Prelegerea nr.1

I. NUMERE APROXIMATIVE

Sumar

- *Erori absolute și erori relative*
- *Propagarea și sursele erorilor*
- *Numere cu virgulă mobilă*
- *Aritmetica virgulei mobile și erorile de rotunjire*
- *Determinarea parametrilor unui sistem de calcul*
- *Efectul erorilor de rotunjire*

1.1. Erori absolute și erori relative

Notăm cu x_* valoarea aproximativă pentru numărul exact x. Dacă $x_*<x$, atunci spunem că x_* aproximează numărul x prin lipsă, iar dacă $x_*>x$, atunci aproximarea lui x prin x_* este prin adaos.

De obicei, în procesul de calcul se înlocuiește valoarea exactă (care, în caz general, nu este cunoscută) prin valoarea sa aproximativă. În felul acesta comitem o eroare. Expresia

$$\Delta(x_*) = |x - x_*|$$

poartă numele de ***eroare absolută***.

Eroarea absolută nu caracterizează suficient de bine precizia cu care se obțin rezultatele. Astfel, de exemplu, dacă $x = 1$ si $x_*=2$, atunci eroarea absolută $\Delta(x_*) = 1$ indică o precizie slabă a măsurării. Dacă $x = 10^{10} + 1$ iar $x_* = 10^{10}$, aceeași eroare absolută $\Delta(x_*) = 1$ caracterizează o precizie remarcabilă. Aceasta ne conduce la noțiunea de ***eroare relativă*** $\boldsymbol{\delta(x_*)}$ care reprezintă raportul dintre eroarea absolută și valoarea aproximativă, adică

$$\delta(x_*) = \frac{|x-x_*|}{|x_*|},$$

dacă $x_* \neq 0$.

În exemplele de mai sus erorile relative sunt egale cu 0.5 respectiv cu 10^{-10},

ceea ce confirmă buna precizie a măsurării în cazul al doilea.

Dacă se cunosc numerele x și x_*, atunci calculul erorii absolute şi relative este imediat. Dar, de obicei, în majoritatea cazurilor se cunoaște numai aproximarea x_*. De aceea se introduce noțiunea de margine (sau limită) a erorii absolute şi relative. Numărul pozitiv ε este o ***margine*** (sau o ***limită***) ***a erorii absolute*** a numărului aproximativ x_* dacă

$$|x - x_*| \leq \varepsilon,$$

iar numărul pozitiv r este o ***limită a erorii relative*** dacă

$$\frac{|x-x_*|}{|x_*|} \leq r.$$

Notația $x=x_* \pm \varepsilon$ semnifică întotdeauna faptul, că $|x - x_*| \leq \varepsilon$, adică

$$x_* - \varepsilon \leq x_* \leq x + \varepsilon.$$

Orice număr aproximativ x_* poate fi scris sub forma

$$x_* = c_1 10^m + c_2 10^{m-1} + \cdots + c_n 10^{m-n+1},$$

unde $c_1, c_2, \ldots, c_n$ sunt cifrele zecimale ale numărului aproximativ x_*. Se știe că zerourile de la începutul numărului servesc numai pentru a fixa poziția virgulei zecimale. Cifrele cuprinse intre prima și ultima cifră diferită de zero sau care indică ordinele păstrate în calcule se numesc ***cifre semnificative***.

Exemplu. Numărul aproximativ

$$x_* = 3 \cdot 10^1 + 6 \cdot 10^0 + 0 \cdot 10^{-1} + 5 \cdot 10^{-2} + 8 \cdot 10^{-3}$$

are cinci cifre semnificative, iar numărul

$$y_* = -(2 \cdot 10^{-3} + 8 \cdot 10^{-4} + 0 \cdot 10^{-5}) = -0.00280$$

are trei cifre semnificative (primele trei zerouri sunt nesemnificative).

Dacă mărimea erorii x_* nu depășește $0.5 \cdot 10^{-t}$ se spune că numărul aproximativ x_* are ***t cifre zecimale corecte***.

Dacă numărul aproximativ se scrie fără a indica limita erorii absolute, atunci în scrierea lui se consideră că toate cifrele sunt corecte. În acest caz zerourile de la sfârșitul numărului nu se aruncă.

De pildă numerele 0.0345 și 0.034500 sunt diferite; eroarea absolută a primului număr nu depășește 0.0001, iar eroarea absolută al celui de-al doilea număr este mai mică ca 10^{-6}.

Exemple: 0.010224 ± 0.000004 are cinci zecimale corecte și patru cifre semnificative; 0.001234±0.000006 are patru zecimale corecte și două cifre semnificative (deoarece valoarea maximă a numărului poate fi 0.001240, iar minima 0.001228 și deci ultimele două zecimale sunt nesigure).

Numărul de zecimale corecte ne permite să ne facem o idee despre mărimea erorii absolute în timp ce numărul de cifre semnificative ne dă o idee sumară despre mărimea erorii relative.

1.2. Propagarea și sursele erorilor

În procesul de calcul aproximativ eroarea se propagă de la o operație la alta. Fie date valorile aproximative x_* și y_* ale valorilor exacte x și y, afectate de erorile ε_x și ε_y, adică fie

$$x = x_* \pm \varepsilon_x \ , \quad y = y_* \pm \varepsilon_y.$$

Atunci avem

$$x_* - \varepsilon_x + y_x - \varepsilon_y \le x + y \le x_* + \varepsilon_x + y_x + \varepsilon_y$$

$$x_* + y_* - (\varepsilon_x + \varepsilon_y) \le x + y \le x_* + y_* + (\varepsilon_x + \varepsilon_y) \ ,$$

sau

$$x + y = x_* + y_* \pm (\varepsilon_x + \varepsilon_y)$$

În mod analog ca în cazul adunării, se va obține

$$x - y = x_* - y_* \pm (\varepsilon_x + \varepsilon_y).$$

Deci, ***la adunare sau scădere, marginea erorii absolute a rezultatului este dată de suma marginilor pentru erorile absolute ale termenilor***.

Se poate demonstra de asemenea ***că la înmulțire și împătrire marginile erorilor relative ale factorilor se adună.***

Fie acum date două numere pozitive x_* și y_* aproximativ egale , afectate de erorile absolute $\Delta(x_*)$ şi $\Delta(y_*)$. Atunci

$$\delta\ (x_* - y_*) = \frac{\Delta(x_* - y_*)}{|x_* - y_*|} \leq \frac{\Delta(x_*) + \Delta(y_*)}{|x_* - y_*|}$$

și, deci, eroarea relativă a diferenței poate fi destul de mare, dacă diferența $|x_* - y_*|$ este foarte mică. Aceasta ne arată că exactitatea relativă poate fi foarte slabă atunci când efectuăm diferența a două numere aproximativ egale.

Exemplu. Vom considera numerele aproximative

$$x=0.1234\pm0.5\cdot10^{-4} \quad \text{și} \quad y=0.1233\pm0.5\cdot10^{-4}$$

atunci $x-y = 0.0001\pm0.0001$ și marginea erorii este tot atât de mare ca şi estimarea rezultatului.

Acest fenomen poartă denumirea de ***anulare prin scădere*** sau de ***neutralizare a termenilor*** . Cele mai serioase erori care apar în calculele efectuate cu ajutorul calculatorului electronic sunt datorate acestui fenomen.

De câte ori este posibil neutralizarea termenilor se evită prin rescrierea formulelor de calcul sau prin alte schimbări în algoritm. De exemplu, o expresie de forma $(\alpha + \gamma)^2 - \alpha^2$ poate fi scrisă sub forma $\gamma(\gamma + 2\alpha)$, sau expresia

$$\frac{\sqrt{\alpha + \gamma} - \sqrt{\alpha}}{b}$$

sub forma

$$\frac{\gamma}{b(\sqrt{\alpha+\gamma}+\sqrt{\alpha})}.$$

Vom mai da un exemplu în care se arată cum poate fi evitată anularea prin scădere. Ecuația de gradul doi

$$x^2+1000.01\cdot x - 2.5245315=0$$

are una din rădăcini egală cu 0.0025245. Dacă calculăm rădăcina cu ajutorul formulei

$$x_1 = \frac{-1000.01 + \sqrt{(1000.01)^2 + 4 \cdot 2.5245315}}{2},$$

efectuând calculele cu opt cifre semnificative, se va obține

$$x_1 = \frac{-1000.0100 + 1000.0150}{2} = 0.0025.$$

Rezultatul obținut are numai două cifre corecte cu toate că radicalul de gradul doi a fost calculat cu opt cifre semnificative.

Dacă vom face calculul aceleiași rădăcini utilizând formula

$$x_1 = \frac{2 \cdot 2.524315}{1000.01 + \sqrt{(1000.01)^2 + 4 \cdot 2.5245315}} = 0.0025245$$

se va obține rezultatul exact.

Precizia calculelor numerice este criteriul cel mai eficient pentru alegerea metodelor de calcul. Analiza erorii dintr-un rezultat numeric este o chestiune esențială în orice calcul, fie că este executat manual, fie de un calculator. Cu toate performanțele calculatoarelor electronice, precizia rezultatelor este influențată de diferite erori. Se pot distinge trei surse de erori:

1. ***Erori provenite din simplificarea modelului fizic***, pentru a fi descris într-un model matematic; erori din măsurările inițiale sau din soluții aproximative ale altor probleme etc. Aceste tipuri de erori se numesc ***erori inerente***. Ele nu pot fi influențate de metoda de calcul.
2. ***Erori de metodă sau de trunchiere***. Majoritatea metodelor numerice necesită un număr infinit de operații aritmetice pentru a ajunge la soluția exactă a problemei. De aceea suntem nevoiți să trunchiem metoda după un număr finit de operații. Ceea ce omitem constituie eroarea de trunchiere.
3. ***Erori de rotunjire*** în datele de intrare în calcule și în datele de ieșire. Multe numere nu pot fi reprezentate exact printr-un număr dat de cifre. Dacă în calcule numerice trebuie să folosim numărul π, îl putem scrie 3.14, 3.14159 sau 3.1415926 etc. Nici un număr irațional nu poate fi reprezentat printr-un număr finit de cifre. Chiar și unele numere raționale nu au o reprezentare exactă. Numărul 1/3 poate fi scris ca 0.3333..., o succesiune a cifrei 3 la partea zecimală.

Datorită proprietăților constructive ale calculatoarelor electronice este necesară limitarea numărului cifrelor semnificative. Un exemplu instructiv este furnizat de

numărul rațional 1/10 care se folosește de multe ori ca dimensiune a "pasului" în foarte mulți algoritmi. În sistemul binar (care folosește pentru reprezentarea numerelor cifrele 0 și 1) fracția 1/10 are o reprezentare infinită 0.00011001100... . În calcule trebuie să ne mărginim la un număr finit de cifre semnificative. Când se adună de zece ori numărul care reprezintă o aproximație binară a numărului 1/10, rezultatul nu va fi egal exact cu unitatea.

1.3. Numere cu virgulă mobilă

Este bine cunoscut că pe majoritatea calculatoarelor moderne numerele reale se reprezintă cu ajutorul virgulei mobile . Un număr scris în virgulă mobilă este compus dintr-o fracție, numită ***mantisă*** și un întreg, numit ***exponent***. Deci,

$$x = \pm m \cdot \beta^e,$$

unde β este baza sistemului de numerație (binar, octal sau hexazecimal), m este mantisa numărului și e este exponentul, afectat de semn. Fracția m satisface

$$\frac{1}{\beta} \leq m < 1$$

și are forma

$$m = \frac{d_1}{\beta^1} + \frac{d_2}{\beta^2} + \cdots + \frac{d_t}{\beta^t},$$

unde numerele întregi $d_1, d_2, \ldots, d_t$, numite cifre, verifică inegalitățile

$$0 \leq d_i \leq \beta - 1, \quad i = 1,2, \ldots, t$$

și $L \leq e \leq U$.

Dacă prima cifră din mantisă este diferită de zero, atunci numărul reprezentat în virgulă mobilă se numește ***normalizat.***

Prin urmare, ***sistemul de calcul cu numere cu virgulă mobilă*** este o mulțime $F(\beta, t, L, U)$ caracterizată de patru parametri: baza β, precizia mașinii t și intervalul exponenților $[L, U]$.

F este o mulțime finită care conține

$$2 \cdot (\beta - 1) \cdot \beta^{t-1} \cdot (U - L + 1) + 1$$

numere. În fig.1.1 este reprezentată mulțimea F formată din 33 puncte pentru sistemul de calcul cu virgulă mobilă, ilustrativ cu următorii parametri:

$$\beta=2,\ t=3,\ L=-1,\ U=2.$$

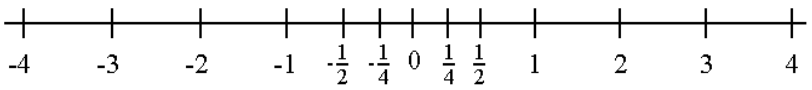

Fig.1.1. Sistemul de calcul F (2,3,-1,-2).

Mulțimea F nu poate reproduce oricât de detaliat structura continuă a numerelor reale. Mai mult, în general nu putem reprezenta în calculator numerele al căror modul depășește cel mai mare element al lui F sau care sunt mai mici în modul decât cel mai mic număr din F.

Mantisa m poate fi scrisă

$$m = \beta^{-t}(d_1\beta^{t-1} + d_2\beta^{t-2} + \cdots + d_t)$$

de unde rezultă, că dacă $d_1 \neq 0$, atunci maximul mărimii m este egal cu $1-\beta^{-t}$, ce corespunde cazului $m_i = \beta - 1$, $i=1, 2, \ldots, t$; valoarea minimală va fi β^{-1} și se obține pentru $d_1 = 1, d_2 = d_3 = \cdots = d_t = 0$.

Fie x un număr real care nu depășește limitele mulțimii F și $x \neq 0$; în calculator acest număr este reprezentat de numărul cu virgulă mobilă notat $fl(x)$, a cărui mantisă m_* se obține din mantisa m a lui x rotunjind-o la t cifre (de aceea spunem că precizia mașinii este t). Dacă se efectuează rotunjirea corectă atunci

$$|m - m_*| \leq \frac{1}{2}\beta^{-t}.$$

Eroarea relativă în $fl(x)$ este

$$\frac{|fl(x) - x|}{|x|} \leq \frac{1}{2}\beta^{1-t},$$

deoarece $\frac{|x_*-x|}{|x|} = \frac{|m-m_*|}{|m|}$ *și* $m \geq \frac{1}{\beta}$.

Numărul $\varepsilon_M = \frac{1}{2}\beta^{1-t}$ se numește ***unitatea (de rotunjire a) mașinii***. Efectuând rotunjirea corectă, numărul $fl(x)$ este cel mai apropiat element de x, care aparține lui

F. Dacă se folosește rotunjirea prin tăiere (se elimină compararea primei cifre neglijate), atunci $\varepsilon_M = \beta^{1-t}$ şi *fl(x)* este cel mai apropiat element din *F*, inferior lui *x* (vezi 1.4).

În afară de parametrul ε_M în practică sunt larg răspândiți încă doi parametri σ și λ: cel mai mic element pozitiv și elementul maximal al lui *F*. Pentru ilustrare vom analiza sistemul *F(*2, 3, -1, 1*)*. El conține numărul zero și toate numerele care au o exprimare binară de forma

$$x = \pm 0.1 d_2 d_3 \cdot 2^e,$$

unde -1 ≤ *e* ≤1, iar fiecare din cifrele d_2 și d_3 este 0 sau 1. Prin urmare, avem trei posibilități pentru valoarea exponentului *(*-1, 0 ori +1 *)* și patru pentru reprezentarea părților fracționare *(*0.100, 0.101, 0.110 și 0.111*)*. Mulțimea *F* constă din 2·1·4·3+1=25 de numere cu virgulă mobilă. În sistemul zecimal de numerație fracțiile de mai sus pot fi scrise respectiv 1/2, 5/8, 3/4 și 7/8 (de exemplu, ***fracția binară*** **0.101** devine $\mathbf{1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 1/2 + 1/8 = 5/8}$). Prin urmare în sistemul de calcul *F (*2,3,-1,1*)* cel mai mic element pozitiv este egal cu $\sigma = 1/2 \cdot 2^{-1} = 1/4$, iar cel mai mare element λ=7/8·2^1 =7/4. În fig.1.2 sunt reprezentate numerele pozitive ale mulțimii *F*.

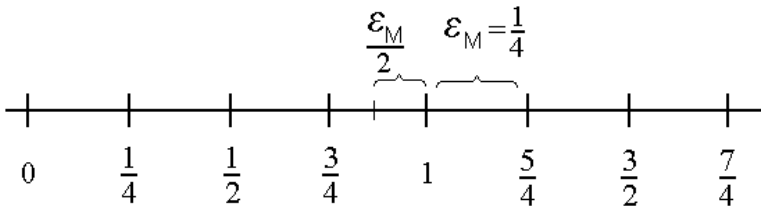

Fig.1.2. Numerele pozitive ale sistemului de calcul *F(*2,3, -1, 1*)*.

Parametrii σ şi λ se reprezintă prin parametrii β, *t*, *L* şi *U* după cum urmează

$$\sigma = \beta^{L-1}, \lambda = \beta^U (1 - \beta^{-t}).$$

Unitatea de rotunjire a mașinii ε_M se mai numește ***epsilon al mașinii*** și este cel mai utilizat parametru ce caracterizează un sistem de calcul dat. Acest parametru ne dă măsura de "discretizare" a sistemului *F* care are loc pentru tot intervalul numerelor nenule în virgulă mobilă. Deci distanta dintre numărul $x \in F$ și numărul

cel mai apropiat de el în sistemul dat nu e mai mică decât $\varepsilon_M|x|/\beta$ și nu e mai mare decât $\varepsilon_M|x|$ (numai dacă numărul x nu este situat în vecinătatea lui zero).

Din fig.1.2 se vede că în vecinătatea lui σ distanța dintre oricare două numere cu virgulă mobilă este mai mică decât σ. Prin urmare aceste numere nu pot fi obținute unul din altul prin adunare. Ele pot fi căpătate decât în calculul expresiilor aritmetice.

Un alt exemplu. Fie β=10, t=6 și L= -100. Atunci $\varepsilon_M=10^{-5}$, $\sigma=10^{-101}$. Între zero și σ nu există nici un număr ce aparține sistemului dat, în timp ce între σ și 10σ sunt 899999 de numere cu virgulă mobilă.

1.4. Aritmetica virgulei mobile și erorile de rotunjire

Pe mulțimea F, care o considerăm un model al mulțimii numerelor reale, se definesc operațiile aritmetice în așa fel încât ele să reproducă modul de efectuare al acestor operații de către calculatorul electronic.

Fie x și $y \in F$. Atunci suma lor exactă nu aparține, însă, întotdeauna mulțimii F. În exemplul în care F este o mulțime formată din 33 de puncte și cu parametrii β=2, t=3, L=-1, U=2 (vezi fig.1.1) putem constata aceasta luând x=5/4 și y=3/8, sau x=3/4 și y=7/8. În particular, în cazul al doilea vom avea

$$0.110\cdot 2^0+0.111\cdot 2^0=0.1101\cdot 2^1 \qquad (3/4+7/8=13/8)$$

Suma calculată nu aparține sistemului de calcul F, deoarece pentru reprezentarea părților fracționare a ei sunt necesare patru cifre binare. Prin urmare, operația de adunare trebuie modelată ea însăși în calculatorul electronic cu ajutorul unei aproximații a ei numită ***adunare cu virgulă mobilă*** (pe care o vom nota cu $\oplus$).

Dacă x și y sunt numere cu virgulă mobilă, iar numărul $x+y$ nu iese din limitele mulțimii F, atunci ar fi ideal ca

$$x \oplus y = fl(x+y).$$

Acest ideal este atins sau aproape atins de majoritatea calculatoarelor electronice. În sistemul de calcul F din exemplul considerat ne putem aștepta că $5/4 \oplus 3/8$ este egal cu 3/2 sau cu 7/4 (deoarece ambele elemente se află la distanță egală de 5/4+3/8).

Diferența dintre $x \oplus y$ și $x+y$ (pentru $x, y \in F$) reprezintă o eroare de rotunjire care se comite în cazul adunării cu virgulă mobilă. Proprietăți asemănătoare sunt adevărate și pentru celelalte operații aritmetice cu virgulă mobilă.

Revenind la exemplul dat de fig.1.1constatăm că 5/4+3/8 sau 3/4+7/8 nu aparțin lui *F* din cauza repartizării elementelor lui *F*. Pe de altă parte, suma 7/2+7/2 nu aparține lui *F* deoarece *7* este mai mare decât elementul maximal al lui *F*. Încercarea de a forma o astfel de sumă atrage la majoritatea calculatoarelor apariția unui ***semnal de depășire***, după care calculele se întrerup deoarece nu este posibil de dat o aproximație de sens pentru numerele care ies din limitele lui *F*.

Deși majoritatea sumelor $x+y$ cu $x \in F, y \in F$, aparțin ele însele lui *F*, foarte rar produsul (exact) obișnuit $x \cdot y$ aparține lui *F*, deoarece, de regulă, el are $2t$ sau $(2t—1)$ cifre semnificative . În afară de aceasta, depășirea este mai probabilă la înmulțire. În fine, în cazul înmulțirii cu virgulă mobilă, este posibilă apariția unui ***zero-mașină***, atunci când pentru $x \neq 0$ și $y \neq 0$, produsul $x \cdot y$ este nenul dar este mai mic, în modul, decât cel mai mic element pozitiv al lui *F* (apariția unui zero-mașină este posibilă și în cazul scăderii deși aceasta se întâmplă foarte rar).

Operațiile de adunare și înmulțire cu virgulă mobilă sunt comutative dar nu sunt asociative; de asemenea nu este adevărată nici distributivitatea.

În continuare ne vom ocupa de erorile de rotunjire. Eroarea de rotunjire reprezintă diferența $|fl(x)-x|$, unde *x* este un număr real, exponentul căruia aparține intervalului $[L, U]$ iar $fl(x)$ este numărul cu virgulă mobilă ce semnifică numărul dat *x* în memoria calculatorului. Pentru $x \neq 0$ eroarea relativă în $fl(x)$ se definește astfel

$$\delta(x) = \frac{|fl(x) - x|}{|x|}$$

și se îndeplinește condiția

$$\delta(x) \leq \varepsilon_M = \begin{cases} \beta^{1-t} & \text{, dacă are loc rotungirea prin trunchiere} \\ \frac{1}{2}\beta^{1-t} & \text{, dacă are loc rotungirea corectă .} \end{cases}$$

Într-adevăr, fie numărul întreg *e*,

$$L<e<U \text{ și } \beta^{e-1} \leq x \leq \beta^{e}$$

În intervalul $[\beta^{e-1}, \beta^{e}]$ numerele în virgulă mobilă sunt repartizate uniform cu pasul β^{e-t}. În cazul rotunjirii prin trunchiere *fl(x)* se află la depărtare de *x* ce nu depășește mărimea β^{e-t} iar în cazul rotunjirii corecte mărimea $\beta^{e-t/2}$, adică

$$|fl(x) - x| \leq \begin{cases} \beta^{e-t} & \text{, in cazul rotunjirii prin taiere} \\ \frac{1}{2}\beta^{e-t} & \text{, in cazul rotunjirii corecte.} \end{cases}$$

Deoarece $x \geq \beta^{e-1}$:

$$\delta(x) \leq \begin{cases} \frac{\beta^{e-t}}{\beta^{e-1}} = \beta^{1-t}, & \text{în cazul rotunjirii prin taiere} \\ \frac{\beta^{e-t}/2}{\beta^{e-1}} = \frac{1}{2}\beta^{1-t}, & \text{pentru rotunjirea corectă.} \end{cases}$$

Din cele de mai sus reiese că rotunjirea prin trunchiere a numerelor se efectuează mai repede decât rotunjirea corectă. Însă eroarea relativă este de două ori mai mare. În afară de aceasta eroarea rotunjirii prin trunchiere are permanent același semn (opus semnului numărului dat) ceea ce în cazul calculelor masive poate să conducă la o acumulare rapidă de erori. De aceea ar fi mai bine să se utilizeze rotunjirea corectă.

Pentru exemplificare vom analiza sistemul de calcul *F(*10,4,-50,50*)*. Numărului *x*=12.467 prin trunchiere îi va corespunde numărul în virgulă mobilă $fl(x)=0.1246\cdot10^{2}$ cu eroarea relativă de rotunjire

$$\delta(x)=0.007/12.467 \approx 0.00056 < \varepsilon_M=10^{-3}=0.001.$$

În cazul rotunjirii corecte vom avea $fl(x)=0.1247\cdot10^{2}$ și

$$\delta(x)=0.003/12.467 \approx 0.00024 < \varepsilon_M=10^{-3}/2=0.0005.$$

1.5. Determinarea parametrilor unui sistem de calcul

Pentru început vom analiza principiile generale de realizare a operațiilor aritmetice la un calculator. Fie două numere nenule reprezentate de numerele cu virgulă mobilă

$$x=m_x\cdot\beta^{p} \qquad \text{și} \qquad y=m_y\cdot\beta^{q}.$$

La adunare și scădere numerele se aduc la exponentul cel mai mare și apoi se adună (scad) mantisele:

$$z = x \pm y = \begin{cases} (m_x \pm m_y \cdot \beta^{-(p-q)}) \cdot \beta^p, & \text{daca } p>q, \\ (m_x \cdot \beta^{-(q-p)} \pm m_y) \cdot \beta^q, & \text{daca } p \le q. \end{cases}$$

Evident că mantisa rezultatului poate deveni mai mare ca unitatea, dar întotdeauna este mai mică decât doi. Dacă în urma operației de adunare mantisa sumei depășește unitatea, atunci trebuie normalizată aceasta prin deplasarea ei cu o poziție spre dreapta, adăugând o unitate la exponent pentru a compensa deplasarea.

Operațiile de înmulțire și împărțire se definesc astfel:

$$z = x \cdot y = (m_x \cdot m_y) \cdot \beta^{p+q},$$

$$z = \frac{x}{y} = \frac{m_x}{m_y}\beta^{(p-q)}, \qquad y \neq 0,$$

adică se înmulțesc (împart) mantisele numerelor *x* și *y*, se normalizează rezultatul, oprindu-se *t* cifre la mantisă și la partea exponențială se adună (scad) exponenții.

După cum vedem, deoarece calculul se face cu un număr anumit de cifre semnificative *t,* modul de rotunjire are o mare importantă. De modul de rotunjire depinde parametrul ε_M care caracterizează exactitatea relativă a sistemului de calcul *F*. Acest parametru poate fi definit ca cel mai mic număr pozitiv care adăugat în sistemul de calcul *F* la unitate dă în rezultat un număr cu virgulă mobilă ce aparține din nou lui *F* și este strict mai mare ca *1*, adică

$$fl(1+\varepsilon_M)>1 .$$

De exemplu, în sistemul de calcul *F(*10,4,-50,50*)* la rotunjirea prin trunchiere $\varepsilon_M = 10^{-3}=0.001$, deoarece

$$fl(1.+0.001)=fl(0.1001\cdot 10^1)=0.1001\cdot 10^1>1$$

și nu există un alt număr ε_M mai mic cu această proprietate . La rotunjirea corectă vom avea $\varepsilon_M = 0.005$, fiindcă

$$fl(1.+0.0005)=fl(0.10005\cdot 10^1)=0.1001\cdot 10^1>1 .$$

La majoritatea mașinilor de calcul se folosesc instrucțiuni speciale care determină modul de rotunjire. De aceea unul și același program poate da rezultate diferite în dependență de modul de rotunjire stabilit de translator. Un mare număr de compilatoare generează programul obiect ca să folosească tăierea. Acest tip de

rotunjire introduce o eroare mai mare decât regula obișnuită de rotunjire, care, după cum s-a mai spus, folosește mult timp de calculator dacă este aplicată la fiecare operație aritmetică.

Pentru a afla care mod de rotunjire e folosit în programele compilatoare este suficient să calculăm unitatea de rotunjire a mașinii ε_M. Dacă $\varepsilon_M = \beta^{1-t}$, atunci mașina de calcul generează modul de rotunjire prin tăiere, iar dacă $\varepsilon_M = \frac{1}{2}\beta^{1-t}$- generează modul de rotunjire corect.

De reținut că și ceilalți parametri (β, t, U, L, σ și λ) pot fi estimați direct la calculator utilizând tehnicile de programare (software) ai acesteia. Programele și argumentarea algoritmilor acestor programe poate fi găsită în literatura de specialitate, lucrări la care și-l trimitem pe cititor pentru a afla și alte detalii.

Trebuie de menționat importanța ordinii efectuării operațiilor aritmetice, deoarece într-o aritmetică a virgulei mobile nu au loc legile asociativă și distributivă. Pentru unele sisteme de calcul

$$(A + 1.0) - A \neq A + (1.0 - A)$$

Într-adevăr, fie de exemplu t=40 și β=2. Atunci în cazul rotunjirii corecte

$$(2^{40}+1.0)-2^{40}=2$$

în cazul rotunjirii prin tăiere

$$(2^{40}+1.0)-2^{40}=0,$$

iar dacă vom muta parantezele, pentru orice mod de rotunjire, vom avea

$$2^{40}+(1.0-2^{40})=1.$$

Deci putem sintetiza că operațiile aritmetice +, -, *, / se realizează la calculatoarele numerice după cum urmează (prin x este notat rezultatul exact al operației aritmetice):

1) dacă $\sigma \leq |x| \leq \lambda$, atunci rezultatul operației se rotunjește;

2) dacă $|x| < \sigma$, atunci rezultatul se anulează, adică apare un zero-mașină;

3) dacă $|x| > \lambda$, atunci calculele se întrerup și apare un semnal de depășire.

1.6. Efectul erorilor de rotunjire

Erorile introduse de calculator în procesul de calcul sunt erorile de rotunjire. Aceste erori se propagă de la o operație aritmetică la alta. Felul în care o eroare se propagă depinde şi de algoritmul de calcul. Aducem câteva exemple ilustrative.

1.6.1. Calculul mediei aritmetice

Media aritmetică dintre două numere *a* şi *b* poate fi calculată prin formulele :

$$c = \frac{a+b}{2} \tag{1.1}$$

$$c = a + \frac{b-a}{2} \tag{1.2}$$

Formula (1.1) necesită cu o operație de adunare mai puțin decât formula (1.2) dar din punct de vedere al exactității nu este întotdeauna cea mai bună. Într-adevăr, fie că calculele se efectuează într-o aritmetică zecimală *(*β=10*)* a virgulei mobile cu trei cifre semnificative *(t*=3*)* şi fie *a*=0.596 și *b*=0.600. Presupunem de asemenea că are loc rotunjirea corectă. Atunci

$$c = \frac{0.596 + 0.600}{2} = \frac{1.20}{2} = 0.600 \ ,$$

cu toate că valoarea corectă a lui *c* este egală cu 0.598. Efectuând calculele după formula (1.2), vom avea

$$c = 0.596 + \frac{0.600 - 0.596}{2} = 0.596 + \frac{0.004}{2} = 0.598 \ .$$

Este necesar să menționăm că în exemplul dat în care înclinăm pentru formula (1.2) numerele *a* și *b* au același semn.

Să considerăm acum un alt exemplu în care *t*=4 și se aplică rotunjirea prin tăiere. Fie *a*=-3.483 iar *b*=8.765. Folosind formula (1.1) obținem

$$c = \frac{-3.483 + 8.765}{2} = \frac{5.282}{2} = 2.641$$

și acest rezultat este exact. Calculul după formula (1.2) ne dă:

$$c = -3.483 + \frac{8.765 + 3.483}{2} = -3.483 + \frac{12.24}{2} =$$

$$= -3.483 + 6.120 = 2.637 \ .$$

Chiar dacă s-ar efectua rotunjirea corectă rezultatul obținut prin formula (1.2) ar fi diferit de valoarea exactă, deoarece în acest caz am avea c=2.642. Prin urmare, în exemplul de mai sus, unde a şi b sunt de semne diferite, vom prefera formula (1.1).

În concluzie, este necesar de a ne folosi de una din formulele (1.1) sau (1.2), în dependență de semnele lui a și b: dacă $sign(a) \neq sign(b)$ atunci $c=(a+b)/2$; în caz contrar $c=a+(b-a)/2$.

1.6.2. Evaluarea recurentă a unei integrale

Ne propunem să calculăm următoarea integrală :

$$I_n = \int_0^1 x^n e^{x-1} dx, \quad n = 1,2, \ldots$$

Integrând prin părți, obținem|

$$\int_0^1 x^n e^{x-1} dx = x^n e^{x-1}\Big|_0^1 - \int_0^1 n x^{n-1} e^{x-1} dx,$$

sau

$$I_n = 1 - n \cdot I_{n-1}\ , \quad n = 2,3, \ldots$$

Pentru n=1 avem

$$I_1 = \int_0^1 x e^{x-1} dx = \frac{1}{e}.$$

Fie aritmetica virgulei mobile cu β=10 şi t=6. Utilizând formula de recurență $I_n = 1 - n \cdot I_{n-1}$, obținem:

$I_1 \approx$0.367879, $I_6 \approx$0.127120,

$I_2 \approx$0.264242, $I_7 \approx$0.110160,

$I_3 \approx$0.207274, $I_8 \approx$0.118720,

$I_4 \approx$0.170904, $I_9 \approx$-0.068480,

$I_5 \approx$0.145480.

Deși $x^n e^x \geq 0,\ \forall x \in [0,1]$ se observă că pentru n=9 am obținut I_n<0, evident contradictoriu. Apariția rezultatului eronat se datorează algoritmului utilizat. Pentru

explicarea acestui fapt notăm cu E_n eroarea din I_n. Atunci E_1 este eroarea din calculul $1/e$=0.367879, deci $E_1 \approx 4.412 \cdot 10^{-7}$. Se vede din formula de recurență că $E_2 = -2E_1$, $E_3 = -3E_2 = (-2)\cdot(-3)E_1 = 3!E_1$ ş.a.m.d. Calculând E_n pentru n=9 vom găsi

$$E_9 \approx 9!*4.412*10^{-7} \approx 0.1601,$$

care este în mod evident mai mare decât I_9, întrucât valoarea exactă I_9 (cu trei cifre semnificative) este egală cu -0.06848+0.1601=0.0916. Se spune ***că algoritmul de calcul folosit este instabil***.

Deci, se vede că algoritmul constituie aici sursa erorilor. Pentru înlăturarea acestui neajuns vom rescrie formula de recurentă sub forma

$$I_{n-1} = \frac{1 - I_n}{n}, \quad n = 2,3, \ldots$$

În acest caz eroarea din I_n la fiecare pas se înmulțește la factorul $1/n$. Prin urmare, pornind cu I_N $(N \gg 1)$ am obţine I_{N-1}, I_{N-2},..., I_3, I_2 și eroarea de rotunjire s-ar micșora la fiecare iterație. Despre astfel de algoritme se spune că au o *stabilitate numerică.* Pentru a calcula valoarea inițială I_N observăm că

$$I_n \le \int_0^1 x^n dx = \frac{x^{n+1}}{n+1}\Big|_0^1 = \frac{1}{n+1},$$

și deci

$$\lim_{n\to\infty} I_n = 0.$$

Alegând I_{20}=0, se admite o eroare ce nu depășește 1/21. Atunci eroarea în I_{19} nu va întrece $(1/20)*(1/21) \approx 0.0024$. Această eroare se va micșora până la $4 \cdot 10^{-8}$ în timpul calculului I_{15} şi devine mai mică decât eroarea de rotunjire. Rezultatele obținute prin recurență inversă sunt următoarele:

$I_{20} \approx 0.0$,	$I_{14} \approx 0.0627322$,
$I_{19} \approx 0.0500000$,	$I_{13} \approx 0.0669477$,
$I_{18} \approx 0.0500000$,	$I_{12} \approx 0.0717733$,
$I_{17} \approx 0.0527778$,	$I_{11} \approx 0.0773523$,
$I_{16} \approx 0.0557190$,	$I_{10} \approx 0.0838771$,
$I_{15} \approx 0.0590176$,	$I_9 \approx 0.0916123$.

Deci este importantă alegerea corectă a algoritmului de calcul. Un algoritm de calcul se numește ***numeric stabil*** dacă aplicat unei probleme cu date inițiale "ușor perturbate" produce o soluție care aproape coincide cu soluția exactă (soluția problemei cu datele inițiale neperturbate).

1.6.3. Exemple de sisteme rău condiționate

Fie sistemul liniar

$$\begin{cases} 5x - 331y = 5, \\ 6x - 397y = 7. \end{cases}$$

Rezolvarea se poate face folosind algoritmul dat de regula lui Cramer

$$\Delta = \begin{vmatrix} 5 & -331 \\ 6 & -397 \end{vmatrix} = 1, \quad x = \begin{vmatrix} 5 & -331 \\ 7 & -397 \end{vmatrix} = 332, \quad y = \begin{vmatrix} 5 & 5 \\ 6 & 7 \end{vmatrix} = 5.$$

Soluția exactă a sistemului este x=332, y=5. Dacă se reia sistemul de mai sus, admițând însă o variație mică a coeficientului 5 de pe lângă x din prima ecuație, adică

$$\begin{cases} 5.01x - 331y = 5, \\ 6x - 397y = 7, \end{cases}$$

același procedeu de calcul ne conduce la soluția x= -111.7845..., y=-1.7070.... S-a produs o catastrofă! Despre un astfel de sistem se spune că este ***rău*** (sau ***prost***) ***condiționat***.

Un alt exemplu de sistem rău condiționat :

$$\begin{cases} x + 2y = 3, \\ 0.499x + 1.001y = 1.5. \end{cases}$$

Soluția exactă este x=y=1.0, ceea ce se poate verifica prin substituire. Dacă înlocuim ecuația a doua a sistemului dat cu ecuația

$$0.5x+1.001y=1.5,$$

atunci soluția devine x=3, y=0.

Sistemul de ecuații

$$\begin{cases} 14x + 13y - 66z = 1, \\ 12x + 11y - 13z = 1, \\ 11x + 10y + 4z = 1, \end{cases}$$

admite o soluție unică

$$x=1,\ y=-1,\ z=0.$$

Înlocuim elementele din partea dreaptă a sistemului *(*1, 1, 1*)* cu 1.001, 0.999 și 1.001 respectiv. Atunci, lucrând doar cu trei cifre semnificative, vom obține

$$x= -0.683,\ y=0.843,\ z=0.006.$$

În practică soluția exactă nu se cunoaște, deci nu putem avea certitudinea că soluția aproximativă calculată este suficient de aproape de soluția exactă. Acest lucru depinde, după cum se vede din cele câteva exemple de mai sus, atât de algoritmul de calcul, cât și de însăși problema considerată. Prin urmare***, rezolvarea numerică a unei probleme nu este o chestiune simplă, adică nu este suficient să facem niște calcule pentru a ajunge la soluția problemei. Mai este necesar de a studia și de a face aprecieri, privind condiționarea problemei și stabilitatea numerică a algoritmilor de calcul.***