REVUE FRANÇAISE D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE

J. CHION

Une méthode d'utilisation de l'algorithme de Routh dans la résolution des équations algébriques

Revue française d'informatique et de recherche opérationnelle, tome 2, nº R3 (1968), p. 3-12

<http://www.numdam.org/item?id=M2AN_1968__2_3_3_0>

R.I.R.O.
(2e année, N° 13, 1968, p. 3-12)

UNE METHODE D'UTILISATION DE L'ALGORITHME DE ROUTH DANS LA RESOLUTION DES EQUATIONS ALGEBRIQUES

par J. CHION [1]

Résumé. — *L'algorithme de Routh, construit à partir des coefficients d'un polynôme, permet de déterminer le nombre de racines dont la partie réelle est positive. Cet algorithme a été employé par E. Aparo (1958) en une méthode qui consiste à translater l'axe imaginaire selon un procédé dichotomique pour isoler les racines correspondant à une même partie réelle.*

On peut faire les remarques suivantes :

1e *Les coefficients du schéma de Routh sont des fonctions rationnelles de* α, *paramètre de translation de l'axe imaginaire.*

2° *Lorsque* α *égale une partie réelle d'une racine, on voit apparaître dans le schéma une ligne de zéros, caractéristique du fait que le polynôme translaté possède des racines imaginaires pures.*

Une partie réelle d'une racine peut donc se chercher, à l'aide de la méthode de Newton, comme une racine des coefficients de la ligne test du schéma, le calcul des dérivées de ces coefficients par rapport à α *faisant l'objet d'un algorithme simple.*

La comparaison avec les temps de calcul du procédé dichotomique s'est toujours montrée avantageuse.

I. — RAPPEL DE L'ALGORITHME DE ROUTH [2]

Soit le polynôme à coefficients réels :

$$f(x) = a_0x^n + b_0x^{n-1} + a_1x^{n-2} + b_1x^{n-3} + a_2x^{n-4} + b_2x^{n-5} + \ldots$$

On rappelle que l'algorithme de Routh permet d'obtenir k, nombre de racines à parties réelles positives de $f(x)$, à partir du schéma suivant :

$$(1) \qquad \begin{array}{lll} a_0 & a_1 & a_2 \ldots \\ b_0 & b_1 & b_2 \ldots \\ c_0 & c_1 & c_2 \ldots \\ d_0 & d_1 & d_2 \ldots \\ \ldots & \ldots & \ldots \end{array}$$

(1) Institut d'Enseignement Supérieur du Bénin Porto-Novo (Dahomey).
(2) Voir F. R. GANTMACHER, *The Theory of Matrices* (traduit du russe), volume 2, Chelsea Publishing Company, New York.

dans lequel chaque ligne, à partir de la 3^e^ est déduite des 2 précédentes selon l'algorithme

$$c_i = a_{i+1} - \frac{a_0}{b_0} b_{i+1}.$$

k est alors donné, dans le cas général, par le nombre de variations de signes dans la 1^re^ colonne du schéma (1)

$$k = V(a_0, b_0, c_0, d_0, \ldots). \tag{2}$$

Si on considère les polynômes

$$\begin{aligned} f_1(x) &= a_0 x^n - a_1 x^{n-2} + a_2 x^{n-4} - a_3 x^{n-6} + \ldots \\ f_2(x) &= b_0 x^{n-1} - b_1 x^{n-3} + b_2 x^{n-5} - b_3 x^{n-7} + \ldots \\ f_3(x) &= c_0 x^{n-2} - c_1 x^{n-4} + c_2 x^{n-6} - c_3 x^{n-8} + \ldots \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{aligned}$$

et si on pose $f(i\omega) = U(\omega) + iV(\omega)$ (ω réel), on vérifie alors que le dernier polynôme donné par le schéma (1) est le PGCD de tous les polynômes de la suite (la ligne suivante étant une ligne de zéros), et que f_1 et f_2 sont respectivement U et V, ou V et U (selon la parité de n) à des coefficients multiplicatifs près. De plus, l'ensemble des racines du PGCD s'identifie à l'ensemble des racines de f 2 à 2 opposées, chacune étant multipliée par i (par exemple les racines réelles de PGCD sont les racines imaginaires pures de f).

II. — METHODES NUMERIQUES DE RESOLUTION DES EQUATIONS ALGEBRIQUES BASEES SUR L'ALGORITHME DE ROUTH

1) Procédé dichotomique

La formule (2) ne donne le nombre k que lorsque le schéma (1) comporte $n + 1$ lignes, ce qui nécessite en particulier que le PGCD soit de degré 0. La méthode numérique déjà existante [1] consiste à translater l'axe imaginaire selon un procédé dichotomique, pour isoler une partie réelle dans une bande de largeur voulue. Cela exige, à chaque itération, le calcul des coefficients du nouveau polynôme translaté et le calcul du nombre k ; d'autre part le procédé dichotomique impose un nombre d'itérations fixe pour atteindre la précision voulue.

2) Autre procédé : le procédé Routh-Newton

Nous appellerons ainsi la méthode d'obtention des parties réelles que nous allons maintenant exposer.

---

(1) Voir E. Aparo, *Un criterio di Routh e sua applicazione al calcolo degli zeri di un polinomio.* Consiglio Nazionale delle Richerche. Publicazioni dell'instituto per le applicazioni del calcolo N 536, Roma, 1958.

Lorsqu'une partie réelle x est obtenue, la translation correspondante conduit à un cas singulier pour le schéma (1) Si p racines de f correspondent à la partie réelle x, le schéma (1) appliqué au polynôme translaté de x (qui possède p racines imaginaires pures) produit un PGCD de degré p et la ligne suivante ne comporte que des zéros, l'apparition de cette ligne de zéros étant justement le test d'obtention du PGCD.

$$
\begin{array}{lllllll}
 & f_1 & a_0\ a_1\ a_2 \dots & & & & \\
(3) & f_2 & b_0\ b_1\ b_2 \dots & & & & \\
 & f_3 & c_0\ c_1\ c_2 \dots & & & & \\
 & \dots & \dots\dots\dots & & & & \\
 & f_{n+1-p} & h_0 & h_1 & h_2 & \dots & \text{PGCD, de degré } p \\
 & f_{n+2-p} & k_0 = 0 & k_1 = 0 & k_2 = 0 & \dots & \text{ligne de zéro}
\end{array}
$$

L'algorithme de E. Aparo recherche ce point x par dichotomie en calculant pour des points voisins de x (donc pour des cas non singuliers) le « nombre de racines à droite ».

L'algorithme de Routh-Newton se propose de rechercher directement le point x qui annule la $(n + 2 - p)^{\text{ième}}$ ligne. Appelons α le paramètre de translation de l'axe imaginaire. Soit

$$a_0(\alpha),\ b_0(\alpha),\ a_1(\alpha),\ b_1(\alpha) \dots$$

les coefficients du polynôme translaté. On a les relations suivantes :

$$
(4) \qquad \begin{cases}
a_0(\alpha) = \dfrac{f^{(n)}(\alpha)}{n!} & a_1(\alpha) = \dfrac{f^{(n-2)}(\alpha)}{(n-2)!} & a_2(\alpha) = \dfrac{f^{(n-4)}(\alpha)}{(n-4)!} \dots \\[2ex]
b_0(x) = \dfrac{f^{(n-1)}(\alpha)}{(n-1)!} & b_1(\alpha) = \dfrac{f^{(n-3)}(\alpha)}{(n-3)!} & b_2(\alpha) = \dfrac{f^{(n-5)}(\alpha)}{(n-5)!} \dots
\end{cases}
$$

avec

$$
\begin{cases}
a_0(0) = a_0 & a_1(0) = a_1 & a_2(0) = a_2 \dots \\
b_0(0) = b_0 & b_1(0) = b_1 & b_2(0) = b_2 \dots
\end{cases}
$$

Les fonctions (4) sont des polynômes en α. Par conséquent, si à partir de là, on construit le schéma de Routh, les lignes suivantes sont des fonctions rationnelles en α et on peut essayer de déterminer le point α tel que [disposition (3)]

$$k_0(\alpha) = 0 \qquad k_1(\alpha) = 0 \qquad k_2(\alpha) = 0, \dots$$

La méthode proposée est la méthode de Newton : partant d'un point x_i, le point suivant sera déterminé, non plus par dichotomie mais par l'itération de Newton

$$x_{i+1} = x_i - \frac{k_0(x_i)}{k_0'(x_i)}.$$

Elle nécessite la connaissance de $k_0'(\alpha)$, $k_1'(\alpha)$, $k_2'(\alpha)$ ..., que nous obtiendrons de la façon suivante :

Soit les 3 lignes consécutives quelconques du schéma

$$\begin{array}{l} m_0(\alpha) \ldots m_i(\alpha) \quad m_{i+1}(\alpha) \ldots \\ n_0(\alpha) \ldots n_i(\alpha) \quad n_{i+1}(\alpha) \ldots \\ p_0(\alpha) \ldots p_i(\alpha) \ldots\ldots\ldots\ldots \end{array}$$

$$p_i(\alpha) = m_{i+1}(\alpha) - \frac{m_0(\alpha)}{n_0(\alpha)} n_{i+1}(\alpha),$$

d'où la possibilité d'obtenir p_i' en dérivant la formule précédente. La connaissance des 2 premières lignes et des 2 premières lignes dérivées permet d'obtenir la 3^e^ ligne dérivée. Or on connaît les dérivées des 2 premières lignes du schéma :

$$(5) \quad \left\{ \begin{array}{lll} a_0'(\alpha) = \dfrac{f^{(n+1)}(\alpha)}{n!} = 0 & a_1'(\alpha) = \dfrac{f^{(n-1)}(\alpha)}{(n-2)!} & a_2'(\alpha) = \dfrac{f^{(n-3)}(\alpha)}{(n-4)!} \ldots \\ b_0'(\alpha) = \dfrac{f^{(n)}(\alpha)}{(n-1)!} & b_1'(\alpha) = \dfrac{f^{(n-2)}(\alpha)}{(n-3)!} & b_2'(\alpha) = \dfrac{f^{(n-4)}(\alpha)}{(n-5)!} \ldots \end{array} \right.$$

on pourra donc, connaissant le schéma, construire le schéma dérivé ligne après ligne et en particulier obtenir les dérivées des fonctions $k_0(\alpha)$, $k_1(\alpha)$ ... qui nous importent.

Pratiquement, ce sont les fonctions et les dérivées au point 0 qui nous sont accessibles; pour $\alpha = 0$ (5) devient :

$$\left\{ \begin{array}{llll} 0 & (n-1)b_0 & (n-3)b_1 & (n-5)b_2 \ldots \\ na_0 & (n-2)a_1 & (n-4)a_2 & (n-6)a_3 \ldots \end{array} \right.$$

Pour obtenir les dérivées en 1 point α quelconque, nous procéderons de la façon suivante :

$$\begin{array}{c} \text{Polynôme} \\ \text{initial} \end{array} \xrightarrow[\text{de l'axe imaginaire}]{\text{Translation } \alpha} \begin{array}{c} \text{Nouveau} \\ \text{polynôme} \end{array} \xrightarrow{\quad} \begin{array}{c} \text{Schéma de Routh} \\ \text{au point 0 pour ce polynôme} \end{array} \xrightarrow{\quad} \begin{array}{c} \text{Schéma dérivé} \\ \text{au point 0 pour ce polynôme} \end{array}$$

l'algorithme donnant une ligne dérivée à partir des 2 lignes et des 2 lignes dérivées précédentes étant simplifié de la façon suivante :

$$(6) \; p_i' = m_{i+1}' - \left[\left(m_0' - \boxed{\frac{m_0}{n_0}}\, n_0' \right) \Big/ n_0 \right] \times n_{i+1} - \boxed{\frac{m_0}{n_0}} \times n_{i+1}'$$

qui permet de ne calculer le facteur entre crochets qu'une seule fois par ligne, et en combinant les procédures de calcul du schéma et du schéma dérivé n'impose qu'un seul calcul des termes entourés d'un rectangle pour les deux procédures.

Nous présentons maintenant une procédure ALGOL du procédé Routh-Newton. Nous l'avons appliqué à la recherche des racines complexes conjuguées simples (la ligne de zéros recherchée possède alors un seul coefficient). Il est mené de la façon suivante :

1^{re} phase : le procédé dichotomique produit un intervalle [BI, BS] encadrant la partie réelle et donnant le nombre de racines dans cette bande.

2^e phase :

1^{er} cas : racine réelle simple dans la bande [BI, BS] : celle-ci est affinée par la méthode de Newton utilisant BI comme valeur de départ.

2^e cas : 2 racines complexes conjuguées ou 2 racines réelles confondues : la partie réelle est affinée à l'aide de l'algorithme Routh-Newton qui effectue dans le schéma de Routh, la recherche du PGCD de degré 2. Les parties imaginaires sont les 2 racines réelles symétriques du PGCD.

3^e cas : le procédé dichotomique trouve plus de 2 racines dans la bande [BI, BS]. La procédure donne alors seulement BI, BS et le nombre de racines.

Enfin, si pour une raison quelconque le procédé Routh-Newton conduit hors de [BI, BS], ce cas est assimilé au précédent.

Paramètres d'entrée de la procédure Routh-Newton :

EPS : précision à atteindre avec le procédé dichotomique avant « d'enclencher » le procédé Routh-Newton.

N : degré de l'équation.

DEFLAT : paramètre booléen **vrai** si on demande d'éliminer les racines au fur et à mesure, **faux** si on ne veut pas cette élimination.

EPSR et EPSC précisions voulues sur les racines réelles pures et les parties réelles des racines complexes.

KSUPR maximum d'itérations dans l'amélioration des racines réelles depuis EPS jusqu'à EPSR.

KSUPC nombre d'itérations maximum du procédé Routh-Newton.

C un tableau réel contenant les coefficients de l'équation rangés par ordre de puissances décroissantes depuis C[1] jusqu'à C[N + 1].

Paramètres de sortie :

K1 entier qui est le double du nombre des parties réelles calculées ou simplement localisées.

Tableau RAC contient successivement :

— dans le cas d'un calcul effectif la partie réelle puis le module des parties imaginaires (zéro pour une racine réelle),

— dans le cas d'une localisation la borne BI puis la borne BS.
Tableau T contient successivement :

— pour un calcul effectif le nombre de parties imaginaires (2 ou 1), puis le nombre d'itérations ($\geqslant 1$) de la méthode de Newton pour 1 racine réelle, du procédé Routh-Newton pour 2 racines complexes,

— pour 1 localisation le nombre de parties imaginaires puis 0. Ce dernier zéro indiquant formellement que la procédure n'a fourni qu'une localisation de la partie réelle.

Enfin nous avons un certain nombre de sous-procédures dont nous ne donnerons que l'en-tête (1) :

réel procédure MAXMOD pour obtenir le rayon d'un cercle contenant toutes les racines.
procédure TRANSLREEL pour obtenir les coefficients du polynôme translaté d'une valeur réelle α.
réel procédure NEWTON de la méthode de Newton.
procédure DEFLAT1 et DEFLAT2 pour éliminer au besoin les racines réelles simples et les racines complexes conjuguées.
procédure HORNER pour calculer les valeurs de $f(\alpha)$ et de $f'(\alpha)$.
procédure SCHEMA donne le schéma de Routh et le nombre k.
procédure DEUX SCHEMAS donne le schéma et le schéma dérivé selon l'algorithme (6).

Voici cette **procédure** ROUTH NEWTON.

```
procedure ROUTH NEWTON (N, C, EPS, EPSR, EPSC, KSUPR, KSUPC,
RAC, K1, T, DEFLAT);
valeur EPS, EPSR, EPSC, C; entier N, KSUPR, KSUPC, K1;
reel EPS, EPSR, EPSC; tableau C, RAC; entier tableau T; booleen DEFLAT;
debut reel procedure MAXMOD (A, N); tableau A; entier N;
      procedure TRANSLREEL (C, CT, N, ALFA); tableau C, CT;
      entier N; reel ALFA;
      reel procedure NEWTON (XO, FO, DFO); valeur FO, DFO;
      reel XO, FO, DFO;
      procedure DEFLAT1 (C, N, X);
      tableau C; entier N; reel X;
      procedure DEFLAT2 (C, N, A, B); valeur B;
      tableau C; entier N; reel A, B;
      procedure HORNER (X, C, F, DF, N);
      reel X, F, DF; tableau C; entier N;
      procedure SCHEMA (T1, C, N, P, V);
```

(1) Voir J. Chion, thèse de 3e Cycle présentée à l'Université de Grenoble.

```
tableau T1, C; entier N, P, V;
procedure DEUXSCHEMAS (C, T1, T2, N, P);
tableau C, T1, T2; entier N, P;
debut
fin DEUXSCHEMAS;
entier P, I, H, M1, M2, K, M, J;
reel AMPL, SUP, X, BI, BS, F, DF, Y;
P : = N ÷ 2 + 1; K1 : = 0;
debut tableau CT[1 : N + 1], T1, T2[1 : N + 1, 1 : P];
    AMPL : = SUP : = MAXMOD (C, N);
    X : = — SUP; J : = I : = H : = N;
    ITER : si H ⩾ I alors
    debut BI : = X; M1 : = H; X : = X + AMPL fin sinon
    debut BS : = X; M2 : = H; X : = X — AMPL fin;
    si AMPL < EPS alors allera SUITE;
    TRANSLREEL (C, CT, J, X);
    SCHEMA (T1, CT, J, P, H);
    AMPL : = AMPL/2.0; allera ITER;
    SUITE : M : = M1-M2; si M = 1 alors
    debut K : = 1;
        ITERREEL : HORNER (X, C, F, DF, J);
        Y : = NEWTON (X, F, DF);
        si ABS (Y — X) > EPSR ∧ K < KSUPR alors
        debut X : = Y; K : = K + 1; allera ITERREEL fin;
        K1 : = K1 + 1; RAC[K1] : = Y; T[K1] : = 1;
        K1 : = K1 + 1;
        RAC[K1] : = 0.0; T[K1] : = K; H : = I : = I — 1;
        si I ⩽ 0 alors allera FINPRO;
        si DEFLAT alors debut J : = J — 1; DEFLAT1 (C, J, Y);
                                    P : = J ÷ 2 + 1 fin
    fin sinon si M ≠ 2 alors
    debut H : = I : = I — M; K1 : = K1 + 1; RAC[K1] : = BI;
        T[K1] : = M; K1 : = K1 + 1; RAC[K1] : = BS;
        T[K1] : = 0; si I ⩽ 0 alors allera FINPRO
    fin sinon
    debut TRANSLREEL (C, CT, J, X);
        DEUXSCHEMAS (CT, T1, T2, J, P); K : = 1;
        pour Y : = NEWTON (X, T1[J, 1], T2[J, 1])
        tant que ABS (Y — X) > EPS ∧ K < KSUPC faire
        si Y ⩽ BI ∨ Y ⩾ BS alors
        debut K1 : = K1 + 1; RAC[K1] : = BI; T[K1] : = M;
            K1 : = K1 + 1; RAC[K1] : = BS; T[K1] : = 0;
            H : = I : = I — 2; si I ⩽ 0 alors allera FINPRO;
            allera ETIQ
```

```
                fin sinon
                debut X : = Y; K : = K + 1;
                      TRANSLREEL (C, CT, J, X);
                      DEUXSCHEMAS (CT, T1, T2, J, P)
                fin;
                K1 : = K1 + 1; RAC[K1] : = Y; T[K1] : = 2;
                K1 : = K1 + 1;
                RAC[K1] : = RAC2(ABS(T1[J — 1,2]/T1[J — 1,1]));
                      T[K1] : = K; H : = I : = I-2;
                si I ⩽ 0 alors allera FINPRO;
                si DEFLAT alors debut J : = J — 2;
                                  DEFLAT2 (C, J, Y, RAC[K1]);
                                  P : = J ÷ 2 + 1
                                  fin
          fin;
          ETIQ : X : = BS; AMPL : = (SUP — X)/2.0; allera ITER;
          FINPRO :
     fin
fin ROUTH NEWTON;
```

Résultats numériques.

Nous donnerons maintenant à titre de vérification, deux exemples de résolution d'équations algébriques. Les essais ont été effectués sur machine IBM 7044 au Laboratoire de Calcul de l'Université de Grenoble, à partir de programmes ALGOL. Les coefficients des polynômes sont donnés par ordre de puissances décroissantes :

Essai 1. polynôme de degré 10.

Coefficients : 1, — 10, 26, — 296, 3 430, 6 372, — 85 892, — 181 816, — 230 215, — 9 246 650, 101 130 250.

Racines exactes : $-5 \pm 2i$, $-3 \pm 7i$, $\pm 5i$, $6 \pm i$, $7 \pm 4i$.

Résultats procédé dichotomique sans déflation; précision demandée EPSC $= 10^{-7}$:

Parties réelles	Modules des parties imaginaires
— 4,9999999	2, 0000000
— 2,9999999	7, 0000086
$0{,}13821591\ 10^{-6}$	5,0000013
6,0000000	1,0000000
6,9999998	4,0000056

Temps de calcul : 8,3 secondes

Résultats procédé Routh-Newton sans déflation; EPS $= 10^{-1}$ EPSC $= 10^{-7}$

Parties réelles	Modules des parties imaginaires	Nombre d'itérations du procédé Routh-Newton
— 5, 0000000	2,0000000	1
— 2, 9999999	6,9999959	3
— 0, 58605494 10^{-7}	5,0000013	5
6, 0000000	1,0000000	4
6, 9999999	4,0000027	6

Temps de calcul : 3,8 secondes

Essai 2. Polynôme de degré 8.

Coefficients : 1, — 8, 56, — 336, 1 680, — 6 720, 20 160.

Racines avec 8 chiffres exacts : — 2,2209395 ± 5,0143688i; 1,5863875 ± 4,8397364i; 4,6345518 ± 2,0883787i.

Résultats procédé dichotomique sans déflation. EPSC $= 5\ 10^{-8}$.

Parties réelles	Modules de parties imaginaires
— 2,2209394	5,0143688
1,5863875	4,8397365
4,6345519	2,0883787

Temps de calcul : 2,5 secondes

Résultats procédé Routh-Newton sans déflation :

EPS $= 10^{-1}$ EPSC $= 5\ 10^{-8}$

Parties réelles	Module des parties imaginaires	Nombre d'itérations
— 2,2209394	5,0143687	5
1,5863875	4,8397363	4
4,6345519	2,0883787	4

Temps de calcul : 1,2 secondes

CONCLUSION

D'autres essais ont été effectués, qui donnent à peu près les mêmes résultats : les temps de calcul sont environ dans le rapport 1/2 en faveur du procédé Routh-Newton. Il semble que la convergence quadratique de ce dernier compense largement le fait que chaque itération coûte plus cher. Un décompte du nombre d'opérations montre d'ailleurs que, quel que soit le degré de l'équation, on peut au moins se permettre la moitié des itérations nécessaires au procédé dichotomique. Ceci est à comparer avec le parallèle méthode de bissection — méthode de Newton dans la recherche des racines réelles. Cette dernière coûte, par itération, deux fois plus cher que la précédente et n'est rentable que si elle procède avec deux fois moins d'itérations.

La procédure donnée n'est pas absolument complète puisqu'elle se borne en fait à la recherche des racines simples, mais quelques modifications peuvent suffire à lui donner une portée générale. Ceci pouvant faire l'objet d'un travail de programmation pure, nous avons préféré, à titre de vérification, nous limiter aux cas les plus courants pour comparer procédé dichotomique et procédé Routh-Newton.