REVUE FRANÇAISE D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE, SÉRIE ROUGE

JEAN-PIERRE CROUZEIX

Brèves communications. Étude de l'erreur d'arrondi dans la résolution de systèmes linéaires par des méthodes itératives

Revue française d'informatique et de recherche opérationnelle, série rouge, tome 5, n° 2 (1971), p. 100-107.

<http://www.numdam.org/item?id=M2AN_1971__5_2_100_0>

R.I.R.O.
(5e année, R-2, 1971, p. 100-107)

ÉTUDE DE L'ERREUR D'ARRONDI DANS LA RÉSOLUTION DE SYSTÈMES LINÉAIRES PAR DES MÉTHODES ITÉRATIVES

par Jean-Pierre CROUZEIX (1)

Sommaire. — *Nous allons comparer du point de vue de l'erreur d'arrondi les trois méthodes suivantes : Jacobi, Gauss-Seidel et Surrelaxation. Après avoir montré que la méthode de Jacobi est la meilleure des trois à ce point de vue, nous proposons un algorithme dans lequel un test basé sur les erreurs d'arrondis nous permet d'approcher le plus près possible la solution.*

1. LES ERREURS D'ARRONDIS ET LEUR PROPAGATION

Supposons que nous travaillons en base λ avc p chiffres de mantisse tout nombre « machine » $x \neq 0$ s'écrit alors

$$x = \pm\ 0, c_1, c_2 \ldots c_p \lambda^n$$

$$\text{avec} \begin{cases} c_i = 0, 1, \ldots, \lambda - 1 \qquad \text{pour} \qquad i = 1, 2, \ldots, p \\ c_1 \neq 0 \end{cases}$$

On pose $\lambda^n = \gamma(x)$ (facteur de cadrage)

Soit T une des quatre opérations élémentaires, a et b deux nombres « machines » et $\varepsilon(x)$ l'erreur commise dans le calcul de $x = aTb$.

On a $\varepsilon(x) = aTb - a\bar{T}b$ ou $\bar{T}$ désigne l'opération machine associée à T. $\varepsilon(x)$ peut s'écrire sous la forme

$$\varepsilon(x) = s\eta.\ (x) \cdot \gamma(x) \text{ où } 0 \leqslant \eta(x) < \lambda^{-p}$$

$s = \pm\ 1$ (s dépend du signe de x et de T).

(1) Département de Mathématiques Appliquées, Université de Clermont-Ferrand.

Nous ferons alors les hypothèses suivantes (cf. [1]) :

a) $\eta(x)$ est uniformément réparti sur $[0, \lambda^{-p}]$

b) les erreurs d'arrondis intervenant sur des opérations différentes sont indépendantes.

On a alors :

$$\hat{\varepsilon}(x) = \text{var}\,(\varepsilon(x)) = \frac{\lambda^{-2p}}{12}\gamma^2(x) \qquad \text{(E.A.)}$$

$\gamma(x)$ est une fonction dont la définition en langage machine est très simple et que l'on aura intérêt à programmer en ce langage ou en assembleur. Si ceci n'est point réalisable au lieu d'écrire $\gamma(x)$ en langage scientifique (Algol ou Fortran) ce qui deviendrait trop coûteux, nous utiliserons l'approximation de var $(\varepsilon(x))$ suivante

$$\hat{\varepsilon}(x) = \text{var}\,(\varepsilon(x)) = \lambda^{-2p}\left[\frac{\lambda}{3} - \frac{\lambda^2}{4(\lambda-1)^2}(\text{Log}\,(\lambda))^2\right]x^2 \qquad \text{(E.R.}$$

obtenue en écrivant que $\dfrac{\varepsilon(x)}{x} = \eta(x)\dfrac{\gamma(x)}{x}$ est le quotient des 2 variables aléatoires suivantes :

$\eta(x)$ qui est uniformément répartie sur $[0,\lambda^{-p}]$

et $\dfrac{x}{\gamma(x)}$ qui est uniformément répartie sur $[\lambda^{-1}, 1]$

qu'on peut supposer indépendantes.

Dans tout ce qui suit :

* $\varepsilon(x)$ désigne l'erreur d'arrondi dans le dernier calcul élémentaire qui donne x,

* $e(x)$ l'erreur commise dans le calcul de x compte tenu de toutes les erreurs d'arrondis précédentes.

2. COMPARAISON DE DEUX ALGORITHMES ITERATIFS

Soient $x_{n+1} = \varphi_1(x_n)$ et $y_{n+1} = \varphi_2(y_n)$ deux algorithmes itératifs convergeant vers la même valeur α.

En raison des erreurs d'arrondis lorsque nous appliquons φ_1 et φ_2 à α, nous n'obtiendrons point exactement α mais deux valeurs approchées

$$\alpha_1 = \overline{\varphi_1(\alpha)} \text{ et } \alpha_2 = \overline{\varphi_2(\alpha)}.$$

Il est naturel de dire que l'algorithme itératif φ_1 est d'autant plus sensible aux erreurs d'arrondis que la quantité $|\alpha - \bar{\varphi}_1(\alpha)|$ est plus grande. Comme

nous ne pouvons connaître exactement cette quantité nous prendrons pour mesurer l'influence de l'erreur d'arrondi

$$\widehat{e(\varphi_1(\alpha))} = \text{var}\,(e(\varphi_1(\alpha))) = \text{var}\,\overline{(\varphi_1(\alpha)} - \alpha) = \text{var}\,(\overline{\varphi_1(\alpha)} - \varphi_1(\alpha))$$

On dira alors que l'algorithme φ_1 est plus sensible aux erreurs d'arrondis que φ_2 si :

$$\text{var}\,(e(\varphi_1(\alpha)) > \text{var}\,(e(\varphi_2(\alpha))$$

3. ETUDES DE METHODES DE JACOBI, GAUSS-SEIDEL ET SURRELAXATION

Soit à résoudre le système linéaire $AX = B$ (S.L.)

on écrit A sous la forme $A = L + D + S$

$$\text{avec} \begin{cases} d_{ii} \neq 0\ \forall_i;\ l_{ij} = 0 \text{ si } j \geqslant i;\ s_{ij} = 0 \text{ si } j \leqslant i \\ d_{ij} = 0 \text{ si } i \neq j \end{cases}$$

on a alors les formules classiques

$$DX^{k+1} = B - (L + S)X^k \qquad \text{(J.) pour Jacobi}$$

$$DX^{k+1} = B - LX^{k+1} - SX^k \qquad \text{(G.S.) pour Gauss-Seidel}$$

$$DX^{k+1} = \omega(B - LX^{k+1} - SX^k) + (1 - \omega)DX^k$$

(*S.R.*) pour la surrelaxation

A. Méthode de Jacobi

Appliquons l'algorithme itératif à la solution X du système linéaire pour faciliter le calcul d'erreur nous poserons $Y = \varphi(X)$ (si nous n'avions pas d'erreur d'arrondi nous aurions $Y = X$).

On obtient pour le calcul de la i$^{\text{ème}}$ composante

$$y_i = \left(b_i - \sum_{i \neq j} a_{ij}x_j\right)/a_{ii}$$

Les calculs se faisant dans l'ordre naturel et en linéarisant les erreurs on obtient :

$$e(y_i) = -\left[\sum_{j \neq i} \varepsilon\,(a_{ij}x_j) + \sum_{j \neq i} \varepsilon(u_{ij})\right]/a_{ii} + \varepsilon\,(y_i) \qquad \text{(J.E.)}$$

avec $u_{ij} = b_i - \sum_{\substack{l=1 \grave{a} j \\ i \neq l}} a_{il}x_l$

Il est donc facile, compte tenu de b et des formules (EA) ou (ER) d'obtenir var $[e(y_i)]$, les différents ε intervenant représentant des variables aléatoires indépendantes :

$$\widehat{e_j(y_i)} = \text{var}\,(e\,(y_i)) = \left[\sum_{j\neq i} \widehat{\varepsilon\,(a_{ij}x_j)} + \sum_{j\neq i} \widehat{\varepsilon\,(u_j)}\right]/a_{ii}^2 + \widehat{\varepsilon\,(y_i)}$$

B. Méthode de Gauss-Seidel

(G.S.) s'écrit sur la i$^{\text{ème}}$ composante

$$y_i = \left(b_i - \sum_{j=1}^{i-1} a_{ij}y_j - \sum_{j=i+1}^{n} a_{ij}x_j\right)/a_{ii}$$

d'où

$$e\,(y_i) = \left[-\sum_{j=1}^{i-1} a_{ij}\,e\,(y_j)\right]/a_{ii}$$

$$+\left[-\sum_{j=1}^{i-1} \varepsilon\,(a_{ij}y_j) - \sum_{j=i+1}^{n} \varepsilon\,(a_{ij}x_j) + \sum_{j=i} \varepsilon\,(u_{ij})\right]/a_{ii} + \varepsilon\,(y_i) \quad \text{(G.S.E.)}$$

avec $u_{ij} = b_i - \sum_{\substack{l\leqslant j\\ l<i}} a_{il}y_l - \sum_{i<l\leqslant j} a_{il}x_l$,

d'où

$$\hat{e}_{G.S.}\,(y_i) = \left(\sum_{j=1}^{i-1} a_{ij}^2\,\widehat{e_{GS}\,(y_j)}\right)/a_{ii}^2$$

$$+\left[\sum_{j=1}^{i-1} \widehat{\varepsilon\,(a_{ij}y_j)} + \sum_{j=i+1}^{n} \widehat{\varepsilon\,(a_{ij}x_j)} + \sum_{j\neq i} \widehat{\varepsilon\,(u_j)}\right]/a_{ii}^2 + \widehat{\varepsilon\,(y_i}$$

Les quantités y_j et x_j sont très voisines, donc il en est de même pour les quantités u_j intervenant dans Jacobi et celles intervenant dans Gauss-Seidel. Il s'ensuit que :

$$\widehat{\varepsilon\,(a_{ij}y_j)} \simeq \widehat{\varepsilon\,(a_{ij}x_j)}$$

et que $\widehat{\varepsilon\,(u_j)}$ de Jacobi $\simeq \widehat{\varepsilon\,(u_j)}$ de Gauss-Seidel.

On a alors

$$e_{GS}\,(y_i) = \left(\sum_{j=1}^{i-1} a_{ij}^2\,e_{GS}\,(y_j)\right) / a_{ii}^2 + \widehat{e_J\,(y_i)}$$

et $\widehat{e_{GS}\,(y_i)} > \widehat{e_J\,(y_i)}$ pour $i = 2, 3, \ldots, n$.

La méthode de Jacobi est donc moins sensible aux erreurs d'arrondis que celle de Gauss-Seidel.

C. Méthode de surrelaxation

(S.R.) s'écrit sur la $i^{ème}$ composante

$$y_i = \underbrace{\omega\left(b_i - \sum_{j=1}^{i-1} a_{ij}y_j - \sum_{j=i+1}^{n} a_{ij}x_j\right)/a_{ii}}_{A''} + (1-\omega)x_i \qquad \text{(S.R.)}$$

La quantité A est celle calculée pour obtenir y_i dans la méthode de Gauss-Seidel, d'autre part le facteur optimal de surrelaxation : ω étant compris entre 1 et 2. Il s'ensuit obligatoirement que

$$\widehat{e_{SR}}(y_i) > \omega^2\, e_{GS}(y_i) > e_{GS}(y_i)$$

donc pour $1 \leqslant \omega < 2$ la méthode de surrelaxation est plus sensible aux erreurs d'arrondis que la méthode de Gauss-Seidel, donc *a fortiori* que celle de Jacobi.

Une approximation de $\widehat{e_{SR}}(y_i)$ à partir de $\widehat{e_{GS}(y_i)}$ sera donnée par

$$\widehat{e_{SR}(y_i)} \simeq \omega^2\ \widehat{e_{GS}(y_i)} + \widehat{\varepsilon(\omega y_i)} + \widehat{\varepsilon((1-\omega)y_i)} + \widehat{\varepsilon(y_i)} \quad \text{(G.S.E.) (S.R.E.)}$$

En conséquence afin d'obtenir le résultat cherché avec la plus grande précision il convient après avoir approché suffisamment la solution par une des méthodes à convergence rapide (Gauss-Seidel) ou (Surrelaxation) de faire les dernières itérations avec la méthode de Jacobi.

Test d'arrêt et description de l'algorithme proposé

Les formules (J.E., G.S.E. et S.R.E.) nous permettent un calcul facile de la variance de $e(x_i^{k+1})$ donc de son écart-type $\sigma(e(x_i^{k+1}))$. Connaissant cet écart-type on obtient des bornes réalistes de l'erreur en prenant 4 fois cet écart-type (cf. [1]). Afin de continuer les calculs le plus loin possible sans pourtant faire cycler le calculateur, il est naturel d'arrêter les calculs lorsque par exemple :

$$|x_i^{k+1} - x_i^k| \leqslant 4\sigma[e(x_i^{k+1})]$$

Il n'est point nécessaire de calculer cet écart-type à toutes les itérations puisque cet écart-type va converger en même temps que x_i^k et que l'on ne cherche qu'une approximation grossière des bornes de l'erreur.

La procédure Algol que nous proposons est la suivante :

On résoud le système linéaire $AX = B$ par la méthode de Gauss-Seidel, à la $N^{ième}$ itération (dans le programme $N = 10$, N doit être modifié en tenant compte de la taille du système linéaire et de son conditionnement), on calcule

une estimation de $\sigma(e(x))$ pour Gauss-Seidel et Jacobi, puis on continue les itérations par Gauss-Seidel jusqu'à ce que

$$|x_i^{k+1} - x_i^k| \leqslant 4\sigma_{GS}[e(x_i^N)] = DGS[I] \quad \forall_i$$

On reprend les itérations par Jacobi jusqu'à ce que

$$|x_i^{k+1} - x_i^k| \leqslant 4\sigma_J[e(x_i^N)] = DX[I] \quad \forall_i$$

La procédure donne X, DX et DGS.

REMARQUE : cette procédure a été écrite pour l'IBM 7044 (en utilisant les formules ER). Afin de l'utiliser sur un autre calculateur il convient de la modifier comme suit :

Remplacer l'instruction : $T := 0.432 \times 4/2**27;$

par
$$T := 4\sqrt{\lambda^{-2p}\left(\frac{\lambda}{3} - \frac{\lambda^2}{4[\lambda-1]^2}[\mathrm{Log}(\lambda)]^2\right)};$$

où λ est la base du calculateur; p le nombre de chiffres de la mantisse ; dans le cas du 7044, $\lambda = 2$, $p = 27$ cela nous donne bien

$$T := 0.432 * 4/2 \uparrow 27 ;$$

Estimation de l'erreur et exemple numérique

Soit X^n le $n^{\text{ième}}$ itéré et X^{n+1} l'itéré suivant, en raison des erreurs d'arrondi on obtient une valeur approchée $\widehat{X_{n+1}}$.

On arrête les calculs lorsque

$$\widehat{|x_i^{n+1}} - x_i^n| < \mathrm{D}X[\mathrm{I}]$$

mais $|x_i^{n+1} - \widehat{x_i^{n+1}}| < \mathrm{D}X[\mathrm{I}]$ d'où $|x_i^{n+1} - x_i^n| < 2\,\mathrm{D}X[\mathrm{I}]$

désignons par J la matrice associée à l'itération de Jacobi et soit X la solution de l'équation $AX = B$ on sait alors que :

$$(X - X^{n+1}) \leqslant (I - J)^{-1}J(X^{n+1} - X^n)$$

Désignons par $k = S_{pp}((I - J)^{-1}J)$
alors

$$\|X - X_{n+1}\|_p \leqslant \|X - X^{n+1}\|_p + \|X^{n+1} - \widehat{X^{n+1}}\| \leqslant (2k+1)\,\|\mathrm{D}X\|_p$$

EXEMPLE

Nous avons traité le problème de Dirichlet ($\Delta u = 0$) sur le carré ABCD avec sur les faces AB et CD respectivement $u = +100$ et $u = -100$ et sur les faces BC et DA $u = 0$ en utilisant une grille à 400 points.

La procédure a été modifiée afin de tenir compte des 0 de la matrice, et du fait que les divisions par 4 en base 2 ne sont sources d'aucune erreur d'arrondi. Le calcul des écarts-types a été fait à la 550e itération. La dernière itération par Gauss-Seidel a été la 636e, 2 itérations ont été ensuite faites par Jacobi (1).

Aux nœuds 200 et 201 on a obtenu :

$u_{200} = 0{,}70249058$ $\quad$ DX [I] $= 0{,}160\ 10^{-7}$ $\quad$ DGS [I] $= 0{,}234\ 10^{-7}$

$u_{201} = -0{,}70248899$ $\quad$ DX [I] $= 0{,}957\ 10^{-8}$ $\quad$ DGS [I] $= 0{,}107\ 10^{-7}$

en raison des symétries nous avons ici $u_{200} = -u_{201}$,

d'autre part si $p = 2$ on a $k = 83$ et $||\mathrm{DX}||_2 = 0{,}153\ 10^{-4}$.

BIBLIOGRAPHIE

[1] J. P. CROUZEIX, *Étude statistique de l'erreur de chute dans la résolution de systèmes linéaires.* Thèse de 3e cycle (1968), Université de Clermont.

(1) Les essais numériques effectués ont toujours montré qu'il suffisait d'un très petit nombre d'itérations supplémentaires par Jacobi, ceci est dû à la relation qui lie les erreurs pour Gauss-Seidel et Jacobi.

PROCEDURE ALGOL

```
PRØCEDURE  ERGS (A, B, X, DX, N) ;
VALEUR  A, B, N ; ENTIER N  ; TABLEAU  A, B, X, DX ;
CØMMENTAIRE  résolution du système linéaire d'ordre n, AX = B par la
méthode de Gauss-Seidel, afin de minimiser les erreurs d'arrondi les dernières
itérations sont faites par la méthode de Jacobi. On arrête les itérations dès
que la différence de 2 vecteurs itérés successifs est inférieure à DX qui
désigne l'erreur maximum que l'on peut commettre sur le calcul d'un itéré ;
DEBUT  TABLEAU  Y, DGS [1 : N] ; BØØLEEN  TEST ;
    REEL  R, S, T, DR, DS ; ENTIER  CAS, I, J, CØMPT ;
    AIGUILLAGE  DIR : = ITERATIØN, ERREUR, CALTEST ;
    INITIALISATIØN : T : = 0.432 * 5/2↑27 ; CAS : = CØMPT : = 1 ;
        PØUR  I : = 1 PAS  1 JUSQU A  N FAIRE  X[I] : = B[I]/A[I, I] ;
    ITERATIØN : PØUR I : = 1 PAS 1 JUSQU A  N FAIRE  Y[I] : = X[I] ;
        PØUR I : = 1 PAS 1 JUSQU A N FAIRE
        DEBUT  R : = B[I] ;
            PØUR  J : = 1 PAS 1 JUSQU A I-1, I+1 PAS 1 JUSQU A N FAIRE
                R : = R - A[I, I] * X[J] ;
            X[I] : = R / A[I, I] ;
        FIN ; SI CØMPT = 10  ALØRS  CAS : = 2 ;
        CØMPT : = CØMPT + 1 ; ALLER A  DIR[CAS] ;
    ERREUR : CØMPT : = CØMPT + 1 ;
        PØUR  I : = 1 PAS  1  JUSQU A  N  FAIRE  Y[I] : = X[I] ;
        PØUR  I : = 1 PAS  1  JUSQU A  N  FAIRE
        DEBUT  DR : = 0 ; R : = B [I] ;
            PØUR  J : = 1  PAS  1  JUSQU A  I-1, I+1  PAS  1  JUSQU A N FAIRE
            DEBUT  S : = A[I, J] * X[J] ; R : = R - S ;
                DR : = DR + S↑2 + R↑2 ;
            FIN ; X[I] : = R : = R/A[I, I] ;
            DR : = DR/A[I, I]↑2 + R↑2 ;
            DX[I] : = SORT (DR) * T ; DS : = 0 ;
            PØUR J : = 1 PAS  1  JUSQU A  I-1  FAIRE
                DS : = DS + (A[I, J] * DGS[J])↑2 ;
            DS : = DS /A[I,I]↑2 + DX[I]↑2 ; DGS[I] : = SQRT(DS) ;
        FIN ; CAS : = 3 ; ALLER A  ITERATION ;
    CALTEST : TEST : = VRAI ;
        PØUR  I : = 1  PAS  1  JUSQU A  N  FAIRE
            TEST : = TEST ∧ABS (Y[I] - X[I]) < DGS[I] ;
        SI ⌉TEST  ALØRS  ALLERA  ITERATIØN ; CØMPT : = CØMPT - 1 ;
    JACØBI : TEST : = VRAI ; CØMPT : = CØMPT + 1 ;
            PØUR I : = 1 PAS  1  JUSQUA  N  FAIRE  Y[I] : = X [I] ;
            PØUR I : = 1 PAS  1  JUSQUA  N  FAIRE
            DEBUT  R : = B[I] ;
                PØUR J : = 1 PAS 1 JUSQUA I-1, I+1 PAS 1 JUSQUA N FAIRE
                    R : = R-A[I, J] * Y[J] ; X[I] : = R/A[I, I] ;
                TEST : = TEST ∧ ABS(X[I] - Y[I]) < DX[I] ;
            FIN ;
            SI ⌉TEST  ALØRS  ALLER A  JACØBI ;
        FIN  PRØCEDURE  ERGS ;
```

nº R-2, 1971.