

1.1.2 Reprezentarea aproximativă a numerelor. Scheme de rotunjire

☞ Oricare ar fi un număr $x \in \mathbb{R}, x \neq 0$, acesta poate fi scris sub forma:

$$x = f \cdot \beta^e + g \cdot \beta^{e-t}$$

- Se consider fracții normalizate: $1/\beta \leq |f| < 1, 0 \leq |g| < 1$

- Numerele mulțimii G se reprezintă, în general, aproximativ \square **operator de rotunjire**

$$fl: G \rightarrow F$$

◆ rotunjirea prin tăiere (trunchiere):

- $fl(x)$ este cel mai apropiat element $c \in F$ de $x \in G$ cu proprietatea $|c| \leq |x|$

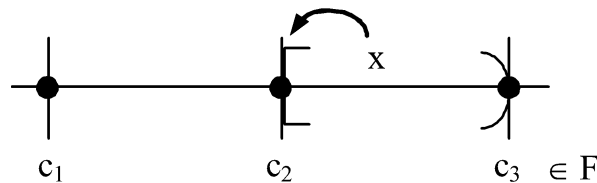


Fig. 1. Principiul rotunjirii prin tăiere (trunchiere): $fl(x) = c_2, \forall x \in [c_2, c_3)$

Exemplul:

Se consideră un sistem de numere în virgulă mobilă cu $\beta = 10$, $t = 4$. Fie numărul:

$$x = 12945,734$$

$$x = 0,12945734 \cdot 10^5 = 0,1294 \cdot 10^5 + 0,5734 \cdot 10^{5-4}$$

$$f = 0,1294; e = 5; g = 0,5734$$



$$fl(x) = 0,1294 \cdot 10^5 \longrightarrow 12940 \neq x$$

♦ rotunjirea simetrică:

- $fl(x)$ este cel mai apropiat element $c \in F$ de $x \in G$

$$fl(x) = \begin{cases} f \cdot \beta^e, & |g| < 0, \frac{\beta}{2} \\ f \cdot \beta^e \pm \beta^{e-t}, & |g| \geq 0, \frac{\beta}{2} \end{cases}$$

- semnul “+” se consideră pentru $f > 0$ și semnul “-” se consideră pentru $f < 0$

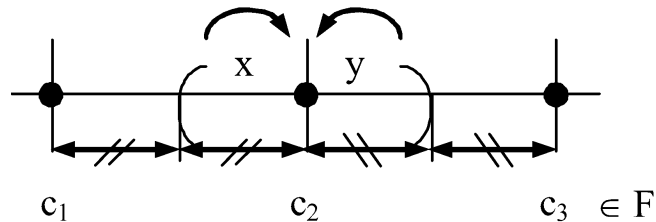


Fig. 2 Principiul rotunjirii simetrice:

$$fl(x) = c_2, fl(y) = c_2; \quad \forall x \in ((c_1 + c_2)/2, c_2), \forall y \in (c_2, (c_2 + c_3)/2)$$

Observație:

Neajunsul acestei maniere de reprezentare \square dacă numărul x este situat la jumătatea distanței dintre două numere consecutive din F , atunci $fl(x)$ poate lua oricare din cele două valori învecinate.

♦ rotunjirea uniformă:

$$fl(x) = \begin{cases} f \cdot \beta^e, & |g| < 0, \frac{\beta}{2} \\ f \cdot \beta^e \pm \beta^{e-t}, & |g| > 0, \frac{\beta}{2} \\ f \cdot \beta^e \pm \beta^{e-t}, & |g| = 0, \frac{\beta}{2}, \text{ ultima cifra } f - \text{ impară} \\ f \cdot \beta^e, & |g| = 0, \frac{\beta}{2}, \text{ ultima cifra } f - \text{ pară} \end{cases}$$

- semnul “+” se consideră pentru $f > 0$ și semnul “-” se consideră pentru $f < 0$;

- este adoptată de standardul IEEE

Exemplu:

Se consideră un sistem de numere în virgulă mobilă cu $\beta = 10$, $t = 4$ și rotunjire uniformă:

$$x = 12944,9942 = 0,1294 \cdot 10^5 + \underbrace{0,49942}_{|g| < 0.5} \cdot 10^{5-4} \longrightarrow \text{fl}(x) = 0,1294 \cdot 10^5 (= 12940 \neq x)$$

$$x = 129551 = 0,1295 \cdot 10^6 + \underbrace{0,51}_{|g| > 0.5} \cdot 10^{6-4} \longrightarrow \text{fl}(x) = 0,1295 \cdot 10^6 + 10^{6-4} (= 129600 \neq x)$$

$$x = 1297,5 = 0,1297 \cdot 10^4 + \underbrace{0,5}_{|g| = 0.5} \cdot 10^{4-4} \longrightarrow \text{fl}(x) = 0,1297 \cdot 10^4 + 10^{4-4} (= 1298 \neq x)$$

$$x = 1296,5 = 0,1296 \cdot 10^4 + \underbrace{0,5}_{|g| = 0.5} \cdot 10^{4-4} \longrightarrow \text{fl}(x) = 0,1296 \cdot 10^4 (= 1296 \neq x)$$

u.c.(f) -
impară

u.c.(f) -
pară

$$e_x = |x - \text{fl}(x)|$$

eroare absolută

$$\varepsilon_x = e_x / |x| \cong e_x / |\text{fl}(x)| = |x - \text{fl}(x)| / |\text{fl}(x)|$$

eroare relativă

$$\varepsilon_x = \frac{|x - fl(x)|}{|fl(x)|} = \frac{|f \cdot \beta^e + g \cdot \beta^{e-t} - fl(x)|}{|fl(x)|} \leq k \cdot \frac{1 \cdot \beta^{e-t}}{(1/\beta) \cdot \beta^e} \cong k \cdot \beta^{1-t}$$

- rotunjirea prin trunchiere $\rightarrow k = 1 \Rightarrow \varepsilon_x = \beta^{1-t}$ - cea mai mare spațiere relativă
- rotunjirea simetrică $\rightarrow k = 1/\beta \Rightarrow \varepsilon_x = \beta^{-t}$ - cea mai mică spațiere relativă

1.1.3 Operații elementare în virgulă mobilă

- se definesc operațiile elementare care au loc cu elementele unei mulțimi F de numere în virgulă mobilă

□ Adunarea/ scăderea

- oricare ar fi două numere x și y din mulțimea G, pentru care există fl(x) și fl(y) aparținând mulțimii F, numărului x + y i se asociază fl(x + y) conform algoritmului:

Pas 1: se reprezintă intern numerele x și y prin $fl(x)$ și, respectiv, $fl(y)$:

$$fl(x) = f_x \cdot \beta^{e_x}, \quad fl(y) = f_y \cdot \beta^{e_y}$$

Pas 2: dacă $e_x \neq e_y \rightarrow$ numărul cu exponent mai mic se aduce la o formă în care exponentul să fie egal cu cel al celuilalt termen \rightarrow denormalizare



deplasarea mantisei spre dreapta, inserând zerouri după virgulă

Pas 3: se adună mantisele și se normalizează rezultatul (dacă este necesar)

Pas 4: din rezultat se păstrează t cifre

Observații:

- ✓ Deplasarea mantisei spre dreapta cu o poziție determină creșterea exponentului cu o unitate, iar deplasarea spre stânga a mantisei cu o poziție determină scăderea exponentului cu o unitate.
- ✓ Scăderea se realizează la fel ca adunarea, cu deosebirea că mantisele se scad (scăderea reprezintă o adunare în care scăzătorul are semn schimbat).
- ✓ Adunarea/ scăderea în virgulă mobilă nu este asociativă

- erori determinate de adunarea/ scăderea în virgulă mobilă:
 - *omiterea catastrofală* □ apare atunci când se adună doi termeni și valoarea absolută a unui termen este mai mică decât precizia de reprezentare a celuilalt termen; în acest caz, rezultatul este dat de termenul cu valoare absolută mai mare;
 - *neutralizarea termenilor* □ apare atunci când se adună numere cu semne diferite și cu valori absolute apropiate; în acest caz, în mod eronat, rezultatul este nul.
- precizia calculelor numerice □ caracterizată de două mărimi constante (“constante de mașină”):

Definiție:

Epsilonul mașină pentru adunare (notat ε_m^+) - cel mai mic număr real pozitiv reprezentabil care schimbă, prin adunare, unitatea mașinii de calcul: $fl(1 + \varepsilon_m^+) > 1$

Epsilonul mașină pentru scădere (notat ε_m^-) - cel mai mare număr real pozitiv reprezentabil care schimbă, prin scădere, unitatea mașinii de calcul: $fl(1 - \varepsilon_m^-) < 1$

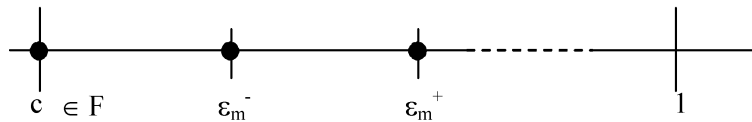


Fig. 3 Conceptul de epsilon mașină

$$fl(1 + \varepsilon_m^+) > 1, \quad fl(1 + c) = 1$$

$$fl(1 - \varepsilon_m^-) < 1, \quad fl(1 - c) = 1$$

□ Înmulțirea/ împărțirea

- oricare ar fi două numere x și y din mulțimea G , pentru care există $fl(x)$ și $fl(y)$ aparținând mulțimii F , numărului $x * y$ i se asociază $fl(x * y)$ conform algoritmului:

Pas 1: se reprezintă intern numerele x și y prin $fl(x)$ și, respectiv, $fl(y)$:

$$fl(x) = f_x \cdot \beta^{e_x}, \quad fl(y) = f_y \cdot \beta^{e_y}$$

Pas 2: se înmulțesc fracțiile ($f_x \cdot f_y$) și se adună exponenții ($e_x + e_y$)

Pas 3: se normalizează rezultatul (dacă este necesar)

Pas 4: din rezultat se păstrează t cifre

Observații:

- ✓ Împărțirea se realizează la fel ca înmulțirea, cu deosebirea că la pasul 2 fracțiile se împart și exponenții se scad.
- ✓ Înmulțirea/ împărțirea în virgulă mobilă nu este asociativă.

1.2 Propagarea erorilor în calculele numerice

- eroarea totală dintr-un calcul numeric este generată de trei surse principale de erori:
 - erorile inerente
 - provin din datele inițiale ale problemei de rezolvat (date care pot fi rezultatele unor măsurători experimentale sau ale altor calcule anterioare)
 - erorile provenite din faptul că se lucrează cu un model aproximativ al fenomenului real implicat în problema de rezolvat.
 - erorile de metodă - datorate metodei numerice utilizate
 - de exemplu, trunchierea unei serii sau construirea unui număr finit de termeni ai unui șir (determinarea limitei)
 - erorile de reprezentare sunt datorate posibilității efective, limitate, de a reprezenta numerele în calculatorul numeric
- într-un calcul numeric erorile se propagă de la o operație la alta, pe măsura ce numărul operațiilor dintr-un calcul crește □ erorile se pot acumula excesiv de mult □ valoare total incorectă a rezultatului unui calcul

👉 reguli generale pentru mărirea preciziei calculelor:

1. când se adună sau se scad numere, este recomandabil să se înceapă cu cele mai mici în valoare absolută, separat pentru cele negative și separat pentru cele pozitive;
2. dacă este posibil, este recomandabil să se evite scăderea a două numere aproximativ egale; o expresie care conține o astfel de scădere poate fi rescrisă
3. dacă regulile generale anterior enunțate nu se pot aplica, atunci se va urmări minimizarea numărului de operații aritmetice implicate

Exemplu:

Se consideră următoarele relații de calcul:

$$h = 1/2, \quad x = 2/3 - h, \quad y = 3/5 - h, \quad e = x + x + x - h,$$

$$f = y + y + y + y + y - h, \quad q = f/e.$$

calcul exact

calcul aproximativ

$$h = 1/2, \quad x = 1/6, \quad y = 1/10, \quad e = 0, \quad f = 0, \quad q = ?$$

Explicații:

$$e \neq 0, \quad f \neq 0, \quad q = \text{finit}$$

$2/3 = 0,(6)$ - are o infinitate de cifre \square se reprezintă aproximativ

$3/5 = 0,6 \longrightarrow 0,6_{\text{zece}} = 0,(1001)_{\text{doi}}$ \square o infinitate de cifre în binar \square se reprezintă aproximativ

1.3 Natura problemelor de calcul și caracterizarea algoritmilor

1.3.1 Natura problemelor de calcul

□ se consideră:

- P - o problemă de calcul;
- D - set de date de intrare exacte;
- D^* - set de date de intrare ușor perturbate față de setul de date D ;
- $P(D)$ - soluția matematică exactă, obținută lucrând cu datele exacte D ;
- $P(D^*)$ - soluția matematică exactă, obținută lucrând cu datele perturbate D^* .

Definiție:

Problema de calcul, P , se spune că este bine condiționată dacă datele exacte ale problemei, D , și datele perturbate ale problemei, D^ , fiind apropiate într-un anumit sens, atunci și soluția exactă matematică a problemei de calcul corespunzătoare datelor exacte, $P(D)$ este apropiată, într-un anumit sens de soluția exactă matematică a problemei de calcul corespunzătoare datelor perturbate. Altfel, se spune că problema de calcul este prost (rău) condiționată.*

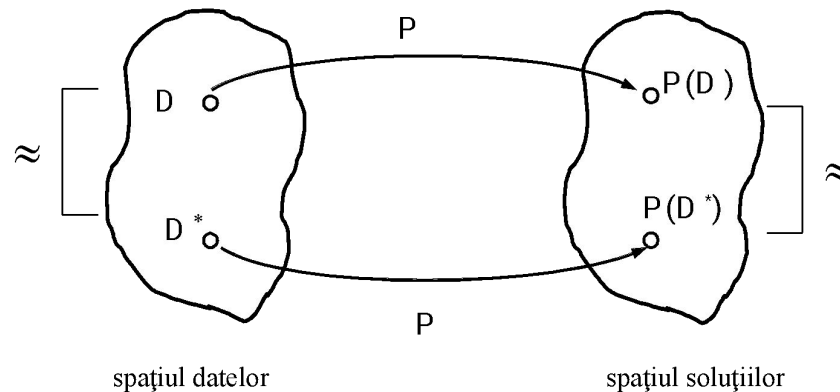


Fig. 4 Conceptul de problemă bine condiționată: $\mathcal{D} \cong \mathcal{D}^* \rightarrow \mathcal{P}(\mathcal{D}) \cong \mathcal{P}(\mathcal{D}^*)$

☞ o problemă este bine condiționată dacă mici perturbații în una sau toate datele inițiale ale problemei conduc la mici perturbații în datele de ieșire (rezultate sau soluții).

☞ numărul de condiție al problemei P □ raportul dintre eroarea absolută în soluțiile problemei de calcul și eroarea absolută în datele de intrare ale problemei de calcul:

$$k(P) = e_p / e_D$$

unde $e_p = |P(D) - P(D^*)|$ și $e_D = |D - D^*|$.

Observație:

Dacă numărul de condiție este egal cu 1 sau are valori apropiate de 1, se spune că problema este bine condiționată. Dacă numărul de condiție este mare sau foarte mare, atunci erorile din datele inițiale sunt amplificate în soluția problemei de calcul exactă matematic, problema de calcul fiind prost condiționată.

1.3.2 Caracterizarea algoritmilor

□ se consideră:

- P - o problemă de calcul;
- D - set de date de intrare exacte;
- D^* - set de date de intrare ușor perturbate față de setul de date D ;
- $P(D)$ - soluția matematică exactă, obținută lucrând cu datele exacte D ;
- $P(D^*)$ - soluția matematică exactă, obținută lucrând cu datele perturbate D^* ;
- P^* - algoritmul care implementează problema de calcul P .

☞ De exemplu, considerând că P este o problemă bine condiționată și că P^* implementează exact soluția P , totuși $P(D) \neq P^*(D)$ datorită aritmeticii în virgulă mobilă.

Definiție:

Un algorithm, P^* , se numește stabil numeric, dacă datele exacte și datele perturbate ale problemei de calcul P fiind apropiate într-un anumit sens, atunci și soluția exactă matematic corespunzătoare setului de date perturbate $P(D^*)$ este apropiată, într-un anumit sens, de soluția algoritmului corespunzătoare setului de date exacte $P^*(D)$. Altfel, algorithmul se spune că este instabil din punct de vedere numeric.

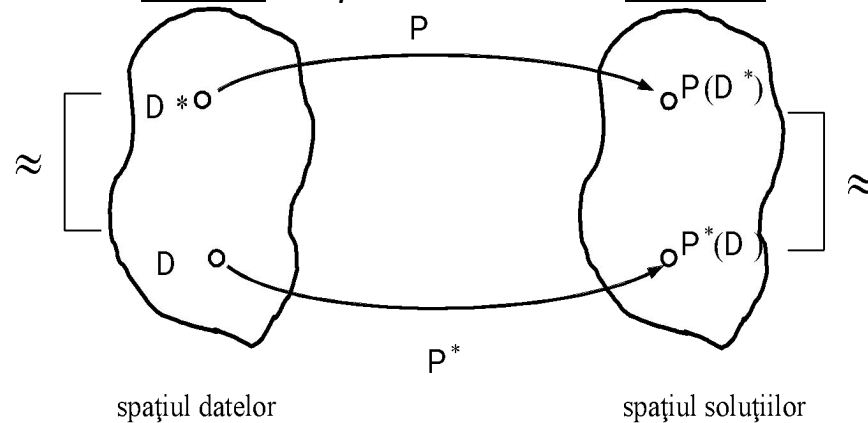


Fig. 5 Conceptul de algorithm stabil numeric: $D \cong D^* \rightarrow P(D^*) \cong P^*(D)$

☞ Altfel spus, erorile din datele de intrare sunt micșorate de un algorithm stabil numeric, un algorithm instabil numeric amplificându-le.

Observații:

- (a) Nu se poate aștepta ca un algoritm stabil numeric să rezolve o problemă prost condiționată cu o precizie mai mare decât a datelor de intrare.
- (b) Un algoritm instabil numeric furnizează, de regulă, rezultate eronate chiar pentru probleme bine condiționate.

Definiție:

Un algoritm numeric se spune că este general dacă este aplicabil pentru un domeniu larg de date de intrare.

Definiție:

Un algoritm se spune că este sigur în funcționare, dacă are prevăzut un mecanism care să avertizeze atunci când erorile au crescut excesiv de mult.

Observație:

Un algoritm instabil numeric poate fi sigur în funcționare, dacă este capabil să detecteze instabilitatea numerică.