Apport de l'ASD dans la programmation scientifique

Jean-Marie Chesneaux

1 La détection des instabilités

En dehors de son but initial qui est l'estimation de la qualité numérique des résultats, la détection d'instabilités numériques en cours d'exécutions est la fonctionnalité la plus importante de l'ASD.

Les deux instabilités numérique les plus caractéristiques sont

1. les division instables, i.e. lorsque le dénominateur est un zéro stochastique,
2. les pertes brutales de précision, i.e. lorsque le nombre de chiffres significatif exact chute brutalement en une seul opération (le seuil est fixé par l'utilisateur) qui ne peut être que la soustraction.

La division instable, voir l'exemple est la plus grave des instabilités et doit absolument être traitée.

Exemple 1 *Dans l'exemple suivant, on veut calculer la valeur du polynôme*

$$P(x,y) = 9.x^4 - y^4 + 2.y^2$$

avec $x = 10864$, $y = 18817$ *puis* $x = 1/3$, $y = 2/3$.
Les valeurs exactes sont $P(10864, 18817) = 1$, $P(1/3, 2/3) = 0.802469135802469135802...$

En arithmétique IEEE double précision arrondi au plus près, on trouve

$$P(10864, 18817) = 2.0000000000000000, \quad P(1/3, 2/3) = 0.8024691358024691$$

et en utilisant l'ASD, on trouve

$$P(10864, 18817) = @.0, \quad P(1/3, 2/3) = 0.802469135802469.$$

L'ASD a permis de mettre en évidence la qualité numérique de chacun des résultats. De plus, deux pertes brutales de précisions sont détectées. En effet, $9.x^4$ et y^4 sont très bien calculés (15 chiffres exacts) alors que $9.x^4 - y^4$ vaut $-0.70815898E + 009$ et n'a plus que 8 chiffres significatifs. On a donc perdu 7 chiffres en une opération et on perd tout le reste lors de l'addition suivante.

Cet exemple illustre également le fait que la qualité numérique dépend autant des données que de l'algorithme.

Exemple 2 *Appliquons la méthode de Gauss pour résoudre le système linéaire suivant :*

$$\begin{pmatrix} 21 & 130 & 0 & 2.1 \\ 13 & 80 & 4.74e8 & 752 \\ 0 & -0.4 & 3.9816e8 & 4.2 \\ 0 & 0 & 1.7 & 9e-9 \end{pmatrix} . X = \begin{pmatrix} 153.1 \\ 849.74 \\ 7.7816 \\ 2.6e-8 \end{pmatrix}$$

Le résultat exact est : $x_{sol}^t = (1, 1, 10^{-8}, 1)$.

En arithmétique virgule flottante simple précision arrondie au plus près, avec ou sans recherche partielle de pivot max par colonne, on obtient :

x_sol(1) = 0.6261987E+02
x_sol(2) = -0.8953979E+01
x_sol(3) = 0.0000000E+00
x_sol(4) = 0.9999999E+00

On s'aperçoit que, lors de la réduction de la troisième colonne, le pivot $A(3,3)$ vaut 4864 mais la valeur mathématique de $a(3,3)$ est zéro. Donc même avec une recherche partielle de pivot, compte tenu du fait que $a(4,3) = 1,7$, $A(3,3)$ est retenu comme choix du pivot et les erreurs d'arrondi explosent.

En ASD, sans recherche partiel de pivot, on obtient :

x_sol(1) = @.0
x_sol(2) = @.0
x_sol(3) = @.0
x_sol(4) = 0.9999999E+00

$@.0$ indique un zéro informatique. De plus, une perte brutale de précision et deux divisions instables sont détectées. Le mauvais choix du pivot et la dégradation des résultats est clairement mis en évidence par l'ASD.

En ASD, avec recherche partiel de pivot, les zéros informatiques sont automatiquement éliminés du choix des pivots. $a(4,3) = 1,7$ est retenu. On interdit ainsi les divisions instables et on obtient

x_sol(1) = 0.100E+01
x_sol(2) = 0.9999E+01
x_sol(3) = 0.1000000E+01
x_sol(4) = 0.9999999E+00

2 Contrôle des programmes

La détection des débranchements instables est un apport essentiel de l'ASD pour le contrôle du bon déroulement des programmes. Concrètement, lors de chaque test, on regarde si la différence entre les deux opérandes est significative. Dans l'affirmative, la réponse au test est validé et le programme se poursuit normalement. Dans le cas contraire, on est en présence d'un test instable, la branche de l'égalité est exécutée et un message avertit l'utilisateur.

Exemple 3 *Reprenons le calcul des racines de l'équation*

$$0.3 * x^2 \ + \ 2.1 * x \ + 3.675 \ = \ 0$$

En programmant :

```
if (delta > 0 ) then
    ....traitement de deux racines reelles dinstinctes
elseif (delta < 0)
   ..... traitement de deux racines complexes
else
  .... traitement d'une racine double
endif
```

en ASD, on obtient

```
d =   @.0
Le discriminant est nul.
La racine double reelle est -.3500000E+01
```

La valeur non significative du discriminant est détectée. Cela signifie que, en raison de la précision limitée de son arithmétique, l'ordinateur ne peut différencier les deux racines. La réponse cohérente pour lui est d'annoncer deux racines confondues.

Le contrôle des débranchements est particulièrement important pour les tests d'arrêt des méthodes itératives.

Ces méthodes, pour la plupart, sont mises en oeuvre dans le but d'approcher la limite éventuelle x_s de $(x_n)_{n\in\mathrm{N}}$ (qui est l'objet mathématique recherché) par un des éléments de la suite. Le problème est d'arrêter correctement le processus.

Il est très difficile avec l'arithmétique virgule flottante de répondre à cette question. On ne peut qu'utiliser des tests d'arrêt basés sur des quantités ε choisies a priori avec tous les inconvénients que cela comporte. Par contre, en utilisant le contrôle de la précision et les concepts de zéro et d'égalité stochastique, on résout le problème en évitant l'utilisation de valeurs ε arbitraires.

Deux cas sont à envisager :

1. Les problèmes à solution contrôlable.

   Cela signifie qu'il existe une fonction Ψ annulée par la solution. Par exemple :

   - La résolution de $\mathcal{F}(\mathcal{X}) = 0$, ici $\Psi(x_s) = \mathcal{F}(x_s) = 0$.
   - La recherche de $\underset{Max}{Min}\mathcal{G}(\mathcal{X})$, ici $\Psi(x_s) = grad(\mathcal{G}(x_s)) = 0$.

   Il faut alors utiliser le *test d'arrêt optimal*

   $$\text{Si} \quad \Psi(x_k) = @.0 \quad \text{alors} \quad arrêt.$$

   Le processus s'arrête lorsque la valeur calculée de $\Psi(x_k)$ ne représente plus que des erreurs d'arrondi. On demande ainsi à l'ordinateur de stopper dès qu'une solution informatiquement satisfaisante est trouvée. En effet, le plus que l'on puisse demander à l'ordinateur est d'annuler la fonctionnelle aux erreurs d'arrondi près. Remarquons qu'en ASD ce test s'écrit tout simplement :

   ```
   if (PSI(X).eq.0) then stop
   ```

   On retrouve la formulation mathématique classique.

2. Les problèmes à solution non contrôlable

   C'est le cas, entre autres, du calcul des séries numériques. Il faut alors utiliser le test d'arrêt sur le piétinement. On remplace les anciens tests

   $$\|X_{k+1} - X_k\| \leq \varepsilon \;\text{ ou }\; \|X_{k+1} - X_k\| \leq \varepsilon.\|X_k\|$$

   par

   $$\|X_{k+1} - X_k\| = @.0.$$

   On décide d'arrêter le processus lorsque la différence entre X_{k+1} et X_k (caractérisée par la norme) n'est plus significative ce qui signifie que la transformation de X_k en X_{k+1} n'apporte plus d'information réelle. Dans le cas d'une suite de nombres réels, du fait des définitions de l'arithmétique stochastique, le test s'écrit :

```
if (X(K+1).eq.X(K)) then stop
```

Remarque 1 *Dans les deux cas, il ne faut pas oublier que l'on a obtenu un résultat quasi-optimal uniquement au sens de la machine. L'ordinateur a poursuivi le calcul des itérés tant que cela apportait de l'information réelle (ce qui est déjà important) mais l'analyse sur l'écart éventuel entre le dernier itéré et la limite reste le travail de l'utilisateur. De même l'utilisation des tests stochastiques ne dispense nullement de prévoir un nombre maximal d'itérations dans le cas où le processus ne convergerait pas.*

Exemple 4 *On veut résoudre*

$$1.47x^3 \ + \ 1.19x^2 \ - \ 1.83x \ + \ 0.45 \ = \ 0$$

par la méthode de Newton avec $x_0 = 0.5$.

En arithmétique virgule flottante avec le test d'arrêt $||x_n - x_{n+1}|| \leq 10^{-14}$, selon le mode d'arrondi, on obtient :

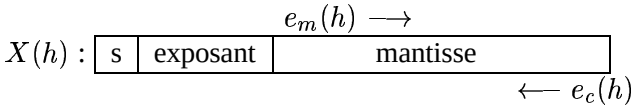
En arrondi au plus près :	$x(24) = 0.4285714332352813$
En arrondi vers zéro :	$x(24) = 0.4285714343480436$
En arrondi vers $-\infty$:	$x(24) = 0.4285714343480436$
En arrondi vers $+\infty$:	$x(30) = 0.4285714326546251$

Dans les 4 cas, la suite est stationnaire mais de façon illusoire. En ASD, avec le test

```
if (X(K+1).eq.X(K)) then stop
```

on trouve $x(25) = 0.428571433$. La vraie racine est $x = 0.4285714285714285714...$ L'utilisation de l'ASD a permis d'arrêter le processus au moment où poursuivre ne présentait plus d'intérêt numérique et de fournir le dernier itéré avec sa juste précision (seulement 9 chiffres).

Les méthodes approchées classiques font intervenir un pas de discrétisation h. Du point de vue théorique, l'erreur de méthode e_m tend vers zéro quand h tend vers zéro alors que, souvent, l'erreur de calcul e_c croît lorsque h décroît. Soit $X(h)$ l'approximation, quand $h \longrightarrow 0$:



Tant que $e_c(h) < e_m(h)$, diminuer h apporte de "l'information" fiable. Par contre, dès que $e_c(h) \geq e_m(h)$, diminuer h ne fera que dégrader la qualité numérique de l'approximation. Il faut s'arrêter lorsque $e_c(h) = e_m(h)$ ce qui définit le pas optimal h^*.

L'ASD permet de résoudre en partie ce problème. En effet, souvent, dans les méthodes approchées, il existe une formule permettant d'estimer l'erreur de méthode, le plus souvent en comparant $X(h)$ et $X(h/2)$. L'utilisation de l'ASD permettant d'estimer e_c, il devient possible de calculer h^*.

Exemple 5 *On désire calculer numériquement*

$$I = \int_{-1}^{1} 20.\cos(20x)\ (2.7x^2 - 3.3x + 1.2)\ dx$$

par la méthode de Simpson.

Soit I_n l'approximation de I obtenue avec $n+1$ points, on montre la relation suivante :

$$I_n - I_{n+1} = \frac{15}{16}.(I_n - I) + \mathcal{O}\left(\frac{1}{256^n}\right)$$

Ainsi, à $\mathcal{O}\left(\frac{1}{256^n}\right)$ près, les chiffres significatifs communs à I_n et I_{n+1} sont quasiment les mêmes que ceux communs à I_n et I. En arrêtant le processus dès que $I_n - I_{n+1} = @.0$, non seulement on optimise le nombre de points (tant que $I_n - I_{n+1}$ est significatif, l'erreur de calcul est inférieure à l'erreur de méthode) mais on est certain, si l'on est dans la phase de convergence, que les chiffres exacts de I_n sont aussi ceux de la valeur exacte I de l'intégrale. Le tableau ci-dessous montre les valeurs successives de I_n, le test d'arrêt étant

```
if (I(n+1).eq.I(n)) then stop
```

La valeur exacte de l'intégrale est $I = 7.316687747285081....$

n	I_n	$\lvert I_n - I \rvert$
1	0.532202672142963E+002	0.459035794670112E+002
2	-0.233434428466744E+002	0.306601305939594E+002
3	-0.235451792663099E+002	0.308618670135950E+002
4	0.106117380632568E+002	0.32950503159717E+001
5	0.742028156692706E+001	0.1035938196419E+000
6	0.732233719854277E+001	0.564945125769E-002
7	0.731702967403266E+001	0.3419267475E-003
8	0.73167089491443E+001	0.212018592E-004
9	0.73166890697896E+001	0.1322504E-005
10	0.73166878299008E+001	0.826158E-007
11	0.7316687752447E+001	0.5162E-008
12	0.7316687747607E+001	0.322E-009
13	0.7316687747305E+001	0.2E-010
14	0.731668774728E+001	@.0

Le calcul s'arrête au moment où $I_n - I_{n+1} = @.0$ et les chiffres exacts de I_n sont bien en commun avec ceux de I.