
a_{k,j} \leftarrow \dfrac{(a_{k,j} - \sum_{s=1}^{k-1} a_{k,s}a_{s,j})}{l_{k,k}} \\
a_{i,k} \leftarrow \dfrac{(a_{i,k} - \sum_{s=1}^{k-1} a_{i,s}a_{s,k})}{a_{k,k}}
\end{array} \right.
\end{array} \right.
\end{array}
\tag{5.78}
$$

În partea dreaptă este dat algoritmul în care se aleg valorile $l_{k,k}$, $k \in \overline{1,n}$, care se memorează separat, iar elementele $l_{i,j}$, $i \in \overline{2,n}$, $j \in \overline{1,i-1}$, $u_{m,p}$, $m \in \overline{1,n}$, $p \in \overline{m,n}$, se plasează pe locul elementelor matricei A.

Din (5.78) observă că pentru a putea obţine factorizarea LU este necesar ca $l_{k,k}$ şi $u_{k,k}$ să fie nenule. Factorizarea LU poate fi implementată eficient folosind tehnica pivotului parţial, caz în care $PA = LU$, iar soluţia se obţine rezolvând sistemul echivalent $LU = Pb$, unde P este matricea care conţine toate permutările de linii efectate asupra matricei A pe parcursul procesului de factorizare. Condiţiile în care o matrice admite o factorizare LU sunt incluse în următoarele teoreme pe care le dăm fără demonstraţie:

Teoremă. Dacă minorii principali ai matricei A sunt nenuli, atunci aceasta admite o factorizare LU.

Teoremă. Dacă A este o matrice reală, simetrică şi pozitiv definită, atunci ea admite o factorizare LU unică, în care $U = L^T$.

În cazul în care A este o matrice reală, simetrică şi pozitiv definită, descompunerea $A = LL^T$ se numeşte *factorizarea Cholesky* şi este dată de algoritmul (5.79).

$$
\begin{array}{l}
k = 1 : n \\
\left[\begin{array}{l}
l_{k,k} \leftarrow (a_{k,k} - \sum_{s=1}^{k-1} l_{k,s}^2)^{1/2} \\
i = (k+1) : n \\
\left[l_{i,k} \leftarrow (a_{i,k} - \sum_{s=1}^{k-1} l_{i,s}l_{k,s})/l_{k,k} \quad . \right.
\end{array} \right.
\end{array}
\tag{5.79}
$$

5.1.10 Descompunerea SVD

În unele cazuri, metodele prezentate anterior eşuează datorită matricei sistemului care are determinantul foarte mic ca valoare absolută. Altfel spus, una din liniile sistemului este *aproape o combinaţie liniară* a celorlalte linii[6]. În aceste situaţii se recomandă descompunerea SVD[7] pe care o prezentăm în acest paragraf. Dăm fără demostraţie următoarea teoremă:

Teoremă Oricare ar fi matricea A de m linii şi n coloane, cu $m \geq n$, ea poate fi descompusă într-un produs de trei matrice

$$A = U \cdot W \cdot V^T \ , \tag{5.80}$$

unde U este o matrice ortogonală $m \times n$, W este o matrice diagonală $n \times n$, iar V este o matrice ortogonală $n \times n$.

Matricele U şi V au coloanele ortonormate, adică

$$U^T U = V^T V = I \ , \tag{5.81}$$

unde I este matricea unitate $n \times n$. Important este faptul că descompunerea (5.80) poate fi făcută şi în cazul în care matricea A este singulară.

În cazul în care matricea A este pătrată ($m = n$), sistemul (5.2) se rezolvă uşor, deoarece inversele matricelor U, V şi W se calculează imediat ţinând cont de (5.81) şi deci

$$A^{-1} = V \cdot diag(1/w_i) \cdot U^T \ , \tag{5.82}$$

iar soluţia sistemului (5.2) este

$$x = V \cdot [diag(1/w_i)] \cdot (U^T \cdot b) \ . \tag{5.83}$$

Dacă matricea A este singulară, unul sau mai multe din elementele w_i sunt nule, iar sistemul (5.2) este nedeterminat şi poate fi compatibil sau incompatibil după cum rangul matricei A este sau nu egal cu rangul matricei extinse A'. Dacă sistemul (5.2) este compatibil, se poate arăta că o soluţie particulară poate fi obţinută prin înlocuirea elementelor infinite din (5.83) cu zero (adică în loc de $1/w_i = \infty$ vom pune $w_i = 0$!). Dacă procedăm în acelaşi fel în cazul în care sistemul (5.83) este incompatibil, se poate demonstra că soluţia obţinută minimizează reziduul

$$R = \|Ax - b\|_2 \ , \tag{5.84}$$

adică vectorul x satisface cel mai bine (5.2) în sensul celor mai mici pătrate.

În cazul în care matricea A este aproape singulară, unul sau mai multe elemente w_i au valori foarte mici în modul. Spunem în acest caz că matricea

[6] Spunem în acest caz că matricea A este *aproape singulară*.

[7] Iniţialele SVD provin de la cuvintele din limba engleză *Singular Value Decomposition*.

A este *rău condiţionată*. *Numărul de condiţionare* al matricei A poate fi definit de raportul (vezi şi (5.64))

$$\mu(A) = w_{\max}/w_{\min} \ , \tag{5.85}$$

unde

$$w_{\max} = \max|w_i|, \ w_{\min} = \min|w_i| \ . \tag{5.86}$$

Cu cât numărul de condiţionare este mai mare, cu atât matricea A este mai rău condiţionată (vezi paragraful 5.1.8), iar una sau mai multe ecuaţii sunt foarte aproape de nişte combinaţii liniare ale celorlate ecuaţii. Metodele de eliminare, chiar şi cu pivotare totală, pot eşua în astfel de cazuri. Folosind descompunerea (5.80), soluţia obţinută (5.83) este efectată de prezenţa elemetelor w_i foarte mici în modul. O modalitate de a evita astfel de probleme constă în renunţarea la ecuaţiile care sunt foarte aproape de nişte combinaţii liniare ale celorlate ecuaţii (şi deci care nu aduc informaţii suplimentare), ceea ce ce se poate realiza prin înlocuirea cu zero a elementelor $1/w_i$ foarte mari în modul. Se obţin astfel soluţii care au reziduul R dat de (5.84) mai mic decât cel dat de soluţia obţinută cu valorile $1/w_i$ ce rezultă din descompunerea (5.80).

Pentru cazul în care numărul de ecuaţii este mai mic decât numărul de necunoscute ($m < n$), se poate obţine o soluţie particulară adăugând $n - m$ linii de zerouri în matricea A obţinând o matrice pătrată $n \times n$, efectuând apoi descompunerea (5.80) şi înlocuind cu zero termenii $1/w_i$ ce corespund valorilor w_i mici în modul.

Tehnica descompunerii (5.80) are implementări puternice în toate bibliotecile matematice de rezolvare a sistemelor de ecuaţii liniare. Algoritmii folosiţi sunt relativ sofisticaţi, depăşind cadrul acestei lucrări, fiind folosiţi în cazurile dificile ale sistemelor cu matrice mari şi rău condiţionate.

5.1.11 Sisteme cu matrice rare

În practică apar deseori sisteme mari a căror matrice prezintă foarte multe elemente nule. Astfel de matrice se numesc *matrice rare*. Aplicarea metodelor de eliminare în forma standard devine neeconomică, atât din punct de vedere al timpului de calcul cât şi din punct de vedere al memoriei utilizate. Se utilizează în acest scop tehnici speciale de *memorare în formă compactă* a elementelor nenule ale matricei, precum şi *metode de ordonare* care urmăresc, prin intermediul unor secvenţe de pivotare, minimizarea numărului de elemente nenule nou create în procesul de eliminare. Ne propunem descrierea sumară a principiilor care stau la baza lucrului cu matrice rare, o prezentare mai amănunţită depăşind cadrul acestei cărţi.

Majoritatea metodelor utilizate pentru rezolvarea sistemelor cu matrice rare sunt metode de eliminare de tip Gauss sau metode de factorizare LU, la care se acordă o atenţie deosebită elementelor nule în scopul micşorării volumului de calcul şi a memoriei utilizate. Aplicarea eficientă şi cu succes a metodelor directe în rezolvarea sistemelor cu matrice rare depinde de cât de "ordonată" (simplă)

Figura 5.1: Forme speciale de matrice rare.

este structura matricei. În bibliotecile matematice sunt implementate tehnici de analiză a structurii matricei şi de transformare a acesteia într-o structură suficient de ordonată. Aceste prelucrări au ca rezultat şi o creştere a numărului de condiţionare a matricei, efect nedorit care se poate ameliora prin echilibrarea (scalarea) matricei, tehnică descrisă la paragraful 5.1.3.

În figura 5.1 sunt reprezentate principalele forme speciale de matrice rare. Aceste forme speciale au avantajul că permit evitarea apariţiei elementelor nenule în afara zonelor haşurate.

Sistemele cu matrice bandă rezultă în general din discretizarea ecuaţiilor cu derivate parţiale. Un caz simplu al matricelor de tip bandă a fost descris la interpolarea cu funcţii spline cubice, paragraful 1.1.2. Pentru sistemele cu matrice bandă, metodele directe de eliminare sau factorizare se dovedesc în general eficiente, cu excepţia cazurilor în care dimensiunea sistemului este foarte mare, când se recomandă metodele iterative sau combinaţii între metode directe şi iterative.

Problema principală care se pune în cazul matricei bandă este memorarea cât mai eficientă a acesteia. Cea mai simplă schemă de memorare a unei matrice simetrice cu lăţimea benzii egală cu $2b-1$ (figura 5.1) cu elementele $a_{i,j}=0$ pentru $|i-j|\geq b$, constă în memorarea doar a elementelor de pe diagonala principală şi a celor de sub ea ($j\leq i$), având nevoie de un vector v de dimensiune $n+(n-1)+\ldots+(n-b+1)=b(2n-b+1)/2$. În cazul în care memorăm matricea pe linii, elementul $a_{i,j}$ se va găsi în vectorul v pe poziţia $k=[1+2+\ldots+(i-1)]+j=i(i-1)/2+j$ dacă $i\leq b$. Dacă $i>b$, vom avea $1+2+\ldots+b=b(b+1)/2$ elemente pe primele b linii, apoi $b(i-b-1)$ elemente

până la linia $i-1$ inclusiv şi $j-(i-b)$ elemente pe linia i, deci elementul $a_{i,j}$ se va găsi pe poziţia $(b-1)i-b(b-1)/2+j$. În cazul matricelor mari care conţin multe elemente nule în interiorul benzii, se pot folosi scheme adecvate care să memoreze numai elementele nenule.

Pentru memorarea unei matrice inferior triunghiulară este nevoie de un vector v de dimensiune $1+2+\ldots+n=n(n+1)/2$ elemente. Dacă matricea este păstrată pe linii, elementele ei sunt memorate în ordinea $a_{1,1}$, $a_{2,1}$, $a_{2,2}$, $a_{3,1}$, $a_{3,2}$, $a_{3,3}$, $a_{4,1}\ldots$, elementul $a_{i,j}$ având în vectorul v poziţia $i(i-1)/2+j$.

Pentru memorarea unei matrice rare oarecare există mai multe tehnici dintre care amintim: memorare prin identificare binară, memorare compactă aleatoare, memorare compactă sistematică, memorare prin utilizarea adreselor înlănţuite, ş.a. De regulă, o tehnică de memorare care minimizează capacitatea de memorie utilizată plăteşte preţul unui timp mai mare de calcul şi viceversa. Pentru exemplificare, prezentăm pe scurt două tehnici: *memorare prin identificare binară* şi *memorare compactă sistematică*. Vom folosi ca exemplu matricea

$$A=\begin{bmatrix} a_{1,1} & 0 & a_{1,3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{3,3} & 0 & a_{3,5} \\ a_{4,1} & 0 & 0 & a_{4,4} & 0 \end{bmatrix}. \tag{5.87}$$

Matricea se memorează folosind două zone: o *zonă primară* în care sunt memorate elementele nenule ale matricei A şi o *zonă secundară* în care sunt păstrate informaţii necesare găsirii rapide a acestora. Ambele tehnici folosesc o zonă primară reprezentată de vectorul

$$v=(a_{1,1},a_{1,3},a_{3,3},a_{3,5},a_{4,1},a_{4,4}). \tag{5.88}$$

În cazul *memorării prin identificare binară*, zona secundară se bazează pe natura binară a sistemului de calcul. Pentru fiecare element $a_{i,j}$ al matricei A se memorează o cifră binară: 0 dacă $a_{i,j}=0$ şi 1 dacă $a_{i,j}\neq 0$. Pentru matricea (5.87), zona secundară va fi dată de succesiunea de cifre binare

$$1\quad 0\quad 1\quad 0\quad 0\quad 0\quad 0\quad 0\quad 0\quad 0\quad 0\quad 0\quad 1\quad 0\quad 1\quad 1\quad 0\quad 0\quad 1\quad 0\ .$$

Această schemă necesită calcule complicate (care să permită accesul la bit) pentru identificarea poziţiei elementelor matricei.

În cazul *memorării compacte sistematice*, avem două zone secundare, prima în care se memorează indicii de coloană c_k al elementului v_k, iar cea de a doua în care se memorează indicele de coloană f_i al primului element nenul din fiecare linie. Pentru matricea (5.87) memorată în vectorul (5.88), cele două zone secundare sunt

$$c=(1,3,3,5,1,4),\ f=(1,3,3,1,3)\ . \tag{5.89}$$

Numărul de elemente nenule de pe linia i este dat de $f_{i+1}-f_i$. Avem deci $3-1=2$ elemente nenule pe prima linie, $3-3=0$ elemente nenule pe linia a doua, ş.a.m.d. Schema prezentată nu este potrivită extragerii informaţiilor asupra coloanelor matricei A, dar se poate concepe o schemă în acest sens.

5.2 Metode iterative

Metodele de eliminare comportă un număr de operaţii care creşte la sistemele mari, cu cubul numărului de ecuaţii din sistem, $O(n^3)$. Operaţii suplimentare sunt cerute de căutarea pivoţilor, în vederea reducerii erorilor de rotunjire. Metodele iterative pot conduce la un număr mai mic de operaţii pe iteraţie. Dacă procedeul iterativ este suficient de rapid convergent se poate obţine o economie importantă de timp de calcul. În plus, metodele iterative permit menţinerea structurii iniţiale a unei matrice rare, rezultând un consum redus de memorie.

Principiul general al metodelor iterative poate fi prezentat prin analogie cu metoda iteraţiei simple (vezi 3.2.1) de rezolvare a ecuaţiei $F(x) = 0$, în care ecuaţia originală este transcrisă ca

$$x = f(x)\,, \tag{5.90}$$

ce conduce la procedeul iterativ

$$x_{k+1} = f(x_k)\,. \tag{5.91}$$

În cazul sistemelor liniare, $Ax = b$ vom forţa o descompunere a matricei A

$$A = A + B - B \tag{5.92}$$

şi vom scrie forma echivalentă a sistemului liniar iniţial

$$Bx = c \equiv (B - A)x + b\,. \tag{5.93}$$

Analog iteraţiei simple, vom considera relaţia (5.93) ca o procedură de obţinere a unei aproximaţii îmbunătăţite a soluţiei sistemului $x^{(k+1)}$, dată fiind o aproximaţie iniţială $x^{(k)}$,

$$Bx^{(k+1)} = (B - A)x^{(k)} + b\,. \tag{5.94}$$

Descompunerea (5.92) se alege astfel încât:

1. şirul aproximaţiilor $\{x^{(k)}\}_{k\in\mathbf{N}}$ să fie cât mai rapid convergent către soluţia sistemului $Ax = b$;
2. sistemul $Bx = c$ să fie uşor de rezolvat.

5.2.1 Metoda iterativă Jacobi

Metoda iterativă Jacobi foloseşte descompunerea

$$A = D - C\,, \tag{5.95}$$

unde D este o matrice diagonală, ce trebuie să fie nesingulară $\det D \neq 0$. Sistemul $Ax = b$ devine

$$Dx = Cx + b \Rightarrow x = D^{-1}C\,x + D^{-1}b, \tag{5.96}$$

şi rezultă procesul iterativ

$$x^{(k+1)} = Px^{(k)} + d; \quad P \equiv D^{-1}C; \quad d \equiv D^{-1}b; \quad k = 0, 1, \dots , \tag{5.97}$$

pentru pornirea căruia este nevoie de specificare unei aproximaţii iniţiale $x^{(0)}$. Matricea P se numeşte *matricea rezolvantă* a sistemului.

Matricea D poate fi chiar diagonala principală a matricei A, în care caz exprimarea pe componente a procedeului este

$$x_i^{(k+1)} = (b_i - \sum_{j=1}^{i-1} a_{i,j}x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j}x_j^{(k)})/a_{i,i},\ i = \overline{1,n}\,, \tag{5.98}$$

cu $a_{i,i} \neq 0$. O generalizare a procedeului constă în separarea elementelor de pe diagonala principală $a_{i,i} = a'_{i,i} + a''_{i,i}$, $a'_{i,i} \neq 0$, rezultând procedeul

$$x_i^{(k+1)} = (b_i - a''_{i,i}x_i^{(k)} - \sum_{j=1}^{i-1} a_{i,j}x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j}x_j^{(k)})/a'_{i,i},\ i = \overline{1,n}\,. \tag{5.99}$$

Va trebui să stabilim în ce condiţii şirurile (5.98) sau (5.99) converg spre soluţia sistemului $Ax = b$. Pentru a determina o condiţie suficientă de convergenţă, vom nota cu x soluţia exactă şi cu $e^{(k)}$ eroarea în aproximaţia k

$$e^{(k)} = x - x^{(k)}\,. \tag{5.100}$$

Făcând diferenţa ecuaţiilor (5.96) şi (5.97), se obţine imediat

$$e^{(k+1)} = Pe^{(k)} \tag{5.101}$$

sau, trecând la norme şi notând cu $e^{(0)}$ eroarea iniţială

$$\left\|e^{(k+1)}\right\| \leq \|P\| \cdot \left\|e^{(k)}\right\| \leq \|P\|^2 \cdot \left\|e^{(k-1)}\right\| \leq \ldots \leq \|P\|^k \cdot \left\|e^{(0)}\right\|\,. \tag{5.102}$$

În consecinţă, o *condiţie suficientă* de convergenţă a procedeului iterativ Jacobi este ca norma matricei rezolvante să fie subunitară

$$\|P\| \leq 1\,. \tag{5.103}$$

Desigur, matricea rezolvantă P depinde de matricea D extrasă din A. Este avantajos, prin urmare, să facem descompunerea astfel încât $\|P\|$ să fie cât mai mică. În mod uzual, se extrage întreaga diagonală a matricei A, adică se utilizează şirul (5.98). În acest caz, îndeplinirea oricăreia dintre condiţiile următoare este suficientă pentru a asigura convergenţa procedeului Jacobi:

(a) *diagonal-dominanţa matricei A pe linii*

$$|a_{i,i}| > \sum_{j=1}^{i-1} |a_{i,j}| + \sum_{j=i+1}^{n} |a_{i,j}|\,, \quad i = \overline{1,n}\,; \tag{5.104}$$

(b) *diagonal-dominanţa matricei A pe coloane*

$$|a_{j,j}| > \sum_{i=1}^{j-1} |a_{i,j}| + \sum_{i=j+1}^{n} |a_{i,j}|\,, \quad j = \overline{1,n}\,; \tag{5.105}$$

(c) satisfacerea relaţiei

$$\sum_{i=1}^{n} \left[\sum_{j=1}^{i-1} (a_{i,j}/a_{i,i})^2 + \sum_{j=I+1}^{n} (a_{i,j}/a_{i,i})^2 \right] < 1\,. \tag{5.106}$$

Condiţiile (5.104-5.106) se obţin utilizând respectiv norma maximă, norma 1 şi norma euclidiană pentru matricea rezolvantă P (vezi paragraful 5.4).

Procedeul Jacobi (5.98) se aplică prin algoritmul

$i = 1 : n$
  dacă $|a_{i,i}| < \delta$ atunci *Termen diagonal practic nul*; stop.
  $m \leftarrow 1/a_{i,i}$; $\quad b_i \leftarrow m\, b_i$
  $j = 1 : n$
    [dacă $i \neq j$ atunci $a_{i,j} \leftarrow m\, a_{i,j}$
$xvechi \leftarrow x_0$; $\quad k \leftarrow 0$;
repetă
  $i = 1 : n$
    $s \leftarrow b_i$
    $j = 1 : n$
      [dacă $i \neq j$ atunci $s \leftarrow s - a_{i,j}\, xvechi_j$
    $xnou_i \leftarrow s$
  $AtinsPrecizie \leftarrow \|xnou - xvechi\| \leq \varepsilon\,(1 + \|xnou\|)$
  $xvechi \leftarrow xnou$; $\quad k \leftarrow k + 1$
până când *AtinsPrecizie* sau $k > k_{\max}$ (5.107)

Mărimea δ este un număr mic, tipic epsilon maşină, ce stabileşte un prag pentru elementele diagonale. Acolo unde se efectuează operaţii identice asupra tuturor elementelor unui vector nu vom mai transcrie explicit bucla de parcugere a indicilor. Spre exemplu, prin $xvechi \leftarrow xnou$ în algoritmul de mai sus se subînţelege

$$j = 1 : n \quad [xvechi_j \leftarrow xnou_j\ .\ . \tag{5.108}$$

5.2.2 Metoda iterativă Gauss-Seidel

În această metodă, ideea constă în a folosi noile valori ale componentelor vectorului necunoscutelor $x^{(k+1)}$ imediat ce au fost calculate. Şirul de iteraţii (5.98) se înlocuieşte cu

$$x_i^{(k+1)} = (b_i - \sum_{j=1}^{i-1} a_{i,j} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{i,j} x_j^{(k)})/a_{i,i},\ i = \overline{1,n}. \tag{5.109}$$

Relaţia (5.109) corespunde la descompunerea matricei A într-o matrice diagonală, nesingulară D, o matrice strict inferior triunghiulară T şi o matrice strict superior triunghiulară S

$$A = D - T - S\,. \tag{5.110}$$

Matricele T, S au elementele diagonale nule. Procedeul iterativ este

$$Dx^{(k+1)} = b + Tx^{(k+1)} + Sx^{(k)} \Rightarrow x^{(k+1)} = Px^{(k)} + d\,, \tag{5.111}$$

cu $P \equiv (D-T)^{-1}S$, $d \equiv (D-T)^{-1}b$. Condiţia de convergenţă este, ca şi la metoda Jacobi, ca norma matricei rezolvante P să fie subunitară

$$\|P\| < 1 \Rightarrow \left\|(D-T)^{-1}S\right\| < 1\,. \tag{5.112}$$

Evaluarea inversei $(D-T)^{-1}$ este costisitoare, necesitând $O(n^3/6)$ operaţii, astfel încât se caută condiţii mai simple de convergenţă a procedeului Gauss-Seidel. Una dintre ele este: *dacă matricea A este diagonal-dominantă pe linii, (5.104), atunci metoda Gauss-Seidel converge cel puţin tot atât de rapid ca metoda Jacobi.* Pentru transcrierea acestei condiţii vom folosi următoarele notaţii

$$p_i \equiv \sum_{j=1}^{i-1} |a_{i,j}/a_{i,i}|\,, \quad q_i \equiv \sum_{j=i+1}^{n} |a_{i,j}/a_{i,i}|\,. \tag{5.113}$$

Condiţia de diagonal-dominanţă pe linii a matricei A revine la a scrie

$$r_i \equiv p_i + q_i < 1, \quad i = \overline{1,n}\,. \tag{5.114}$$

Să demonstrăm enunţul anterior în aceste condiţii. Sistemul de ecuaţii $Ax = b$ se scrie sub forma

$$x_i = (b_i - \sum_{j=1}^{i-1} a_{i,j}x_j - \sum_{j=i+1}^{n} a_{i,j}x_j)/a_{i,i},\ i = \overline{1,n} \tag{5.115}$$

cu x_i valorile exacte ale necunoscutelor şi $a_{i,i}$ presupuşi nenuli ($a_{i,i} \neq 0$). Făcând scăderea membru cu membru a sistemelor de egalităţi (5.115) şi (5.109), se obţine

$$x_i - x_i^{(k+1)} = -\sum_{j=1}^{i-1} \frac{a_{i,j}}{a_{i,i}}(x_j - x_j^{(k+1)}) - \sum_{j=i+1}^{n} \frac{a_{i,j}}{a_{i,i}}(x_j - x_j^{(k)})\,, \tag{5.116}$$

sau, observând că în (5.116) au apărut componentele vectorului eroare absolută şi trecând la module

$$\left|e_i^{(k+1)}\right| \leq \sum_{j=1}^{i-1} \left|\frac{a_{i,j}}{a_{i,i}} e_j^{(k+1)}\right| + \sum_{j=i+1}^{n} \left|\frac{a_{i,j}}{a_{i,i}} e_j^{(k)}\right| \tag{5.117}$$

Inegalitatea poate fi majorată în continuare, introducând norma maximă a vectorului eroare,

$$\left\|e^{(k)}\right\|_\infty = \max_j \left|e_j^{(k)}\right| . \tag{5.118}$$

În acest fel, utilizând şi notaţiile (5.113), se obţine

$$\left|e_i^{(k+1)}\right| \le p_i \left\|e^{(k+1)}\right\|_\infty + q_i \left\|e^{(k)}\right\|_\infty . \tag{5.119}$$

Inegalitatea (5.119), fiind adevărată pentru orice $i = \overline{1,n}$, este adevărată şi pentru acel $i = i^*$ pentru care $|e_i^{(k+1)}|$ este maxim. Deci:

$$\left\|e^{(k+1)}\right\|_\infty \le p_i \left\|e^{(k+1)}\right\|_\infty + q_i \left\|e^{(k)}\right\|_\infty \Rightarrow \tag{5.120}$$

$$\left\|e^{(k+1)}\right\|_\infty \le \frac{q_i^*}{1-p_i^*} \left\|e^{(k)}\right\|_\infty . \tag{5.121}$$

Pe de altă parte, utilizând condiţia (5.114), rezultă

$$\frac{q_i^*}{1-p_i^*} = \frac{r_i^* - q_i^*}{1-p_i^*} \le \frac{r_i^* - r_i^* p_i^*}{1-p_i^*} = r_i^* < 1 . \tag{5.122}$$

Deoarece, în cazul diagonal-dominanţei matricei A, cantitatea $r = \max_i r_i$ reprezintă tocmai norma maximă a matricei rezolvante a procedeului Jacobi, ($r \equiv \|P_J\|$) din (5.121) rezultă că rata de micşorare a erorii de la o aproximaţie la alta este mai mică sau cel mult egală cu r, adică procedeul Gauss-Seidel converge cel puţin tot atât de rapid ca procedeul Jacobi.

Pentru aplicaţii se recomandă ca, în cazul când matricea A este dominantă pe linii, să se ia startul cu ecuaţia $i = i_1$ din (5.109) în care se realizează dominanţa maximă a elementului diagonal. Prima necunoscută calculată va fi x_{i_1}. În acest fel, se realizează reducerea maximă a erorii la start, deoarece

$$p_{i_1} = 0, \quad q_{i_1} = r_{i_1} = \text{minim} . \tag{5.123}$$

În continuare, pot fi ordonate celelalte ecuaţii după gradul de dominanţă a elementelor diagonale. În cazul când matricea are elementele dominante pe coloane (5.105), procedeul Gauss-Seidel este de asemenea convergent.

Există şi alte condiţii în care procedeul este convergent. Un alt criteriu, important pentru aplicaţii, este că iteraţia Gauss-Seidel este convergentă dacă matricea A este simetrică şi pozitiv definită.

Algoritmul pentru procedeul Gauss-Seidel este

$$
\begin{array}{l}
xvechi \leftarrow x_0; \quad xnou \leftarrow x_0; \quad k \leftarrow 0; \\
\text{repetă} \\
\quad \left[\begin{array}{l}
i = 1 : n \\
\quad \left[\begin{array}{l}
s \leftarrow b_i \\
j = 1 : n \\
\quad [\text{dacă } i \neq j \text{ atunci } s \leftarrow s - a_{i,j}\, xnou_j \\
xnou_i \leftarrow s
\end{array} \right. \\
AtinsPrecizie \leftarrow \|xnou - xvechi\| \leq \varepsilon\,(1 + \|xnou\|) \\
xvechi \leftarrow xnou; \quad k \leftarrow k + 1
\end{array} \right. \\
\text{până când } AtinsPrecizie \text{ sau } k > k_{\max}\,.
\end{array}
\tag{5.124}
$$

Se observă asemănarea cu algoritmul pentru metoda Jacobi (5.107). Ca şi în acel algoritm s-a considerat că liniile matricei A şi ale vectorului b au fost împărţite la termenii de pe diagonală.Vectorul $xvechi$ este utilizat numai în condiţia de stabilire a convergenţei.

5.2.3 Accelerarea convergenţei metodelor iterative

Accelerarea convergenţei procedeelor iterative este posibilă dacă avem informaţii privind valorile proprii ale matricei rezolvante. Într-adevăr, condiţia necesară şi suficientă pentru ca orice metodă iterativă să fie convergentă este

$$\rho(P) = \max_i |\lambda_i| < 1 \tag{5.125}$$

unde λ_i, $i = \overline{1,n}$ sunt valorile proprii ale matricei P. Valoarea proprie maximă în modul este numită *raza spectrală* şi se notează cu $\rho(P)$. Dificultatea aplicării acestui criteriu constă în faptul că, în general, problema determinării valorilor proprii ale unei matrice este mai complicată decât problema rezolvării sistemului $Ax = b$. De aceea, se caută obţinerea unor metode de accelerare care să ocolească problema determinării valorilor proprii, utilizând informaţii minime. Vom da două exemple.

Cazul matricelor pozitiv definite.

Există sisteme ale căror matrice A sunt cunoscute ca fiind pozitiv definite (de obicei şi simetrice), adică având valorile proprii pozitive. Fie m cea mai mică şi M cea mai mare valoare proprie ($M > m > 0$). Aceste valori (vezi Cap. 6) se determină mai uşor.

Se defineşte şirul de iteraţii

$$x^{(k+1)} = x^{(k)} + \beta(b - Ax^{(k)}) = (I - \beta A)x^{(k)} + \beta b\,, \tag{5.126}$$

care aminteşte de tehnica folosită pentru accelerarea convergenţei iteraţiei simple (3.19). Matricea rezolvantă este

$$P = I - \beta A, \quad \lambda(P) = 1 - \beta\lambda(A) \tag{5.127}$$

având valorile proprii $\lambda(P)$ funcţii de parametrul de accelerare β. Valoarea minimă a lui $\max|\lambda(P)|$ se obţine pentru

$$\beta = \frac{2}{m+M}, \quad \rho(P) = \frac{M-m}{M+m} < 1 \tag{5.128}$$

şi este subunitară pentru orice $M, m > 0$. Se observă importanţa raportului M/m pentru convergenţa procedeului iterativ. Acest raport reprezintă o evaluare a *numărului de condiţionare* al matricei A, notat $\mu(A)$.

Accelerarea convergenţei procedeului Gauss-Seidel.

Şirul iterativ (5.126) se poate generaliza introducând o matrice B, uşor inversabilă, astfel încât

$$x^{(k+1)} = x^{(k)} + \beta B(b - Ax^{(k)}) = (I - \beta BA)x^{(k)} + \beta Bb\,. \tag{5.129}$$

Matricea B şi parametrul β se aleg astfel încât matricea rezolvantă a sistemului (5.129),

$$P = I - \beta BA\,, \tag{5.130}$$

să aibă o rază spectrală $\rho(P)$ cât mai mică. Se observă că, în cazul procedeului Jacobi (5.97), se obţine $\beta = 1$ şi $B = D^{-1}$. Din punctul de vedere al aplicaţiilor, este util să observăm că procedeele de accelerare a iteraţiilor iau în calcul o parte convenabilă din elementul de pe diagonala principală (vezi 5.99). Spre exemplu, în cazul metodei Gauss-Seidel şirul de iteraţii (5.109) se înlocuieşte prin

$$x_i^{(k+1)} = (1-\beta)x_i^{(k)} + \beta(b_i - \sum_{j=1}^{i-1} a_{i,j}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{i,j}x_j^{(k)})/a_{i,i} \tag{5.131}$$

pentru $i = \overline{1,n}$. Din punct de vedere matriceal, (5.131) este echivalentă cu descompunerea matricei A sub forma

$$A = \beta^{-1}D - (\beta^{-1} - 1)D - T - S\,. \tag{5.132}$$

Comparând (5.132) cu (5.110) se observă că singura modificare este reâmpărţirea diagonalei matricei A, o operaţie puţin costisitoare. Şirul iterativ va fi definit acum sub forma

$$Dx^{(k+1)} = (1-\beta)Dx^{(k)} + \beta(Tx^{(k+1)} + Sx^{(k)} + b)\,, \tag{5.133}$$

de unde, prin înmulţire cu D^{-1}, se obţine

$$x^{(k+1)} = (1-\beta)x^{(k)} + \beta D^{-1}(Tx^{(k+1)} + Sx^{(k)} + b)\,, \tag{5.134}$$

expresie echivalentă cu (5.131). Explicitând $x^{(k+1)}$ din (5.134) se obţine

$$\begin{aligned} x^{(k+1)} \;=\; & (I - \beta D^{-1}T)^{-1}\left[(1-\beta)I + \beta D^{-1}S\right]x^{(k+1)} \\ & + (I - \beta D^{-1}T)^{-1}\beta D^{-1}b\,, \end{aligned} \tag{5.135}$$

matricea rezolvantă având expresia

$$P = (D - \beta T)^{-1} \left[(1 - \beta) D + \beta S \right] . \tag{5.136}$$

Pentru $\beta = 1$, din (5.136) se regăseşte matricea rezolvantă a metodei Gauss-Seidel standard. O condiţie necesară de convergenţă a şirului iterativ (5.135) este ca produsul valorilor proprii în modul să fie subunitar. În acest fel se obţine intervalul de valori ale parametrului de accelerare β

$$0 < \beta < 2 . \tag{5.137}$$

Metodele care utilizează $0 < \beta < 1$ se numesc *metode de subrelaxare*, iar metodele cu $1 < \beta < 2$ se numesc *metode de suprarelaxare*. În lipsa altor informaţii privind valorile proprii ale matricei P, parametrul β se alege prin încercări.

5.3 Comparaţii între metode

Am prezentat o serie de algoritmi din cele două clase mari de metode: de eliminare şi iterative. Să efectuăm acum o comparaţie între cele două clase de metode.

Ca număr de operaţii.

Oricare din metodele iterative prezentate necesită $O(n^2)$ operaţii aritmetice per iteraţie. Prin urmare, metodele iterative devin avantajoase dacă numărul k de iteraţii nu este prea mare. Spre exemplu, o comparaţie cu metoda de eliminare Gauss conduce la concluzia că, pentru a avea un număr apropiat de operaţii, este necesar ca numărul de iteraţii k să satisfacă inegalitatea

$$k \leq 1 + n/3 \cong n/3 . \tag{5.138}$$

Inegalitatea (5.138) este cu atât mai uşor de satisfăcut cu cât numărul de ecuaţii n este mai mare. Practic, condiţia (5.138) se realizează pentru n **&** 50.

Ca propagare a erorilor.

Metodele iterative sunt, în general, mai avantajoase, deoarece erorile sunt diminuate la fiecare pas prin înmulţirea cu norma matricei rezolvante, care este subunitară ($\|P\| < 1$). De fapt, rata de diminuare a erorii corespunde razei spectrale a matricei rezolvante. Totodată, la fiecare iteraţie se introduc erori de rotunjire.

Să considerăm că dorim să obţinem un rezultat cu q zecimale exacte. În acest scop sunt necesare k iteraţii, astfel încât, în absenţa erorilor de rotunjire, se obţine (vezi 5.102)

$$\left\| e^{(k)} \right\| \leq \|P\|^k \cdot \left\| e^{(0)} \right\| \leq 10^{-q} , \tag{5.139}$$

$e^{(0)}$ fiind eroarea de start. Logaritmând a doua inegalitate în baza 10, rezultă

$$k \lg \|P\| + \lg \left\| e^{(0)} \right\| \leq -q \,. \tag{5.140}$$

Prin urmare, se obţine o condiţie pentru numărul de iteraţii k

$$k \geq -\left(q + \lg \left\| e^{(0)} \right\| \right) / \lg \|P\| \,, \quad \|P\| < 1 \,. \tag{5.141}$$

Rezultă că numărul de iteraţii necesar creşte cu precizia impusă q, cu eroarea iniţială $e^{(0)}$ şi scade cu $-\lg \|P\|$. Dacă se adaugă şi efectul erorilor de rotunjire, marginea (5.141) trebuie majorată.

Deoarece $e^{(0)}$ nu se cunoaşte, se pot utiliza evaluări obţinute cu ajutorul unor prime iteraţii. Într-adevăr, din inegalitatea (5.139) se poate obţine

$$\left\| e^{(k)} \right\| \leq \frac{\alpha}{1-\alpha} \left\| x^{(k)} - x^{(k-1)} \right\| \leq \frac{\alpha^k}{1-\alpha} \left\| x^{(1)} - x^{(0)} \right\| \leq 10^{-q} \,, \tag{5.142}$$

cu $\alpha \equiv \|P\|$. Pentru procedeul Jacobi, $\|P\|$ este uşor de calculat. Pentru procedeul Gauss-Seidel, în cazul matricei diagonal-dominante pe linii se ia (vezi şi 5.121)

$$\alpha = \max_i \frac{q_i}{1-p_i} \tag{5.143}$$

ceea ce corespunde la folosirea normei $\|\|_\infty$ pentru eroare. În acest fel se obţine o evaluare a numărului de iteraţii în funcţie de prima aproximaţie $x^{(1)}$

$$k \geq -\left(q + \lg \left\| x^{(1)} - x^{(0)} \right\| - \lg \|1-\alpha\| \right) / \lg \alpha \,. \tag{5.144}$$

5.4 Elemente de calcul matriceal

Aşa cum s-a văzut, în rezolvarea sistemelor de ecuaţii (dar nu numai aici) intervin operaţii cu matrice, îndeosebi matrice pătrate. Operăm de multe ori cu matrice, chiar când acest lucru nu apare în mod explicit, aşa cu se întâmplă spre exemplu în cazul eliminării unor necunoscute din ecuaţiile sistemului.

Presupunem cunoscute operaţiile de adunare şi înmulţire ale matricelor. Amintim doar că adunarea este definită pentru matrice de acelaşi tip (cu acelaşi număr de linii şi coloane) şi că, la înmulţirea de matrice, numărul de coloane ale matricei din stânga trebuie să fie egal cu numărul de linii ale matricei din dreapta. Această condiţie se realizează întotdeauna pentru matricele pătrate de aceeaşi dimensiune. Produsul matricelor nu este comutativ.

Fie A o matrice pătrată de dimensiuni $n \times n$. Matricea obţinută prin înmulţirea lui A cu ea însăşi de k ori este tot o matrice pătrată şi se notează cu A^k. Prin analogie cu puterile scalarilor, prin A^0 se înţelege matricea unitate I, o matrice pătrată de dimensiune n. Cu puterile matricelor pătrate se pot construi polinoame de matrice, în mod asemănător cu cazul scalarilor.

Matricele pot fi înmulţite (la dreapta sau la stânga) cu vectori (coloană sau linie) de dimensiuni corespunzătoare, rezultatul fiind un vector. Ca urmare, o matrice poate fi privită şi ca un operator (funcţie sau aplicaţie) definit pe un spaţiu vectorial cu valori care reprezintă, de asemenea, elementele unui spaţiu vectorial (de dimensiune finită).

Ca şi în cazul vectorilor, compararea matricelor (într-un anumit sens) cu numere reale se poate face introducând *norme de matrice*. Norma unei matrice oarecare A se notează cu $\|A\|$ şi îndeplineşte axiomele generale ale normelor

1. $\|A\| > 0$;
2. $\|\alpha A\| = |\alpha| \, \|A\|$ cu $\alpha \in \mathbf{C}$;
3. $\|A + B\| \leq \|A\| + \|B\|$.

Din acestea mai rezultă

$$\|A\, B\| \leq \|A\| \cdot \|B\| \; . \tag{5.145}$$

Dacă x este un vector asupra căruia operează matricea A, atunci normele $\|A\|$ şi $\|x\|$ sunt *compatibile* dacă are loc inegalitatea

$$\|A\, x\| \leq \|A\| \cdot \|x\| \; . \tag{5.146}$$

Se poate arăta că inegalităţile (5.145) şi (5.146) se pot deduce din condiţiile impuse normelor operatorilor liniari şi continui pe spaţii vectoriale de dimensiune finită (vezi şi 1.4.3). Printre normele de matrice des utilizate care îndeplinesc condiţiile impuse se numără:

1. *norma maximă*

$$\|A\|_\infty = \sup_i \sum_j |a_{i,j}| \, , \tag{5.147}$$

   dată de suma maximă a modulelor elementelor matricei, pe linii;

2. *norma unu*

$$\|A\|_1 = \sup_j \sum_i |a_{i,j}| \, , \tag{5.148}$$

   dată de suma maximă a modulelor elementelor matricei, pe coloane;

3. *norma euclidiană*

$$\|A\|_E = \left(\sum_{i,j} |a_{i,j}|^2 \right)^{1/2} \; ; \tag{5.149}$$

4. *norma spectrală*

$$\|A\|_2 = (\rho(A^*A))^{1/2}\,, \tag{5.150}$$

adică raza spectrală a matricei produs A^*A, cu A^* matricea adjunctă a lui A, care se obţine prin transpunerea şi conjugarea complexă a matricei A

$$A^* = \bar{A}^T\,. \tag{5.151}$$

Pentru vectori se utilizează normele precizate în (1.105). Normele de matrice sunt compatibile cu normele vectoriale de acelaşi indice. În particular, norma spectrală $\|A\|_2$ este compatibilă cu norma euclidiană a vectorului x. Dacă vectorul x are componente complexe norma sa euclidiană este definită de

$$\|x\|_2 = \sqrt{(x,x)} = \sum_{i=1}^{n} x_i \bar{x}_i\,. \tag{5.152}$$

Produsul scalar a doi vectori ce au componente complexe este definit ca

$$x \cdot y = (x,y) = \sum_{i=1}^{n} x_i \bar{y}_i \tag{5.153}$$

şi are proprietăţile

$$(x,y) = \overline{(y,x)}, \quad (\lambda x, y) = \lambda (x,y), \quad (x, \lambda y) = \bar{\lambda}(x,y). \tag{5.154}$$

Un şir de matrice A_n este convergent dacă şirurile $\{a_{i,j}\}_n$ sunt convergente. Limitele acestor şiruri formează o matrice A care este, prin definiţie, limita şirului de matrice A_n. În acest fel, convergenţa şirului de matrice s-a redus la convergenţa şirurilor de numere. În mod similar, se introduce convergenţa şirurilor de vectori. O condiţie suficientă pentru convergenţa seriei matriceale

$$S = \sum_{n=0}^{\infty} A^n \tag{5.155}$$

este ca norma matricei A să fie subunitară, $\|A\| > 1$. În acest caz se obţine suma seriei

$$S = (I - A)^{-1}\,, \tag{5.156}$$

egală cu inversa matricei $I - A$. Se observă analogia cu seria geometrică de numere. Din (5.156) se obţine, trecând la norme, o inegalitate importantă

$$\left\|(I-A)^{-1}\right\| \le \sum_{n=0}^{\infty} \|A\|^n = (1 - \|A\|)^{-1}. \tag{5.157}$$

Condiţia necesară şi suficientă pentru ca seria (5.155) să fie convergentă este ca toate valorile proprii λ_i ale matricei A să fie subunitare în modul, adică raza spectrală $\rho(A)$ a matricei A să fie subunitară

$$\rho(A) = \max_i |\lambda_i| < 1\,. \tag{5.158}$$

Deoarece orice procedeu iterativ este de forma

$$x^{(k+1)} = Px^{(k)} + d\,, \tag{5.159}$$

unde P este matricea rezolvantă, rezultă că

$$x^{(k+1)} = (I + P + P^2 + \ldots + P^k)d + P^{k+1}x^{(0)}\,. \tag{5.160}$$

Deoarece vectorii d şi $x^{(0)}$ sunt arbitrari, pentru ca procedeul iterativ să fie convergent, trebuie ca seria de matrice să fie convergentă, deci ca raza spectrală a matricei P să fie subunitară, $\rho(P) < 1$.

Capitolul 6

Vectori şi valori proprii

6.1 Elemente introductive

Fie A o matrice pătrată, de dimensiuni $n \times n$. Se numeşte *valoare proprie* o mărime scalară λ care satisface relaţia

$$Ax = \lambda x\,, \tag{6.1}$$

vectorul x fiind *vectorul propriu* corespunzător valorii proprii λ. Sistemul de ecuaţii liniare (6.1) se poate scrie $(A - \lambda I)x = 0$, deci este omogen şi admite o soluţie nenulă dacă şi numai dacă determinantul sistemului este nul

$$P(\lambda) \equiv \det(A - \lambda I) = 0\,. \tag{6.2}$$

Polinomul de gradul n, $P(\lambda)$ se numeşte *polinomul caracteristic* al matricei A iar ecuaţia (6.2) se numeşte *ecuaţie caracteristică*. Conform teoremei fundamentale a algebrei există n valori proprii care sunt tocmai rădăcinile polinomului caracteristic. Mulţimea valorilor proprii se numeşte *spectrul matricei* A şi se notează

cu $spec(A)$. Vectorii proprii, ca soluţii ale unor sisteme de ecuaţii omogene, sunt determinaţi până la o constantă multiplicativă. Găsirea valorilor şi vectorilor proprii prezintă o importanţă practică deosebită pentru rezolvarea sistemelor de ecuaţii diferenţiale liniare, pentru studiul stabilităţii mişcărilor mecanice sau de altă natură, inclusiv pentru studiul stabilităţii diverselor metode numerice. Trebuie remarcat însă că determinarea practică a valorilor şi vectorilor proprii nu se face, în general, plecând de la construirea polinomului caracteristic, deoarece mici abateri în valorile coeficienţilor acestui polinom conduc la abateri mari în valorile soluţiilor (se spune că problema rezolvării ecuaţiei caracteristice este *rău condiţionată*). Ca urmare, procedeele care determină valorile proprii prin construirea polinomului caracteristic sunt limitate la matrice de dimensiuni mici ($n \leq 10$) unde erorile de rotunjire introduse nu sunt importante.

Vom da în continuare o serie de procedee, precizând avantajele şi limitele lor de aplicabilitate. Ne vom limita, în general, la cazul matricelor reale.

6.2 Metode pentru câteva valori proprii

Metodele numerice pentru rezolvarea problemei $Ax = \lambda x$ se pot clasifica în metode ce urmăresc determinarea tuturor valorilor proprii şi în metode ce determină doar câteva dintre valorile proprii. Ne îndreptăm mai întâi atenţia asupra celei de-a doua clase de metode. Prezintă un interes deosebit în aplicaţii determinarea valorilor proprii extremale, care delimitează spectrul matricei A. De aici şi marea varietate de metode concepute pentru a determina valorile proprii cele mai mari sau mai mici. Vom considera câteva dintre metodele cele mai folosite.

6.2.1 Metoda puterii directe

Metoda puterii directe permite determinarea valorii proprii de modul maxim, în ipoteza că vectorii proprii ai matricei formează o bază, adică sunt liniar independenţi. Acest lucru se întâmplă întotdeana când valorile proprii sunt distincte. Dacă există valori multiple, atunci vectorii proprii pot sau nu să fie liniar independenţi. Spre exemplu, matricea unitate I are toate valorile proprii egale ($\lambda_i = 1$, $i = \overline{1,n}$), dar vectorii proprii sunt liniar independenţi. Se consideră un vector y cunoscut, de obicei de formă cât mai simplă. In ipoteza că vectorii proprii x_i , $i = \overline{1,n}$, ai matricei A formează o bază, putem scrie

$$y = \sum_{i=1}^{n} c_i x^{(i)}\,, \tag{6.3}$$

unde c_i sunt scalari.

Cazul valorilor proprii reale.

Valorile proprii, reale λ_i, $i = \overline{1,n}$ se pot ordona sub forma

$$|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n| \tag{6.4}$$

astfel încât valoarea proprie λ_1 este cea mai mare dintre ele în modul. Înmulţind repetat vectorul y cu matricea A şi utilizând relaţia (6.1), se obţine

$$z^{(1)} = Ay = \sum_{i=1}^{n} c_i A x^{(i)} = \sum_{i=1}^{n} c_i \lambda_i x^{(i)} \tag{6.5}$$

$$\dots \tag{6.6}$$

$$z^{(k)} = A^k y = A z^{(k-1)} = \sum_{i=1}^{n} c_i \lambda_i^k x^{(i)} = \lambda_1^k \left[c_1 x^{(1)} + O\left(\lambda_2^k / \lambda_1^k\right) \right] \tag{6.7}$$

$$z^{(k+1)} = A^{k+1} y = \sum_{i=1}^{n} c_i \lambda_i^{k+1} x^{(i)} = \lambda_1^{k+1} \left[c_1 x^{(1)} + O\left(\lambda_2^{k+1} / \lambda_1^{k+1}\right) \right] \tag{6.8}$$

Pentru k suficient de mare termenii de ordinul $(\lambda_2/\lambda_1)^k$ devin neglijabili, astfel încât putem aproxima

$$z^{(k)} \cong \lambda_1^k c_1 x^{(1)}, \quad z^{(k+1)} \cong \lambda_1^{k+1} c_1 x^{(1)} . \tag{6.9}$$

Prin urmare, luând o componentă de indice i a vectorilor $z^{(k)}$ şi $z^{(k+1)}$, se obţine valoarea proprie de modul maxim

$$\lambda_1 \cong z_i^{(k+1)} / z_i^{(k)} . \tag{6.10}$$

O estimare îmbunătăţită a valorii proprii este furnizată de *raportul Rayleigh*

$$\lambda_1 \cong \left[z^{(k)}\right]^T z^{(k+1)} / \left[z^{(k)}\right]^T z^{(k)} = \left[z^{(k)}\right]^T A z^{(k)} / \left[z^{(k)}\right]^T z^{(k)} . \tag{6.11}$$

În ceea ce priveşte vectorul propriu corespunzător $x^{(1)}$, acesta are direcţia $z^{(k+1)}$. Practic, vectorul y se alege sub forme ca: $y = (1, 0, \dots)$, $y = (1, 1, \dots)$ sau alte forme convenabile, deci coeficienţii c_i nu trebuie cunoscuţi efectiv, condiţiile de aplicabilitate ale procedeului (vectori proprii liniar independenţi, $c_1 \neq 0$) verificându-se prin comportarea şirurilor $z_i^{(k+1)} / z_i^{(k)}$, $\overline{1, n}$. Dacă aceste şiruri par convergente, începând cu anumit k, şi tind spre aceaşi valoare (care reprezintă, la limită, valoarea proprie λ_1) atunci se poate trage concluzia că procedeul este aplicabil. Pentru reducerea erorilor de rotunjire se recomandă normarea prin împărţire cu $\left\|z^{(k)}\right\|$, în fiecare etapă. Dacă nu există o singură valoare proprie reală de modul maxim, spre exemplu dacă

$$|\lambda_1| = |\lambda_2| > |\lambda_3| \geq \dots \geq |\lambda_n| , \tag{6.12}$$

atunci după k iteraţii se obţine

$$z^{(k)} \cong \lambda_1^k \left(c_1 x^{(1)} + c_2 x^{(2)} \right), \quad z^{(k+1)} \cong \lambda_1^{k+1} \left(c_1 x^{(1)} + c_2 x^{(2)} \right) , \tag{6.13}$$

termenii neglijaţi fiind de ordinul $(\lambda_3/\lambda_1)^k$. Se observă că valoarea proprie se poate calcula ca şi până acum, din (6.10); în schimb vectorii proprii $x^{(1)}$ şi $x^{(2)}$

intră în combinaţie liniară, cu cel puţin unul dintre coeficienţii c_1, c_2 nenul. Se poate scrie însă $Ax^{(1)} = \lambda_1 x^{(1)}$, $Ax^{(2)} = \lambda_1 x^{(2)}$ de unde se deduce imediat că

$$\lambda_1 \left[\lambda_1^k \left(c_1 x^{(1)} + c_2 x^{(2)}\right)\right] = A \left[\lambda_1^k \left(c_1 x^{(1)} + c_2 x^{(2)}\right)\right], \tag{6.14}$$

adică $z^{(k)}$ este de asemenea un vector propriu corespunzător valorii λ_1. Un al doilea vector propriu liniar independent de $z^{(k)}$ se va găsi în mod similar, repetând procedeul pentru un alt vector $y^{(1)}$, liniar independent de y. Viteza de convergenţă a procedeului depinde de modulul raportului λ_3/λ_1.

Cazul valorilor proprii complex conjugate.

Presupunem că valorile proprii de modul maxim λ_1, λ_2 sunt complex conjugate

$$\lambda_1 = \bar{\lambda}_2, \quad |\lambda_1| = |\lambda_2| > |\lambda_3| \geq \ldots \geq |\lambda_n|, \tag{6.15}$$

şi că vectorii proprii x_i, $i = \overline{1,n}$ sunt liniar independenţi. Vom observa, mai întâi, că vectorii proprii $x^{(1)}$ şi $x^{(2)}$ sunt de asemenea complex conjugaţi, aşa cum rezultă imediat din relaţia de definiţie (6.1) pentru matrice reale. Metoda se aplică similar cu cazul valorilor proprii reale dar necesită utilizarea rezultatelor din mai multe iteraţii succesive dacă dorim evitarea lucrului cu variabile complexe. In iteraţia k se obţine

$$z^{(k)} = \lambda_1^k \left[c_1 x^{(1)} + \bar{\lambda}_1^k/\lambda_1^k c_2 x^{(2)} + O\left(\lambda_3^k/\lambda_1^k\right)\right], \; x^{(2)} = \overline{x^{(1)}}, \; c_2 = \bar{c}_1 . \tag{6.16}$$

Ultima egalitate se obţine ţinând cont că s-a plecat cu un vector real y. Se caută o modalitate de a exprima cantităţile reale $\lambda_1 + \overline{\lambda_1}$, $\lambda_1\overline{\lambda_1}$ funcţie de mărimile disponibile pe parcursul iteraţiilor. Pentru k suficient de mare se obţin combinaţiile

$$u_{ki} \equiv z_i^{(k+1)} z_i^{(k-1)} - (z_i^{(k)})^2 = \left|c_i x_i^{(1)}\right|^2 (\lambda_1\overline{\lambda_1})^{k-1}(\lambda_1 - \overline{\lambda_1})^2, \tag{6.17}$$

$$v_{ki} \equiv z_i^{(k+2)} z_i^{(k-1)} - z_i^{(k+1)} z_i^{(k)} = u_{ki}(\lambda_1 + \overline{\lambda_1}), \tag{6.18}$$

de pe o componentă i oarecare. Din relaţiile (6.17-6.18) se obţin valorile proprii, observând că

$$\lambda_1 + \overline{\lambda_1} = v_{ki}/u_{ki} + O(\lambda_3^k/\lambda_1^k), \; \lambda_1\overline{\lambda_1} = u_{k+1,i}/u_k + O(\lambda_3^k/\lambda_1^k). \tag{6.19}$$

Notând $\lambda_1 \equiv a + ib$, rezultă

$$a = v_{ki}/2u_{ki}, \quad a^2 + b^2 = u_{k+1,i}/u_{ki}. \tag{6.20}$$

În ceea ce priveşte vectorul propriu corespunzător, $x^{(1)}$ se poate proceda în felul următor: vectorul propriu fiind determinat până la o constantă multiplicativă, putem lua această constantă egală cu c_1, astfel încât

$$c_1 x_i^{(1)} = \rho_i e^{i\theta_1}, \quad i = \overline{1,n}. \tag{6.21}$$

Notând apoi cu σ argumentul numărului complex λ_1 din (6.16) se deduce

$$z_i^{(k)} = \lambda_1^k c_1 x^{(1)} + \overline{\lambda_1^k c_1 x^{(1)}} = 2\,|\lambda_1|^k \cos(\theta_i + \sigma), \quad i = \overline{1,n}\,, \tag{6.22}$$

iar din (6.17) se obţine

$$u_{ki} = -4\,|\lambda_1|^{2k}\,\rho_i^2 \sin^2\sigma < 0\,. \tag{6.23}$$

Putem acuma deduce necunoscutele ρ_i, θ_i

$$\rho_i = \sqrt{-u_{ki}}/2\,|\lambda|^k \sin\sigma, \quad \cos(\theta_i + \sigma) = z_i^k \sin\sigma/\sqrt{-u_{ki}}\,. \tag{6.24}$$

EXEMPLU. Să se determine valoarea proprie de modul maxim a matricei

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 1 \end{bmatrix}$$

Rezolvare. Deoarece matricea A este simetrică, toate valorile proprii sunt reale. Pentru aflarea valorii proprii de modul maxim vom aplica metoda puterii pornind cu vectorul $y = (1,0,0)^T$. Primele câteva iteraţii sunt redate în tabelul de mai jos. S-au utilizat notaţiile

$$r_i^{(k)} = z_i^{(k+1)}/z_i \tag{6.25}$$

iar R este raportul Rayleigh.

k	$z_1^{(k)}$	$z_2^{(k)}$	$z_2^{(k)}$	$r_1^{(k)}$	$r_2^{(k)}$	$r_3^{(k)}$	R
2	14	10	10	4.571	5.800	7.200	5.5455
3	64	58	72	–	–	–	–
5	2196	1882	2228	5.758	5.701	5.646	5.7013
6	12644	10730	12580	–	–	–	

Valorile proprii ale matricei sunt $\lambda_1 = 5.701562$, $\lambda_2 = -2$, $\lambda_3 = -0.01562$. Se observă obţinerea primelor 4 cifre semnificative după circa 6 iteraţii. O estimare mai bună a valorii proprii este furnizată de raportul Rayleigh. Această comportare este caracteristică pentru matricele simetrice, unde raportul Rayleigh tinde către valoarea proprie conform

$$\left[z^{(k)}\right]^T z^{(k+1)} / \left[z^{(k)}\right]^T z^{(k)} = \lambda_1 + O\left(\lambda_1^{2k}/\lambda_2^{2k}\right).$$

De asemenea, se observă creşterea rapidă a componentelor vectorului $z^{(k)}$ pe măsură ce k creşte. De aceea se recomandă împărţirea la norma vectorului la fiecare iteraţie.

6.2.2 Metoda puterii inverse

Este utilizată pentru a găsi cea mai mică valoare proprie în modul, diferită de zero. În aceste condiţii, matricea A este nesingulară şi A^{-1} există. Notând cu λ_1 această valoare proprie de modul minim, se scrie

$$0 < |\lambda_1| < |\lambda_2| \leq \ldots \leq |\lambda_n|\,. \tag{6.26}$$

Din $Ax = \lambda x$ se obţine

$$x = \lambda A^{-1}x \Rightarrow A^{-1}x = \lambda^{-1}x, \tag{6.27}$$

sau că matricea inversă are valorile proprii $1/\lambda$. Prin urmare, $1/\lambda_1$ este valoarea proprie de modul maxim a matricei A^{-1} şi se poate aplica metoda puterii directe acestei matrice, în ipoteza că vectorii proprii formează o bază. Plecând de la un vector iniţial arbitrar $y = z^{(0)}$, se formează şirul

$$z^{(k+1)} = A^{-1}z^{(k)}, \quad k = 0, 1, \ldots . \tag{6.28}$$

Practic, nu se calculează inversa matricei A, deoarece este mai economică rezolvarea repetată a sistemului de ecuaţii

$$Az^{(k+1)} = z^{(k)}, \tag{6.29}$$

prin eliminare gaussiană de exemplu. Mai departe, toate consideraţiile de la metoda puterii directe sunt aplicabile şi în acest caz.

6.2.3 Metoda deplasării

Se bazează pe observaţia că, dacă matricea A are valoarea proprie λ atunci matricea $A - qI$, unde q este un scalar, are valoarea proprie $q - \lambda$. Scalarul q reprezintă tocmai *deplasarea* şi poate fi utilizat pentru a găsi şi alte valori proprii decât cele maxime sau minime în modul. Spre exemplu, dacă am găsit valoarea de modul maxim λ_1 (folosind metoda puterii directe), atunci valoarea proprie cea mai depărtată de λ_1, fie aceasta λ_j, va avea proprietatea

$$|\lambda_j - \lambda_1| = \sup_{i=\overline{1,n}} |\lambda_j - \lambda_1| . \tag{6.30}$$

În consecinţă, metoda puterii directe aplicată matricei $A - \lambda_1 I$ ne va conduce la valoarea proprie λ_j.

Metoda deplasării originii poate fi folosită în combinaţie cu metoda puterii inverse atunci când cunoaştem aproximativ valoarea proprie $\lambda \cong \mu$. În aceste condiţii matricea $A - \mu I$ are o valoare proprie mică, în modul, deci şirul similar cu (6.29)

$$(A - \mu I)z^{(k+1)} = z^{(k)} \tag{6.31}$$

va converge rapid, furnizând valoarea exactă λ.

6.3 Determinarea tuturor valorilor şi vectorilor proprii

Metodele prezentate până în prezent, deşi foarte simple, permit determinarea numai a unor valori proprii particulare, fiind limitate şi de condiţia ca vectorii

proprii să formeze un sistem liniar independent. Metodele mai generale, care permit determinarea tuturor valorilor şi vectorilor proprii, se bazează, în esenţă, pe aducerea matricei la o formă particulară prin transformări similare.

Două matrice pătrate A şi B de dimensiune n se numesc matrice *asemenea* (sau *similare*), dacă există o matrice P, inversabilă, astfel încât au loc relaţiile

$$B = PAP^{-1}, \quad A = P^{-1}BP\,. \tag{6.32}$$

Două matrice asemenea au aceleaşi valori proprii. Într-adevăr, dacă λ este o valoare proprie a matricei A, adică $Ax = \lambda x$ cu $x \neq 0$, ţinând cont de relaţiile (6.32) se obţine

$$P^{-1}BPx = \lambda x \Rightarrow B(Px) = \lambda(Px), \tag{6.33}$$

adică matricea B are aceleaşi valori proprii ca şi A. În plus, dacă pentru o valoare proprie λ matricea A are vectorul propriu x, atunci pentru matricea B vectorul propriu corespunzător lui este

$$y = Px\,, \tag{6.34}$$

adică se obţine din vectorul x prin înmulţire cu matricea P de premultiplicare.

Este de dorit ca transformările similare specificate prin matricele P, P^{-1} cu care înmulţim matricea A, să conducă la matrice B "simple", având valorile proprii şi vectorii proprii uşor de determinat. Există mai multe astfel de forme simple. Prima pe care o vom considera în cele ce urmează este forma Frobenius utlizată în metoda Danilevschi. Apoi vom prezenta algoritmii LR şi QR de aducere a matricei A la forma triunghiulară.

6.4 Metoda Danilevschi

Această metodă aduce matricea A la forma

$$F = \begin{bmatrix} p_1 & p_2 & \dots & p_{n-1} & p_n \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}, \tag{6.35}$$

denumită *formă Frobenius*, prin transformări similare. Avantajul formei Frobenius este obţinerea imediată a ecuaţiei caracteristice a matricei

$$(-1)^n \det(F - \lambda I) = \lambda^n - \sum_{i=1}^{n} p_i \lambda^{n-i} = 0\,. \tag{6.36}$$

Vom deduce relaţia (6.36) pe o cale care să permită totodată şi construirea vectorilor proprii y ai matricei F. În acest scop, vom scrie, conform definiţiei

(6.1), egalitatea $Fy = \lambda y$, adică

$$\begin{bmatrix} p_1 & p_2 & \dots & p_{n-1} & p_n \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \lambda \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} . \tag{6.37}$$

Operând de la ultima linie către prima, se obţine

$$y_{n-j} = \lambda y_{n-j+1} , \tag{6.38}$$

pentru $j = \overline{1, n-1}$, şi

$$\sum_{i=1}^{n} p_i y_i = \lambda y_i . \tag{6.39}$$

Exprimând componentele $y_1, y_2, \dots, y_{n-1}$ în funcţie de y_n, din ecuaţiile (6.38) rezultă

$$y_{n-1} = \lambda y_n, \quad y_{n-2} = \lambda^2 y_n, \dots , \quad y_1 = \lambda^{n-1} y_n . \tag{6.40}$$

Înlocuind în (6.39), rezultă

$$y_n \left(\lambda^n - \sum_{i=1}^{n} p_i \lambda^{n-i} \right) = 0 . \tag{6.41}$$

Deoarece $y_n \neq 0$ (altfel toate componentele vectorului y ar fi nule), din (6.41) se obţine tocmai ecuaţia caracteristică (6.36). Luând $y_n = 1$, vectorul propriu y are expresia

$$y = (\lambda^{n-1}, \lambda^{n-2}, \dots , \lambda, 1)^T . \tag{6.42}$$

În concluzie, vectorii proprii ai matricei Frobenius se construiesc cu ajutorul puterilor valorii proprii, adică pot fi determinaţi imediat ce au fost determinate valorile proprii. Această proprietate remarcabilă pune în evidenţă şi anumite limitări ale formei Frobenius. Astfel, dacă valorile proprii sunt multiple, vectorii corespunzători ai unei matrice Frobenius nu pot fi decât liniar dependenţi. Această proprietate nu este însă general valabilă pentru orice matrice. Spre exemplu, matricea unitate are toate valorile proprii egale, în timp ce vectorii proprii sunt toţi liniar independenţi. Concluzia este că forma Frobenius nu este suficient de generală, adică nu orice matrice poate fi adusă la această formă prin transformări similare. Totuşi, o matrice oarecare A ar putea fi descompusă în blocuri, dintre care cele corespunzătoare valorilor proprii distincte, ar fi blocuri Frobenius.

În metoda Danilevschi aducerea la forma Frobenius a unei matrice pătrate A se face începând cu ultima linie, după schema

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n-1} & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n-1} & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n-1} & a_{n-1,n} \\ a_{n,1} & a_{n,2} & \dots & \boxed{a_{n,n-1}} & a_{n,n} \end{bmatrix} \tag{6.43}$$

$$\rightarrow \begin{bmatrix} a^{(1)}_{1,1} & a^{(1)}_{1,2} & \dots & a^{(1)}_{1,n-1} & a^{(1)}_{1,n} \\ a^{(1)}_{2,1} & a^{(1)}_{2,2} & \dots & a^{(1)}_{2,n-1} & a^{(1)}_{2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a^{(1)}_{n-1,1} & a^{(1)}_{n-1,2} & & a^{(1)}_{n-1,n-1} & a^{(1)}_{n-1,n} \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}. \tag{6.44}$$

S-a presupus $a_{n,n-1} \neq 0$ şi s-au eliminat celelalte elemente din linia a n-a. Dacă $a_{n,n-1} = 0$, se folosesc permutări de coloane. Mai mult, permutările de coloane sunt indicate pentru a avea elemente $a_{n,n-1}$ cât mai mari în modul, în vederea reducerii erorilor de rotunjire. Prin urmare, se recomandă utilizarea pivoţilor (totali sau parţiali), analog cu eliminarea gaussiană. Pentru schema (6.43), s-au folosit multiplicatorii

$$m_{n-1,j} = -a_{n,j}/a_{n,n-1} \text{ pentru } j \neq n-1, \quad m_{n-1,n-1} = 1/a_{n,n-1}\,. \tag{6.45}$$

Se observă că indicii multiplicatorilor se aleg dintre indicii elementelor cu care se construiesc aceşti multiplicatori, diferiţi de indicele liniei care se modifică.

Eliminările şi apariţia elementului 1 nu sunt însă suficiente; mai trebuie ca matricea obţinută să fie similară cu A. În acest scop, vom observa că etapa (6.43) revine la înmulţirea lui A la dreapta cu matricea M_1,

$$M_1 = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ m_{n-1,1} & m_{n-1,2} & \dots & m_{n-1,n-1} & m_{n-1,n} \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}, \tag{6.46}$$

obţinută din matricea unitate, prin introducerea multiplicatorilor. Prin urmare, matricea obţinută în dreapta, în schema (6.43), este matricea AM_1. Faptul că M_1 se aplică la dreapta este legat de operarea cu coloane (multiplicatorii $m_{n-1,j}$ se aplică unor coloane); pe de altă parte, la înmulţire, matricea din dreapta se introduce prin coloane. Matricea inversă M_1^{-1}, necesară pentru a face transformarea similară, este obţinută din matricea unitate prin introducerea în

matricea unitate, în linia $n-1$, a liniei elementelor modificate în matricea A

$$M_1^{-1} = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n-1} & a_{n,n} \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}. \tag{6.47}$$

Verificarea egalităţii $M_1^{-1}M_1 = I$ se face uşor, observând că în matricea produs rămân nemodificate toate liniile cu excepţia liniei $n-1$, singura care trebuie verificată ca aparţinând matricei unitate I. În această verificare, ca şi în cele care urmează este utilă următoarea regulă simplă: dacă, la înmulţirea a două matrice, matricea din stânga are o linie de indice i din matricea unitate I, atunci în matricea produs linia i coincide cu linia i a matricei din dreapta; dacă matricea din dreapta are o coloană j din matricea unitate I, atunci coloana j din matricea produs este dată de matricea din stânga.

Prin urmare, după o primă etapă se obţine matricea similară

$$A_2 = M_1^{-1}A_1M_1, \quad A_1 \equiv A\,. \tag{6.48}$$

Se observă că linia n deja obţinută, din matricea Frobenius, se păstrează la înmulţirea cu M_1^{-1}. Procedeul se repetă pentru liniile $n-1$, $n-2$, …, 2 din matricele similare A_2, A_3, …, A_{n-1} până se ajunge la forma Frobenius. Odată determinate valorile proprii, vectorii proprii y ai matricei Frobenius se deduc din (6.42) iar, vectorii proprii x ai matricei A se deduc acumulând postmultiplicările (vezi 6.33)

$$x = M_1M_2 \dots M_{n-1}y\,. \tag{6.49}$$

Dacă într-o etapă se folosesc permutări de linii, este necesar să permutăm şi coloanele de acelaşi indice cu al liniilor de permutare şi invers astfel încât să se păstreze similitudinea cu matricea A. Aceste permutări trebuie să fie făcute astfel încât să nu afectăm elementele de tip Frobenius deja produse. Aceasta revine în etapa k (când pivotul este $a^{(k)}_{n-k+1,n-k}$) să nu permutăm linii şi coloane de indice mai mare ca $n-k$. Rezultă că pivotul poate fi realizat numai cu permutări de coloane.

Dacă într-o etapă k nu este posibil să obţinem un element diferit de zero în poziţia de sub diagonală, înseamnă că toate elementele din stânga lui $a^{(k)}_{n-k+1,n-k}$ sunt nule. Rezultă că procedeul de aducere la forma Frobenius nu mai poate fi continuat în maniera iniţială. Se poate proceda prin subdivizarea matricei A în blocuri dar nu vom intra în detalii aici.

Metoda Danilevschi permite determinarea valorilor proprii distincte. De asemenea se pot valorile proprii multiple, dar numai dacă vectorii proprii asociaţi nu sunt liniar independenţi. Deoarece metoda construieşte polinomul caracteristic (nu valorile proprii), pot apare erori mari în calculul numeric. Într-adevăr, erori mici în coeficienţii unui polinom pot produce abateri mari în valorile rădăcinilor

polinomului (inclusiv apariţia unor valori complexe în locul celor reale). Analiza erorilor se poate efectua ţinând seama că transformările se fac după schema

$$A_{i+1} = M_i^{-1} A_i M_i \,. \tag{6.50}$$

Rezultă că erorile se transmit după regula

$$\frac{\|\delta A_{i+1}\|}{\|A_{i+1}\|} \le \mu(M_i) \left(\frac{\|\delta A_i\|}{\|A_i\|} + \frac{\|\delta M_i\|}{\|M_i\|} + \frac{\left\|\delta M_i^{-1}\right\|}{\left\|M_i^{-1}\right\|} \right) . \tag{6.51}$$

Prin urmare, este util ca numărul de condiţionare al matricelor M_i să fie egal cu unitatea ($\mu(M_i) = 1$, într-o normă oarecare). Această condiţie nu este, în general, realizabilă, aşa cum rezultă din forma matricelor M_i. Dacă, spre exemplu, $a_{n-i+1,n-i}$ este pivot astfel încât

$$\|M_i\|_1 = 1 + \max_{j \ne n-i} \left| \frac{a_{n-i+1,j}}{a_{n-i+1,n-i}} \right| , \quad \|M_i\|_1 = |a_{n-i+1,n-i}| > 1 \,, \tag{6.52}$$

rezultă (folosind norma $\|\|_1$ în numărul de condiţionare)

$$\mu_1(M_i) = |a_{n-i+1,n-i}| + \max_{j \ne n-i} |a_{n-i+1,j}| > 1 \,. \tag{6.53}$$

În consecinţă, metoda se recomandă la determinarea valorilor şi vectorilor proprii de mici dimensiuni ($n \le 10$), prezentând avataje din punct de vedere al numărului de operaţii.

EXEMPLU. Pentru matricea de la exemplul anterior să se determine toate valorile proprii folosind metoda Danilevschi.

Rezolvare. Se aduce matricea A la forma Frobenius, folosind transformări elementare. În prealabil se aduce pivotul 3 în poziţia (3,2) prin permutarea coloanelor 1,2. Matricea de permutare este P_{12}. Notând

$$A_1 = P_{12} A P_{12} = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 1 & 3 \\ 2 & 3 & 1 \end{bmatrix} , \; M_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2/3 & 1/3 & -1/3 \\ 0 & 0 & 0 \end{bmatrix} ,$$

am obţinut o nouă matrice similară cu A. Se elimină elementele a_{31}şi a_{33} de pe ultima linie, operând cu coloana a doua. În locul elementului $a_{32} = 3$ se obţine valoarea 1, împărţind cu 3 toată coloana a doua. Toate operaţiile revin la înmulţirea la dreapta cu matricea M_1, formată cu multiplicatori. Se obţine

$$A_1 M_1 = \begin{bmatrix} -1/3 & 2/3 & 4/3 \\ 4/3 & 1/3 & 8/3 \\ 0 & 1 & 0 \end{bmatrix} .$$

În acest fel, s-au obţinut pe poziţia ultimei linii elemente corespunzătoare din matricea Frobenius. Pentru ca transformarea să fie similară se va înmulţi la stânga cu matricea

$$M_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 3 & 1 \\ 0 & 0 & 1 \end{bmatrix} ,$$

se obţine

$$A_2 = M_1^{-1} A_1 M_1 = \begin{bmatrix} -1/3 & 2/3 & 4/3 \\ 10/3 & 10/3 & 32/3 \\ 0 & 1 & 0 \end{bmatrix} .$$

În etapa următoare se elimină elementele 10/3 şi 32/3 de pe linia a doua din matricea A, operând cu coloana întâia. În poziţia (2,1) se obţine elementul 1. Totul revine la a înmulţi la dreapta cu matricea M_2, iar la stânga cu inversa ei M_2^{-1}

$$M_2 = \begin{bmatrix} -3/10 & -1 & -16/5 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \; M_2^{-1} = \begin{bmatrix} 10/3 & 10/3 & 32/2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

astfel încât

$$A_3 = M_2^{-1} A_2 M_2 = \begin{bmatrix} 3 & 14 & 8 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} = F,$$

A_3 fiind o matrice Frobenius. Valorile proprii sunt soluţiile ecuaţiei caracteristice

$$\lambda^3 - 3\lambda^2 - 14\lambda - 8 = 0 \, .$$

Se obţin valorile proprii

$$\lambda_1 = (5 + \sqrt{41}), \; \lambda_2 = -2, \; \lambda_3 = (5 - \sqrt{41})/2 \, .$$

Vectorii proprii ai matricei Frobenius F sunt

$$y_i = (\lambda_i^2, \lambda_1, 1)^T, \; i = 1, 2, 3.$$

Pentru a obţine vectorii proprii x_i ai matricei iniţiale A, se utilizează matricele de postmultiplicare

$$x_i = P_{12} M_1 M_2 y_i, \; i = 1, 2, 3.$$

Deşi s-a lucrat cu fracţii raţionale, s-a utilizat pivot pentru a atrage atenţia aupra tehnicii uzuale de lucru cu numere maşină afectate de erori de rotunjire.

6.5 Metodele QR şi LR

Metodele QR şi LR, pe care le vom prezenta în continuare, determină valorile şi vectorii proprii ale unei matrice pătrate A, prin aducerea acesteia la o matrice superior triunghiulară prin transformări de similitudine. Valorile proprii ale unei matrice triunghiulare T sunt chiar elementele de pe diagonală deoarece calculul polinomului caracteristic conduce la

$$P(\lambda) = \det(T - \lambda I) = \begin{vmatrix} t_{1,1} - \lambda & t_{1,2} & \dots & t_{1,n} \\ 0 & t_{2,2} - \lambda & \dots & t_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & t_{n,n} - \lambda \end{vmatrix} \tag{6.54}$$

şi dezvoltând după minori obţinem

$$\det(T - \lambda I) = (t_{1,1} - \lambda)(t_{2,2} - \lambda) \dots (t_{n,n} - \lambda)\,, \tag{6.55}$$

un polinom cu rădăcinile $t_{1,1}$, $t_{2,2}$, $\dots$, $t_{n,n}$.

Algoritmul QR reprezintă culminarea unui lung proces de căutare a unor algoritmi robuşti de determinare a tuturor valorilor proprii ale unei matrice şi are o importanţă deosebită în aplicaţii. În aplicarea practică a algoritmului QR apar o serie de etape intermediare ce pot crea impresia că metoda este destul de complicată. După cum vom vedea însă, esenţa algoritmului este *aplicarea metodei puterii simultan asupra mai multor vectori iniţiali.*

6.5.1 Rezultate teoretice preliminarii

Pentru prezentarea completă a acestor metode vom avea nevoie de o serie de rezultate teoretice. Vom include şi demonstraţiile acestor rezultate deoarece furnizează indicaţii asupra modului în care se pot construi algoritmi numerici.

Am văzut, în cadrul metodei Danilevschi, că nu orice matrice poate fi adusă la forma Frobenius. Acum urmărim aducerea la forma triunghiulară deci, apare imediat întrebarea dacă există o transformare de similitudine între o matrice oarecare A şi o matrice triunghiulară. Scopul principal al acestei secţiuni este să investigăm acest aspect.

Transformări unitare.

În plus, analiza de eroare de la metoda Danilevschi a arătat importanţa obţinerii unor transformări similare la care numărul de condiţionare să fie egal cu unitatea. Dacă matricele de transformare M sunt *unitare*, adică dacă $M^*M = I$ atunci vom avea în mod evident $\mu(M) = 1$ dacă folosim norma spectrală (5.150)

$$\mu(M) = \|M\|_2 \left\|M^{-1}\right\|_2 = \|M\|_2 \|M^*\|_2 = \sqrt{\rho(M^*M)\,\rho(MM^*)} = \sqrt{\rho(I)\rho(I)} = 1\,.$$

Prin M^* s-a notat *adjuncta* matricei M (vezi (5.151)).

Vom folosi una din notaţiile Q, U, R pentru matrice unitare. Valorile proprii ale unei matrice unitare au modulul egal cu unitatea. Într-adevăr, dacă efectuăm produsul scalar al vectorilor coloană din fiecare membru a relaţiei de valori proprii

$$Qx = \lambda x\,, \tag{6.56}$$

cu vectorul linie x^*Q^* rezultă

$$x^*Q^*Qx = x^*Q^*\lambda x\,. \tag{6.57}$$

Prin operaţiile de transpunere şi complex conjugare a relaţiei (6.56) rezultă

$$x^*Q^* = \bar{\lambda}x^*\,, \tag{6.58}$$

ce înlocuit în partea din dreapta a (6.57) conduce la

$$x^*Q^*Qx = \bar{\lambda}x^*\lambda x = \lambda\bar{\lambda}x^*x. \tag{6.59}$$

Dar Q este unitară $Q^*Q = I$, aşa că rezultă

$$x^*x = (\lambda\bar{\lambda})x^*x\,, \tag{6.60}$$

de unde $\lambda\bar{\lambda} = 1$, deci modulul unei valori proprii a lui Q este egal cu unitatea. Cum valorile proprii ale inversei $Q^{-1} = Q^*$ sunt $1/\lambda$ rezultă că şi acestea au modulul egal cu unitatea.

Reflectori elementari.

Să considerăm acum modalităţile de obţinere a unei matrice unitare. Există două procedee mai utilizate. Primul, denumit *procedeul Householder*, construieşte o matrice unitară pornind de la un vector coloană v prin

$$U = I - vv^*\,. \tag{6.61}$$

Vectorul v^* este un vector linie astfel încât produsul vv^* este o matrice. Să vedem ce condiţii trebuie impuse vectorului v pentru ca matricea U să fie unitară. Adjuncta lui U este

$$U^* = (I - vv^*)^* = I^* - \overline{(v\bar{v}^T)}^T = I - (\bar{v}v^T)^T = I - v\bar{v}^T = I - vv^*, \tag{6.62}$$

adică tot U. Se spune că matricea U este *autoadjunctă*. Avem

$$U^*U = (I - vv^*)(I - vv^*) = I - 2vv^* + vv^*vv^*\,. \tag{6.63}$$

Cum v este vector coloană şi v^* este un vector linie, v^*v este un produs scalar ce este chiar pătratul normei euclidiene a lui v

$$v^*v = \sum_{i=1}^{n} \bar{v}_i v_i = \|v\|_2^2\,. \tag{6.64}$$

Se obţine

$$U^*U = I + (\|v\|_2^2 - 2)vv^*\,, \tag{6.65}$$

deci matricea U este unitară dacă matricea vv^* este nulă sau dacă

$$\|v\|_2 = \sqrt{2}\,. \tag{6.66}$$

Prima posibilitate implică $v = 0$ şi nu prezintă interes pentru aplicaţii. O matrice unitară de forma (6.61) cu v ce satisface (6.66) se numeşte *reflector elementar*. Denumirea provine de la modul de transformare a coordonatelor

unui punct. Spre exemplu, în două dimensiuni ($n = 2$) dacă $v = [\sqrt{2}, 0]^T$ se obţine

$$U = I - \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}. \quad (6.67)$$

Coordonatele $[x, y]^T$ ale unui punct se transformă conform

$$U \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -x \\ y \end{bmatrix}, \quad (6.68)$$

adică s-a efectuat o reflexie faţă de axa Oy pe care vectorul v este ortogonal. În cazul general, vectorul v defineşte normala unui plan faţă de care se reflectă coordonatele unui punct.

În aplicaţii vom avea de evaluat produsul U^*A sau AU. S-ar părea că sunt necesare $O(n^3)$ operaţii aritmetice pentru aplicarea unui reflector elementar. Lucrurile nu stau aşa şi este esenţial pentru economicitatea metodei QR să observăm că reflectorul elementar U *nu trebuie calculat în mod explicit.* Putem evalua rezultatul AU prin

$$AU = A(I - vv^*) = A - Avv^* = A - wv^* \quad (6.69)$$

cu $w = Av$. Evaluarea lui w costă $O(n^2)$ operaţii şi efectuarea produsului exterior wv^* încă $O(n^2)$ pentru un total de $O(2n^2)$, *mult* mai mic decât $O(n^3)$. Dat fiind vectorul v, algoritmul de aplicare la dreapta a unei transformări Householder asupra matricei A este

$$\begin{array}{l|l} j = 1 : n & j = 1 : n \\ \quad \left[\begin{array}{l} w_j \leftarrow 0 \\ k = 1 : n \\ \quad [w_j \leftarrow w_j + a_{jk} v_k \end{array} \right. & \quad \left[\begin{array}{l} k = 1 : n \\ \quad [a_{jk} \leftarrow a_{jk} - v_j w_k \ . \end{array} \right. \end{array} \quad (6.70)$$

Analog evaluăm U^*A prin

$$U^*A = (I - vv^*)A = A - vv^*A = A - vw^T \quad (6.71)$$

cu $w^T = v^*A$. Algoritmul este o modificare simplă a celui de mai sus.

Rotaţii elementare.

Al doilea procedeu de obţinere a unei matrice unitare este cunoscut sub numele de *procedeu Givens* sau *procedeu Jacobi.* Şi acest procedeu are o semnificaţie geometrică, matricea R efectuând *rotaţia* coordonatelor unui punct în jurul unei axe oarecare. În acest caz matricea unitară R_{jk} se obţine din matricea unitate I prin modificarea a patru elemente conform

$$\begin{aligned} r_{j,j} = e^{i\alpha} \cos\theta, \quad r_{j,k} = e^{i\beta} \sin\theta, \\ r_{k,j} = -e^{-i\beta} \sin\theta, \quad r_{k,k} = e^{-i\alpha} \cos\theta \, . \end{aligned} \quad (6.72)$$

Se poate verifica uşor prin calculul produsului R^*R că matricea R este unitară. Parametrii reali α, β, θ definesc rotaţia efectuată, fiind vorba de o rotaţie de unghi θ. Dacă matricea A asupra căreia aplicăm o rotaţie Givens are toate elementele reale atunci $\alpha = \beta = 0$. Axa de rotaţie este definită de produsul vectorial al versorului de pe direcţia j cu cel de pe direcţia k.

Să considerăm cazul unei matrice A reale. Situaţia tipică de aplicare a unei rotaţii Givens este următoarea: dorim să anulăm un singur element subdiagonal din A, fie acesta $a_{jk} \neq 0$ cu $j > k$. Vom presupune că elementele subdiagonale de pe coloanele de indice $i < k$ au fost deja anulate. La premultiplicarea matricei A cu $R^*_{jk} = R^T_{jk}$ se vor produce modificări ale liniilor j şi k ale matricei. Spre exemplu, pentru $j = 4$, $k = 2$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & 0 & -s \\ 0 & 0 & 1 & 0 \\ 0 & s & 0 & c \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

$$= \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & ca_{22} - sa_{42} & ca_{23} - sa_{43} & ca_{24} - sa_{44} \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & sa_{22} + ca_{42} & sa_{23} + ca_{43} & sa_{24} + ca_{44} \end{bmatrix},$$

cu $c = \cos\theta$, $s = \sin\theta$. Condiţia ce trebuie satisfăcută de θ este $sa_{22} + ca_{42} = 0$ în exemplul de mai sus, sau

$$sa_{kk} + ca_{jk} = 0 \tag{6.73}$$

în cazul general. Ecuaţia trigonometrică (6.73) nu se rezolvă direct. Este mai economic să calculăm pe s, c prin

$$\begin{array}{l} \text{dacă } |a_{jk}| > |a_{kk}| \\ \quad \text{atunci } t \leftarrow -a_{kk}/a_{jk}; \quad s \leftarrow 1/\sqrt{1+t^2}; \quad c \leftarrow s\,t \\ \quad \text{altfel } t \leftarrow -a_{jk}/a_{kk}; \quad c \leftarrow 1/\sqrt{1+t^2}; \quad s \leftarrow c\,t \end{array} \tag{6.74}$$

cu o singură extragere de radical şi evitarea împărţirii la un număr foarte mic.

Ca şi în cazul transformării Householder, matricea R_{jk} nu se calculează explicit ca apoi să se efectueze produsul $R^*_{jk}A$, deoarece aplicarea algoritmului general de înmulţire a două matrice ar costa $O(n^3)$ operaţii. Se calculează *efectul* aplicării rotaţiei $R^*_{jk}A$ prin

$$\begin{array}{l} i = k : n \\ \quad [a_{ki} \leftarrow ca_{ki} - sa_{ji}; \quad a_{ji} \leftarrow sa_{ki} + ca_{ji} \end{array} \tag{6.75}$$

cu un cost de doar $O(4(n-k))$. În cazul postmultiplicării AR_{jk} se procedează în mod asemănător.

Un pas în aducerea la forma triunghiulară.

Acum că avem la dispoziţie procedee de construcţie a unei matrice unitare, să vedem modul în care acestea pot fi folosite pentru a aduce pe A la forma

triunghiulară. Prin analogie cu procesul de eliminare Gauss, ne propunem să eliminăm mai întâi elementele de sub diagonala principală, de pe prima coloană a matricei A. Trebuie să găsim o matrice unitară U ce realizează transformarea

$$U^*AU = \begin{bmatrix} \alpha & w_1 & \dots & w_{n-1} \\ 0 & a_{2,2}^{(1)} & \dots & a_{2,3}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{2,n}^{(1)} & \dots & a_{n,n}^{(1)} \end{bmatrix}. \tag{6.76}$$

Elementul α de pe poziţia $1,1$ trebuie să fie chiar o valoare proprie a lui A deoarece U^*AU are aceleaşi valori proprii ca şi A, iar calculul polinomului caracteristic al matricei U^*AU conduce la apariţia factorului $(\lambda - \alpha)$. Este convenabil să introducem o diviziune în porţiuni a matricei din dreapta sub forma

$$U^*AU = \begin{bmatrix} \lambda & w^T \\ 0_{n-1} & A_{n-1} \end{bmatrix} \tag{6.77}$$

unde λ este o valoare proprie a lui A, w un vector coloană de dimensiune $n-1$, 0_{n-1} vectorul nul de dimensiune $n-1$, iar A_{n-1} o matrice pătrată de ordinul $n-1$.

Prima coloană a unei matrice se obţine prin înmulţirea acesteia la dreapta cu vectorul $e^{(1)} = [1, 0, \dots, 0]^T$

$$U^*AUe^{(1)} = \begin{bmatrix} \lambda \\ 0_{n-1} \end{bmatrix} = \lambda e^{(1)}. \tag{6.78}$$

Dar $U^* = U^{-1}$ şi înmulţind la stânga cu U se obţine

$$AUe^{(1)} = \lambda Ue^{(1)}, \tag{6.79}$$

deci condiţia ca U să conducă la eliminarea elementelor subdiagonale de pe prima coloană a matricei A este ca $Ue^{(1)}$ să fie proporţional cu vectorul propriu al matricei A asociat valorii proprii λ. Notăm prin x acest vector propriu. Deoarece x este determinat până la o constantă multiplicativă putem presupune $\|x\|_2 = 1$ şi

$$Ue^{(1)} = \beta x, \quad \beta \neq 0, \quad \beta \in \mathbf{C}. \tag{6.80}$$

Relaţia (6.80) ne spune că trebuie să găsim o transformare a vectorului $e^{(1)}$ în vectorul x printr-o matrice unitară. Interpretările geometrice ale procedeelor Householder şi Givens ne furnizează imediat modalitatea de construcţie a lui U. Cum o reflexie sau o rotaţie nu vor modifica modulul unui vector va trebui să avem

$$\left\| e^{(1)} \right\|_2 = |\beta| \cdot \|x\|_2 = |\beta| = 1. \tag{6.81}$$

Dacă adoptăm procedeul Householder, matricea U este dată de $U = I - vv^*$ cu v vectorul normalei la planul faţă de care reflexia lui $y = \beta^{-1}e^{(1)}$ este chiar

Figura 6.1: Interpretarea geometrică a procedeului Householder.

x. Prin analogie cu cazul bidimensional (fig. 6.1) unde $x - y$ este normala la planul de reflexie, bănuim că

$$v = a(x - y)\,, \tag{6.82}$$

cu a o constantă ce este determinată din condiţia ca U să fie unitară

$$\|v\|_2^2 = 2 \Rightarrow a^2 = 2\,\|x - y\|_2^{-2}\,. \tag{6.83}$$

Intuiţia din cazul bidimensional este verificată pe cazul general prin calcul,

$$\begin{aligned} Ux - y &= (I - vv^*)x - y = x - vv^*x - y = x - a^2(x - y)(x^* - y^*)x - y \\ &= (x - y)[1 - a^2(x^*x - y^*x)]\,. \end{aligned}$$

Trebuie ca $Ux - y = 0$ pentru x arbitrar, deci impunem

$$1 - a^2(x^*x - y^*x) = 0. \tag{6.84}$$

Înlocuind a^2 din (6.83) se obţine

$$\|x - y\|_2^2 - 2(x^*x - y^*x) = 0, \tag{6.85}$$

sau, ţinând cont că $\|x - y\|_2^2 = \sqrt{(x - y, x - y)} = (x - y) \cdot (x^* - y^*)$ (vezi relaţia 5.153) şi că $x^*x = y^*y$,

$$y^*x = x^*y. \tag{6.86}$$

Relaţia (6.86) determină parametrul β,

$$(\beta^{-1}e^{(1)})^*x = x^*\beta^{-1}e^{(1)} \Rightarrow \beta x_1 = \overline{\beta x_1}\,, \tag{6.87}$$

unde x_1 este prima componentă a lui x. Condiţia de mai sus spune că βx_1 trebuie să fie o cantitate reală. Cum $|\beta| = 1$ rezultă

$$\beta = \bar{x}_1 / |x_1| \tag{6.88}$$

dacă $|x_1| \neq 0$. Dacă $x_1 = 0$ atunci luăm $\beta = 1$. Matricea U este complet determinată.

Un calcul asemănător se poate efectua pentru a determina şi matricea de rotaţie ce realizează eliminarea elementelor subdiagonale de pe prima coloană. Nu vom intra în detalii deoarece acest procedeu este de regulă mai costisitor din punct de vedere al operaţiilor aritmetice, datorită necesităţii de a evalua radicali. Excepţia de la această regulă generală este pentru o matrice cu un mare număr de zerouri deja existente sub diagonala principală. În acest caz se poate proceda după cum s-a exemplificat în prezentarea transformării Givens.

Teorema Schur.

Odată eliminaţi termenii subdiagonali de pe prima coloană calculul se poate repeta pentru prima coloană a matricei A_1 şi aşa mai departe, până la aducerea completă a matricei originale A la forma triunghiulară. Metoda de mai sus este enunţată formal în următoarea teoremă importantă.

Teoremă. (Schur) Orice matrice pătrată A de dimensiuni $n \times n$ este unitar similară cu o matrice triunghiulară T.

Demonstraţie. Vom proceda prin inducţie. Orice matrice de ordinul $n = 1$ este triunghiulară. Să considerăm acum teorema adevărată pentru orice matrice de ordinul $n-1$, spre exemplu pentru matricea A_{n-1} din (6.77) pentru care matricea unitară de aducere la forma triunghiulară este U_{n-1}

$$U_{n-1}^* A_{n-1} U_{n-1} = T_{n-1}\,. \tag{6.89}$$

Matricea

$$U_n = U \begin{bmatrix} 1 & 0_{n-1}^T \\ 0_{n-1} & U_{n-1} \end{bmatrix} \tag{6.90}$$

cu U construit ca mai sus realizează aducerea matricei A la forma tringhiulară. Într-adevăr, avem

$$\begin{aligned} U_n^* A U_n &= \begin{bmatrix} 1 & 0_{n-1}^T \\ 0_{n-1} & U_{n-1}^* \end{bmatrix} U^* A U \begin{bmatrix} 1 & 0_{n-1}^T \\ 0_{n-1} & U_{n-1} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0_{n-1}^T \\ 0_{n-1} & U_{n-1}^* \end{bmatrix} \begin{bmatrix} \lambda & w_{n-1} \\ 0_{n-1} & A_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0_{n-1}^T \\ 0_{n-1} & U_{n-1} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0_{n-1}^T \\ 0_{n-1} & U_{n-1}^* \end{bmatrix} \begin{bmatrix} \lambda & w_{n-1} U_{n-1}^* \\ 0_{n-1} & A_{n-1} U_{n-1} \end{bmatrix} = \begin{bmatrix} \lambda & w_{n-1} U_{n-1}^* \\ 0_{n-1} & T_{n-1} \end{bmatrix} \end{aligned}$$

unde ultima matrice este de formă triunghiulară. **2**

6.5.2 Algoritmi auxiliari

Metodele LR şi QR cuprind un număr de algoritmi subsidiari ce le fac să fie eficace. Să considerăm mai întâi aceste părţi componente a întregului algoritm LR sau QR.

Deflaţie.

Teorema Schur ne asigură că o transformare unitară, similară de la matricea A la o formă triunghiulară există. Însă suntem în situaţia neplăcută că trebuie să cunoaştem deja vectorii şi valorile proprii pentru a determina efectiv matricea de transformare. Modul efectiv în care se aplică rezultatele de mai sus este în reducerea ordinului matricei cu care se lucrează în momentul când una dintre valori proprii a fost izolată. Acest proces este cunoscut ca *deflaţie* şi cuprinde următoarele etape:

1. se calculează vectorul propriu x asociat valorii proprii λ;
2. se ia $\beta = x_1/|x_1|$ pentru $x_1 \neq 0$ sau $\beta = 1$ pentru $x_1 = 0$;
3. se ia $a = \sqrt{2}/\left\|x - \beta e^{(1)}\right\|$, $v = a(x - \beta e^{(1)})$ şi $U = I - vv^*$;
4. se calculează U^*AU şi se izolează submatricea A_{n-1} (6.77).

La sfârşitul unei etape de deflaţie se obţine o matrice mai mică asupra căreia se poate opera în continuare cu o economie de operaţii aritmetice faţă de cazul în care s-ar încerca determinarea transformării globale ce aduce pe A la forma triunghiulară.

Factorizarea QR.

Am văzut în 5.1.9 cum o matrice A se poate descompune într-un produs de două matrice, una inferior triunghiulară şi alta superior triunghiulară. Există multe alte tipuri de *factorizare*

$$A = BC \tag{6.91}$$

a unei matrice A. O clasă importantă o constituie *factorizările ortogonale*, la care vectorii coloană ai matricei B sunt ortogonali (vezi 1.4.2) doi câte doi. Printre factorizările ortogonale, cea mai utilă este factorizarea QR a lui Householder

$$A = QR \tag{6.92}$$

în care Q este o matrice unitară şi R o matrice superior triunghiulară. După cum sugerează aproprierea de nume, factorizarea QR este o parte esenţială a metodei QR de determinare a valorilor proprii.

Deoarece Q este unitară din (6.92) se obţine

$$Q^*A = R\,. \tag{6.93}$$

Ideea de bază a algoritmului de factorizare este construirea pas cu pas a matricei Q^* prin eliminarea succesivă a elementelor subdiagonale de pe coloanele matricei A

$$U_{n-1} \cdot \ldots \cdot U_2 U_1 A = R \tag{6.94}$$

unde U_1 elimină elementele subdiagonale de pe coloana 1, U_2 cele de pe coloana 2 şi aşa mai departe. Dacă A este de formă generală, cea mai eficientă metodă de construire a matricelor U_j $j \in \overline{1, n-1}$ este prin aplicarea reflectorilor elementari. Matricea Q este

$$Q = U_1 U_2 \cdot \ldots \cdot U_{n-1}, \tag{6.95}$$

unde am ţinut cont că $U_j = U_j^*$ pentru un reflector elementar.

Algoritmul de factorizare construieşte în mod explicit matricea R. Nu este nevoie de construirea explicită a matricei Q. Aceasta se memorează prin succesiunea de vectori v_j din definirea reflectorilor U_j

$$U_j = I - v_j v_j^* \,. \tag{6.96}$$

Tot calculul poate fi organizat astfel încât vectorii v_j să fie păstraţi în locaţiile din matricea A în care s-au creat elemente nule. Costul unei factorizări este de $O(n^3)$ operaţii pentru o matrice A oarecare.

6.5.3 Formularea metodelor QR şi LR

Suntem acum pregătiţi pentru prezentarea metodelor QR şi LR de calcul a valorilor proprii. Începem cu metoda QR. Am văzut că odată cunoscută o valoare proprie putem reduce ordinul problemei prin aplicarea tehnicii de deflaţie. O idee imediată este folosirea metodei puterii pentru a determina valoarea proprie cea mai mare în modul. Am putea reduce ordinul problemei, aplica din nou metoda puterii asupra noi matrice şi aşa mai departe. Să rescriem metoda puterii folosind notaţii ce sugerează generalizarea procedeului

$$\begin{aligned} z^{(s)} &= A q^{(s-1)} \\ q^{(s)} &= z^{(s)} / \left\| z^{(s)} \right\|_2 \\ \lambda &= \left[q^{(s)} \right]^* A q^{(s)} \,, \end{aligned} \tag{6.97}$$

pentru $s = 1, 2, \ldots$. Algoritmul porneşte cu un vector iniţial de normă egală cu unitatea $q^{(0)}$, şi efectuează normalizarea vectorului $z^{(s)}$ (aducerea la norma egal cu unitatea) la fiecare iteraţie. Valoarea proprie este estimată prin raportul Rayleigh (6.11), ţinându-se cont că $q^{(s)}$ este normalizat.

Să încercăm acum să aplicăm metoda puterii asupra mai multor vectori iniţiali

$$Q^{(0)} = \left[\begin{array}{cccc} q_1^{(0)} & q_2^{(0)} & \ldots & q_n^{(0)} \end{array} \right] \tag{6.98}$$

în speranţa obţinerii mai multor valori proprii *simultan*. În acest caz va trebui ca matricea $Q^{(0)}$ să fie ortogonală pentru a asigura liniar independenţa vectorilor de start. Algoritmul devine

$$\begin{aligned} Z^{(s)} &= A Q^{(s-1)} \\ Q^{(s)} R^{(s)} &= Z^{(s)} \\ T^{(s)} &= \left[Q^{(s)} \right]^* A Q^{(s)} \,. \end{aligned} \quad . \tag{6.99}$$

Modificarea esenţială este înlocuirea operaţiei de normalizare din (6.97) cu o descompunere ortogonală de tip QR. Se asigură astfel independenţa liniară a vectorilor asupra cărora va acţiona matricea A în iteraţia următoare, ceea ce ajută la accelerarea convergenţei. Valorile proprii apar acum pe diagonala matricei $T^{(s)}$ ce nu este altceva decât generalizarea raportului Rayleigh.

Nu vom intra în toate detaliile tehnice asupra convergenţei procedeului de mai sus. Putem însă afirma că şirul de matrice $T^{(s)}$ tinde către o matrice cvasitriunghiulară ce are aceleaşi valori proprii ca şi A. O matrice cvasitriunghiulară diferă de una triunghiulară doar prin blocuri izolate de dimensiune 2×2 grupate în jurul diagonalei principale. Aceste blocuri sunt de regulă asociate valorilor proprii complexe.

Să vedem legătura dintre $T^{(s)}$ şi $T^{(s-1)}$. Începem cu $T^{(s-1)}$

$$T^{(s-1)} = \left[Q^{(s-1)}\right]^* AQ^{(s-1)} = \left[Q^{(s-1)}\right]^* Z^{(s)} = \left[Q^{(s-1)}\right]^* Q^{(s)}R^{(s)}. \quad (6.100)$$

Cum un produs de matrice unitare este tot o matrice unitară, am obţinut

$$T^{(s-1)} = Q_T^{(s-1)}R_T^{(s-1)} \text{ cu } Q_T^{(s-1)} \equiv \left[Q^{(s-1)}\right]^* Q^{(s)}, \quad R_T^{(s-1)} \equiv R^{(s)}\,. \quad (6.101)$$

Pe de altă parte din $Q^{(s)}R^{(s)} = Z^{(s)} = AQ^{(s-1)}$ rezultă $R^{(s)} = \left[Q^{(s)}\right]^* AQ^{(s-1)}$ şi putem scrie $T^{(s)}$ ca

$$T^{(s)} = \left[Q^{(s)}\right]^* AQ^{(s)} = \left[Q^{(s)}\right]^* A \left[Q^{(s-1)}\right]^* Q^{(s-1)}Q^{(s)} = R_T^{(s-1)}Q_T^{(s-1)}\,. \quad (6.102)$$

Rezultă următoarea regulă de evaluare a matricei $T^{(s)}$: se calculează factorizarea QR a matricei $T^{(s-1)}$şi se inversează factorii

$$Q^{(s-1)}R^{(s-1)} = T^{(s-1)} \Rightarrow T^{(s)} = R^{(s-1)}Q^{(s-1)}\,. \quad (6.103)$$

În metoda LR calculele se desfăşoară asemănător

$$L^{(s-1)}R^{(s-1)} = T^{(s-1)} \Rightarrow T^{(s)} = L^{(s-1)}R^{(s-1)}\,. \quad (6.104)$$

Dezavantajul metodei LR constă în necesitatea efectuării de operaţii de pivotare pentru a preveni împărţirile la zero sau numere foarte mici în modul şi o convergenţă mai lentă comparativ cu metoda QR.

6.5.4 Reducerea numărului de operaţii la factorizare

Elementele şirului $T^{(s)}$ din (6.103) pot doar *converge* către o matrice cvasitriunghiulară, dar pe parcursul iteraţiilor au o formă oarecare. Aceasta face ca operaţia de factorizare QR să fie excesiv de costisitoare. Este convenabil ca iteraţiile QR să se efectueze asupra unei matrice ce are aceleaşi valori proprii ca

şi A, dar să prezinte o formă ce economiseşte din operaţiile aritmetice necesare factorizării. O astfel de formă este

$$H = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & \dots & h_{1,n-1} & h_{1,n} \\ h_{2,1} & h_{2,2} & h_{2,3} & \dots & h_{2,n-1} & h_{2,n} \\ 0 & h_{3,2} & h_{3,3} & \dots & h_{3,n-1} & h_{3,n} \\ 0 & 0 & h_{4,3} & \dots & h_{4,n-1} & h_{4,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & h_{n,n-1} & h_{n,n} \end{bmatrix}$$

denumită *formă Hessenberg superioară*. Analog matricea transpusă se spune că este de *formă Hessenberg inferioară*. Prin formă Hessenberg, fără vreun alt calificativ, vom înţelege forma superioară.

Algoritmul QR devine în acest caz

$$Q^{(s-1)}R^{(s-1)} = H^{(s-1)} \Rightarrow H^{(s)} = R^{(s-1)}Q^{(s-1)}\,. \tag{6.105}$$

Forma Hessenberg se păstrează la inversarea factorilor deoarece $Q^{(s-1)}$ se poate construi astfel încât să fie o matrice Hessenberg. Odată determinată forma Hessenberg iniţială, fiecare iteraţie QR va necesita $O(n^2)$ operaţii aritmetice.

Aducerea la forma Hessenberg prin matrice cu multiplicatori.

În cadrul algoritmului LR, aducerea la forma Hessenberg se efectuează prin matrice cu multiplicatori, care elimină elementele a_{jk} cu $j > k+1$ de pe fiecare coloană, începând cu prima. Spre exemplu, pentru eliminarea elementelor de pe coloana 1, matricea de multiplicatori este

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & -m_{3,2} & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -m_{n,2} & 0 & \dots & 1 \end{bmatrix}, \quad m_{i,2} = a_{i,1}/a_{2,1}, \tag{6.106}$$

pentru $i \in \overline{3,n}$. Se poate remarca similitudinea cu matriciele de la eliminarea gaussiană. Matricea se poate scrie ca

$$M_1 = I - m_2 e_2^T, \tag{6.107}$$

$$m_2 = \begin{bmatrix} 0 & 0 & m_{3,2} & \dots & m_{n,2} \end{bmatrix}^T, \quad e_2 = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \end{bmatrix}^T. \tag{6.108}$$

Inversa este

$$M_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & m_{3,2} & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & m_{n,2} & 0 & \dots & 1 \end{bmatrix} = I + m_2 e_2^T, \tag{6.109}$$

ceea ce se poate verifica prin calcul direct

$$M_1 M_1^{-1} = (I - m_2 e_2^T)(I + m_2 e_2^T) \tag{6.110}$$
$$= I - m_2 e_2^T + m_2 e_2^T - m_2 \left(e_2^T m_2\right) e_2^T = I\,. \tag{6.111}$$

Matricea M_1^{-1} va înmulţi, la dreapta, matricea A, pentru ca transformarea să fie similară şi valorile proprii să nu se modifice. Se observă că zerourile obţinute în coloana i de matricea $M_1 A$ se menţin şi în matricea similară $M_1 A M_1^{-1}$, deoarece prima coloană din matricea M_1^{-1} este identică cu prima coloană din matricea unitate. Acest lucru a fost posibil tocmai pentru că matricea M_1 are coloana multiplicatorilor deplasată cu un indice mai spre dreapta faţă de matricea de eliminare gaussiană. Rezultă că orice matrice A poate fi adusă la forma Hessenberg prin transformări similare (eventual folosind permutări) în număr finit.

Dacă elementul $a_{2,1}$ este nul, un alt element $a_{i,2}$ $i \in \overline{3,n}$ poate fi adus în poziţia $(1,2)$, prin permutări de linii. Evident, dacă toţi coeficienţii $a_{i,1} = 0$, $i \in \overline{2,n}$, rezultă că $a_{1,1}$ este valoare proprie şi procedeul se va aplica matricei rămase, de dimensiuni $(n-1) \times (n-1)$. Permutările de linii sau coloane sunt recomandate chiar dacă $a_{2,1} \neq 0$, pentru obţinerea de pivot parţial sau total. Inversa matricei de permutare a liniilor sau coloanelor k şi 1 este tot P_{k1} ($P_{k1}^{-1} = P_{k1}$). Regula este următoarea: dacă am permuta liniile k şi 1, este necesar pentru ca transformarea să fie similară, să permutăm şi coloanele de aceeaşi indici. Numărul de operaţii pentru aducerea la forma Hessenberg fără permutări este aproximativ $2n^3/3 - 5n^2/2$.

Aducerea la forma Hessenberg prin transformări similare unitare.

Matricea A se poate aduce la forma Hessenberg prin transformări similare ce folosesc reflectori sau rotaţii elementare. Aceste metode au avantajul unei condiţionări optime, deşi numărul de operaţii este mai mare decât în cazul folosirii multiplicatorilor. În cazul general este mai eficientă folosirea reflectorilor şi nu vom descrie decât acest procedeu.

Ideea de bază este aceeaşi ca la folosirea multiplicatorilor: se aduc rând pe rând coloanele matricei A la forma Hessenberg. După etapa $s-1$ coloanele 1, 2, ..., $s-1$ sunt în formă Hessenberg iar matricea are forma

$$A^{(s-1)} = \left[\begin{array}{cccccc|cc} a_{11}^{(s-1)} & a_{12}^{(s-1)} & \dots & a_{1,s-1}^{(s-1)} & a_{1,s}^{(s-1)} & \dots & a_{1,n}^{(s-1)} \\ a_{21}^{(s-1)} & a_{22}^{(s-1)} & \dots & a_{2,s-1}^{(s-1)} & a_{2,s}^{(s-1)} & \dots & a_{2,n}^{(s-1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{s,s-1}^{(s-1)} & a_{s,s}^{(s-1)} & \dots & a_{s,n}^{(s-1)} \\ \hline 0 & 0 & \dots & 0 & a_{s+1,s}^{(s-1)} & \dots & a_{s+1,n}^{(s-1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & a_{n,s}^{(s-1)} & \dots & a_{n,n}^{(s-1)} \end{array}\right]. \tag{6.112}$$

Să determinăm modul în care aducem coloana s la forma Hessenberg. Este comod să introducem o transcriere pe blocuri a matricei,

$$A^{(s-1)} = \left[\begin{array}{c|c} B & C \\ \hline D & E \end{array}\right], \tag{6.113}$$

unde B, C, D, E sunt submatrice din $A^{(s-1)}$. Submatricea B este de dimensiuni $s \times s$ şi este de formă Hessenberg. Submatricea C este de dimensiuni $(n-s) \times (n-s)$. Submatricea D este de dimensiuni $(n-s) \times s$ şi are elemente nenule doar pe ultima coloană. Fie d această coloană,

$$d = \left[\begin{array}{cccc} a_{s+1,s}^{(s-1)} & a_{s+2,s}^{(s-1)} & \dots & a_{n,s}^{(s-1)} \end{array}\right]^T. \tag{6.114}$$

Submatricea E este de dimensiuni $(n-s) \times (n-s)$. Transformarea unitară trebuie să lase neafectată partea deja adusă la forma Hessenberg, B. Forma matricei de transformare din etapa s este

$$U_s = \left[\begin{array}{c|c} I & 0 \\ \hline 0 & U \end{array}\right] \tag{6.115}$$

deoarece calculul produsului $U_s^* A^{(s-1)} U_s$ conduce la

$$\left[\begin{array}{c|c} I & 0 \\ \hline 0 & U^* \end{array}\right] \left[\begin{array}{c|c} B & C \\ \hline D & E \end{array}\right] \left[\begin{array}{c|c} I & 0 \\ \hline 0 & U \end{array}\right] = \left[\begin{array}{c|c} I & 0 \\ \hline 0 & U^* \end{array}\right] \left[\begin{array}{c|c} B & CU \\ \hline D & EU \end{array}\right] \tag{6.116}$$

$$= \left[\begin{array}{c|c} B & CU \\ \hline U^*D & U^*EU \end{array}\right]. \tag{6.117}$$

Pentru un reflector elementar $U^* = U$. Deducem condiţia de obţinere a formei Hessenberg în etapa s

$$Ud = \beta e^{(1)}, \tag{6.118}$$

cu $e^{(1)} = (1, 0, \dots, 0)^T$, un vector unitar cu $n-s$ componente. Aceasta este similară problemei (6.80), iar interpretarea geometrică a reflectorului Householder ne furnizează soluţia

$$U = I - vv^*, \quad v = a(d - \beta e^{(1)}), \tag{6.119}$$

cu $\beta = -(d_1/|d_1|)\ \|d\|_2$, $a = \sqrt{2}\ \left\|d - \beta e^{(1)}\right\|_2$, $d_1 = a_{s+1,s}^{(s-1)}$.

6.5.5 Accelerarea metodelor QR şi LR

Aplicarea factorizării QR sau LR asupra formei Hessenberg a unei matrice reduce numărul de operaţii aritmetice efectuate într-o iteraţie. O a doua cale de reducere a timpului de calcul este reducerea numărului de iteraţii. Iteraţiile QR şi LR tind să aducă forma Hessenberg la o formă cvasi-triunghiulară. Pentru a reduce numărul de iteraţii necesare atingerii acestei forme se pot folosi tehnici

de accelerare sugerate de modificările efectuate asupra metodei puterii pentru a obţine metoda puterii inverse sau cea a deplasării.

Cea mai utilizată tehnică este cea a *deplasării originii*. Se urmăreşte ameliorarea convergenţei atunci când apare un bloc 2×2 pe diagonală. Relaţiile de iterare QR devin

$$H^{(s)} - k_s I = Q^{(s)} R^{(s)} \tag{6.120}$$

$$H^{(s+1)} - k_s I = R^{(s)} Q^{(s)} \tag{6.121}$$

unde k_s este *deplasarea*. Deplasarea se poate alege ca fiind ultimul element de pe diagonală, $h_{n,n}^{(s)}$ sau rădăcina cea mai apropriată în modul de $h_{n,n}^{(s)}$ a ecuaţiei

$$\begin{vmatrix} h_{n-1,n-1}^{(s)} - k & h_{n-1,n}^{(s)} \\ h_{n,n-1}^{(s)} & h_{n,n}^{(s)} - k \end{vmatrix} = 0\,. \tag{6.122}$$

Odată asigurată izolarea unei valori proprii se trece la deflaţia matricei.

6.5.6 Calculul vectorilor proprii

Calculul vectorilor proprii pentru o matrice cvasisuperior triunghiulară se face uşor, valorile proprii fiind cunoscute. Fie matricea superior triunghiulară de dimensiuni 5×5

$$H_s = \begin{bmatrix} \lambda_1 & a_1 & b_1 & c_1 & d_1 \\ 0 & \lambda_2 & b_2 & c_2 & d_2 \\ 0 & 0 & \lambda_3 & c_3 & d_3 \\ 0 & 0 & 0 & \lambda_4 & d_4 \\ 0 & 0 & 0 & 0 & \lambda_5 \end{bmatrix} \tag{6.123}$$

formă la care presupunem că s-a ajuns după un număr s de iteraţii. Notăm mai departe cu y un vector propriu oarecare al matricei care satisface relaţia

$$H_s y = \lambda_i y,\ i \in \overline{1,5},\ y = (y_1, y_2, y_3, y_4, y_5)^T\,. \tag{6.124}$$

Pentru componentele y_i, se obţine sistemul

$$\begin{aligned} y_5(\lambda_i - \lambda_5) &= 0 \\ y_4(\lambda_i - \lambda_4) &= d_4 y_5 \\ y_3(\lambda_i - \lambda_3) &= d_3 y_5 + c_3 y_4 \\ y_2(\lambda_i - \lambda_2) &= d_2 y_5 + c_2 y_4 + b_2 y_3 \\ y_1(\lambda_i - \lambda_1) &= d_1 y_5 + c_1 y_4 + b_1 y_3 + a_1 y_2 \end{aligned} \tag{6.125}$$

Sistemul (6.125) poate determina componentele y_i, cel mult până la o constantă multiplicativă. Acest sistem este simplu, principala dificultate nu este legată de găsirea necunoscutelor ci mai degrabă de discernarea soluţiilor care apar.

Cazul valorilor proprii distincte.

In acest caz se obţin 5 vectori liniar independenţi. Spre exemplu, pentru $i = 1$, din primele 4 ecuaţii (6.125) rezultă

$$y_2 = y_3 = y_4 = y_5 = 0\,, \tag{6.126}$$

în timp ce ultima ecuaţie este identic verificată. Concluzia este: y_1 poate lua orice valoare diferită de zero (vectorii proprii sunt prin definiţie nenuli). Luând $y_1 = 1$ se poate scrie vectorul propriu corespunzător valorii λ_1

$$y^{(1)} = (1, 0, 0, 0, 0)^T \tag{6.127}$$

Pentru $i = 2$ ($\lambda_i = \lambda_2$), din (6.125) se obţine

$$y_3 = y_4 = y_5 = 0,\ y_1(\lambda_2 - \lambda_1) = a_1 y_2,\ y_2 \neq 0, \tag{6.128}$$

vectorul propriu corespunzător fiind

$$y^{(2)} = (y_1, 1, 0, 0, 0)^T,\ y_1 = a_1/(\lambda_2 - \lambda_1) \tag{6.129}$$

şi aşa mai departe.

Cazul valorilor proprii multiple.

Fie $\lambda_1 = \lambda_2 \neq \lambda_3 \neq \lambda_4 \neq \lambda_5$. Din primele 3 ecuaţii (6.125) se obţine

$$y_3 = y_4 = y_5 = 0 \tag{6.130}$$

iar din ultimele două

$$0 \cdot y_2 = 0,\ 0 \cdot y_1 = a_1 y_2\,. \tag{6.131}$$

Cele două egalităţi (6.131) conduc la concluzii diferite, în funcţie de coeficientul a_1, şi anume

$$a_1 = 0 \Rightarrow y_1, y_2 \text{ arbitrari, nu toţi nuli.} \tag{6.132}$$

Cu y_1, y_2 arbitrari şi y_3, y_4, y_5 nuli, se pot obţine doi vectori liniar independenţi, corespunzător valorilor proprii egale, λ_1, λ_2

$$y^{(1)} = (1, 0, 0, 0, 0)^T,\ y^{(2)} = (0, 1, 0, 0, 0)^T,\ a_1 = 0\,. \tag{6.133}$$

Pentru $a_1 \neq 0$ din (6.131) se obţine

$$y_2 = 0,\ y_1 \neq 0 \text{ (altfel arbitrar).} \tag{6.134}$$

Prin urmare, dacă $a_1 \neq 0$ nu se pot obţine doi vectori liniar independenţi corespunzător valorilor proprii egale λ_1, λ_2. Rezultă că sistemul de vectori proprii ai matricei H_s (şi deci nici ai matricei iniţiale A, similară H_s) nu pot forma o bază. Se spune că matricea este *defectivă*. Este de aşteptat că, în

urma erorilor de calcul numeric, natura vectorilor proprii să apără modificată, mai ales când valorile proprii sunt multiple sau foarte apropriate. Din acest punct de vedere, trebuie spus că procedeul QR este cel mai stabil, furnizând rezultatele cele mai apropriate de situaţia reală. Se observă că, determinarea vectorilor proprii se poate face suficient de comod din forma triunghiulară, fără a fi necesară reducerea la forma canonică Jordan (care este o matrice triunghiulară mai particulară). Aducerea la forma canonică Jordan nu este recomandată, această operaţie putând introduce erori mari datorită instabilităţii numerice. De fapt, în general, aducerea unei matrice (superior) triunghiulare la forma canonică Jordan prin transformări unitare sau chiar prin transformări simple stabilizate (cu pivot) nu este garantată.

Odată determinaţi vectorii proprii y ai matricei triunghiulare, vectorii proprii x ai matricei iniţiale A se deduc utilizând matricele de postmultiplicare aplicate lui A.

EXEMPLU. Să se aducă la forma superior Hessenberg, prin transformări similare matricea

$$A = \begin{bmatrix} 12 & -20 & 41 \\ 9 & -15 & -63 \\ 20 & 50 & 35 \end{bmatrix}.$$

Să se calculeze apoi factorizarea LR a matricei Hessenberg obţinute.

Rezolvare. Vom folosi transformări de eliminare. Avem de eliminat un singur element $a_{31} = 20$. Deoarece elementul $a_{21} = 9$ nu este pivot vom schimba între ele liniile 2 şi 3 şi coloanele 2 şi 3 pentru a obţine o matrice similară

$$A_1 = P_{23}AP_{23} = \begin{bmatrix} 12 & 41 & -20 \\ 20 & 35 & 50 \\ 9 & -63 & -15 \end{bmatrix}$$

Pentru a elimina elementul $a_{31} = 9$ vom folosi pivotul $a_{21} = 20$ şi multiplicatorul $m_{32} = a_{31}/a_{21} = 9/20$, ceea ce duce la matricele

$$M_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -9/20 & 1 \end{bmatrix}, \; M_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 9/20 & 1 \end{bmatrix}.$$

Se obţine

$$A_2 = M_1 A_1 M_1^{-1} = \begin{bmatrix} 12 & 32 & -20 \\ 20 & 115/2 & 50 \\ 0 & -765/8 & -75/2 \end{bmatrix} = H$$

de formă Hessenberg.

Pentru descompunerea LR a matricei H, trebuie eliminate elementele $h_{21} = 20$ şi $h_{32} = -765/8$. Deoarece $h_{11} = 12$ nu este pivot, se permută liniile 1 şi 2 şi se obţine

$$H_1 = P_{12}H = \begin{bmatrix} 20 & 115/2 & 50 \\ 12 & 32 & -20 \\ 0 & -765/8 & -75/2 \end{bmatrix}.$$

Rezultă multiplicatorul $l_{21} = -h_{21}/h_{11} = -3/5$ şi matricea de multiplicare la stânga L_1

$$L_1 = \begin{bmatrix} 1 & 0 & 0 \\ -3/5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ L_1H_1 = \begin{bmatrix} 20 & 115/2 & 50 \\ 0 & -5/2 & -50 \\ 0 & -765/8 & -75/2 \end{bmatrix}.$$

Pentru eliminarea elementului din poziţia (3,2) permutăm liniile 2 şi 3 pentru a obţine un pivot. Obţinem

$$P_{23}L_1H_1 = \begin{bmatrix} 20 & 115/2 & 50 \\ 0 & -765/8 & -75/2 \\ 0 & -5/2 & -50 \end{bmatrix}$$

$$L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -4/153 & 1 \end{bmatrix},\ R = \begin{bmatrix} 20 & 25/2 & 50 \\ 0 & -765/8 & -75/2 \\ 0 & 0 & -2500/51 \end{bmatrix}$$

$$L_2P_{23}L_1P_{12}H = R\,.$$

Mai departe, ţinând cont că $P_{23}P_{23} = I$ se deduce

$$L_2(P_{23}L_1P_{23})P_{23}P_{12}H = R$$

unde matricea $P_{23}L_1P_{23}$ similară cu L_1, o vom nota cu L_{11}

$$L_{11} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3/5 & 0 & 1 \end{bmatrix},\ L_2L_{11} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3/5 & -4/153 & 1 \end{bmatrix}.$$

Rezultă că s-a obţinut o descompunere LR pentru matricea H permutată

$$P_{23}P_{12}H = LR,\ L = (L_2L_{11})^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 3/5 & 4/153 & 1 \end{bmatrix}$$

în timp ce matricea H are descompunerea

$$H = L'R,\ L' = P_{12}P_{23}L = \begin{bmatrix} 3/5 & 4/153 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

Capitolul 7

Metode de optimizare

Metodele de optimizare sunt în general metode de descreştere, ce determină minimul unei funcţii U de n variabile reale care se numeşte *funcţie scop* sau *funcţie obiectiv*. De aici şi denumirea lor de *metode de minimizare a funcţiilor de mai multe variabile*. Evident, problema găsirii maximului revine la minimizarea funcţiei cu semn schimbat. Metodele de descreştere au *convergenţă globală*, adică permit găsirea soluţiei chiar dacă punctul de plecare este îndepărtat de soluţie.

Metodele de optimizare au un domeniu de aplicabilitate foarte larg. Pe de o parte, majoritatea fenomenelor naturii sau economice reprezintă compromisuri între cauze contradictorii, şi ca atare multe din problemele ingineriei, economiei, matematicii, statisticii, medicinei, dar mai cu seamă procesele decizionale se pot formula ca probleme de optimizare. Pe de altă parte, majoritatea metodelor numerice pot fi reformulate ca probleme de optimizare. Aceste reformulări duc uneori la obţinerea unor metode performante, cum ar fi cele pentru rezolvarea sistemelor de ecuaţii liniare, pe care le prezentăm în secţiunea 7.6.1 şi cele pentru rezolvarea sistemelor de ecuaţii neliniare pe care le vom prezenta în capitolul 8.1.

Figura 7.1: Minime pentru o funcţie de o singură variabilă.

Un punct de *extrem* (*minim* sau *maxim*) poate fi *global* (valoarea extremă din tot domeniul), sau *local*, (valoarea extremă dintr-un subdomeniu), poate fi *simplu* sau *multiplu* (fig. 7.1). În general, interesează minimul global, care se poate obţine: (a) repetând calculul pentru puncte de start diferite şi alegând apoi punctul în care valoarea funcţiei U este cea mai mică; (b) după obţinerea minimului local, se dă o perturbaţie într-o direcţie oarecare; dacă algoritmul revine în acelaşi punct, atunci acesta este un candidat serios pentru minim global.

Pentru ca un punct x^* să fie un minim local unic în cazul unei probleme de optimizare fără restricţii, trebuie îndeplinite *condiţiile de optimalitate*, care sunt condiţii *suficiente*

$$\nabla U(x^*) = 0,\ G(x^*) \equiv \nabla^2 U(x^*) > 0\ , \tag{7.1}$$

adică în punctul de minim gradientul funcţiei scop trebuie să fie nul iar matricea derivatelor parţiale de ordinul al doilea, numită *matrice hessian*, trebuie să fie *pozitiv definită*[1].

Clasificarea metodelor de optimizare poate fi făcută după mai multe criterii. Din punct de vedere al restricţiilor impuse variabilelor, avem probleme de optimizare *fără restricţii* şi *cu restricţii*. După felul funcţiei scop, avem probleme de optimizare *liniare*, în care atât funcţia cât şi restricţiile sunt liniare şi probleme de optimizare *neliniare*. După efortul de calcul al derivatelor, avem metode de tip *Newton* la care se calculează matricea hessian (care conţine derivatele parţiale de ordinul al doilea) şi vectorul gradient (derivatele de ordinul întâi), metode *cvasi-Newton* şi metode de *gradienţi conjugaţi*, la care se calculează numai derivatele parţiale de ordinul întâi, şi metode la care nu se calculează derivate parţiale.

[1] O matrice G se numeşte *pozitiv definită*, dacă oricare ar fi vectorul nenul y, are loc inegalitatea $y^T G y > 0$. În cazul unidimensional ($n = 1$) în locul matricei G avem derivata a doua, care, dacă este pozitivă, atunci graficul funcţiei "ţine apă", iar funcţia are un minim.

Metodele de optimizare sunt metode iterative. Valoarea aproximativă punctului unde funcţia U ia cea mai mică valoare se calculează iterativ cu formula

$$x^{k+1} = x^k + \alpha_k p^k,\ k = 1, 2, \dots\ , \tag{7.2}$$

unde p^k este o *direcţie de descreştere*, iar α_k un număr real pozitiv care se determină astfel încât $U(x^{k+1}) < U(x^k)$. Pentru start este necesar un punct de plecare $x^0 \in \mathbf{R}^n$.

7.1 Minimizarea în lungul unei direcţii

Minimizarea în lungul unei direcţii reprezintă o etapă a calculului minimului unei funcţii de mai multe variabile. Spre exemplu, scalarul α_k din relaţia (7.2) poate fi determinat ca o soluţie a problemei de optimizare $\alpha_k = \min_{\alpha}\{U(x^k + \alpha p^k)\}$, unde vectorii x^k şi p^k sunt cunoscuţi, iar funcţia U rămâne de o sigură variabilă, $U = f(\alpha)$.

Fie $f : \mathbf{R} \to \mathbf{R}$ funcţia reală de o variabilă reală al cărei minim dorim să-l determinăm. În cazul în care derivata f' este uşor de determinat analitic, problema determinării extremului funcţiei f este echivalentă cu rezolvarea ecuaţiei $f'(x) = 0$. Se verifică apoi care din soluţiile astfel determinate corespund unui minim al funcţiei f, calculând o valoare în imediata vecinătate a extremului şi comparând-o cu valoarea funcţiei în punctul de extrem. Atunci când minimizarea unidirecţională reprezintă o etapă a calculului minimului unei funcţii de mai multe variabile, derivata f' este în general imposibil de determinat analitic.

Determinarea minimului funcţiei f cuprinde două etape: (a) *localizarea minimului*, etapă în care se obţine un interval (a, b) care conţine minimul şi (b) *reducerea lungimii intervalului* care conţine minimul, până când lungimea acestuia devine suficient de mică, adică $|b - a| < e$, cu e impus. Se poate folosi şi o condiţie de lungime relativă a intervalului de forma $|b - a| < \varepsilon|a|$, având în vedere faptul că $\varepsilon \geq \sqrt{\varepsilon_m}$, unde ε_m reprezintă distanţa relativă minimă între două numere reprezentabile, şi care este de acelaşi ordin cu epsilon maşină ε_M (vezi capitolul 4).

Exemplu. Pentru a verifica afirmaţia de mai sus, aproximăm funcţia f folosind seria Taylor în jurul punctului a din vecinătatea punctului de minim

$$f(b) \cong f(a) + f''(a)(b - a)^2/2\ . \tag{7.3}$$

S-au neglijat termenii de ordinul trei şi mai mare şi s-a ţinut cont că în vecinătatea minimului $f' \approx 0$. Valorile a şi b nu pot fi luate oricât de apropiate decât dacă este îndeplinită condiţia

$$|f(b) - f(a)| > \varepsilon_m |f(a)|\ , \tag{7.4}$$

astfel încât reprezentările numerelor $f(a)$ şi $f(b)$ să fie diferite. Rezultă

$$|b - a| \cong \sqrt{2\varepsilon_m \frac{|f(a)|}{f''(a)}} = |a|\sqrt{\varepsilon_m}\sqrt{2\frac{|f(a)|}{a^2 f''(a)}}\ . \tag{7.5}$$

Figura 7.2: Localizarea minimului pentru o funcţie de o variabilă.

Dacă $\sqrt{2|f(a)|/(a^2 f''(a))} \sim 1$, atunci $|b-a| \sim |a|\sqrt{\varepsilon_m}$, iar condiţia pe care o impunem $|b-a| < \varepsilon|a|$ duce la limitarea $\varepsilon > \sqrt{\varepsilon_m}$. Spre exemplu, dacă $\varepsilon_m = 10^{-8}$, determinarea minimului o putem face cel mult până când lungimea intervalului ce conţine minimul devine mai micădecât $\sqrt{10^{-8}} = 10^{-4}$.

Localizarea minimului.

Pentru localizarea minimului este nevoie de cel puţin trei puncte (fig. 7.2), spre deosebire de localizarea soluţiei unei ecuaţii $f(x) = 0$, unde sunt necesare numai două puncte care îndeplinesc condiţia $f(a) \cdot f(b) < 0$. Având trei puncte $a < b < c$, minimul x_m este conţinut de intervalul (a, c) dacă $f(a) > f(b) < f(c)$. Fiind date două valori $a < b$ pentru care $f(a) > f(b)$, localizarea minimului se poate face după următorul algoritm:

Date: $a < b,\ f(a) > f(b)$
$f_a \leftarrow f(a);\ f_b \leftarrow f(b)$
Repetă
  $c \leftarrow b + k(b-a),\ f_c \leftarrow f(c)$
  dacă $f_c > f_b$
    atunci $x_m \in (a, c)$, **stop**
    altfel $a = b,\ b = c,\ f_a = f_b,\ f_b = f_c$.

Pasul de căutare nu trebuie să fie constant, ba dimpotrivă, este avantajos ca el să crească de la o etapă la alta astfel încât localizarea minimului să fie făcută cât mai repede: $h_{k+1} = k \cdot h_k$, cu $k > 1$.

Algoritmul prezentat se poate îmbunătăţi folosind o interpolare parabolică. Prin (a, f_a), (b, f_b) şi (c, f_c) se duce o parabolă şi se calculează punctul de minim u al acesteia. Dacă $u > c$, se recomandă ca lungimea $u - c$ să nu fie prea mare, spre exemplu $u - c < 50(c - b)$. Dacă $u \in (b, c)$ şi $f(b) > f(u) < f(c)$, atunci

minimul funcţiei f este între b şi c. În caz contrar, avem de a face cu un eşec al extrapolării parabolice.

Determinarea minimului.

Etapa de determinare efectivă a minimului x_m, se poate face fie prin reducerea lungimii intervalului care-l conţine, fie aproximând funcţia $f(x)$ cu o funcţie mai simplă $g(x)$ pentru care minimul se poate determina pe cale analitică; spre exemplu, prin cele trei puncte (a, f_a), (b, f_b), (c, f_c), se poate duce un polinom de interpolare de gradul al doilea, al cărui minim se calculează cu uşurinţă. Prima variantă este sigură, dar are dezavantajul unei convergenţe slabe (liniare). A doua variantă converge în general mult mai repede, dar există şi posibilitatea ca punctul calculat ca minim al parabolei să iasă din intervalul care conţine minimul. Un algoritm din prima categorie, de reducere a lungimii intervalului, cunoscut sub denumirea de *algoritmul secţiunii de aur*, este dat de (7.6) şi este descris în continuare.

Algoritmul secţiunii de aur.

Date: $a < b < c;\ f(a) > f(b) < f(c);\ \varepsilon > \sqrt{\varepsilon_M};\ w = 0.38197;\ w_1 = 1 - w$
$x_0 \leftarrow a;\ x_3 \leftarrow c;\ f_0 \leftarrow f(a);\ f_3 \leftarrow f(c)$
dacă $|c - b| > |b - a|$
  atunci $x_1 \leftarrow b;\ x_2 \leftarrow b + w(c - b)$
  altfel $x_2 \leftarrow b;\ x_1 \leftarrow b - w(b - a)$
$f_1 \leftarrow f(x_1);\ f_2 \leftarrow f(x_2)$
cât timp $|x_3 - x_0| > \varepsilon|x_1 + x_2|$
  dacă $f_2 < f_1$
    atunci $x_0 \leftarrow x_1;\ x_1 \leftarrow x_2;\ x_2 \leftarrow w_1 x_1 + w\, x_3$
      $f_0 \leftarrow f_1;\ f_1 \leftarrow f_2;\ f_2 \leftarrow f(x_2)$
    altfel $x_3 \leftarrow x_2;\ x_2 \leftarrow x_1;\ x_1 \leftarrow w_1 x_2 + w\, x_0$
      $f_3 \leftarrow f_2;\ f_2 \leftarrow f_1;\ f_1 \leftarrow f(x_1)$
dacă $f_1 < f_2$
  atunci $x_{\min} \leftarrow x_1;\ f_{\min} \leftarrow f_1$
  altfel $x_{\min} \leftarrow x_2;\ f_{\min} \leftarrow f_2$. (7.6)

Fie $a < b < c$ cu $f_a > f_b < f_c$, nu neapărat $b = (a + c)/2$. Dacă notăm $w = (b-a)/(c-a)$, atunci $1-w = (c-b)/(c-a)$. Se pune problema să alegem punctul $x \in (a, c)$ astfel încât să micşorăm intervalul care conţine punctul de minim. Să presupunem (b, c) acest intervalul de lungime mai mare unde vom alege pe x şi să notăm $z = (x - b)/(c - a)$. Tripletul care conţine minimul x_m va fi $a < b < x$, sau $b < x < c$, de lungimi raportate $(x - a)/(c - a) = w + z$, respectiv $(c - b)/(c - a) = 1 - w$. Strategia este de a minimiza cea mai dezavantajoasă posibilitate, lucru care se realizează impunând ca cele două lungimi să fie egale: $1 - w = w + z$, rezultând $z = 1 - 2w$. Cum această strategie a fost folosită şi la determinarea punctului b la pasul anterior, adică $(x - b)/(c - b) = (b - a)/(c - a) = w$, rezultă $x - b = w(c - b) = z(c - a)$, sau

$1 - w = (c - b)/(c - a) = z/w$. Rezultă în final ecuaţia $w^2 - 3w + 1 = 0$, cu soluţia $w = (3 - \sqrt{5})/2 \cong 0.38197$. Poziţia punctului x este acum determinată, $x = b + w(c - b) = c - (1 - w)(c - b)$, adică la distanţa $0.38197(c - b)$ de b şi la $0.61803(c - b)$ de punctul c. Lungimea intervalului care conţine minimul se reduce cu factorul 0.61803 de la o iteraţie la alta, valoare apropiată de 0.5 de la metoda bisecţiei.

Metoda secţiunii de aur se bazează pe strategia "pesimistă" a considerării celei mai dezavantajoase situaţii posibile, presupunând că minimul se află de fiecare dată în intervalul cel mai mare, astfel încât se impune ca lungimile celor două intervale să fie egale rezultând un algoritm lent dar sigur. Pentru mărirea vitezei de deplasare către minim, se poate construi o parabolă printre punctele (a, f_a), (b, f_b) şi (c, f_c) şi se calculează minimul acesteia (dar care poate să fie la fel de bine şi un maxim!)

$$x = b - \frac{1}{2}\frac{(b-a)^2[f(b) - f(c)] - (b-c)^2[f(b) - f(a)]}{(b-a)[f(b) - f(c)] - (b-c)[f(b) - f(a)]} \ . \tag{7.7}$$

De asemenea, se poate folosi o interpolare cu o parabolă cubică, caz în care se foloseşte şi derivata funcţiei f.

Un algoritm foarte utilizat, care combină robusteţea metoda secţiunii de aur cu viteza interpolării parabolice, este *algoritmul lui Brent*. Algoritmul foloseşte şase puncte a, b, u, v, t şi x, nu neapărat distincte, definite după cum urmează: a şi b delimitează intervalul care conţine minimul, x este punctul în care funcţia f are cea mai mică valoare calculată până la un moment dat, t este valoarea anterioară a lui x, v este valoarea anterioară a lui t, iar u este punctul în care s-a făcut ultima evaluare a funcţiei f. Interpolarea parabolică se face folosind punctele x, t şi v. Pentru ca rezultatul interpolării parabolice să fie acceptat, trebuie impuse câteva măsuri de siguranţă: (a) minimul calculat să fie în intervalul (a, b); (b) deplasarea faţă de ultima valoare care aproximează minimul lui f să fie mai mică dacât jumătatea deplasării anterioare, pentru a avea certitudinea că avem un proces convergent către punctul minim; (c) punctul de minim calculat u să nu fie foarte apropiat de o valoare anterior calculată p, adică $|u - p| > \varepsilon\, p$, cu $\varepsilon > 0$. Pentru oprirea iteraţiilor se foloseşte condiţia

$$|b - a| \leq e = 2|x|\varepsilon + \varepsilon_s \ , \tag{7.8}$$

adică lungimea itervalului care conţine punctul de minim să fie mai mică decât o lungime impusă e, iar ε_s este un număr mic (spre exemplu $\varepsilon_s = \varepsilon$) introdus pentru a evita nedeterminarea care apare pentru cazul în care punctul de minim este chiar $x = 0$. Condiţia (7.8) asigură precizia relativă de calcul pentru punctul de minim dată de relaţia

$$|x - x_{\min\ exact}| \leq \varepsilon |\, x| + \varepsilon_s \ , \tag{7.9}$$

unde reamintim că valoarea ε nu trebuie să fie mai mică decât rădăcina pătrată a lui epsilon maşină, adică $\varepsilon \geq \sqrt{\varepsilon_M}$.

Algoritmul conţine o primă parte destinată iniţializării variabilelor

$$
\begin{array}{l}
b \leftarrow c;\ f_b \leftarrow f_c;\ u \leftarrow b;\ f_u \leftarrow f_b;\ w = 0.381966;\ maxiter = 100 \\
\text{dacă } f_b < f_a \\
\quad \text{atunci } t \leftarrow b;\ f_t \leftarrow f_b;\ v \leftarrow a;\ f_v \leftarrow f_a \\
\quad \text{altfel } t \leftarrow a;\ f_t \leftarrow f_a;\ v \leftarrow b;\ f_v \leftarrow f_b \\
iter \leftarrow 1;\ ud \leftarrow 0;\ \delta x = b - a; \\
x \leftarrow 0.5(b+a);\ f_x \leftarrow f(x);\ e \leftarrow \varepsilon(2\,|x| + 1)
\end{array}
\tag{7.10}
$$

urmată de bucla iterativă principală

```
cât timp (b − a) > e şi iter ≤ maxiter
    x_m ← 0.5(b + a)
    dacă |δx| > 0.5ud sau u − a < e sau b − u < e
        atunci
            dacă x > x_m atunci δx ← w(a − x) altfel δx ← w(b − x)
            ud ← max(|b − x|, |a − x|)
        altfel
            r ← (x − t)(f_x − f_v); q ← (x − v)(f_x − f_t); p ← (x − v)q − (x − t)r
            δx ← −0.5p/(q − r);  ud ← |δx|
    f_u ← f(u); u ← x + δx
    dacă f_u ≤ f_x
        atunci
            dacă u ≥ x atunci a ← x altfel b ← x
            v ← t; t ← x; x ← u; f_v ← f_t; f_t ← f_x; f_x ← f_u
        altfel
            dacă u < x atunci a ← u altfel b ← u

        dacă f_u ≤ f_t sau t = x
            atunci
                v ← t; t ← u; f_v ← f_t;  f_t ← f_u
            altfel
                dacă f_u ≤ f_v sau v = x sau v = t   atunci v ← u; f_v ← f_u
    e ← ε(2 |x| + 1);  iter ← iter + 1
```

7.2 Metode de minimizare fără calculul derivatelor

Cea mai simplă modalitate de determinare a minimului unei funcţii reale de mai multe variabile reale, $U : \mathbf{R}^n \rightarrow \mathbf{R}$, constă în minimizarea funcţiei U de-a lungul direcţiilor definite de versorii e^i şi este dată de algoritmul (7.11). Algoritmul (7.11) se repetă până când se obţine minimul funcţiei U (se poate folosi o condiţie de forma (7.18)). Această metodă, numită şi *metoda căutării unidimensionale*, se dovedeşte ineficientă pentru că, fie nu este capabilă să atingă minimul, fie apropierea de acesta se face cu paşi mici.

Devine clar că pentru a obţine o metodă mai eficientă este necesar a găsi direcţii de deplasare mai eficiente. Astfel de direcţii sunt *direcţiile conjugate*.

$$\begin{array}{l} \text{Date: } x^0,\ U(x) \\ i = 1 : n \\ \quad \left[\ x^i = \min\limits_{\alpha \in \mathbf{R}} \{U(x^{i-1} + \alpha e^i)\} \ \ . \right. \end{array} \tag{7.11}$$

Direcţii conjugate.

Fie Φ forma pătratică definită de

$$\Phi\left(x\right) = U(x^k) + \sum_{j=1}^{n} (x_j - x_j^k)^T \left.\frac{\partial U}{\partial x_j}\right|_k + \frac{1}{2} \sum_{j,p=1}^{n} (x_j - x_j^k)^T \left.\frac{\partial^2 U}{\partial x_j \partial x_p}\right|_k (x_p - x_p^k) \ . \tag{7.12}$$

Forma (7.12) coincide cu primii trei termeni din dezvoltarea în serie Taylor a funcţiei $U(x)$ în vecinătatea punctului x^k. Notând cu g^k gradientul funcţiei U în punctul x^k şi cu G_k *matricea hessian* în acelaşi punct, expresia (7.12) se poate scrie vectorial

$$\Phi(x) = U_k + (x - x^k)^T g^k + \frac{1}{2}(x - x^k)^T G_k (x - x^k) \ , \tag{7.13}$$

iar gradientul funcţiei Φ este

$$\nabla\Phi = g^k + G_k(x - x^k) \ . \tag{7.14}$$

Punctul x^k s-a obţinut pentru valoarea α care asigură un minim pentru funcţia $\Phi(x^{k-1} + \alpha p^{k-1})$, deci vectorul g^k va fi perpendicular pe direcţia p^{k-1}, adică $(p^{k-1})^T \cdot g^k = 0$ (vezi 7.3). Deplasarea spre aproximaţia următoare x^{k+1} a minimului funcţiei Φ va trebui făcută astfel încât să nu se piardă câştigul obţinut pe direcţia p^{k-1}. Pentru aceasta trebuie ca gradientul funcţiei în punctul x^{k+1} să rămână perpendicular pe direcţia p^{k-1}, adică $(g^{k+1})^T \cdot p^{k-1} = 0$. Ţinând cont de (7.14), putem scrie $\nabla\Phi = g^{k+1} + G_k(x - x^{k+1})$. Scăzând cele două relaţii şi ţinând cont că deplasarea din x^k în x^{k+1} s-a făcut după direcţia p^k, rezultă

$$g^{k+1} = g^k + G(x^{k+1} - x^k) = g^k + \alpha_k G p^k \ , \tag{7.15}$$

unde $\alpha_k \in \mathbf{R}$. Înmulţind relaţia (7.15) cu $(p^{k-1})^T$, se obţine

$$(p^{k-1})^T g^{k+1} = (p^{k-1})^T g^k + \alpha_k (p^{k-1})^T G p^k \tag{7.16}$$

şi cum $(p^{k-1})^T g^{k+1} = (p^{k-1})^T g^k = 0$, rezultă condiţia care trebuie îndeplinită de direcţiile de deplasare

$$(p^{k-1})^T G p^k = 0 \ . \tag{7.17}$$

Două direcţii care îndeplinesc condiţia (7.17) se numesc *direcţii G-conjugate*. Pentru o formă pătratică Φ, folosind deplasări pe direcţii conjugate definite de (7.17), minimul funcţiei este atins după exact n deplasări. Pentru aceasta, mai trebuie îndeplinită o *condiţie necesară*: la fiecare etapă, minimizarea funcţiei Φ după direcţia p^k trebuie făcută *exact* pentru a asigura condiţia $(p^k)^T \cdot g^k = 0$. Pentru funcţii care sunt diferite de o formă pătratică, minimul nu se atinge după n deplasări, dar algoritmul are o convergenţă bună.

Rămâne problema determinării acestor direcţii fără a folosi matricea hessian G. Powell a fost primul care a găsit o modalitate de a obţine un set de n direcţii conjugate folosind următorul algoritm:

Date: x^0, $U(x)$, ε, n, *maxiter*
$l = 1 : maxiter$
  $u^i = e^i$ Se iniţializează direcţiile de căutare după fiecare $n+1$ iteraţii.
  $U_0 \leftarrow U(x^0)$
  $k = 1 : (n+1)$
    $i = 1 : n$
      $x^i \leftarrow \min\limits_{\alpha \in \mathbf{R}} \{U(x^{i-1} + \alpha\, u^i)\}$ Se minimizează U în lungul direcţiilor u^i.
    $i = 1 : (n-1)$
      $u^i \leftarrow u^{i+1}$
    $u^n \leftarrow x^n - x^0$ Se înlocuieşte ultima direcţie a setului cu direcţia care trece prin punctele x^nşi x^0.
    $x^0 \leftarrow \min\limits_{\alpha \in \mathbf{R}} \{U(x^n + \alpha\, u^n)\}$ Se reiniţializează punctul de start.
  dacă $|U - U_0| < \varepsilon(1 + |U|)$ atunci *minim determinat* **stop.**

Condiţia de oprire folosită este de forma

$$|U_{k+1} - U_k| < \varepsilon(1 + |U_{k+1}|) \ , \tag{7.18}$$

unde am văzut la 7.1 că $\varepsilon \geq \sqrt{\varepsilon_M}$, ε_M fiind epsilon maşină. Condiţia (7.18) asigură precizia relativă ε pentru valoarea minimului $U(x^*)$, dacă acesta ia valori depărtate de zero. Dacă valoarea minimă este chiar zero sau foarte apropiată de zero, atunci testul (7.18) asigură precizia absolută ε.

Powell a demostrat încă din 1964 că, pentru o formă pătratică Φ (7.13), k iteraţii din algoritmul prezentat mai sus produc un set de direcţii u^i dintre care ultimele k sunt *G-conjugate*, cu condiţia ca minimizările în lungul direcţiilor u^i să fie făcute exact. O iteraţie se obţine după $n+1$ minimizări după direcţiile date de vectorii u^i, $i = \overline{1,n}$, şi $x^n - x^0$. Există tendinţa ca direcţiile construite să devină liniar dependente. Acest lucru poate fi evitat (a) printr-o iniţializare a direcţiilor $u^i = e^i$ la fiecare $n+1$ iteraţii (aşa cum s-a şi procedat mai sus), sau (b) în loc de a renunţa fără discernământ la ultima direcţie, se renunţă la direcţia care a produs cea mai mare descreştere.

Revenind la relaţia (7.13), să observăm că pentru a defini o formă pătratică este nevoie de un număr de parametrii de ordinul n^2. Mai exact, matricea G fiind simetrică, suma elementelor distincte este $1+2+\ldots+n = n(n+1)/2$. Dacă ţinem cont şi de cele n componente ale gradientului, rezultă $n(n+3)/2$ parametrii

ce definesc o formă pătratică. În timp ce matricea hessian G este constantă, gradientul se modifică, dar este nul în punctul de minim. În concluzie, indiferent de metoda pe care o vom folosi, cu sau fără calculul derivatelor funcţiei scop, chiar pentru o formă pătratică, determinarea minimului înainte de obţinerea celor $O(n^2)$ parametrii este cel mult întâmplătoare.

Diferenţa esenţială între metodele de optimizare constă în modul în care se calculează aceşti parametrii. Spre exemplu, în cazul unei forme pătratice, folosind matricea hessian, determinarea minimului revine la rezolvarea sistemului de ecuaţii liniare $G \cdot (x - x^0) = -g^0$ (vezi relaţia 7.14). Evaluarea derivatelor, când acest lucru este posibil fără dificultăţi suplimentare, va constitui cea mai eficientă modalitate de obţinere a informaţiei necesare determinării minimului.

7.3 Metoda gradientului

Este cea mai simplă metodă de descreştere care foloseşte derivatele funcţiei U. Se bazează pe observaţia că suprafeţele cu $n-1$ dimensiuni de forma

$$U(x) = K_i = \text{constant},\ i = 1, 2, 3, \dots\ , \tag{7.19}$$

denumite şi *suprafeţe de nivel* pentru funcţia $U(x)$, sunt astfel dispuse încât constantele K_i iau valori crescătoare când ne deplasăm pe direcţia pozitivă a vectorului gradient

$$g \equiv grad\, U \equiv \nabla U \equiv \left(\frac{\partial U}{\partial x_1}, \frac{\partial U}{\partial x_2}, ..., \frac{\partial U}{\partial x_n} \right)^T . \tag{7.20}$$

Din cele trei notaţii uzuale din (7.20), o vom prefera pe prima, adică în mod curent vom nota gradientul cu g.

Deoarece g este variabil, deplasarea trebuie să fie suficient de mică pentru ca plecând dintr-un punct oarecare, să întâlnim suprafeţe de nivel superioare. Dacă, dimpotrivă, ne deplasăm în sens opus $(-g\ )$, vom întâlni constante de nivel K_i cu valori descrescătoare.

EXEMPLU. Fie funcţia $U = x^2 + y^2$. Suprafeţele de nivel sunt cilindri coaxiali cu axa Oz iar $\sqrt{K_i}$ reprezintă razele cercurilor de intersecţie cu planul xOy. Pe de altă parte, gradientul este $g \equiv \nabla U = 2(\vec{i}x + \vec{j}y)$, $\vec{i}$ şi $\vec{j}$ fiind versorii axelor Ox şi Oy. Se observă că în orice punct gradientul este orientat în sensul creşterii razei, deci a creşterii constantelor K_i (fig. 7.3).

Pentru determinarea minimului funcţiei U, construim şirul de iteraţii

$$x^{k+1} = x^k - \alpha_k g^k,\ \ U(x^{k+1}) < U(x^k)\ , \tag{7.21}$$

unde $x^k = (x_1^k, x_2^k, ..., x_n^k)^T$, iar

$$g^k = \left(g_1^k, g_2^k, ..., g_n^k\right)^T = \left(\frac{\partial U(x^k)}{\partial x_1}, \frac{\partial U(x^k)}{\partial x_2}, ..., \frac{\partial U(x^k)}{\partial x_n} \right)^T \tag{7.22}$$

Figura 7.3: Gradientul funcţiei $U = x^2 + y^2$.

este vectorul gradient în x^k şi α_k un scalar pozitiv. Direcţia $p^k = -g^k$ este o direcţie de descreştere pentru funcţia U în punctul x^k (este chiar direcţia de *maximă* descreştere în x^k). Determinarea scalarului α_k se face cu una din metodele prezentate în 7.1. Dacă minimul α_k se determină exact, atunci gradientul g^{k+1} este perpendicular pe g^k, adică

$$x^{k+1} = x^k - \alpha_k^* g^k,\ (g^k)^T \cdot g^{k+1} = 0\ . \tag{7.23}$$

Ultima egalitate (7.23) exprimă faptul că punctul în care produsul scalar al gradienţilor g_k şi g_{k+1} este nul reprezintă un punct de *minim unidirecţional*. Într-adevăr, dacă vectorul g^k ar avea o componentă diferită de zero după direcţia $-g^{k+1}$, atunci funcţia U ar scădea în continuare, ceea ce contrazice definiţia minimului.

Metoda gradientului utilizată cu valoarea optimă pentru scalarul α_k (caz în care spunem că am folosit un *pas optimal* sau *pas Cauchy*), are o *viteză de convergenţă liniară*, adică distanţa la punctul de minim tinde spre zero ca o progresie geometrică de raţie subunitară (vezi 7.5). Convergenţa devine lentă, de regulă, când numărul de variabile n creşte şi de asemenea, în vecinătatea minimului când vectorul g tinde către zero. Metoda de căutare a minimului după direcţia $-g^k$, utilizând un singur parametru α_k, a fost descrisă în 7.1.

În continuare, dăm un algoritm general pentru metodele de optimizare care

folosesc gradientul funcţiei scop:

$$
\begin{array}{l}
\text{Date: } x^0,\ U(x),\ \nabla U(x),\ \varepsilon,\ \textit{maxiter} \\
x \leftarrow x^0;\ U_k \leftarrow U(x^0);\ g^k \leftarrow \nabla U(x^0) \\
p \leftarrow g^k \qquad \text{Iniţializarea direcţiei de descreştere.} \\
i = 1 : \textit{maxiter} \\
\left[\begin{array}{l}
x \leftarrow \min\limits_{\alpha \in \mathbf{R}} \{U(x + \alpha p)\} \\
U_{k+1} \leftarrow U(x);\ g^{k+1} \leftarrow \nabla U(x) \\
\text{dacă } U_{k+1} \geq U_k \text{ atunci } \textit{eşec } \textbf{stop}. \\
[\textit{test convergenţă}] \\
[\textit{actualizarea direcţiei de descreştere } p] \\
U_k \leftarrow U_{k+1}\ .
\end{array} \right.
\end{array}
\tag{7.24}
$$

Algoritmii care folosesc calculul gradientului funcţiei scop U se deosebesc prin modul de calcul al direcţiei de descreştere p. Câteva precizări sunt necesare în cele ce urmează. O primă problemă o constituie alegerea metodei de minimizare unidirecţională, pentru care recomandăm algoritmul lui Brent. O a doua problemă o constituie precizia cu care trebuie efectuată minimizarea unidirecţională. Unele metode (cum ar fi metoda gradientului) nu necesită un calcul precis, în timp ce convergenţa altor metode impune un calcul exact al minimului unidirecţional (metodele cvasi-Newton, metodele de gradient conjugat). Un criteriu practic pentru o minimizare unidirecţională aproximativă este legat de proiecţia direcţiei de descreştere pe direcţia gradientului (care se numeşte *derivată direcţională*)

$$
|(p^k)^T \nabla U(x^k + \alpha_k p^k)| \leq \eta |(p^k)^T g^k|,\ 0 \leq \eta < 1\ . \tag{7.25}
$$

Pentru $\eta = 0$ avem $(p^k)^T g^{k+1} = 0$, adică minimizarea unidirecţională s-a făcut exact. Pe lângă criteriul (7.25) se poate impune o *condiţie de descreştere suficientă* de forma

$$
U_{k+1} - U_k \leq -\mu \alpha_k (g^k)^T p^k\ . \tag{7.26}
$$

Valori uzuale: $\mu \in [10^{-5}, 10^{-1}]$, $\eta \in (\mu, 1)$.

O a treia problemă o constituie testul de convergenţă. În general, trei criterii sunt mai des folosite:

$$
\begin{array}{ll}
1. & \|x^{k+1} - x^k\| \leq \varepsilon_x (1 + \|x^{k+1}\|) \\
2. & \|U_{k+1} - U_k\| \leq \varepsilon_U (1 + \|U_{k+1}\|) \\
3. & \|\nabla U(x^{k+1})\| \leq \varepsilon_g\ .
\end{array}
\tag{7.27}
$$

Primul criteriu arată că primele t cifre semnificative ale aproximaţiilor punctul de minim din două iteraţii succesive sunt identice, dacă $\varepsilon_x = 10^{-t}$, în timp ce cel de-al doilea criteriu arată că primele t cifre semnificative ale valorii funcţiei scop sunt identice, $\varepsilon_U = 10^{-t}$. S-au luat măsuri de precauţie pentru cazul în care punctul de minim este chiar $x = 0$ sau valoarea minimă este $U = 0$, scriind spre exemplu $\|x^{k+1} - x^k\| \leq \varepsilon_x \|x^{k+1}\| + \varepsilon_s$ în loc de $\|x^{k+1} - x^k\| \leq \varepsilon_x$.

Criteriului 3 trebuie folosit cu atenţie, deoarece satisfacerea lui nu înseamnă în mod obligatoriu atingerea unui punct de minim, ci poate fi şi un punct de maxim sau punct şa (un punct de inflexiune în cazul unidimensional). Trebuie reamintit (vezi 7.1) că valorile impuse pentru precizia relativă trebuie să îndeplinească condiţiile $\varepsilon_x \leq \sqrt{\varepsilon_M}$, $\varepsilon_U \leq \sqrt{\varepsilon_M}$, iar pentru criteriul 3, $\varepsilon_g \leq \sqrt[3]{\varepsilon_M}$, unde ε_M este epsilon maşină..

7.4 Metoda Newton

Aşa cum s-a precizat, metoda gradientului are o viteză de convergenţă liniară, adică la fiecare iteraţie distanţa la punctul de minim scade ca termenii unei progresii geometrice de raţie subunitară. Acest lucru se mai poate scrie şi în funcţie de comportarea raportului distanţelor până la punctul de minim, notat cu x^*, pentru două iteraţii succesive, k şi $k+1$

$$\lim_{k\to\infty} \sup \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|^r} = c \ . \tag{7.28}$$

Dacă $r = 1$ şi $c \in (0,1)$, atunci convergenţa este *liniară*. Dacă c este apropiat de unitate, convergenţa este lentă. Dacă $r = 1$ şi $c = 0$, avem *convergenţă superliniară*. Dacă $r > 1$ şi c este finit, r se numeşte *ordin de convergenţă*. Pentru $r = 2$ convergenţa este *pătratică*.

Metoda Newton se bazează pe aproximarea funcţiei de minimizat $U(x)$, la iteraţia k, cu o formă pătratică $\Phi_k\ (x)$ definită de (7.13). Funcţia Φ_k are un minim $x = x^-$ dacă matricea G_k este *pozitiv definită*[1] , adică dacă pe orice direcţie ne-am îndepărta de punctul x^- există inegalitatea $\Phi_k(x) - \Phi_k(x^-) > 0$ într-o vecinătate suficient de mică. Punctul de minim $x = x^-$, este un *punct staţionar*, adică derivatele parţiale de ordinul întâi, deci gradientul $\nabla\Phi_k$, se anulează în acest punct. Prin urmare, pentru valori suficient de mici ale distanţei până la minim $\|x - x^-\| < \varepsilon$, putem scrie ţinând cont de (7.13)

$$\Phi_k(x) - \Phi_k(x^-) \cong \tfrac{1}{2}(x - x^-)^T G_k (x - x^-) > 0 \ . \tag{7.29}$$

Pe de altă parte, rezolvarea sistemului $\nabla\Phi_k = 0$, se poate face utilizând metoda iterativă Newton (vezi 8.2)

$$G_k(x^{k+1} - x^k) = -g^k \Rightarrow \ x^{k+1} = x^k - G_k^{-1} g^k = x^- \ . \tag{7.30}$$

Pentru determinarea punctului de minim (x^- reprezintă aproximaţia x^{k+1} pentru minimul x^* al funcţiei U) este necesară rezolvarea unui sistem de n ecuaţii liniare cu n necunoscute, adică circa $n^3/3 + n^2$ operaţii, în cazul când folosim metoda de eliminare a lui Gauss (vezi capitolul 5). Din a doua egalitate (7.30), se observă că direcţia de deplasare $x^- - x^k$ spre punctul de minim nu este direcţia $-g^k$ a metodei gradientului, aceasta fiind modificată prin înmulţire cu inversa matricei hessiene, G_k^{-1}. Direcţia de descreştere p^k dată de relaţia

$$p^k = -G_k^{-1} g^k, \ (g^k)^T p^k < 0 \ , \tag{7.31}$$

Figura 7.4: Deplasarea către minim.

pentru fiecare iteraţie k, se numeşte *direcţie Newton*. Proiecţia direcţiei Newton după direcţia gradientului este negativă deoarece matricele G_k şi G_k^{-1} sunt pozitiv definite

$$(g^k)^T p^k = -(g^k)^T G_k^{-1} g^k < 0 \ , \tag{7.32}$$

inegalitatea $(g^k)^T p^k < 0$ fiind definitorie pentru orice direcţie de descreştere p^k.

Ca şi în cazul metodei Newton pentru sisteme neliniare (vezi capitolul 5), aproximarea cu o formă pătratică conduce la rezultate bune dacă *aproximaţia iniţială este suficient de apropiată de soluţie*. Aceasta înseamnă că, dacă hessiana există şi este pozitiv definită, începând cu un anumit x^k suficient de aproape de minimul $x = x^*$, metoda Newton de descreştere converge iar convergenţa este pătratică.

În forma prezentată, metoda Newton are o serie de neajunsuri. Astfel, dacă într-un anumit punct x^k, hessiana G_k nu este pozitiv definită, direcţia Newton nu mai este o direcţie de descreştere şi se poate ca $U_{k+1} > U_k$ (punctul A, figura 7.4). În zonele în care funcţia U are o variaţie plată, putând fi aproximată local cu o funcţie de gradul 1 (hiperplan), hessiana se anulează ($G = 0$), iar ecuaţia (7.30) nu are soluţie (punctul L, figura 7.4). Dacă în aceste cazuri introducem o matrice pozitiv definită în locul hessianei, deplasarea spre minimul x^* va continua şi în aceste situaţii. În cazul unidimensional, aceasta revine la a urma în punctele A şi L direcţia parabolelor tangente la funcţia y de minimizat, dar având tot timpul o curbură pozitivă, direcţiile modificate obţinute ducându-ne în punctele m_A şi m_L respectiv.

Având la bază metoda Newton, au fost dezvoltate o serie de metode care păstrează convergenţa locală bună (pătratică), dar prezintă şi o convergenţă globală, eliminând neajunsurile prezentate anterior. Dintre acestea, prezentăm în continuare *metoda regiunii de încredere în model*.

Algoritmul metodei regiunii de încredere în model (7.35) se bazează ca şi metoda Newton pe aproximarea funcţiei scop $U(x)$ cu o formă pătratică $\Phi(x)$ dată de (7.13). Fiind vorba de o aproximare valabilă local, se stabileşte o *zonă de încredere* în model, adică zona în care se admite că Φ aproximează bine pe U. În această zonă care este o sferă cu centrul în x^k de rază δ_k, se determină punctul de minim pentru $\Phi(x)$. Dacă minimul aproximantei $\Phi(x)$ se află în afara regiunii de încredere, acesta nu este luat în considerare. Rămân de rezolvat două probleme: mărimea razei δ_k a regiunii de încredere şi determinarea direcţiei de descreştere p^k.

Mărimea razei regiunii de încredere la pasul $k+1$ se alege în funcţie de valoarea anterioară şi de raportul r_k dintre reducerea efectivă şi cea prezisă,

$$r_k = [U(x^{k+1}) - U(x^k)]/[\Phi(x^{k+1}) - \Phi(x^k)] \ . \tag{7.33}$$

Pentru valori mici ale raportului r_k (evident $r_k > 0$ dacă ne deplasăm spre minim), se ia $\delta_{k+1} < \delta_k$, în timp ce pentru valori care tind către unitate se poate lua $\delta_{k+1} > \delta_k$.

Direcţia de căutare p^k se determină plecând de la direcţia Newton definită de (7.31), rescrisă în forma

$$(G_k + \lambda I)p^k = -g^k \ , \tag{7.34}$$

unde λ se alege astfel încât matricea $G_k + \lambda I$ să fie pozitiv definită, evitându-se astfel una din situaţiile nedorite care apar în aplicarea metodei Newton. Pentru $\lambda = 0$ regăsim metoda Newton dacă noua aproximaţie a punctului de minim nu iese în afara regiunii de încredere, adică dacă $\|p^k\|_2 < \delta_k$. Pentru $\lambda \geq 0$ se caută o direcţie de descreştere pentru care minimul aproximantei Φ să se afle chiar la frontiera regiunii de încredere, adică $\|p^k\|_2 = \delta_k$. Deoarece matricea $G_k + \lambda I$ este pozitiv definită, rezolvarea sistemului de ecuaţii (7.34) se face mai eficient folosind descompunerea Cholesky $G_k + \lambda I = R^T R$ (vezi capitolul 5). Numărul maxim de iteraţii *maxiter* în care apreciem că trebuie găsit minimul se alege cu atât mai mare cu cât numărul n de variabile ale funcţiei scop U este mai mare. Numărul de iteraţii np pentru calculul direcţiei de descreştere nu trebuie să fie mai mare decât 3, pentru ca efortul de calcul depus pentru o iteraţie să nu fie exagerat de mare.

Algoritmul regiunii de încredere în model.
Date: x^0, $U(x)$, $\nabla U(x)$, $G(x)$, μ, η, γ_1, γ_2, δ_0, λ_0, $\varepsilon, \varepsilon_p$, *maxiter*, *np*
[Valori uzuale pentru constante: $\mu = 0.25$, $\eta = 0.75$, $\gamma_1 = 0.5$, $\gamma_2 = 2$]
$x \leftarrow x^0$; $\delta \leftarrow \delta_0$; $\lambda \leftarrow \lambda_0$; $U_k \leftarrow U(x^0)$; $g^k \leftarrow \nabla U(x^0)$; $G_k \leftarrow G(x^0)$; $\Phi_k \leftarrow U_k$
$k = 1 : maxiter$

Calculul direcţiei de descreştere p^k
- $d \leftarrow 1$; $ip \leftarrow 1$
- cât timp $|d| > \varepsilon_p|\lambda| + 10^{-5}$şi $ip < np$
  - factorizare Cholesky $G_k + \lambda I = R^T R$
  - rezolvă sistemul $R^T R p^k = -g^k$
  - rezolvă sistemul $R^T q = -p^k$
  - $d \leftarrow (\|p^k\|/\|q\|)^2(\|p^k\|/\delta - 1)$
  - $\lambda \leftarrow \lambda + d$
  - $ip \leftarrow ip + 1$

(7.35)

$x^{k+1} \leftarrow x^k + p^k$; $U_{k+1} \leftarrow U(x^{k+1})$; $g^{k+1} \leftarrow \nabla U(x^{k+1})$; $G_{k+1} \leftarrow G(x^{k+1})$
$\Phi_{k+1} \leftarrow U_k + (p^k)^T g^{k+1} + \frac{1}{2}(p^k)^T G_{k+1}(p^k)$
$d \leftarrow U_{k+1} - U_k$
dacă $|d| < \varepsilon|U_{k+1}|$ atunci *minim calculat* **stop.**
$r_k \leftarrow d/[\Phi(x^{k+1}) - \Phi(x^k)]$
dacă $r_k > \mu$ atunci $x^k \leftarrow x^{k+1}$; $g^k \leftarrow g^{k+1}$; $G_k \leftarrow G_{k+1}$
Actualizarea razei regiunii de încredere
- dacă $r_k \leq \mu$
  atunci $\delta \leftarrow \gamma_1 \delta$
  altfel dacă $r_k > \eta$
  atunci $\delta \leftarrow \gamma_2 \delta$.

7.5 Metode cvasi-Newton

Metodele cvasi-Newton constau în aproximarea matricei G cu o matrice B, *simetrică şi pozitiv definită*. Pentru a combina proprietăţile de convergenţă globală ale metodei gradientului cu cele de convergenţă locală rapidă ale metodei Newton, ecuaţia pentru direcţia Newton p^k (7.30) se înlocuieşte cu ecuaţia

$$B_k p^k = -g^k \ , \tag{7.36}$$

iar deplasarea se ia cu ponderea α_k

$$s^k \equiv x^{k+1} - x^k = \alpha_k p^k \ , \tag{7.37}$$

α_k fiind determinat în fiecare etapă prin condiţia de minim al funcţiei de o variabilă $f(\alpha_k) \equiv U(x^k + \alpha_k p^k)$.

Pentru actualizarea matricei B_k la pasul $k+1$, adică pentru calculul matricei B_{k+1}, se dezvoltă funcţia U în serie Taylor în jurul punctului x^{k+1}, păstrându-se doar primii trei termeni

$$U(x) = U(x^{k+1}) + (x - x^{k+1})^T g^{k+1} + \tfrac{1}{2}(x - x^{k+1})^T G_{k+1}(x - x^{k+1}) \ , \quad (7.38)$$

pentru care

$$g \cong g^{k+1} + G_{k+1}(x - x^{k+1}) \cong g^{k+1} + B_{k+1}(x - x^{k+1}) \ , \quad (7.39)$$

Impunând ca B_{k+1} să aproximeze matricea hessian G_{k+1} pentru $x = x^k$, se obţine *condiţia cvasi-Newton*

$$B_{k+1}s^k = g^{k+1} - g^k \equiv y^k \ . \quad (7.40)$$

Relaţia (7.40) nu este suficientă pentru determinarea matricei B_{k+1}, astfel încât ea trebuie completată cu o condiţie suplimentară. De exemplu, putem impune ca matricea B_{k+1} să nu difere mult de B_k, adică $\|B_{k+1} - B_k\|$ să fie minim. Folosind norma Frobenius ponderată şi determinând matricea B_{k+1} ca o soluţie a problemei de minimizare

$$\min\{\|W^{-1/2}(B_{k+1} - B_k)W^{-1/2}\|_F \mid B_{k+1}s^k = y^k\} \ , \quad (7.41)$$

unde matricele B şi W sunt simetrice şi pozitiv definite, iar W satisface condiţia cvasi-Newton $Ws^k = y^k$, se obţine *formula Davidon-Fletcher-Powell* (DFP)

$$B_{k+1} = B_k + \frac{z^k(y^k)^T + y^k(z^k)^T}{(y^k)^T s^k} - \frac{(z^k)^T s^k}{[(y^k)^T s^k]^2} y^k(y^k)^T \quad (7.42)$$
$$z^k \equiv y^k - B_k s^k = y^k + \alpha_k g^k \ .$$

În mod similar, rezolvarea problemei de optimizare

$$\min\{\|W^{1/2}(B_{k+1}^{-1} - B_k^{-1})W^{1/2}\|_F \mid B_{k+1}s^k = y^k\} \ , \quad (7.43)$$

conduce la *formula Broyden-Fletcher-Goldfarb-Shanno* $(BFGS)$

$$B_{k+1} = B_k + \frac{y^k(y^k)^T}{(y^k)^T s^k} - \frac{B_k s^k (s^k)^T B_k}{(s^k)^T (B_k s^k)} \ . \quad (7.44)$$

Ambele formule (7.42) şi (7.44) se pot simplifica ţinând cont că $B_k s^k = -\alpha_k g^k$ şi au următoarea proprietate remarcabilă: dacă B_k este pozitiv definită atunci B_{k+1} este pozitiv definită.

Relaţiile (7.36) şi (7.37) se pot scrie şi sub forma

$$x^{k+1} = x^k - \alpha_k B_k^{-1} g^k \ . \quad (7.45)$$

Avantajul relaţiei (7.45) constă în eliminarea rezolvării sistemului de ecuaţii liniare. Notând $H = B^{-1}$, pentru DFP se obţine

$$H_{k+1} = H_k + \frac{s^k (s^k)^T}{(y^k)^T s^k} - \frac{H_k y^k (y^k)^T H_k}{(y^k)^T (H_k y^k)} \ , \tag{7.46}$$

iar pentru *BFGS*

$$\begin{aligned} H_{k+1} &= \left(I - \frac{s^k (y^k)^T}{(y^k)^T s^k} \right) H_k \left(I - \frac{y^k (s^k)^T}{(y^k)^T s^k} \right) + \frac{s^k (s^k)^T}{(y^k)^T s^k} = \\ &= H_k - \frac{H_k y^k (s^k)^T + s^k (y^k)^T H_k}{(y^k)^T s^k} + \frac{s^k (y^k)^T H_k y^k (s^k)^T}{[(y^k)^T s^k]^2} + \frac{s^k (s^k)^T}{(y^k)^T s^k} \ . \end{aligned} \tag{7.47}$$

Formulele pentru inversa matricei hessiene prezintă avantajul principal al numărului mai mic de operaţii, însă sunt mai puţin stabile din punct de vedere al păstrării pozitivităţii matricelor H. În plus, matricele B_k, simetrice şi pozitiv definite, admit o descompunere Choleski de forma $B_k = R_k^T R_k$. În acest caz, rezolvarea sistemului (7.36) necesită numai $O(n^2)$, operaţii. În plus, factorii descompunerii Choleski se pot actualiza direct în $O(n^2)$ operaţii.

În toate cazurile, ca matrice iniţială se poate lua matricea unitate sau orice matrice simetrică şi pozitiv definită. Se observă că formula *BFGS* pentru matricea B este mai simplă, în timp ce pentru matricea H este preferabilă formula *DFP*. La aplicarea formulelor de mai sus, se evită înmulţirea matricelor, utilizându-se numai produse de vectori sau de vectori şi matrice. Convergenţa acestor metode este globală şi superliniară.

Există pericolul ca, din cauza acumulării erorilor de rotunjire, matricele B sau H să devină singulare sau să-şi piardă pozitivitatea. O modalitate de corecţie constă în reiniţializarea matricelor B sau H cu matricea unitate după n iteraţii.

Convergenţa metodelor cvasi-Newton depinde esenţial de exactitatea cu care se determină minimul unidirecţional la fiecare etapă. Metoda *BFGS* se dovedeşte mai puţin sensibilă din acest punct de vedere, fiind mai stabilă decât metoda *DFP*. Aplicate unor funcţii pătratice ($U = x^T Ax$, A matrice constantă), metodele cvasi-Newton converg în maximum n iteraţii (n fiind numărul de variabile independente) când matricea B va conţine elementele matricei hessiene.

7.6 Metode de gradient conjugat

Metodele de gradient conjugat nu se deosebesc esenţial de metodele cvasi-Newton din punct de vedere al scopului, şi anume obţinerea minimului unei forme pătratice în n iteraţii. Ambele clase de metode necesită calculul derivatelor parţiale de ordinul întâi şi au aceeaşi convergenţă superliniară. Deosebirea esenţială constă în faptul că metodele de gradient conjugat nu necesită memorarea unei matrice.

Fie gradientul formei pătratice Φ definită de (7.13), evaluat în punctul x^{k+1}

$$\nabla\Phi(x^{k+1}) \equiv g^{k+1} = g^k + G(x^{k+1} - x^k) = g^k + \alpha_k G p^k \ , \tag{7.48}$$

unde am consderat $x^{k+1} = x^k + \alpha_k p^k$, iar direcţiile de descreştere construite după regula

$$p^{k+1} = -g^{k+1} + \beta_k p^k \ . \tag{7.49}$$

Avem de determinat doi parametrii: α_k şi β_k. Pentru a păstra avantajul câştigat la iteraţiile anterioare (vezi 7.2), vom impune ca direcţiile p^{k+1} şi p^k să fie *G-conjugate*, adică $(p^{k+1})^T G p^k = 0$. Înmulţind la dreapta transpusa relaţiei (7.49) cu Gp^k, se obţine

$$\beta_k = (g^{k+1})^T G p^k / (p^k)^T G p^k \ . \tag{7.50}$$

Înmulţind la dreapta aceeaşi relaţie cu Gp^{k+1}, se obţine egalitatea

$$(p^{k+1})^T G p^{k+1} = -(g^{k+1})^T G p^{k+1} \ , \tag{7.51}$$

Punând condiţia ca vectorii g^k şi g^{k+1} să fie perpendiculari ($(g^{k+1})^T g^k = 0$), folosind relaţia (7.51) şi scoţând din (7.48) $Gp^k = (g^{k+1} - g^k)/\alpha_k$, din relaţia (7.50) se obţine

$$\beta_k = -(g^{k+1})^T G p^k / (g^k)^T G p^k = (g^{k+1})^T g^{k+1} / (g^k)^T g^k \ . \tag{7.52}$$

Pentru determinarea lui α_k vom folosi din nou condiţia ca vectorii g^k şi g^{k+1} să fie perpendiculari. Înmulţind relaţia (7.48) cu $(g^{k+1})^T$, respectiv cu $(g^k)^T$, se obţin relaţiile

$$\alpha_k = -(g^k)^T g^k / (g^k)^T G p^k = (g^{k+1})^T g^{k+1} / (g^{k+1})^T G p^k \ , \tag{7.53}$$

relaţii care nu sunt utile din punct de vedere practic deoarece nu avem la dispoziţie hessiana G. Vom arăta însă că ponderea α_k din relaţia (7.48) este chiar valoarea care rezultă din problema de optimizare $\alpha_k = \min\limits_{\alpha \in \mathbf{R}} \{U(x^k + \alpha p^k)\}$. Pentru aceasta trebuie să arătăm că vectorii g^{k+1} şi p^k sunt perpendiculari, adică $(p^k)^T g^{k+1} = 0$. Înmulţind la stânga relaţia (7.48) cu $(p^k)^T$ şi ţinând cont de relaţile (7.49) şi (7.51), se obţine succesiv

$$\begin{aligned} (p^k)^T g^{k+1} &= (p^k)^T g^k + \alpha_k (p^k)^T G p^k \\ &= (-g^k + \beta_{k-1} p^{k-1})^T g^k - \alpha_k (g^k)^T G p^k \\ &= -(g^k)^T g^k + \beta_{k-1} (p^{k-1})^T g^k - (g^k)^T g^{k+1} + (g^k)^T g^k \\ &= \beta_{k-1} (p^{k-1})^T g^k \ . \end{aligned} \tag{7.54}$$

Dacă la pasul anterior căutarea unidirecţională s-a făcut exact, adică ponderea α_{k-1} a fost determinată astfel încât $(p^{k-1})^T g^k = 0$, atunci avem $(p^k)^T g^{k+1} = 0$.

Am obţinut astfel setul de direcţii *G-conjugate* p^k construite cu relaţia (7.49), cu valoarea β_k dată de relaţia (7.52). Pentru construirea setului de direcţii p^k

nu este nevoie de matricea hessian G, dar este necesar ca ponderile α_k să fie calculate exact.

Deşi teoretic am găsit o singură cale de obţinere a unui set de direcţii G-*conjugate*, în practică se întâlnesc mai multe variante, în funcţie de modul cum se calculează parametrul β :

Metoda Fletcher-Reeves $\beta_k = (g^{k+1})^T g^{k+1}/(g^k)^T g^k$;
Metoda Polak-Ribière $\beta_k = (g^{k+1})^T y^k/(g^k)^T g^k$, $y^k = g^{k+1} - g^k$;
Metoda Hestens-Stiefel $\beta_k = (g^{k+1})^T y^k/(y^k)^T p^k$.

Pentru o formă pătratică relaţiile de mai sus sunt echivalente. Astfel, numărătorul relaţiei *Polak-Ribière* se mai scrie $(g^{k+1})^T(g^{k+1} - g^k) = (g^{k+1})^T g^{k+1}$, deoarece $(g^{k+1})^T g^k = 0$, iar numitorul relaţiei *Hestens-Stiefel* se scrie $(y^k)^T p^k = (g^{k+1})^T p^k - (g^k)^T p^k = 0 - (g^k)^T g^k = -(g^k)^T g^k$.

Se observă tendinţa metodei Polak-Ribière de reiniţializare a direcţiei de căutare după direcţia gradientului simplu în situaţiile dificile, când $x^{k+1} - x^k$ este foarte mic şi $g^{k+1} \approx g^k$. Din această cauză, pentru funcţii obiectiv complicate metoda Polak-Ribière se dovedeşte a fi mai robustă. Metoda Hestens-Stiefel are şi la numitor termenul $y^k = g^{k+1} - g^k$, fapt care provoacă dificultăţi în situaţia mai sus amintită.

Avansând la fiecare iteraţie cu pasul optim $\alpha_k = \alpha_k^*$, convergenţa metodelor de gradient conjugat este superliniară. Pentru a evita efectele nedorite datorate acumulării erorilor de rotunjire, se recomandă reiniţializarea algoritmului după $n + 1$ paşi, punându-se $\beta_{k+1} = 0$. Ea este sugerată şi de faptul că, pentru o funcţie pătratică, metodele de gradient conjugat converg în cel mult n iteraţii.

Metodele de gradient conjugat necesită $O(n)$ operaţii pe iteraţie; de asemenea, au proprietăţi de convergenţă globală. Ca urmare, la rezolvarea problemelor cu un număr mare de variabile se recomandă folosirea metodelor de gradient conjugat în locul metodelor cvasi-Newton.

7.6.1 Rezolvarea sistemelor de ecuaţii liniare folosind metode de optimizare

Metodele de optimizare pot fi aplicate cu succes pentru rezolvarea sistemelor de ecuaţii liniare (capitolul 5), îndeosebi pentru sisteme mari, atunci când metodele directe nu mai sunt eficiente. Deoarece numărul de variabile este mare, în astfel de situaţii sunt preferate metodele de gradient conjugat.

Pentru a folosi o metodă de optimizare, trebuie definită funcţia obiectiv U. Fie sistemul de ecuaţii liniare

$$Ax = b \ , \tag{7.55}$$

unde A este o matrice simetrică pozitiv definită, adică avem $A^T = A$ şi $x^T Ax > 0$ pentru orice $x \neq 0$. În aceste condiţii, problema rezolvării sistemului (7.55) este echivalentă cu problema minimizării formei pătratice

$$U(x) = \langle x, Ax \rangle - 2 \langle x, b \rangle \ , \tag{7.56}$$

unde pentru produsul scalar s-a utilizat notaţia

$$\langle x, b\rangle \equiv x^T b = \sum_{i=1}^{n} x_i b_i \ . \tag{7.57}$$

În cazul în care matricea A este simetrică, gradientul funcţiei U este dat de relaţia

$$g(x) \equiv \nabla U(x) = -2(b - Ax) \ . \tag{7.58}$$

Se observă că funcţia U are un extrem pentru soluţia x a sistemului (7.55). În lungul unei direcţii de descreştere p avem relaţiile

$$\begin{aligned} U(x+\alpha p) &= \langle x+\alpha p, A(x+\alpha p)\rangle - 2\langle x+\alpha p, b\rangle \\ &= \langle x, Ax\rangle + \alpha\langle x, Ap\rangle + \alpha\langle p, Ax\rangle + \alpha^2\langle p, Ap\rangle - 2\langle x, b\rangle - 2\alpha\langle p, b\rangle \\ &= U(x) + 2\alpha\langle p, Ax\rangle - 2\alpha\langle p, b\rangle + \alpha^2\langle p, Ap\rangle \\ &= U(x) + 2\alpha\langle p, Ax - b\rangle + \alpha^2\langle p, Ap\rangle \ . \end{aligned} \tag{7.59}$$

Am folosit egalitatea $\langle x, Ap\rangle = \langle A^T x, p\rangle = \langle p, A^T x\rangle$, şi cum $A^T = A$, rezultă $\langle x, Ap\rangle = \langle p, Ax\rangle$. Observăm cum coeficientul lui α^2 este pozitiv, $\langle p, Ap\rangle > 0$ deoarece matricea A este pozitiv definită, deci funcţia U are un minim pentru $\alpha = \alpha^*$ care poate fi determinat din condiţia

$$\frac{d}{d\alpha} U(x+\alpha p) = 2\langle p, Ax - b\rangle + 2\alpha\langle p, Ap\rangle \ , \tag{7.60}$$

rezultând

$$\alpha^* = \langle p, b - Ax\rangle \,/\, \langle p, Ap\rangle \ . \tag{7.61}$$

Cu valoarea α^* dată de (7.61), obţinem minimul funcţiei U după direcţia p

$$\begin{aligned} U(x+\alpha^* p) &= U(x) + \alpha^*\left[2\langle p, Ax - b\rangle + \alpha^*\langle p, Ap\rangle\right] \\ &= U(x) + \alpha^*\left[2\langle p, Ax - b\rangle + \langle p, b - Ax\rangle\right] \\ &= U(x) - \alpha^*\langle p, b - Ax\rangle \\ &= U(x) - \langle p, b - Ax\rangle^2 \,/\, \langle p, Ap\rangle \ . \end{aligned} \tag{7.62}$$

Dăm în continuare algoritmul de determinare a minimului funcţiei U dată de (7.56) folosind metoda gradientului pentru care direcţia de descreştere este $p = -g$ dată de (7.58). Condiţia de oprire folosită este $\|b - Ax\|_2^2 < \varepsilon$.

Date: $x^0, A, b, maxiter, \varepsilon$
$iter \leftarrow 1$; $norm \leftarrow 1$; $x \leftarrow x^0$
cât timp $norm > \varepsilon$ şi $iter \leq maxiter$
$\quad p \leftarrow b - Ax$; $norm \leftarrow \langle p, p\rangle$
$\quad \alpha \leftarrow norm / \langle p, Ap\rangle$
$\quad x \leftarrow x + \alpha p$.

Determinarea minimului funcţiei U dată de (7.56) se poate face folosind orice altă metodă de optimizare. Spre exemplu, aplicarea metodei *Fletcher-Reeves* duce la algoritmul (7.65). În partea stângă am prezentat o variantă mai uşor de urmărit în timp ce în partea dreaptă este o variantă optimizată pentru codificarea într-un limbaj de programare. Testul de oprire este $\left\langle r^{k+1}, r^{k+1}\right\rangle < \varepsilon$, unde r^{k+1} este reziduul la pasul $k+1$ care coincide cu gradientul funcţiei scop definită de (7.56). Iteraţiile se opresc şi în cazul în care înaintarea spre minim devine lentă, adică $\left\langle p^k, p^k\right\rangle < \delta$.

Numărul de iteraţii necesare obţinerii soluţiei sistemului (7.56) depinde de numărul de condiţionare al matricei A (vezi capitolul 5). Pentru a mări viteza de convergenţă, se poate îmbunătăţi condiţionarea sistemului (7.56) prin transformarea sa într-un sistem echivalent

$$\hat{A}\hat{x} = \hat{b}\ , \tag{7.63}$$

unde

$$\hat{A} = S^T A S,\ \hat{x} = S^{-1}x,\ \hat{b} = S^T b\ . \tag{7.64}$$

Spunem că am efectuat o *precondiţionare* a sistemului.

Algoritmul Fletcher-Reeves pentru sisteme liniare.

Date: $x^0, A, b, maxiter, \varepsilon, \delta$
$r^0 \leftarrow b - Ax^0$
$p^0 \leftarrow r^0$
$k = 0 : (maxiter - 1)$
$\left[\begin{array}{l} \text{dacă } \left\langle p^k, p^k\right\rangle^{1/2} < \delta \textbf{stop} \\ \alpha_k \leftarrow \left\langle r^k, r^k\right\rangle / \left\langle p^k, Ap^k\right\rangle \\ x^{k+1} \leftarrow x^k + \alpha_k p^k \\ r^{k+1} \leftarrow r^k - \alpha_k A p^k \\ \text{dacă } \left\langle r^{k+1}, r^{k+1}\right\rangle < \varepsilon \textbf{stop} \\ \beta_k \leftarrow \left\langle r^{k+1}, r^{k+1}\right\rangle / \left\langle r^k, Ar^k\right\rangle \\ p^{k+1} \leftarrow r^{k+1} + \beta_k p^k\ . \end{array}\right.$

Date: $x^0, A, b, maxiter, \varepsilon, \delta$
$x \leftarrow x^0;\ r \leftarrow b - Ax;\ p \leftarrow r$
$c \leftarrow \langle r, r\rangle\ ;\ d \leftarrow c$
$k = 1 : maxiter$
$\left[\begin{array}{l} \text{dacă } \langle p, p\rangle^{1/2} < \delta \textbf{stop} \\ z \leftarrow Ap;\ \alpha \leftarrow c / \langle p, z\rangle \\ x \leftarrow x + \alpha p \\ r \leftarrow r - \alpha z \\ d \leftarrow \langle r, r\rangle \\ \text{dacă } d < \varepsilon \textbf{stop} \\ p \leftarrow r + (d/c)p \\ c \leftarrow d\ . \end{array}\right.$ (7.65)

Dacă matricea sistemului original are multe elemente nule (matrice rară), operaţia de precondiţionare poate duce la pierderea acestui avantaj. Din acest punct de vedere, în loc de a efectua o precondiţionare explicită a sistemului original urmată apoi de rezolvarea numerică a sistemului obţinut, este mai avantajoasă din punct de vedere numeric folosirea unei tehnici de precondiţionare implicită folosind relaţiile

$$\begin{array}{c} \hat{x}^k = S^{-1}x^k;\ \hat{p}^k = S^{-1}p^k \\ \hat{r}^k = \hat{b} - \hat{A}\hat{x}^k = S^T b - (S^T A S)(S^{-1}x^k) = S^T r^k \\ \tilde{r} = S\hat{r}^k = Q^{-1}r^k, \text{ unde } Q^{-1} = S\,S^T\ . \end{array} \tag{7.66}$$

În aceste condiţii obţinem

$$\begin{aligned}\hat{\alpha}_k &= \langle \hat{r}^k, \hat{r}^k \rangle / \left\langle \hat{p}^k, \hat{A}\hat{p}^k \right\rangle \\ &= \langle S^T r^k, S^T r^k \rangle / \langle S^{-1}p^k, (S^T AS)(S^{-1}p^k) \rangle \\ &= \langle SS^T r^k, r^k \rangle / \langle SS^{-1}p^k, Ap^k \rangle = \langle Q^{-1}r^k, r^k \rangle / \langle p^k, Ap^k \rangle \\ &= \langle \tilde{r}^k, r^k \rangle / \langle p^k, Ap^k \rangle \ . \end{aligned} \tag{7.67}$$

Relaţia de deplasare către minimul ce reprezintă soluţia sistemului (7.63) este

$$\hat{x}^{k+1} = \hat{x}^k + \hat{\alpha}_k \hat{p}^k \ , \tag{7.68}$$

sau ţinând cont de (7.66), se obţine

$$S^{-1}x^{k+1} = S^{-1}x^k + \hat{\alpha}_k S^{-1}p^k \ . \tag{7.69}$$

Înmulţind la stânga cu S relaţia (7.69), obţinem

$$x^{k+1} = x^k + \hat{\alpha}_k p^k \ . \tag{7.70}$$

Analog se obţin relaţiile

$$r^{k+1} = r^k - \hat{\alpha}_k A p^k, \ \hat{\alpha}_k = \langle \tilde{r}^{k+1}, r^{k+1} \rangle / \langle \tilde{r}^k, r^k \rangle, \ p^{k+1} = \tilde{r}^{k+1} + \hat{\alpha}_k p^k \ . \tag{7.71}$$

Dăm în continuare algoritmul pentru rezolvarea sistemului de ecuaţii liniare folosind metoda de gradient conjugat *Fletcher-Reeves* cu precondiţionare.

Date: $x^0, A, b, maxiter, \varepsilon, \delta$
$r^0 \leftarrow b - Ax^0$
rezolvă sistemul $Q\tilde{r}^0 = r^0$
$p^0 \leftarrow r^0$
$k = 0 : (maxiter - 1)$

- dacă $\langle p^k, p^k \rangle^{1/2} < \delta$ **stop**
- $\hat{\alpha}_k \leftarrow \langle \tilde{r}^k, r^k \rangle / \langle p^k, Ap^k \rangle$
- $x^{k+1} \leftarrow x^k + \hat{\alpha}_k p^k$
- $r^{k+1} \leftarrow r^k - \hat{\alpha}_k A p^k$
- rezolvă sistemul $Q\tilde{r}^{k+1} = r^{k+1}$
- dacă $\langle \tilde{r}^{k+1}, r^{k+1} \rangle < \varepsilon$ atunci
  - dacă $\langle r^{k+1}, r^{k+1} \rangle^2 < \varepsilon$ **stop**
- $\hat{\beta}_k \leftarrow \langle \tilde{r}^{k+1}, r^{k+1} \rangle / \langle \tilde{r}^k, r^k \rangle$
- $p^{k+1} \leftarrow \tilde{r}^{k+1} + \hat{\beta}_k p^k$.

Date: $x^0, A, b, maxiter, \varepsilon, \delta$
$x \leftarrow x^0$; $r \leftarrow b - Ax$
rezolvă sistemul $Qz = r$
$p \leftarrow r$; $c \leftarrow \langle z, r \rangle$
$k = 1 : maxiter$

- dacă $\langle p, p \rangle^{1/2} < \delta$ **stop**
- $z \leftarrow Ap$; $\alpha \leftarrow c / \langle p, z \rangle$
- $x \leftarrow x + \alpha p$
- $r \leftarrow r - \alpha z$
- rezolvă sistemul $Qz = r$
- $d \leftarrow \langle z, r \rangle$
- dacă $d < \varepsilon$
  - atunci
    - $e \leftarrow \langle r, r \rangle$
    - dacă $e < \varepsilon$ **stop**
- $p \leftarrow z + (d/c)p$
- $c \leftarrow d$.

Din nou, în partea stângă prezentăm o variantă mai uşor de urmărit în timp ce în partea dreaptă optimizată pentru codificarea într-un limbaj de programare.

Deoarece la fiecare pas trebuie rezolvat un sistem de forma $Qx = y$, matricea Q trebuie aleasă astfel încât sistemul să fie uşor de rezolvat. Dacă spre exemplu matricea Q este diagonală, atunci efortul de calcul pentru rezolvarea sistemului este minim la fiecare pas, în timp ce forme mai complicate ale matricei Q permit obţinerea unui sistem mai bine condiţionat şi deci un număr mai mic de iteraţii necesare obţinerii soluţiei. Ne aflăm în faţa unui exemplu tipic de compromis între un efort minim pe iteraţie dar număr mare de iteraţii şi un efort mai mare pe iteraţie care duce la micşorarea numărului de iteraţii.

7.7 Metode specifice de optimizare

Din această categorie fac parte metodele care sunt special construite pentru funcţii scop ce au forme particulare. Fie funcţia scop de forma

$$U(x) = \frac{1}{2}\sum_{j=1}^{n} F_j^2(x) = \frac{1}{2}\|F\|_2^2 \ , \tag{7.72}$$

unde $F : \mathbf{R}^n \to \mathbf{R}^n$. O problemă de optimizare cu funcţia scop de forma (7.72) se mai întâlneşte sub denumirea de *problema celor mai mici pătrate neliniare*. Observăm că

$$g_i = \partial U/\partial x_i = \sum_{j=1}^{n} (\partial F_j/\partial x_i) F_j, \ i \in \overline{1,n} \ , \tag{7.73}$$

sau vectorial

$$\nabla U(x) \equiv g(x) = J^T(x)F(x) \ , \tag{7.74}$$

unde J este matricea jacobiană a funcţiei F. Notând cu $[G_i]_{j,p} = (\partial^2 F_i)/(\partial x_j \partial x_p)$, se obţine

$$\nabla^2 U(x) = J^T(x)J(x) + Q(x) \ , \tag{7.75}$$

unde cu Q s-a notat matricea

$$Q(x) = \sum_{i=1}^{n} F_i(x)G_i(x) \ . \tag{7.76}$$

Dacă minimul funcţiei U este zero sau foarte mic, atunci în vecinătatea acestuia avem valori mici pentru funcţiile F_i astfel încât se poate neglija termenul $Q(x)$ din relaţia (7.75), aproximând[2] hessiana cu relaţia

$$G(x) = \nabla^2 U(x) \cong J^T(x)J(x) \ , \tag{7.77}$$

obţinând deci o metodă cu eficienţă comparabilă metodelor de tip Newton. Dintre metodele care rezolvă problema celor mai mici pătrate neliniare, metoda *Levenberg-Marquardt* face parte din categoria metodelor regiunii de încredere în model prezentate la 33. Algoritmul prezentat pentru metoda regiunii de încredere poate fi aplicat direct pentru metoda Levenberg-Marquardt, dar ţinând cont de aproximarea hessianei cu (7.77) pot fi aduse îmbunătăţiri modului de calcul al direcţiei de descreştere p.

[2] Această ipoteză nu mai este justificată dacă $\sqrt{U(x^*)}$ este de acelaşi ordin de mărime cu valoarea proprie maximă a matricei $J^T J$, unde cu x^* s-a notat punctul în care U ia valoarea minimă.

Figura 7.5: Exemplu de restricţie liniară.

7.8 Probleme de optimizare cu restricţii

Majoritatea problemelor practice se formulează ca probleme de optimizare la care variabilele funcţiei obiectiv sunt supuse unor restricţii. Este bine de reţinut faptul că deseori metodele de optimizare pentru probleme cu restricţii au la bază metodele de optimizare fără restricţii, cu mici modificări. Spaţiul nu ne permite prezentarea în detaliu a metodelor de optimizare cu restricţii, dar câteva noţiuni de bază sunt necesare.

Restricţiile pot fi *liniare* sau *neliniare*. Restricţiile liniare pot fi scrise în forma generală

$$Ax = b \ , \tag{7.78}$$

când avem de a face cu restricţii egalităţi liniare, sau

$$Ax \geq b \ , \tag{7.79}$$

când avem de a face cu restricţii inegalităţi liniare. Matricea A are m linii, unde m este numărul de restricţii impuse celor n variabile ale funcţiei scop. Un punct $x \in \mathbf{R}^n$ este *admisibil* dacă îndeplineşte condiţiile (7.78) sau (7.79) după cum este cazul. Spre exemplu, în cazul $n = 2$, o restricţie egalitate de forma $x_1 - x_2 = 1$ impune ca minimul să fie căutat pe o dreapta (fig. 7.5), iar o restricţie inegalitate de forma $x_1 - x_2 \leq 1$ face ca punctele admisibile să fie plasate în semiplanul de deasupra dreptei $x_2 = 1 - x_1$. Se observă că restricţiile egalitate reduc dimensiunea problemei: minimizarea unei funcţii scop de n variabile supuse la m restricţii *independente*, se poate obţine aplicând o metodă de optimizare fără restricţii pentru o funcţie scop de $n - m$ variabile.

O primă posibilitate de rezolvare a problemelor cu restricţii liniare este de a folosi o metodă de optimizare fără restricţii împreună cu o *strategie de penalizare* a funcţiei scop dacă algoritmul încearcă să iasă din domeniul D al punctelor

admisibile. Practic se redefineşte funcţia scop

$$\hat{U}(x) = \begin{cases} U(x), & \text{pentru } x \in D \\ \mu, & \text{pentru } x \notin D \end{cases} \tag{7.80}$$

unde $\mu \geq U(x^0)$. Această strategie are dezavantajul micşorării vitezei de convergenţă, în special dacă minimul este în apropierea frontierei lui D.

O altă posibilitate o constituie *tehnicile de transformare*. Spre exemplu, în cazul particular al unor restricţii tip *margini simple* de forma $\alpha_i \leq x_i \leq \beta_i$, $i \in \overline{1, m'}$, $x_i \leq \gamma_i$, $i \in \overline{m'+1, m}$ se poate utiliza următoarea transformare de tip *Valentine*:

$$\begin{array}{c} x_i = \frac{1}{2}[(\beta_i - \alpha_i)\cos z_i + (\beta_i + \alpha_i)],\ i \in \overline{1, m'} \\ x_i = \gamma_i - z_i^2,\ i \in \overline{m'+1, m}\ . \end{array} \tag{7.81}$$

Tehnicile bazate pe transformări prezintă dezavantajul creşterii neliniarităţii funcţiei scop transformate $\hat{U}$. În plus, $\hat{U}$ poate deveni nemărginită sau poate avea derivate discontinue ori hessiana nedefinită sau singulară. Metode mai performante se bazează pe combinaţii între tehnicile algebrei liniare şi cele ale optimizării fără restricţii.

Restricţiile neliniare pot fi la rândul lor de tip egalitate sau inegalitate

$$f_i(x) = 0, \text{ sau } f_i(x) \geq 0,\ i \in \overline{1, n}\ , \tag{7.82}$$

unde $f_i : \mathbf{R}^n \to \mathbf{R}$ iar m este numărul de restricţii. Pentru rezolvarea problemelor de optimizare cu restricţii neliniare se pot folosi tehnici de penalizare, extensii ale metodelor de rezolvare a problemelor cu restricţii liniare sau metode specifice restricţiilor neliniare cum ar fi metoda de lagrangean proiectat.

Algoritmul Simplex.

Numeroase probleme din practică conduc la necesitatea optimizării unei funcţii scop liniare[3] de forma

$$U(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{n} \alpha_j x_j\ , \tag{7.83}$$

unde cel puţin un coeficient α_j este nenul, cu restricţii de forma

$$\begin{array}{rcl} x_j & \geq & 0,\ j \in \overline{1, n}\ , \\ \sum_{j=1}^{n} a_{ij} x_j & \geq & b_i,\ i \in \overline{1, m_1},\ b_i \geq 0\ , \\ \sum_{k=1}^{n} a_{kj} x_j & \leq & b_k,\ k \in \overline{m_1+1, m_2},\ b_k \geq 0\ , \\ \sum_{l=1}^{n} a_{lj} x_j & = & b_l,\ l \in \overline{m_2+1, m},\ b_l \geq 0\ . \end{array} \tag{7.84}$$

[3] O funcţie $U : \mathbf{R}^n \to \mathbf{R}$ este liniară dacă îndeplineşte următoarele două condiţii: $U(\lambda x) = \lambda U(x)$ şi $U(\alpha x + \beta y) = \alpha U(x) + \beta U(y)$, oricare ar fi vectorii $x, y \in \mathbf{R}^n$ şi scalarii $\alpha, \beta, \lambda \in \mathbf{R}$.

Figura 7.6: Un exemplu simplu de programare liniară.

Vom urmări în continuare obţinerea unei *valori maxime* (pe care o vom numi în continuare *valoare optimă*) a funcţiei U. Problema determinării unei valori minime pentru U se reduce la determinarea unei valori maxime pentru $-U$. Restricţiile de forma $x_j \geq 0$ sunt impuse de semnificaţia fizică a variabilelor. Frontiera *domeniului admisibil*, care conţine toţi vectorii $x \in \mathbf{R}^n$ care îndeplinesc simultan condiţiile (7.84) şi care se numesc *vectorii admisibili*, este definită de ecuaţiile liniare obţinute prin transformarea inegalităţilor în egalităţi. Vectorul admisibil care realizează valoarea maximă a funcţiei scop U se numeşte *vector optimal*. Condiţiile $b_i \geq 0$ pot fi întotdeauna îndeplinite (în cazul în care $b_p < 0$, se înmulţeşte cu -1 inegalitatea respectivă). Deoarece gradientul unei funcţii liniare de forma (7.83) este nenul în orice punct, rezultă că valoarea optimă se află pe frontiera domeniului admisibil. Această valoare maximă este finită numai dacă restricţiile (7.84) definesc un poliedru convex. Mai mult, dacă valoarea optimă este unică, atunci ea se va găsi într-unul din vârfurile poliedrului.

Problema determinării valorii optime a funcţiei U se reduce principial la căutarea acesteia pe frontiera domeniului valorilor admisibile. Vectorul optimal trebuie să satisfacă sistemul de m ecuaţii liniare ce definesc frontiera şi care conţin n necunoscute, x_j, $j \in \overline{1,n}$, n **7** m.

EXEMPLU. În figura (7.6) este exemplificat cazul

$$U(x_1, x_2) = x_1 + x_2, \tag{7.85}$$

cu restricţiile

$$x_1 + x_2 \leq 10,\ -2x_1 + x_2 \leq 3,\ -x_1 + x_2 \geq -3,\ x_1 \geq 0,\ x_2 \geq 0\ . \tag{7.86}$$

Pe figură se observă domeniul admisibil, precum şi vectorul optimal unic $x = (1.75, 6.50)$ aflat pe frontieră.

În cazul în care n este mare, parcurgerea la întâmplare a frontierei poate fi total ineficientă. Algorimul *simplex* oferă o modalitate de căutare a valorii

optime asigurând în acelaşi timp două lucruri: (a) valoarea funcţiei scop U creşte de la un pas la altul; (b) valoarea optimă este obţinută în maximum M paşi, unde $M = \max(n, m)$. Este interesant că deşi algoritmul simplex a fost elaborat încă din 1948 de G.B. Dantzig iar proprietatea (b) de mai sus era confirmată practic, aceasta a fost demostrată riguros abia în 1982 de S. Smale. Pentru prezentarea algoritmului simplex, vom apela la două exemple simple.

EXEMPLUL I. Se cere să se găsească valoarea maximă a funcţiei

$$U = x_2 - 3x_3 \ , \tag{7.87}$$

cu restricţiile

$$x_1 + 2x_2 - 3x_3 = 3, \ x_2 - x_3 - x_4 = -1, \ x_1, x_2, x_3, x_4 \geq 0 \ , \tag{7.88}$$

şi bineînţeles $x_i \geq 0$. Se observă valoarea funcţiei U este cu atât mai mare cu cât x_2 este mai mare şi x_3 mai mic. Rescriem prima relaţie (7.88) în forma $x_1 = 3 - 2x_2 + 3x_3 \geq 0$. Ultima inegalitate este impusă de condiţia $x_1 \geq 0$; rezultă $x_2 \leq \frac{3}{2}(1 + x_3)$. Pentru

Tabelul 7.1: Exemplu de calcul pentru algoritmul simplex

		x_2	x_3				x_1	x_3				x_1	x_4
U	0	1	-1		U	3/2	$-1/2$	1/2		U	4	-1	-1
x_1	3	-2	3	$\rightarrow$	x_2	3/2	$-1/2$	3/2	$\rightarrow$	x_2	9	-2	-3
x_4	1	1	-2		x_4	5/2	$-1/2$	$-1/2$		x_3	5	-1	-2

valoarea $x_3 = 0$ care convine din punctul de vedere al unei valori cât mai mari pentru U, se obţine $x_2 \leq 3$. Luând $x_3 = 0$, $x_2 = 3/2$, rezultă $x_1 = 0$, $x_4 = 5/2$ şi $U = 3/2$.

EXEMPLUL II. Considerăm funcţia scop

$$U = x_2 - x_3 \ , \tag{7.89}$$

cu restricţiile

$$x_1 + 2x_2 - 3x_3 = 3, \ -x_2 + 2x_3 + x_4 = 1, \ x_1, x_2, x_3, x_4 \geq 0 \ . \tag{7.90}$$

Procedând la fel ca în exemplul precedent, obţinem $x_1 = 3 - 2x_2 + 3x_3 \geq 0$ rezultând $x_2 \leq \frac{3}{2}(1 + x_3)$ şi $x_4 = 1 + x_2 - 2x_3 \geq 0$, rezultând $x_3 \leq \frac{1}{2}(1 + x_2)$. Suntem tentaţi de valorile $x_3 = 0$, $x_2 = 3$ cu care am obţinut cu atâta uşurinţă $U = 3$ în exemplul precedent. Dacă introducem însă inegalitatea pentru x_2 obţinută mai sus în expresia funcţiei scop, obţinem $U \leq \frac{3}{2} + \frac{1}{2}x_3$, rezultat care sugerează că valoarea minimă nu se obţine pentru $x_3 = 0$. Într-adevăr, putem scrie $x_2 \leq \frac{3}{2}(1 + x_3) \leq \frac{3}{2}[1 + \frac{1}{2}(1 + x_2)]$, rezultând $x_2 = 9$, $x_3 = 5$, $x_1 = 0$, $x_4 = 0$, $U = 4$. Începem să simţim nevoia unei metodologii mai riguroase, fără de care este clar că nu vom putea rezolva problemele cu mai multe variabile.

Prezentăm algoritmul simplex folosindu-ne de exemplul 2 şi cu ajutorul tabelului 7.1. Observăm în prima parte a tabelului (primele 4 coloane) că variabilele x_1 şi x_4 apar distinct de variabilele x_2 şi x_3, de care depinde în mod explicit funcţia U. Variabilele x_1 şi x_4 care apar pe prima coloană se numesc *variabile de bază*. Variabilele de bază sunt acele variabile care apar într-o singură relaţie-restricţie (şi *numai într-una*), având coeficientul pozitiv şi care se pot exprima *numai* în funcţie de celelalte variabile. În consecinţă, funcţia scop se poate exprima în aşa fel încât să nu conţină nici o variabilă de bază. În exemplul nostru avem variabile de bază x_1 şi x_4. Din tabel citim de pe prima linie $U = 0 + 1 \cdot x_2 - 1 \cdot x_3$, de pe linia a doua $x_1 = 3 - 2x_2 + 3x_3$ s.a.m.d. De pe linia funcţiei scop căutăm un coeficient pozitiv şi găsim 1 pe coloana variabilei x_2, adică la creşterea valorii x_2 creşte şi valoarea funcţiei scop U. Pe coloana lui x_2 căutăm un coeficient negativ, şi găsim valoarea -2 pe linia lui x_1. Înseamnă că nu putem creşte oricât de mult valoarea x_2, deoarece acest lucru poate duce la obţinerea unei valori negative pentru variabila de bază pentru care am găsit coeficientul negativ, în cazul nostru x_1. De fapt, rata creşterii funcţiei U este dată de multiplicatorul $p_{ij} = \frac{\alpha_j}{|a_{ij}|} \cdot b_i$, unde $\alpha_j = 1$ este coeficientul lui x_2 din expresia funcţiei U, $a_{ij} = -2$ este coeficientul negativ de pe coloana lui x_2, iar $b_i = 3$ este termenul liber din expresia variabilei de bază x_1, $(i = 1, j = 2)$. Dacă pe coloana lui x_2 toţi coeficienţii ar fi fost pozitivi, ar fi însemnat că puteam creşte oricât de mult pe x_2, iar funcţia scop ar fi crescut de

asemenea oricât de mult, caz în care problema este încheiată. Pasul următor constă în interschimbarea variabilelor x_2 şi x_1, adică din relaţia de definiţie a variabilei de bază x_1 se explicitează x_2, iar expresia obţinută este folosită pentru a obţine U şi x_4 numai în funcţie de x_1 şi x_3. Se obţin relaţiile $x_2 = \frac{1}{2}(3 - x_1 + 3x_3)$, $U = \frac{1}{2}(3 - x_1 + x_3)$ şi $x_4 = \frac{1}{2}(5 - x_1 - x_3)$, relaţii prezente prin coeficienţi în partea a doua a tabelului 7.1. În general, se procedează la interschimbarea variabilelor x_i şi x_j corespunzătoare unui multiplicator $p_{ij} = \frac{\alpha_j}{|a_{ij}|} \cdot b_i$ maxim. În continuare, căutăm un coeficient pozitiv pe linia funcţiei U, şi găsim valoarea $\frac{1}{2}$ pe coloana x_3, adică U încă mai creşte cu x_3. Repetăm procedeul, căutăm un coeficient negativ pe coloana x_3, găsim valoarea $-\frac{1}{2}$ pe linia x_4 şi interschimbăm variabilele x_3 şi x_4 ca la pasul anterior. Obţinem ultimele patru coloane din tabelul 7.1. Pe linia funcţiei U citim $U = 4 - x_1 - x_4$. Calculul s-a încheiat, deoarece coeficienţii variabilelor din expresia funcţiei U sunt toţi negativi. Soluţia o citim pe coloana funcţiei scop, adică $U = 4$, $x_2 = 9$, $x_3 = 5$, iar pentru celelalte variabile obţinem $x_1 = x_4 = 0$.

Ştim în acest moment să rezolvăm o problemă de programare liniară în *forma standard restrânsă*. Forma standard se caracterizează prin faptul că cele m restricţii ($m < n$) sunt egalităţi. În plus, matricea A ce se formează cu coeficienţii a_{ij}, $i \in \overline{1,m}$, $j \in \overline{1,n}$, trebuie să fie nedegenerată, adică $rang(A) = m$. În plus, dacă dispunem de un set de $n-m$ variabile de bază, spunem că avem o problemă în forma standard restrânsă. După cum vom vedea în continuare, orice problemă de programare liniară de forma (7.83) se poate transforma într-o formă standard restrânsă.

Restricţiile inegalităţi se transformă în restricţii egalităţi cu ajutorul *variabilelor auxiliare*. În cazul în care nu putem izola variabilele de bază ca în exemplele precedente, adică nu putem izola $n - m$ variabile care să apară într-o singură relaţie-restricţie cu coeficient pozitiv, apelăm la *variabile fictive* (sau *artificiale*). Spre exemplu, restricţiile (7.84) se rescriu în forma

$$\begin{aligned} x_j &\geq 0,\ j \in \overline{1,n}\ , \\ z_i &= b_i - \sum_{j=1}^{n} a_{ij}x_j + y_i,\ i \in \overline{1,m_1},\ b_i \geq 0\ , \\ z_k &= b_k - \sum_{k=1}^{n} a_{kj}x_j - y_k,\ k \in \overline{m_1+1,m_2},\ b_k \geq 0\ , \\ z_l &= b_l - \sum_{l=1}^{n} a_{lj}x_j,\ l \in \overline{m_2+1,m},\ b_l \geq 0\ , \end{aligned} \tag{7.91}$$

unde am introdus variabilele auxiliare $y_i \geq 0$, $i \in \overline{1,m_2}$ şi variabilele artificiale $z_i \geq 0$, $i \in \overline{1,n}$. Un set de variabile de bază se obţine prin determinarea valorii maxime pentru funcţia scop auxiliară

$$U' = -\sum_{i=1}^{n} z_i\ , \tag{7.92}$$

care va corespunde cu $z_i = 0$, utilizând acelaşi mod de calcul. În continuare se aplică algoritmul simplex pentru aflarea valorii maxime a funcţiei U date de (7.83). Dacă nu se poate obţine valoarea maximă pentru U' cu $z_i = 0$, înseamnă că nu există nici un vector admisibil cu restricţiile date de (7.84).

Figura 7.7: Domeniul de calcul pentru problema (7.93).

EXEMPLU. Să se găsească minimul funcţiei

$$\begin{gathered} U(x_1,x_2)=F_1^2(x_1,x_2)+F_2^2(x_1,x_2)\ , \\ F_1(x_1,x_2)\equiv x_1+3\lg|x_1|-x_2^2,\ F_2(x_1,x_2)\equiv 2x_1^2+1-x_1x_2-5x_1\ , \end{gathered} \tag{7.93}$$

situat în dreptunghiul $0\le x_1\le 4$, $0\le x_2\le 3$.

Rezolvare . Întrucât domeniul în care se caută soluţiile este destul de mare, se pune problema alegerii unei valori iniţiale. O modalitate o constituie împărţirea domeniul în mai multe subdomenii (12 pătrate în exemplul nostru) prin paralele la axele de coordonate (figura 7.7) şi evaluarea funcţiei U în centrele $C_i (i=1,2,\dots,12)$, obţinându-se valorile din tabelul 7.2.

Tabelul 7.2: Valori ale funcţiei U în domeniul de calcul

Punct	C_1	C_2	C_3	C_4	C_5	C_6
U	0.9945	5.3624	5.9612	31.4492	7.9348	4.8236
Punct	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
U	9.0558	5.0507	24.6631	25.4427	17.0483	0.9060

Pe baza tabelului, se poate stabili o ordine pentru punctele de start în vederea găsirii minimelor: C_{12}, C_1, C_6, C_8 ş.a.m.d.

În tabelul 7.3 sunt prezentate comparativ rezultatele obţinute cu metodele gradient, Fletcher-Reeves, Polak-Ribère, Hestens-Stiefel şi Davidon-Fletcher-Powell. Pentru punctul de start $x^0=(3.5,2.5)$ toate metodele converg la soluţia $x_1=3.48744$, $x_2=2.26163$. S-a folosit testul de convergenţă $\|g^k\|_\infty<10^{-5}$. Metodele *DFP* şi cele de gradient conjugat converg rapid, dar remarcăm numărul mare de evaluări

Tabelul 7.3: Rezultatele obţinute la rezolvarea sistemului (7.93)

Metoda	Nr. iteraţii	Nr. eval. funcţie	Nr. eval. gradient
Gradient simplu	17	478	18
Hestens-Stiefel	6	155	7
Fletcher-Reeves	8	225	9
Polak-Ribère	6	155	7
DFP	6	168	7

Figura 7.8: Drumul parcurs de metodele DFP şi gradient simplu.

ale funcţiei U, acesta datorându-se *exclusiv* preciziei impuse la calculul scalarului α, convergenţa acestor metode depinzând de precizia cu care se calculează α optim. Folosind ca punct de start punctul $(0.5, 0.5)$, metodele de optimizare determină minimul local al funcţiei U dat de $x = (0.47980, -0.32323)$. Folosind aceeaşi condiţie $\|g^k\|_\infty < 10^{-5}$, metoda gradientului are nevoie pentru determinarea minimului de 7 iteraţii, Hestens-Stiefel 8, Fletcher-Reeves 9 şi Polak-Ribière 10 iteraţii.

În figura (7.8) este reprezentat drumul parcurs de metodele *DFP* şi *gradient simplu* pentru punctul de start $x^0 = (3.5, 2.5)$. curbele de nivel $U = K_i$ sunt trasate pentru valorile $K_i \in \{0.906, 0.7, 0.5, 0.3, 0.128, 0.1, 0.05, 0.026, 0.01, 0.0052, 0.00076\}$. Primul pas este identic pentru ambele metode. Se observă ineficienţa metodei gradientului care avansează în zig-zag. Cu cât distanţa faţă de minim scade, metoda gradientului devine şi mai ineficientă. Explicaţia constă în faptul că metoda gradientului se bazează doar pe informaţia furnizată de gradientul funcţiei scop, care este nul în punctul de minim şi foarte mic în vecinătatea acestuia.

Capitolul 8

Rezolvarea sistemelor neliniare

Vom considera n sisteme de n ecuaţii cu n necunoscute, de forma

$$F_i(x_1, x_2, ..., x_n) = 0,\ i \in \overline{1, n}\ , \tag{8.1}$$

unde F_i reprezintă funcţii cunoscute de n variabile $x_1, x_2, ..., x_n$, presupuse *continue*, împreună cu derivatele lor parţiale până la un ordin convenabil (de obicei, până la ordinul doi) pentru ca anumite relaţii de calcul să fie valabile. Se va urmări găsirea soluţiilor reale ale sistemului (8.1) într-un anumit domeniu de interes, domeniu în care se consideră valabile şi proprietăţile de continuitate impuse funcţiilor F_i şi derivatelor lor.

Sistemul (8.1) se scrie vectorial

$$F(x) = 0\ , \tag{8.2}$$

unde $x = (x_1, x_2, \dots, x_n)^T$ şi $F = (F_1, F_2, \dots, F_n)^T$. Notaţia mai apropiată de scrierea iniţială ar fi fost $F(x^T)$, dar s-a renunţat la indicarea transpunerii, fapt neesenţial de altfel. Cu notaţia (8.2), soluţiile reale se caută pe un domeniu $D \subset \mathbf{R}^n$, iar $F : D \to \mathbf{R}^n$.

Metodele de rezolvare pot fi sugerate de metodele utilizate la rezolvarea ecuaţiilor într-o singură variabilă (capitolul 3). Analogia dintre sistemul (8.2) şi ecuaţia într-o singură variabilă (3.1), deşi formală, este utilă. Rezolvarea unui sistem de n ecuaţii neliniare cu n necunoscute este *mult mai dificilă* în

cazul general $n \geq 2$ decât pentru $n = 1$, din mai două motive. În primul rând, pentru $n \geq 2$ nu mai este posibilă *localizarea soluţiei*. Nu mai dispunem de o metodă analoagă metodei înjumătăţirii intervalelor pentru separarea unui interval în care se găseşte sigur o soluţie, ca în cazul unei singure variabile. De fapt, mulţimea vectorilor x nu poate fi ordonată, începând cu $n \geq 2$. Localizarea soluţiei reprezintă un avantaj covârşitor al cazului unidimensional, pentru că o soluţie localizată nu mai poate fi scăpată. Din această cauză, ori de câte ori este posibil, se recomandă retranscrierea sistemului astfel încât rezolvarea acestuia să se reducă la rezolvarea unei probleme unidimensionale. În cazul $n-$dimensional, nu se poate şti niciodată cu certitudine că soluţia se află într-o anumită zonă până când soluţia nu este calculată. În al doilea rând, un sistem de n ecuaţii cu n necunoscute poate avea una, niciuna sau mai multe soluţii reale. În cazul în care sistemul are mai multe soluţii, trebuie găsită acea soluţie care are *semnificaţie fizică* (în general rezolvăm numeric ecuaţii pentru că acestea fac parte dintr-un model matematic care corespunde, în limita unor ipoteze, unui fenomen fizic, economic, etc.).

Rezolvarea unor sisteme de ecuaţii neliniare implică utilizarea unui *algoritm iterativ*: pornind de la o aproximaţie iniţială, algoritmul o va îmbunătăţii de la o iteraţie la alta până când va fi îndeplinită o condiţie de convergenţă. Convergenţa algoritmului, în special pentru problemele multi-dimensionale, poate depinde în mod *esenţial* de cât bună este valoarea iniţială. Această valoare trebuie obţinută, ori de câte ori este posibil, pe baza unui studiu analitic.

8.1 Iteraţia simplă

Metoda iteraţiei simple este, aşa cum sugerează şi numele, atât iterativă cât şi simplă; din păcate însă nu este în general şi eficientă. Prezenţa ei în acest capitol reprezintă un punct de plecare şi de comparaţie pentru celelalte metode.

Sistemul (8.2) se mai poate scrie sub forma echivalentă

$$F_i \equiv x_i - f_i(x) = 0,\ i \in \overline{1,n}\ , \tag{8.3}$$

unde $f_i(x)$ sunt funcţii de n variabile. *Iteraţia simplă* este definită prin şirul de iteraţii

$$x_i^{k+1} = f_i(x^k),\ i \in \overline{1,n},\ k = 0, 1, 2, \dots\ , \tag{8.4}$$

unde indicele superior indică iteraţia. Startul se ia cu vectorul iniţial x^0 din domeniul D în care se caută soluţia.

Presupunând că există o soluţie reală $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$, astfel încât $\alpha = f_i(\alpha)$, vom putea calcula distanţele la aceasta cu ajutorul diferenţelor

$$x_i^{k+1} - \alpha_i = f_i(x^k) - \alpha_i = f_i(x^k) - f_i(\alpha) = \sum_{j=1}^{n} (x_j^k - \alpha_j) \frac{\partial f_i}{\partial x_j}(\xi^k),\ i \in \overline{1,n}\ .$$

Ultima egalitate de mai sus s-a obţinut aplicând formula creşterilor finite pentru o funcţie de n variabile. Derivatele parţiale $\partial f_i / \partial x_j$ se evaluează în punctul ξ^k

aparţinând cubului $n-$dimensional (x^k, α); ξ^k depinde atât de iteraţie cât şi de funcţia f_i. Notând mai departe cu m_i modulul maxim al derivatelor parţiale $m_i = \sup_j |\partial f_i(x)/\partial x_j|\,,\ x \in (x^k, \alpha)$, se obţin imediat inegalităţile

$$|x_i^{k+1} - \alpha_i| \le m_i \sum_{j=1}^{n} |x_j^k - \alpha_j|,\ i \in \overline{1,n}\ . \tag{8.5}$$

Se observă că distanţa dintre două componente $|x_i^{k+1} - \alpha_i|$ are o margine care depinde de distanţele dintre toate componentele vectorilor x^k şi α. Ajungem astfel, în mod firesc, la necesitatea de a introduce *norme de vectori*. Spre exemplu, luând modulul maxim al diferenţelor componentelor celor doi vectori x şi α, se obţine

$$\|x^{k+1} - \alpha\|_\infty \le n \cdot m \cdot \|x^k - \alpha\|_\infty, \text{ unde } m \equiv \sup_i m_i\ . \tag{8.6}$$

Din relaţia (8.6) rezultă că o *condiţie suficientă* pentru ca şirul de iteraţii (8.4) să fie convergent către soluţia α este

$$n \cdot m < 1, \text{ sau } \sup_{i,j} |\partial f_i/\partial x_j| < 1/n,\ i,j \in \overline{1,n}\ . \tag{8.7}$$

Condiţia (8.7) coincide cu condiţia suficientă de convergenţă (3.13) pentru $n = 1$. Se observă că restricţia este cu atât mai severă, cu cât numărul de variabile independente este mai mare.

O sumare a inegalităţilor (8.5) după indicele i conduce la norma $\|\ \|_1$ (suma modulelor componentelor vectorilor)

$$\|x^{k+1} - \alpha\|_1 \le \|x^k - \alpha\|_1 \sum_{i=1}^{n} m_i \le n \cdot m \cdot \|x^k - \alpha\|_1\ . \tag{8.8}$$

Se observă că utilizarea normei $\|\ \|_1$ conduce la aceeaşi *condiţie suficientă* de convergenţă (8.7).

O condiţie mai slabă de convergenţă se poate obţine scriind succesiv

$$|x_i^{k+1} - \alpha_i| \le \|x^k - \alpha\|_\infty \sum_{j=1}^{n} \left|\frac{\partial f_i(\xi^k)}{\partial x_j}\right|,\ i \in \overline{1,n},\ \xi^k \in (x^k, \alpha)\ , \tag{8.9}$$

sau pentru indicele $i_{\max}$ care corespunde normei vectoriale maxime

$$\|x^{k+1} - \alpha\|_\infty \le \|x^k - \alpha\|_\infty \sum_{j=1}^{n} \left|\frac{\partial f_{i_{\max}}(\xi^k)}{\partial x_j}\right|\ . \tag{8.10}$$

Deşi ξ^k depinde şi de indicele i prin forma funcţiei $f_i(x)$, putem evalua suma din dreapta inegalităţilor (8.9) şi (8.10) în funcţie de norma maximă a matricei jacobiene J a sistemului de funcţii f

$$\sum_{j=1}^{n} \left|\frac{\partial f_i(\xi^k)}{\partial x_j}\right| \le \sup_i \sum_{j=1}^{n} \left|\frac{\partial f_i(\xi^k)}{\partial x_j}\right| \le \|J(\bar{\xi}^k)\|_\infty\ , \tag{8.11}$$

$\bar{\xi}^k$ fiind un punct independent de indicele i. Prin urmare, se obţine *condiţia suficientă de convergenţă*

$$\|J(\bar{\xi}^k)\|_\infty < 1,\ \bar{\xi}^k \in (x^k, \alpha)\ . \tag{8.12}$$

Condiţia (8.12) este în general mai slabă decât (8.7).

O formulare generală a condiţiei în care se realizează convergenţa iteraţiei simple este (într-un spaţiu Banach) ca sistemul de funcţii f să fie *contractant*, adică pentru orice pereche de puncte x^1, x^2 să se realizeaze inegalitatea

$$\|f(x^2) - f(x^1)\| \le q\|x^2 - x^1\|,\ \text{unde } q \in [0,1)\ , \tag{8.13}$$

Condiţiile de convergenţă se pot testa în domeniul în care se află soluţia. Aceste testări pot implica însă un volum mare de calcul, când numărul de variabile este mare sau/şi soluţia este localizată slab.

Din existenţa a două condiţii (8.7) şi (8.12) care, în general, nu coincid, putem înţelege mai bine în ce sens aceste condiţii sunt doar suficiente. Ele nu sunt neapărat necesare, în sensul că ar putea exista o condiţie mai puţin restrictivă care să asigure descreşterea distanţei la rădăcină, începând, dacă nu chiar cu prima iteraţie, cel puţin cu un anumit $k > 1$.

Dacă suntem suficient de aproape de soluţie, condiţia (8.12) poate fi scrisă folosind orice altă normă a matricei jacobiene. Viteza de convergenţă este cu atât mai mare cu cât norma (subunitară) este mai mică.

În cazul când (8.13) este satisfăcută, distanţa la rădăcină poate fi evaluată cu relaţia

$$\|x^{k+1} - \alpha\| \le \frac{q^k}{1-q}\|x^1 - x^0\|,\ q \in [0,1]\,,\ k = 1, 2, \dots\ . \tag{8.14}$$

8.2 Metoda iterativă Newton

Ca şi în cazul unei singure variabile (de fapt, chiar într-o măsură mai mare), condiţiile suficiente de convergenţă ale iteraţiei simple sunt destul de restrictive pentru a o face impracticabilă în majoritatea cazurilor. De aceea, vom căuta alte metode iterative cu domeniu mai larg şi viteză mai mare de convergenţă. În acest scop, vom aplica formula lui Taylor pentru o funcţie de n variabile $F_i(x)$ sub forma

$$F_i\,(x) = F_i(x^k) + \sum_{j=1}^{n}(x_j - x_j^k)\frac{\partial F_i}{\partial x_j} + \frac{1}{2}\sum_{j,p=1}^{n}(x_j - x_j^k)(x_p - x_p^k)\frac{\partial^2 F_i(\xi^k)}{\partial x_j \partial x_p}\ .$$

Dacă în expresiile de mai sus neglijăm ultimul termen şi facem $x = x^{k+1}$, atunci condiţia de deplasare spre soluţie, $F_i(x^{k+1}) = 0$, ne conduce la sistemul

$$\sum_{j=1}^{n}(x_j^{k+1} - x_j^k)\frac{\partial F_i(x^k)}{\partial x_j} = -F_i(x^k),\ i \in \overline{1,n}\ , \tag{8.15}$$

care reprezintă *metoda iterativă Newton*, ce ne furnizează vectorul x în aproximaţia $k+1$ în funcţie de datele din aproximaţia anterioară.

Matricea sistemului (8.15) se construieşte cu derivatele parţiale de ordinul întâi ale funcţiilor $F_i(x),\ i \in \overline{1,n}$, formând matricea jacobiană a sistemului de funcţii F care se notează cu $J(x)$, $F'(x)$ sau $F_x(x)$. Vom adopta prima notaţie.

Matriceal, sistemul de ecuaţii (8.15) se scrie

$$J(x^k)(x^{k+1} - x^k) = -F(x^k)\ . \tag{8.16}$$

Sistemul de ecuaţii liniare (8.15) sau (8.16) se poate rezolva cu metoda eliminării gaussiene (capitolul 5). Matricea jacobiană se calculează analitic sau, dacă nu este posibil, numeric, utilizând diferenţe finite.

Pentru a studia convergenţa metodei, considerăm $x = \alpha$ (α fiind o soluţie). Obţinem

$$0 = F_i(x^k) + \sum_{j=1}^{n} (\alpha_j - x_j^k)\frac{\partial F_i}{\partial x_j} + V_i,\ i \in \overline{1,n}\ , \tag{8.17}$$

unde s-a notat cu V_i cantitatea

$$V_i = \frac{1}{2}\sum_{j,p=1}^{n} (\alpha_j - x_j^k)\frac{\partial^2 F_i(\xi^k)}{\partial x_j \partial x_p}(\alpha_p - x_p^k),\ i \in \overline{1,n}\ . \tag{8.18}$$

Vectorial, relaţia (8.17) se poate scrie

$$J(x^k)(\alpha - x^k) = -F(x^k) - V\ . \tag{8.19}$$

Scăzând egalităţile (8.16) şi (8.19), se obţine

$$J(x^k)\cdot(x^{k+1} - \alpha) = V,\ \text{sau}\ x^{k+1} - \alpha = J^{-1}(x^k)\cdot V\ . \tag{8.20}$$

Din (8.20), folosind o normă oarecare se obţine o evaluare a distanţei la rădăcină în aproximaţia $k+1$

$$\|x^{k+1} - \alpha\| \le \|J^{-1}(x^k)\|\cdot\|V\|\ . \tag{8.21}$$

Dacă se ia norma maximă, din (8.18) rezultă

$$\|V\|_\infty = \sup_i |V_i| \le M\sum_{j,p=1}^{n} |x_j^k - \alpha_j|\cdot|x_p^k - \alpha_p| \le n^2 M\|x^k - \alpha\|_\infty^2\ , \tag{8.22}$$

$$\text{unde}\ M = \frac{1}{2}\sup_{i,j,p}\left|\frac{\partial^2 F_i}{\partial x_j \partial x_p}\right|\ . \tag{8.23}$$

Se obţine deci

$$\|x^{k+1} - \alpha\|_\infty \le C\|x^k - \alpha\|_\infty^2,\ \text{unde}\ C = M\cdot n^2\cdot\sup_k \|J^{-1}(x^k)\|_\infty\ . \tag{8.24}$$

Plecând de la o aproximaţie iniţială x^0, se obţin succesiv

$$\begin{array}{c} \|x^1-\alpha\|_\infty \le C\cdot\|x^0-\alpha\|_\infty^2 = C^{-1}(C\cdot\|x^0-\alpha\|_\infty)^{2^1} \\ \|x^2-\alpha\|_\infty \le C\cdot\|x^1-\alpha\|_\infty^2 = C^{-1}(C\cdot\|x^0-\alpha\|_\infty)^{2^2} \\ \cdots \\ \|x^k-\alpha\|_\infty \le C^{-1}(C\cdot\|x^0-\alpha\|_\infty)^{2^k}\ . \end{array} \tag{8.25}$$

Prin urmare, metoda converge dacă este satisfăcută inegalitatea

$$C\cdot\|x^0-\alpha\|_\infty < 1\ . \tag{8.26}$$

Dacă se utilizează norma 1, cu ajutorul relaţiei (8.18) se obţine

$$\begin{array}{c} \|V\|_1 = \sum_{i=1}^n |V_i| \le \|x^k-\alpha\|_1^2 \sum_{i=1}^n M_i \le nM\|x^k-\alpha\|_1^2\ , \\ \text{unde } M_i = \frac{1}{2}\sup\limits_{i,j}\left|\dfrac{\partial^2 F_i}{\partial x_j \partial x_p}\right|,\ \text{iar } M = \sup\limits_i M_i\ . \end{array} \tag{8.27}$$

Introducând apoi norma 1 în relaţia (8.21) şi folosind inegalitatea (8.27), rezultă

$$\|x^{k+1}-\alpha\|_1 \le C_1\|x^k-\alpha\|_1^2,\ \text{unde } C_1 = M\cdot n\cdot\sup_k\|J^{-1}(x^k)\|_1\ . \tag{8.28}$$

Parcurcând aceleaşi etape ca în relaţiile (8.21), rezultă că, dacă este îndeplinită inegalitatea

$$C_1\|x^0-\alpha\|_1 < 1\ , \tag{8.29}$$

metoda are o convergenţă pătratică, ca şi în cazul când utilizăm norma maximă. Condiţia (8.29) poate fi mai avantajoasă decât (8.26), deoarece scalarul C_1 este proporţional cu n, în comparaţie cu scalarul C, care este proporţional cu n^2. În schimb, norma 1 este mai mare, în general, decât norma maximă fiind satisfăcută însă inegalitatea

$$n\cdot\|x^0-\alpha\|_\infty \ge \|x^0-\alpha\|_1,\ \text{pentru } n\ge 1\ . \tag{8.30}$$

Convergenţa are loc întotdeauna dacă este satisfăcută o condiţie de forma (8.26) sau (8.29), independent de norma utilizată. În toate cazurile, pentru verificarea condiţiilor de convergenţă, sunt necesare evaluări ale inversei matricei jacobiene şi ale derivatelor parţiale de ordinul al doilea ale funcţiilor $F_i(x)$, distanţa $x^0-\alpha$ fiind însă necunoscută.

O condiţie de convergenţă care utilizează rezultatele calculului într-o primă aproximaţie a fost dată de *Kantorivici*. În acest scop se evaluează derivatele de ordinul 2 într-o sferă închisă U_0 de centru x^0 şi de rază $2\left\|x^1-x^0\right\|$. Condiţia de convergenţă este

$$h_0 \equiv 2K\|J^{-1}(x^0)\|_\infty\|x^1-x^0\|_\infty \le 1\ , \tag{8.31}$$

$$\text{unde } K = \sup_i\sum_{j,p}\sup_{x\in U_0}\left|\frac{\partial^2 f_i(x)}{\partial x_j\partial x_p}\right|\ . \tag{8.32}$$

Notând cu $\alpha \in U_0$ rădăcina, eroarea este evaluată prin relaţia

$$\|x^k - \alpha\|_\infty \leq (1/2)^{k-1}(h_0^{2^k-1})\|x^1 - x^0\|_\infty \ . \tag{8.33}$$

Rezultă în toate cazurile că, dacă jacobiana $J(x)$ este nesingulară şi derivatele de ordinul 2 sunt mărginite problema care mai rămâne este găsirea unui punct de plecare x^0 *"suficient de aproape"* de soluţie. Desigur, se poate proceda la împărţirea domeniului D în subdomenii mai mici, în care să căutăm soluţia succesiv plecând, spre exemplu, din centrul subdomeniului. Reiniţializarea s-ar face dacă după un număr fixat de iteraţii şirul x^k nu se stabilizează, sau iese în afara subdomeniului considerat. Acest mod de lucru se poate dovedi însă nepractic în cazul unui număr mare de variabile. Spre exemplu, dacă fiecare latură a unui cub $n-$dimensional se divide în N părţi, numărul de subdomenii care rezultă este N^n, ceea ce duce la numere foarte mari pentru N, $n \geq 5$. De aceea, pentru apropierea de soluţie se utilizează metode care converg în condiţii mai largi. Acestea sunt *metodele de descreştere* prezentate în capitolul 7.

8.3 Metode cvasi-Newton

Am văzut că metoda Newton cere evaluarea matricei jacobiene $J(x)$, adică evaluarea a n^2 funcţii la fiecare iteraţie. Dacă nu este posibilă obţinerea pe cale analitică a expresiilor derivatelor parţiale ce intră în componenţa matricei jacobiene, valoarea acestora poate fi aproximată folosind o formulă cu diferenţe finite de forma

$$(\partial F_i)/(\partial x_j) \approx [F_i(x_1, x_2, \ldots, x_j + h, \ldots, x_n) - F_i(x_1, x_2, \ldots, x_n)]/h \ , \tag{8.34}$$

fapt care nu reduce însă numărul de operaţii. Numărul de operaţii poate fi redus dacă se utilizează o matrice B care aproximează jacobiana, astfel încât în locul sistemului (8.16) se va scrie

$$B_k s^k = -F^k \ , \tag{8.35}$$

unde $s^k \equiv x^{k+1} - x^k$ şi $F^k = F(x^k)$. Aproximaţia B_k a jacobienei trebuie actualizată în etapa următoare, lucru se face scriind că B_{k+1} aproximează $J(x^{k+1})$, adică

$$F^k \cong F^{k+1} + J(x^{k+1})(x^k - x^{k+1}) \cong F^{k+1} - B_{k+1}s^k \ . \tag{8.36}$$

Rezultă relaţia *cvasi-Newton*

$$B_{k+1}s^k = y^k, \text{ unde } y^k \equiv F^{k+1} - F^k \ . \tag{8.37}$$

Relaţia (8.35) nu este suficientă pentru determinarea elementelor matricei B_{k+1}. Ea se completează cu condiţia ca matricele B_{k+1} şi B_k să difere cât mai puţin de la o iteraţie la alta, adică B_{k+1} se determină ca soluţie a problemei de optimizare

$$B_{k+1} = \min\{\|B - B_k\|_2 \mid B \cdot s^k = y^k\} \ . \tag{8.38}$$

Vom rezolva problema de optimizare (8.38) indirect, impunând ca B_k şi B_{k+1} să nu difere pe direcţiile perpendiculare pe s^k, adică pentru orice vector z pentru care $z^T \cdot s^k = 0$, să avem

$$B_k z = B_{k+1} z \ . \tag{8.39}$$

Din (8.39) şi (8.37) se deduce că $(B_{k+1} - B_k)$ se poate alege de forma

$$B_{k+1} - B_k = c \cdot (s^k)^T \tag{8.40}$$

c fiind un vector arbitrar. Din relaţiile (8.35) şi (8.37) rezultă succesiv

$$(B_{k+1} - B_k)s^k = F^{k+1} \Rightarrow c(s^k)^T s^k = F^{k+1} \Rightarrow c = F^{k+1}/(s^k)^T s^k \tag{8.41}$$

Rezultă formula iterativă *Broyden*

$$B_{k+1} = B_k + \frac{F^{k+1}(s^k)^T}{(s^k)^T s^k} = B_k + \frac{(y^k - B_k s^k)(s^k)^T}{(s^k)^T s^k} \ . \tag{8.42}$$

Vom arăta acum că, matricele B_{k+1} furnizate de (8.42) sunt cele mai apropiate în normă euclidiană sau Frobenius de matricele B_k. Într-adevăr, se poate scrie

$$\begin{aligned} \|B_{k+1} - B_k\|_2 &= \|(y^k - B_k s^k)(s^k)^T\|_2/(s^k)^T s^k = \|(B - B_k)s^k(s^k)^T\|_2/(s^k)^T s^k \\ &\le \|B - B_k\|_2 \, \|s^k(s^k)^T\|_2/(s^k)^T s^k = \|B - B_k\|_2 \ . \end{aligned}$$

Am ţinut seama că norma euclidiană, a matricei $s^k(s^k)^T$ este egală cu produsul scalar $(s^k)^T \cdot s^k$

$$\|s \cdot s^T\|_2^2 \equiv \sum_{i=1}^n \sum_{j=1}^n s_i^2 s_j^2 = \sum_{i=1}^n s_i^2 \sum_{j=1}^n s_j^2 = [(s)^T \cdot s]^2 \ . \tag{8.43}$$

În acest fel, plecând de la o aproximaţie iniţială B_0 a jacobienei (care ar putea fi chiar jacobiana în x^0, $B_0 = J(x^0)$), matricele B se construiesc cu numai n evaluări de funcţii. Preţul acestei simplificări îl repezintă scăderea vitezei de convergenţă de la pătratică la superliniară.

Rezolvarea sistemului de ecuaţii (8.35), care necesită circa $n^3/3$ operaţii, poate fi evitată construind iterativ matricea inversă B_k^{-1}. Un avantaj al formei (8.40) este că se matricea B_{k+1} se poate inversa direct. Mai general, oricare ar fi matricea A şi vectorii u, v, există egalitatea

$$(A + u \cdot v)^{-1} = A^{-1} - \frac{A^{-1} u \, v^T A^{-1}}{1 + v^T \cdot (A^{-1} u)} \ . \tag{8.44}$$

Relaţia (8.44) se verifică direct, înmulţind cu matricea $(A + u \cdot v^T)$. Făcând $A = B_k$, $u = c$, $v = s^k$, din (8.44) se obţine

$$H_{k+1} = H_k + \frac{(s^k - H_k y^k)(s^k)^T H_k}{(s^k)^T (H_k y^k)}, \text{ unde } H_k = B_k^{-1} \ , \tag{8.45}$$

cu condiţia $(s^k)^T \cdot (H_k y^k) \neq 0$. În acest fel, rezultă

$$x^{k+1} = x^k - H_k F^k \ . \tag{8.46}$$

Aplicarea formulei (8.46) cere evaluarea a n funcţii (F^{k+1}) şi $O(n^2)$ operaţii pentru determinarea poziţiei x^{k+1}. Prin urmare, numărul de operaţii scade în comparaţie cu (8.37) dar metoda se dovedeşte mai puţin stabilă din punct de vedere numeric.

Evitarea rezolvării sistemului de ecuaţii liniare (8.35) se poate face folosind o factorizare $B_k = Q_k R_k$ unde Q_k este ortogonală şi R_k este superior triunghiulară. Rezolvarea sistemului (8.35) se va face în $O(n^2)$ operaţii, iar matricea B_{k+1} se poate obţine tot în $O(n^2)$.

În concluzie, metodele cvasi-Newton reduc numărul de operaţii pentru o iteraţie cu preţul realizării doar a unei convergenţe locale.

8.4 Metoda gradientului

Aplicarea metodei gradientului prezentată în capitolul 7 la rezolvarea sistemelor de ecuaţii neliniare se bazează pe observaţia că rezolvarea sistemului (8.1) poate fi formulată ca o problemă de minimizare pentru o funcţie ajutătoare $U(x)$, care va fi scrisă sub forma

$$U(x) = \tfrac{1}{2} \sum\nolimits_{j=1}^{n} F_j^2(x) \ . \tag{8.47}$$

În acest fel, vom avea avantajul convergenţei globale a metodei gradientului, în raport cu metodele de tip Newton care au convergenţă locală. Rezolvarea sistemului de ecuaţii $F_j(x) = 0$ este echivalentă cu găsirea unui minim $U = 0$. Vectorul gradient al funcţiei ajutătoare este dat de relaţia

$$\nabla U(x) \equiv g(x) = J^T(x) J(x) \ . \tag{8.48}$$

Ţinând seama de forma specială (8.47) a funcţiei U şi de convergenţa şirului $U(x^k)$ la zero, putem da unele expresii simplificate pentru parametrul α_k. Folosind spre exemplu o aproximaţie liniară pentru $U(x^k - \alpha_k g^k)$ vom scrie, dezvoltând în serie în jurul valorii $\alpha_k = 0$

$$\alpha_k = U(x^k)/(g^k)^T \cdot g^k, \ (g^k)^T g^k = \sum\nolimits_{j=1}^{n} (g_j^k)^2 \ . \tag{8.49}$$

Rezultă noua poziţie a punctului x în aproximaţia $k+1$

$$x^{k+1} = x^k - g^k \cdot U(x^k) / \sum\nolimits_{j=1}^{n} (g_j^k)^2 \ , \tag{8.50}$$

cu condiţia ca funcţia scop să descrească, adică $U(x^{k+1}) < U(x^k)$.

Putem însă folosi o aproximaţie liniară pentru funcţiile $F_j(x^k - \alpha_k g^k)$, $j \in \overline{1,n}$, de forma

$$F_j(x^k - \alpha_k g^k) \cong F_j(x^k) - \alpha_k \sum_{p=1}^{n} (\partial F_j(x^k)/\partial x_p) \cdot g_p^k \ . \tag{8.51}$$

Funcţia $f(\alpha_k) \equiv U(x^k - \alpha_k g^k)$ şi derivata $f'(\alpha_k)$ le putem aproxima cu relaţiile

$$f(\alpha_k) \cong \frac{1}{2}\sum_{j=1}^{n}\left[F_j(x^k) - \alpha_k v_j^k\right]^2, \quad v_j^k \equiv \sum_{p=1}^{n}\frac{\partial F_j(x^k)}{\partial x^p} g_p^k \ , \tag{8.52}$$

$$f'(\alpha_k) \cong -\sum_{j=1}^{n}\left[F_j(x^k) - \alpha_k v_j^k\right] v_j^k \ , \tag{8.53}$$

unde am notat cu v_j^k, $j \in \overline{1,n}$ componentele vectorului $v^k = J(x^k)\cdot g^k$ obţinut prin înmulţirea matricei jacobiene J cu vectorul gradient g. Folosind relaţia (8.53), din ecuaţia $f'(\alpha_k) = 0$ rezultă o valoare aproximativă pentru α_k optim

$$\alpha_k^* \cong \sum_{j=1}^{n} F_j(x^k)\cdot v_j^k / \sum_{j=1}^{n}(v_j^k)^2 = (v^k)^T \cdot F(x^k)/(v^k)^T \cdot v^k \ , \tag{8.54}$$

noua poziţie a punctului de minim aproximativ fiind $x^{k+1} = x^k - \alpha_k^* g^k$.

Expresiile aproximative (8.49) şi (8.54) sunt utile deoarece convergenţa metodei gradientului nu este condiţionată de determinarea precisă a valorii α_k^*; singura condiţie este ca valoarea funcţiei U să scadă, adică $U_{k+1} < U_k$. Determinarea unei valori mai exacte pentru α_k optim se poate face cu una din metodele prezentate la 7.1. Deoarece, pe măsură ce ne apropiem de minim, componentele gradientului se micşorează (în punctul de minim avem, prin definiţie $g = 0$), erorile introduse prin calcul fac ca direcţia spre minim să fie din ce în ce mai puţin precis determinată. Pe de altă parte, deplasările spre minim devin din ce în ce mai mici. În consecinţă, metoda gradientului converge rapid în primii paşi, apoi din ce în ce mai lent. Pe de altă parte, metoda Newton are o comportare inversă, viteza ei de convergenţă crescând pe măsură ce ne apropiem de soluţie. Aceste observaţii sugerează o combinare a celor două metode în felul următor: plecând dintr-un punct x^0, se aplică metoda gradientului câteva iteraţii k_0 ($k_0 \le n$), atâta timp cât converge rapid, verificându-se şi dacă funcţia $U \to 0$. Se aplică apoi metoda Newton, plecând din punctul x_{k_0}, pentru a creşte viteza de convergenţă şi a realiza o precizie ridicată a soluţiei. Procesul este asemănător etapelor de localizare şi rafinare de la rezolvarea unei singure ecuaţii neliniare $f(x) = 0$ (3).

Algoritmul metodei hibride.

Date: x^0, $U(x)$, $J(x)$, δ_0, λ_0, ε, *maxiter*
$x \leftarrow x^0$; $\delta \leftarrow \delta_0$; $F^k \leftarrow F(x^0)$; $U_k \leftarrow U(x^0)$
$k = 1$: *maxiter*
- *Calculul direcţiei de descreştere* p^k
- $x \leftarrow x + p$; $F^{k+1} \leftarrow F(x)$; $J_k \leftarrow J(x)$
- [*Actualizare* δ] - ca la MRI (7.35).
- dacă $U_{k+1} > U_k$ *eşec* **stop.**
- dacă $|U_{k+1} - U_k| < \varepsilon|U_{k+1}|$minim calculat **stop.**
- $F^k \leftarrow F^{k+1}$

(8.55)

Figura 8.1: Direcţia de descreştere pentru metoda hibridă.

$J_k \leftarrow J(x)$
rezolvă sistemul $J_k p_N = -F^k$
dacă $\|p_N\| \leq \delta$
  atunci $p \leftarrow p_N$
  altfel
    $p_G \leftarrow \{-\alpha J_k^T F^k | U(x - \alpha J_k^T F^k) = \text{minim}\}$
    $n_G \leftarrow \|p_G\|$
    dacă $n_G \leq \delta$
      atunci $p \leftarrow \{p_G + \gamma(p_N - p_G) \,|\, \|p\| = \delta\}$
      altfel $p \leftarrow (\delta / n_G) p_g$

8.5 Metoda hibridă

Această metodă face parte din clasa metodelor regiunii de încredere în model prezentate la 33. În principiu, metoda hibridă urmăreşte combinarea avantajului de convergenţă locală pătratică a metodei Newton cu convergenţa globală a metodei gradientului. Pentru funcţia scop U de forma (7.72) se foloseşte ca aproximantă o formă pătratică dată de (7.13). Pentru matricea hessian se poate folosi aproximaţia dată de (7.77). Dacă evaluarea exactă a derivatelor parţiale este imposibilă sau foarte dificilă, putem fie să folosim o matrice care aproximează hessiana ca în metoda Broyden, fie să le evaluăm numeric folosind diferenţe finite de forma

$$\partial F_i / \partial x_j \cong [F(x_1, \ldots, x_j + h, \ldots, x_n) - F(x_1, \ldots, x_j, \ldots, x_n)]/h \ . \quad (8.56)$$

Algoritmul metodei hibride este asemănător cu cel al metodei regiuni de încredere, cu o deosebire la calculul direcţiei de descreştere. Mai întâi se încearcă avansul după direcţia Newton dată de (7.31). Dacă pasul Newton p_N^k iese din afara regiunii de încredere (fig.8.1), adică $\|p_N^k\|_2 > \delta_k$, atunci se calculează pasul optimal pe direcţia gradientului negativ $p_G^k = -\alpha_k^* g^k$, iar dacă $\|p_G^k\|_2 < \delta_k$, atunci

Figura 8.2: Curbele de nivel ale funcţiilor F_1 şi F_2.

se caută caută o deplasare convenabilă între cele două direcţii, folosind o medie ponderată $p^k = \gamma p_N^k + (1-\gamma)p_G^k$. Observăm cum pentru $\gamma = 1$ regăsim direcţia Newton, iar pentru $\gamma = 0$ deplasarea se face pe direcţia gradientului negativ cu pas optimal. Algoritmul pentru metoda hibridă de rezolvare a sistemelor de ecuaţii neliniare este dat de (8.55).

EXEMPLU 1. Să se găsească soluţiile sistemului de ecuaţii

$$\begin{cases} x_1 + 3\lg|x_1| = x_2^2 \\ 2x_1^2 + 1 = x_1x_2 + 5x_1 \ , \end{cases} \tag{8.57}$$

situate în dreptunghiul $0 \le x_1 \le 4$, $0 \le x_2 \le 3$.

Rezolvare. Sistemul (8.57) rescrie în forma

$$\begin{cases} F_1(x_1, x_2) \equiv x_1 + 3\lg|x_1| - x_2^2 = 0 \\ F_2(x_1, x_2) \equiv 2x_1^2 + 1 - x_1x_2 - 5x_1 = 0 \ . \end{cases} \tag{8.58}$$

Vom utiliza metodele Newton şi Broyden, precum şi metoda gradientului cu relaţiile aproximative (8.49) şi (8.54) pentru calculul scalarului α. Metodele de optimizare de gradient, gradient conjugat şi Davidon-Fletcher-Powell au fost aplicate în exemplul din capitolul (7) pentru funcţia scop $U = \frac{1}{2}(F_1^2 + F_2^2)$. Matricea jacobiană a sistemului de funcţii (F_1, F_2) este

$$J = \begin{pmatrix} 1 + 3(\lg e)/x_1 & -2x_2 \\ 4x_1 - x_2 - 5 & -x_1 \end{pmatrix} \ . \tag{8.59}$$

Curbele de nivel[1] $F_1 = 0$ şi $F_2 = 0$ sunt date în figura (8.2). Soluţia sistemului (8.58) reprezintă din punct de vedere geometric, punctele în care curbele de nivel din planul $z = 0$ ale celor două funcţii se intersectează. După cum se vede şi din figură,

[1] *Curba de nivel* reprezintă acea curbă din planul $z = K$ pe care o funcţie de două variabile $z(x, y)$ are aceeaşi valoare K. Pentru trei variabile avem *suprafeţe de nivel*.

Tabelul 8.1: Rezultatele obţinute la rezolvarea sistemului (8.57)

Metoda	Nr. iteraţii	Nr. eval. funcţie	Nr. eval. gradient
Newton	3	4	4
Broyden	14	15	0
Gradient (8.49)	34	104	35
Gradient (8.54)	34	104	35

sistemul admite două soluţii, dintre care numai una este în domeniul de interes. Se observă că mai există o zonă, marcată cu cercul A, unde curbele de nivel ale celor două funcţii se apropie destul de mult. Astfel de zone sunt dificile şi pot provoca divergenţa metodelor de rezolvare.

Drept criterii de convergenţă se recomandă, într-o normă de vectorială oarecare

$$\|F^{k+1}\| < \varepsilon_F, \ \|x^{k+1} - x^k\| < \varepsilon_x \|x^{k+1}\| \ , \tag{8.60}$$

unde valorile ε_F şi ε_x se aleg în funcţie de precizia dorită, ε_x este de forma 10^{-t}, cu t numărul de cifre semnificative exacte dorite pentru soluţie. Pentru $\varepsilon_F = 10^{-11}$, cu metoda Newton folosind ca punct de plecare $x_1 = (3.5, 2.5)$, se determină soluţia $x_1 = 3.48744$, $x_2 = 2.26163$ în 3 iteraţii (tabelul 8.1). Cu metoda gradientului soluţia este obţinută după 34 iteraţii cu ambele formule (8.49) şi (8.54) pentru scalarul α. Deşi s-au efectuat mai multe iteraţii decât cu metoda gradientului cu minimizare unidimensională mai exactă (când au fost necesare numai 17 iteraţii), numărul de evaluări ale funcţiei scop U este sensibil mai mic, 104 faţă de 478.

Metoda Newton eşuează dacă se porneşte din punctul $x_1 = (0.5, 0.5)$, deşi valoarea funcţiei U în acest punct este relativ mică, $U = 0.9945$. La fel se întâmplă şi pentru alte puncte de plecare cu valori x_1 apropiate de zero. Explicaţia constă în faptul că atât valoarea funcţiei F_1, cât şi valoarea derivatei $\partial F_1/\partial x_1$ sunt infinite în $x = 0$. Plecând din acelaşi punct, $x_1 = (0.5, 0.5)$, metoda gradientului cu scalarul α calculat cu relaţia (8.49) oscilează circa 50 iteraţii în jurul minimului local $U(0.4798, -0.32323) \cong 0.476$, după care "scapă" şi "sare" la cea dea doua soluţie a sistemului, $x_1 = 1.4589$, $x_2 = -1.39677$. Folosind formula (8.49), metoda gradientului oscilează în jurul minimul local al funcţiei U. Practic determină acest minim, care însă nu este o soluţie a sistemului (8.58).

Cea de-a doua soluţie a sistemului (care nu este cerută în enunţul problemei), $x_1 = 1.4589$, $x_2 = -1.39677$, poate fi obţinută cu metoda Newton în 5 iteraţii pornind din punctul $x^0 = (2.5, -0.5)$.

Trebuie menţionat că problema propusă poate fi rezolvată mai simplu, exprimând pe x_2 din ecuaţia a doua în funcţie de x_1 şi înlocuind în prima ecuaţie. Se obţin relaţiile

$$x_2 = 2x_1 - 5 + 1/x_1, \quad F(x_1) = x_1 + 3\lg|x_1| - (2x_1 - 5 + 1/x_1)^2 \ = \ 0 \ .$$

Graficul funcţiei $F(x_1)$ este dat în figura 8.3. Ecuaţia obţinută se poate rezolva cu metoda Newton-Raphson, sau orice altă metodă de rezolvare a unei ecuaţii neliniare. Avantajul principal constă în posibilitatea *localizării* rădăcinilor. Reducerea numărului

Figura 8.3: Graficul funcţiei $F(x)$.

de ecuaţii cuplate în sistem prin substituiri, este în general avantajoasă. În plus, ori de câte ori este posibil, este benefică obţinerea unei dependenţe de o singură variabilă pentru a folosi avantajul localizării soluţiei.

Exemplu 2. Să considerăm o problemă reală din domeniul investigaţiilor radar. Un impuls radar pleacă din E şi suferă reflexii de pe două frontiere. Un receptor, plasat în R, măsoară timpul după care este recepţionat semnalul, T_2. Un alt receptor plasat în E măsoară timpul de parcugere pe verticală, t_2. Interesează determinarea adâncimii z_1. Necunoscutele sunt x_1, c_1 şi z_1. Poziţia punctului de refracţie depinde de proprietăţile dielectrice ale mediului. Timpul de parcurgere a distanţei $EABCR$ este

$$T_2 = \sqrt{h^2 + (d - x_1)^2}/c + \sqrt{z_1^2 + x_1^2}/c_1 \,. \tag{8.61}$$

Poziţia punctului de refracţie se determină din aplicarea principiului lui Fermat

$$\partial T_2/\partial x_1 = \dot{0} \tag{8.62}$$

Rezultă următorul sistem de ecuaţii

$$f_1(x_1, c_1, z_1) = t_2 - 2(h/c + z_1/c_1) \tag{8.63}$$

$$f_2(x_1, c_1, z_1) = T_2 - 2\left(\sqrt{h^2 + (d - x_1)^2}/c + \sqrt{z_1^2 + x_1^2}/c_1\right) \tag{8.64}$$

$$f_3(x_1, c_1, z_1) = x_1 c\sqrt{h^2 + (d - x_1)^2} - (d - x_1)c_1\sqrt{z_1^2 + x_1^2} \tag{8.65}$$

Rezolvarea sistemului prin metoda Newton nu necesită decât o singură iteraţie pentru atingerea preciziei cerute. Exemplu de date numerice:

$$t_2 = 10.4 \text{ ns}, T_2 = 11.3 \text{ ns} \Rightarrow z_1 = 36.0 \text{ cm}, c_1 = 12.3 \text{ cm/ns}. \tag{8.66}$$

Figura 8.4: Drumul optic al unei unde prin două medii.

Capitolul 9

Rezolvarea ecuaţiilor diferenţiale

9.1 Consideraţii generale

Ecuaţiile diferenţiale ordinare (sau ecuaţiile diferenţiale care conţin ca necunoscute funcţii care depind de o singură variabilă independentă) se întâlnesc în mod curent în descrierea fenomenelor din natură. Funcţiile necunoscute pot fi viteze, temperaturi, densităţi, sarcini electrice, etc. Variabila independentă este în multe cazuri timpul, când se descriu fenomene evolutive, sau o coordonată spaţială, când se descriu fenomene unidimensionale.

În ecuaţiile diferenţiale apar derivatele de diverse ordine ale funcţiilor necunoscute. Ordinul cel mai mare al derivatei funcţiei necunoscute dă *ordinul ecuaţiei diferenţiale*. Dacă avem un sistem de ecuaţii diferenţiale (adică mai multe ecuaţii care conţin mai multe funcţii necunoscute, în număr egal cu numărul ecuaţiilor diferenţiale), *ordinul sistemului* este egal cu suma ordinelor ecuaţiilor diferenţiale care îl formează.

În general, ecuaţia diferenţială (sau sistemul de ecuaţii diferenţiale) nu determină complet funcţia (sau funcţiile) necunoscută. Pentru a determina complet soluţia, este necesar ca, odată cu ecuaţia (sau sistemul) să se impună un număr de condiţii suplimentare egal cu ordinul ecuaţiei diferenţiale (sau sistemului de ecuaţii diferenţiale).

Deoarece rezolvarea ecuaţiilor de ordin mai mare ca unu se poate reduce la rezolvarea unor sisteme formate numai din ecuaţii diferenţiale de ordinul întâi, vom prezenta la început metodele de rezolvare numerică a acestor ecuaţii mai simple. Trebuie spus că, până la apariţia calculatoarelor electronice, chiar rezolvarea unei singure ecuaţii diferenţiale de ordinul întâi putea ridica dificultăţi mari, de natură practică (volum mare de calcule pentru obţinerea unei precizii acceptabile).

Se consideră o funcţie $y = y(x)$, continuă şi derivabilă pe intervalul de definiţie (sau cel puţin pe intervalul pe care se caută soluţia); fie $[a, b]$ intervalul pe care se caută soluţia. Ecuaţia diferenţială de ordinul întâi se scrie sub *forma implicită*

$$E(x, y, y') = 0, \text{ unde } y' \equiv dy/dx \ , \tag{9.1}$$

$x \in [a, b]$ fiind variabila independentă. Se presupune că expresia $E(x, y, y')$ se poate explicita în raport cu derivata de ordinul întâi y' obţinând *forma explicită*

$$y' = f(x, y) \ , \tag{9.2}$$

funcţia $f(x, y)$ fiind definită şi derivabilă pe un dreptunghi $[a, b] \times [c, d] \subset R^2$. Funcţia $f(x, y)$ se numeşte *funcţie pantă*, deoarece în punctul de coordonate (x, y) valoarea ei este egală cu valoarea derivatei întâi a funcţiei $y(x)$, numeric egală cu panta tangentei la grafic în acest punct. Pentru a determina complet soluţia se dă condiţia suplimentară

$$x = x_0, \ y = y_0; \ x_0 \in [a, b], \ y_0 \in [c, d] \ , \tag{9.3}$$

unde de obicei $x_0 = a$ sau $x_0 = b$.

9.2 Metode cu paşi separaţi

Pentru rezolvarea numerică a ecuaţiei (9.2) cu condiţia (9.3) se împarte intervalul $[a, b]$ în n diviziuni. Punând $x_0 = a$ şi $x_n = b$, se notează cu x_m $(m = 0, 1, \ldots, n)$ nodurile diviziunii, iar cu $h = x_{m+1} - x_m$ pasul de integrare.

O metodă cu *paşi separaţi* (sau metodă *pas cu pas*) este o metodă care determină valoarea funcţiei la pasul $m + 1$ folosind numai valori de la pasul m.

9.2.1 Formule Euler

Cea mai simplă metodă pas cu pas este metoda Euler de ordinul întâi. În acest caz, deplasarea din punctul x_m se face după direcţia tangentei în acest punct la curba $y = y(x)$. Notând cu K_1 valoarea derivatei y'_m, conform cu (9.2) se poate scrie

$$K_1 = y'_m = f(x_m, y_m) \ , \tag{9.4}$$

iar formula Euler de ordinul 1 este

$$\bar{y}_{m+1} = \bar{y}_m + hK_1 \ . \tag{9.5}$$

Plecând de la valoarea y_0 pentru $x = x_0$, aplicând formula (9.5) se obţin valorile $\bar{y}_m$, $m = 1, 2, \ldots, n$ ale funcţiei necunoscute, adică o *funcţie reţea* care aproximează funcţia căutată (barele indică valori aproximative).

Pentru a determina eroarea de trunchiere pentru formula (9.5), se admite că funcţia y poate fi reprezentată printr-o formulă Taylor astfel încât

$$y_{m+1} = y_m + hy'_m + (h^2/2)y''(\xi_m), \text{ unde } \xi_m \in (x_m, x_{m+1}) \ . \tag{9.6}$$

Ţinând seama că ambele formule (9.5) şi (9.6) se aplică plecând din $x = x_m$, rezultă *eroarea de trunchiere* pentru un pas e_{Tm}

$$e_{Tm} = y_{m+1} - \bar{y}_{m+1} = (h^2/2)y''(\xi_m) \ , \tag{9.7}$$

proporţională cu pasul diviziunii la puterea a doua. Pentru întregul interval, în cazul cel mai defavorabil, erorile se adună

$$e_T = \sum_{m=1}^{n} e_{Tm} = \frac{h^2}{2} n y''(\xi') = \frac{b-a}{2} h y''(\xi'), \ \xi' \in (a, b) \tag{9.8}$$

şi deci eroarea este proporţională cu pasul la puterea întâia. De aceea, formula (9.5) este denumită formulă de *ordinul întâi*, indicând mărimea erorii de trunchiere în raport cu pasul diviziunii.

Formule Euler de ordinul 2.

O formulă de ordinul 2 este de forma

$$\bar{y}_{m+1} = \bar{y}_m + h(w_1 K_1 + w_2 K_2) \ , \tag{9.9}$$

unde K_1 este dat de (9.4), iar K_2 este de forma

$$K_2 = f(x + \alpha h, y + \beta h K_1) \ , \tag{9.10}$$

iar parametrii α, β, w_1 şi w_2 se determină din condiţia ca (9.9) să coincidă cu primii trei termeni din dezvoltarea în serie Taylor a funcţiei $y(x)$ în jurul punctului x_m

$$y_{m+1} = y_m + hy'_m + (h^2/2)y''_m + (h^3/6)y'''(\xi_m), \ \xi_m \in (x_m, x_{m+1}) \ , \tag{9.11}$$

unde derivata de ordinul 2, y'' se calculează astfel

$$y'' = (y')' = [f(x, y)]' = f_x + f_y y', \ f_x \equiv \partial f/\partial x, \ f_y \equiv \partial f/\partial y \ . \tag{9.12}$$

Ţinând cont de faptul că $y'_m = f(x_m, y_m) = K_1$, din (9.11) rezultă

$$y_{m+1} = y_m + hK_1 + (h^2/2)(f_{xm} + f_{ym}K_1) + (h^3/6)y'''(\xi_m) \ , \tag{9.13}$$

unde $f_{xm} \equiv \partial f(x_m, y_m)/\partial x$. Pe de altă parte, utilizând o formulă Taylor pentru K_2 (ca funcţie de două variabile), din (9.10) se obţine

$$K_2 = K_1 + h(\alpha f_{xm} + \beta K_1 f_{ym}) + O(h^2) \ . \tag{9.14}$$

Deoarece K_2 este derivata întâi a funcţiei y, vom păstra numai termenii de ordinul h. Introducând (9.14) în (9.9), se obţine

$$\bar{y}_{m+1} = \bar{y}_m + h(w_1 + w_2)K_1 + h^2 w_2(\alpha f_{xm} + \beta K_1 f_{ym}) + O(h^3) \ . \tag{9.15}$$

Identificând relaţia (9.15) cu primii trei termeni din (9.13), rezultă sistemul

$$w_1 + w_2 = 1, \ w_2\alpha = 1/2, \ w_2\beta = 1/2 \ . \tag{9.16}$$

Sistemul (9.16) are patru necunoscute şi numai trei ecuaţii. Teoretic există o infinitate de posibilităţi de a construi o metodă de ordinul 2, două variante fiind mai des întâlnite. Prima dintre acestea se obţine alegând $w_2 = 1/2$. Rezultă $\alpha = \beta = 1$, $w_1 = 1/2$, obţinându-se formula *Euler îmbunătăţită*

$$\bar{y}_{m+1} = \bar{y}_m + \tfrac{h}{2}(K_1 + K_2), \ K_2 = f(x_m + h, \bar{y}_m + hK_1) \ . \tag{9.17}$$

Cea dea doua se obţine alegând $w_2 = 1$. Rezultă $\alpha = \beta = 1/2$, $w_1 = 0$, obţinându-se formula *Euler modificată*

$$\bar{y}_{m+1} = \bar{y}_m + hK_2, \ K_2 = f(x_m + h/2, \bar{y}_m + K_1 h/2) \ . \tag{9.18}$$

Pentru ambele variante K_1 se calculează cu relaţia (9.4). Eroarea de trunchiere la pasul m este $e_{Tm} = y_{m+1} - \bar{y}_{m+1} = O(h^3)$. În consecinţă, formulele (9.17) şi (9.18) vor da o eroare de ordinul h^2 relativ la întregul interval.

Formulele Euler admit o interpretare geometrică, prezentată în figura (9.1). În punctul $A(x_m, \bar{y}_m)$ se construieşte tangenta la graficul funcţiei $y(x)$ de pantă K_1. Punctul B se află pe această tangentă, $y_B = \bar{y}_m + hK_1$ reprezentând valoarea aproximativă pentru $y(x_{m+1})$ obţinută cu metoda Euler de ordinul întâi. În punctul $B(x_{m+1}, y_B)$, se evaluează prima derivată a funcţiei y, $\bar{y}'_B = f(x_{m+1}, y_B) \equiv K_2$. Bara de deasupra semnifică faptul că este vorba de o valoare aproximativă, valoarea exactă s-ar obţine în punctul E. Valoarea $\bar{y}_{m+1}$ se obţine cu metoda *Euler îmbunătăţită* aproximând funcţia pe intervalul $[x_m, x_{m+1}]$ cu bisectoarea unghiului format de cele două pante, K_1 şi K_2. Pe figură, $\bar{y}_{m+1} = y_C$.

Pentru metoda *Euler modificată*, în punctul $A(x_m, \bar{y}_m)$ se construieşte tangenta la graficul funcţiei $y(x)$ de pantă K_1. Punctul $D(x_{m+1} + h/2, y_D)$ se află pe această tangentă, $y_D = \bar{y}_m + (h/2)K_1$. În punctul D, se evaluează prima derivată a funcţiei y: $\bar{y}'_D = f(x_{m+1} + h/2, y_D) \equiv K_2$. Bara de deasupra semnifică faptul că este vorba de o valoare aproximativă, valoarea exactă s-ar obţine în punctul F. Valoarea $\bar{y}_{m+1}$ se obţine cu metoda *Euler modificată* aproximând funcţia pe intervalul $[x_m, x_{m+1}]$ cu dreapta de pantă K_2. Pe figură, $\bar{y}_{m+1} = y_C$.

Figura 9.1: Interpretarea geometrică a metodelor Euler de ordinul 2.

9.2.2 Formule Runge-Kutta

Din prezentarea anterioară se observă că, luând în calcul mai multe puncte (deci folosind mai multe evaluări ale funcţiei pantă $f(x, y)$), se obţin formule de ordin de precizie mai ridicat. De aici apare ideea de a utiliza o expresie generală de forma

$$y_{m+1} = y_m + h \sum_{i=1}^{s} w_i K_i \ , \tag{9.19}$$

$$K_i = f(x_m + \alpha_i h, y_m + h \sum_{j=1}^{i-1} \beta_{ij} K_j), \ i \in \overline{1,s}, \ j \in \overline{1,i-1}, \ \alpha_1 = \beta_{10} = 0 \ , \tag{9.20}$$

unde s este *ordinul formulei*, iar scalarii w_i, α_i, β_{ij} se determină din condiţia ca expresia (9.19) să coincidă cu formula Taylor până la termenii în h^s inclusiv. S-a renunţat a se mai pune bara pentru a indica valorile aproximative. Pentru $s = 4$, relaţiile (9.20) se scriu

$$\begin{aligned} K_1 = f(x_m, y_m), \ K_2 = f(x_m + \alpha_2 h, y_m + \beta_{21} h K_1) \ , \\ K_3 = f[x_m + \alpha_3 h, y_m + h(\beta_{32} K_2 + \beta_{31} K_1)] \ , \\ K_4 = f[x_m + \alpha_4 h, y_m + h(\beta_{43} K_3 + \beta_{42} K_2 + \beta_{41} K_1)] \ . \end{aligned} \tag{9.21}$$

Pentru $s = 2$ se regăsesc formulele Euler de ordinul 2. Numărul de parametri este mai mare decât numărul de condiţii pentru ca aceştia să fie unic determinaţi începând $s \geq 2$. În consecinţă, pentru acelaşi ordin $s \geq 2$, se obţin diverse formule, unii dintre parametri fiind aleşi din motive de simplitate sau de simetrie a formulelor. Formulele astfel obţinute se numesc *formule de tip Runge-Kutta*. Una dintre cele mai utilizate este formula *Runge-Kutta standard* de ordinul 4,

care utilizează derivate calculate la capete şi la jumătatea pasului. Aceasta are forma

$$y_{m+1} = y_m + (h/6)(K_1 + 2K_2 + 2K_3 + K_4) \ , \tag{9.22}$$

$$\begin{array}{c} K_1 = f(x_m, y_m), \ K_2 = f(x_m + \frac{h}{2}, y_m + \frac{h}{2}K_1) \\ K_3 = f(x_m + \frac{h}{2}, y_m + \frac{h}{2}K_2), \ K_4 = f(x_m + h, y_m + hK_3) \ . \end{array} \tag{9.23}$$

9.2.3 Formule Runge-Kutta-Gill

O altă formulă Runge-Kutta de ordin 4, ceva mai complicată, dar care prezintă interes din punctul de vedere al controlului propagării erorilor, este dată de următorul algoritm:

$$\begin{array}{l} \text{Date: } x_0,\ x_n,\ Y^0 \\ x = x_0,\ y = Y^0,\ Q_0 = 0,\ h = (x_n - x_0)/n \\ m = 0 : (n-1), \\ \left[\begin{array}{l} K_1 \leftarrow f(x_m, Y^m),\ Y_1 \leftarrow Y^m + \frac{h}{2}(K_1 - 2Q_0), \\ \quad Q_1 \leftarrow Q_0 + 3[\frac{1}{2}(K_1 - 2Q_0)] - \frac{1}{2}K_1, \\ K_2 \leftarrow f(x_m + \frac{h}{2}, Y_1),\ Y_2 \leftarrow Y_1 + h(1 - 1/\sqrt{2})(K_2 - Q_1), \\ \quad Q_2 \leftarrow Q_1 + 3[(1 - 1/\sqrt{2})(K_2 - Q_1)] - (1 - 1/\sqrt{2})K_2, \\ K_3 \leftarrow f[x_m + \frac{h}{2}, Y_2],\ Y_3 \leftarrow Y_2 + h(1 + 1/\sqrt{2})(K_3 - Q_2) \\ \quad Q_3 \leftarrow Q_2 + 3[(1 + 1/\sqrt{2})(K_3 - Q_2)] - (1 + 1/\sqrt{2})K_3, \\ K_4 \leftarrow f(x_m + h, Y_3),\ y_{m+1} \leftarrow Y_3 + \frac{h}{6}(K_4 - 2Q_3), \\ \quad Q_4 \leftarrow Q_3 + 3[\frac{1}{6}(K_4 - 2Q_3)] - \frac{1}{2}K_4 \\ Q_0 \leftarrow Q_4 \ . \end{array} \right. \end{array} \tag{9.24}$$

Prezenţa radicalilor în expresiile (9.24) este utilă deoarece aceştia introduc întotdeauna o eroare de rotunjire a cărei evoluţie se poate controla la fiecare pas. Această observaţie stă la baza metodelor de tip *Runge-Kutta-Gill*. În metodele Runge-Kutta-Gill de ordinul 4, erorile de rotunjire se iau în consideraţie la fiecare pas prin patru componente Q_1, Q_2, Q_3, Q_4. Se atrage atenţia că relaţiile (9.24) nu trebuie modificate (nu se fac simplificări, reduceri, etc.), ci se păstrează în forma în care sunt scrise, tocmai pentru a putea urmări propagarea erorilor de rotunjire. Valorile parametrului Q_4 dau o indicaţie asupra tendinţei (creştere, staţionare, descreştere) a acestor erori de rotunjire. Astfel, dacă $\|Q_4\|$ depăşeşte o anumită margine impusă[1], se reduce pasul h. De asemenea, pasul h poate fi mărit dacă se apreciază că valoarea $\|Q_4\|$ este prea mică[2].

[1] Algoritmul este prezentat în forma generală pentru rezolvarea sistemelor de ecuaţii diferenţiale, care vor fi prezentate mai jos. În acest caz, Q şi K sunt vectori.

[2] Trebuie avut în vedere faptul că metoda Runge-Kutta-Gill este superioară metodei Runge-Kutta standard din punct de vedere al propagării erorilor de rotunjire *numai dacă toate calculele se efectuează cu acelaşi număr de cifre semnificative*. Spre exemplu, lucrând cu un calculator personal, nu vom observa diferenţe semnificative între cele două metode, deoarece calculele intermediare se efectuează pe 80 de biţi, indiferent de numărul de biţi folosiţi de compilator pentru reprezentarea unui număr real.

9.2.4 Alegerea pasului la rezolvarea ecuaţiei diferenţiale

Estimarea erorii la formulele pas cu pas se poate face repetând calculul cu pas modificat. Spre exemplu, pentru a verifica dacă pasul h este suficient de mic pentru asigurarea unei precizii corespunzătoare, se avansează doi paşi din punctul x_m cu pasul h, apoi se reface calculul cu pasul dublu $2h$, după schema

$$x_m + h + h = x_m + 2h \ . \tag{9.25}$$

În aceste condiţii, neglijând erorile de rotunjire pe distanţa h, utilizând o formulă Taylor, se poate scrie

$$\begin{aligned} y(x+2h) &= y_{2h} + (2h)^{s+1} y^{(s+1)}_{(\xi_1)}/(s+1)! \ , \\ y(x+2h) &= y_h + h^{s+1}\left(y^{(s+1)}_{(\xi_{21})} + y^{(s+1)}_{(\xi_{22})}\right)/(s+1)! \\ &= y_h + 2h^{s+1} y^{(s+1)}_{(\xi_2)}/(s+1)! \ , \end{aligned} \tag{9.26}$$

unde $\xi_1, \xi_2 \in (x, x+2h)$, $\xi_{21} \in (x, x+h)$, $\xi_{22} \in (x+h, x+2h)$. Pentru $s = 4$ avem

$$\begin{aligned} y_{2h} &= y(x) + 2hy'_{(x)} + \tfrac{(2h)^2}{2} y''_{(x)} + \tfrac{(2h)^3}{6} y'''_{(x)} + \tfrac{(2h)^4}{24} y^{(4)}_{(x)} \\ y_h &= y(x) + hy'_{(x)} + \tfrac{h^2}{2} y''_{(x)} + \tfrac{h^3}{6} y'''_{(x)} + \tfrac{h^4}{24} y^{(4)}_{(x)} + \\ &\quad + hy'_{(x+h)} + \tfrac{h^2}{2} y''_{(x+h)} + \tfrac{h^3}{6} y'''_{(x+h)} + \tfrac{h^4}{24} y^{(4)}_{(x+h)} \ . \end{aligned}$$

Considerând derivatele aproximativ egale

$$y^{(s+1)}(\xi_1) \cong y^{(s+1)}(\xi_2) \equiv y^{(s+1)}(\xi), \ \xi \in (x, x+2h) \ , \tag{9.27}$$

din (9.26) se obţine

$$y_h - y_{2h} = 2h^{s+1}\frac{2^s - 1}{(s+1)!} y^{(s+1)}(\xi) \Rightarrow \frac{2h^{s+1}}{(s+1)!} y^{(s+1)}(\xi) = \frac{y_h - y_{2h}}{2^s - 1} \ . \tag{9.28}$$

Am notat y_h, y_{2h} valorile aproximative ale funcţiei y în $x+2h$ obţinute cu paşii de integrare h respectiv $2h$. Putem corecta valoarea y_h, presupusă mai exactă decât y_{2h}, cu relaţia

$$y(x+2h) \cong y_h + e, \ \ e \equiv (y_h - y_{2h})/(2^s - 1) \ . \tag{9.29}$$

Am obţinut de asemenea o posibilitate de a evalua eroarea absolută prin termenul e. Se poate deci impune micşorarea în continuare a pasului, dacă e depăşeşte o anumită margine superioară sau, dimpotrivă, mărirea pasului în cazul în care e are o valoare prea mică.

Să presupunem că dorim ca eroarea absolută să nu depăşească valoarea impusă e_0. În general, eroarea unei metode de ordinul s este de forma

$$e = h^{s+1} y^{(s+1)}_{(\xi)}/(s+1)! \ . \tag{9.30}$$

Se obţine

$$\left|\frac{e_0}{e}\right| = \frac{h_0^{s+1}}{h^{s+1}}\left|\frac{y_{(\xi_0)}^{(s+1)}}{y_{(\xi)}^{(s+1)}}\right| \Rightarrow h_0 = S\cdot h\cdot\left(\frac{e_0}{|e|}\right)^{\frac{1}{s+1}} , \tag{9.31}$$

unde $S = |y_{(\xi_0)}^{(s+1)}/y_{(\xi)}^{(s+1)}|^{\frac{1}{s+1}}$ este, pentru h suficient de mic, de ordinul unităţii (rezultate bune în majoritatea cazurilor pot fi obţinute cu $S = 0.9$). Dacă eroarea obţinută cu pasul h este mai mare decât cea impusă ($e > e_0$), atunci calculul trebuie reluat cu un pas mai mic, h_0 fiind o estimare bună a mărimii acestui pas. Dacă în schimb eroarea obţinută este mult mai mică decât cea impusă ($e \ll e_0$), înseamnă că pasul de integrare este mai mic decât cel necesar, iar relaţia (9.31) poate fi folosită pentru estimarea pasului următor.

De regulă, în locul erorii absolute se foloseşte eroarea relativă definită ca $\varepsilon = |e/y_{ref}|$, unde y_{ref} este o valoare de referinţă a necunoscutei y (spre exemplu ultima valoare calculată, $y_{ref} = y_{m+1} = y_h$). În acest caz se poate scrie

$$h_0 = S\cdot h\cdot(\varepsilon_0/\varepsilon)^{\frac{1}{s+1}} . \tag{9.32}$$

Metoda prezentată este fiabilă, dar conduce la creşterea timpului de calcul, ceea ce nu se justifică întotdeauna. Pentru obţinerea unei estimări a erorii absolute e cu pasul h, efectuăm $s+s+s = 3s$ evaluări ale funcţiei pantă $f(x,y)$, dintre care $K_1 = f(x_m, y_m)$ este aceeaşi pentru integrarea cu h şi $2h$. Pentru Runge-Kutta de ordinul 4, sunt necesare 11 evaluări ale funcţiei pantă pentru doi paşi, deci 5.5 evaluări pentru un pas. Rezultă un efort de calcul suplimentar de 1.5 evaluări ale funcţiei pantă pentru a obţine o estimare a erorii absolute şi a pasului optim de integrare. În cazul în care pasul h este relativ mare, factorul de siguranţă S poate diferi mult de unitate. În această situaţie, o estimare a erorii se face comparând cele două valori obţinute

$$e \cong y_h - y_{2h},\ \varepsilon \cong (y_h - y_{2h})/y_h . \tag{9.33}$$

O modalitate mai simplă pentru a verifica mărimea pasului de integrare la metoda Runge-Kutta standard, este evaluarea raportului

$$r = |(K_2 - K_3)/(K_1 - K_2)| . \tag{9.34}$$

Observând că pantele K_2, K_3 se evaluează în acelaşi punct $x = x_m + h/2$ iar K_1, K_2 se evaluează în puncte diferite, pe considerente empirice, se recomandă menţinerea pasului dacă raportul r este de ordinul 10^{-2}. În cazul unor ordine de mărime diferite de 10^{-2}, pasul se modifică (spre exemplu, se înjumătăţeşte dacă $r \sim 10^{-1}$).

O altă cale de a obţine informaţii privitoare la precizia calculelor şi a pasului necesar este de a folosi două formule Runge-Kutta de ordine s şi p, de regulă $p = s+1$. Aceasta înseamnă $s+p = 2s+1$ evaluări în total pentru a obţine o valoare aproximativă cu o eroare de ordinul h^s precum şi o estimare a acestei erori şi a pasului optim de integrare. La prima vedere, preţul plătit pare prea

Tabelul 9.1: Valorile coeficienţilor pentru Kutta-Merson

i	a_i	$a_i - b_i$	α_i	β_{i1}	β_{i2}	β_{i3}	β_{i4}
1	1/6	1/15	0				
2	0	0	1/3	1/3			
3	0	−3/10	1/3	1/6	1/6		
4	2/3	4/15	1/2	1/8	0	3/8	
5	1/6	−1/30	1	1/2	0	−3/2	2

mare. În realitate nu este aşa deoarece ambele formule folosesc *aceleaşi* puncte pentru evaluarea funcţiei pantă, adică numai p *evaluări*! Acest lucru este posibil, deoarece raportul dintre numărul de evaluări ale funcţiei pantă şi ordinul metodei creşte cu ordinul metodei după cum urmează:

Număr de evaluări	1	2	3	4	5	6	7	8
Ordinul metodei Runge-Kutta	1	2	3	4	4	5	6	6 .

Formulele din categoria de mai sus se pot scrie în forma generală

$$\begin{gathered} y_{m+1} \equiv y_p = y_m + h\sum_{i=1}^{p} a_i K_i,\ y_s = y_m + h\sum_{i=1}^{p} b_i K_i\ , \\ K_i = f(x+\alpha_i h, y_m + h\sum_{j=1}^{i-1}\beta_{ij}K_j)\ . \end{gathered} \tag{9.35}$$

O estimare a erorii de trunchiere pentru y_s poate fi obţinută cu relaţia

$$e = y_p - y_s = h\sum_{i=1}^{p}(a_i - b_i)K_i\ , \tag{9.36}$$

iar pasul de integrare necesar obţinerii unei precizii impuse se calculează cu relaţia

$$h_0 = S\cdot h\cdot(e_0/|e|)^{1/p}\ . \tag{9.37}$$

Din această categorie, prezentăm metodele *Kutta-Merson* şi *Runge-Kutta-Fehlberg*.

Metoda Kutta-Merson foloseşte două formule Runge-Kutta de ordine 4, respectiv 5. Coeficienţii din formulele (9.35) şi (9.36) sunt daţi în tabelul (9.1). Pentru obţinerea relaţiei (9.37) este nevoie de numai 5 evaluări ale funcţiei pantă, faţă de 5.5 necesare în varianta integrării cu pasul dublu cu eroarea absolută dată de (9.33). Varianta pasului dublu este însă mai robustă.

S-a construit o metodă de ordinul 4 împreună cu o estimare a erorii cu numai 5 evaluări ale funcţiei pantă.

Metoda Runge-Kutta-Fehlberg foloseşte o formulă de ordinul 4 cu 5 evaluări ale funcţiei pantă şi o formulă de ordinul 5 cu 6 evaluări. Aceasta înseamnă 6 evaluări în total pentru a obţine o valoare aproximativă cu o eroare de ordinul h^5 precum şi o estimare a acestei erori şi a pasului optim de integrare. Valorile coeficienţilor a_i, b_i, α_i, β_{ij} sunt daţi în tabelul 9.2.

Tabelul 9.2: Valorile coeficienţilor pentru Runge-Kutta-Fehlberg

i	a_i	$a_i - b_i$	α_i	β_{i1}	β_{i2}	β_{i3}	β_{i4}	β_{i5}
1	16/135	1/360	0	0				
2	0	0	1/4	1/4				
3	6656/12825	−128/4275	3/8	3/32	9/32			
4	28561/56430	−2197/75240	12/13	1932/2197	−7200/2197	7296/2197		
5	−9/50	1/50	1	439/216	−8	3680/513	−845/4104	
6	2/55	2/55	1/2	− 8/27	2	3544/2565	1859/4104	−11/40

9.3 Extrapolare Richardson. Metoda Bulirsch-Stoer

Tehnica extrapolării de tip Richardson constă în construirea unei funcţii $y_{m+1}(h)$ prin calcularea mai multor valori y_{m+1} folosind diferiţi paşi $h = x_{m+1} - x_m$, aproximărea acesteia cu o funcţie mai simplă (spre exemplu polinom) şi calcularea unei valori aproximative y^{ex}_{m+1} corespunzătoare unei valori h mai mici, în particular pentru $h = 0$.

Fie $y_m \equiv y(x_m)$ valori calculate până la un moment dat şi $x_{m+1} = x_m + H$ punctul în care dorim să calculăm o valoarea aproximativă y_{m+1}. Intervalul $(x, x + H)$ cu $x \equiv x_m$ se împarte în n subintervale egale, $x_{i+1} - x_i \equiv h = H/n$, $i \in \overline{1, n}$. Pentru diferite valori n, deci paşi de integrare diferiţi, se obţin mai multe valori aproximative ale necunoscutei y în punctul x_{m+1}. Se obţin astfel perechi de valori $(h, y_{m+1}(h))$, cu ajutorul cărora se poate calcula o valoare aproximativă $y_{m+1}(0)$. De asemenea, având mai multe valori y_{m+1} pentru paşi de integrare diferiţi, se poate estima eroarea $e \equiv y - y_{m+1}$. Dacă aceasta este prea mare, calculul se repetă pentru un n mai mare (respectiv, un pas mai mic). Extrapolarea Richardson dă rezultate foarte bune pentru ecuaţii diferenţiale a căror soluţii sunt funcţii monotone (sau fără variaţii mari ale primei derivate) şi nu au puncte singulare în intervalul de interes.

Pentru obţinerea unui algoritm performant, trebuie precizate următoarele: (a) metoda de integrare pe intervalul $(x, x + H)$; (b) metoda de aproximare a funcţiei $y_{m+1}(h)$.

Metoda Bulirsch-Stoer, foloseşte următoarea strategie de mărire a numărului de intervale n:

$$n_j = 2n_{j-2} \Rightarrow n = 2, 4, 6, 8, 12, 16, \dots \ , \tag{9.38}$$

care evită scăderea prea rapidă a pasului de integrare în cazul uzual $n_j = 2n_{j-1}$. Pentru integrarea numerică a ecuaţiei diferenţiale se foloseşte o metodă

de ordinul 2, definită de următorul algoritm:

$$\begin{array}{l} z_0 \equiv y(x_m);\ z_1 = z_0 + h \cdot f(x_m, y_0) \\ i = 1 : (n-1) \\ \quad [z_{i+1} = z_{i-1} + 2hf(x_m + ih, y_i) \\ y_{m+1} = \frac{1}{2}[z_n + z_{n-1} + hf(x + H, z_n)] \ . \end{array} \tag{9.39}$$

Unul din avantajele formulelor (9.39) constă în faptul că avem nevoie de numai o evaluare a funcţiei pantă. Avantajul esenţial îl constituie însă faptul că dezvoltarea în serie Taylor a erorii conţine numai termenii cu puteri *pare*

$$e \equiv y_{m+1} - y(x + H) = \sum\nolimits_{i=1}^{\infty} a_i h^{2i} \ . \tag{9.40}$$

Astfel, dacă avem două valori y_n şi $y_{n/2}$ obţinute cu n respectiv $n/2$ subintervale, obţinem o valoare îmbunătăţită

$$y(x + H) \approx (1/3)(4y_n - y_{n/2}) \tag{9.41}$$

de *ordinul patru* şi nu trei cum ar fi fost normal pentru o formulă de ordinul doi.

Pentru extrapolare la valoarea $h = 0$, Burlisch şi Stoer au găsit mai eficientă aproximarea cu funcţii raţionale în locul interpolării polinomiale.

9.4 Metode cu paşi legaţi

Principalele deficienţe ale metodelor cu paşi separaţi sunt: (a) necesitatea evaluării la fiecare pas a unui număr de funcţii cel puţin egal cu ordinul metodei; (b) evaluarea dificilă a erorii de trunchiere.

Deficienţele semnalate pot fi înlăturate folosind metode care utilizează mai bine valorile deja calculate ale funcţiei necunoscute. Aceste metode se numesc *metode cu paşi legaţi* sau *metode multipas*. Ideea este următoarea: să presupunem, revenind la ecuaţia (9.2), că se cunoaşte soluţia $y(x)$. Atunci funcţia pantă $f(x, y(x))$ este o anumită funcţie de x, notată cu $\Phi(x) \equiv f(x, y(x))$. Integrând ecuaţia (9.2) termen cu termen, pe un interval $[x_{m-k}, x_{m+1}]$, $m = 0, 1, 2, \ldots$, se obţine

$$y_{m+1} = y_{m-k} + \int_{x_{m-k}}^{x_{m+1}} \Phi(x)\, dx \tag{9.42}$$

(figura 9.2). În relaţia (9.42), funcţia $\Phi(x)$ este necunoscută, dar poate fi aproximată utilizând spre exemplu un polinom de interpolare.

9.4.1 Formule explicite

Pentru a lua în consideraţie valori deja calculate ale funcţiei $y(x)$, vom utiliza polinomul cu diferenţe la stânga. Introducem numărul real α prin relaţia

$$x = x_m + \alpha h,\ \alpha \leq 1 \ . \tag{9.43}$$

Figura 9.2: Funcţia $\Phi(x)$.

Funcţia $\Phi(x)$ se va reprezenta în forma

$$\Phi(x) \equiv \Phi(x_m + \alpha h) = \Phi_m + \alpha \nabla \Phi_m + \ldots + (-1)^r C_{-\alpha}^r \nabla^r \Phi_m + R_r \ , \quad (9.44)$$

r fiind gradul polinomului de interpolare, ∇ operatorul diferenţă la stânga, iar R_r restul la interpolare

$$R_r = (-1)^{r+1} h^{r+1} C_{-\alpha}^{r+1} \Phi^{(r+1)}(\xi), \ \xi \in (x_{m-r}, x) \ . \quad (9.45)$$

Ţinând seama de (9.44), relaţia (9.42) se scrie

$$y_{m+1} = y_{m-k} + h \int_{-k}^{1} \left(\Phi_m + \alpha \nabla \Phi_m + \ldots + (-1)^r C_{-\alpha}^r \nabla^r \Phi_m \right) d\alpha + \delta_m, \quad (9.46)$$

δ_m fiind eroarea la integrare la pasul m

$$\delta_m = (-1)^{r+1} h^{r+2} \int_{-k}^{1} C_{-\alpha}^{r+1} \Phi^{r+1}(\xi) \, d\alpha \ . \quad (9.47)$$

Constatăm că avem doi parametri la dispoziţie: r, gradul polinomului de interpolare, grad care dă *ordinul de precizie al formulei* (egal cu $r+1$) şi k, parametrul care dă intervalul de integrare. Existenţa acestor doi parametri permit obţinerea de formule diferite, la acelaşi ordin de precizie. În cele ce urmează dăm unele formule mai des folosite obţinute direct din aplicarea formulelor generale (9.46) şi (9.47).

a) Pentru $r = 3$, $k = 0$, se obţine *formula predictor a lui Adams*

$$\begin{gathered} y_{m+1}^{(A)} = y_m + \tfrac{h}{24}(55 f_m - 59 f_{m-1} + 37 f_{m-2} - 9 f_{m-3}) \ , \\ \delta_m = \tfrac{251}{720} h^5 \Phi^{(4)}(\xi) = \tfrac{251}{720} h^5 \Phi_m^{(4)} + O(h^6) \ . \end{gathered} \quad (9.48)$$

b) Pentru $r = 3$, $k = 3$, se obţine *formula predictor a lui Milne*

$$\begin{aligned} y_{m+1}^{(M)} &= y_{m-3} + \tfrac{4h}{3}(2f_m - f_{m-1} + 2f_{m-2}) \ , \\ \delta_m &= \tfrac{14}{45}h^5\Phi^{(4)}(\xi) = \tfrac{14}{45}h^5\Phi_m^{(4)} + O(h^6) \ . \end{aligned} \tag{9.49}$$

c) Pentru $r = 5$, $k = 5$, se obţine o *formulă de ordinul 6*

$$\begin{aligned} y_{m+1} &= y_m + \tfrac{3h}{10}(11f_m - 14f_{m-1} + 26f_{m-2} - 14f_{m-3} + 11f_{m-4}) \ , \\ \delta_m &= \tfrac{41}{140}h^7\Phi^{(6)}(\xi) = \tfrac{41}{140}h^7\Phi_m^{(6)} + O(h^8) \ . \end{aligned} \tag{9.50}$$

Observaţii.

1. Formulele prezentate în acest paragraf sunt *explicite* pentru că toţi termenii din dreapta semnului egal sunt cunoscuţi. Formulele explicite cu paşi legaţi se mai numesc *formule predictor*.

2. Formulele explicite multipas necesită evaluarea unei singure funcţii f_m la fiecare pas, indiferent de ordinul de pecizie al formulei.

3. Erorile de trunchiere δ_m au expresii simple care permit evaluarea erorii în mod simplu. Spre exemplu, utilizând formulele *Adams* şi *Milne*, se obţine, cu o aproximaţie de ordinul $O(h^6)$

$$y_{m+1} = y_{m+1}^{(A)} + \tfrac{251}{27}(y_{m+1}^{(M)} - y_{m+1}^{(A)}) + O(h^7) \ . \tag{9.51}$$

4. Formulele multipas nu pot lua startul singure; pentru start se utilizează o formulă pas cu pas, în general de acelaşi ordin.

5. Luând în considerare şi evaluarea erorii, economia de calcul pentru metodele cu paşi legaţi este încă şi mai mare, în comparaţie cu metodele cu paşi separaţi.

9.4.2 Formule implicite

Deoarece polinomul de interpolare obţinut din (9.44) se extrapolează în x_{m+1}, erorile δ_m ale formulelor multipas explicite sunt relativ mari. Pentru reducerea acestor erori, se folosesc *formule implicite*. Formulele implicite cu paşi legaţi se mai numesc *formule corector*. După cum vom vedea, formulele corector nu pot fi folosite singure, ele având nevoie de o valoare aproximativă iniţială pe care o îmbunătăţesc iterativ. Această valoare aproximativă este furnizată de o formulă predictor, ansamblul celor două formule formând o *formulă predictor-corector*.

În scopul evitării extrapolării, la construirea polinomului de interpolare se ia în consideraţie şi valoarea y_{m+1}. Se redefineşte variabila α (9.43) prin

$$x = x_{m+1} + \alpha h, \ \alpha \leq 0 \tag{9.52}$$

iar funcţia $\Phi(x)$ se reprezintă sub forma similară cu (9.44)

$$\Phi(x) \equiv \Phi(x_{m+1}+\alpha h) = \Phi_{m+1} + \alpha\nabla\Phi_{m+1} + \ldots + (-1)^r C^r_{-\alpha}\nabla^r\Phi_{m+1} + R_r \ , \tag{9.53}$$

restul fiind dat de relaţia (9.45). Relaţiile (9.42) şi (9.53) ne conduc la

$$\begin{gathered} y_{m+1} = y_{m-k} + h\int_{-k-1}^{0}(\Phi_{m+1} + \alpha\nabla\Phi_{m+1} + \ldots + (-1)^r C^r_{-\alpha}\nabla^r\Phi_{m+1})d\alpha + \delta_m \ , \\ \delta_m = (-1)^{r+1}h^{r+2}\int_{-k-1}^{0} C^{r+1}_{-\alpha}\Phi^{r+1}(\xi)d\alpha \ . \end{gathered} \tag{9.54}$$

De această dată, expresiile pentru y_{m+1} sunt implicite deoarece Φ_{m+1} depinde de valoarea necunoscută y_{m+1}, $\Phi_{m+1} = f(x_{m+1}, y_{m+1})$. Pentru a vedea modul de lucru efectiv, se consideră mai întâi câteva formule uzuale

a) *Formula corector a lui Adams* se obţine pentru $r = 3, k = 0$

$$\begin{gathered} y^{(A)}_{m+1} = y_m + \tfrac{h}{24}(9f_{m+1} + 19f_m - 5f_{m-1} + f_{m-2}) \ , \\ \delta_m = -\tfrac{19}{720}h^5\Phi^{(4)}(\xi) = -\tfrac{19}{720}h^5\Phi^{(4)}_m + O(h^6) \ . \end{gathered} \tag{9.55}$$

b) *Formula corector a lui Milne* se obţine pentru $r = 3, k = 1$

$$\begin{gathered} y^{(M)}_{m+1} = y_{m-1} + \tfrac{h}{3}(f_{m+1} + 4f_m + f_{m-1}) \ , \\ \delta_m = -\tfrac{1}{90}h^5\Phi^{(4)}(\xi) = -\tfrac{1}{90}h^5\Phi^{(4)}_m + O(h^6) \ . \end{gathered} \tag{9.56}$$

c) Pentru $r = 5, k = 3$, se obţine următoarea *formulă de ordinul 6*:

$$\begin{gathered} y_{m+1} = y_{m-3} + \tfrac{2h}{45}(7f_{m+1} + 32f_m + 12f_{m-1} + 32f_{m-2} + 7f_{m-3}) \ , \\ \delta_m = -\tfrac{8}{945}h^7\Phi^{(6)}(\xi) = -\tfrac{8}{945}h^7\Phi^{(4)}_m + O(h^8) \ . \end{gathered} \tag{9.57}$$

Se constată că toate formulele implicite au structura

$$y_{m+1} = a + b\,h\,f(x_{m+1}, y_{m+1}) \ , \tag{9.58}$$

unde a şi b sunt constante. Relaţia (9.58) reprezintă o ecuaţie în necunoscuta y_{m+1} care poate fi rezolvată numeric iterativ. Condiţia suficientă ca şirul de iteraţii

$$y^{(k+1)}_{m+1} = a + b\,h\,f(x_{m+1}, y^{(k)}_{m+1}) \tag{9.59}$$

să fie convergent este (vezi 3.2.1)

$$|b \cdot h \cdot \partial f/\partial y| < 1 \tag{9.60}$$

şi poate fi satisfăcută întotdeauna pentru h suficient de mic. Problema este ca precizia dorită să se atingă astfel încât numărul de evaluări ale funcţiei pantă

$f(x,y)$ să fie mai mic decât ordinul metodei, în caz contrar dispare avantajul volumului de calcul mai redus al formulelor multipas, în raport cu formulele pas cu pas. De aceea, pentru start se alege valoarea $y_{m+1}^{(0)}$ dată de o formulă multipas explicită, de acelaşi ordin, formulă utilizată ca predictor, în timp ce formula implicită (9.58) foloseşte drept corector. O asemenea combinaţie se numeşte *formulă predictor-corector*.

Metoda predictor-corector ne furnizează posibilitatea de evaluare a erorii şi de a face astfel o nouă corecţie. Astfel, utilizând formulele Adams (9.48) şi (9.55), se obţine

$$y_{m+1} = y_{m+1}^{(Ac)} - \tfrac{19}{270}(y_{m+1}^{(Ac)} - y_{m+1}^{(Ap)}) + O(h^6) \ . \tag{9.61}$$

Diferenţa dintre valorile date de cele două metode, ne permite să apreciem oportunitatea schimbării pasului.

La fel, neglijând erorile de rotunjire, utilizând formulele *predictor-corector Milne* de ordinul 4 (9.49) şi (9.56), se obţine

$$y_{m+1} = y_{m+1}^{(Mc)} - \tfrac{1}{29}(y_{m+1}^{(Mc)} - y_{m+1}^{(Ap)}) + O(h^6) \ . \tag{9.62}$$

9.5 Propagarea erorilor. Stabilitate.

După cum s-a văzut în capitolul 4, orice calcul numeric este afectat de *erori de metodă* şi de *erori de rotunjire*. O metodă în care erorile introduse într-o anumită etapă nu sunt amplificate la paşii următori de calcul se numeşte *stabilă*. Trebuie observat însă că această definiţie se aplică numai pentru *probleme corect puse*, adică pentru ecuaţii diferenţiale şi condiţii la limite pentru care soluţiile există şi sunt stabile. O soluţie este *instabilă* în sensul lui *Hadamard* dacă variaţii mici în parametri introduc variaţii mari în soluţie. Evident că în asemenea cazuri orice metodă numerică utilizată va conduce la erori necontrolabile.

Pe de altă parte, erorile absolute pot fi amplificate, dar în acelaşi timp, este posibil ca erorile relative în modul să rămână pe loc sau chiar să scadă, ca urmare a creşterii valorilor soluţiei, în modul. În acest caz, se poate spune că metoda este *relativ stabilă*.

Pentru a scoate în evidenţă mai concret situaţiile care pot apare, vom considera că, la pasul $x = x_m$, ecuaţia de rezolvat are forma

$$y' = f_m + a_m(x - x_m) + b_m(y - y_m) \equiv f(x,y) \ , \tag{9.63}$$

cu condiţia ca pentru $x = x_m$, $y = y_m$, iar a_m, b_m, f_m sunt constante pe intervalul (x_m, x_{m+1}). Vom presupune $b_m \neq 0$, altfel problema se reduce la o simplă integrală obişnuită. Ecuaţia (9.63) are soluţia exactă

$$\begin{gathered} y = y_m - A_m e^{b_m(x-x_m)} + A_m + B_m(x - x_m), \\ A_m = -f_m/b_m - a_m/b_m^2, \ B_m = -a_m/b_m, \ b_m \neq 0 \ . \end{gathered} \tag{9.64}$$

Aşa cum se observă, membrul drept din (9.63) conţine primii 3 termeni din dezvoltarea Taylor a funcţiei de două variabile $f(x,y)$. Dacă $f(x,y)$ este o

funcţie de gradul întâi în variabilele x, y, atunci expresia (9.64) este exactă, dacă valorile y_m sunt exacte. Pentru $x = x_{m+1}$, soluţia (9.64) devine

$$y_{m+1} = y_m - A_m e^{b_m h} + A_m + B_m h \ . \tag{9.65}$$

Să comparăm rezultatul (9.65) cu formulele Runge-Kutta de ordinul 4 şi Milne-predictor de acelaşi ordin. Pentru ecuaţia (9.63), formula Runge-Kutta (9.22) dă

$$\bar{y}_{m+1}^{(RK)} = \bar{y}_m^{(RK)} - \bar{A}_m \sum\nolimits_{n=0}^{4} (b_m h)^n / n! + \bar{A}_m + \bar{B}_m h + e_{Rm} \ , \tag{9.66}$$

e_{Rm} fiind eroarea de rotunjire cu care obţinem numeric valoarea $\bar{y}_{m+1}^{(RK)}$. S-a folosit bara pentru a indica valorile aproximative. Pe de altă parte, s-a avut în vedere că, la pasul $x = x_m$, problema se rezolvă prin metoda Runge-Kutta pentru o condiţie iniţială alterată ($y = \bar{y}_m$).

Eroarea absolută e_{m+1} este, conform definiţiei, diferenţa $y_{m+1} - \bar{y}_{m+1}^{(RK)}$

$$e_{m+1}^{(RK)} = e_m^{(RK)} - (A_m - \bar{A}_m)(e^{b_m h} - 1) - \bar{A}_m \frac{(b_m h)^5}{5!} e^{b_m \xi_m} + (B_m - \bar{B}_m) - e_{Rm} \ . \tag{9.67}$$

S-a folosit formula Taylor pentru a înlocui

$$\sum\nolimits_{n=0}^{4} (b_m h)^n / n! = e^{b_m h} - \tfrac{(b_m h)^5}{5!} e^{b_m \xi_m}, \ \xi \in (x_m, x_{m+1}) \ . \tag{9.68}$$

Prin sumare de la $x = x_0$ la $x = x_{m+1}$, din (9.67) se obţine

$$e_{m+1}^{(RK)} = e_0 - \sum_{i=0}^{m} \left[(A_i - \bar{A}_i)(e^{b_i h} - 1) + \bar{A}_i \frac{(b_i h)^5}{5!} e^{b_i \xi} - (B_i - \bar{B}_i) h + e_{Ri} \right] . \tag{9.69}$$

Erorile iniţiale $e_0^{(RK)}$şi e_{R0} putând fi considerate nule, iar coeficienţii A_i, $\bar{A}_i$ şi B_i, $\bar{B}_i$, egali, relaţia (9.69) se poate scrie

$$e_{m+1}^{(RK)} = \sum_{i=1}^{m} \left[(1 - e^{b_i h})(A_i - \bar{A}_i) - \bar{A}_i \frac{(b_i h)^5}{5!} e^{b_i \xi} + (B_i - \bar{B}_i) h + e_{Ri} \right] \ . \tag{9.70}$$

Pe de altă parte, prin sumare, din (9.65) se obţine

$$y_{m+1} = y_0 + \sum_{i=0}^{m} \left[(1 - e^{b_i h}) A_i + B_i h \right] \ . \tag{9.71}$$

Eroarea relativă ε_{m+1} va fi raportată la valoarea exactă y_m

$$\varepsilon_{m+1}^{(RK)} = e_{m+1}^{(RK)} / y_{m+1} \ . \tag{9.72}$$

Din relaţiile (9.70) şi (9.71) se deduce că

1. Dacă $b_m h \in (0,1)$, atunci $e^{b_m} > 1$, unde $b_m = (\partial f/\partial x)_m$, eroarea absolută $e_{m+1}^{(RK)}$ creşte în modul cu numărul m de paşi parcurşi. Totodată creşte valoarea y_{m+1} dacă $A_m \neq 0$. Prin urmare, eroarea relativă $\varepsilon_{m+1}^{(RK)}$ poate fi mică, iar soluţia *relativ stabilă*. Cazul critic se produce pentru

$$A_m = 0, y''_m = a_m + b_m f_m = 0 \tag{9.73}$$

în care caz termenul exponenţial dispare din soluţia exactă. Această situaţie se poate produce pentru anumite condiţii la limite. În fapt, anularea derivatei y'' în fiecare punct conduce la o funcţie de gradul întâi. Ca urmare, atât eroarea absolută $e_{m+1}^{(RK)}$ cât şi cea relativă, $\varepsilon_{m+1}^{(RK)}$ pot creşte oricât de mult, *indiferent de metoda numerică folosită*. De aceea se spune că soluţia, pentru condiţia la limită care satisface (9.73), este *inerent instabilă numeric*.

2. Dacă $b_m h \in (-1,0)$, atunci $e^{b_m h} < 1$, iar soluţia poate deveni instabilă odată cu scăderea în modul a valorilor soluţiei.

În continuare, vom arăta particularităţile *propagării erorilor* pentru formulele multipas. Pentru a aplica formula *Milne* (9.56) ecuaţiei (9.63), vom scrie

$$\bar{f}_{m+1} = \bar{f}_m + \bar{a}_m h + \bar{b}_m(\bar{y}_{m+1}^{(M)} - \bar{y}_m^{(M)}) \tag{9.74}$$

de unde, prin sumare, se obţine

$$\bar{f}_{m+1} = \bar{f}_0 + \sum_{i=0}^{m} \left[\bar{a}_i h + \bar{b}_i(\bar{y}_{i+1}^{(M)} - \bar{y}_i^{(M)})\right], \ i \in \overline{0,m}\ , \tag{9.75}$$

bara indicând că valorile se calculează cu erori de trunchiere şi rotunjire. Pentru a_i =constant= a, b_i =constant= b, din (9.75) se obţine

$$\bar{f}_{m+1} = \bar{f}_0 + \bar{a}h(m+1) + \bar{b}(\bar{y}_{m+1}^{(M)} - y_0),\ m = 0,1,2,\dots\ . \tag{9.76}$$

Ţinând seama de (9.76), din formula Milne (9.56) se obţine următoarea ecuaţie în diferenţe pentru $y_m^{(M)}$

$$\begin{aligned} & (1 - \bar{b}h/3)\bar{y}_{m+1}^{(M)} - 4\bar{b}h/3\bar{y}_m^{(M)} - (1 + \bar{b}h/3)\bar{y}_{m+1}^{(M)} \\ = \ & 2h(f_0 - \bar{b}y_0) + 2a\,h^2 m\ . \end{aligned} \tag{9.77}$$

Ecuaţia în diferenţe (9.77) se rezolvă similar cu ecuaţiile diferenţiale cu coeficienţi constanţi, neomogene. Astfel, ecuaţia omogenă (fără membrul drept) admite soluţii de forma $\bar{y}_m = r^m$, numerele r fiind date de soluţiile ecuaţiei caracteristice

$$(1 - \bar{b}h/3)r^2 - (4\bar{b}h/3)r - (1 + \bar{b}h/3) = 0\ . \tag{9.78}$$

Se obţin soluţiile

$$\begin{aligned} r_1 &= (2\bar{b}h + \sqrt{9 + 3\bar{b}^2h^2})/(3 - \bar{b}h) \\ r_2 &= -(1/r_1)(3 + \bar{b}h)/(3 - \bar{b}h) = (2\bar{b}h - \sqrt{9 + 3\bar{b}^2h^2})/(3 - \bar{b}h)\ . \end{aligned} \tag{9.79}$$

Soluţia generală a ecuaţiei (9.77) se obţine adunând la soluţia ecuaţiei omogene o soluţie particulară. Astfel, introducând şi o eroare de rotunjire, e_{Rm}, rezultă

$$\bar{y}_m^{(M)} = C_1 r_{1m} + C_1 r_{2m} + y_0 - (\bar{a} + \bar{b} f_0)/\bar{b}^2 - m(\bar{a}h)/\bar{b} + e_{Rm} \ . \tag{9.80}$$

Pentru determinarea constantelor C_1 şi C_2, se pun condiţiile

$$\begin{cases} x = x_0, \ m = 0, \ \bar{y}_0^{(M)} = y_0, \ e_{R0} = 0 \\ x = x_1, \ m = 1, \ \bar{y}_1^{(M)} = \bar{y}_1 \ , \end{cases} \tag{9.81}$$

valoarea $\bar{y}_1$ fiind furnizată de o metodă pas cu pas sau de o altă metodă. Pe de altă parte, soluţia exactă (9.65), pentru $a_i = a$ =constant şi $b_i = b$ =constant, devine

$$y_{m+1} = y_m - A_m(e^{b\,h} - 1) - h\,a/b \ . \tag{9.82}$$

Înlocuind apoi A_m din (9.64) şi f_m dintr-o relaţie de tipul (9.76) din (9.82) se obţine următoarea ecuaţie în diferenţe finite

$$y_{m+1} - y_m e^{b\,h} = [(a + bf_0)/b^2 - y_0](e^{b\,h} - 1) - ha/b + h\,(a/b)(e^{b\,h} - 1)m \ .$$

Soluţia ecuaţiei în diferenţe finite de mai sus este

$$y_m = K e^{b\,h_m} + y_0 - m(a + bf_o)/b + e'_{Rm} \ , \tag{9.83}$$

unde e'_{Rm} este o constantă de rotunjire, iar constanta K se determină din condiţia $x = x_0$, $y_m = y_0$, $e'_{R0} = 0$. Rezultă soluţia exactă în absenţa erorilor de rotunjire

$$y_m = \frac{(a + bf_0)}{b^2}(e^{b\,h_m} - 1) - \frac{a\,h}{b} m + y_0 + e'_{Rm} \ . \tag{9.84}$$

Se observă prezenţa termenului exponenţial $e^{b\,h_m}$. Pe de altă parte, utilizând condiţia (9.81), ecuaţia (9.80) devine

$$\bar{y}_m^{(M)} = \tfrac{1}{\bar{b}^2}(\bar{a} + \bar{b}\,\bar{f}_0)(r_{1m} - 1) + C_2(r_{2m} - r_{1m}) - \tfrac{1}{\bar{b}}\bar{a}\,h\,m + y_0 + e_{Rm} \ . \tag{9.85}$$

Impunând, pentru comparaţie, condiţia ca la primul pas erorile de trunchiere să fie neglijabile şi luând în plus $\bar{a} = a$, $\bar{b} = b$, vom scrie

$$\bar{y}_m^{(M)} = y_1 = \tfrac{1}{b^2}(a + b\,f_0)(e^{b\,h} - 1) - h\,a/b + e'_{R1} \ . \tag{9.86}$$

Se deduce astfel expresia constantei C_2 din soluţia (9.85)

$$\begin{aligned} C_2 &= \tfrac{3 - b\,h}{29 + 3b^2h^2}\left[\tfrac{a + bf_0}{b^2}(r_1 - e^{b\,h}) + e_{R1} - e'_{R1}\right] = \\ &= \tfrac{3 - b\,h}{29 + 3b^2h^2}\left[\tfrac{a + bf_0}{b^2}\left(\tfrac{b^5h^5}{180} + O(b^6h^6)\right) + e_{R1} - e'_{R1}\right] \ . \end{aligned} \tag{9.87}$$

În (9.87) s-a ţinut seama că dezvoltările în serie Taylor ale funcţiilor r_1 şi e^{bh} coincid până la termenii în b^5h^5. Mai precis

$$r_1 = \sum_{i=0}^{4} \frac{(bh)'}{i!} + \frac{b^5h^5}{72} + O(b^6h^6),\ |b\,h| < 1\ . \tag{9.88}$$

Eroarea absolută în cazul metodei Milne-predictor este

$$e_m^{(M)} = y_m - \bar{y}_m^{(M)} = \tfrac{1}{b^2}(a + b\,f_0)(e^{b\,h_m} - r_{1m}) - C_2(r_{2m} - r_{1m}) + e'_{Rm} - e_{Rm}\ ,$$

depinzând atât de r_1 cât şi de r_2 (vezi (9.79)). Se observă apariţia unui termen suplimentar în ecuaţia cu diferenţe Milne, legat de existenţa a două soluţii r_1, r_2 ale ecuaţiei în diferenţe (9.77), faţă de soluţia exactă (9.84). Dintre cele două soluţii, r_1 este cea care urmează îndeaproape exponenţiala e^{bh}, în timp ce r_2 este o *soluţie străină*.

Dacă $bh \in (0, 1)$, atunci $r_1 > 1$, iar r_2 având dezvoltarea $r_2 = -(1 - bh/3 + b^2h^2/18 + \dots)$ este, în modul, subunitar ($|r_2| < 1$). Ca urmare, soluţia proprie, r_{1m}, este preponderentă, iar erorile de trunchiere sunt proporţionale cu $|e^{bh} - r_1|$, dacă $(a + bf_0) \neq 0$. Soluţia este *relativ stabilă*: eroarea relativă $|\varepsilon_m|$ descreşte.

Dacă $bh \in (-1, 0)$, atunci termenul în r_{2m} este preponderent ($|r_2 > 1$) şi soluţia este *numeric instabilă*, atât eroarea absolută cât şi cea relativă crescând în modul.

Cazul critic este: $a + bf_0 = 0$, $\bar{a} + \bar{b}f_0 \neq 0$, caz în care avem

$$C_2 \frac{3 - b\,h}{2\sqrt{9 + 3\bar{b}^2h^2}} \left[\frac{\bar{a} + \bar{b}f_0}{\bar{b}^2} r_1 + e_{R1} - e'_{R1} \right]\ , \tag{9.89}$$

iar soluţia este instabilă numeric, ca şi în cazul metodei Runge-Kutta (*instabilitate inerentă*).

9.6 Sisteme de ecuaţii diferenţiale. Ecuaţii de ordin superior

Sistemele de ecuaţii diferenţiale ordinare se rezolvă numeric prin extinderea unor procedee utilizate la rezolvarea ecuaţiei diferenţiale de ordinul întâi. Aspecte noi sunt introduse însă de condiţiile impuse pentru selecţionarea soluţiei. Din acest punct de vedere distingem două categorii de probleme: (a) probleme cu valori iniţiale (*Cauchy*); (b) probleme cu valori la limite. În ambele cazuri se consideră sistemul de ecuaţii diferenţiale adus la forma canonică, adică la forma

$$y_i' = f_i(x, y_i),\ i \in \overline{1, n}\ , \tag{9.90}$$

n fiind numărul de ecuaţii, egal cu numărul funcţiilor necunoscute $y_i(x)$. Funcţiile $f_i(x, y_i)$ pot conţine pe lângă variabila independentă x, toate necunoscutele y_i, dar nu pot conţine nici o derivată.

Ordinul sistemului de ecuaţii diferenţiale se obţine adunând ordinele tuturor ecuaţiilor din sistem. Numărul de condiţii necesare pentru ca soluţia să fie unic determinată este egal cu ordinul sistemului (în cazul de faţă ordinul este n).

9.6.1 Probleme cu valori iniţiale

În acest caz, considerând un interval $[a,b]$ pe care se construieşte soluţia, condiţiile sunt impuse într-un singur punct, uzual $x=a$. Combinaţia de ecuaţii diferenţiale cu condiţii într-un sigur punct,

$$y_i' = f(x, y_1, , y_n),\ x=a,\ y=y_{i0},\ i\in\overline{1,n}\ , \tag{9.91}$$

se numeşte problemă *Cauchy*, sau problemă cu *valori iniţiale*. Se presupun îndeplinite condiţiile necesare pentru ca problema Cauchy să admită soluţie unică. Fie x_i, $i=0,1,\dots,N$ o partiţie uniformă a intervalului $[a,b]$, $x_0=a$, $x_N=b$ având pasul $h=(x_N-x_0)/N$. Aplicând formula Runge-Kutta de ordinul patru (9.22) pentru fiecare ecuaţie a sistemului (9.90) se obţine

$$y_{i,m+1} = y_{i,m} + \tfrac{h}{6}(K_{i1}+2K_{i2}+2K_{i3}+K_{i4}),\ i\in\overline{1,n}\ , \tag{9.92}$$

unde

$$\begin{array}{ll} K_{i1}=f_i(x_m,y_{i,m}), & K_{i2}=f_i(x_m+\frac{h}{2},y_{i,m}+\frac{h}{2}K_{i1}),\\ K_{i3}=f_i(x_m+\frac{h}{2},y_{i,m}+\frac{h}{2}K_2), & K_{i4}=f_i(x_m+h,y_{i,m}+hK_{i3})\ . \end{array} \tag{9.93}$$

Ordinea de calcul a derivatelor este următoarea: se calculează K_{i1} pentru toţi indicii $i=1,2,\dots,n$, apoi K_{i2}, ş.a.m.d. Plecând din punctul x_0 cu valorile iniţiale date de (9.91) se calculează valorile $y_{i,m}$ pas cu pas.

Criteriile de modificare a pasului sunt similare cu cele din cazul unei singure ecuaţii diferenţiale. Folosirea *criteriului simplu* constă în calculul raportului

$$r=\sup_i\left|\frac{K_{i2}-K_{i3}}{K_{i1}-K_{i2}}\right|\sim 0.01 \tag{9.94}$$

şi înjumătăţirea pasului dacă $r\geq 0.1$, sau dublarea pasului dacă $r\leq 0.001$.

Criteriul calcului dublu cu paşii h şi $2h$ consideră, după modelul (9.25) cantitatea δ

$$\delta=\sup_i\frac{|y_i^{(h)}-y_i^{(2h)}|}{2^s-1}\ , \tag{9.95}$$

unde $y_i^{(h)}$este calculat cu pasul h iar $y_i^{(2h)}$cu pasul $2h$, s fiind ordinul metodei. Pasul se păstrează dacă δ se află într-un interval considerat dinainte. Limitele pentru δ se stabilesc în funcţie de problema care se rezolvă. În cazul când se lucrează adimensional, $\delta\in(10^{-6},10^{-4})$ este un interval uzual, în funcţie de precizia dorită şi de calculator. În locul normei maxime din relaţiile (9.94) şi (9.95), se pot folosi orice alte norme de vectori.

În mod similar, pentru rezolvarea sistemului (9.90) se pot utiliza metode multipas. Pentru formula *predictor-corector Adams*, spre exemplu, se scrie

$$\begin{array}{l} y_{i,m+1}^{(Ap)} = y_{i,m}+\frac{h}{24}(55f_{i,m}-59f_{i,m-1}+37f_{i,m-2}-9f_{i,m-3})\ ,\\ y_{i,m+1}^{(Ac)} = y_{i,m}+\frac{h}{24}(9f_{i,m+1}+19f_{i,m}-5f_{i,m-1}+f_{i,m-2})\ ,\\ y_{i,m+1} = y_{i,m+1}^{(Ac)}-\frac{19}{270}(y_{i,m+1}^{(Ac)}-y_{i,m+1}^{(Ap)}),\ i\in\overline{1,n}\ . \end{array} \tag{9.96}$$

Condiţia de convergenţă a procedeului predictor-corector (vezi relaţiile (8.7) şi (8.12)) este luată sub una din formele

$$h \cdot \sup |\partial f_i/\partial y_j| < (1/n)(24/9);\ h \cdot |\partial f_i/\partial y_j|_\infty < (24/9),\ i,j \in \overline{1,n}\ , \quad (9.97)$$

unde $J = (\partial f_i/\partial y_j)_{i,j\in\overline{1,n}}$ este matricea jacobiană a funcţiilor pantă.

Criteriul de menţinere sau de modificare a pasului îl reprezintă păstrarea sub o anumită limită a diferenţei de valori între formula predictor şi formula corector, spre exemplu

$$\tfrac{19}{270} \sup |y_{i,m+1}^{(Ac)} - y_{i,m+1}^{(Ap)}| < \varepsilon\ , \quad (9.98)$$

cu $\varepsilon = 10^{-s}$, s fiind numărul de cifre semnificative exacte dorite în soluţia aproximativă. Dacă s-a lucrat adimensional, normându-se rezonabil variabilele, $\varepsilon \in (10^{-6}, 10^{-4})$ pentru o precizie corespunzând la $4 \div 6$ zecimale.

9.6.2 Probleme cu valori la limite

Considerăm în cele ce urmează sistemul de ecuaţii cu condiţiile la limite

$$y_i' = f(x, y_1, , y_n),\ \begin{cases} x = x_0, & y_j = y_{j,0},\ j \in \overline{1,n_1}\ , \\ x = x_N, & y_k = y_{k,N},\ k \in \overline{1,n_2}\ . \end{cases} \quad (9.99)$$

Avem deci n_1 condiţii impuse la un capăt al intervalului şi n_2 condiţii la celălalt capăt, cu $n_1 + n_2 = n$. Deoarece în punctul iniţial $x = x_0$ nu sunt date toate valorile funcţiilor $y_{i,0}$, $i \in \overline{1,n}$, startul nu poate fi luat. Metodele de rezolvare a problemelor cu condiţii la limite se împart în două mari categorii: (a) *metoda tirului* care se bazează pe transformarea problemei cu condiţii la limite într-o problemă cu condiţii iniţiale, prin estimarea valorilor lipsă de la unul din capetele intervalului în aşa fel încât să fie îndeplinite condiţiile la limită de la celălalt capăt; (b) *metode cu diferenţe finite* în care se folosesc expresii cu diferenţe finite pentru aproximarea derivatelor pe o diviziune a intervalului de integrare.

Metoda tirului.

Din punct de vedere al dificultăţii rezolvării unei probleme cu condiţii la limite, distingem două situaţii pe care le vom trata separat: (a) probleme cu valori la limite uniparametrice, când $n_1 = 1$ sau $n_2 = 1$; (b) probleme cu valori la limite cu doi sau mai mulţi parametri.

Probleme cu valori la limite uniparametrice. Să considerăm că, pentru sistemul (9.90) condiţiile (9.91) sunt de forma

$$\begin{cases} x = x_0, & y_j = y_{j,0},\ j \in \overline{1,n-1}\ , \\ x = x_N, & y_n = y_{n,N}^*\ , \end{cases} \quad (9.100)$$

adică nu este cunoscută valoarea $y_{n,0}$ la $x = x_0$. Vom considera $\lambda \equiv y_{n,0}$ ca parametru şi vom observa că acesta trebuie determinat astfel încât la $x = x_N$ să fie satisfăcută condiţia din (9.100), adică λ este soluţia ecuaţiei

$$F(\lambda) \equiv y_{n,N}(\lambda) - y^*_{n,N} = 0,\ \lambda \equiv y_{n,0}\ . \tag{9.101}$$

Fie λ^* valoarea care satisface ecuaţia (9.101). Pentru a o găsi numeric, trebuie mai întâi *localizată*, adică trebuie găsite două valori λ_{k-1}, λ_{k-2} astfel încât $F(\lambda)$ să-şi schimbe semnul. Aceste valori se aleg prin încercări, având în vedere şi eventuale alte informaţii (s.e. de natură fizică). Se introduce apoi un şir iterativ folosind *metoda coardei* prezentată în capitolul 3, adică pentru fiecare pereche λ_{k-2}, λ_{k-1}, cu $F(\lambda_{k-2}) \cdot F(\lambda_{k-1}) < 0$ se determină λ_k cu relaţia

$$\lambda_k = \lambda_{k-1} - \frac{F(\lambda_{k-1})(\lambda_{k-1} - \lambda_{k-2})}{F(\lambda_{k-1}) - F(\lambda_{k-2})}\ . \tag{9.102}$$

În iteraţia următoare se atribuie valoarea λ_k uneia din valorile λ_{k-1}, λ_{k-2} astfel încât să se păstreze schimbarea de semn. Procedeul se opreşte dacă sunt îndeplinite condiţiile $|F(\lambda)| < \varepsilon_F$, sau $|\lambda_k - \lambda_{k-1}| < \varepsilon_\lambda |\lambda_k| + \varepsilon_s$, unde ε_s este valoarea introdusă pentru eventualitatea când $\lambda^* = 0$.

Un caz particular îl constituie ecuaţiile diferenţiale de ordinul doi[3] de forma

$$y'' = f(x, y, y'),\ y(x_0) = \alpha,\ y(x_N) = \beta\ . \tag{9.103}$$

În cazul în care f este de forma

$$f(x, y, y') = u(x) + v(x)y + w(x)y'\ , \tag{9.104}$$

atunci ecuaţia este liniară iar soluţia este de forma

$$y(x) = \mu y_1(x) + (1 - \mu) y_2(x)\ , \tag{9.105}$$

unde y_1 şi y_2 sunt două soluţii obţinute pentru două valori diferite ale parametrului $\lambda \equiv y'(x_0)$, iar μ se determină din condiţia ca $y(x_N) = \beta$ rezultând

$$\mu = [\beta - y_2(x_N)]/[y_1(x_N) - y_2(x_N)]\ . \tag{9.106}$$

Cele două soluţii se pot obţine simultan, rezolvând sistemul

$$\begin{array}{c} y_1' = y_3,\ y_2' = y_4,\ y_3' = f(x, y_1, y_3),\ y_4' = f(x, y_2, y_4)\ , \\ y_1(x_0) = \alpha,\ y_2(x_0) = \alpha,\ y_3(x_0) = \lambda_1 = 0,\ y_4(x_0) = \lambda_2 = 1\ , \end{array} \tag{9.107}$$

unde pentru cele două valori ale parametrului λ am ales $\lambda_1 = 0$ şi $\lambda_2 = 1$, iar pentru rezolvare putem folosi oricare din metodele pas cu pas. Valorile $y_1(x_m)$ şi $y_2(x_m)$ obţinute prin integrare numerică trebuie memorate în doi vectori pentru a putea construi în final soluţia cu relaţia (9.105) folosind μ dat de (9.106).

[3] Vezi paragraful 9.6.3 pentru ecuaţii diferenţiale de ordin superior.

În cazul în care ecuaţia (9.103) este neliniară, în locul metodei coardei (9.102) se poate folosi metoda Newton pentru rezolvarea ecuaţiei (9.101), în care se construieşte şirul de aproximaţii ale valorii λ^* dat de relaţia

$$\lambda_{k+1} = \lambda_k - F(\lambda_k)/F'(\lambda_k) \ , \tag{9.108}$$

unde

$$F(\lambda) \equiv y_N(\lambda) - \beta = 0, \ \lambda \equiv y'(x_0) \ . \tag{9.109}$$

Pentru determinarea derivatei funcţiei F, $F' \equiv \partial F/\partial\lambda = \partial y(x_N)/\partial\lambda$, se derivează funcţia y'' dată de (9.103) în raport cu λ

$$\frac{\partial y''}{\partial\lambda} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial\lambda} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial\lambda} + \frac{\partial f}{\partial y'}\frac{\partial y'}{\partial\lambda} \ . \tag{9.110}$$

Notând $u \equiv \partial y/\partial\lambda$ şi ţinând cont că

$$\frac{\partial x}{\partial\lambda} = 0, \ \frac{\partial y''}{\partial\lambda} = \frac{\partial}{\partial\lambda}\left(\frac{\partial^2 y}{\partial x^2}\right) = \frac{\partial^2}{\partial x^2}\left(\frac{\partial y}{\partial\lambda}\right) = u'' \ , \tag{9.111}$$

şi că $\partial y(x_0)/\partial\lambda = 0$, $\partial y'(x_0)/\partial\lambda = 1$, obţinem ecuaţia diferenţială de ordinul doi cu condiţii iniţiale

$$u'' = u\frac{\partial f(x,y,y')}{\partial y} + u'\frac{\partial f(x,y,y')}{\partial y'}, \ u(x_0) = 0, \ u'(x_0) = 1 \ . \tag{9.112}$$

Cu λ ales se rezolvă numeric cu o metodă pas cu pas sistemul de ecuaţii diferenţiale

$$\begin{gathered} y_1' = y_2, \ y_2' = f(x,y,y'), \ y_3' = y_4, \ y_4' = y_3(\partial f/\partial y) + y_4(\partial f/\partial y') \ , \\ y_1(x_0) = \alpha, \ y_2(x_0) = \lambda, \ y_3(x_0) = 0, \ y_4(x_0) = 1 \ , \end{gathered} \tag{9.113}$$

unde $y_1 \equiv y$, $y_3 \equiv u$. Valoarea $u(x_N) \equiv y_3(x_N)$ este tocmai $F'(\lambda)$, astfel încât o valoare mai bună pentru λ se poate calcula cu relaţia (9.108). Ca test de oprire se poate folosi condiţia

$$|\lambda_{k+1} - \lambda_k| < \varepsilon|\lambda_{k+1}| + \varepsilon_s \ , \tag{9.114}$$

unde valoarea ε_s (pentru care se poate lua $\varepsilon_s = \varepsilon$) trebuie introdusă pentru a elimina nedeterminarea ce poate apărea dacă valoarea exactă este chiar $\lambda = 0$. Pentru $\varepsilon_s = 0$ şi $\varepsilon = 10^{-s}$, condiţia (9.114) este îndeplinită atunci când λ_{k+1} şi λ_k au s cifre semnificative identice.

Probleme cu valori la limite cu doi sau mai mulţi parametri.

În cazul când problema are mai mulţi parametri dificultăţile cresc; pasul esenţial îl reprezintă trecerea de la un parametru la doi parametri, mai departe extinderea fiind evidentă. Pentru exemplificare să presupunem sunt impuse condiţiile

$$\left\{ \begin{array}{ll} x = x_0, & y_j = y_{j,0}, \ j \in \overline{1, n-2} \ , \\ x = x_N, & y_{n-1} = y^*_{n-1,N}, \ y_{n,N} = y^*_{n,N} \ . \end{array} \right. \tag{9.115}$$

Vom considera ca parametri valorile $y_{n-1,0} = \lambda_1$, $y_{n,0} = \lambda_2$. Soluţiile sistemului (9.115) trebuie să verifice sistemul de ecuaţii

$$\begin{cases} F_1(\lambda_1, \lambda_2) \equiv y_{n-1,N}(\lambda_1, \lambda_2) - y^*_{n-1,N} = 0 \ , \\ F_2(\lambda_1, \lambda_2) \equiv y_{n,N}(\lambda_1, \lambda_2) - y^*_{n,N} = 0 \ . \end{cases} \tag{9.116}$$

Cu $y_{n-1,N}(\lambda_1, \lambda_2)$ şi $y_{n,N}(\lambda_1, \lambda_2)$ s-au notat valorile funcţiilor y_{n-1} respectiv y_n care se obţin prin integrare numerică la $x = x_N$ cu parametrii λ_1 şi λ_2 aleşi. Pentru a găsi soluţia $(\lambda_1^*, \lambda_2^*)$ care satisface condiţiile (9.116) se poate aplica una din metodele de rezolvare a sistemelor de ecuaţii neliniare prezentate în capitolul 8, sau se poate aplica o metodă de optimizare pentru funcţia scop

$$U(\lambda_1, \lambda_2) = \tfrac{1}{2}[F_1^2(\lambda_1, \lambda_2) + F_2^2(\lambda_1, \lambda_2)] \ . \tag{9.117}$$

Spre exemplu, utilizând metoda gradientului (vezi 8.4), se scrie şirul de iteraţii $\lambda^{k+1} = \lambda^k - \alpha_k g^k$ unde g este vectorul gradient, $g^k = \nabla U(\lambda_1^k, \lambda_2^k) = J^T \cdot F$, J este jacobiana funcţiei vectoriale $F = (F_1, F_2)^T$. Parametrul $\alpha_k > 0$ se determină optimizând descreşterea funcţiei U în lungul direcţiei gradientului negativ $-g$.

Plecând dintr-un punct ales iniţial $(\lambda_1^0, \lambda_2^0)$, este necesară cunoaşterea matricei jacobian J_0 pentru a găsi direcţia de descreştere. Cum în general este imposibil de determinat expresia analitică a jacobienei, aceasta se poate aproxima cu ajutorul diferenţelor la dreapta

$$J_0 \cong \begin{pmatrix} [F_1(\lambda_{10} + h_1, \lambda_{20}) - F_{10}]/h_1 & [F_1(\lambda_{10}, \lambda_{20} + h_2) - F_{10}]/h_2 \\ [F_2(\lambda_{10} + h_1, \lambda_{20}) - F_{20}]/h_1 & [F_1(\lambda_{10}, \lambda_{20} + h_2) - F_{20}]/h_2 \end{pmatrix} . \tag{9.118}$$

Procedeul este acelaşi indiferent de numărul de parametri ($n \geq 2$). Dacă numărul de parametri este mai mare decât 2, se recomandă o metodă de descreştere *cvasi-Newton* sau de *gradient conjugat*.

Vom observa că numărul real de parametri nu poate depăşi 2 dacă ordinul sistemului $n \leq 4$, 3 dacă $n \leq 6$, în general $[n/2]$ (parte întreagă) deoarece putem alege convenabil ca punct de plecare x_0 sau x_N, adică punctul unde se dau cele mai multe condiţii.

Metode cu diferenţe finite.

Se bazează pe folosirea unei o reţele de puncte de diviziune ale intervalului care se stabileşte înainte de integrarea ecuaţiilor diferenţiale. De regulă nu se cunosc dinainte zonele în care funcţiile necunoscute prezintă variaţii mai rapide unde nodurile reţelei ar trebui să fie mai dese. Deşi există tehnici adaptive şi pentru metodele cu diferenţe finite, care constau în îndesirea reţelei în zonele de variaţie rapidă a funcţiilor necunoscute, acestea sunt mai complexe decât cele corespunzătoare metodelor pas cu pas folosite la metoda tirului, astfel încât metodele cu diferenţe finite se recomandă să fie folosite pentru rezolvarea problemelor cu condiţii la limită a căror soluţie nu prezintă zone de variaţie rapidă.

Fie un sistem de ecuaţii diferenţiale $y_i' = f_i(x, y_1, \dots, y_n)$ scris în forma vectorială

$$Y' = F(x, Y) \ , \tag{9.119}$$

cu condiţiile la limită de forma (9.99). Se aproximează derivatele cu diferenţe finite la dreapta

$$y_i'(x_m) = (y_i^{m+1} - y_i^m)/h + O(h), \ i \in \overline{1,n}, \ m \in \overline{0, N-1} \ , \tag{9.120}$$

sau, cu efecte benefice asupra preciziei, cu diferenţe centrate

$$y_i'(x_m) = (y_i^{m+1} - 2y_i^m + y_i^{m-1})/(2h) + O(h^2), \ i \in \overline{1,n}, \ m \in \overline{1, N-1} \ , \tag{9.121}$$

unde pentru simplitate am considerat o diviziune cu pasul constant $h = x_{m+1} - x_m$. Considerând spre exemplu schema cu diferenţe finite la dreapta, sistemul (9.119) se scrie în forma

$$Y^{m+1} - Y^m = hF[\tfrac{1}{2}(x_{m+1} + x_m), \tfrac{1}{2}(Y^{m+1} + Y^m)] + O(h^2) \ . \tag{9.122}$$

Neglijând termenii $O(h^2)$, relaţia (9.122) se scrie în forma

$$E^{m+1} = Y^{m+1} - Y^m - hF[\tfrac{1}{2}(x_{m+1} + x_m), \tfrac{1}{2}(Y^{m+1} + Y^m)] \ = 0, \ m \in \overline{0, N-1} \ , \tag{9.123}$$

obţinând un sistem de $n \times N$ ecuaţii neliniare cu $n \times (N+1)$ necunoscute y_i^m, pentru a cărui rezolvare vom folosi metoda iterativă Newton prezentată la **??**. Cele n ecuaţii lipsă se completează folosind condiţiile la limită (9.99) care uneori pot fi date în forma generală

$$\begin{array}{l} L_{0j}(x_0, y_1, \dots, y_n) = 0, \ j \in \overline{1, n_1} \ , \\ L_{Nk}(x_0, y_1, \dots, y_n) = 0, \ k \in \overline{1, n_2} \ , \end{array} \tag{9.124}$$

unde L_{0j} şi L_{Nk} sunt expresii liniare sau neliniare care conţin funcţiile necunoscute y_i, $i \in \overline{1,n}$. Relaţiile (9.124) se pun în aceaşi formă cu (9.123)

$$\begin{array}{l} E_i^0 \equiv 0, \ i \in \overline{1, n-n_1}, \quad E_i^0 = L_{0i}(x_0, y_1, \dots, y_n), \ i \in \overline{n-n_1+1, n}, \\ E_i^{N+1} = L_{Ni}(x_0, y_1, \dots, y_n), \ i \in \overline{1, n_2}, \quad E_i^{N+1} \equiv 0, \ i \in \overline{n_2+1, n} \ . \end{array} \tag{9.125}$$

Aşa cum se va vedea în continuare, este convenabil ca cele n_1 condiţii la limita $x = x_0$ să fie plasate în vectorul E^0 pentru ultimele valori ale indicelui i, iar cele de la limita $x = x_N$ pentru primele valori ale indicelui i. Ecuaţiile (9.123) se liniarizează folosind o dezvoltare în serie Taylor pentru funcţiiile de mai multe variabile $E_i^{m+1}(y_1^m, \dots, y_n^m, y_1^{m+1}, \dots, y_n^{m+1})$, pentru care vom folosi notaţia vectorială $E^{m+1}(Y^m, Y^{m+1})$

$$\begin{array}{l} E^{m+1}(Y^m + \delta Y^m, Y^{m+1} + \delta Y^{m+1}) \cong E^{m+1}(Y^m, Y^{m+1}) + \\ \quad + \sum_{i=1}^n (\partial E^{m+1}/\partial y_{i,m}) \delta y_{i,m} + \sum_{i=1}^n (\partial E^{m+1}/\partial y_{i,m+1}) \delta y_{i,m+1} \ . \end{array} \tag{9.126}$$

Se obţine un sistem de ecuaţii liniare de forma

$$\sum_{j=1}^{n} J_{i,j}\delta y_i^m + \sum_{j=1}^{n} J_{i,n+j}\delta y_i^{m+1} = -E_i^m,\ J_{i,j} = \frac{\partial E_i^m}{\partial y_j^m},\ J_{i,n+j} = \frac{\partial E_i^m}{\partial y_j^{m+1}}\ . \tag{9.127}$$

Deşi sistemul (9.127) este de regulă mare[4] se poate rezolva foarte eficient cu metoda eliminării gaussiene dacă de ţine cont de structura matricei sistemului care conţine elementele nenule grupate în blocuri de dimensiuni $n \times 2n$. Relaţia (9.128) ilustrează structura sistemului liniar pentru $n = 3$ şi $N = 4$, cu 2 condiţii la $x = x_0$, $y_1(x_0) = y_{10}$, $y_2(x_0) = y_{20}$ şi o condiţie la $x = x_N$, $y_1(x_N) = y_{1N}$. Au fost reprezentate numai elementele nenule. Pentru astfel de sisteme cu matrice rare există tehnici eficiente de optimizare a memoriei utilizate (vezi 5.1.11), cu preţul unor calcule suplimentare.

După cum am văzut la 8.2, metoda iterativă Newton converge rapid dacă dispunem de un punct suficient de apropiat de soluţie. Pentru obţinerea aproximaţii de start, se poate folosi o metodă pas cu pas cu care se integrează ecuaţiile diferenţiale cu condiţiile la limită completate la unul din capetele intervalului, la fel ca la metoda tirului. În unele cazuri avem de rezolvat de mai multe ori acelaşi sistem de ecuaţii diferenţiale care suferă mici modificări. În asemenea cazuri, o soluţie obţinută poate fi folosită ca punct de start pentru calculele următoare.

$$\begin{bmatrix}
\mathrm{x} & \mathrm{x} & \mathrm{x} & & & & & & & & & \\
\mathrm{x} & \mathrm{x} & \mathrm{x} & & & & & & & & & \\
\mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & & & & & & \\
\mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & & & & & & \\
\mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & & & & & & \\
 & & & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & & & \\
 & & & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & & & \\
 & & & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & & & \\
 & & & & & & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} \\
 & & & & & & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} \\
 & & & & & & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} & \mathrm{x} \\
 & & & & & & & & & \mathrm{x} & \mathrm{x} & \mathrm{x}
\end{bmatrix} \times \begin{bmatrix} y_3^0 \\ y_1^1 \\ y_2^1 \\ y_3^1 \\ y_1^2 \\ y_2^2 \\ y_3^2 \\ y_1^3 \\ y_2^3 \\ y_3^3 \\ y_2^N \\ y_3^N \end{bmatrix} = - \begin{bmatrix} E_3^0 \\ E_1^1 \\ E_2^1 \\ E_3^1 \\ E_1^2 \\ E_2^2 \\ E_3^2 \\ E_1^3 \\ E_2^3 \\ E_3^3 \\ E_2^N \\ E_3^N \end{bmatrix} \tag{9.128}$$

9.6.3 Ecuaţii diferenţiale de ordin superior

Vom considera ecuaţia diferenţială de forma

$$y^{(n)} = E(x, y', y', \ldots, y^{(n-1)})\ , \tag{9.129}$$

[4] Spre exemplu pentru cinci funcţii necunoscute şi o discretizare cu 200 de noduri avem 1000 de ecuaţii cu 1000 de necunoscute.

E fiind o funcţie care asigură existenţa soluţiei pe un interval $[a,b] \subset \mathbf{R}$. Se observă că ecuaţia (9.129) poate fi transformată într-un sistem de ecuaţii diferenţiale de forma canonică (9.90), notând

$$y_1 \equiv y,\ y_2 \equiv y',\ y_3 \equiv y'', \dots, y_n \equiv y^{(n-1)}\ . \tag{9.130}$$

În acest caz, funcţiile $f_i = (x_i, y_1, y_2, \dots, y_n)$ au expresiile

$$f_i = y_{i+1},\ i \in \overline{1, n-1};\ f_n = E(x, y_1, y_2, \dots, y_n) \tag{9.131}$$

şi deci problema se reduce la rezolvarea unui caz particular de sistem de ecuaţii diferenţiale. Vom observa că problema cu valori iniţiale revine la cunoaşterea funcţiei y şi a derivatelor până la ordinul $n-1$ inclusiv în punctul $x = x_0$.

9.7 Sisteme cu scări disparate

Sistemele de ecuaţii diferenţiale ordinare cu scări disparate provin de regulă din descrierea unor fenomene evolutive cu scări de timp diferite. De exemplu, controlul automat al traiectoriei unei aeronave impune corecţii rapide în cazul unor abateri de la traiectoria programată. Un alt exemplu poate fi preluat în cazul reacţiilor chimice complexe cu o gamă largă de valori pentru vitezele de reacţie. În astfel de situaţii, condiţiile de stabilitate ale metodelor de rezolvare impun paşi de integrare exagerat de mici şi care nu sunt justificaţi de vitezele de variaţie ale unor funcţii necunoscute.

Pentru exemplificare, vom considera sistemul

$$\begin{cases} y_1' = 998y_1 + 1998y_2, & y_1(0) = 1\ , \\ y_2' = -999y_1 - 1999y_2, & y_2(0) = 0\ , \end{cases} \tag{9.132}$$

cu soluţia $y_1 = 2e^{-x} - e^{-1000x}$, $y_2 = -e^{-x} + e^{-1000x}$. În cazul folosirii unei metode explicite, termenul e^{-1000x} din soluţia exactă impune un pas de integrare $h \ll 1/1000$ pentru asigurarea stabilităţii, chiar dacă e^{-1000x} devine complet neglijabil în comparaţie cu e^{-x} imediat ce ne depărtăm de origine. În astfel de situaţii, se recomandă folosirea unei formule implicite. În general, folosirea unor formule implicite duce la condiţii mult mai puţin restrictive din punct de vedere al stabilităţii. În cazul ecuaţiilor diferenţiale liniare, folosind o formulă implicită, se obţine o metodă necondiţionat stabilă.

Se pot folosi formulele de tip predictor-corector prezentate la 9.4. O dificultate suplimentară o constituie în acest caz startul metodei, care trebuie făcut cu o metodă pas cu pas. Folosirea unei formule explicite ar duce la necesitatea utilizării unui pas foarte mic, astfel încât se recomandă folosirea unei formule implicite. Spre exemplu, o formulă implicită de ordinul întâi se scrie

$$y_i^{m+1} = y_i^m + hf_i(x_{m+1}, y_k^{m+1}),\ i,k \in \overline{1,n}\ , \tag{9.133}$$

obţinându-se un sistem de ecuaţii neliniare cu necunoscutele y_i^{m+1}, care se poate liniariza dezvoltând în serie Taylor funcţiile de $n+1$ variabile $f_i(x_{m+1}, y_k^{m+1})$

şi păstrând numai termenii liniari

$$y_i^{m+1} = y_i^m + h\left[f_i(x_{m+1}, y_k^m) + \sum_{j=1}^{n}(y_j^{m+1} - y_j^m)\frac{\partial f_i(x_m, y_k^m)}{\partial y_j}\right]. \qquad (9.134)$$

Obţinem astfel o *metodă semi-implicită*. Mărimea pasului va fi impusă acum în principal de precizie (care este de ordinul întâi) şi nu de stabilitatea metodei, care este în general asigurată.

Pentru a obţine o metodă de ordinul al doilea, vom scrie

$$y_i^{m+1} = y_i^m + \tfrac{h}{2}[f_i(x_m, y^m) + f_i(x_{m+1}, y_m^{m+1})]\ . \qquad (9.135)$$

Liniarizând din nou obţinem

$$y_i^{m+1} = y_i^m + \frac{h}{2}\left[f_i(x_m, y_k^m) + f_i(x_{m+1}, y_k^m) + \sum_{j=1}^{n}(y_j^{m+1} - y_j^m)\frac{\partial f_i(x_m, y_k^m)}{\partial y_j}\right].$$

Este posibilă şi obţinerea unor metode pas cu pas de ordine mai mari de 2 cu preţul unor calcule laborioase, care nu întotdeauna sunt justificate dacă folosim o liniarizare de forma (9.134) sau celei de mai sus.

EXEMPLUL 1. Fie ecuaţia diferenţială $y'' = 4kxy' - (4k^2x^2 + m^2 - 2k)y + m^2e^{kx^2}$, $y(0) = 1, y'(0) = m \cdot a$. Se cere soluţia pe intervalul $[0, 1.6]$, pentru (a) $a = 1, m = 15, k = 1$ şi (b) $a = 13, m = 1, k = 0$.

Rezolvare. Ecuaţia are soluţia $y = [1 + a\sin(m\,x)]e^{kx^2}$, deci vom putea compara rezultatele numerice cu cele exacte. În figura 9.3 sunt reprezentate funcţiile $y(x)$ şi $y^{(6)}(x)$ pentru $k = 1$. Rezultatele calculelor sunt date în tabelele 9.3 şi 9.4, pentru (a), respectiv (b). S-au folosit metodele Runge-Kutta-Gill, Adams şi Milne cu paşii (a) $h =$ 0.05, 0.02, 0.01, respectiv (b) $h =$ 0.2, 0.1, 0.05. Pentru metodele predictor-corector sunt date trei seturi de rezultate: P predictor, $C1$ o singură corecţie, $C2$ două corecţii. Valorile din tabele reprezintă erorile absolute calculate cu formula $e = (\bar{y} - y_{exact})$ şi amplificate cu 10^4, respectiv 10^6. Pe ultima coloană este dat numărul de evaluări ale funcţiei pantă F pentru $x = 1.6$ care poate fi calculat cu relaţia $n_F = s{\cdot}r{+}(N{-}r)(1{+}c)$, unde s este ordinul metodei pas-cu-pas folosită pentru start, r este gradul polinomului de interpolare folosit în formula multipas, $N = (x_{\max} - x_{\min})/h$ este numărul de paşi, iar c este numărul de corecţii efectuate cu formula corector.

Pentru cazul (a), pasul $h = 0.05$ se dovedeşte a fi prea mare. Spre exemplu, eroarea absolută în $x = 0.4$ pentru metoda Runge-Kutta este $-8.9 \cdot 10^{-3}$. Expresia erorii pentru o metodă de ordinul 4 este de forma $e = Ch^5\Phi^{(4)}(\xi)$, unde C este o constantă, iar $\Phi(x) \equiv f(x, y(x)) = y''(x)$. Pentru exemplul considerat avem

$$\begin{aligned}\Phi^{(4)} \quad \equiv \quad & y^{(6)}(x) = [120 + 720x^2{+}480x^4{+}64x^6{+}\cos(15x)(8\,313\,300x - 525\,600x \\ & {+}2880x^5{+}\sin(15x)(-9\,912\,255 + 2\,876\,220x^2{-}53\,520x^4{+}64x^6)]e^{x^2},\end{aligned}$$

deci $e(0.4) = C \cdot (3 \cdot 10^{-7}) \cdot (6.8 \cdot 10^6) \cong 2.2C$. Pentru acest pas metodele predictor-corector dau rezultate foarte proaste. Pentru $h = 0.01$ şi $x = 1.61$, eroarea absolută

Figura 9.3: Graficele funcţiilor y, $y^{(6)}$.

este de ordinul 10^{-3} la toate metodele folosite. Creşterea preciziei prin folosirea formulelor implicite de tip corector nu este posibilă deoarece pasul este încă prea mare (relaţia (9.60)). Pentru $h = 0.01$ iese în evidenţă superioritatea (din punct de vedere al efortului de calcul) metodelor predictor-corector. Spre exemplu, folosind formulele Milne cu 2 corecţii, cu preţul a 484 evaluări ale funcţiilor pantă, se obţine acelaşi ordin de precizie cu metoda Runge-Kutta care necesită 640 evaluări, adică cu 32% mai mult, iar dacă folosim o singură corecţie, rezultate bune se obţin cu preţul a numai 326 evaluări, adică aproape de două ori mai puţine evaluări decât cu Runge-Kutta.

Pentru cazul (b), erorile absolute se menţin în limite normale (adică $O(h^5)$) pentru toate formulele aplicate. Derivata din expresia erorii absolute este $\Phi^{(4)}(x) \equiv y^{(6)}(x) = -13\sin(x)$, astfel că putem folosi paşi de integrare mai mari. În Tabelul 9.4 sunt date şi erorile absolute (multiplicate cu 10^6) obţinute aplicând relaţia (9.62), iar δ reprezintă valoarea corecţiei din această relaţie, $\delta = \frac{1}{29}(y_{m+1}^{(c)} - y_{m+1}^{(p)})$. După cum se observă din relaţiile (9.48) şi (9.49) precizia formulei Adams-predictor este mai mare decât a formulei Milne-predictor, iar din relaţiile (9.55) şi (9.56) se poate vedea cum formula Milne-corector este în general mai precisă decât Adams-corector. Faţă de cazul precedent, folosirea celei de-a doua corecţii duce la rezultate mai exacte. Cele mai bune rezultate le avem pentru $h = 0.05$ cu Milne folosind formula corector de două ori. Ordinul de precizie este acelaşi cu cel dat de formula Runge-Kutta, dar cu un efort de calcul cu aproximativ 30% mai mic, în timp ce folosirea unei singure corecţii duce la un efort de calcul cu 83% mai mic faţă de Runge-Kutta şi rezultate acceptabile. Metoda Runge-Kutta se dovedeşte însă mai stabilă în situaţii "dificile" decât formulele predictor-corector care sunt mai sensibile datorită extrapolării din formula predictor şi a procedeului de tip "iteraţie simplă" din formula corector a cărei condiţie suficientă de convergenţă este dată de (9.60). Aplicarea formulei (9.62) nu duce la rezultate sensibil mai exacte, însă δ reprezintă o informaţie utilă privind ordinul de mărime a erorii absolute, în special în situaţiile în care soluţia nu are variaţii puternice.

În tabelul 9.5 sunt datele referitoare la efortul de calcul necesar obţinerii unei

Tabelul 9.3: $a = 1$, $m = 15$, $k = 1$, erori absolute $\times 10^4$

Metoda	$x = 0.4$	$x = 0.8$	$x = 1.2$	$x = 1.6$	$nr.F$
$h = 0.05$					
RKG	-89	-61	546	4750	128
$Adams - P$	-197	-1656	-3441	1446	41
$C1$	-740	-2732	-5517	-1818	70
$C2$	-55	-1009	-6492	-34264	99
$Milne - P$	50	1089	-44215	452606	41
$C1$	-475	1344	-45857	533875	70
$C2$	-10	-156	-1311	-8149	99
$h = 0.02$					
RKG	-3.6	-7.0	-3.55	66.9	320
$Adams - P$	-9.65	-39	-160	-673	89
$C1$	-6.61	-42	-183	-772	166
$C2$	8.4	16	-4.2	-272	243
$Milne - P$	-4.8	-16.3	-49	-157	89
$C1$	-3.5	-16.7	-74	-239	166
$C2$	1.7	5.1	4.2	-43	243
$h = 0.01$					
RKG	-0.25	-0.54	-0.6	2.8	640
$Adams - P$	0.028	-0.51	-4.8	-31	169
$C1$	0.210	-0.45	-5.2	-32	326
$C2$	0.640	1.8	1.4	-9.8	483
$Milne - P$	-0.17	-0.42	-1.74	-8	169
$C1$	-0.02	-0.38	-2.18	-10	326
$C2$	0.13	0.34	0.46	-1.4	484

soluţii aproximative cu precizie impusă. Pentru cazul (a) s-a impus o precizie relativă de 10^{-6}, iar pentru cazul (b), o precizie absolută de 10^{-6}. Pentru metodele pas cu pas, Runge-Kutta ($RK4$), Kutta-Merson (KM) şi Runge-Kutta-Fehlberg (RKF), s-a folosit integrarea adaptivă descrisă la 9.2.4, în timp ce pentru formulele predictor-corector *Adams* şi *Milne* s-a integrat cu pas constant. Pentru primul caz, dintre metodele unipas metoda Runge-Kutta s-a dovedit mai eficientă, în timp ce pentru cazul al doilea mai eficientă este metoda Runge-Kutta-Fehlberg. Nu putem trage concluzii definitive pe baza exemplelor considerate, pe de o parte pentru că doar două exemple reprezintă prea puţină informaţie pentru a generaliza, iar pe de altă parte pentru că, efortul de calcul de la metodele adaptive depinde uneori şi de o alegere "norocoasă" a pasului de plecare, dacă micşorarea pasului se face prin înjumătăţire. Aplicarea relaţiei (9.29) este cu atât mai eficientă cu cât derivata $\Phi^{(4)} \equiv y^{(6)}$ are variaţii cât mai mici. Aceeaşi observaţiei este valabilă şi în ceea ce priveşte estimarea pasului optim de integrare dat de (9.32), unde este esenţială folosirea unei valori pentru factorul de siguranţă de cel mult 0.9.

Pentru cazul (b) ordinul metodelor îşi spune cuvântul, efortul de calcul fiind sensibil

egal pentru toate metodele folosite. Pentru cazul (a) rezultate mai bune se obţin cu metodele multipas. Rezultatele mai bune ale formulei Milne le punem pe seama preciziei superioare a formulei corector faţă de formula corector Adams. Rezultatele mai slabe date de formulele Runge-Kutta-Merson şi Runge-Kutta-Fehlberg se datoresc în principal aproximărilor ce intervin în estimarea erorii, vezi (9.36).

În concluzie, ori de câte ori timpul de calcul nu costituie o problemă şi nu avem informaţii suplimentare privind soluţia unui sistem de ecuaţii diferenţiale, se recomandă metoda Runge-Kutta-Gill cu criteriul pasului dublu, ca fiind cea mai sigură cale.

EXEMPLUL 2. Să se rezolve ecuaţia diferenţială $f''' + f \cdot f'' = 0$ cu condiţiile la limite pe intervalul $[0, \infty]$, $f(0) = 0, f'(0) = 0, f'(\infty) = 1$. (Ecuaţia este întâlnită la calculul stratului limită laminar incompresibil pe o placă plană. Este cunoscută sub denumirea de *soluţia lui Blasius*).

Rezolvare. Se scrie sistemul de ecuaţii diferenţiale echivalent, folosind notaţiile $y_1 = f(x)$, $y_2 = f'(x)$, $y_3 = f''(x)$. Sistemul obţinut este $y_1' = y_2$, $y_2' = y_3$, $y_3' = -y_1 y_2$ cu condiţiile la limite $y_1(0) = y_2(0) = 0$, $y_2'(\infty) = 1$. Prin urmare avem de rezolvat o problemă bilocală. Un element suplimentar îl constituie faptul că intervalul pe care se caută soluţia este infinit. În acest caz, putem proceda pe două căi: (a) se face o schimbare de variabilă independentă convenabilă, care transformă intervalul $[0, \infty)$ într-un interval finit (transformarea trebuie să fie o bijecţie); (b) se porneşte calculul cu un anumit pas, urmărindu-se dacă funcţiile y_1, y_2, y_3 au tendinţa să se stabilizeze după o anumită distanţă pe x. Parametrul problemei este $y_3(0)$ despre care se ştie că este pozitiv (din considerente fizice).

S-a plecat cu valoarea $y_3(0) = 0.20$, folosind un pas constant în x ($h = 0.20$) şi s-a integrat până la $x = 6$. S-a obţinut $y_2(6) - 1 = -0.434162$. Pentru $y_3(0) = 0.60$ s-a obţinut $y_2(6) - 1 = 0.177466$, prin urmare s-a realizat o schimbare de semn. În continuare, s-a aplicat metoda coardei. Rezultatele iteraţiilor după parametrul $\lambda = y_3(0)$ şi funcţia scop $U(\lambda) = y_2(6, \lambda) - 1$ sunt date în tabelul 9.6. S-a folosit formula Runge-Kutta de ordinul 4.

Criteriul de oprire a iteraţiilor a fost $|U(\lambda)| < \varepsilon$. La iteraţia a 4-a s-a realizat această condiţie pentru $\varepsilon = 10^{-4}$. Cu $y_3(0) = 0.469625$ obţinut în această a 4-a iteraţie, s-au calculat funcţiile y_1, y_2 şi y_3 pentru diferite valori ale lui x (pasul $h = 0.20$), ale căror valori sunt date în tabelul 9.7.

Din tabelul 9.7 se observă variaţia lentă a funcţiilor y_2 şi y_3 pentru $x \geq 3$, ceea ce justifică oprirea calculului la $x = 6$ (în loc de $x \to \infty$). Creşterea uşoară a funcţiei y_2 peste valoarea 1 de la $x = 5.4$ la $x = 6.0$, sugerează o eventuală necesitate de creştere a preciziei. Astfel, s-a mai efectuat o iteraţie (a cincea) pentru obţinerea unei valori mai precise pentru λ. Din tabelul 9.6 se obţine valoarea $y_3(0) = 0.469604$ ($\varepsilon = 10^{-5}$. Cu această valoare îmbunătăţită rezultă $y_2(5.6) = 0.9999926$ şi $y_2(6.0) = 1.000005$, precizie care poate fi considerată cu totul satisfăcătoare.

Tabelul 9.4: $a = 13$, $m = 1$, $k = 0$, erori absolute $\times 10^6$

$Metoda$	$x = 0.4$	$x = 0.8$	$x = 1.2$	$x = 1.6$	$nr.F$
$h = 0.2$					
RKG	-67	-112	-106	-38	32
$Adams - P$	-67	-112	-941	-190	17
$C1$			210	476	22
$C2$			54	52	27
$Milne - P$			-900	-481	17
$C1$			13	139	22
$extrapolare$			-18	118	22
δ			-1	0.8	22
$C2$			-61	-42	27
$h = 0.1$					
RKG	-4	-6	-6	-1	64
$Adams - P$		-30	-5	13	25
$C1$		11	22	23	38
$C2$		7	10	2	51
$Milne - P$	-4	-36	-20	-2	25
$C1$		0.4	5	7	38
$extrapolare$		-0.9	4	7	38
δ		-0.09	0.3	0.5	38
$C2$		-1	-0.2	-1.3	51
$h = 0.05$					
RKG	-0.3	-0.4	-0.3	0.03	128
$Adams - P$	-1	-0.1	0.6	0.7	41
$C1$	0.3	1	1	0.8	70
$C2$	0.3	0.8	0.8	0.03	99
$Milne - P$	-1	-0.9	-0.3	0.1	41
$C1$	-0.03	0.2	0.3	0.3	70
$extrapolare$	-0.08	0.1	0.1	0.3	70
δ	-0.01	0.01	0.02	0.02	70
$C2$	-0.05	0.08	0.09	-0.05	99

Tabelul 9.5: Numărul de evaluări ale funcţiei pantă

$Metoda$	$RK4$	KM	RKF	$Adams$	$Milne$
(a)	1812	3800	3400	2142	1608
(b)	84	40	36	48	48

Tabelul 9.6: Exemplul 2

Iter	λ_1	λ_2	$U(\lambda_1)$	$U(\lambda_2)$
0	0.20	0.600000	-0.434162	0.177466
1	0.20	0.483939	-0.434162	0.020252
2	0.20	0.471284	-0.434162	0.002388
3	0.20	0.469800	-0.434162	$2.8267 \cdot 10^{-4}$
4	0.20	0.469625	-0.434162	$3.3475 \cdot 10^{-5}$
5	0.20	0.469604	-0.434162	$3.9646 \cdot 10^{-6}$

Tabelul 9.7: Funcţiile y_1, y_2, y_3

x	y_1	y_2	y_3
0.0	0.0	0.0	0.469625
0.6	0.084392	0.280590	0.461758
1.2	0.333678	0.545273	0.410585
1.8	0.728912	0.761092	0.300455
2.4	1.231590	0.901101	0.167563
3.0	1.795654	0.969088	0.067715
3.6	2.385699	0.992919	0.019338
4.2	2.983684	0.998850	0.003870
4.8	3.583403	0.999893	0.000543
5.4	4.183392	1.000022	0.000054
6.0	1.783410	1.000034	$3.7475 \cdot 10^{-6}$

Capitolul 10

Ecuaţii diferenţiale cu derivate parţiale

Ecuaţiile diferenţiale cu derivate parţiale (sau pe scurt, *ecuaţii cu derivate parţiale - EDP*) sunt întâlnite în mod inevitabil atunci când se modelează fenomene reale, a căror complexitate implică, în general, determinarea unor funcţii de două sau mai multe variabile independente. Într-adevăr, o descriere mai apropiată de realitate a fenomenelor care se desfăşoară în spaţiu comportă cel puţin două variabile independente, iar urmărirea desfăşurării lor poate necesita introducerea unei noi variabile independente, *timpul*.

Diversitatea ecuaţiilor şi sistemelor de ecuaţii diferenţiale este extraordinară, iar condiţiile la limite şi iniţiale (al căror rol este deosebit de important) nu fac decât să adauge noi cazuri şi tipuri de probleme de rezolvat. Nu este deci de mirare că în domeniul *EDP* (în special al ecuaţiilor neliniare) teoria matematică nu este suficient de elaborată. Din aceste motive, abordarea numerică, deşi capabilă, în principiu, să furnizeze date noi, nu poate fi în măsură "să ţină loc şi de teorie", recomandându-se o extremă prudenţă în validarea rezultatelor

obţinute în cazurile neacoperite suficient de teorie. Reconfirmarea rezultatelor obţinute pe diverse căi, suficient de distincte între ele, este o metodă utilă, poate singura în anumite situaţii extreme, dar trebuie luată cu titlu provizoriu, testarea ulterioară rămânând deschisă.

Ecuaţiile cu derivate parţiale pot fi clasificate în funcţie de mai multe criterii; după ordinul derivatelor parţiale, se clasifică în ecuaţii de ordinul întâi, ordinul al doilea, ordinul n; după caracterul de liniaritate se împart în ecuaţii liniare, cvasi-liniare şi neliniare; după tipul influenţei domeniului de integrare asupra soluţiei într-un punct, se împart în ecuaţii eliptice, parabolice şi hiperbolic; după forma condiţiilor la limită întâlnim probleme Dirichlet, Neuman şi mixt.

În cele ce urmează, vom aborda unele tipuri de *EDP* mai uzuale, cu condiţii iniţiale şi la limite pentru care teoria asigură existenţa şi unicitatea soluţiilor.

10.1 Ecuaţii cu derivate parţiale de ordinul I

Ecuaţia cu derivate parţiale de ordinul întâi se scrie sub forma

$$\sum_{i=1}^{n} A_i(x_1, \dots, x_n, u) \frac{\partial u}{\partial x_i} = B(x_1, \dots, x_n, u) \ , \tag{10.1}$$

în care u este *funcţia necunoscută*, x_i, $i \in \overline{1,n}$ *variabilele independente*, iar funcţiile A_i, $i \in \overline{1,n}$ şi B depind cel mult de funcţia u (nu şi de derivatele parţiale $\partial u/\partial x_i$). Dacă A_i şi B nu depind de funcţia u ecuaţia se numeşte *liniară*; dacă $B \equiv 0$, ecuaţia se numeşte *omogenă*.

Rezolvarea ecuaţiei cu derivate parţiale (10.1) se reduce la rezolvarea unui sistem de ecuaţii diferenţiale ordinare numit *sistem caracteristic*

$$\frac{dx_1}{A_1(x_1, \dots, x_n, u)} = \dots = \frac{dx_n}{A_n(x_1, \dots, x_n, u)} = \frac{du}{B(x_1, \dots, x_n, u)} \ . \tag{10.2}$$

Soluţia ecuaţiei (10.1) este o suprafaţă $n-$dimensională într-un domeniu $\Omega^{n+1} \subset \mathbf{R}^{n+1}$, de forma $F(x_1, \dots, x_n, u) = 0$ sau $u = f(x_1, \dots, x_n)$ care verifică ecuaţia (10.1) şi anumite condiţii de selecţie. Problema este studiată complet pentru cazul când suprafaţa integrală $n-$dimensională trece printr-o suprafaţă $(n-1)-$dimensională Γ dată în domeniul $(n+1)-$dimensional (problemă Cauchy). Suprafaţa Γ poate fi dată sub forma intersecţiei a două suprafeţe $n-$dimensionale

$$\left.\begin{array}{l} F_1(x_1, \dots, x_n, u) = 0 \\ F_2(x_1, \dots, x_n, u) = 0 \end{array}\right\} \ (\Gamma) \ . \tag{10.3}$$

În principiu, soluţia generală a sistemului de ecuaţii diferenţiale (10.2), sistem de ordinul n, depinde de n constante arbitrare C_i, $i \in \overline{1,n}$ şi se poate scrie sub forma

$$\varphi_i(x_1, \dots, x_n, u) = C_i, \ i \in \overline{1,n} \ . \tag{10.4}$$

Funcţiile $\varphi_i(x_1, \ldots, x_n, u)$ sunt determinate de forma sistemului caracteristic. Suprafeţele $\varphi_i = C_i$ se numesc *suprafeţe caracteristice*, iar intersecţiile lor depinzând de un singur parametru, se numesc *linii caracteristice*.

Condiţiile (10.3), împreună cu (10.4) formează un sistem de $n+2$ ecuaţii din care, în principiu, prin exprimarea celor $n+1$ variabile $x_1, \ldots, x_n, u$ în funcţie de C_i, $i \in \overline{1,n}$ şi introducerea lor în a $(n+2)$-a ecuaţie rămasă, se obţine o relaţie între parametrii C_i, sub forma

$$\Phi(C_1, \ldots, C_n) = 0 \tag{10.5}$$

care, prin (10.4), dă tocmai soluţia $F(x_1, \ldots, x_n, u) = 0$ a problemei

$$\Phi(C_1, \ldots, C_n) = \Phi(\varphi_1, \ldots, \varphi_n) \equiv F(x_1, \ldots, x_n, u) = 0 \ . \tag{10.6}$$

În calculul numeric, soluţia se caută într-un anumit volum $(n+1)-$dimensional Ω^{n+1} care conţine suprafaţa Γ dată de (10.3). Se alege o diviziune convenabilă a suprafeţei Γ. Valorile în noduri $(x_{10}^{(k)}, \ldots, x_{n0}^{(k)}, u_0^{(k)}) \in \Gamma$, $k = 1, 2, \ldots, N$, sunt condiţii iniţiale pentru sistemul de ecuaţii diferenţiale (10.2), ale cărui soluţii sunt liniile de câmp ale vectorului de componente $(A_i,\ i \in \overline{1,n},\ B)$. Se obţin N linii de câmp ale suprafeţei integrale.

Dacă $B \equiv 0$ (ecuaţie *omogenă*), sistemul (10.2) se simplifică deoarece o integrală primă este $u = const$. Rămâne de rezolvat sistemul de ecuaţii diferenţiale

$$\frac{dx_1}{A_1(x_1, \ldots, x_n, u_0)} = \ldots = \frac{dx_n}{A_n(x_1, \ldots, x_n, u_0)} \ , \tag{10.7}$$

u_0 fiind dată de condiţia iniţială (10.3). Din (10.7) se obţin liniile de câmp ale unui vector de componente A_i, $i \in \overline{1,n}$ pe care $u = const$.

Scheme explicite.

O primă etapă în rezolvarea numerică a unei ecuaţii cu derivate parţiale o constituie *discretizarea*, care constă pe de o parte în îdivizarea domeniului cu ajutorul unei reţele de calcul, iar pe de altă parte în înlocuirea ecuaţiei cu derivate parţiale cu o ecuaţie mai simplă. Există mai multe metode pentru înlocuirea ecuaţiei cu derivate parţiale: metode cu diferenţe finite, metode cu volume finite, metode cu elemente finite, metode spectrale. În acest capitol, vom prezenta elementele de bază pentru rezolvarea numerică a ecuaţiilor cu derivate parţiale folosind metode cu diferenţe finite.

Considerăm ecuaţia unidimensională a propagării undelor

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0,\ x \in [0,1],\ t \in [0,T] \ , \tag{10.8}$$

unde a este o constantă pozitivă. Am folosit notaţia $u_i^n \equiv u(x_i, t_n)$. Pentru a rezolva ecuaţia (10.8), sunt necesare *condiţii iniţiale* de forma

$$u(x, 0) = f(x) \ . \tag{10.9}$$

Figura 10.1: Reţeaua de calcul pentru ecuaţia (10.8).

În multe cazuri sunt date şi condiţii la limită

$$u(0,t) = g_0(t),\ u(1,t) = g_1(t)\ , \tag{10.10}$$

funcţiile f, g_0 şi g_1 fiind date. Pentru a rezolva numeric ecuaţia (10.8), se împarte domeniul dreptunghiular de dimensiuni 1 şi T cu ajutorul unei reţele cu paşi egali, pentru simplitate, h pe direcţia Ox şi k pe direcţia Ot, ca în figura 10.1. Una din modalităţile de a obţine o ecuaţie cu diferenţe finite este de a folosi dezvoltarea în serie Taylor a funcţiei $u(x,t)$ în jurul punctului (x_i, t_n)

$$\begin{aligned} u(x_{i-1},t_n) &= u(x_i,t_n) - h\frac{\partial u(x_i,t_n)}{\partial x} + O(h^2), \\ u(x_i,t_{n+1}) &= u(x_i,t_n) + k\frac{\partial u(x_i,t_n)}{\partial t} + O(k^2)\ , \end{aligned} \tag{10.11}$$

unde $x_i = ih,\ i \in \overline{0,I},\ t_n = nk,\ n \in \overline{0,N},\ h = 1/I,\ k = T/N$. Rezultă

$$\frac{\partial u(x_i,t_n)}{\partial x} \equiv \left.\frac{\partial u}{\partial x}\right|_i^n = \frac{u_i^n - u_{i-1}^n}{h} + O(h),\ \left.\frac{\partial u}{\partial t}\right|_i^n = \frac{u_i^{n+1} - u_i^n}{k} + O(k)\ . \tag{10.12}$$

Neglijând termenii $O(k)$ şi $O(h)$, rezultă ecuaţia în diferenţe finite

$$v_i^{n+1} = v_i^n + c(v_{i-1}^n - v_i^n),\ i \in \overline{1,I},\ n \in \overline{0,N}\ , \tag{10.13}$$

unde cu v_i^n s-a notat valoarea aproximativă a funcţiei $u(ih,nk)$, iar $c = ak/h$ este cunoscut sub denumirea de *număr Courant*[1]. Relaţia (10.13) este o schemă *explicită* cu diferenţe finite, deoarece valorile v_i^{n+1} se determină direct, folosind numai valorile v_i^n de la momentul de timp anterior. Valorile v_i^n sunt aproximative deoarece din dezvoltările (10.11) am folosit numai termenii de ordinul

Figura 10.2: Rezultate obţinute cu schema explicită (10.13).

întâi. Spunem că formulele (10.12) au ordinul întâi de precizie. Observăm cum pentru $c = 1$ se obţine soluţia exactă $v_i^{n+1} = v_{i-1}^n$.

EXEMPLU. Vom rezolva ecuaţia

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \tag{10.14}$$

cu condiţiile iniţiale şi la limită

$$u(x,0) = 0,\ 0 < x \leq 1,\ u(0,t) = 1,\ t \geq 0\ . \tag{10.15}$$

La momentul iniţial de timp $t = 0$ funcţia necunoscută u este nulă pentru toate valorile x din domeniul de calcul, mai puţin valoarea $x = 0$ unde există o perturbaţie $u = 1$. Această perturbaţie se va propaga în timp şi spaţiu. Rezultatele obţinute cu schema explicită (10.13) sunt date în figura 10.2. Se observă diferenţe mari între rezultatele obţinte cu reţele diferite. Deşi ar fi fost de aşteptat ca cele mai bune rezultate să fie obţinute pentru $k = 0.01$, acestea se obţin pentru $h = 0.1$ ($k/h = 1$). Rezultate slabe sunt obţinute pentru $k = 0.1$, $h = 0.01$, care diferă mult de cele obţinute cu $k = 0.1$, $h = 0.1$. Deşi pasul mai mic $h = 0.01$ pe direcţia Ox însemnă o aproximare mai bună a derivatei spaţiale şi ne-ar îndreptăţi să sperăm la rezultate mai bune, acest lucru nu se întâmplă, ba din contra, rezultatele nu mai au nici o semnificaţie fizică. Aşa cum vom vedea mai departe, acest lucru se datoreşte faptului că schema (10.13) este instabilă pentru valori $k/h > 1$.

După cum am văzut din exemplul precedent, folosirea unei reţele foarte fine (altfel spus, norma reţelei este foarte mică) nu este întotdeauna suficientă.

[1]După matematicianul Richard Courant (1988-1972) care a avut o contribuţie importantă în domeniul metodelor numerice pentru rezolvarea ecuaţiilor neliniare cu derivate parţiale.

Reţeaua de calcul trebuie de multe ori să îndeplinească anumite condiţii pentru ca o schemă cu diferenţe finite să fie convergentă.

Definiţie. O metodă cu diferenţe finite este *convergentă*, dacă soluţia obţinută cu ajutorul ecuaţiei cu diferenţe converge către soluţia exactă atunci când norma reţelei tinde la zero.

Ţinând cont de (10.2), caracteristicile ecuaţiei (10.8) sunt date de

$$dt = dx/a \ , \tag{10.16}$$

adică drepte de pantă $dt/dx = 1/a$. Condiţia de convergenţă pentru schema explicită (10.13) este este dată de *criteriul CFL* (*Courant-Friederichs-Lewy*) care impune ca *domeniul de influenţă numerică* să includă *domeniul de influenţă fizică* (fig.10.1; vezi şi 10.2.3, fig.10.15). Domeniul de influenţă a punctului B este format din totalitatea punctelor care primesc informaţie din B. Domeniul de influenţă fizică este mărginit la dreapta de BD, în timp ce domeniul de influenţă numerică este mărginit la dreapta de BA. Schema explicită (10.13) este convergentă deci dacă reţeaua îndeplineşte condiţia

$$0 < c \equiv ak/h \leq 1 \ . \tag{10.17}$$

Dacă condiţia (10.17) nu este îndeplinită, schema (10.13) nu este convergentă deoarece nu este *stabilă* (vezi par. 43).

Dacă pentru aproximarea derivatei spaţiale în locul diferenţei la stânga (10.12) folosim diferenţe la dreapta

$$\left.\frac{\partial u}{\partial x}\right|_i^n = \frac{u_{i+1}^n - u_i^n}{h} + O(h), \ \left.\frac{\partial u}{\partial t}\right|_i^n = \frac{u_i^{n+1} - u_i^n}{k} + O(k) \ , \tag{10.18}$$

obţinem

$$v_i^{n+1} = v_i^n + c(v_i^n - v_{i+1}^n) \ . \tag{10.19}$$

Utilizând schema (10.19) pentru rezolvarea ecuaţiei (10.14), valorile v_i^{n+1} pentru toate momentele de timp rămân egale cu valorile iniţiale v_i^0, deoarece schema (10.19) nu permite propagarea informaţiei de la stânga la dreapta, ci numai de la dreapta la stânga. Schema (10.19) ar putea fi folosită în cazul $c < 0$, caz în care schema (10.12) nu mai este adecvată.

Putem încerca o aproximare mai bună pentru derivata spaţială folosind diferenţe centrate

$$\left.\frac{\partial u}{\partial x}\right|_i^n = \frac{u_{i+1}^n - u_{i-1}^n}{2h} + O(h^2) \ , \tag{10.20}$$

obţinându-se

$$v_i^{n+1} = v_i^n + (c/2)(v_{i-1}^n - v_{i+1}^n) \ , \tag{10.21}$$

Figura 10.3: Rezultate obţinute pentru ecuaţia (10.14), cu schemele explicite (10.21) şi (10.22) pentru $t = 0.5$.

schemă care este instabilă pentru orice valoare a numărului Courant c. Observăm cum un *ordin mai mare în aproximarea derivatelor parţiale, nu înseamnă neapărat precizie mai bună.*

O schemă explicită des întâlnită este schema *Lax-Wendroff* dată de

$$v_i^{n+1} = (1-c^2)v_i^n - \tfrac{c}{2}(1-c)v_{i+1}^n + \tfrac{c}{2}(1+c)v_{i-1}^n \ , \tag{10.22}$$

care are ordinul de precizie $O(h^2)$. Pentru $c = 1$ se obţine soluţia exactă $v_i^{n+1} = v_{i-1}^n$, la fel ca la schema (10.13).

Exemplu. În figura (10.3) sunt date rezultatele obţinute la momentul de timp $t = 0.5$ pentru ecuaţia (10.14) rezolvată cu schemele explicite (10.21) şi (10.22). Rezultatele obţinute cu schema (10.21) sunt proaste chiar şi pentru $h = k = 0.05$, în timp ce rezultate foarte bune se obţin cu schema (10.22) pentru $h = k = 0.05$, acceptabile pentru $h = 0.05$,$k = 0.01$, şi proaste pentru $h = 0.05$, $k = 0.1$ deoarece schema este instabilă.

Scheme implicite.

Convergenţa condiţionată este o caracteristică a schemelor explicite şi impune adeseori paşi de timp foarte mici. Pentru a evita acest neajuns, se folosesc schemele implicite, în care derivatele spaţiale se aproximează folosind valorile aproximative v_i nu la momentul n, ci la momentul $n+1$,

$$\left.\frac{\partial u}{\partial x}\right|_i^{n+1} = \frac{u_{i+1}^{n+1} - u_i^{n+1}}{h} + O(h) \ . \tag{10.23}$$

Figura 10.4: Rezultate obţinute pentru ecuaţia (10.14), cu schemele implicite (10.24) şi (10.25) pentru $t = 0.5$.

Se obţin ecuaţiile cu diferenţe finite

$$v_i^{n+1} = (cv_{i-1}^{n+1} + v_i^n)/(1+c),\ i = 1, 2, \dots\ . \tag{10.24}$$

Schema (10.24) este necondiţionat convergentă.

O altă schemă implicită este cea atribuită lui *Wendroff*, dată de relaţia

$$v_i^{n+1} = v_{i-1}^n + \tfrac{1-c}{1+c}(v_i^n - v_{i-1}^n), \tag{10.25}$$

care este de asemenea necondiţionat convergentă.

Exemplu. În figura (10.4) sunt date rezultatele obţinute la momentul de timp $t = 0.5$ pentru ecuaţia (10.14) rezolvată cu schemele implicite (10.24) şi (10.25). Amândouă schemele sunt convergente pentru toate valorile h şi k. Rezultate foarte bune s-au obţinut cu schema (10.25) pentru $h = k = 0.05$ şi acceptabile în rest.

10.2 Ecuaţii cu derivate parţiale de ordinul II

Vom considera ecuaţia cu derivate parţiale de ordinul doi *cvasiliniară* de forma

$$\sum_{i=1}^{n} A_i \frac{\partial^2 u}{\partial x_i^2} + \sum_{i=1}^{n} B_i \frac{\partial u}{\partial x_i} + C = 0\ , \tag{10.26}$$

cu A_i, B_i, $C = f(x_1, \dots, x_n, u)$. Forma (10.26) nu conţine derivate mixte (este o *formă canonică*) şi se poate obţine prin schimbări de variabile adecvate, după cum se va putea vedea în continuare. Ecuaţia (10.26) este *de tip*

(*a*) *eliptic*, dacă toţi coeficienţii $A_i(x_1, \dots, x_n, u)$, $i \in \overline{1,n}$ au acelaşi semn;

(*b*) *parabolic*, dacă există un indice j astfel încât $A_j = 0$, $A_i \neq 0\,(i \neq j)$ şi $B_j \neq 0$;

(*c*) *hiperbolic*, dacă toţi coeficienţii A_i au acelaşi semn, cu excepţia unuia care are semn opus.

Această clasificare este importantă, deoarece este legată de modul în care un punct din domeniu este influenţat de (comunică cu) punctele din vecinătate.

În cazul ecuaţiei de tip *eliptic*, punct este influenţat de toate punctele din orice vecinătate (disc, bulă) a lui. Un exemplu tipic de ecuaţie de tip eliptic, este *ecuaţia lui Laplace*

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0 \ . \tag{10.27}$$

Datorită influenţei reciproce o problemă de tip eliptic nu se poate rezolva numeric decât simultan pentru toate punctele din domeniu. Condiţiile la limită în acest caz se impun pe frontiere închise.

În cazul ecuaţiei de tip *parabolic* există posibilitatea unui marş în calculul numeric, în direcţia x_j pentru care $A_j = 0$. Ecuaţia se scrie sub forma

$$B_j \frac{\partial u}{\partial x_j} = F(x_1, \dots, x_n, u, \frac{\partial u}{\partial x_i}, \frac{\partial^2 u}{\partial x_i^2}),\ i \in \overline{1,n},\ i \neq j \ , \tag{10.28}$$

iar problema se rezolvă simultan numai pentru punctele situate pe suprafeţele $x_j = const$, nu pentru toate nodurile din domeniu, ceea ce simplifică esenţial calculul. Problemele cu ecuaţii de tip parabolic sunt comparativ mai simple (la acelaşi număr de variabile) decât cele cu ecuaţii de tip eliptic. Exemplu tipic de ecuaţie de tip *parabolic*, este *ecuaţia de transmitere a căldurii în regim nestaţionar*

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \ , \tag{10.29}$$

t fiind timpul. Ecuaţia (10.29) în trei variabile independente (x, y, t) este, în general, mai uşor de rezolvat numeric decât ecuaţia (10.27) în variabilele (x, y, z).

În cazul ecuaţiei de tip *hiperbolic*, există puncte care nu se pot influenţa reciproc. Un exemplu îl reprezintă mişcările supersonice, unde perturbaţiile mici sunt limitate la interiorul unui con, denumit *con Mach*. Prin urmare, în rezolvarea numerică trebuie evitată comunicarea numerică a nodurilor care nu comunică fizic între ele. Un exemplu tipic de ecuaţie de tip hiperbolic este *ecuaţia undelor*

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2} \ , \tag{10.30}$$

a fiind viteza de propagare a undei ($a = const$). În cazul ecuaţiilor *hiperbolice* există mai multe *direcţii caracteristice distincte* de-a lungul cărora se poate

Figura 10.5: Domeniul de calcul.

avansa plecând de la o stare iniţială. Problemele pot include însă, pe lângă *condiţii iniţiale*, şi *condiţii la limite* caz în care soluţiile sunt esenţial diferite.

Pentru exemplificare considerăm ecuaţia

$$a\,u_{xx} + b\,u_{xy} + c\,u_{yy} = f \ , \tag{10.31}$$

unde $u_x \equiv \partial u/\partial x$, $u_y \equiv \partial u/\partial y$, $u_{xx} \equiv \partial^2 u/\partial x^2$, $u_{xy} \equiv \partial^2 u/\partial x \partial y$, $u_{yy} \equiv \partial^2 u/\partial y^2$, iar a, b, c şi f sunt funcţii de x, y şi u. Se cunosc valorile funcţiei u şi ale derivatelor u_x şi u_y pe curba Γ şi sub aceasta (figura 10.5). Se pune problema dacă aceste valori cunoscute sunt suficiente pentru a putea determina valorile derivatelor de ordinul al doilea, u_{xx}, u_{xy} şi u_{yy}. Vom scrie

$$\begin{aligned} d(u_x) &= u_{xx}dx + u_{xy}dy \\ d(u_y) &= u_{yx}dx + u_{yy}dy \ . \end{aligned} \tag{10.32}$$

Ecuaţiile (10.31) şi (10.32) formează sistemul

$$\begin{bmatrix} a & b & c \\ dx & dy & 0 \\ 0 & dx & dy \end{bmatrix} \begin{bmatrix} u_{xx} \\ u_{xy} \\ u_{yy} \end{bmatrix} = \begin{bmatrix} f \\ d(u_x) \\ d(u_y) \end{bmatrix} \ . \tag{10.33}$$

Soluţia sistemului (10.33) există şi este unică dacă determinantul matricei sistemului este nenul, adică

$$a(dy)^2 - b\,dx\,dy + c(dx)^2 \neq 0 \ . \tag{10.34}$$

Dacă curba Γ are panta $m = dy/dx$ astfel încât $a\,m^2 - b\,m + c = 0$, atunci sistemul (10.33) este nedeterminat, necunoscutele u_{xx}, u_{xy}, u_{yy} neputând fi determinate în mod unic.

Ecuaţia (10.34) poate avea:

(*a*) două rădăcini reale distincte, $m_1 \neq m_2$, dacă $b^2 - 4\,a\,c > 0$; în acest caz ecuaţia este de tip *hiperbolic*;

(*b*) două rădăcini reale confundate, $m_1 = m_2$, dacă $b^2 - 4\,a\,c = 0$; ecuaţia este de tip *parabolic*;

(*c*) două rădăcini complex conjugate, dacă $b^2 - 4\,a\,c < 0$; ecuaţia este de tip *eliptic*.

Nedeterminarea care se obţine în cazul în care Γ este o curbă caracteristică a unei ecuaţii de tip hiperbolic sugerează ideea că este posibil ca pe această curbă ecuaţia (10.31) să poată fi înlocuită cu o ecuaţie mai simplă. După cum vom vedea la 10.2.4, ecuaţia cu derivate parţiale de ordinul al doilea se poate înlocui pe o curbă caracteristică cu o ecuaţie diferenţială ordinară, mult mai simplu de rezolvat numeric.

10.2.1 Ecuaţii cu derivate parţiale de tip parabolic

Vom considera cea mai simplă ecuaţie de tip parabolic sub forma

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}\ . \tag{10.35}$$

Ecuaţia (10.35) este un caz particular al ecuaţiei (10.31), cu $y \equiv t$, $a = 1$, $b = 0$, $c = 0$, $f = 0$. Avem $b^2 - 4ac = 0$, deci ecuaţia este de tip parabolic.

Exemplu. Ecuaţia (10.35) se întâlneşte în transmiterea căldurii în regim nestaţionar. Absenţa constantelor fizice în (10.35) se explică prin faptul că s-au introdus variabile adimensionalizate: u, temperatura adimensionalizată, t, x timpul şi coordonata spaţială, de asemenea adimensionalizate. Transferul căldurii printr-o bară în care una din dimensiuni este mult mai mare decât celelelalte două şi deci fenomenul se poate considera unidimensional, în regim nestaţionar, este descris de ecuaţia

$$\frac{\rho c_p}{\lambda}\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2}\ , \tag{10.36}$$

unde ρ este densitatea materialului în kg/m^3, c_p căldura specifică în $j/kg \cdot K$, iar λ este conductivitatea termică, $j/m \cdot grad$. Ecuaţia (10.36) are condiţiile iniţiale şi la limite

$$\begin{gathered} T(x,0) = F(x),\ x \in [0,L] \\ T(0,t) = G_0(t),\ T(L,t) = G_1(t),\ t \geq 0\ . \end{gathered} \tag{10.37}$$

Din punct de vedere fizic, $F(x)$ reprezintă distribuţia de temperatură într-o bară de lungime L, la momentul iniţial $t = 0$, iar $G_0(t)$ şi $G_1(t)$ temperaturile la care se menţin capetele barei, spre exemplu, ca urmare a contactului cu mediul ambiant sau cu alte corpuri. În locul temperaturii date, se poate impune fluxul de temperatură, adică derivata $\partial T/\partial x$, de regulă nul. Funcţiile G_0 şi G_1 sunt, în cazul general, funcţii de timp, în timp ce funcţia $F(x)$ este în general funcţie de coordonata spaţială x. Vom lua în considerare pentru simplitate în cele ce urmează cazul în care funcţiile G_0 şi G_1 sunt constante, $G_0 = T_0$, $G_1 = T_1$.

Se introduc coordonata adimensionalizată $\bar{x}$ şi temperaturatura adimensionalizată u prin relaţiile

$$\bar{x} = x/L,\ u = (T - T_1)/(T_0 - T_1)\ . \tag{10.38}$$

Ecuaţia (10.36) devine

$$\frac{\rho c_p}{\lambda} L^2 \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial \bar{x}^2}\ . \tag{10.39}$$

Variabila adimensionalizată pentru timp este sugerată de coeficientul derivatei $\partial u/\partial t$. Folosirea unei variabile adimensionale pentru timp este posibilă numai dacă $\rho\, c_p/\lambda = const.$

$$\bar{t} = t/[(\rho c_p/\lambda)L^2]\ . \tag{10.40}$$

Adimensionalizarea este recomandată deoarece conduce în general la reducerea numărului de parametrii, iar valorile variabilelor dependente şi a funcţiilor necunoscute au acelaşi ordin de mărime.

Se obţine în final ecuaţia (10.35). În continuare renunţăm la notaţia cu bară pentru variabilele adimensionalizate.

Ecuaţia (10.35) se rezolvă pentru următoarele condiţii:

(a) *condiţii iniţiale* ($t = 0$)

$$u(x, 0) = f(x),\ x \in [0, 1]\ ; \tag{10.41}$$

(b) *condiţii la limite* ($x = 0;\ x = 1$)

$$u(0, t) = g_0(t),\ u(1, t) = g_1(t),\ t \geq 0\ . \tag{10.42}$$

În relaţiile (10.41) şi (10.42), $f(x)$, $g_0(t)$ şi $g_1(t)$ sunt funcţii date.

Metoda de rezolvare numerică constă în împărţirea domeniului dreptunghiular (figura 10.6) de dimensiuni 1 şi T (durata pe care urmărim desfăşurarea fenomenului) în subdiviziuni prin noduri cu paşi egali[2], h în direcţia x şi k în direcţia t

$$h = 1/I,\ k = T/N\ , \tag{10.43}$$

I şi N fiind întregi suficient de mari. Un nod oarecare de coordonate $(ih,\ nk)$ va fi notat cu doi indici (i, n), $i \in \overline{0, I}$, $n \in \overline{0, N}$. Pentru variabila timp vom utiliza indici superiori.

Ecuaţia (10.35) se va scrie în noduri, aproximând derivatele parţiale cu diferenţe finite. Această operaţie se numeşte *discretizare* (cu diferenţe finite) deoarece valorile funcţiei se vor calcula doar în noduri, adică pe o mulţime discretă de puncte.

Deoarece calculul numeric ne va furniza doar valori aproximative pentru funcţia u, vom scrie

$$u = v + w\ , \tag{10.44}$$

v reprezentând valorile aproximative ale funcţiei u, iar w eroarea.

[2]Folosirea diviziunilor echidistante duce la simplificarea relaţiilor ulterioare, nefiind obligatorie.

Figura 10.6: Discretizarea ecuaţiilor parabolice.

Scheme explicite.

Aproximând derivatele parţiale cu diferenţe finite sub forma[3]

$$(v_i^{n+1} - v_i^n)/k = (v_{i-1}^n - 2v_i^n + v_{i+1}^n)/h^2 \ , \tag{10.45}$$

se obţine o expresie *explicită* pentru valoarea funcţiei la momentul de timp $n+1$ în funcţie de valorile la momentul anterior n

$$v_i^{n+1} = \lambda v_{i-1}^n + (1-2\lambda)v_i^n + \lambda v_{i+1}^n \ , \tag{10.46}$$

unde am notat cu λ *parametrul reţelei*

$$\lambda = k/h^2 \ . \tag{10.47}$$

Se observă că pentru calculul valorii v_i^{n+1} sunt necesare trei valori la momentul n: v_{i-1}^n, v_i^n şi v_{i+1}^n. Pentru $n = 0$, aceste valori sunt cunoscute din condiţiile iniţiale şi la limite. Într-adevăr, din (10.46), (10.41) şi (10.42) se obţine

$$\begin{gathered} v_i^1 = \lambda f_{i-1} + (1-2\lambda)f_i + \lambda f_{i+1}, \ i \in \overline{1, I-1} \\ v_0^1 = g_0(k); \ v_I^1 = g_1(k); \ v_i^0 = f(x_i) \, . \end{gathered} \tag{10.48}$$

În acest fel, calculul poate continua pas cu pas în direcţia axei timpului. Relaţia (10.46) este explicită deoarece sistemul de $I+1$ ecuaţii care se obţine pentru $i \in \overline{0, I}$ este practic rezolvat. Rămâne de studiat problema *convergenţei* metodei explicite (definiţia convergenţei a fost dată la 41).

În cazul de faţă, trebuie studiată comportarea soluţiei dată de schema explicită (10.45) sau (10.46) cu condiţiile la limite şi iniţiale (10.48).

[3]Aproximarea se face cu ajutorul dezvoltării în serie Taylor, după cum se va vedea în continuare când se va analiza convergenţa schemei.

Într-un punct oarecare (ih, nk), ecuaţia exactă (10.35) se scrie

$$(\partial u/\partial t)_i^n = (\partial^2 u/\partial x^2)_i^n \ . \tag{10.49}$$

Derivatele parţiale din (10.49) se vor exprima cu ajutorul dezvoltărilor în serie Taylor ale valorilor funcţiei $u(x,t)$ în puncte vecine. Astfel,

$$\begin{aligned} u_{i\pm1}^n = u_i^n \pm h\,(\partial u/\partial x)_i^n + \tfrac{h^2}{2}\left(\partial^2 u/\partial x^2\right)_i^n \pm \tfrac{h^3}{6}\left(\partial^3 u/\partial x^3\right)_i^n + \\ + \tfrac{h^4}{24}\left(\partial^4 u/\partial x^4\right)_i^n \pm \tfrac{h^5}{120}\left(\partial^5 u/\partial x^5\right)_i^n + O(h^6)\ , \end{aligned} \tag{10.50}$$

$$u_i^{n+1} = u_i^n + k\,(\partial u/\partial t)_i^n + \tfrac{k^2}{2}\left(\partial^2 u/\partial t^2\right)_i^n + O(k^3)\ , \tag{10.51}$$

de unde rezultă

$$\begin{aligned} (\partial u/\partial t)_i^n &= \tfrac{1}{k}(u_i^{n+1} - u_i^n) - \tfrac{k}{2}\left(\partial^2 u/\partial t^2\right)_i^n + O(k^2)\ , \\ \left(\partial^2 u/\partial x^2\right)_i^n &= \tfrac{1}{h^2}(u_{i-1}^n - 2u_i^n + u_{i+1}^n) - \tfrac{h^2}{12}\left(\partial^4 u/\partial x^4\right)_i^n + O(h^4)\ . \end{aligned} \tag{10.52}$$

S-a presupus că derivatele parţiale ale funcţiei u în raport cu variabilele t, x sunt definite până la ordinele necesare. Cu ajutorul relaţiilor (10.52), ecuaţia (10.49) conduce la sistemul

$$u_i^{n+1} = \lambda u_{i-1}^n + (1-2\lambda)u_i^n + \lambda u_{i+1}^n + z_i^n,\ i \in \overline{1, I-1},\ n \in \overline{0, N}\ . \tag{10.53}$$

Faţă de ecuaţiile (10.46) pentru valorile aproximative în noduri v_i^n, sistemul (10.53) pentru valorile exacte u_i^n conţine reziduul z_i^n având expresia

$$z_i^n/k = (k/2)(\partial^2 u/\partial t^2)_i^n - (h^2/12)(\partial^4 u/\partial x^4)_i^n + O(k^2) + O(h^4)\ . \tag{10.54}$$

Scăzând relaţiile (10.53) şi (10.46), se obţine pentru eroarea w definită de (10.44)

$$w_i^{n+1} = \lambda w_{i-1}^n + (1-2\lambda)w_i^n + \lambda w_{i+1}^n + z_i^n\ . \tag{10.55}$$

Condiţia de convergenţă a schemei cu diferenţe, se poate formula sub forma

$$w_i^n \to 0,\ \text{când } I, N \to \infty\ . \tag{10.56}$$

Pentru a stabili în ce condiţii se realizează (10.56), vom urmări comportarea modulelor erorilor maxime. Din (10.56) se obţine pentru $\lambda > 0$

$$\left|w_i^{n+1}\right| \le \lambda\left|w_{i-1}^n\right| + |1-2\lambda|\,|w_i^n| + \lambda\left|w_{i+1}^n\right| + |z_i^n|\ . \tag{10.57}$$

Notăm cu $w^n = \max\limits_i |w_i^n|$. Inegalitatea (10.57) fiind adevărată pentru orice i, este adevărată şi pentru acel $i_{\max}$ care realizează maximul modulului; valoarea acestui $i_{\max}$ fiind necunoscută, o vom nota tot cu i. În acest fel, din (10.57) se obţine succesiv

$$\begin{aligned} w^{n+1} &\le \lambda\left|w_{i-1}^n\right| + |1-2\lambda|\,|w_i^n| + \lambda\left|w_{i+1}^n\right| + |z_i^n|\,, \\ w^{n+1} &\le (2\lambda + |1-2\lambda|)w^n + z^n,\ \text{unde } z^n = \max_i |z_i^n|\ . \end{aligned} \tag{10.58}$$

Pentru ca eroarea w^n să nu crească de la un moment de timp n la momentul următor $n+1$, este necesar şi suficient ca

$$2\lambda + |1-2\lambda| \leq 1 \Rightarrow 0 < \lambda \leq 1/2 \ . \tag{10.59}$$

Cu condiţia (10.59), se obţine

$$w^{n+1} \leq w^n + z^n, \ n \in \overline{0,N} \ . \tag{10.60}$$

Sumând egalitatea (10.60) termen cu termen de la 0 la $N-1$, şi efectuând reducerile, rezultă

$$w^n \leq w_0 + \sum_{n=0}^{N-1} z^n \leq w_0 + N z_{\max} = w_0 + T\frac{z_{\max}}{k} \ , \tag{10.61}$$

unde w_0 reprezintă eroarea la momentul iniţial $t=0$, iar $z_{\max} = \max\limits_n z^n$. Deoarece problema se rezolvă pentru condiţiile iniţiale şi la limite impuse, eroarea w_0 este nulă ($w_0 = 0$); în orice caz, chiar dacă în condiţiile impuse sunt erori, acestea nu se datorează metodei.

Cu această observaţie şi ţinând cont de (10.54), se obţine

$$\lim_{N\to\infty} w^n \leq 0, \ w^n \to 0, \text{ când } N \to \infty \ , \tag{10.62}$$

adică metoda explicită cu diferenţe finite converge, cu condiţia (10.59). *Convergenţa condiţionată* este o situaţie general întâlnită la metodele *explicite* şi conduce la creşterea volumului de calcul pentru obţinerea unei precizii corespunzătoare. Spre exemplu, pentru $h = 0.01$, sunt necesari cel puţin 5000 de paşi pentru $T = 1$ ($k \leq 5 \cdot 10^{-5}$). În acelaşi timp, este interesant de observat că, pentru k fixat, micşorarea pasului h după direcţia x, poate duce la rezutate eronate dacă nu se respectă condiţia $\lambda \leq 1/2$.

Urmărind (figura 10.6), vom încerca să dăm o explicaţie calitativă a fenomenului de *convergenţă condiţionată* pentru schemele explicite. Astfel, cunoscând valorile funcţiei u pe linia AB, se pot calcula valorile aproximative v_i^n pentru punctele din interiorul triunghiului ABP, fără a utiliza valorile impuse de condiţiile la limite. Altfel spus, punctul P primeşte influenţă (în procesul numeric de calcul) numai de la punctele din interiorul triunghiului ABP, fapt ce ar corespunde fizic unei ecuaţii de tip hiperbolic şi nu parabolic. O schemă cu diferenţe 100% parabolică ar trebui să nu poată determina valoarea funcţiei u într-un punct de pe linia CD fără ca valorile din punctele C şi D să fie luate în calcul. O astfel de comportare o au *schemele implicite* care vor fi prezentate în continuare.

Un aspect interesant al schemelor cu diferenţe finite îl constituie *schemele optimale*. Acestea constau în alegerea parametrului reţelei λ astfel încât eroarea de trunchiere să fie cât mai mică. Pentru exemplificare, vom considera schema explicită (10.46),

$$u_i^{n+1} = \lambda u_{i-1}^n + (1-2\lambda)u_i^n + \lambda u_{i+1}^n + z_i^n \ , \tag{10.63}$$

cu eroarea de trunchiere z_i^n dată de relaţia (10.54). Pentru ecuaţia căldurii (10.35) se arată uşor că $\partial^2 u/\partial t^2 = \partial^4 u/\partial x^4$, astfel încât din (10.54) se vede uşor că pentru $k = h^2/6$, adică $\lambda = 1/6$, eroarea de trunchiere devine

$$z_i^n = O(k^3 + kh^4) = O(k^3) = O(h^6) \ , \tag{10.64}$$

adică s-a câştigat un ordin de precizie numai prin alegerea adecvată a reţelei de calcul. Acest câştig nu este însă atât de spectaculos, deoarece egalitatea $\partial^2 u/\partial t^2 = \partial^4 u/\partial x^4$ este adevărată pentru ecuaţia $\partial u/\partial t = \partial^2 u/\partial x^2$ cu condiţia ca derivatele respective să fie evaluate exact. Ori, folosind diferenţe finite, această egalitate este verificată aproximativ, în funcţie de ordinul aproximării şi de pasul reţelei. Relaţia (10.54) se scrie pentru $\lambda = 1/6$ în forma

$$z_i^n = kh^2/2[(\partial^2 u/\partial t^2)_i^n - (\partial^4 u/\partial x^4)_i^n] + O(k^3 + kh^4) \ , \tag{10.65}$$

unde diferenţa $(\partial^2 u/\partial t^2)_i^n - (\partial^4 u/\partial x^4)_i^n$ nu este strict egală cu zero.

Scheme implicite.

Revenind la ecuaţia (10.35), vom utiliza aceeaşi reţea de puncte, cu deosebirea că, faţă de schema explicită (10.45), în membrul drept vom introduce valorile funcţiei aproximante v la momentul $n+1$. Se scrie deci

$$(v_i^{n+1} - v_i^n)/k = (v_{i-1}^{n+1} - 2v_i^{n+1} + v_{i+1}^{n+1})/h^2, \ i \in \overline{1, I-1}, \ n \in \overline{0, N} \ . \tag{10.66}$$

Sistemul de ecuaţii (10.66) nu mai este explicit, ci reprezintă un sistem de ecuaţii *liniare* cu matrice *tridiagonală*, care furnizează valorile funcţiei în noduri, la momentul de timp $n+1$

$$\lambda v_{i-1}^{n+1} - (1+2\lambda)v_i^{n+1} + \lambda v_{i+1}^{n+1} = -v_i^n, \ i \in \overline{1, I-1}, \ n \in \overline{0, N} \ . \tag{10.67}$$

Sistemul de ecuaţii (10.67) se rezolvă în mod repetat, pentru momente de timp succesive. Spre exemplu, pentru $n = 0$, se obţine sistemul

$$Av^{(1)} = b^{(0)} \ , \tag{10.68}$$

cu $v^{(1)} = (v_1^1, v_2^1, \ldots, v_{I-1}^1)^T$, $x_i = ih$,

$$b^{(0)} = -(f(x_1) + \lambda g_0(k), f(x_2), \ldots, f(x_{I-2}), f(x_{I-1}) + \lambda g_1(k))^T \ , \tag{10.69}$$

$$A = \begin{pmatrix} -(1+2\lambda) & \lambda & 0 & \ldots & 0 \\ \lambda & -(1+2\lambda) & \lambda & \ldots & 0 \\ 0 & \lambda & \ddots & \ddots & 0 \\ & \ddots & \ddots & \ddots & \lambda \\ 0 & & 0 & \lambda & -(1+2\lambda) \end{pmatrix} . \tag{10.70}$$

Se observă că matricea sistemului $A \in \mathcal{M}_{I-1}$ depinde numai de parametrul reţelei λ. Pentru rezolvarea sistemului liniar (10.68) se poate folosi metoda directă descrisă la 1.1.2, care ţine cont de forma tridiagonală a matricei A. Ţinând cont de particularităţile sistemului (10.70), algoritmul de rezolvare a ecuaţiei căldurii folosind schema implicită (10.66) se poate scrie formal după cum urmează:

Date : condiţiile iniţiale $f(x)$; condiţiile la limită $g_0(t)$; $g_1(t)$, h, λ, T

$k \leftarrow \lambda \cdot h^2$

Factorizare $A = L \cdot R$

$r_1 \leftarrow -(1+2\lambda)$

$i = 2 : (I-1)$

$\left[\begin{array}{l} l_i \leftarrow \lambda / r_{i-1} \\ r_i \leftarrow r_1 - l_i \lambda \end{array} \right.$

$i = 0 : I$

$[v_i^0 \leftarrow f(ih)$

$n = 0 : (T/k)$

$\left[\begin{array}{l} v_0^{n+1} \leftarrow g_0((n+1)k) \\ v_I^{n+1} \leftarrow g_1((n+1)k) \end{array} \right.$

Rezolvă sistemul $Lz = b$

$z_1 \leftarrow -v_1^n - \lambda v_0^{n+1}$

$i = 2 : (I-2)$

$[z_i \leftarrow -v_i^n - l_i z_{i-1})$

$z_{I-1} \leftarrow -v_{i-1}^n - \lambda v_I^{n+1}$

Rezolvă sistemul $Rv = z$

$v_{I-1}^{n+1} \leftarrow z_{I-1}/r_{I-1}$

$i = (I-2) : 1$

$[v_i^{n+1} \leftarrow (z_i - \lambda v_{i+1}^{n+1})/r_i$.

(10.71)

Deoarece matricea este diagonal dominantă $(1 + 2\lambda > 2\lambda)$, sistemul poate fi rezolvat şi prin *metode iterative* (s.e. prin metoda *Gauss-Seidel*). Valorile funcţiei la momentul $n = 1$, $v^{(1)}$, sunt utilizate pentru calculul valorilor $v^{(2)}$ de la momentul $n = 2$, ş.a.m.d. Schema *implicită* (10.67) *converge necondiţionat* (se va demonstra mai jos), adică indiferent de valorile parametrului reţelei λ. Această independenţă a convergenţei de parametrul reţelei este o proprietate comună a schemelor implicite.

Precizia schemelor implicite se poate îmbunătăţi folosind combinaţii ale valorilor funcţiei luate la momentele n şi $n+1$. Notând cu $\theta \in (0,1)$ un coeficient-pondere, se scrie

$$(v_i^{n+1} - v_i^n)/k = [(1-\theta)(v_{i-1}^n - 2v_i^n + v_{i+1}^n) + \theta(v_{i-1}^{n+1} - 2v_i^{n+1} + v_{i+1}^{n+1})]/h^2 \ . \tag{10.72}$$

În acest fel, derivata $\partial u/\partial t$ se calculează pentru un punct intermediar momentelor de timp n şi $n+1$, eroarea de discretizare fiind $O(k^2)$, faţă de $O(k)$ în cazurile anterioare. În multe cazuri se ia $\theta = 1/2$ metodă denumită *Crank-Nicolson*, astfel încât sistemul de ecuaţii pentru calculul valorilor funcţiei în noduri devine

$$\lambda v_{i-1}^{n+1} - 2(1+\lambda)v_i^{n+1} + \lambda v_{i+1}^{n+1} = -\lambda v_{i-1}^n - 2(1-\lambda)v_i^n - \lambda v_{i+1}^n \ . \tag{10.73}$$

Se observă o creştere a gradului de dominanţă a elementului diagonal de la $1 + 1/2\lambda$ la $1 + 1/\lambda$. Se poate arăta că pentru $\theta = (6\lambda - 1)/12\lambda$, eroarea de discretizare în derivata $\partial^2 u/\partial x^2$ devine de ordinul lui h^4.

Convergenţă, consistenţă, stabilitate.

S-a definit anterior *convergenţa schemelor cu diferenţe finite* prin condiţia ca soluţia aproximativă v să se apropie oricât de mult de soluţia exactă u, atunci când norma diviziunii tinde la zero ($k \to 0$, $h \to 0$).

Deoarece convergenţa unei scheme cu diferenţe este mai greu de demostrat, se definesc şi alte proprietăţi mai slabe ale schemei, care se pot verifica mai uşor, fiecare în parte, iar împreună pot asigura convergenţa. Astfel de proprietăţi, mai slabe, sunt *consistenţa* şi *stabilitatea*.

Definiţie. O schemă cu diferenţe finite este *consistentă* (cu ecuaţia cu derivate parţiale pe care o aproximează) dacă tinde spre ecuaţia exactă atunci când norma diviziunii tinde la zero ($k \to 0$, $h \to 0$).

De obicei schemele provenite din aproximările derivatelor prin diferenţe finite, având erorile de discretizare $O(h^p + k^m)$, $p, m \in \mathbf{N}^+$ sunt *consistente*. Există însă şi combinaţii mai puţin "ortodoxe" care conduc la erori de discretizare de ordine de mărime care depind de rapoarte ale paşilor, spre exemplu $O(h^p/k^m)$, $p, m \in \mathbf{N}^+$. În acest caz, eroarea de discretizare nu tinde neapărat la zero odată cu norma reţelei, ci poate avea valori finite.

EXEMPLU. Considerăm schema de discretizare a ecuaţiei căldurii sub forma

$$(v_i^{n+1} - v_i^{n-1})/2k = (v_{i-1}^n - v_i^{n-1} - v_i^{n+1} + v_{i+1}^n)/h^2 \ , \tag{10.74}$$

în care valoarea aproximativă a funcţiei v_i^n a fost scrisă ca medie aritmetică a valorilor v_i^{n-1} şi v_i^{n+1}, $2v_i^n = v_i^{n-1} + v_i^{n+1}$. Pentru a stabili eroarea de trunchiere, dezvoltăm în serie Taylor funcţia u în jurul punctului (i, n)

$$\begin{aligned} u_{i\pm1}^n &= u_i^n \pm hu_x + \tfrac{h^2}{2}u_{xx} \pm \tfrac{h^3}{6}u_{xxx} + \tfrac{h^4}{24}u_{xxxx} \pm \tfrac{h^5}{120}u_{xxxxx} + O(h^6) \\ u_i^{n\pm1} &= u_i^n \pm ku_t + \tfrac{k^2}{2}u_{tt} \pm \tfrac{k^3}{6}u_{ttt} + O(k^4) \ . \end{aligned}$$

Adunând şi scăzând relaţiile pentru u_i^{n-1}şi u_i^{n+1}, rezultă

$$2u_i^n = u_i^{n-1} + u_i^{n+1} - k^2u_{tt} + O(k^4), \ u_t = \tfrac{1}{2k}(u_i^{n+1} - u_i^{n-1}) + \tfrac{k^2}{6}u_{ttt} + O(k^4) \ .$$

Pentru u_{xx} se obţine relaţia

$$u_{xx} = \tfrac{1}{h^2}(u_{i-1}^n - u_i^{n-1} - u_i^{n+1} + u_{i+1}^n) - \tfrac{h^2}{12}u_{xxxx} + \tfrac{k^2}{h^2}u_{tt} + O(\tfrac{k^4}{h^2}) + O(h^4) \ ,$$

iar pentru ecuaţia căldurii

$$\begin{aligned} (u_i^{n+1} - u_i^{n-1})/(2k) &= (u_{i-1}^n - u_i^{n-1} - u_i^{n+1} + u_{i+1}^n)/h^2 - \\ &-(h^2/12)u_{xxxx} + (k^2/h^2)u_{tt} + O(k^4/h^2) + O(h^4) - (k^3/6)u_{ttt} + O(k^4) \ . \end{aligned}$$

Eroarea de discretizare conţine termenul $(k/h)^2 \left(\partial^2 u/\partial t^2\right)_i^n$ care poate fi diferit de zero dacă $h \to 0$ şi $k = ch \to 0$, unde c este o constantă diferită de zero. Deci, dacă diviziunile k, h tind la zero, spre exemplu fiind tot timpul egale ($c = 1$), schema (10.74) aproximează de fapt o ecuaţie care conţine termenul $\partial^2 u/\partial t^2$ şi care nu mai reprezintă ecuaţia căldurii. Dacă avem $k/h \ll 1$, $c \ll 1$, efectul acestui termen de ordinul doi, devine neglijabil.

Definiţie. O schemă numerică de calcul (în particular o schemă cu diferenţe) este *stabilă* dacă nu amplifică erorile în decursul aplicării ei.

În afara erorilor provenite din trunchiere sau rotunjire, surse de instabilitate pot fi şi nepotrivirile (discontinuităţile) din condiţiile la limite şi iniţiale. Modalităţile de amplificare a erorilor sunt foarte variate, astfel încât *stabilitatea* poate fi studiată în diverse moduri. În cele ce urmează, vom utiliza *stabilitatea în sensul lui von Neuman*. Importanţa studierii consistenţei şi stabilităţii este dată de

Teorema lui Lax. O schemă de calcul *consistentă* şi *stabilă* este *convergentă*.

În cele ce urmează, vom aplica teorema lui Lax la studiul unor scheme despre care ştim precis că sunt consistente, astfel încât stabilitatea devine suficientă pentru convergenţă.

Pentru schema implicită (10.67), vom studia stabilitatea în sensul lui *von Neuman*. Se introduce în schemă o perturbaţie w, de forma

$$w = \Psi(t)e^{j\beta x},\ \beta \in \mathbf{R},\ j = \sqrt{-1}\ , \tag{10.75}$$

unde $\Psi(t)$ este *amplitudinea*, iar $\beta \in \mathbf{R}$ *numărul de undă* al perturbaţiei. Deoarece sursa perturbaţiilor în calculul numeric o constituie erorile de calcul s-a păstrat notaţia w. Prin urmare, în (10.67) se face înlocuirea

$$v_i^n \leftarrow v_i^n + w_i^n;\ w_i^n = \Psi^n e^{j\beta x_i}\ . \tag{10.76}$$

Rezultă ecuaţia care dă evoluţia perturbaţiei

$$\lambda w_{i-1}^{n+1} - (1+2\lambda)w_i^{n+1} + \lambda w_{i+1}^{n+1} = -w_i^n\ . \tag{10.77}$$

Împărţind (10.77) membru cu membru cu w_i^n, se obţine

$$\begin{aligned} &(\Psi^{n+1}/\Psi^n)[\lambda e^{-j\beta h} - (1+2\lambda) + \lambda e^{j\beta h}] = -1\ , \\ &|\Psi^{n+1}/\Psi^n| \le 1/|1+2\lambda(1-\cos\beta h)| = 1/(1+4\lambda\sin^2\tfrac{\beta h}{2}) \le 1\ . \end{aligned} \tag{10.78}$$

Din (10.78) rezultă că raportul amplitudinilor perturbaţiei la momentele $n+1$ şi n scade sau rămâne cel mult nemodificat, oricare ar fi pasul h şi lungimea de undă β a perturbaţiei. În consecinţă, perturbaţiile schemei cu diferenţe date de erorile de diverse tipuri, nu se amplifică, deci *schema implicită este necondiţionat stabilă*.

Exemplu. Revenind la schema explicită (10.46) vom studia studia stabilitatea acesteia. Ecuaţia care dă evoluţia perturbaţiei w dată de (10.75) este

$$w_i^{n+1} = \lambda w_{i-1}^n + (1-2\lambda)w_i^n + \lambda w_{i+1}^n\ .$$

Rezultă condiţia

$$|\psi^{n+1}/\psi^n| = |\lambda e^{-j\beta h} + (1-2\lambda) + \lambda e^{j\beta h}| = |2\lambda(\cos\beta h - 1) + 1| \le 1,$$

sau

$$|4\lambda\sin^2\beta h/2 - 1| \le 1 \Rightarrow \lambda \le 1/2.$$

Condiţia de convergenţă $\lambda \le 1/2$ a schemei explicite (10.46) este necesară deci asigurării stabilităţii.

Figura 10.7: Discretizarea ecuaţiei parabolice cu două variabile spaţiale.

Ecuaţii parabolice cu două variabile spaţiale.

Vom lua ca model ecuaţia căldurii adimensionalizată

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \ , \tag{10.79}$$

cu condiţiile iniţiale şi la limite

$$\begin{array}{l} u(x,y,0) = h(x,y),\ 0 \le x \le 1,\ 0 \le y \le 1, \\ u(0,y,t) = f_0(y,t),\ u(1,y,t) = f_1(y,t),\ 0 \le y \le 1,\ 0 < t \le T, \\ u(x,0,t) = g_0(x,t),\ u(x,1,t) = g_1(x,t),\ 0 \le x \le 1,\ 0 < t \le T \ . \end{array} \tag{10.80}$$

Vom considera cazul unei reţele rectangulare (figura 10.7), unde pentru simplitate considerăm paşii de discretizare spaţial egali, $h_x = h_y = h$

$$x = i\,h,\ y = j\,h,\ t = n\,k,\ i \in \overline{1,I},\ j \in \overline{1,J},\ n \in \overline{1,N} \ . \tag{10.81}$$

Aproximând derivatele spaţiale cu diferenţe finite la momentul de timp n, se obţine relaţia *explicită*

$$v_{i,j}^{n+1} = (1-4\lambda)v_{i,j}^n + \lambda[(v_{i+1,j}^n + v_{i-1,j}^n) + (v_{i,j+1}^n + v_{i,j-1}^n)] \ . \tag{10.82}$$

Eroarea de trunchiere a relaţiei (10.82) este $O(k^2 + k\,h^2)$, unde k este pasul de timp. Condiţia de *convergenţă* a schemei explicite (10.82) se determină în mod similar cu cazul unei singure variabile spaţiale

$$0 < \lambda \equiv k/h^2 \le 1/4 \ . \tag{10.83}$$

Restricţia (10.83) impusă parametrului reţelei pentru ca schema explicită (10.82) să fie stabilă, duce la necesitatea folosirii unor paşi de timp mici, rezultând un volum mare de calcule. Acest neajuns poate fi înlăturat prin folosirea unei

scheme *implicite*. Spre exemplu, se poate folosi schema *Crank-Nicolson* pentru ecuaţia (10.79)

$$v_{i,j}^{n+1} - v_{i,j}^{n} = \frac{k}{2}\left[\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)_{i,j}^{n} + \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)_{i,j}^{n+1}\right], \tag{10.84}$$

obţinându-se

$$\begin{aligned} v_{i,j}^{n+1} - v_{i,j}^{n} \quad = \quad & \frac{\lambda}{2}(v_{i+1,j}^{n} - 2v_{i,j}^{n} + v_{i-1,j}^{n} + v_{i,j+1}^{n} - 2v_{i,j}^{n} + v_{i,j-1}^{n} + \\ & v_{i+1,j}^{n+1} - 2v_{i,j}^{n+1} + v_{i-1,j}^{n+1} + v_{i,j+1}^{n+1} - 2v_{i,j}^{n+1} + v_{i,j-1}^{n+1}) \end{aligned} \tag{10.85}$$

Schema *Crank-Nicolson* (10.85) este *stabilă* pentru orice valoare a parametrului reţelei $\lambda = k/h^2$. Pentru fiecare nivel de timp n, se obţine un sistem de $(I-1)(J-1)$ ecuaţii liniare unde $I \cdot h = 1$ şi $J \cdot h = 1$. Acest sistem nu mai are matrice tridiagonală ca în cazul schemei implicite (10.67) pentru ecuaţia căldurii cu o singură variabilă spaţială, neajuns care poate fi evitat folosind *metoda direcţiilor alternante*.

Metoda direcţiilor alternante.

Un sistem de n ecuaţii liniare cu n necunoscute, cu matrice este tridiagonală se rezolvă direct. În cazul ecuaţiilor parabolice cu o singură variabilă spaţială folosirea schemei implicite (10.67) a dus la rezolvarea unui astfel de sistem[4]. Obţinerea şi în cazul ecuaţiilor parabolice cu două variabile spaţiale a unui sistem liniar tridiagonal, se poate realiza dacă folosim o schemă *semi-implicită*. Spre exemplu, vom folosi o formulă implicită pentru aproximarea derivatei spaţiale $\partial^2 u/\partial x^2$

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{i,j}^{n+1} \cong (u_{i+1,j}^{n+1} - 2\,u_{i,j}^{n+1} + u_{i-1,j}^{n+1})/h^2 \tag{10.86}$$

şi o formulă explicită pentru derivata spaţială $\partial^2 u/\partial y^2$

$$\left.\frac{\partial^2 u}{\partial y^2}\right|_{i,j}^{n} \cong (u_{i,j+1}^{n} - 2\,u_{i,j}^{n} + u_{i,j-1}^{n})/h^2 \ . \tag{10.87}$$

Pentru fiecare $j \in \overline{1, J}$, se obţine un sistem de $I-1$ ecuaţii liniare[5] cu tot atâtea necunoscute

$$-\lambda\, v_{i-1,j}^{n+1} + (1+2\lambda)v_{i,j}^{n+1} - \lambda v_{i+1,j}^{n+1} = \lambda v_{i,j-1}^{n} + (1-2\lambda)v_{i,j}^{n} + \lambda v_{i,j+1}^{n} \ . \tag{10.88}$$

[4]Nu orice schemă implicită duce la apariţia unui sistem cu matrice tridiagonală, ci numai cele la care diferenţele finite pentru aproximarea derivatei spaţiale folosesc numai trei puncte alăturate la momentul de timp $n+1$.

[5]Numărul de puncte pe o dreaptă $y = const$ este $I+1$, punctele de pe frontieră fiind date prin condiţiile la limite.

Pentru determinarea valorilor $v_{i,j}^{n+1}$, trebuie rezolvate $J-1$ astfel de sisteme (vezi figura 10.7). Efortul de calcul este sensibil mai mic la rezolvarea a $J-1$ sisteme tridiagonale de câte $I-1$ ecuaţii decât pentru rezolvarea unui singur sistem de $(I-1)(J-1)$ ecuaţii cu tot atâtea necunoscute.

La pasul următor este recomandat să folosim o formulă explicită pentru aproximarea derivatei spaţiale $\partial^2 u/\partial x^2$ şi o formulă implicită pentru derivata spaţială $\partial^2 u/\partial y^2$, obţinând

$$-\lambda\, v_{i,j-1}^{n+2} + (1+2\lambda)v_{i,j}^{n+2} - \lambda v_{i,j+1}^{n+2} = \lambda v_{i-1,j}^{n+1} + (1-2\lambda)v_{i,j}^{n+1} + \lambda v_{i+1,j}^{n+1}\ . \tag{10.89}$$

Metoda direcţiilor alternante constă în aplicarea relaţiei (10.88) pentru determinarea valorilor aproximative ale funcţiei necunoscute de la momentul de timp $n+1$, apoi a relaţiei (10.89) pentru determinarea valorilor de la momentul de timp $n+2$, după care procedeul se repetă.

Ecuaţii parabolice neliniare.

Problemele prezentate până aici legate de rezolvarea numerică a ecuaţiilor cu derivate parţiale folosind scheme cu diferenţe finite şi stabilitatea acestora au fost particularizate pentru ecuaţii *liniare* cu *coeficienţi constanţi*. Ecuaţia cu derivate parţiale (10.26) este *liniară* dacă coeficienţii A_i, B_i şi C sunt cel mult funcţii de variabilele independente. Dacă numai coeficienţii derivatelor parţiale de ordin maxim depind de variabilele independente şi de funcţia necunoscută u, dar nu şi de derivatele parţiale ale acesteia, atunci ecuaţia se numeşte *cvasiliniară*.

Ecuaţiile parabolice neliniare pot fi rezolvate cu ajutorul schemelor explicite. Aşa cum am văzut anterior, stabilitatea schemelor explicite impune restricţii privind valorile parametrului reţelei $\lambda = k/h^2$. În cazul ecuaţiilor neliniare, aceste restricţii devin şi mai dure, în plus ele depinzând şi de funcţia necunoscută u, motiv pentru care schemele explicite sunt puţin utilizate.

Pentru exemplificare, considerăm ecuaţia neliniară

$$\frac{\partial^2 u}{\partial x^2} = f(x,t,u,\frac{\partial u}{\partial t},\frac{\partial u}{\partial x})\ , \tag{10.90}$$

unde $f(x,t,u,u_x,u_t)$ este o expresie care conţine variabilele independente x şi t, funcţia necunoscută u şi derivatele parţiale de ordinul întâi $\partial u/\partial x$, $\partial u/\partial t$. O schemă implicită se poate obţine dacă scriem

$$\delta_x^2 v_i^{n+1} = f[ih,(n+1)k,v_i^{n+1},(v_i^{n+1}-v_i^n)/k,(v_{i+1}^{n+1}-v_{i-1}^{n+1})/2h]\ , \tag{10.91}$$

sau prescurtat $\delta_x^2 v_i^{n+1} \equiv f_i^{n+1}$, unde

$$\delta_x^2 v_i^{n+1} \equiv (v_{i-1}^{n+1} - 2v_i^{n+1} + v_{i+1}^{n+1})/h^2\ . \tag{10.92}$$

Folosind *metoda iteraţiei simple*, se construieşte şirul

$$\delta_x^2 (v_i^{n+1})^{m+1} = (f_i^{n+1})^{m+1},\ m = 0,1,2,\dots\ , \tag{10.93}$$

până când $|(v_i^{n+1})^{m+1} - (v_i^{n+1})^m| < \varepsilon$. Pentru $m = 0$, $(v_i^{n+1})^0$ se obţine rezolvând ecuaţia neliniară

$$\delta_x^2(v_i^{n+1})^0 = (f_i^{n+1})^0 \equiv f_i^n \equiv f[ih, nt, v_i^n, ((v_i^{n+1})^0 - v_i^n)/k, (v_{i+1}^n - v_{i-1}^n)/2h] \ . \tag{10.94}$$

Dacă şirul (10.93) converge, convergenţa este liniară. După cum se observă, la fiecare iteraţie m trebuie rezolvat un sistem tridiagonal de ecuaţii liniare.

O altă modalitate de liniarizare a sistemului neliniar (10.91) o constituie *metoda Newton*. Termenul din dreapta egalităţii (10.90) se poate dezvolta în serie Taylor ca o funcţie de mai multe variabile, păstrând numai termenii de ordinul întâi

$$\begin{aligned} f(x,t,u^{m+1},u_t^{m+1},u_x^{m+1}) \quad &= \quad f(z^m) + (u^{m+1} - u^m)\frac{\partial f(z^m)}{\partial u} \\ &\quad + (u_t^{m+1} - u_t^m)\frac{\partial f(z^m)}{\partial u_t} + (u_x^{m+1} - u_x^m)\frac{\partial f(z^m)}{\partial u_x} \ , \end{aligned}$$

unde $z^m \equiv (x, t, u^m, u_t^m, u_x^m)$. Rezultă un sistem tridiagonal de forma

$$a_i(v_{i-1}^{n+1})^{m+1} - b_i(v_i^{n+1})^{m+1} + c_i(v_{i+1}^{n+1})^{m+1} = d_i, \ i \in \overline{1, I-1} \ , \tag{10.95}$$

unde

$$\begin{aligned} &a_i = \tfrac{1}{h^2} + \tfrac{1}{2h}\tfrac{\partial f(z^m)}{\partial u_x}, \ b_i = \tfrac{2}{h^2} + \tfrac{\partial f(z^m)}{\partial u} + \tfrac{1}{k}\tfrac{\partial f(z^m)}{\partial u_t}, \ c_i = \tfrac{1}{h^2} - \tfrac{1}{2h}\tfrac{\partial f(z^m)}{\partial u_x}, \\ &d_i = f(z^m) - (\tfrac{\partial f(z^m)}{\partial u} + \tfrac{1}{k}\tfrac{\partial f(z^m)}{\partial u_t})v_{i,n+1}^m - \tfrac{1}{2h}\tfrac{\partial f(z^m)}{\partial u_x}(v_{i+1,n+1}^m - v_{i-1,n+1}^m) \ . \end{aligned} \tag{10.96}$$

În obţinerea relaţiilor (10.96) s-a ţinut cont de relaţia $(v_i^n)^{m+1} - (v_i^n)^m \approx 0$, deoarece valorile $v_{i,n}$ sunt cunoscute, fiind calculate anterior. Dacă şirul (10.96) converge, convergenţa este pătratică, dar la fiecare iteraţie trebuie evaluate derivatele parţiale $\partial f(z^m)/\partial u$, $\partial f(z^m)/\partial u_t$, $\partial f(z^m)/\partial u_x$, derivate a căror expresii analitice trebuie determinate anterior. În cazul ecuaţiilor parabolice cu două sau trei variabile spaţiale, liniarizarea sistemului neliniar se face în mod asemănător, dar sistemul liniar ce rezultă nu mai are matrice tridiagonală. Pentru a obţine sisteme liniare cu matrice tridiagonală, se poate folosi metoda direcţiilor alternante, prezentată anterior.

O metodă mai simplă şi elegantă este propusă de *Saul'yev*. Vom exemplifica această metodă pentru ecuaţia cădurii adimensionalizată (10.35) în care vom aproxima derivata spaţială $(\partial^2 u/\partial x^2)_i^n$ cu formula cu diferenţe centrate

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_i^n \cong \frac{1}{h}\left(\left.\frac{\partial u}{\partial x}\right|_{i+1/2}^n - \left.\frac{\partial u}{\partial x}\right|_{i-1/2}^n\right) \ , \tag{10.97}$$

în care $(\partial u/\partial x)_{i-1/2}^n$ se înlocuieşte cu $(\partial u/\partial x)_{i-1/2}^{n+1}$, iar derivatele de ordinul întâi se aproximează tot cu diferenţe centrate

$$\left.\frac{\partial u}{\partial x}\right|_{i+1/2}^n \cong \frac{1}{h}(v_{i+1}^n - v_i^n), \ \left.\frac{\partial u}{\partial x}\right|_{i+1/2}^{n+1} \cong \frac{1}{h}(v_i^{n+1} - v_{i-1}^{n+1}) \ . \tag{10.98}$$

Figura 10.8: Ecuaţia căldurii rezolvată cu schema explicită (10.46).

Folosind diferenţe la dreapta pentru $\partial u/\partial t$, se obţine

$$(1+\lambda)v_i^{n+1} = v_i^n + \lambda(v_{i-1}^{n+1} - v_i^n + v_{i+1}^n)\ . \tag{10.99}$$

Formula (10.99) este explicită dacă efectuează calculele de la frontiera din stânga la cea din dreapta pentru toţi paşii de timp $n = 1, 2, 3, \ldots$. În mod analog, dacă în (10.97) se înlocuieşte $(\partial u/\partial x)_{i+1/2}^n$ cu $(\partial u/\partial x)_{i+1/2}^{n+1}$, se obţine

$$(1+\lambda)v_i^{n+1} = v_i^n + \lambda(v_{i+1}^{n+1} - v_i^n + v_{i-1}^n)\ . \tag{10.100}$$

Formula (10.100) este explicită dacă efectuează calculele de la frontiera din dreapta la cea din stânga pentru toţi paşii de timp $n = 1, 2, 3, \ldots$. Se recomandă folosirea alternativă a formulelor (10.99) şi (10.100).

Exemplu. În figura 10.8 sunt prezentate rezultatele obţinute folosind schema explicită (10.46) pentru ecuaţia adimensionalizată a căldurii

$$\partial u/\partial t = \partial^2 u/\partial x^2,\ u(x,0) = 4x(1-x),\ u(0,t) = u(1,t) = 0,$$

pentru două momente de timp diferite şi trei valori ale parametrului reţelei. Pasul h s-a menţinut constant la valoarea $h = 0.1$. Se observă instabilitatea schemei explicite pentru $\lambda = 0.55$. Rezultatele cele mai bune s-au obţinut pentru $\lambda = 0.1$, datorită faptului că în acest caz pasul de timp este cel mai mic şi deci eroarea de trunchiere este cea mai mică.

În figura 10.9 sunt date rezultatele obţinute cu schema implicită (10.67) şi cu schema *Saul'yev*. Se observă cum ambele scheme sunt stabile şi pentru $\lambda = 2$ (ele sunt de altfel *necondiţionat stabile*). De această dată pasul de timp k s-a menţinut constant la valoarea $k = 0.001$, modificarea parametrului reţelei λ facându-se prin îndesirea reţelei pe direcţia x. Se observă de asemenea cum rezultate mai bune se

Figura 10.9: Ecuaţia căldurii rezolvată cu schema implicită Saul'yev (10.99), (10.100).

obţin pentru $\lambda = 0.1$, deşi $\lambda = 2$, implică acelaşi pas de timp şi o reţea spaţială mai fină. Pentru a explica acest lucru, să considerăm eroarea de trunchiere a schemei explicite (10.67) care are aceeaşi formă cu (10.54). Ţinând cont de faptul că $k = \lambda h^2$ obţinem

$$z_i^n = (k^2/2)[(\partial^2 u/\partial t^2)_i^n - (\partial^4 u/\partial x^4)_i^n] + O(k^3) \ . \tag{10.101}$$

Pentru ecuaţia căldurii avem $\partial^2 u/\partial t^2 = \partial^4 u/\partial x^4$. Putem considera $(\partial^2 u/\partial t^2)_i^n \cong (\partial^4 u/\partial x^4)_i^n$ pentru ecuaţia cu diferenţe finite, astfel încât ecuaţia (10.101) se scrie

$$z_i^n = (k^2/2)(\partial^2 u/\partial t^2)_i^n [1 - 1/(6\lambda)] + O(k^3) \ . \tag{10.102}$$

Funcţia $f(\lambda) = 1 - 1/(6\lambda)$ are un minim pentru $\lambda = 1/6$, $f(1/6) = 0$. Pentru valori $\lambda > 1/6$, funcţia $f(\lambda)$ este crescătoare, deci eroare de trunchiere creşte.

10.2.2 Ecuaţii cu derivate parţiale de tip eliptic

Pentru studiul soluţiilor numerice ale ecuaţiilor cu derivate parţiale de tip eliptic, vom lua ca model *ecuaţia lui Poisson*

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y) \ . \tag{10.103}$$

Pentru ecuaţia (10.103), se pot formula trei tipuri de probleme, după condiţiile impuse pe frontiera domeniului $D \subset \mathbf{R}^2$ unde se caută soluţia. Astfel există:

1. *Probleme de tip Dirichlet.* În acest caz, se dă valoarea funcţiei u pe frontiera Γ a domeniului D

$$u(x,y)|_\Gamma = f(x,y) \ , \tag{10.104}$$

$f(x,y)$ fiind o funcţie cunoscută.

2. *Probleme de tip Neumann.* În acest caz se dă derivata funcţiei u după direcţia normalei $\vec{n}$ la curba du/dn, presupusă netedă

$$\left.\frac{du}{dn}\right|_\Gamma = g(x,y) \ , \tag{10.105}$$

$g(x,y)$ fiind o funcţie cunoscută.

3. *Probleme mixte (Dirichlet şi Neuman).* În acest caz se impune pe frontiera Γ o condiţie de forma

$$a(x,y)\ u(x,y)|_\Gamma + b(x,y)\ \left.\frac{du}{dn}\right|_\Gamma = c(x,y) \ , \tag{10.106}$$

$a(x,y)$, $b(x,y)$ şi $c(x,y)$ fiind funcţii cunoscute.

Discretizarea se poate face în toate cazurile folosind o reţea cu paşi egali în direcţia axelor x, y ($h_x = h_y = h$). Mai mult, printr-o transformare simplă de coordonate, se poate plasa întotdeauna domeniul D în interiorul unui pătrat de latură egală cu unitatea având unul din vârfuri în originea axelor (figura 10.10). În acest fel, N fiind numărul de intervale în direcţia x sau y, pasul este $h = 1/N$. Coordonatele unui punct din reţea sunt (ih, jk), i şi j fiind numărul de paşi parcurşi în cele două direcţii, plecând din origine. Dacă frontiera Γ este curbilinie, nodurile reţelei se împart în două categorii: (a) - noduri de categoria N_I, care au toate nodurile vecine (la distanţa h) în interiorul domeniului D sau pe Γ; (b) - noduri de categoria N_E, care au cel puţin un punct la distanţa h în exteriorul lui D.

Aproximarea cu diferenţe se face diferit pentru cele două categorii de noduri. Pentru un nod $M_{ij} \in N_I$, aproximarea cu ajutorul diferenţelor finite are eroarea $O(h^2)$ şi se scrie

$$v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1} - 4v_{i,j} = h^2\rho_{i,j} \ , \tag{10.107}$$

unde $i, j \in \overline{1, N-1}$, $M_{ij} \in N_I$. Dacă domeniul D este rectangular, atunci (10.107) reprezintă unica formă de aproximare.

În cazul ecuaţiei lui Laplace

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \ , \tag{10.108}$$

relaţia (10.107) se poate scrie

$$v_{i,j} = \tfrac{1}{4}(v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1}),\ i, j \in \overline{1, N-1},\ M_{ij} \in N_I, \tag{10.109}$$

Figura 10.10: Domeniu de calcul pentru ecuaţia Poisson.

adică valoarea aproximativă a funcţiei u (pe care continuăm să o notăm cu v) într-un punct oarecare este media aritmetică a valorilor din punctele vecine.

O problemă care se pune este şi *modul de numerotare* a punctelor din reţea. În general, se recomandă numerotarea care duce la matrice cât mai simetrice şi diagonal dominante, cel puţin pe anumite linii sau coloane. Aceasta asigură proprietăţi de convergenţă mai bune ale metodelor iterative de rezolvare a sistemului (10.107), care are, în general un număr mare de necunoscute ($(N-1)^2$ în cazul domeniului pătrat).

Un mod recomandabil de numerotare care duce la matrice bloc-tridiagonale este cel care satisface condiţia de constanţă a sumei indicilor

$$i + j = k,\ k = 2, 3, 4, \ldots, 2N - 2\ . \tag{10.110}$$

Atât pentru ecuaţia *Poisson* cât şi *Laplace*, după ce se aproximează cu diferenţe finite, rezultă un sistem de $(N-1)^2$ ecuaţii cu $(N-1)^2$ necunoscute. Ecuaţiile care alcătuiesc sistemul sunt liniare dacă provin din discretizarea unei ecuaţii cu derivate parţiale liniare. În cazul ecuaţiilor cu derivate parţiale neliniare, prin discretizare se obţine un sistem de ecuaţii algebrice neliniare.

Pentru rezolvarea sistemului de ecuaţii liniare (10.109) scris matriceal

$$Av = b \tag{10.111}$$

se pot folosi atât metode directe cât şi metode iterative (vezi capitolul 5). Numărul de necunoscute este în general mare; spre exemplu, pentru ecuaţia Laplace în spaţiu, pentru $N = 11$, rezultă un sistem de 1000 ecuaţii cu 1000 necunoscute, iar pentru $N = 101$, rezultă 10^6 ecuaţii. Numărul mare de necunoscute pune probleme din punct de vedere al stocării matricei A, iar numărul mare de operaţii pune probleme din punct de vedere al erorilor de rotunjire. Metodele directe sunt folosite în cazul problemelor mici, atunci când matricea A este rară (multe elemente sunt nule) sau când matricea A este de tip bandă. În celelate cazuri, metodele iterative se dovedesc mult mai eficiente.

Metodele iterative se clasifică în *metode iterative punctuale*, la care calculul valorilor necunoscute la iteraţia $k+1$, $v_{i,j}^{(k+1)}$, se face pentru fiecare punct în parte, şi *metode iterative în bloc*, la care se calculează mai multe valori simultan. Vom exemplifica folosirea metodelor iterative pentru ecuaţia Laplace cu discretizarea (10.109).

Metodele iterative punctuale.

Prezentăm pe scurt metodele *Jacobi*, *Gauss-Seidel*, *supra-relaxărilor succesive*, *gradientului*.

(*a*) *Metoda Jacobi*. Relaţia (10.109) se aplică în forma

$$v_{i,j}^{(k+1)} = \tfrac{1}{4}(v_{i-1,j}^{(k)} + v_{i+1,j}^{(k)} + v_{i,j-1}^{(k)} + v_{i,j+1}^{(k)}) \ , \tag{10.112}$$

unde $i,j \in \overline{1,N-1}$, $k = 0,1,2\ldots$. Valoarea $v_{i,j}^{(k+1)}$ se calculează folosind *numai* valori de la iteraţia anterioară. În consecinţă, valorile $v_{i,j}^{(k)}$ trebuie păstrate până când noile valori sunt calculate în totalitate, ceea ce poate constitui un dezavantaj.

(*b*) *Metoda Gauss-Seidel*. Relaţia (10.109) se aplică în forma

$$v_{i,j}^{(k+1)} = \tfrac{1}{4}(v_{i-1,j}^{(k+1)} + v_{i+1,j}^{(k)} + v_{i,j-1}^{(k+1)} + v_{i,j+1}^{(k)}) \ , \tag{10.113}$$

unde $i,j \in \overline{1,N-1}$, $k = 0,1,2\ldots$, deci valorile calculate la iteraţia $k+1$ (punctele sub linia groasă din figura (10.11) sunt imediat introduse în calcul.

(*c*) *Metoda supra-relaxărilor succesive* se bazează pe accelerarea covergenţei cu ajutorul unui *factor de relaxare*[6] $\beta \in (1,2)$

$$v_{i,j}^{(k+1)} = v_{i,j}^{(k)} + \beta[\bar{v}_{i,j}^{(k+1)} - v_{i,j}^{(k)}] = (1-\beta)v_{i,j}^{(k)} + \beta\bar{v}_{i,j}^{(k+1)}, \tag{10.114}$$

unde $\bar{v}_{i,j}^{(k+1)}$ reprezintă valoarea calculată la iteraţia $k+1$, folosind de exemplu metoda *Gauss-Seidel*.

(*d*) *Metoda gradientului*. Fie funcţia

$$U(v) = v^T Av - 2b^T v \ , \tag{10.115}$$

care are un minim egal cu $-b^T A^{-1} b$ când $v = A^{-1}b$. Se observă că minimul funcţiei $U(v)$ coincide cu soluţia sistemului $Av = b$. Metoda gradientului prezentată la 7.3, constă în construirea şirului $v^{k+1} = v^k - \alpha_k g^k$, $\alpha_k > 0$, unde $g^k = \nabla U(v^k) = 2(Av^k - b) \equiv -2r^k$, $r^k \equiv b - Av^k$ reprezentând reziduul la iteraţia k. Mai eficiente sunt metodele de gradient conjugat a căror aplicare la rezolvarea sistemelor de ecuaţii liniare a fost prezentată la **??**.

O problemă comună metodelor iterative o constituie *parcurgerea reţelei de calcul* (sau renumerotarea necunoscutelor). Pentru exemplul considerat, parcurgerea se poate face fie pe orizontală, fie pe verticală (fig. 10.11), evident, existând mai multe posibilităţi de parcurgere.

[6] În cazul problemelor neliniare, se utilizează deseori un factor de *subrelaxare* $\beta \in (0,1)$.

Figura 10.11: Parcurgerea domeniului de calcul la metodele iterative.

Metode iterative în bloc.

După cum am văzut, *metodele iterative punctuale* calculează valoarea $v_{i,j}^{(k+1)}$ pe baza unei formule explicite, în care toţi termenii din dreapta semnului egal sunt consideraţi cunoscuţi. Pentru a înţelege mai bine principiul *metodelor iterative în bloc*, considerăm metoda Gauss-Seidel, pentru care rescriem relaţia (10.113) în forma

$$v_{i,j}^{(k+1)} = \tfrac{1}{4}(v_{i-1,j}^{(k+1)} + v_{i+1,j}^{(k)} + v_{i,j-1}^{(k+1)} + v_{i,j+1}^{(k+1)}) \ . \tag{10.116}$$

unde $i, j \in \overline{1, N-1}$, $k = 0, 1 \ldots$. Relaţia (10.116) nu mai este explicită, deoarece toate valorile de pe linia i sunt considerate la iteraţia $k+1$. Pentru $j = const$, rezultă un sistem de $I-1$ ecuaţii cu $I-1$ necunoscute cu matrice tridiagonală

$$-v_{i-1,j}^{(k+1)} + 4v_{i,j}^{(k+1)} - v_{i+1,j}^{(k+1)} = v_{i,j-1}^{(k+1)} + v_{i,j+1}^{(k)} \ . \tag{10.117}$$

Redefinirea formulei explicite într-o formulă implicită duce la creşterea vitezei de convergenţă, pentru care se plăteşte preţul unei complicări a algoritmului.

Viteza de convergenţă este în general îmbunătăţită dacă după folosirea formulei implicite (10.116) la iteraţia $k+1$ se foloseşte formula

$$v_{i,j}^{(k+2)} = \tfrac{1}{4}(v_{i-1,j}^{(k+2)} + v_{i+1,j}^{(k+2)} + v_{i,j-1}^{(k+2)} + v_{i,j+1}^{(k+1)}) \tag{10.118}$$

la iteraţia următoare, adică schimbarea sensului de parcurgere a punctelor din domeniul de calcul. De această dată, rezultă un sistem de $J-1$ ecuaţii cu $J-1$ necunoscute, cu matrice tridiagonală

$$-v_{i,j-1}^{(k+2)} + 4v_{i,j}^{(k+2)} - v_{i,j+1}^{(k+2)} = v_{i-1,j}^{(k+2)} + v_{i+1,j}^{(k+1)} \ . \tag{10.119}$$

O altă metodă iterativă în bloc este *metoda direcţiilor alternante* prezentată anterior la paragraful 43. Derivatele parţiale din ecuaţia Laplace $u_{xx} + u_{yy} = 0$

le vom înlocui de data aceasta cu diferenţe finite după schema

$$(v_{i-1,j}^{(k+1)} - 2v_{i,j}^{(k+1)} + v_{i+1,j}^{(k+1)}) + (v_{i,j-1}^{(k)} - 2v_{i,j}^{(k)} + v_{i,j+1}^{(k)}) = 0, \tag{10.120}$$

adică u_{xx} cu o formulă implicită, iar u_{yy} cu o formulă explicită. Din relaţia (10.120), adăugând şi scăzând valoarea $4v_{i,j}$, rezultă

$$(v_{i-1,j}^{(k+1)} - 2v_{i,j}^{(k+1)} + v_{i+1,j}^{(k+1)}) + (v_{i,j-1}^{(k)} - 2v_{i,j}^{(k)} + v_{i,j+1}^{(k)}) + 4v_{i,j} - 4v_{i,j} = 0 \ .$$

Considerând prima valoare $v_{i,j}$ la iteraţia k şi cea dea a doua la iteraţia $k+1$, rezultă

$$v_{i,j}^{(k+1)} = v_{i,j}^{(k)} + \tfrac{1}{4}(v_{i-1,j}^{(k+1)} - 2v_{i,j}^{(k+1)} + v_{i+1,j}^{(k+1)}) + \tfrac{1}{4}(v_{i,j-1}^{(k)} - 2v_{i,j}^{(k)} + v_{i,j+1}^{(k)})$$

sau

$$-\tfrac{1}{4}v_{i-1,j}^{(k+1)} + \tfrac{3}{2}v_{i,j}^{(k+1)} - \tfrac{1}{4}v_{i+1,j}^{(k+1)} = \tfrac{1}{4}v_{i,j-1}^{(k)} - \tfrac{1}{2}v_{i,j}^{(k)} + \tfrac{1}{4}v_{i,j+1}^{(k)} \ . \tag{10.121}$$

Am obţinut o relaţie identică cu (10.88) pentru $\lambda = \frac{1}{4}$, relaţia (10.121) reprezentând formula de bază pentru *metoda direcţiilor alternante*. În mod analog, folosind o formulă implicită pentru u_{yy} şi una explicită pentru u_{xx}, se obţine

$$-\tfrac{1}{4}v_{i,j-1}^{(k+2)} + \tfrac{3}{2}v_{i,j}^{(k+2)} - \tfrac{1}{4}v_{i,j+1}^{(k+2)} = \tfrac{1}{4}v_{i-1,j}^{(k+1)} - \tfrac{1}{2}v_{i,j}^{(k+1)} + \tfrac{1}{4}v_{i+1,j}^{(k+1)} \ . \tag{10.122}$$

Observăm cum pe baza unor prelucrări efectuate asupra relaţiei (10.120), am obţinut o ecuaţie cu diferenţe finite, pentru o altă ecuaţie cu derivate parţiale, şi anume pentru ecuaţia de tip parabolic $u_t = u_{xx} + u_{yy}$. Altfel spus, schemele cu diferenţe finite (10.121) şi (10.122) nu sunt consistente cu ecuaţia cu derivate parţiale de tip eliptic $u_{xx} + u_{yy} = 0$, ci cu ecuaţia parabolică $u_t = u_{xx} + u_{yy}$, a cărei soluţie pentru valori de timp suficient de mari coincide cu soluţia ecuaţiei $u_{xx} + u_{yy} = 0$. În general, soluţia unei probleme de echilibru (care nu depinde de timp), poate fi obţinută ca o soluţie a unei probleme fictive dependente de timp, pentru $t = T$ suficient de mare. De fapt, rezolvarea unei probleme eliptice cu o metodă iterativă, este analoagă cu rezolvarea unei probleme dependente de timp, numai că pasul de timp variază de la punct la punct.

Cazul frontierelor curbe.

În acest caz, mulţimea N_E a punctelor care au noduri la distanţa h exterioare domeniului D este nevidă. Scrierea ecuaţiilor cu diferenţe finite pentru aceste puncte depinde de natura condiţiilor impuse pe frontieră. Vom considera cazul problemelor de tip *Dirichlet* şi *Neuman*, problema *mixtă* fiind o combinaţie a celor două.

În cazul problemelor de tip *Dirichlet*, sunt date valorile funcţiei u pe frontiera Γ. Fie punctul $1 \in N_E$ (figura 10.12) punctul în care dorim să aproximăm cu diferenţe ecuaţia lui *Poisson* (10.103). Am notat cu ah şi bh, $a, b \in (0, 1)$

Figura 10.12: Frontieră curbă, condiţii Dirichlet.

distanţele de la punctul 1 la punctele A, B de pe frontieră. Folosind formula Taylor, se obţine

$$\begin{aligned} u_A &= u_1 - ah(\partial u/\partial x)_1 + \tfrac{1}{2}a^2h^2(\partial^2 u/\partial x^2)_1 + O(h^3) \ , \\ u_B &= u_1 + bh(\partial u/\partial y)_1 + \tfrac{1}{2}b^2h^2(\partial^2 u/\partial y^2)_1 + O(h^3) \ . \end{aligned} \tag{10.123}$$

Deoarece, pe lângă derivatele de ordinul 2 care apar în ecuaţia lui Poisson, au apărut şi derivate de ordinul 1, aceste din urmă trebuie eliminate. În acest scop, sunt necesare încă două relaţii. Ca urmare, se iau încă două puncte din reţea, punctele 2 şi 3 (figura 10.12) şi se scrie

$$\begin{aligned} u_2 &= u_1 + h(\partial u/\partial x)_1 + \tfrac{1}{2}h^2(\partial^2 u/\partial x^2)_1 + O(h^3) \ , \\ u_3 &= u_1 - h(\partial u/\partial y)_1 + \tfrac{1}{2}h^2(\partial^2 u/\partial y^2)_1 + O(h^3) \ . \end{aligned} \tag{10.124}$$

Combinând primele relaţii din (10.123) şi (10.124) între ele, iar apoi şi ultimele, se obţine

$$\begin{aligned} u_A + au_2 &= (1+a)u_1 + \tfrac{h^2}{2}a(1+a)(\partial^2 u/\partial x^2)_1 + O(h^3) \ , \\ u_B + au_3 &= (1+b)u_1 + \tfrac{h^2}{2}b(1+b)(\partial^2 u/\partial y^2)_1 + O(h^3) \ . \end{aligned} \tag{10.125}$$

Ecuaţia (10.103) se scrie, în punctul 1

$$(\partial^2 u/\partial x^2)_1 + (\partial^2 u/\partial y^2)_1 = \rho_1 \ . \tag{10.126}$$

Ca urmare, exprimând derivatele de ordinul 2 din relaţiile (10.125) se obţine ecuaţia

$$-u_1\left(\frac{1}{a}+\frac{1}{b}\right) + \frac{u_2}{1+a} + \frac{u_3}{1+b} = \frac{h^2}{2}\rho_1 - \frac{u_A}{a(1+a)} - \frac{u_B}{b(1+b)}, \ u_1 \in N_E \ . \tag{10.127}$$

Din relaţiile (10.125), se observă că derivatele de ordinul 2 se calculează în punctele de categoria N_E cu precizia $O(h)$, faţă de $O(h^2)$ în relaţiile (10.107). Ca urmare, eroarea de discretizare în (10.127) este mai mare. Acest aspect

Figura 10.13: Frontieră curbă, condiţii Neuman.

este mai general: scrierea echivalentului în diferenţe finite în punctele de lângă frontieră, de categoria N_E, se face cu o precizie mai mică.

În cazul problemelor de tip *Neuman*, sunt date la frontieră derivatele după direcţia normalei la Γ. Notând cu $\vec{n}_1,\ \vec{n}_2,\ \vec{n}_3$ versorii vectorilor $1A,\ 2A,\ 2A$ (figura 10.13), punctul A fiind cel mai apropiat de punctul 1 în care vrem să scriem o relaţie pentru valorile funcţiei necunoscute, vom scrie

$$\vec{n}_k = \alpha_k \vec{i} + \beta_k \vec{j},\ k = 1, 2, 3 \tag{10.128}$$

şi vom obţine, cu o aproximaţie $O(h^2)$

$$u_k = u_A - l_k(\alpha_k \partial u/\partial x + \beta_k \partial u/\partial y)_A + O(h^2),\ l_k = \overline{Ak},\ k = 1, 2, 3\ . \tag{10.129}$$

Eliminând, mai departe, pe u_A, rezultă

$$\begin{aligned} u_2 - u_1 &= (l_1\alpha_1 - l_2\alpha_2)(\partial u/\partial x)_A + (l_1\beta_1 - l_2\beta_2)(\partial u/\partial y)_A\ , \\ u_3 - u_1 &= (l_1\alpha_1 - l_3\alpha_3)(\partial u/\partial x)_A + (l_1\beta_1 - l_3\beta_3)(\partial u/\partial y)_A\ . \end{aligned} \tag{10.130}$$

Pe de altă parte, derivata normală

$$\left(\frac{du}{dn}\right)_A = \alpha_A \left(\frac{\partial u}{\partial x}\right)_A + \beta \left(\frac{\partial u}{\partial y}\right)_A \tag{10.131}$$

este cunoscută, astfel încât relaţia căutată între u_1, u_2, u_3 este tocmai condiţia de compatibilitate a sistemului de 3 ecuaţii cu necunoscutele $(\partial u/\partial x)_A$, $(\partial u/\partial y)_A$ format cu relaţiile (10.130) şi (10.131)

$$\begin{vmatrix} u_2 - u_1 & l_1\alpha_1 - l_2\alpha_2 & l_1\beta_1 - l_2\beta_2 \\ u_3 - u_1 & l_1\alpha_1 - l_3\alpha_3 & l_1\beta_1 - l_3\beta_3 \\ (du/dn)_A & \alpha_A & \beta_A \end{vmatrix}\ . \tag{10.132}$$

Ecuaţia (10.132) înlocuieşte ecuaţia lui *Poisson* în punctul 1, cu aproximaţia $O(h)$ care reprezintă eroarea la interpolarea liniară a derivatelor în relaţia (10.129).

Exemplu. Fie ecuaţia Laplace cu condiţiile la limită

$$u_{xx} + u_{yy} = 0,\ u(0, y) = u(1, y) = 1,\ u(x, 0) = u(x, 1) = 0\ . \tag{10.133}$$

Figura 10.14: Distribuţia de temperaturi într-o placă.

Soluţia ecuaţiei (10.133) reprezintă distribuţia de temperaturi în regim staţionar pentru o placă pătrată. S-a folosit pentru rezolvare schema (10.109) pentru o reţea cu pasul $h = 0.1$. În punctele din colţurile plăcii s-a luat valoarea $u = 0.5$ deoarece în aceste puncte condiţiile la limite prezintă discontinuităţi. Distribuţia de temperaturi, precum şi curbele de nivel $u = K$, pentru $K \in \{0.2, 0.4, 0.6, 0.8\}$ sunt date în figura (10.14). Pentru rezolvarea sistemului de ecuaţii liniare s-au folosit metodele *Jacobi*, *Gauss-Seidel* şi *relaxărilor succesive*. S-a folosit condiţia de oprire a iteraţiilor $\|v^{k+1} - v^k\| < \varepsilon$, cu $\varepsilon = 10^{-4}$. Numărul de iteraţii necesare obţinerii preciziei impuse poate fi citit din tabelul următor

metoda	*Jacobi*	$G-S$	$\beta = 1.2$	$\beta = 1.4$	$\beta = 1.5$	$\beta = 1.6$	$\beta = 1.8$
nr.iter.	121	68	48	31	23	22	42

Se observă superioritatea metodei *supra-relaxărilor succesive* faţă de celelalte metode pentru $\beta \in [1.2, 1.8]$, precum şi existenţa unui minim pentru $\beta = 1.6$, minim care depinde în general de ecuaţia care se rezolvă. Pentru $\beta = 1$ se regăseşte metoda *Gauss-Seidel*, iar pentru $\beta = 2$ metoda nu converge.

10.2.3 Ecuaţii cu derivate parţiale de tip hiperbolic

Pentru studiul soluţiilor numerice ale ecuaţiilor cu derivate parţiale de tip hiperbolic, vom lua ca model *ecuaţia undelor*

$$\frac{\partial^2 u}{\partial t^2} - \frac{\partial^2 u}{\partial x^2} = 0 \ , \tag{10.134}$$

a cărei soluţie generală este de forma

$$u(x,t) = f(x+t) + g(x-t) \ , \tag{10.135}$$

unde f şi g sunt două funcţii arbitrare de clasă C^2. Ecuaţia (10.134) poate avea

Figura 10.15: Domeniile de influenţă şi de dependenţă.

(*a*) *numai condiţii iniţiale*

$$u(x,0)=f(x),\ \partial u/\partial t(x,0)=g(x),\ -\infty<x<+\infty\ , \tag{10.136}$$

caz în care funcţiile f şi g din (10.134) sunt chiar cele date de (10.136), sau

(*b*) *condiţii iniţiale şi condiţii la limite*

$$\begin{aligned} &u(x,0)=f(x),\ \partial u/\partial t(x,0)=g(x),\ 0<x<1\ ,\\ &u(0,t)=h_0(t),\ u(1,t)=h_1(t)\ . \end{aligned} \tag{10.137}$$

În acest caz, pentru a nu avea discontinuităţi între condiţiile iniţiale şi cele la limite, trebuie ca $f(0)=h_0(0)$ şi $g(0)=h_1(0)$.

Ecuaţia undelor are caracteristicile

$$\varphi_1(x,t)=x-t,\ \varphi_2(x,t)=x+t\ . \tag{10.138}$$

Cele două caracteristici care trec printr-un punct $P(x_0,t_0)$ (figura 10.15) taie axa Ox în punctele A şi B. Valoarea soluţiei în punctul P, depinde numai de valorile din interiorul triunghiul PAB. Segmentul $\overline{AB}$ se numeşte *domeniu (interval) de dependenţă*. Pe de altă parte, prin punctul $Q(x_0,0)$ de pe axa Ox, trec cele două caracteristici de ecuaţii $x+t=x_0$ şi $x-t=x_0$ care reprezintă *domeniul de influenţă* al punctului Q, adică acea parte a domeniului care este influenţată de valoarea soluţiei din punctul Q.

O problemă importantă constă în stabilirea condiţiilor în care soluţia ecuaţiei (10.134) cu (10.136) sau (10.137) *există* şi este *unică*. Altfel spus, se pune problema stabilirii valorilor ce trebuie date pe frontierele domeniului de calcul pentru ca soluţia să existe şi să fie unică.

Pentru ecuaţia de ordinul al doilea (10.31), se disting trei cazuri la care se poate demonstra că soluţia există şi este unică (figura 10.16):

Figura 10.16: Condiţiile de existenţă şi unicitate.

1. Valorile funcţiei u *şi* $u_x \equiv \partial u/\partial x$ sunt date pe o *curbă necaracteristică* CD. Domeniul în care soluţia există şi este unică este mărginit de caracteristicile CE şi DE şi de curba CD (figura din stânga).
2. Valorile u *sau* u_x sunt date pe curba caracteristică CF şi pe curba necaracteristică CD. Domeniul în care soluţia există şi este unică este interiorul $CDEF$ (figura din mijloc).
3. Valorile u *sau* u_x sunt date pe două curbe caracteristice ce trec prin acelaşi punct, CE şi CD. Domeniul în care soluţia există şi este unică este interiorul $CEFD$ (figura din dreapta).

Pentru rezolvarea ecuaţiilor hiperbolice vom prezenta *metoda caracteristicilor* şi metode de discretizare cu *diferenţe finite*.

10.2.4 Metoda caracteristicilor

Metoda caracteristicilor reprezintă o metodă *naturală* pentru rezolvarea ecuaţiilor hiperbolice. Dimensiunea caracteristicii (din punct de vedere al geometriei) este $n-1$, unde n este numărul de coordonate. Spre exemplu, pentru $n=2$ avem *linii caracteristice*, pentru $n=3$ avem *suprafeţe caracteristice*, ş.a.m.d. Eventualele discontinuităţi care pot apărea în interiorul domeniului de calcul şi a căror poziţie este în general necunoscută înainte de a obţine soluţia, se propagă după direcţiile carateristice, fapt care constituie un avantaj suplimentar al metodei caracteristicilor faţă de metodele de discretizare cu diferenţe finite.

Reluăm ecuaţia (10.31), care pentru $b^2 - a\,c > 0$ este de tip hiperbolic cu două caracteristici reale şi distincte. Pentru ca sistemul (10.33) se fie compatibil, trebuie ca simultan cu determinantul matricei sistemului să se anuleze şi determinantul

$$\begin{vmatrix} a & f & c \\ dx & d(u_x) & 0 \\ 0 & d(u_y) & dy \end{vmatrix} \tag{10.139}$$

Figura 10.17: Linii caracteristice.

rezultând

$$\begin{aligned} am_1 d(u_x)/dx + c\, d(u_y)/dx - f\, m_1 = 0, \\ am_2 d(u_x)/dx + c\, d(u_y)/dx - f\, m_2 = 0, \end{aligned} \tag{10.140}$$

cu $m_{1,2}$ pantele celor două caracteristici date de

$$m_{1,2} \equiv \frac{dy}{dx} = \frac{1}{2a}[b \pm \sqrt{b^2 - 4ac}] \ . \tag{10.141}$$

Pentru $n = 2$, ecuaţiile cu derivate parţiale se transformă în lungul direcţiilor caracteristice în ecuaţii diferenţiale ordinare, acesta reprezentând avantajul principal al metodei caracteristicilor. S-a obţinut un sistem de ecuaţii diferenţiale format din ecuaţiile (10.140) şi (10.141), cu necunoscutele m_1, m_2, $d(u_x)/dx$ şi $d(u_y)/dx$. Pentru rezolvvarea sistemului, presupunem că se cunosc valorile u, u_x şi u_y pe curba Γ diferită de o curbă caracteristică. În figura 10.17 se pot observa cele două familii de caracteristici notate m_1 şi m_2. Deoarece caracteristicile depind în general de soluţia u, simultan cu determinarea valorii soluţiei u_P în punctul R se calculează şi poziţia acestuia. În principiu, orice metodă numerică prezentată la capitolul **??** se poate aplica pentru rezolvarea sistemului de ecuaţii diferenţiale obţinut. Folosind *metoda Euler*, segmentul PR este aproximat cu o linie dreaptă[7] astfel încât precizia este de ordinul distanţei dintre

[7] Eroarea de discretizare poate fi redusă folosind arce de parabolă în locul segmentelor de dreaptă, cu preţul unor calcule mai laborioase.

două puncte considerate în calcul. Se obţin relaţiile

$$\begin{aligned}
&(y_R - y_P)/(x_R - x_P) = (m_{1R} + m_{1P})/2 \\
&(y_R - y_Q)/(x_R - x_Q) = (m_{2R} + m_{2Q})/2 \\
&(a_R m_{1R} + a_P m_{1P})(u_{xR} - u_{xP})/2 + (c_R + c_P)(u_{yR} - u_{yP})/2 \\
&\qquad -(f_R + f_P)(y_R - y_P)/2 = 0 \\
&(a_R m_{2R} + a_Q m_{2Q})(u_{xR} - u_{xQ})/2 + (c_R + c_Q)(u_{yR} - u_{yQ})/2 \\
&\qquad -(f_R + f_Q)(y_R - y_Q)/2 = 0
\end{aligned} \tag{10.142}$$

cu necunoscutele x_R, y_R, u_{xR}, u_{yR} şi u_R (u_R apare indirect prin coeficienţii a_R, b_R, c_R şi f_R). Pentru determinarea valorii u_R este nevoie de o aproximaţie a relaţiei

$$du = u_x\, dx + u_y\, dy\ . \tag{10.143}$$

Se scrie

$$\begin{aligned}
u_R - u_P &= (u_{xR} + u_{xP})(x_R - x_P)/2 + (u_{yR} + u_{yP})(y_R - y_P)/2 \\
u_R - u_Q &= (u_{xR} + u_{xQ})(x_R - x_Q)/2 + (u_{yR} + u_{yQ})(y_R - y_Q)/2\ .
\end{aligned} \tag{10.144}$$

Ecuaţiile (10.142) şi una din (10.144) reprezintă un sistem de 5 ecuaţii neliniare cu necunoscutele x_R, y_R, u_{xR}, u_{yR} şi u_R. Sistemul se poate rezolva iterativ cu metoda iteraţiei simple (capitolul 8.1), luând pentru start valorile necunoscute din punctul R egale cu cele din punctul de plecare al caracteristicii, P sau Q. Spre exemplu, prima ecuaţie din (10.142) se scrie

$$(y_R - y_P)/(x_R - x_P) = (m_{1P} + m_{1P})/2 = m_{1P}\ . \tag{10.145}$$

Procedeul se repetă pentru calculul valorilor soluţiei în celelalte puncte ale reţelei de curbe caracteristice (punctele T, S, etc.). Obervăm cum în cazul folosirii celei mai simple metode de rezolvare a sistemului de ecuaţii diferenţiale, pentru fiecare punct al reţelei de calcul se rezolvă un sistem de ecuaţii liniare.

Metoda hibrid Hartree.

În cazul unei ecuaţii hiperbolice de ordinul al doilea cu două variabile independente dintre care una este timpul, se poate cere soluţia u la momente de timp specificate. Metoda caracteristicilor aplicată direct nu poate asigura acest lucru, astfel că se foloseşte o metodă hibrid între o metodă cu diferenţe finite şi metoda caracteriticilor. Această metodă constă în folosirea unei reţele de discretizare fixate înainte de determinarea soluţiei (figura 10.18), cu pasul h pentru Ox şi k pentru Ot. Presupunem că s-au efectuat calculele până la momentul de timp $t = n\,k$ şi trebuie să calculăm soluţia pentru momentul de timp următor. Valorile u, u_x, u_y sunt calculate deci în nodurile reţelei până la linia n. Caracteristicile care trec prin punctul R intersectează linia n în punctele P şi Q. Valorile u, u_x, u_y se pot determina prin interpolare folosind valorile din nodurile de pe linia n (B, A, D). Folosind metoda Euler pentru rezolvarea sistemului de ecuaţii diferenţiale ordinare ce rezultă pe direcţiile caracteristice RP şi RQ, se obţine acelaşi sistem neliniar (10.142) şi (10.144), necunoscutele fiind de data aceasta u_R, u_{xR}, u_{yR}, x_P şi x_Q, deoarece x_R, y_R, y_P şi y_Q sunt cunoscute.

Figura 10.18: Metoda hibrid Hartree.

10.2.5 Scheme cu diferenţe finite

Pentru ecuaţiile hiperbolice de ordinul al doilea, metoda caracteristicilor reprezintă varianta cea mai convenabilă şi precisă de rezolvare numerică. Unul dintre avantajele majore ale metodei caracteristicilor faţă de metodele bazate pe scheme cu diferenţe finite constă în faptul că discontinuităţile care există eventual în condiţiile iniţiale se propagă de-a lungul direcţiilor caracteristice.

În cazul în care ecuaţiile nu sunt foarte complicate iar soluţiile nu prezintă discontinuităţi, se pot folosi metode cu diferenţe finite. Alegerea unei scheme explicite sau implicite depinde esenţial de tipul problemei: numai cu condiţii iniţiale sau cu condiţii iniţiale şi la limită.

Scheme explicite.

Vom considera problema cu condiţii iniţiale

$$u_{xx} = u_{tt}, \quad u(x,0) = f(x), \quad u_t(x,0) = g(x) \ . \tag{10.146}$$

Derivatele parţiale le aproximăm cu diferenţe centrate

$$u_{xx} \cong (v_{i-1}^n - 2v_i^n + v_{i+1}^n)/h^2, \ u_{tt} \cong (v_i^{n-1} - 2v_i^n + v_i^{n+1})/k^2 \ . \tag{10.147}$$

Rezultă

$$v_i^{n+1} = m^2(v_{i-1}^n + v_{i+1}^n) + 2(1-m^2)v_i^n - v_i^{n-1}, \text{ unde } m = k/h \ . \tag{10.148}$$

Pentru $n = 0$, avem date valorile $v_i^0 = f(ih)$. În relaţia (10.148) apar şi valorile $v_{i,}^{-1}$, care se calculează cu ajutorul celei de a doua condiţii iniţiale (10.146). Se aproximează derivata în raport cu timpul $u_t \equiv \partial u/\partial t$ folosind diferenţe centrate

$$g(ih) \equiv g_i = \ u_t|_i^0 \equiv \ \partial u/\partial t|_i^0 \cong (v_i^1 - v_i^{-1})/2k \Rightarrow v_i^{-1} = v_i^1 - 2kg_i \ . \tag{10.149}$$

Figura 10.19: Caracteristicile ecuaţiei $u_{xx} = u_{tt}$, exact şi numeric.

Pentru $n = 0$ rezultă

$$v_i^1 = \tfrac{1}{2}m^2(f_{i-1} + f_{i+1}) + (1 - m^2)f_i + kg_i \ . \tag{10.150}$$

În figura 10.19 sunt date punctele care intră în calculul valorii v_i^{n+1}, precum şi domeniul de dependenţă fizică pentru punctul C. Dreptele AC şi BC sunt caracteristicile ecuaţiei $u_{xx} = u_{tt}$ care trec prin punctul C. Soluţia este deci unic definită în triunghiul ABC. Dacă $m = k/h > 1$, atunci se vor calcula numeric valori aproximative ale soluţiei u în afara domeniului în care avem soluţie analitică. Dacă $m \leq 1$, atunci soluţia numerică se află interiorul domeniului de dependenţă fizică al punctului C. *În general, o schemă cu diferenţe finite este stabilă, dacă domeniul de influenţă numerică include domeniului de influenţă fizică (analitică).*

Pentru a studia stabilitatea schemei (10.148), se consideră soluţia de forma

$$u(x,t) = \sum_{n=0}^{\infty} e^{\alpha_n t} e^{j\beta_n x} \ , \tag{10.151}$$

unde $j \equiv \sqrt{-1}$. Ecuaţia (10.146) fiind liniară, acţionează principiul superpoziţiei astfel că este suficient să se studieze un singur termen din (10.151). Avem

$$v_i^n = e^{\alpha nk} e^{j\beta ih} \tag{10.152}$$

şi introducând în (10.148) se obţine

$$e^{\alpha k} + e^{-\alpha k} = 2 - 4m^2 \sin^2 \tfrac{\beta h}{2} \ , \tag{10.153}$$

sau

$$e^{2\alpha k} - 2(1 - 2m^2 \sin^2 \tfrac{\beta h}{2})e^{\alpha k} + 1 = 0 \ . \tag{10.154}$$

Pentru a evita situaţia în care soluţia numerică creşte când $k \to \infty$, trebuie ca $e^{\alpha k} \leq 1$. Din (10.154) se observă că produsul $|e^{\alpha k}|_1 \cdot |e^{\alpha k}|_2 = 1$, deci singura

Figura 10.20: Coarda vibrantă, $n = 2$, $t = 1$.

posibilitate pentru a asigura stabilitatea este $|e^{\alpha k}|_1 = |e^{\alpha k}|_2 = 1$. Cele două soluţii ale ecuaţiei (10.154) sunt

$$e^{\alpha k}_{1,2} = A \pm \sqrt{\Delta}, \text{ unde } A = 1 - 2m^2 \sin^2 \tfrac{\beta h}{2},\ \Delta = A^2 - 1\ . \tag{10.155}$$

Se observă că pentru a avea $e^{\alpha k}_{1,2} = 1$, este necesar ca $\Delta \le 0$, adică

$$(1 - 2m^2 \sin^2 \tfrac{\beta h}{2})^2 - 1 \le 0,\ \text{ sau }\ -1 \le 1 - 2m^2 \sin^2 \tfrac{\beta h}{2} \le 1\ . \tag{10.156}$$

A doua inegalitate (10.156) este îndeplinită automat, iar din prima inegalitate rezultă

$$m^2 \le 1/\sin^2 \tfrac{\beta h}{2}, \tag{10.157}$$

adică *domeniul de influenţă numerică să includă domeniul de influenţă fizică.*

Cazul particular $m = 1$ prezintă un interes special. Din (10.156) rezultă

$$-1 \le 1 - 2\sin^2 \tfrac{\beta h}{2} \Rightarrow \sin^2 \tfrac{\beta h}{2} \le 1\ .$$

Pentru $\sin^2 \frac{\beta h}{2} = 1$, obţinem $\Delta = 0$, $e^{\alpha k}_{1,2} = -1$. Relaţia (10.148) devine

$$v_i^{n+1} = v_{i-1}^n + v_{i+1}^n - v_{i,n-1}\ . \tag{10.158}$$

Schema (10.158) satisface *exact* ecuaţia (10.146). Pentru a arăta acest lucru, folosim dezvoltarea în serie Taylor a funcţiei u în jurul punctului (ih, nk), obţinând

$$\begin{aligned} u_{xx} &= \tfrac{1}{h^2}(u_{i-1}^n - 2u_i^n + u_{i+1}^n) - \tfrac{h^2}{12}u_{xxxx} + O(h^4)\ , \\ u_{tt} &\approx \tfrac{1}{k^2}(u_{i,n-1} - 2u_i^n + u_i^{n+1}) - \tfrac{k^2}{12}u_{tttt} + O(k^4)\ . \end{aligned} \tag{10.159}$$

Înlocuind în (10.146), rezultă

$$\begin{aligned} u_i^{n+1} &= m^2(u_{i-1}^n + u_{i+1}^n) + 2(1-m^2)u_i^n - u_i^{n-1} + \\ &\quad + \frac{k^4}{12}u_{tttt} - \frac{k^2h^2}{12}u_{xxxx} + O(k^6) + O(k^2h^4)\ , \\ u_i^{n+1} &\overset{m=1}{=} u_{i-1}^n + u_{i+1}^n - u_i^{n-1} + \frac{h^4}{12}\left(\frac{\partial^4 u}{\partial t^4} - \frac{\partial^4 u}{\partial x^4}\right) + \frac{h^6}{360}\left(\frac{\partial^6 u}{\partial t^6} - \frac{\partial^6 u}{\partial x^6}\right) + O(h^8)\ . \end{aligned} \tag{10.160}$$

În cazul în care funcţia u este de p ori derivabilă, se poate uşor arăta că dacă $u_{xx} = u_{tt}$, atunci $\partial^p u/\partial x^p = \partial^p u/\partial t^p$, astfel că ultima relaţie (10.160) devine

$$u_i^{n+1} = u_{i-1}^n + u_{i+1}^n - u_i^{n-1}, \tag{10.161}$$

astfel încât schema cu diferenţe (10.158) are eroarea de trunchiere nulă.

Scheme implicite.

O schemă implicită poate fi scrisă în general ca

$$v_i^{n-1} - 2v_i^n + v_i^{n+1} = m^2[\theta\delta_x^2 v_i^{n+1} + (1-2\theta)\delta_x^2 v_i^n + \theta\delta_x^2 v_{i,n-1}], \tag{10.162}$$

unde θ este un parametru de relaxare, iar $\delta_x^2 v_i^n \equiv v_{i-1}^n - 2v_i^n + v_{i+1}^n$. În relaţia (10.162), valoarea aproximativă a derivatei u_{tt} se calculează ca o medie ponderată între valorile aproximative la momentele de timp $n-1$, n, $n+1$. Se poate arăta că pentru $\theta \geq 1/4$, schema (10.162) este *stabilă* oricare ar fi parametrul reţelei de calcul m. Pentru $\theta = 0$, se obţine schema explicită (10.148), iar pentru $\theta = 1/2$ se obţine schema

$$\begin{aligned} v_i^{n+1} - 2v_i^n + v_{i+1}^{n+1} &= \tfrac{1}{2}m^2[\delta_x^2 v_i^{n+1} + \delta_x^2 v_{i,n-1}] = \\ &= \tfrac{1}{2}m^2[(v_{i+1}^{n+1} - 2v_i^{n+1} + v_{i-1}^{n+1}) + (v_{i+1,n-1} - 2v_{i,n-1} + v_{i-1,n-1})\ . \end{aligned} \tag{10.163}$$

Exemplu. Fie ecuaţia adimensionalizată a coardei vibrante cu condiţiile iniţiale şi la limite date de

$$u_{tt} = u_{xx},\ u(0,t) = u(1,t) = 0,\ u(x,0) = \sin(\pi n x),\ u_t(x,0) = 0\ . \tag{10.164}$$

Pentru rezolvare s-a folosit schema explicită (10.148). Soluţia analitică a problemei (10.164) este

$$u(x,t) = \cos(\pi n t)\ \sin(\pi n x)$$

În figura 10.20 sunt reprezentate rezultatele pentru trei valori diferite ale pasului k, cu $m = 1$ ($k = h$). Se observă rezultatele foarte bune obţinute pentru $k = 0.01$, bune pentru $k = 0.05$, în timp ce pentru $k = 0.1$, rezultatele sunt nesatisfăcătoare. Sunt date de asemenea rezultatele obţinute pentru două valori $m \neq 1$, $h = 0.05$, $k = 0.1$, ($m = 2$), şi $h = 0.1$, $k = 0.01$, ($m = 0.1$). La acelaşi pas de timp $k = 0.1$, pentru $h = 0.05$ ($m = 2$) se obţin rezultate mai slabe decât pentru $h = 0.1$ ($m = 1$), deşi în primul caz h este mai mic. De altfel, pentru $m > 1$ schema (10.148) nu este stabilă, fiind de aşteptat ca pentru valori mai mari în timp rezultatele să fie din ce în ce mai

Figura 10.21: Coarda vibrantă, $n = 8$, $t = 1$.

proaste. Pentru $m = 0.1$ ($h = 0.1$) nu se observă diferenţe semnificative faţă de $m = 1$, pentru acelaşi pas de timp $k = 0.01$. Ţinând cont de (10.160), ne-am fi aşteptat ca rezultatele să fie bune indiferent de h, dacă $m = 1$. Schema (10.158) este exactă pentru $m = 1$, numai în măsura în care derivatele parţiale sunt suficient de precis calculate. Relaţia $\partial^p u/\partial x^p = \partial^p u/\partial t^p$ este obţinută pe baza faptului că egalitatea există pentru $p = 2$. După cum se observă, rezultate nesatisfăcătoare s-au obţinut acolo unde funcţia u variază rapid, pasul $h = 0.1$, fiind prea mare pentru a obţine o valoare suficient de precisă pentru derivatele parţiale u_{xx}. Acest fapt devine şi mai clar urmărind figura 10.21 unde sunt date rezultatele pentru $n = 8$, unde din nou rezultate bune se obţin pentru $h = 0.01$, în timp ce pentru celelalte valori rezultatele numerice sunt foarte proaste. Pentru $h = 0.03$, $k = 0.05$, $m = 1.67$, rezultatele sunt inacceptabile datorită faptului că schema (10.148) este instabilă.

BIBLIOGRAFIE

Abramowitz, M., Stegun, I.A., *Hanbook of Mathematical Functions*, Dover Publications, New York, 1965.

Atkinson, K., *Elementary Numerical Analysis*, Wiley, New York, 1985.

Bakhvalov, N. *Méthodes Numériques*, Editions MIR, 1976.

Berbente, C., Zancu, S., Tătăranu, C., Mitran, S., *Metode Numerice de Calcul pentru Aviaţie*, Litografia Institutului Politehnic Bucureşti, 1988.

Berbente, C., Mitran, S., Zancu, S., Pleter, O., Tătăranu, C., *Metode Numerice de Calcul şi Aplicaţii*, Litografia Institutului Politehnic Bucureşti, 1992.

Blumenfeld, M., *Calculul Barelor cu Calculatorul Numeric*, Editura Tehnică, Bucureşti, 1975.

Brent, R., *Algorithms for Minimization without Derivatives*.

Brigham, E.O., *The Fast Fourier Transform*, Prentice-Hall, Englewood Cliffs, 1974.

Bucur, C.M., Popeea, C.A., Simion, Gh., *Matematici Speciale - Calcul Numeric*, Editura Didactică şi Pedagogică, Bucureşti, 1983.

Carnahan, B., Luther, H.A., *Applied Numerical Methods*, John Wiley & Sons, New York, 1969.

Demidovitch, B., Maron, I., *Éléments de Calcul Numerique*, Editions MIR, Moscou, 1973.

Dennis, J.E., More, J., "Quasi-Newton Methods – Motivation and Theory", *SIAM Rev.*, 1984.

Dodescu, Gh., Toma, M., *Metode de calcul numeric*, Editura Didactică şi Pedagogică, Bucureşti, 1976.

Filciacova, V.P., *Transformări Conforme ale Domeniilor de Tip Special*, Editura Nancov Dumka, Kiev, 1972.

Golub, G.H. and van Loan, C.F., *Matrix Computations*, 2nd Ed., John Hopkins University Press, Baltimore, 1989.

Gourlay, A.R., Watson, G.A., *Computational Methods for Matrix Eigenproblems*, John Wiley & Sons, New York, 1973.

Halanay, A., Matematici, *Calcul Numeric şi Programare*, Litografia Universităţii Bucureşti, 1979.

Hardy, D.W., Walker, C., *Creating Documents with Scientific Word and Scientific WorkPlace*, Brooks-Cole, 1995.

Kincaid, D., Cheney, W., *Numerical Analysis – Mathematics of Scientific Computing*, Brooks-Cole Publishing Company, Pacific Grove, California, 1991.

Knuth, D. *The TEXbook*, Addison Wesley and AMS, 1986.

Lamport, L. , *LATEX – A Document Preparation System*, Addison-Wesley, Reading, Massachusetts, 1986.

MacCracken, D.D., Dorn, S.W., *Numerical Methods and Fortran Programming*, John Wiley & Sons, New York, 1965.

Mandelbrot, B. *The Fractal Geometry of Nature*, Freeman, New York, 1982.

Măruşter, St., *Metode numerice în rezolvarea ecuaţiilor nelineare*, Editura Tehnică, Bucureşti, 1981.

Micula, Gh., *Funcţii Spline şi Aplicaţii*, Editura Tehnică, Bucureşti, 1978.

Press, W.P., Teukolsky, S.A, Vetterling, W.T., Flannery, B.P., *Numerical Recipes in FORTRAN – The Art of Scientific Computing*, 2nd Edition, Cambridge University Press, 1995.

Schwartz, H.R., *Numerical Analysis of Symmetric Matrices*, Prentice Hall Inc., New York, 1973.

Sima, V., Varga, A., *Practica Optimizării Asistată de Calculator*, 1987.

Stănăşilă, O., *Analiza Matematică*, Editura Didactică şi Pedagogică, Bucureşti, 1981.

Wolfram, S., *Mathematica – A System for Doing Mathematics by Computer*, 2nd Ed., Addison-Wesley, 1991.