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 fracţii normalizate: $1/\beta \leq |f| < 1, \quad 0 \leq |g| < 1$

- Numerele mulţimii G se reprezintă, în general, aproximativ ⇨ ***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 $|c| \leq |x|$

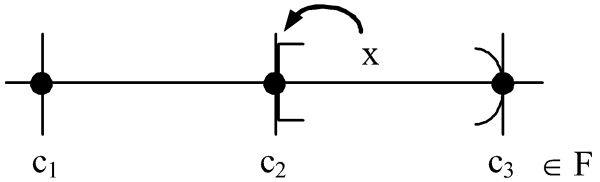

Fig. 1. Principiul rotunjirii prin tăiere (trunchiere): $fl(x) = c_2, \forall x \in [c_2, c_3)$

Exemplul:

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

$$x = 12945{,}734$$

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

$$f = 0{,}1294;\ e = 5;\ g = 0{,}5734$$

$$\Downarrow$$

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

- <u>*rotunjirea simetrică*</u>:

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

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

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

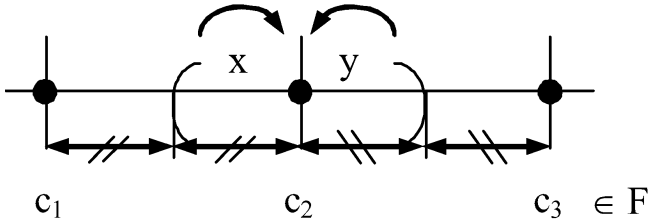

Fig. 2 Principiul rotunjirii simetrice:

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

<u>Observaţie:</u>

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

- *<u>rotunjirea uniformă</u>*:

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

- semnul “+” se consideră pentru f > 0 şi semnul “-“ se consideră pentru f < 0;
- este adoptată de standardul IEEE

Exemplu:

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

$|g| < 0.5$

$$x = 12944{,}9942 = 0{,}1294 \cdot 10^5 + 0{,}49942 \cdot 10^{5-4} \longrightarrow \mathrm{fl}(x) = 0{,}1294 \cdot 10^5 \ (= 12940 \neq x)$$

$|g| > 0.5$

$$x = 129551 = 0{,}1295 \cdot 10^6 + 0{,}51 \cdot 10^{6-4} \longrightarrow \mathrm{fl}(x) = 0{,}1295 \cdot 10^6 + 10^{6-4} \ (= 129600 \neq x)$$

$|g| = 0.5$

$$x = 1297{,}5 = 0{,}1297 \cdot 10^4 + 0{,}5 \cdot 10^{4-4} \longrightarrow \mathrm{fl}(x) = 0{,}1297 \cdot 10^4 + 10^{4-4} \ (= 1298 \neq x)$$

u.c.(f) - impară

$|g| = 0.5$

$$x = 1296{,}5 = 0{,}1296 \cdot 10^4 + 0{,}5 \cdot 10^{4-4} \longrightarrow \mathrm{fl}(x) = 0{,}1296 \cdot 10^4 \ (= 1296 \neq x)$$

u.c.(f) - pară

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

eroare absolută

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

eroare relativă

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

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

1.1.3 Operaţii elementare în virgulă mobilă

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

 <u>Adunarea/ scăderea</u>

- oricare ar fi două numere x şi y din mulţimea G, pentru care există fl(x) şi fl(y) aparţinând mulţimii F, numărului x + y i se asociază fl(x + y) conform algoritmului:

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

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

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

⇩

deplasarea mantisei spre dreapta, inserând zerouri după virgulă

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

Pas 4: din rezultat se păstrează t cifre

Observaţii:

- ✔ Deplasarea mantisei spre dreapta cu o poziţie determină creşterea exponentului cu o unitate, iar deplasarea spre stânga a mantisei cu o poziţie 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 mobile:
  - *omiterea catastrofală* apare atunci când se adună doi termeni şi 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 şi cu valori absolute apropiate; în acest caz, în mod eronat, rezultatul este nul.

- *precizia calculelor numerice* caracterizată de două mărimi constante ("constante de maşină"):

Definiţie:

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

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

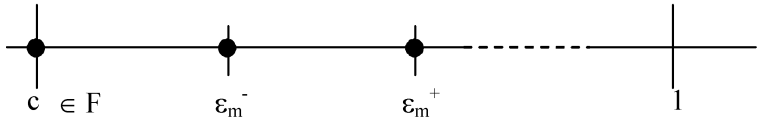

Fig. 3 Conceptul de epsilon maşină

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

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

- <u>Înmulţirea/ împărţirea</u>

- oricare ar fi două numere x şi y din mulţimea G, pentru care există fl(x) şi fl(y) aparţinând mulţimii F, numărului x * y i se asociază fl(x * y) conform algoritmului:

<u>Pas 1</u>: se reprezintă intern numerele x şi y prin fl(x) şi, respectiv, fl(y):

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

<u>Pas 2</u>: se înmulţesc fracţiile $(f_x \cdot f_y)$ şi se adună exponenţii $(e_x + e_y)$

<u>Pas 3</u>: se normalizează rezultatul (dacă este necesar)

<u>Pas 4</u>: din rezultat se păstrează t cifre

<u>Observaţii</u>:

✔ Împărţirea se realizează la fel ca <u>înmulţirea</u>, cu deosebirea că la pasul 2 fracţiile se impart şi exponenţii se scad.

✔ <u>Înmulţirea/ împărţirea</u> 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 iniţiale 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 şir (determinarea limitei)
  - *erorile de reprezentare* sunt datorate posibilităţii effective, limitate, de a reprezenta numerele în calculatorul numeric
    - se manifestă în datele iniţiale, intermediare şi în cele de ieşire (rezultatele finale)

- într-un calcul numeric erorile se propagă de la o operaţie la alta, pe măsură ce numărul operaţiilor dintr-un calcul creşte erorile se pot acumula excesiv de mult valoare total incorectă a rezultatului unui calcul

☞ *<u>reguli generale pentru mărirea preciziei calculelor</u>*:

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

Exemplu:

Se consideră următoarele relaţii 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

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

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

Explicaţii:

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

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

1.3 Natura problemelor de calcul şi 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 uşor perturbate faţă de setul de date D;
  - $P(D)$ - soluţia matematică exactă, obţinută lucrând cu datele exacte D;
  - $P(D^*)$ – soluţia matematică exactă, obţinută lucrând cu datele perturbate D^*.

Definiţie:

<u>Problema de calcul</u>, P, se spune că este <u>bine condiţionată</u> dacă datele exacte ale problemei, D, şi datele perturbate ale problemei, D^*, fiind apropiate într-un anumit sens, atunci şi soluţia exactă matematic a problemei de calcul corespunzătoare datelor exacte, $P(D)$ este apropiată, într-un anumit sens de soluţia exactă matenatic a problemei de calcul corespunzătoare datelor perturbate. Altfel, se spune că problema de calcul este prost (rău) condiţionată.

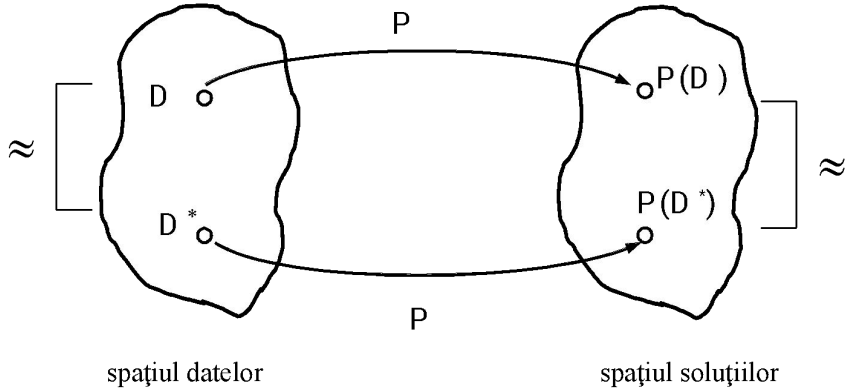

Fig. 4 Conceptul de problemă bine condiţionată: $\mathcal{D} \cong \mathcal{D}^* \rightarrow \mathcal{P}(\mathcal{D}) \cong \mathcal{P}(\mathcal{D}^*)$

☞ o *problemă este bine condiţionată* dacă mici perturbaţii în una sau toate datele iniţiale ale problemei conduc la mici perturbaţii în datele de ieşire (rezultate sau soluţii).

☞ *numărul de condiţie al problemei* $\mathcal{P}$ □ raportul dintre eroarea absolută în soluţiile problemei de calcul şi eroarea absolută în datele de intrare ale problemei de calcul:

$$k(\mathcal{P}) = e_{\mathcal{P}} / e_{\mathcal{D}}$$

unde $e_{\mathcal{P}} = |\mathcal{P}(\mathcal{D}) - \mathcal{P}(\mathcal{D}^*)|$ şi $e_{\mathcal{D}} = |\mathcal{D} - \mathcal{D}^*|$.

Observaţie:

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

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 uşor perturbate faţă de setul de date D;
  - $P(D)$ - soluţia matematică exactă, obţinută lucrând cu datele exacte D;
  - $P(D^*)$ – soluţia matematică exactă, obţinută lucrând cu datele perturbate D^*;
  - P^* - algoritmul care implementează problema de calcul P.

☞ De exemplu, considerând că P este o problemă bine condiţionată şi că P^* implementează exact soluţia P, totuşi $P(D) \neq P^*(D)$ datorită aritmeticii în virgulă mobilă.

Definiţie:

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

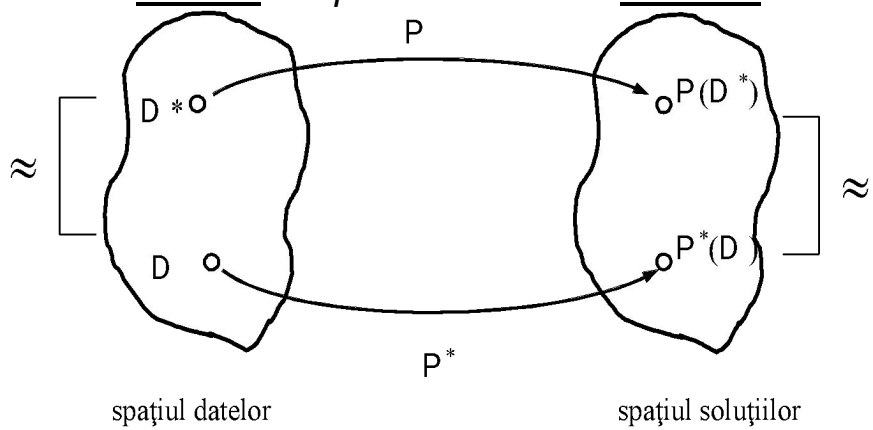

Fig. 5 Conceptul de algoritm stabil numeric: $\mathcal{D} \cong \mathcal{D}^* \rightarrow \mathcal{P}(\mathcal{D}^*) \cong \mathcal{P}^*(\mathcal{D})$

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

Observaţii:

(a) Nu se poate aştepta ca un algoritm stabil numeric să rezolve o problemă prost condiţionată cu o precizie mai mare decât a datelor de intrare.

(b) Un algoritm instabil numeric furnizează, de regulă, rezultate eronate chiar pentru probleme bine condiţionate.

Definiţie:

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

Definiţie:

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

Observaţie:

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