Deux méthodes de résolution d'équations algébriques, le procédé des réduites, l'algorithme de Routh

Jean Chion

n° d'ordre :

UNIVERSITE DE GRENOBLE

FACULTE DES SCIENCES

DEUX METHODES DE RESOLUTION D'EQUATIONS ALGEBRIQUES : LE PROCEDE DES REDUITES, L'ALGORITHME DE ROUTH

-:-:-:-:-:-

THESE
POUR OBTENIR

LE TITRE DE DOCTEUR DE SPECIALITE
(MATHEMATIQUES APPLIQUEES)

-:-:-:-:-:-

Présentée par

Jean CHION

Licencié ès Sciences

-:-:-:-:-:-

Thèse soutenue le 26 novembre 1965 Devant la commission d'Examen

MM. KUNTZMANN Président
GASTINEL
LAURENT } Examinateurs

FACULTE DES SCIENCES

LISTE DES PROFESSEURS

DOYENS HONORAIRES M. FORTRAT P.

M. MORET L.

DOYEN M. WEIL L.

PROFESSEURS TITULAIRES

MM.	NEEL L.	MAGNETISME
	HEILMANN R.	CHIMIE ORGANIQUE
	KRAVTCHENKO J.	MECANIQUE RATIONNELLE
	CHABAUTY C.	MATHEMATIQUES PURES
	PARDE M.	POTAMOLOGIE
	BENOIT J.	RADIOELECTRICITE
	CHENE M.	CHIMIE PAPETIERE
	BESSON J.	ELECTROCHIMIE
	WEIL L.	THERMODYNAMIQUE
	FELICI N.	ELECTROSTATIQUE
	KUNTZMANN J.	MATHEMATIQUES APPLIQUEES
	BARBIER R.	GEOLOGIE APPLIQUEE
	SANTON L.	MECANIQUE DES FLUIDES
	OZENDA P.	BOTANIQUE
	FALLOT M.	PHYSIQUE INDUSTRIELLE
	GALVANI O.	MATHEMATIQUES
	MOUSSA A.	CHIMIE NUCLEAIRE ET RADIOACTIVITE
	TRAYNARD P.	CHIMIE GENERALE
	SOUTIF M.	PHYSIQUE GENERALE
	CRAYA A.	HYDRODYNAMIQUE
	REULOS R.	THEORIE DES CHAMPS
	AYANT Y.	PHYSIQUE APPROFONDIE
	GALISSOT F.	MATHEMATIQUES PURES
Mlle	LUTZ E.	MATHEMATIQUES GENERALES
MM.	BLAMBERT M.	MATHEMATIQUES
	BOUCHEZ R.	PHYSIQUE NUCLEAIRE
	LLIBOUTRY L.	GEOPHYSIQUE
	MICHEL R.	GEOLOGIE ET MINERALOGIE
	BONNIER E.	METALLURGIE
	DESSAUX G.	PHYSIOLOGIE ANIMALE
	PILLET E.	ELECTROTECHNIQUE
	DEBELMAS J.	GEOLOGIE GENERALE
	GERBER R.	MATHEMATIQUES PURES
	PAUTHENET R.	ELECTROTECHNIQUE
	VAUQUOIS B.	CALCUL ELECTRONIQUE
	SILBER R.	MECANIQUE DES FLUIDES
	MOUSSIEGT J.	ELECTRONIQUE
	BARBIER J.C.	PHYSIQUE EXPERIMENTALE

	BUYLE-BODIN M.	ELECTRONIQUE
	KOSZUL J.L.	MATHEMATIQUES
	DREYFUS B.	THERMODYNAMIQUE
	VAILLANT F.	ZOOLOGIE
	KLEIN J.	MATHEMATIQUES PURES
	SENGEL P.	ZOOLOGIE
	ARNAUD P.	CHIMIE
	BARJON R.	PHYSIQUE NUCLEAIRE
	BARNOUD F.	BIOSYNTHESE DE LA CELLULOSE

PROFESSEURS ASSOCIES

MM.	WAGNER H.	BOTANIQUE
	NAPP-ZINN K.	BOTANIQUE

PROFESSEURS SANS CHAIRE

Mme	KOFLER L.	BOTANIQUE
	DEPASSEL R.	MECANIQUE
	PERRET R.	SERVOMECANISME
Mme	BARBIER M.J.	ELECTROCHIMIE
	COHEN J.	PHYSIQUE
	GIDON P.	GEOLOGIE
Mme	SOUTIF J.	PHYSIQUE GENERALE
	GIRAUD P.	GEOLOGIE
	GASTINEL N.	MATHEMATIQUES APPLIQUEES
	LACAZE A.	THERMODYNAMIQUE
	GLENAT R.	CHIMIE ORGANIQUE
	BRISSONNEAU P.	PHYSIQUE GENERALE
	DUCROS P.	MINERALOGIE
	ANGLES D'AURIAC	MECANIQUE DES FLUIDES
	ROBERT A.	CHIMIE PAPETIERE
	COUMES A.	ELECTRONIQUE
	PEBAY-PEROULA	PHYSIQUE
	DEGRANGE C.	ZOOLOGIE
	GAGNAIRE D.	CHIMIE PAPETIERE
	RASSAT A.	CHIMIE
	PERRIAUX J.	GEOLOGIE
	BARRA J.	MATHEMATIQUES APPLIQUEES

PROFESSEURS HONORAIRES

MM.	FORTIER A.	MECANIQUE DES FLUIDES
	BRELOT M.	MATHEMATIQUES
	WOLFERS F.	PHYSIQUE
	DORIER A.	ZOOLOGIE

MAITRES DE CONFERENCES

MM.	BIAREZ J.	MECANIQUE DES FLUIDES
	DODU J.	MECANIQUE DES FLUIDES

	DOLIQUE J.M.	ELECTRONIQUE
	HACQUES G.	MATHEMATIQUES APPLIQUEES
	LANCIA R.	PHYSIQUE AUTOMATIQUE
	POULOUJADOFF M.	ELECTROTECHNIQUE
	KAHANE A.	PHYSIQUE
Mme	BONNIER J.	CHIMIE
Mme	KAHANE J.	PHYSIQUE
	DEPORTES C.	CHIMIE MINERALE
	DEPOMMIER P.	PHYSIQUE NUCLEAIRE
	CAUQUIS G.	CHIMIE GENERALE
	BONNET G.	PHYSIQUE
Mme	BOUCHE L.	MATHEMATIQUES
	COLOBERT L.	PHYSIOLOGIE ANIMALE
	PAYAN J.J.	MATHEMATIQUES
	CAUBET J.P.	MATHEMATIQUES
	LAURENT P.	MATHEMATIQUES APPLIQUEES
	BERTRANDIAS J.P.	MATHEMATIQUES APPLIQUEES
	BRIERE G.	PHYSIQUE
	LAJZEROWICZ J.	PHYSIQUE
	VALENTIN J.	PHYSIQUE
	DESRE P.	METALLURGIE
	BONNETAIN L.	CHIMIE MINERALE

MAITRE DE CONFERENCES ASSOCIE

MM.	RADELLI L.	GEOLOGIE

Je tiens à exprimer ma plus vive gratitude à Monsieur le Professeur Gastinel qui a dirigé ce travail et qui a su, par ses conseils et encouragements, m'aider à le mener à bien.

Je remercie Monsieur le Professeur Kuntzmann et Monsieur Laurent, Maître de Conférences, de m'avoir fait l'honneur d'être président et membre du Jury.

J'adresse ma plus grande reconnaissance à Mademoiselle Tournoud et à Monsieur Mounet pour la part essentielle qu'ils ont pris dans l'accomplissement matériel de cette tâche.

Enfin, je suis très redevable à tous les membres du Laboratoire de Calcul de Grenoble de l'ambiance sympathique qui y règne, plus particulièrement à mes collègues de bureau Messieurs Robert, Petit, Miellou et Carasso qui ont su m'aider par leur amitié et leur dynamisme.

PREMIERE PARTIE

-:-:-:-

APPLICATION DE LA METHODE DES REDUITES A LA RESOLUTION DES EQUATIONS ALGEBRIQUES

-:-:-:-

INTRODUCTION

La méthode des réduites, appliquée à la recherche des racines d'une équation algébrique, permet d'obtenir la racine de plus petit module, quand elle est unique ; mais elle peut fort bien conduire, en procédant par déflation, aux k premières racines, si celles-ci sont toutes de modules différents. En cela, elle procède comme bien d'autres méthodes, entre autres comme la méthode de Newton, la méthode des puissances, les méthodes de Bernoulli, la méthode de Graeffe ... etc. Notons aussi qu'elle peut, si l'ordre des coefficients de l'équation est inversé, servir à traiter la plus grande racine en module, toujours quand elle est unique.

Dans un premier chapitre, en reprenant les idées de Fürstenau [2] et de Naegelsbach [3] ses deux auteurs, nous développerons cette méthode et nous montrerons comment l'adapter au calcul numérique.

Le chapitre suivant sera réservé à une présentation de l'algorithme Quotient-Différence, procédé de calcul, dû à Rutishauser (voir en [5] l'exposé de Monsieur Henrici), qui conduit globalement à toutes les racines, et plus généralement à tous les pôles d'une fonction méromorphe.

Le chapitre III nous permettra de montrer l'identité de la première colonne de l'algorithme Quotient-Différence avec la suite d'éléments donnée par la méthode des réduites, ou plutôt avec la suite des inverses de ces éléments.

et de comparer cette dernière avec la méthode de Bernoulli. Une démonstration, ne reposant pas sur les propriétés des séries du champ complexe et suggérée par la comparaison avec la méthode de Bernoulli, sera aussi donnée de la convergence de la méthode des réduites vers la racine de plus petit module.

Enfin le chapitre IV sera consacré aux exemples et vérifications numériques.

CHAPITRE I

LA METHODE DES REDUITES

I - LA METHODE DES REDUITES APPLIQUEE AU CALCUL DE LA RACINE DE PLUS PETIT MODULE D'UN POLYNOME, QUAND ELLE EST UNIQUE.

On s'est inspiré, dans l'exposé qui va suivre, de l'ouvrage de F. Riesz "Les systèmes d'équations linéaires à une infinité d'inconnues" ([1] p 9 - 10 - 11).

Etant donnée l'équation

$$① \quad c_o x^m + c_1 x^{m-1} + c_2 x^{m-2} + \ldots + c_{m-2} x^2 + c_{m-1} x + c_m = 0$$

(avec $c_m \neq 0$ et $c_o \neq 0$) ,
on se propose de calculer z_1 , sa racine de plus petit module, dans le cas où elle est unique. Le mérite de cette méthode revient à Fürstenau qui l'expose dans son mémoire [2] .

Formons le "système infini" suivant :

$$② \quad \begin{aligned} -c_m &= c_{m-1} x_1 + c_{m-2} x_2 + \ldots + c_o x_m \\ 0 &= c_m x_1 + c_{m-1} x_2 + \ldots + c_1 x_m + c_o x_{m+1} \\ 0 &= c_m x_2 + \ldots + c_2 x_m + c_1 x_{m+1} + c_o x_{m+2} \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{aligned}$$

On voit le procédé de formation du système : il consiste

à multiplier successivement l'équation ① par des puissances de plus en plus élevées de x et à remplacer x^n par x_n .

Le résultat démontré par Fürstenau est le suivant :

$$③ \qquad z_1 = - c_m \times \lim_{n\to\infty} \frac{\Delta_{n-1}}{\Delta_n} \quad (\Delta_o = 1) ,$$

où Δ_n est le déterminant formé des coefficients de x_1 , ..., x_n dans les n premières équations. Ce résultat se trouve démontré dans le mémoire de Fürstenau d'une façon tout-à-fait rigoureuse, mais par un calcul assez laborieux. D'après une remarque de H. Naegelsbach [3] , il est plus simple de montrer que la fonction

$$R(x) = \frac{1}{c_m + c_{m-1}\, x + \ldots\ldots + c_o\, x^m}$$

admet le développement en série entière

$$④ \quad R(x)= \frac{1}{c_m} - \frac{\Delta_1}{c_m^2} x + \frac{\Delta_2}{c_m^3} x^2 - \frac{\Delta_3}{c_m^4} x^3 + \ldots. + (-1)^p \frac{\Delta_p}{c_m^{p+1}} x^p + \ldots ;$$

Nous vérifierons d'abord ce résultat et nous verrons ensuite comment la formule ③ peut s'en déduire.

c_m n'étant pas nul, O n'est pas racine de l'équation ① et R(x) peut effectivement se développer en série entière pour $|x| < |z_1|$.

Posons :

$$R(x) = a_o + a_1 x + a_2 x^2 + \ldots\ldots\ldots\ldots + a_k x^k + \ldots$$

Dans ce domaine, l'identité

$$1 \equiv (c_m + c_{m-1}\, x + \ldots + c_o x^m) \times (a_o + a_1 x + a_2 x^2 + \ldots + a_k\, x^k + \ldots),$$

nous permet de calculer les coefficients a_p pour $p = 0,1,2\ldots$

Nous avons successivement :

$$\begin{aligned}
&c_m \quad a_o && = 1\\
&c_{m-1} \quad a_o + c_m\, a_1 && = 0\\
&c_{m-2} \quad a_o + c_{m-1}\, a_1 + c_m\, a_2 && = 0\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots
\end{aligned}$$

D'une façon générale, pour $p \geqslant 1$, a_p est donné par

⑤ $$c_o\, a_{p-m} + c_1\, a_{p-m+1} + \ldots + c_{m-1}\, a_{p-1} + c_m\, a_p = 0 ,$$

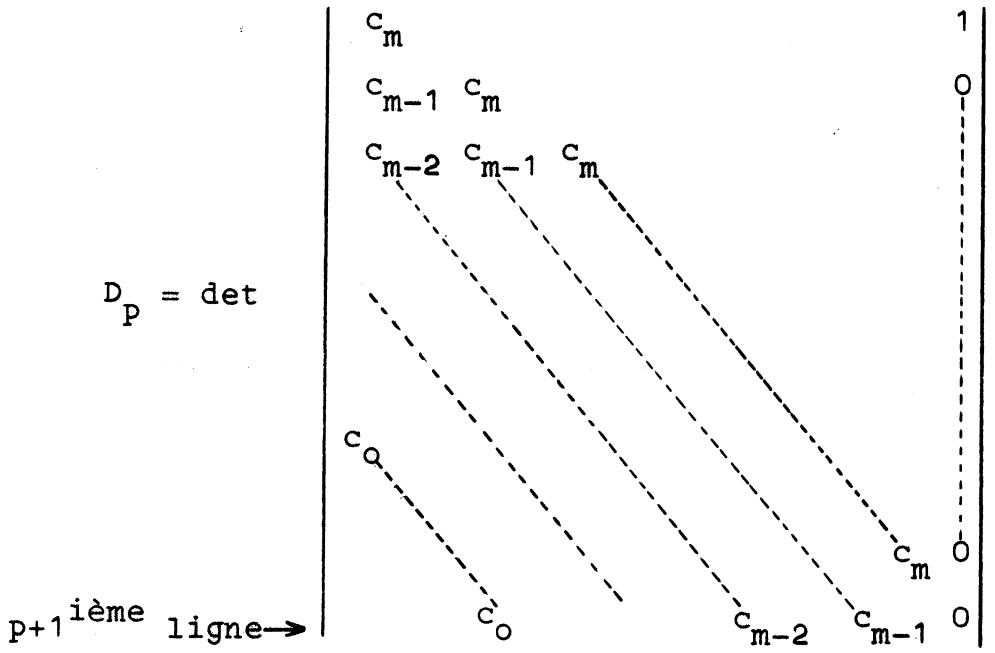
en considérant que tous les coefficients a_i d'indices $i < 0$ sont nuls. Les $p+1$ premiers coefficients a_o , a_1 , $\ldots$ a_p sont solution du système linéaire de $p+1$ équations à $p+1$ inconnues :

$$\begin{bmatrix}
c_m & & & & \\
c_{m-1} & c_m & & & \\
c_{m-2} & c_{m-1} & c_m & & \\
\ddots & \ddots & \ddots & \ddots & \\
c_c & \ddots & \ddots & \ddots & \\
 & \ddots & \ddots & \ddots & \\
 & & c_o & c_{m-2} & c_{m-1} & c_m
\end{bmatrix}
\times
\begin{bmatrix} a_o \\ a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix}
=
\begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} .$$

De ce système, on tire

$$a_p = \frac{D_p}{c_m^{p+1}}$$

où c_m^{p+1} est la valeur du déterminant des coefficients et



Remarque : afin de conserver les notations d'indices précédentes D_p est un déterminant d'ordre $p+1$.

Par suite, en développant D_p suivant la dernière colonne et en reprenant les notations initiales :

$$D_p = (-1)^p \quad \Delta p \qquad (p = 0, 1, \ldots \; ; \Delta o = 1)$$

d'où :

⑥ $$a_p = (-1)^p \quad \frac{\Delta p}{c_m^{p+1}}$$

Pour achever la démontration il suffit, A étant le résidu de $R(x)$ par rapport à z_1 , de remarquer que les pôles

de la fraction rationnelle

$$r(x) = R(x) - \frac{A}{1 - \frac{x}{z_1}} \quad ,$$

sont plus éloignés de l'origine que le point z_1 ; la série :

$$r(x) = \frac{1}{c_m} - A + \left(-\frac{\Delta_1}{c_m^2} - \frac{A}{z_1}\right)x + \left(\frac{\Delta_2}{c_m^3} - \frac{A}{z_1^2}\right)x^2 + \ldots + \left((-1)^n \frac{\Delta_n}{c_m^{n+1}} - \frac{A}{z_1^n}\right) x^n + \ldots$$

converge donc pour $x = z_1$;

En particulier ses termes tendent vers zéro et par conséquent

$$\frac{\Delta_n \times (-z_1)^n}{c_m^{n+1}} \xrightarrow[n\to\infty]{} A$$

Il en résulte que

$$-c_m \frac{\Delta_{n-1}}{\Delta_n} = z_1 \times \frac{\dfrac{\Delta_{n-1} \times (-z_1)^{n-1}}{c_m^n}}{\dfrac{\Delta_n \times (-z_1)^n}{c_m^{n+1}}} \xrightarrow[n\to\infty]{} z_1$$

Notons que si nous reprenons la notation

$$R(x) = a_o + a_1 x + a_2 x^2 + \ldots\ldots + a_p x^p + \ldots\ldots ,$$

un résultat dû à König [7] nous permet d'éviter cette dernière partie de la démonstration. En effet, König montra en 1884 que si la seule singularité sur le cercle de convergence de $R(x)$ est un pôle simple z_1 , on a :

⑦ $$z_1 = \lim_{p \to \infty} -\frac{a_{p-1}}{a_p}$$

Par conséquent

$$z_1 = \lim_{p \to \infty} -\frac{(-1)^{p-1} \frac{\Delta_{p-1}}{c_m^p}}{(-1)^p \frac{\Delta_p}{c_m^{p+1}}}$$

d'où notre résultat.

Remarque : l'exposé précédent est valable aussi bien pour z_1 complexe que pour z_1 réelle. Cependant, la restriction "z_1 unique" impose, dans le cas des polynômes à coefficients réels, que z_1 soit une racine réelle, simple et telle que $-z_1$ ne soit pas racine.

II-UTILISATION DES RESULTATS PRECEDENTS DANS UN ALGORITHME CONDUISANT A z_1.

Les formules ③ ou ⑦ du paragraphe précédent donnent une suite $z_1^{(1)}, z_1^{(2)}, \ldots, z_1^{(n)}, \ldots$ qui tend vers z_1.

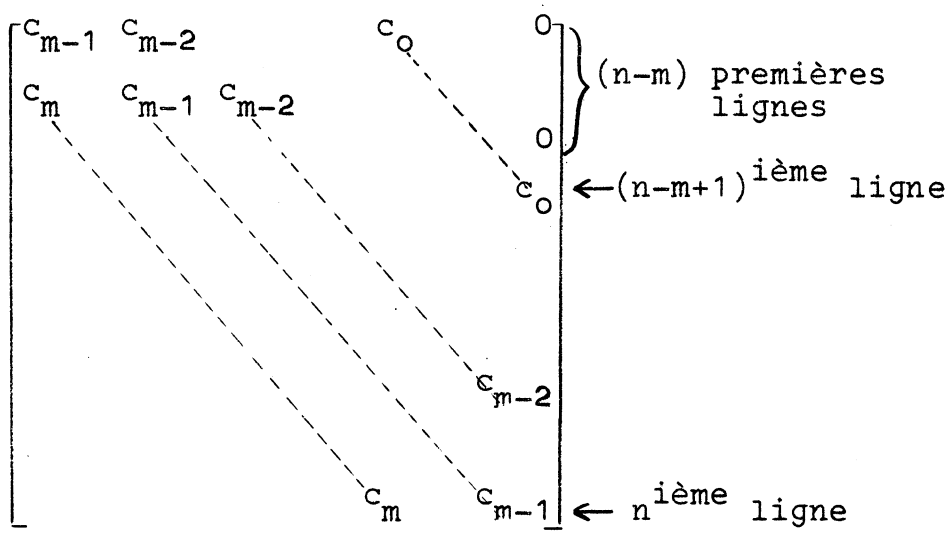
En utilisant les deux variantes permettant d'achever la démonstration précédente, nous obtenons

⑧ $$z_1^{(n)} = -c_m \frac{\Delta_{n-1}}{\Delta_n} = \frac{a_{n-1}}{a_n} \quad (n = 1, 2, \ldots; \Delta_0 = 1)$$

Il s'agit de trouver un moyen commode de calculer numériquement cette suite. Pour cela, nous allons essayer d'obtenir le rapport $\frac{\Delta_{n-1}}{\Delta_n}$ sans calculer Δ_{n-1} ni Δ_n.

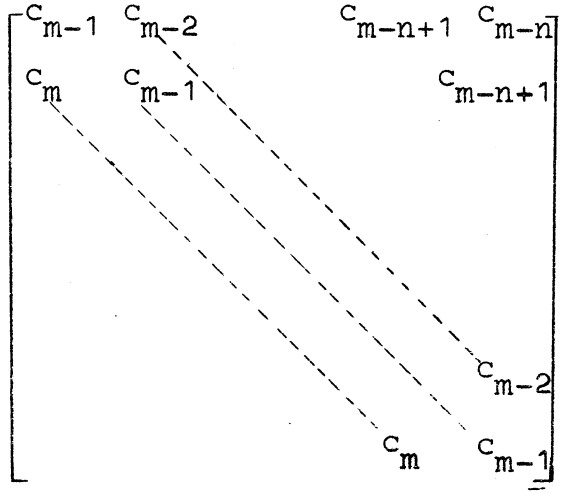
Quand $n > m$, Δ_n est de la forme suivante :

$$\Delta_n = \det$$



Lorsque $n \leqslant m$ Δ_n est par contre de la forme :

$$\Delta_n = \det$$



Une méthode simple consiste à trianguliariser cette matrice par l'algorithme de Gauss. Soient $c_{m-1}^{(1)}, c_{m-1}^{(2)}, \ldots, c_{m-1}^{(n)}$ la suite des pivots obtenus au cours des n étapes de la triangularisation. Nous avons alors

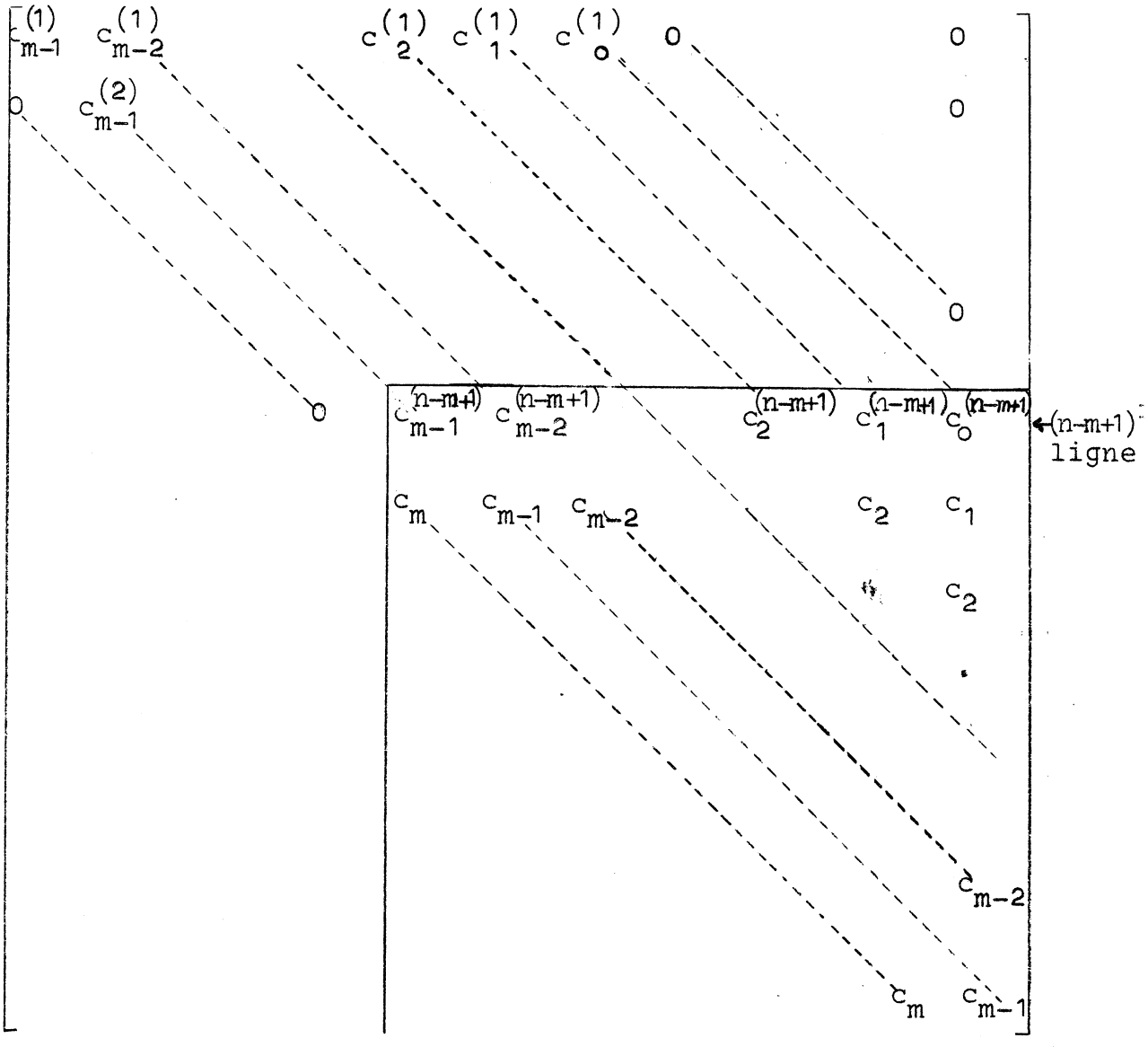
$$z_1^{(n)} = - c_m \times \frac{\Delta_{n-1}}{\Delta_n} = - c_m \times \frac{1}{c_{m-1}^{(n)}} \quad .$$

Il semble donc nécessaire d'obtenir $c_{m-1}^{(n)}$ sans passer par le calcul de Δ_n qui donnerait lieu, dans la plupart des cas, à des dépassements de capacité.

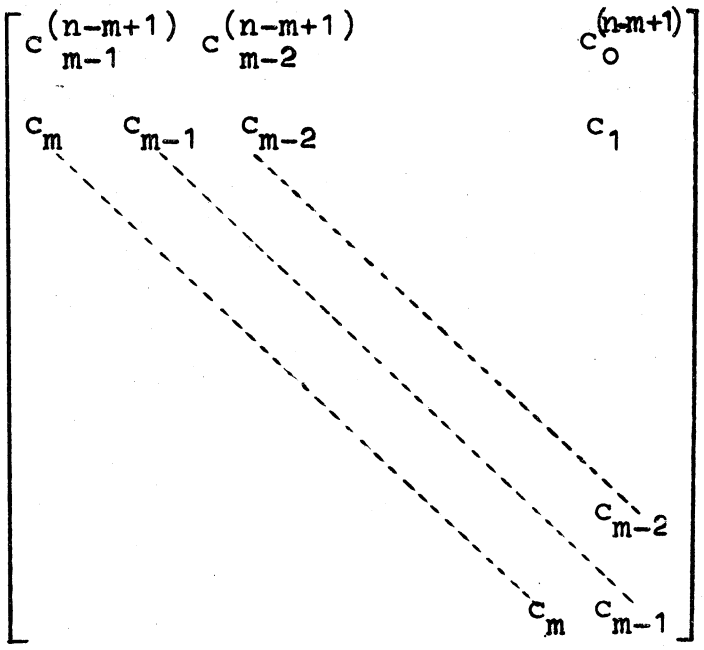
Nous envisagerons directement le cas $n > m$ et nous verrons ensuite comment il est possible d'y inclure le cas $n \leqslant m$.

Supposons que nous ayions effectué les (n-m+1) premières étapes de l'algorithme de Gauss. Nous avons alors la disposition suivante :

⑨



Il reste à régulariser la matrice suivante (d'ordre m):



ce qui fait encore (m-1) étapes pour achever la triangularisation.

Nous avons alors successivement :

$$\begin{vmatrix} c_{m-1}^{(n-m+1)} & c_{m-2}^{(n-m+1)} & \cdots\cdots\cdots\cdots & c_{o}^{(n-m+1)} \\ 0 & c_{m-1}^{(n-m+2)} & \cdots\cdots\cdots\cdots & c_{1}^{(n-m+2)} \\ 0 & c_{m} & \cdots\cdots\cdots\cdots & c_{2} \end{vmatrix}$$

avec $$c_{1}^{(n-m+2)} = c_{1} - \frac{c_{m}}{c_{m-1}^{(n-m+1)}} \; c_{o}^{(n-m+1)}$$

et comme l'élément immédiatement au dessus de $c_{o}^{(n-m+1)}$ est nul,

$c_{o}^{(n-m+1)} = c_{o}$

Donc $$c_{1}^{(n-m+2)} = c_{1} + c_{o} \; z_{1}^{(n-m+1)} \; .$$

Ensuite nous avons :

$$\begin{vmatrix} c_{m-1}^{(n-m+1)} & c_{m-2}^{(n-m+1)} & c_{m-3}^{(n-m+1)} & \cdots\cdots\cdots & c_{o}^{(n-m+1)} \\ 0 & c_{m-1}^{(n-m+2)} & c_{m-2}^{(n-m+2)} & \cdots\cdots\cdots & c_{1}^{(n-m+2)} \\ 0 & 0 & c_{m-1}^{(n-m+3)} & \cdots\cdots\cdots & c_{2}^{(n-m+3)} \\ 0 & 0 & c_{m} & \cdots\cdots\cdots & c_{3} \end{vmatrix}$$

avec
$$c_{2}^{(n-m+3)} = c_2 - \frac{c_m}{c_{m-1}^{(n-m+2)}} c_1^{(n-m+2)}$$

$$= c_2 + c_1^{(n-m+2)} z_1^{(n-m+2)}$$

$$= c_2 + c_1 z_1^{(n-m+2)} + c_o z_1^{(n-m+2)} z_1^{(n-m+1)}$$

Nous obtenons ainsi tous les éléments de la dernière colonne de la matrice à triangulariser. Pour démontrer par récurrence que $c_{m-1}^{(n)}$ est de cette forme, nous allons supposer que

$$c_{m-2}^{(n-1)} = c_{m-2} + c_{m-3} z_1^{(n-2)} + c_{m-4} z_1^{(n-2)} z_1^{(n-3)} + \ldots + c_o z_1^{(n-2)} \times \ldots \times z_1^{(n-m+1)} .$$

Nous avons alors la disposition suivante :

$$\begin{bmatrix} c_{m-1}^{(n-m+1)} & & & c_o^{(n-m+1)} \\ & c_{m-1}^{(n-m+2)} & & c_1^{(n-m+2)} \\ & & c_{m-1}^{(r-m+3)} & c_2^{(n-m+3)} \\ & & & \ddots \\ & & & c_{m-1}^{(n-1)} \; c_{m-2}^{(n-1)} \\ & & & c_m \quad c_{m-1} \end{bmatrix}$$

qui nous conduit à

$$c_{m-1}^{(n)} = c_{m-1} - \frac{c_m}{c_{m-1}^{(n-1)}} \; c_{m-2}^{(n-1)}$$

ou bien

$$c_{m-1}^{(n)} = c_{m-1} + c_{m-2}^{(n-1)} \; z_1^{(n-1)}$$

ce qui achève la démontration par récurrence et nous donne :

⑩ $$c_{m-1}^{(n)} = c_{m-1} + c_{m-2} \, z_1^{(n-1)} + c_{m-3} \, z_1^{(n-1)} z_1^{(n-2)} + .. + c_o \, z_1^{(n-1)} x .. \\ .. x \, z_1^{(n-m+1)} .$$

lorsque $n \leqslant m$, le problème est le même, la sous-matrice que nous avons encadrée dans la disposition ⑨ et sur laquelle nous avons fait porter la démonstration par récurrence étant cette fois directement la matrice complète.

Le dernier pivot $c_{m-1}^{(n)}$ s'exprime alors de la façon suivante :

$$c_{m-1}^{(n)} = c_{m-1} + c_{m-2} \, z_1^{(n-1)} + c_{m-3} \, z_1^{(n-1)} z_1^{(n-2)} + .. + c_{m-n} z_1^{(n-1)} x .. x \, z_1^{(1)}$$

Afin de simplifier les notations, nous emploierons dans les deux cas la formule ⑩ en adoptant la convention suivante :

Nous considérerons que tous les éléments z_i de la suite, tels que leur indice i est inférieur strictement à 1, sont nuls.

Ceci étant acquit, le terme $z_1^{(n)}$ s'exprime de la façon suivante :

$$(11)\quad z_1^{(n)} = -\frac{c_m}{c_{m-1} + c_{m-2}\, z_1^{(n-1)} + c_{m-3}\, z_1^{(n-1)} z_1^{(n-2)} + \ldots + c_0 z_1^{(n-1)} \times \ldots \times z_1^{(n-m+1)}}$$

formule qui est valable quel que soit n en conservant la convention précédente.

Remarques :

- $z_1^{(1)} = \dfrac{-c_m}{c_{m-1}^{(1)}} = -\dfrac{c_m}{c_{m-1}}$

- le calcul d'un des éléments de la suite, lorsque $n > m$, utilise les $m-1$ éléments précédents ; lorsque $n \leqslant m$, il se sert de tous les éléments précédents.

- Pour l'obtention effective du terme $z_1^{(n)}$, nous apporterons à la formule (11) une modification dans le calcul du dénominateur. Celui-ci sera traité par un schéma analogue au schéma de Horner. Pour "représenter" le calcul de $z_1^{(n)}$, nous donnerons la formule schématique suivante :

$$(12)\quad z_1^{(n)} = -\frac{c_m}{(((\, c_0 x z_1^{(n-m+1)} + c_1) x z_1^{(n-m+2)} + c_2) x \ldots + c_{m-2}) x z_1^{(n-1)} + c_{m-1}}$$

ainsi que les instructions ALGOL qui explicitent la "boucle" du dénominateur. Si les coefficients du polynôme sont rangés dans un tableau C (c_0 étant dans C [o] ,...), les termes de la suite dans un tableau Z et le dénominateur accumulé dans un réel D, le calcul de $z_1^{(n)}$ sera effectué selon le groupe d'instructions :

```
D  :=  C [0] ;
pour i  :=  1  pas  1  jusquà  M-1  faire
D  :=  D x Z[N-M+i] +  C [i] ;
Z [N]  :=  - C [M]/ D  ;
```

CHAPITRE II

L'ALGORITHME QUOTIENT-DIFFERENCE

I - INTRODUCTION.

Nous suivons, dans cet exposé, le travail de Monsieur Henrici [5] , l'algorithme Quotient-Différence étant dû à Monsieur Rutishauser.

L'algorithme Quotient-Différence (noté algorithme Q D) est avant tout une méthode de détermination des pôles d'une fonction méromorphe à partir des coefficients de son développement de Taylor à l'origine. Mais il peut être appliqué à des problèmes aussi divers que le calcul des valeurs propres et vecteurs propres d'une matrice, la détermination du développement en fraction continue d'une fonction donnée par une série de puissances, le calcul des zéros d'un polynôme ... etc.

Deux problèmes se posent, qui sont en relation directe avec la recherche des racines d'une équation algébrique :

problème I : Etant donné le développement de Taylor à l'origine d'une fonction $f(z)$ rationnelle, déterminer les pôles de $f(z)$.

problème II : Etant donné le développement de Taylor à l'origine d'une fonction $f(z)$ méromorphe dans $|z| < R$, déterminer les pôles de $f(z)$ dans ce domaine.

Nous montrerons comment l'algorithme Q D permet de

résoudre ces 2 problèmes et plus particulièrement (théorème IV, fin du chapitre) le problème I dans le cas où il n'y a qu'un seul pôle z_k sur le cercle $(0, |z_k|)$.

Afin de ne pas trop alourdir l'exposé et aussi parce que ce travail n'est pas centré sur l'algorithme Q D dans sa généralité, nous avons donné certains lemmes ou théorèmes sans démonstration, en renvoyant aux ouvrages appropriés.

II - LEMMES PREPARATOIRES A L'ETUDE DE L'ALGORITHME QUOTIENT-DIFFERENCE.

Le problème de la détermination des pôles d'une fonction méromorphe est aussi ancien que la théorie des fonctions analytiques elle-même. König [7] (1884) et Hadamard s'y sont intéressés.

1°) Déterminants de Hankel

Ces déterminants constituent un outil de base dans l'étude des singularités de la fonction f(z) dont $\sum_{n=0}^{\infty} a_n z^n$ est le développement à l'origine.

On pose :

$$H_0^{(n)} = 1$$

$$\textcircled{1} \qquad H_m^{(n)} = \begin{vmatrix} a_n, & a_{n+1}, & \cdots\cdots & a_{n+m-1} \\ a_{n+1}, & a_{n+2}, & \cdots\cdots & a_{n+m} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n+m-1}, & a_{n+m}, & \cdots & a_{n+2m-2} \end{vmatrix} \quad \begin{matrix} n = 0, 1, 2, \ldots \\ m = 1, 2, \ldots \end{matrix}$$

Remarque : Si on considère que, pour $n < 0$, $a_n = 0$ dans le développement de $f(z)$, on peut fort bien envisager $H_m^{(n)}$ avec $n<0$. Par contre m est nécessairement supérieur ou égal à 0.

Des propriétés de ces déterminants vont résulter la méthode d'Aitken (§ III) puis l'algorithme Quotient-Différence (§ IV) qui améliore considérablement cette dernière.

2°) Propriétés des déterminants de Hankel :

Lemme 1 :

pour toutes valeurs positives de m on a

$$\textcircled{2} \qquad \left[H_m^{(n)}\right]^2 - H_m^{(n-1)} H_m^{(n+1)} + H_{m+1}^{(n-1)} H_{m-1}^{(n+1)} = 0$$

On se reportera à [5] p.25 pour la démonstration. Notons cependant que la relation ② est valable quel que soit n positif ou négatif.

Lemme 2 :

Soit $f(z) = f_o(z) + r(z)$ où $f_o(z)$ est holomorphe dans $|z| \leqslant R = \frac{1}{\Lambda}$ et $r(z)$ rationnelle. Soit $\sum_{n=o}^{\infty} a_n z^n$ le développement de Taylor à l'origine de $f(z)$.

Soit $z_j = \frac{1}{\lambda_j}$ les pôles de $r(z)$ (d'ordres respectifs m_j, $j = 1, 2, \ldots, k$).

Supposons que ces pôles satisfont à :

$$0 < |z_1| \leqslant |z_2| \leqslant \cdots \leqslant |z_k| < R .$$

Alors, si $m = \sum_{j=1}^{k} m_j$,

$$\textcircled{3} \qquad H_m^{(n)} = C(\lambda_1^{m_1} \cdot \lambda_2^{m_2} \times \ldots \times \lambda_k^{m_k})^n \left\{1+O\left(\left(\frac{\Lambda}{\lambda_k}\right)^n\right)\right\} \qquad (*)$$

où C est une constante non nulle indépendante de n.

(*) On rappelle que la notation $y=O(x)$ signifie qu'il existe une constante K non nulle telle que $|y| \leqslant K |x|$

Ce lemme est essentiel pour la suite de cette étude.

Nous ferons la démonstration dans le cas où tous les pôles sont simples, c'est-à-dire $m_j = 1$ et $k = m$. On se reportera à [6] pour la démonstration générale.

On peut alors écrire :

$$f(z) = \frac{\alpha_1}{z-z_1} + \frac{\alpha_2}{z-z_2} + \ldots\ldots + \frac{\alpha_m}{z-z_m} + f_o(z)$$

avec des α_j non nuls.

On a : $\frac{\alpha_j}{z-z_j} = \frac{\alpha_j}{-z_j(1-\frac{z}{z_j})} = -\frac{\alpha_j}{z_j}(1-\frac{z}{z_j})^{-1}$,

c'est-à-dire $\frac{\alpha_j}{z-z_j} = -\frac{\alpha_j}{z_j}\sum_{n=o}^{\infty}(\frac{z}{z_j})^n$, dans le cercle $|z|<|z_j|$,

ou bien $\frac{\alpha_j}{z-z_j} = -\alpha_j \lambda_j \sum_{n=o}^{\infty} \lambda_j^n z^n$.

Finalement, on obtient le développement en puissances de z $f(z) = \sum_{n=o}^{\infty} a_n z^n$, avec

④
$$\begin{cases} a_n = A_1 \lambda_1^n + A_2 \lambda_2^n + \ldots\ldots + A_m \lambda_m^n + b_n \\ A_j = -\alpha_j \lambda_j . \end{cases}$$

Le terme b_n, dans le coefficient a_n résulte uniquement du développement en puissances de z de la fonction $f_o(z)$ holomorphe dans le cercle $|z| \leqslant R$. Le développement de $f(z)$ précédent est valable pour $|z| < |z_1|$. On a de plus, pour une constante M convenable, l'inégalité suivante, provenant de

la formule de Cauchy :

⑤ $$|b_n| < M \Lambda^n$$

Nous allons essayer d'évaluer le déterminant $H_m^{(n)}$.

Par linéarité d'un déterminant par rapport à ses colonnes, on peut écrire

⑥ $$H_m^{(n)} = \sum D_m^{(n)} + \sum \hat{D}_m^{(n)} ,$$

les déterminants $\hat{D}_m^{(n)}$ se distinguant des $D_m^{(n)}$ par le fait qu'ils contiennent au moins une colonne de b, les $D_m^{(n)}$ n'en contenant pas. Il reste à préciser la signification des sommations.
On a :

⑦ $$D_m^{(n)} = \begin{vmatrix} A_{k_1} \lambda_{k_1}^n , A_{k_2} \lambda_{k_2}^{n+1} , \ldots\ldots , A_{k_m} \lambda_{k_m}^{n+m-1} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ A_{k_1} \lambda_{k_1}^{n+m-1} , A_{k_2} \lambda_{k_2}^{n+m} , \ldots\ldots , A_{k_m} \lambda_{k_m}^{n+2m-2} \end{vmatrix} .$$

Dans $\sum D_m^{(n)}$, la sommation est étendue à toutes les

permutations avec répétitions k_1 , k_2 , ... k_m des nombres 1, 2, ..., m.

Dans la seconde somme, $\hat{D}^{(n)}_m$ est un déterminant de la même forme que $D^{(n)}_m$, à ceci près que au moins une de ses colonnes

⑧
$$\begin{bmatrix} A_{k_\mu} \lambda_{k_\mu}^{n+\mu-1} \\ A_{k_\mu} \lambda_{k_\mu}^{n+\mu} \\ \vdots \\ A_{k_\mu} \lambda_{k_\mu}^{n+\mu+m-2} \end{bmatrix} \qquad \mu = 1, 2, \ldots, m ,$$

est remplacée par la colonne correspondante :

⑨
$$\begin{bmatrix} b_{n+\mu-1} \\ b_{n+\mu} \\ \vdots \\ b_{n+\mu+m-2} \end{bmatrix} .$$

La question de la sommation des $\hat{D}^{(n)}_m$ est un peu délicate. Elle est étendue à toutes les combinaisons (avec répétitions) k_1, k_2, ..., k_m des nombres 1, 2, ..., m-1, m, s qui contiennent au moins une fois le nombre s (ici la valeur de s n'importe pas, car s n'est évidemment qu'un repère). La ou les, $\mu^{\text{ième}}$ colonnes, celles qui correspondent à $k_\mu = s$ dans la permutation, étant justement remplacées par la, ou les, colonnes ⑨ .

exemple : m = 3

une combinaison précédente pourra être $k_1 = 2$, $k_2 = s$, $k_3 = s$. Les deuxième et troisième colonnes sont à remplacer , le $D_3^{(n)}$ correspondant étant alors :

$$\begin{vmatrix} A_2 \lambda_2^n & b_{n+1} & b_{n+2} \\ A_2 \lambda_2^{n+1} & b_{n+2} & b_{n+3} \\ A_2 \lambda_2^{n+2} & b_{n+3} & b_{n+4} \end{vmatrix}$$

$$\uparrow \mu=2 \qquad \uparrow \mu=3$$

Evaluons maintenant la 1ère somme.

On peut d'abord constater que tous les déterminants $D_m^{(n)}$ correspondant à des combinaisons $k_1 \; k_2 \ldots k_m$ à éléments répétés, sont nuls, car si $k_i = k_j = k (i > j)$ on met λ_k^{i-j} en facteur dans la $i^{ème}$ colonne et l'on obtient 2 colonnes identiques. Il suffit donc d'étendre la sommation à toutes les permutations.

Alors :

⑩ $$\sum D_m^{(n)} = \prod_{i=1}^{m} A_i \lambda_i^n \cdot P_m \quad ,$$

avec :

⑪ $$P_m = \sum \begin{vmatrix} \lambda_{k_1}^0 & \lambda_{k_2}^1 & \ldots\ldots\ldots & \lambda_{k_m}^{m-1} \\ \lambda_{k_1}^1 & \lambda_{k_2}^2 & \ldots\ldots\ldots & \lambda_{k_m}^{m} \\ \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots \\ \lambda_{k_1}^{m-1} & \lambda_{k_2}^{m} & \ldots\ldots\ldots & \lambda_{k_m}^{2m-2} \end{vmatrix} \quad ,$$

la sommation étant étendue à toutes les permutations $k_1 \; k_2 .. k_m$ des nombres 1, 2, ..., m.

On va montrer que $P_m \neq 0$.

$P_m = P_m(\lambda_1, \lambda_2, \ldots, \lambda_m)$ est un polynôme homogène de degré $m(m-1)$ en les variables $\lambda_1, \ldots, \lambda_m$. Il s'annule quand 2 λ_i sont égaux et ne change pas de valeur quand 2 λ_i sont intervertis car P_m est une somme étendue à toutes les permutations.

P_m contient donc tous les facteurs $(\lambda_i - \lambda_j)^2$ avec $1 \leqslant i < j \leqslant m$. Comme ce produit de facteurs est de degré $m(m-1)$, on en déduit que :

⑫ $$P_m = c_m \prod_{1 \leqslant i < j \leqslant m} (\lambda_i - \lambda_j)^2 \quad ,$$

c_m étant une constante numérique.

Nous allons montrer que $c_m = 1$ par récurrence sur m. Pour $m = 1$ on a : $D_1^{(n)} = A_1 \lambda_1^n$; $\sum D_1^{(n)} = A_1 \lambda_1^n P_1 = A_1 \lambda_1^n$; donc $P_1 = 1$; par suite $c_1 = 1$.

Supposons que $c_{m-1} = 1$ et imposons $\lambda_1 = 0$ dans la formule ⑪ . Seuls, les déterminants tels que $k_1 = 1$ ne sont pas nuls. On obtient :

$$P_m(0, \lambda_2, \ldots, \lambda_m) = \sum \begin{vmatrix} 1 & \lambda_{k_2}^1 & \cdots\cdots & \lambda_{k_m}^{m-1} \\ 0 & \lambda_{k_2}^2 & \cdots\cdots & \lambda_{k_m}^{m} \\ \vdots & \vdots & & \vdots \\ 0 & \lambda_{k_2}^m & \cdots\cdots & \lambda_{k_m}^{2m-2} \end{vmatrix} ,$$

la sommation étant étendue à toutes les permutations $k_2, \ldots k_m$ des nombres 2, ... , m. Donc :

$$P_m(0, \lambda_2, \ldots, \lambda_m) = \sum_{\substack{\text{même} \\ \text{sommation}}} 1 \times \begin{vmatrix} \lambda_{k_2}^2 & \cdots & \lambda_{k_m}^m \\ \vdots & & \vdots \\ \lambda_{k_2}^m & \cdots & \lambda_{k_m}^{2m-2} \end{vmatrix} .$$

Par suite :

$$P_m(0, \lambda_2, \ldots, \lambda_m) = (\lambda_2 \times \ldots \times \lambda_m)^2 \times P_{m-1}(\lambda_2, \ldots, \lambda_m) .$$

Donc :

$$P_m(0, \lambda_2, \ldots, \lambda_m) = (\lambda_2 \times \ldots \times \lambda_m)^2 \times \prod_{2 \leqslant i < j \leqslant m} (\lambda_i - \lambda_j)^2$$

En comparant cette expression de P_m à la formule (12) évaluée avec $\lambda_1 = 0$, on déduit que $c_m = 1$.

Ceci prouve que, dans (10) , $P_m \not\equiv 0$.

En observant (12) (avec $c_m = 1$) et (10) , on conclut que :

$$(13) \qquad \sum D_m^{(n)} = C (\lambda_1 \times \lambda_2 \times \ldots \times \lambda_m)^n ,$$

C étant une constante non nulle et indépendante de n (d'une façon plus précise $C = \prod_{i=1}^{m} A_i \times \prod_{1 \leqslant i < j \leqslant m} (\lambda_i - \lambda_j)^2$). Il est très simple d'autre part de voir que $D_m^{(n)} = K(\lambda_1 \times \lambda_2 \times \ldots \times \lambda_m)^n$ avec K indépendant de n.

Pour estimer la seconde somme dans (6) on remarque d'abord que les seuls $\hat{D}_m^{(n)}$ non nuls sont ceux pour lesquels les indices k_μ des colonnes autres que les colonnes de b_i , sont tous différents.

Puisque dans $\hat{D}_m^{(n)}$ on a au moins une colonne de b_i , on déduit de (5) que pour n tendant vers l'infini :

$$\hat{D}_m^{(n)} = 0\ ((\lambda_1 \times \lambda_2 \times \ldots \times \lambda_{m-1} \times \Lambda)^n\ )$$

Puisque $\sum \hat{D}_m^{(n)}$ est une somme finie, elle est du même ordre de grandeur.

Ceci ajouté à la relation (13) achève de démontrer le lemme 2 dans le cas où tous les pôles sont simples.

Corollaire 2.1

pour n assez grand :

(14) $$H_m^{(n)} \neq 0$$

Corollaire 2.2

Quand n tend vers l'infini :

(15) $$H_m^{(n+1)}/H_m^{(n)} \longrightarrow \lambda_1^{m_1} \times \lambda_2^{m_2} \times \ldots \times \lambda_k^{m_k}$$

Remarques :

- Ces deux corollaires sont évidemment valables dans les mêmes hypothèses que le lemme 2 .
- Le corollaire 2.2 est donné sous sa forme la plus générale.

Le corollaire 2.2 apporte une réponse aux problèmes I et II dans le cas où tous les pôles recherchés sont de modules différents. En effet, la relation (15) est valable pour $m = 1, 2, \ldots$ et fournit, quand n tend vers l'infini, successivement les produits des m premiers λ_i, lorsque les hypothèses du lemme 2 sont vérifiées. En particulier, il est nécessaire qu'on puisse trouver un cercle ouvert contenant m pôles et n'en contenant pas m+1.

Dans le cas où $f(z)$ est une fraction rationnelle, le calcul du produit de tous les λ_i est grandement facilité. En effet, on peut trouver un cercle ouvert de rayon R contenant tous les pôles et par suite $f(z) = r(z)$ et $f_0(z) \equiv 0$. Les coefficients b_n provenant du développement de $f_0(z)$ sont nuls. Donc

$$H_m^{(n)} = C(\lambda_1^{m_1} \times \lambda_2^{m_2} \times \ldots \times \lambda_k^{m_k})^n$$

et

$$H_m^{(n+1)}/H_m^{(n)} = \lambda_1^{m_1} \times \lambda_2^{m_2} \times \ldots \times \lambda_k^{m_k}$$

quel que soit n.

III - EMPLOI DES DETERMINANTS DE HANKEL PAR LA METHODE D'AITKEN.

Numériquement, la réponse apportée par le corollaire 2.2 est insuffisante tant qu'on ne sait pas calculer les déterminants de Hankel. La principale contribution d'Aitken fut de remarquer que le lemme 1 pouvait fournir des $H_k^{(n)}$ récursivement. On le voit très bien sur le schéma suivant :

⑯

$$\begin{array}{ccccccc}
H_0^{(0)} & & & & & & \\
H_0^{(1)} & H_1^{(0)} & & & & & \\
H_0^{(2)} & H_1^{(1)} & H_2^{(0)} & & & & \\
H_0^{(3)} & H_1^{(2)} & H_2^{(1)} & H_3^{(0)} & & & \\
H_0^{(4)} & H_1^{(3)} & H_2^{(2)} & H_3^{(1)} & H_4^{(0)} & & \\
H_0^{(5)} & H_1^{(4)} & H_2^{(3)} & H_3^{(2)} & H_4^{(1)} & H_5^{(0)} &
\end{array}$$

Rappelons la formule ②

$$\left[H_k^{(n)} \right]^2 - H_k^{(n-1)} H_k^{(n+1)} + H_{k+1}^{(n-1)} H_{k-1}^{(n+1)} = 0 .$$

Cette formule relie, par exemple, les déterminants notés sur le schéma qu'on peut construire de deux façons différentes :

- soit en le faisant évoluer de gauche à droite, les deux premières colonnes étant respectivement une colonne de 1 et une colonne des a_i
- soit, si ses deux premières diagonales sont connues, en le faisant progresser de haut en bas. Cette façon de procéder présente l'avantage d'élaborer toutes les colonnes simultanément, mais pose le problème de déterminer les deux premières diagonales. Aitken a montré comment les choisir dans le cas de la recherche des racines d'un polynôme [8].

Avec le schéma d'Aitken, le rapport $H_m^{(n+1)} / H_m^{(n)}$ conduit, dans le cas de pôles de modules différents, et quand n augmente indéfiniment, aux produits des m premiers $\lambda_i (m=1,2,..)$

IV - DEFINITION DE L'ALGORITHME QUOTIENT-DIFFERENCE.

L'algorithme Quotient-Différence est une forme améliorée de la méthode d'Aitken. On peut le caractériser de la façon suivante :

- l'algorithme Q D donne directement des nombres qui tendent vers les inverses des pôles de la fonction considérée.
- la détermination des deux premières diagonales (comme dans la méthode d'Aitken) est résolue pour les problèmes qui nous occupent.

Soit $f(z)$ méromorphe dans $|z| \leqslant R$ et $\sum_{n=0}^{\infty} a_n z^n$ son développement à l'origine. Supposons que f possède des pôles aux points $z_i = \lambda_i^{-1}$ ($i = 1, 2, \ldots, N$; un pôle de multiplicité q apparaissant q fois dans la suite). Assurons-nous du rangement suivant :

$$0 < |z_1| \leqslant |z_2| \leqslant \ldots\ldots \leqslant |z_N| < R \qquad (17)$$

Ajoutons maintenant l'hypothèse supplémentaire que z_k soit le seul pôle sur le cercle $(0, |z_k|)$, ce qui se traduit dans (17) par

$$|z_{k-1}| < |z_k| < |z_{k+1}| \qquad (18)$$

Alors, selon le lemme 2

$$\frac{H_{k-1}^{(n)}}{H_{k-1}^{(n+1)}} \xrightarrow[n\to\infty]{} \frac{1}{\lambda_1 \times \ldots \times \lambda_{k-1}}$$

et

$$\frac{H_{k}^{(n+1)}}{H_{k}^{(n)}} \xrightarrow[n\to\infty]{} \lambda_1 \times \ldots \times \lambda_{k-1}\ \lambda_k \ .$$

Par conséquent, si on pose

$$q_k^{(n)} = \frac{H_k^{(n+1)}\ H_{k-1}^{(n)}}{H_k^{(n)}\ H_{k-1}^{(n+1)}} \ , \qquad (19)$$

on a

$$q_k^{(n)} \xrightarrow[n\to\infty]{} \lambda_k \qquad (20)$$

Remarque :

Si on a $0 < |z_1| < |z_2|$, le lemme 2 s'applique au calcul de λ_1 et donne

$$\frac{H_1^{(n+1)}}{H_1^{(n)}} \xrightarrow[n\to\infty]{} \lambda_1 \ .$$

Comme d'autre part on a posé $H_0^{(n)} = 1$ quel que soit n, on peut définir $q_1^{(n)}$ selon (19) :

$$q_1^{(n)} = \frac{H_1^{(n+1)}\ H_0^{(n)}}{H_1^{(n)}\quad H_0^{(n+1)}} \ .$$

Il est facile alors de voir que si on pose $z_0 = 0$ $q_1^{(n)}$ vérifie (20) moyennant l'hypothèse (18) . Comme $H_1^{(i)} = a_i$, nous en déduisons que :

(21)
$$\begin{cases} q_1^{(n)} = \dfrac{a_{n+1}}{a_n} \\ q_1^{(n)} \xrightarrow[n\to\infty]{} \lambda_1 \end{cases}$$

Nous pouvons résumer tous les résultats précédents en un théorème.

Théorème 1 :

Soit la fonction f(z) méromorphe dans $|z| \leqslant R$, dont $\sum_{n=0}^{\infty} a_n z^n$ est le développement à l'origine. Soient $z_i = \lambda_i^{-1}$ ses pôles ($i = 1, 2, \ldots, N$; ou $i = 1, 2, \ldots$) que nous supposons énumérés en nombre égal à leur ordre de multiplicité et classés selon les modules croissants. Soit $z_0 = 0$ et dans le cas d'un nombre fini de pôles $z_{N+1} = R$. Pour tout z_k tel que $|z_{k-1}| < |z_k| < |z_{k+1}|$ (avec éventuellement $k \leqslant N$) , le coefficient $q_k^{(n)}$ est défini au moins à partir d'un certain rang et :

$$q_k^{(n)} \xrightarrow[n \to \infty]{} \lambda_k$$

Les quotients $q_k^{(n)}$ forment une partie des colonnes ($k = 1, 2, \ldots$) du schéma de l'algorithme Q D. A chaque valeur de k correspond une colonne. Nous allons voir que les déterminants $H_k^{(n)}$ n'interviendront plus directement si nous introduisons les quantités auxilliaires suivantes :

(22)
$$\begin{cases} 1_0^{(n)} = 0 \\ 1_k^{(n)} = \dfrac{H_{k+1}^{(n)} \; H_{k-1}^{(n+1)}}{H_k^{(n)} \; H_k^{(n+1)}} \end{cases} \qquad k = 1, 2, \ldots$$

Remarque :

On peut définir $1_k^{(n)}$ et $q_k^{(n)}$ pour n négatif, sous réserve que les dénominateurs soient non nuls, car nous avons vu qu'on pouvait définir $H_k^{(n)}$ pour n négatif. Nous nous servirons plus loin de ce fait quand nous voudrons faire "démarrer" l'algorithme Q D, relatif à la recherche des racines d'un polynôme, dans le sens haut $\longrightarrow$ bas.

Utilisant les définitions (19) et (22), le lemme 1 et beaucoup de perspicacité, Rutishauser a découvert les relations suivantes que nous énoncerons sous forme de théorème.

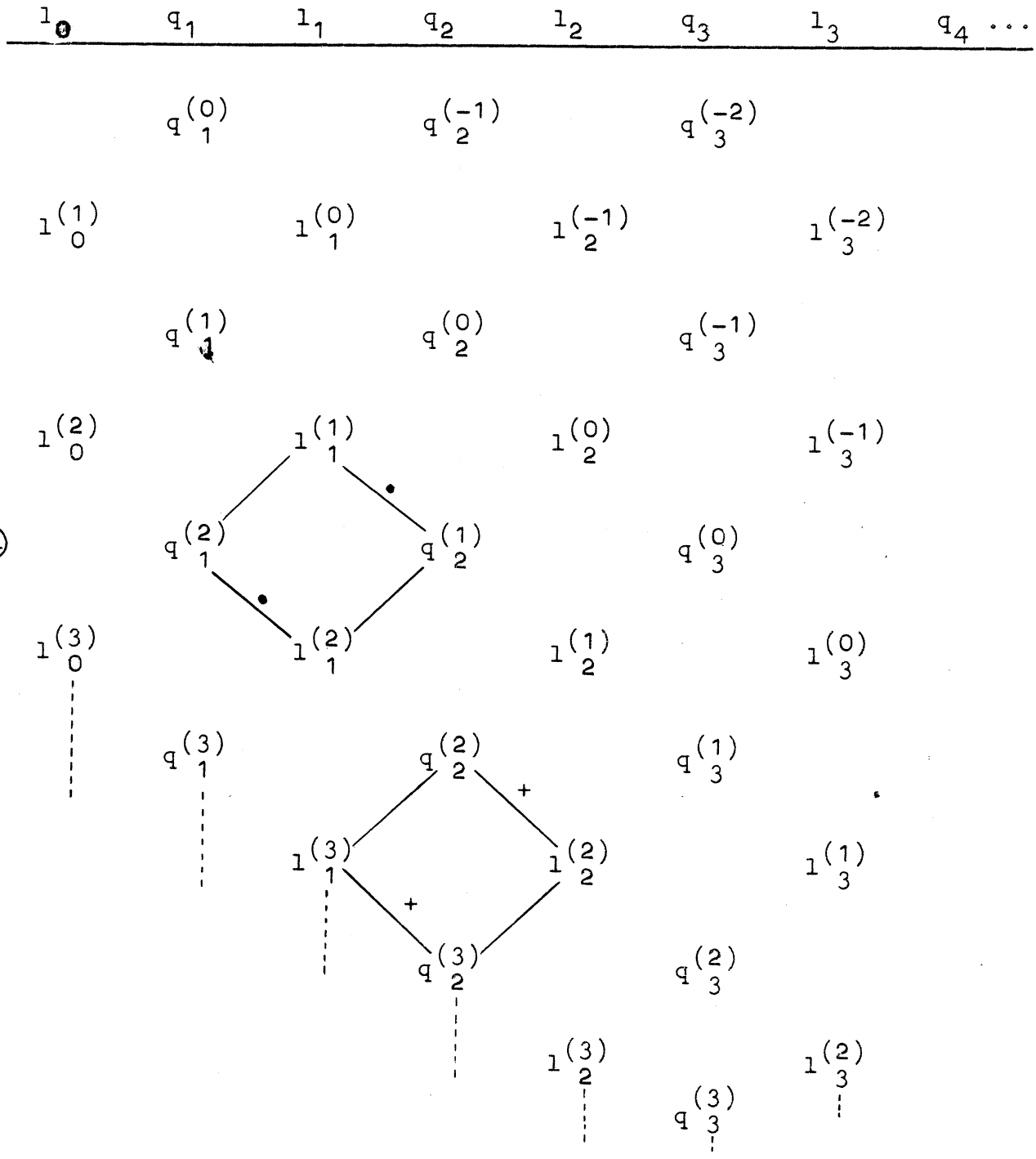
Théorème 2 :

Pour toutes valeurs des indices pour lesquelles les quantités ci-dessous sont définies, on a les relations :

(23)
$$\begin{cases} q_{k+1}^{(n)} \times 1_k^{(n)} = q_k^{(n+1)} \times 1_k^{(n+1)} \\ q_k^{(n)} + 1_k^{(n)} = q_k^{(n+1)} + 1_{k-1}^{(n+1)} \end{cases}$$

La première relation est une conséquence des définitions (19) et (22) ; la seconde se vérifie en employant deux fois l'identité du lemme 1 (démonstration en [5] p.31). Elles se "voient" très bien si on construit le schéma suivant, où les quantités mises en jeu sont, bien entendu, supposées définies :

(24)



La première relation (23) énonce qu'un losange centré sur une colonne 1 est telle que le produit des éléments les plus hauts et à droite est égal au produit des éléments les plus bas à gauche. La seconde énonce qu'on a une relation semblable avec les sommes pour un losange centré sur une colonne q.

Il est clair qu'on peut faire évoluer le schéma, comme dans la méthode d'Aitken, soit de gauche à droite (en remarquant que $1_0^{(n)} = 0$, $n = 1, 2 \ldots$; $q_1^{(n)} = \frac{a_{n+1}}{a_n}$, $n=0, 1, \ldots$), soit de haut en bas, mais alors se pose le problème de faire "démarrer" le schéma. La première façon de procéder semble par expérience numériquement instable. Ceci se confirme de la façon suivante : d'après le lemme 2, $1_k^{(n)} = O((\frac{\lambda_{k+1}}{\lambda_k})^n)$; donc chaque colonne q est obtenue, à partir de la précédente par multiplication par des quotients de quantités petites (exemple $q_2^{(1)} = q_1^{(2)} \times \frac{1_1^{(2)}}{1_1^{(1)}}$), ce qui est mauvais numériquement. Au contraire, le deuxième processus semble numériquement stable.

V - APPLICATION DE L'ALGORITHME Q D A LA DETERMINATION DES ZEROS D'UN POLYNOME.

Citons un théorème qui sera utile pour la construction du schéma dans ce cas

Théorème 3 :

Si f(z) est une fraction rationnelle de la forme

$$f(z) = \frac{b_0 z^p + b_1 z^{p-1} + \ldots + b_p}{c_0 z^m + c_1 z^{m-1} + \ldots + c_m} \quad (c_0 \neq 0 \; c_m \neq 0)$$

et si les 2m premières colonnes du schéma QD existent, alors

$$1_m^{(n)} = 0 \quad \text{pour} \quad n \geqslant p+1-m$$

([5] p. 32)

Nous avons vu que les relations (23) sont valables dès que les quantités mises en jeu sont définies. En particulier, elles sont valables pour certaines valeurs négatives de n. Nous nous servirons ici de ce fait pour déterminer, non plus les deux premières diagonales, mais les deux premières lignes du schéma.

Soit le polynôme :

$$P(z) = c_m + c_{m-1} z + \ldots + c_0 z^m$$

Nous supposerons $c_i \neq 0 \quad (i = 0, 1, \ldots, m)$;

Il est clair que si l'on pose

$$f(z) = \frac{1}{P(z)} ,$$

et si $\sum_{n=0}^{\infty} a_n z^n$ est le développement à l'origine de $f(z)$, les zéros de $P(z)$ peuvent être déterminés comme les pôles de la fraction rationnelle $f(z)$ à l'aide de l'algorithme Q D .

Une identité dûe à Wronski nous donne :

(25) $$\frac{1}{f(z)} = \sum_{k=0}^{\infty} \frac{(-1)^{k(k+1)/2}}{a_0^{k+1}} \; H_k^{(-k+2)} \; z^k \quad .$$

Or $$\frac{1}{f(z)} = P(z) = \sum_{k=0}^{m} c_{m-k} \; z^k \quad ;$$

En identifiant les coefficients, nous obtenons :

(26) $$\begin{cases} H_k^{(-k+2)} = (-1)^{k(k+1)/2} \; a_0^{k+1} \; c_{m-k} & \text{pour } k=0,\ldots,m \\ H_k^{(-k+2)} = 0 & \text{pour } k>m \end{cases}$$

D'autre part, pour $k \geqslant 1$:

- tous les déterminants $H_k^{(-k)}$ sont nuls car leur première ligne $a_{-k}, \ldots, a_{-1}$ est une ligne nulle.
- nous avons $H_k^{(-k+1)} = (-1)^{k(k-1)/2} \; a_0^k$ car les $H_k^{(-k+1)}$ sont du type :

$$\begin{vmatrix} 0 & & 0 & a_0 \\ 0 & & a_0 & a_1 \\ \vdots & & \ddots & \vdots \\ 0 & a_0 & & \\ a_0 & a_1 & \cdots\cdots & a_{k-1} \end{vmatrix}$$

Notons en outre que quel que soit n $H_0^{(n)} = 1$ et que $a_0 = \frac{1}{c_m}$

Résumons ceci :

(27)
$$\begin{cases} H_0^{(n)} = 1 & \text{pour tout } n \\ H_k^{(-k)} = 0 & \text{pour } k \geqslant 1 \\ H_k^{(-k+1)} = (-1)^{k(k-1)/2}\, a_0^k & \text{pour } k \geqslant 0 \\ a_0 = \dfrac{1}{c_m} \end{cases}$$

Calculons $q_k^{(-k+1)}$ pour $1 \leqslant k \leqslant m$

D'après (19) :

$$q_k^{(-k+1)} = \frac{H_k^{(-k+2)}\, H_{k-1}^{(-k+1)}}{H_k^{(-k+1)}\, H_{k-1}^{(-k+2)}} = \frac{H_k^{(-k+2)}\, H_{k-1}^{(-(k-1))}}{H_k^{(-k+1)}\, H_{k-1}^{(-k+2)}} .$$

Pour $1 \leqslant k \leqslant m$ tous les déterminants utilisés sont définis et les dénominateurs non nuls.

D'après (27) :

- pour $k-1 \geqslant 1$, ou $k > 1$ $\quad H_{k-1}^{(-(k-1))} = 0$

- pour $k = 1$ $\begin{cases} H_{k-1}^{(-(k-1))} = H_{k-1}^{(-k+2)} = 1 \\ H_k^{(-k+1)} = a_0 \end{cases}$

D'après (26) , pour $k = 1$:

$$H_k^{(-k+2)} = -\, a_0^2\, c_{m-1} = -\, a_0\, \frac{c_{m-1}}{c_m} .$$

Donc :

(28)
$$\begin{cases} q_1^{(0)} = -\dfrac{c_{m-1}}{c_m} \\ q_k^{(-k+1)} = 0 \qquad \text{pour} \quad 2 \leqslant k \leqslant m \end{cases}$$

Les formules (26) et (27) permettent aussi de calculer $l_k^{(-k+1)}$ pour $1 \leqslant k \leqslant m$. On obtient en procédant de la même manière :

(29)
$$\begin{cases} l_k^{(-k+1)} = \dfrac{c_{m-k-1}}{c_{m-k}} & \text{pour } 1 \leqslant k \leqslant m-1 \\ l_m^{(-m+1)} = 0 \end{cases}$$

Les relations (28), (29) et le théorème 3 nous servent à remplir les deux premières lignes, les première et dernière colonnes du schéma (24), c'est-à-dire les termes écrits ci-dessous. Nous pourrons, à partir de là, faire progresser le schéma de haut en bas à l'aide des règles des losanges de Rutishauser.

valeurs de départ du schéma :

l_0	q_1	l_1	q_2	l_2	l_{m-1}	q_m	l_m
	$-\frac{c_{m-1}}{c_m}$		0			0	
0		$\frac{c_{m-2}}{c_{m-1}}$		$\frac{c_{m-3}}{c_{m-2}}$	$\frac{c_0}{c_1}$		0
0							0
0							0
⋮							⋮

Le théorème 1 du § IV nous permet alors d'énoncer :

Théorème 4 :

Soit le polynôme

$$P(z) = c_0 z^m + c_1 z^{m-1} + \ldots + c_{m-1} z + c_m ,$$

dont les coefficients sont tous non nuls.
Supposons que le schéma QD existe avec les valeurs de départ précédentes.
Supposons les zéros de $P(z)$ rangés de la façon suivante :

$$0 < |z_1| \leqslant |z_2| \leqslant \ldots \leqslant |z_m|$$

Posons $z_0 = 0$ et $z_{m+1} = \infty$.
Alors, pour tout k tel que $|z_{k-1}| < |z_k| < |z_{k+1}|$ et $1 \leqslant k \leqslant m$, on a :

$$\lim_{n \to \infty} q_k^{(n)} = z_k^{-1} .$$

CHAPITRE III

LA METHODE DES REDUITES COMPAREE AVEC :
- LA PREMIERE COLONNE DE L'ALGORITHME QUOTIENT-DIFFERENCE
- LA METHODE DE BERNOULLI

I - COMPARAISON AVEC LA METHODE DE BERNOULLI.

1. Rappel de la méthode de Daniel Bernoulli

C'est une très ancienne méthode (1728) - ([4] p. 193 et [5] p. 28) - pour obtenir la racine de plus grand module d'une équation algébrique.

Soit l'équation

$$\textcircled{1} \qquad c_0 x^m + c_1 x^{m-1} + \dots\dots\dots + c_{m-1} x + c_m = 0 \quad (c_0 \neq 0),$$

supposée avoir une racine de plus grand module unique z_m.

Considérons l'équation aux différences suivantes (en la variable x_n) :

$$\textcircled{2} \qquad c_0 x_n + c_1 x_{n-1} + \dots\dots\dots + c_{m-1} x_{n-m+1} + c_m x_{n-m} = 0.$$

Cette équation donne , par récurrence, la valeur de x_n quand on connait les m valeurs précédentes $x_{n-1}, \dots, x_{n-m}$. Il est donc nécessaire d'avoir m valeurs arbitraires de départ $x_0, \dots, x_{m-1}$ pour que l'équation ② soit utilisable.

On montre que, dans ces conditions

$$\textcircled{3} \qquad \frac{x_n}{x_{n-1}} \xrightarrow[n\to\infty]{} z_m$$

Rappelons à titre indicatif que la théorie des équations aux différences linéaires donne pour solution générale de ② , dans le cas où toutes les racines sont distinctes

$$④ \qquad x_n = \alpha_1 z_1^n + \alpha_2 z_2^n + \ldots\ldots + \alpha_m z_m^n ,$$

α_1, α_2 , ... , α_m étant des constantes arbitraires et z_1, z_2 , ... , z_m les racines de l'équation ① . Dans le cas de racines multiples (nous supposerons z_1 d'ordre 3 pour fixer les idées) la solution générale de ② est alors

$$⑤ \qquad x_n = (\alpha_1 + n\,\alpha_2 + n^2\,\alpha_3) z_1^n + \alpha_4 z_4^n + \ldots + \alpha_m z_m^n$$

La méthode de Bernoulli consiste donc à choisir m valeurs arbitraires de départ x_0, x_1, ... , x_{m-1} , à construire la suite des itérés x_n et à former la suite des rapports $\frac{x_n}{x_{n-1}}$, suite qui tend vers z_m.

Il est alors évident que la méthode de Bernoulli permet aussi d'obtenir la racine de plus petit module z_1 de l'équation ① , toujours dans le cas d'une telle racine unique. En effet, considérons l'équation

$$⑥ \qquad c_m x^m + c_{m-1} x^{m-1} + \ldots\ldots\ldots + c_1 x + c_0 = 0 \quad (c_m \neq 0).$$

Elle admet comme racines les inverses des racines de l'équation ① . En particulier, la racine de plus grand module de ⑥ est l'inverse de la racine de plus petit module de ① . Si nous considérons alors l'équation aux différences

$$⑦ \qquad c_m x_n + c_{m-1} x_{n-1} + \ldots\ldots\ldots + c_1 x_{n-m+1} + c_0 x_{n-m} = 0,$$

nous voyons immédiatement que

$$\frac{x_n}{x_{n-1}} \xrightarrow[n\to\infty]{} \frac{1}{z_1} \quad ,$$

ou bien

⑧ $$\frac{x_{n-1}}{x_n} \xrightarrow[n\to\infty]{} z_1 \qquad (n = 1, 2 \ldots)$$

2. Comparaison Méthode des Reduites-Méthode de Bernoulli

La relation liant m+1 coefficients consécutifs dans le développement en série entière $\sum_{n=0}^{\infty} a_n x^n$ de

$$R(x) = \frac{1}{c_m + c_{m-1} x + \ldots + c_0 x^m}$$

est donnée par la formule ⑤ du chapitre I :

⑨ $$c_m a_n + c_{m-1} a_{n-1} + \ldots + c_1 a_{n-m+1} + c_0 a_{n-m} = 0 .$$

D'autre part, d'après la formule ⑧ du même chapitre, le terme de la suite tendant vers z_1 fournie par la méthode des réduites s'exprime par

⑩ $$\begin{cases} z_1^{(n)} = \dfrac{a_{n-1}}{a_n} \qquad (n = 1, 2, \ldots) \\ z_1^{(n)} \xrightarrow[n\to\infty]{} z_1 \end{cases}$$

En comparant respectivement les formules ⑦ et ⑨ .et les formules ⑧ et ⑩ , il apparait clairement que la méthode des réduites est une méthode de Bernoulli appliquée au calcul de la racine de plus petit module, avec cependant deux

différences :

- alors que dans la méthode de Bernoulli les m valeurs x_0 , ... , x_{m-1} de départ sont arbitraires, dans la méthode des réduites ces valeurs sont les m premiers coefficients a_0 , ... , a_{m-1} du développement en série entière de R(x), et sont donc des valeurs imposées.
- la méthode de Bernoulli, appliquée à la recherche de la racine de plus petit module, utilisant l'équation aux différences ⑦ calcule effectivement les différentes valeurs x_n et effectue les rapports $\frac{x_{n-1}}{x_n}$, tandis que l'algorithme que nous avons donné de la méthode des réduites ne calcule pas les différents coefficients a_n du développement en série de R(x), mais détermine directement les rapports $\frac{a_{n-1}}{a_n}$ en se servant des m-1 rapports précédents.

Malgré ces différences, nous devons nous attendre à une vitesse de convergence semblable pour les deux méthodes.

II - DEMONSTRATION DE LA CONVERGENCE DE LA METHODE DES REDUITES.

Nous donnerons ici une démonstration ne faisant pas, contrairement à celle du chapitre I, appel à la théorie des fonctions analytiques. Reprenant les notations de ce chapitre, nous voulons démontrer que

$$z_1 = - c_m \times \lim_{n\to\infty} \frac{\Delta_{n-1}}{\Delta_n} .$$

Nous utiliserons pour cela la formule ⑩ du même chapitre, donnant l'expression du $n^{ième}$ pivot dans la trian-

gularisation par la méthode de Gauss, ou plutôt la formule moins élaborée suivante :

$$(11)\qquad c_{m-1}^{(n)} = c_{m-1} + c_{m-2} \times \left(\frac{-c_m}{c_{m-1}^{(n-1)}}\right) + c_{m-3} \times \left(\frac{-c_m}{c_{m-1}^{(n-1)}}\right) \times \left(\frac{-c_m}{c_{m-1}^{(n-2)}}\right) + \ldots$$

$$+ c_0 \times \left(\frac{-c_m}{c_{m-1}^{(n-1)}}\right) \times \left(\frac{-c_m}{c_{m-1}^{(n-2)}}\right) \times \ldots \times \left(\frac{-c_m}{c_{m-1}^{(n-m+1)}}\right) .$$

Les produits

$$c_{m-1}^{(n)} \times c_{m-1}^{(n-1)} \times c_{m-1}^{(n-2)} \times \ldots\ldots \times c_{m-1}^{(n-m+1)} \times \Delta_{n-m}$$

$$c_{m-1}^{(n-1)} \times c_{m-1}^{(n-2)} \times \ldots\ldots \times c_{m-1}^{(n-m+1)} \times \Delta_{n-m}$$

$$c_{m-1}^{(n-2)} \times \ldots\ldots \times c_{m-1}^{(n-m+1)} \times \Delta_{n-m}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$c_{m-1}^{(n-m+1)} \times \Delta_{n-m}$$

étant respectivement égaux à Δ_n , Δ_{n-1}, Δ_{n-2}, ..., Δ_{n-m+1} , formons un cas particulier pour le premier produit

$$(12)\qquad \Delta_n = c_{m-1}^{(n)} \times \left[c_{m-1}^{(n-1)} \times c_{m-1}^{(n-2)} \times \ldots\ldots \times c_{m-1}^{(n-m+1)} \times \Delta_{n-m} \right]$$

En reportant la valeur (11) de $c_{m-1}^{(n)}$ dans (12) , on obtient, en tenant compte des produits précédents, l'expression de Δ_n suivante :

$$\Delta_n = c_{m-1}\Delta_{n-1} + c_{m-2}\,(-c_m)\Delta_{n-2} + c_{m-3}(-c_m)^2\Delta_{n-3} + \ldots + c_0(-c_m)^{m-1}\Delta_{n-m} ,$$

qu'on peut tranformer en multipilant ses deux membres par $-c_m$;

ce qui donne :

$$⑬ \quad c_m\Delta_n + c_{m-1}(-c_m)\Delta_{n-1} + c_{m-2}(-c_m)^2\Delta_{n-2} + c_{m-3}(-c_m)^3\Delta_{n-3} + \ldots$$
$$\ldots + c_0(-c_m)^m\Delta_{n-m} = 0$$

Cette relation de récurrence lie m+1 détermiants consécutifs dans la méthode des réduites (nous n'avons envisagé que le cas où $n > m$).
Posons :

$$\begin{aligned}
\Delta_n &= x_n \\
(-c_m)\,\Delta_{n-1} &= x_{n-1} \\
(-c_m)^2\Delta_{n-2} &= x_{n-2} \\
&\cdots\cdots\cdots\cdots \\
(-c_m)^m\Delta_{n-m} &= x_{n-m}
\end{aligned}$$

La relation ⑬ se transforme alors en l'équation aux différences ⑦ que nous venons de voir :

$$c_m\,x_n + c_{m-1}\,x_{n-1} + c_{m-2}\,x_{n-2} + \ldots + c_1 x_{n-m+1} + c_0\,x_{n-m} = 0 .$$

Tenant pour acquise la convergence de la méthode de Bernoulli, nous en déduisons que

$$\frac{x_{n-1}}{x_n} \xrightarrow[n\to\infty]{} z_1 ,$$

soit, en repassant à la terminologie initiale :

$$-c_m \frac{\Delta_{n-1}}{\Delta_n} \xrightarrow[n\to\infty]{} z_1$$

Cette démonstration de la convergence de la méthode des réduites prouve en même temps qu'on est en présence d'une méthode de Bernoulli (appliquée au calcul de la racine de plus petit module) dont les "valeurs de départ" sont fixées de la façon suivante :

(14)

$$
\begin{aligned}
x_0 &= (-c_m)^m \times \Delta_0 = (-c_m)^m \\
x_1 &= (-c_m)^{m-1} \times \Delta_1 = (-c_m)^{m-1}\, c_{m-1}^{(1)} \\
x_2 &= (-c_m)^{m-2} \times \Delta_2 = (-c_m)^{m-2}\, c_{m-1}^{(1)}\, c_{m-1}^{(2)} \\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
x_p &= (-c_m)^{m-p} \times \Delta_p = (-c_m)^{m-p}\, c_{m-1}^{(1)} \times \ldots \times c_{m-1}^{(p)} \\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
x_{m-1} &= (-c_m) \times \Delta_{m-1} = (-c_m)\, c_{m-1}^{(1)} \times \ldots \times c_{m-1}^{(m-1)}
\end{aligned}
$$

Remarque : Ces "valeurs de départ" sont définies à un coefficient multiplicatif près. Ainsi, au § I, nous avons montré que $a_0, \ldots a_{m-1}$ sont les "valeurs de départ". Nous avons pour $p = 0, 1, \ldots, m-1$

$$a_p = (-1)^p \frac{\Delta_p}{c_m^{p+1}}$$

$$x_p = (-1)^{m-p} (c_m)^{m-p} \Delta_p$$

$$\frac{x_p}{a_p} = (-1)^m \; c_m^{m+1}$$

III - IDENTITE DE LA PREMIERE COLONNE DE L'ALGORITHME QUOTIENT-DIFFERENCE ET DE LA SUITE DES INVERSES DES ELEMENTS DONNES PAR LA METHODE DES REDUITES.

Cette identité est immédiate à vérifier.

La suite $z_1^{(n)}$ de la méthode des réduites est telle que:

$$\begin{cases} z_1^{(n)} = \dfrac{a_n}{a_{n+1}} \quad (n = 0, 1, 2, \ldots) ; \\ z_1^{(n)} \xrightarrow[n\to\infty]{} z_1 \end{cases}$$

(pour uniformiser les notations avec celles qui ont été employées lors de l'exposé sur l'algorithme Quotient-Différence, nous avons fait commencer la suite à $z_1^{(0)}$).

D'autre part, $q_1^{(n)}$ la première colonne de l'algorithme Quotient-Différence, dont l'expression est donnée par la formule (21) du chapitre II, est telle que

$$\begin{cases} q_1^{(n)} = \dfrac{a_{n+1}}{a_n} \quad (n = 0, 1, 2, \ldots) \\ q_1^{(n)} \xrightarrow[n\to\infty]{} \dfrac{1}{z_1} \end{cases}$$

D'où notre résultat :

$$z_1^{(n)} = \frac{1}{q_1^{(n)}} \qquad (n = 0, 1, 2, \ldots)$$

La première colonne de l'algorithme Q D est formée des inverses des éléments de la suite de la méthode des réduites. La différence entre les deux réside dans le fait que la méthode des réduites n'utilise aucune colonne auxiliaire pour s'élaborer, contrairement à la première colonne du schéma Q D . On pourrait donc songer à utiliser la méthode des

réduites pour permettre de faire évoluer le schéma Q D de gauche à droite. Malheureusement, cette façon de le construire est moins stable que la méthode haut vers bas exposée au chapitre II, l'expérience, comme nous l'avons dit, le confirmant nettement.

Pour terminer le parallèle, il n'est pas inutile de noter que cette première colonne est aussi une méthode de Bernoulli appliquée au calcul de l'inverse de la plus petite racine, c'est-à-dire pratiquée sur l'équation dont l'ordre des coefficients est inversé.

CHAPITRE IV

RESULTATS NUMERIQUES

Ce chapitre est consacré à la partie pratique de ce travail. Nous donnons une procédure calculant les NB premières racines réelles traitables par la méthode des réduites. Cette procédure ALGOL a reçu le nom de procedure REDUITE ITER. Suivrons diverses comparaisons et vérifications numériques.

I - PROCEDURE REDUITE ITER

Paramètres d'entrée

- les entiers M, NB et K

  M : degré de l'équation

  NB : nombre de racines qu'on veut trouver. Décrivons à ce sujet le fonctionnement de la procédure : elle est capable de calculer les NB racines réelles x_i (i=1,...,NB) telles que

$$|x_1| \lneq |x_2| \lneq \cdots\cdots \lneq |x_{NB}| \lneq |z_j| \quad (j = NB+1,\ldots,M).$$

Ces racines sont calculées successivement par ordre de module croissant et éliminées par déflation.

K : nombre d'itérations maximum.

- le tableau C dont l'indice varie de 0 à M et qui contient les coefficients de l'équation rangés suivant l'ordre des puissances décroissantes. C n'est pas altéré par l'éxécution de la procédure.

- le réel $\emptyset$, précision voulue sur les racines : la racine est considérée comme atteinte lorsque 2 itérés consécutifs sont à une distance inférieure à $\emptyset$. Sinon le Kième itéré est pris comme racine.

Paramètre de sortie

Un seul paramètre : le tableau RAC dont l'indice varie de 1 à NB, qui contient les racines par ordre de module croissant.

```
procedure REDUITE ITER (M,C,NB,K,RAC,∅) ;
valeur ∅,C,M; entier M,NB,K;reel ∅ ;
tableau RAC,C;
debut tableau R [0:K];
    entier N,I,J ; reel RA ;
    reel procedure REDUITE (D,N);
    tableau D; entier N;
    debut entier I,J,P; reel S;
        R[0]:= -D[N] / D[N-1] ;
        pour I := 1 pas 1 jusque K faire
        début Si I≤ N-1 alors
            debut S:= D[N-I-1]; P:= I fin
            sinon debut S:= D[0] ; P:= N-1 fin;
            pour J:= P pas -1 jusqua 1 faire
            S:= SxR[I-J]+ D[N-J] ;
```

```
                    R[I] := -D[N]/S;
                    Si ABS (R[I] - R[I-1]) < Ø alors
                    debut REDUITE := R[I] · allera SORT fin
                fin ;
                REDUITE:= R[K] ; SORT:
            fin REDUITE;

            pour I := 1 pas 1 jusqua NB faire
            debut RA := RAC[I]:= REDUITE (C,M);
                M := M-1;
                pour j:=1 pas 1 jusqua M faire
                C[J]:= C[J] + C[J-1] x RA
            fin

fin REDUITE ITER;
```

II - RESULTATS NUMERIQUES

Le but de ces essais numériques est de montrer sur des exemples que la méthode des Réduites et la méthode de Bernoulli appliquée au calcul de la racine de plus petit module et utilisant les valeurs de départ (14) du chapitre III à un coefficient multiplicatif près, produisent, aux différences d'algorithme et aux erreur de calcul près, la même suite de nombres. Nous nous proposons aussi de vérifier pareille identité entre la méthode des Réduites et les inverses de la première colonne de l'algorithme QD.

Nous prendrons 3 exemples :

essai 1 : le polynôme
$x^3 - 93\,x^2 + 2882x - 29760$
qui admet les racines exactes 30, 31, 32.

essai 2 : le polynôme
$x^5 - 50x^4 + 977x^3 - 9316x^2 + 43332x - 78624$
qui possède 6,8,9,13,14 comme racines.

essai 3 : le polynôme
$x^{10} - 55x^9 + 1320x^8 - 18150x^7 + 157773x^6 - 902055x^5$
$+ 3416930x^4 - 8409500x^3 + 12753576x^2 - 10628640x$
$+ 3628800$
dont les racines exactes sont 1,2,3,4,5,6,7,8,9,10

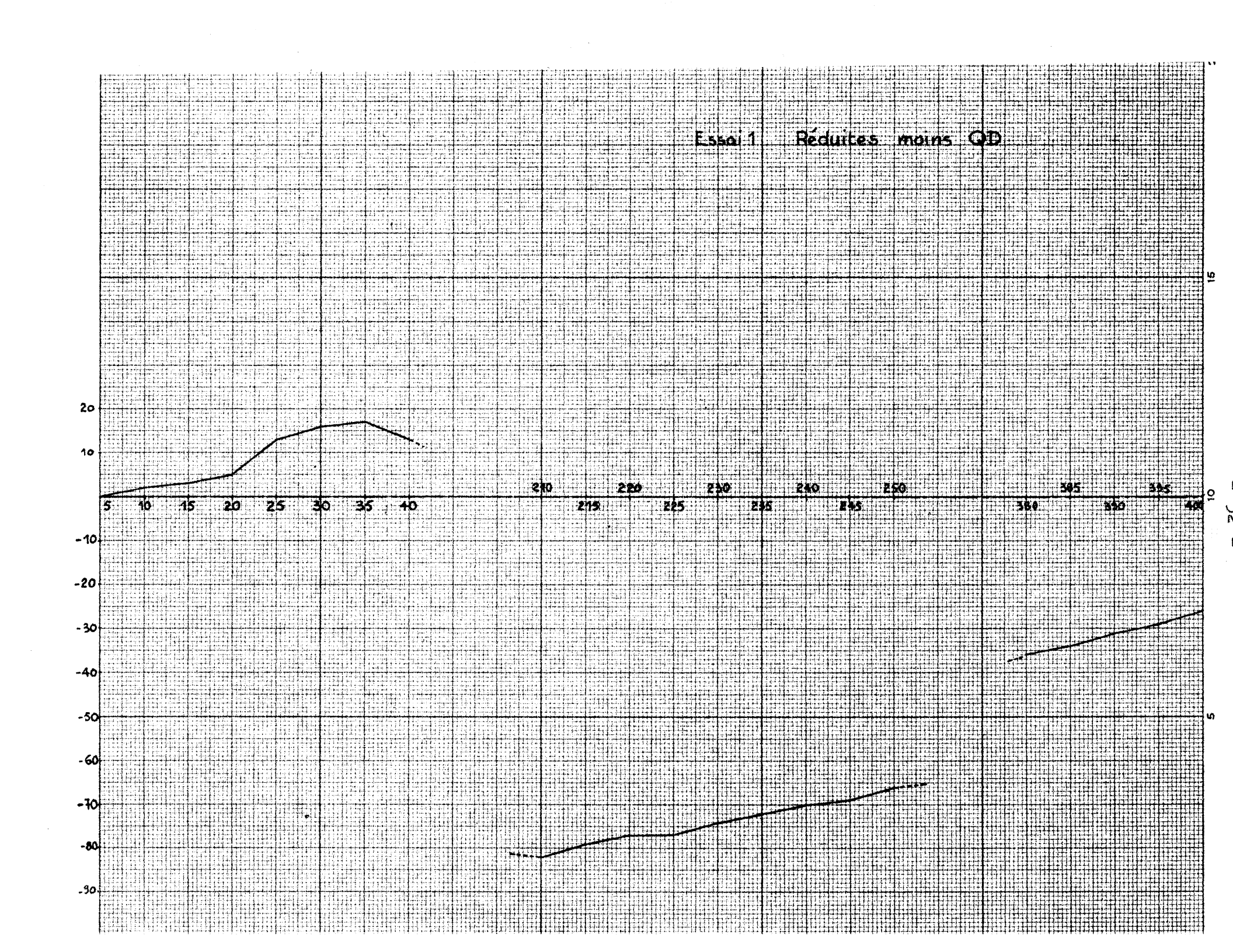
Nous donnons les résultats sur des graphiques. Nous avons porté en abscisse les itérations successives et en ordonnée (avec 10^{-8} comme unité)

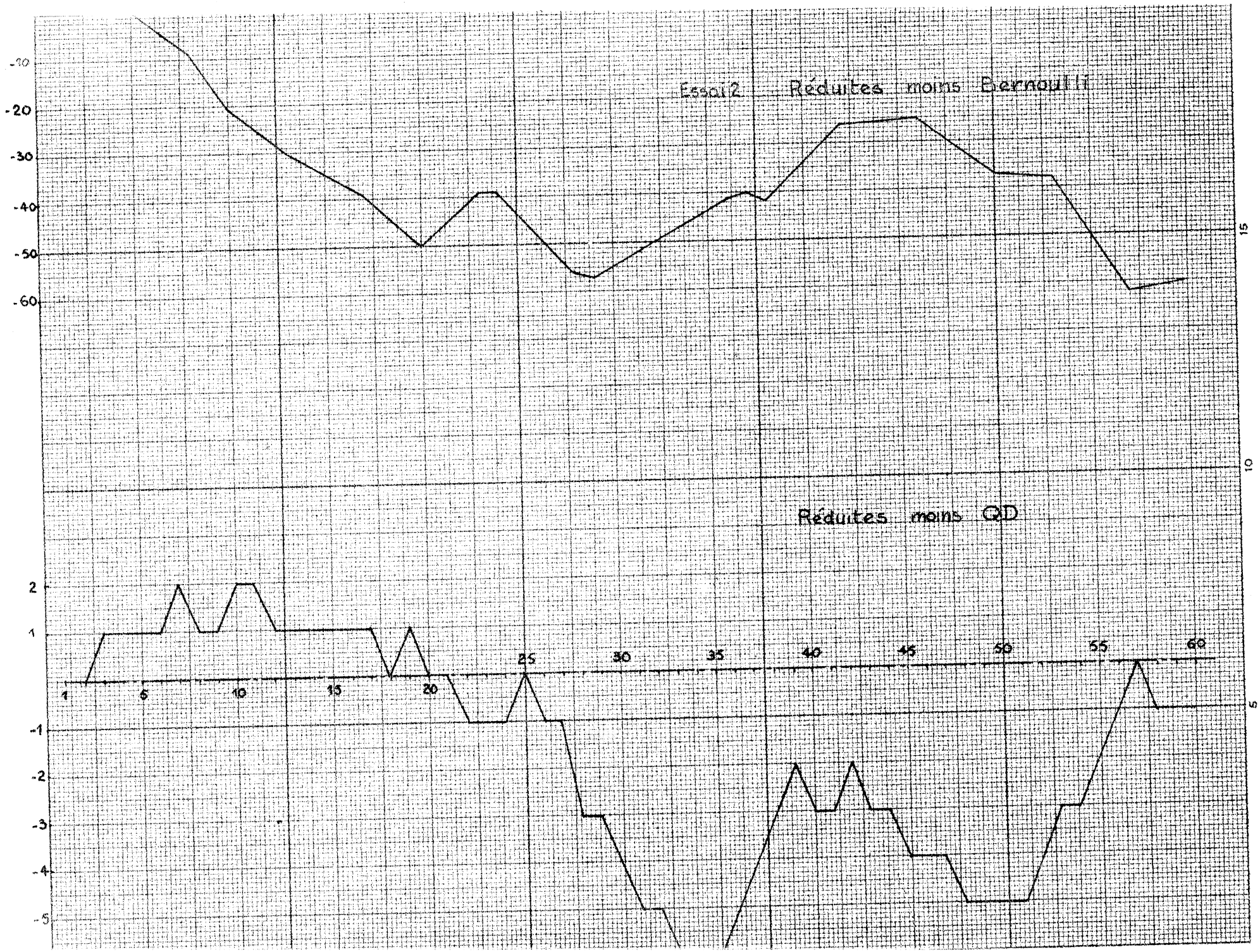
- sous le titre "Réduites moins Bernoulli" les différences, pour chaque itéré, entre le terme donné par la méthode des Réduites et le terme produit par la méthode de Bernoulli.

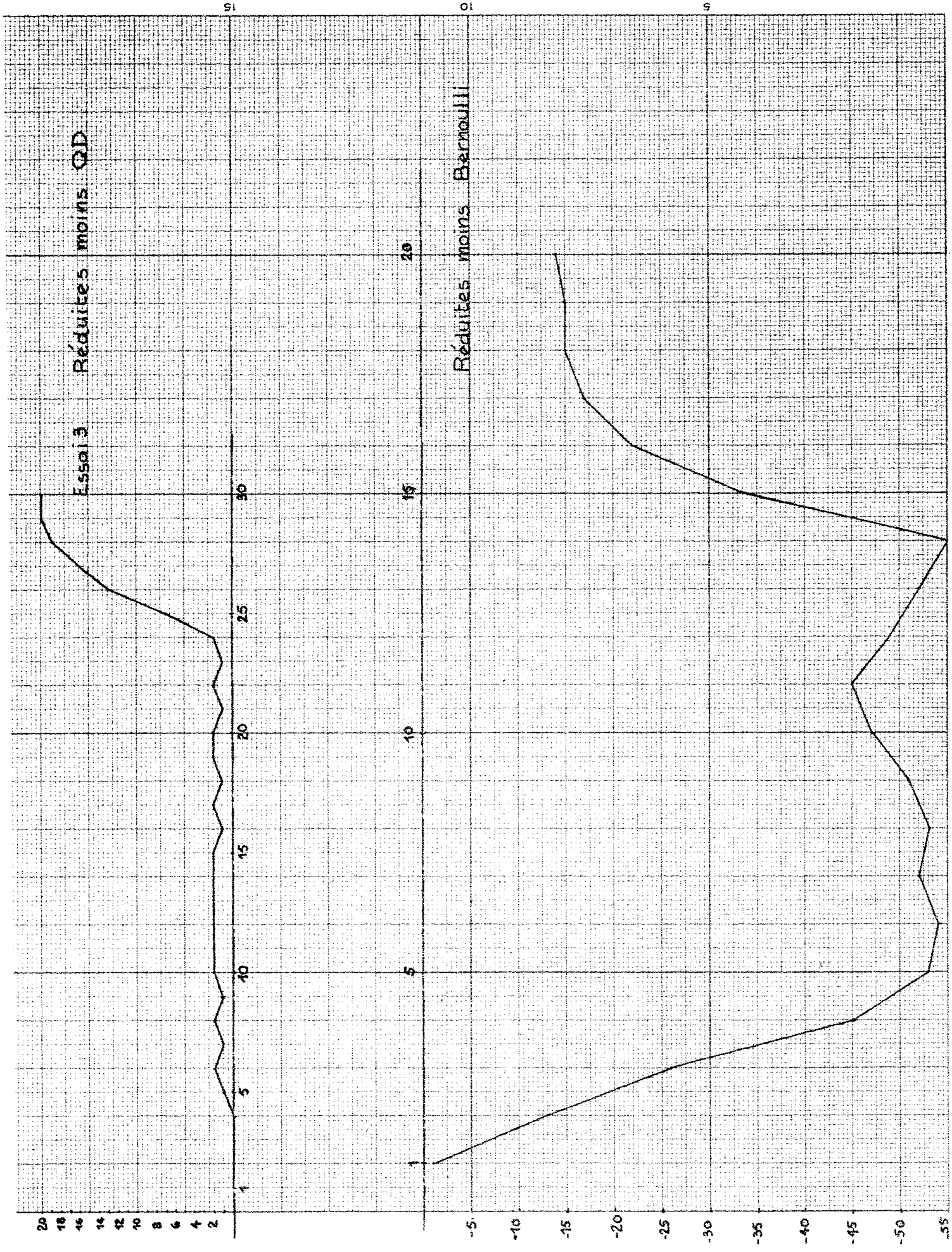
- sous le titre "Réduites moins QD" ces mêmes différences pour la méthode des Réduites et les inverses de la première colonne QD.

Ces essais ont été éffectués pour la plus petite des racines; la méthode des Réduites se stabilise respectivement pour ces 3 essais vers 29,999764, 5,9999983 et 0,99999990 tandis que la méthode de Bernoulli oscille entre 30,000060 et 30,000080, 6,0000020 et 6,0000045, 1,0000000 et 1,0000004.

Essai 1 Réduites moins Bernoulli
15
10
5
-50
-100
-250
-300
-350


Essai 1 Réduites moins QD
20
10
-10
-20
-30
-40
-50
-60
-70
-80
-90
5 10 15 20 25 30 35 40
210 215 220 225 230 235 240 245 250
355 360 365 370
15
10
5


Essai 2
Réduites moins Bernoulli
Réduites moins QD
-10
-20
-30
-40
-50
-60
15
10
5
2
1
-1
-2
-3
-4
-5
1
5
10
15
20
25
30
35
40
45
50
55
60


Essai 3 Réduites moins QD
Réduites moins Bernoulli

III - CONCLUSION

Les exemples que nous venons de voir confirment nos prévisions. Nous pourrons en particulier, utiliser la méthode des réduites dans tous les cas justifiant l'emploi de la méthode de Bernoulli avec en plus les avantages suivants :

1°) La méthode de Bernoulli passe par le calcul de x_n (③ Chap. III)

$$\frac{x_{n-1}}{x_n} = z_1^{(n)} \xrightarrow[n\to\infty]{} z_1$$

Donc, pour un rang n suffisant, $|x_n|$ est voisin de $|x_{n-1}/z_1|$, ce qui, dans le cas où z_1 est, en module, très petit devant 1, conduit très rapidement à des x_n grands et dans le cas où le module de z_1 est grand par rapport à 1, conduit non moins rapidement à des x_n petits. Au contraire (formule (10) chap. I), la méthode des Réduites définit $z_1^{(n)}$ par

$$z_1^{(n)} = \frac{-c_m}{S^{(n)}} \xrightarrow[n\to\infty]{} z_1,$$

l'algorithme passant par le calcul des $S^{(n)}$ successifs qui tendent à s'aligner en grandeur sur $|c_m/z_1|$. Il apparait clairement qu'on évite par la méthode des Réduites beaucoup de dépassements ou de soupassements de capacité qui se produisent en employant la méthode de Bernoulli. L'expérience nous l'a confirmé : dans cette dernière méthode, nous sommes très souvent obligés de procéder à une normalisation des termes x_n, ainsi dans l'essai 1 (tendance à soupasser la capacité) nous avons été obligés, pour mener à bien l'expérience de procéder à 19 normalisations qui consistaient à multiplier les m termes x_i les plus "récents" par un même nombre grand.

2°) Si nous faisons un décompte des opérations à effectuer dans chacune des deux méthodes, nous trouvons un avantage à employer la méthode des Réduites, même si l'on ne tient pas compte des opérations de normalisation éventuelles. Nous avons la disposition suivante :

Méthode des Réduites

$z_1^{(1)}$
$z_1^{(2)}$
.
.
.
$z_1^{(m-1)}$
$z_1^{(m)}$
$z_1^{(m+1)}$
.
.
.
$z_1^{(m+k-1)}$

Méthode de Bernoulli

$z_1^{(m)}$
$z_1^{(m+1)}$
.
.
.
$z_1^{(m+k-1)}$

k itérés

Méthode de Bernoulli

Les k itérations exigent le calcul de x_m, $x_{m+1}, \ldots, x_{m+k-1}$, c'est-à-dire k calculs du type

$$x_i = - \frac{c_{m-1}\, x_{i-1} + c_{m-2}\, x_{i-2} + \ldots + c_0\, x_{i-m}}{c_m}$$

qui nécessitent chacun

$$\begin{cases} 1 \text{ division} \\ m \text{ multiplications} \\ m \text{ additions} \end{cases}$$

Pour avoir $z_1^{(i)} = \frac{x_{i-1}}{x_i}$ il faut 1 division. La méthode de Bernoulli pour calculer 1 racine, coûte en définitive

$$\begin{cases} 2k \text{ divisions} \\ mk \text{ multiplications} \\ mk \text{ additions} \end{cases}$$

Méthode des Réduites

Les k itérations exigent k calculs du type :

$$z_1^{(i)} = \frac{c_m}{(((\ c_0 \times z_1^{(i-m+1)} + c_2) \times z_1^{i-m+2} + c_2) \times \ldots + c_{m-2}) \times z_1^{i-1} + c_m}$$

c'est-à-dire

$$\begin{cases} k \text{ divisions} \\ (m-1)k \text{ multiplications} \\ (m-1)k \text{ additions} \end{cases}$$

le calcul des m-1 valeurs $z_1^{(1)}\ z_1^{(2)}, \ldots, z_1^{(m-1)}$ demande en tout

$$\begin{cases} (m-1) \text{ divisions} \\ \frac{(m-2)\ (m-1)}{2} \text{ multiplications} \\ \frac{(m-2)\ (m-1)}{2} \text{ additions} \end{cases}$$

soit, pour le calcul d'une racine par la méthode des Réduites

$$\begin{cases} k+m-1 \text{ divisions} \\ (m-1)k + \dfrac{(m-2)\ (m-1)}{2} \text{ additions} \\ (m-1)k + \dfrac{(m-2)\ (m-1)}{2} \text{ multiplications.} \end{cases}$$

exemple nous prendrons un polynôme de degré 5 est nous envisagerons successivement k = 10, k = 20 et k = 40

	méthode de Bernoulli	méthode des Réduites
k=10	20 divisions 50 multiplications 50 additions	14 divisions 46 multiplications 46 additions
k=20	40 100 100	24 86 86
k=40	80 200 200	44 166 166

Ceci est d'ailleurs confirmé par la résolution complète du polynôme de l'essai 3 et par l'examen des temps de calcul. Ce polynôme a des racines particulièrement mal conditionnées ([4] p.144, [9]p 38). Nous avons pris $\emptyset = 2\ 10^{-8}$ et K, le nombre maximum d'itérations = 300 pour tester les 2 méthodes. Nous avons obtenu :

méthode des Réduites	méthode de Bernoulli
0,99999990	1,0000004
1,9999958	2,0000007
3,0001007	3,0000040
3,9992920	3,9996184
5,0019882	5,0029857
5,9979888	5,9894843
6,9992850	7,0213013
8,0027413	7,9758983
8,9984001	9,0144225
10,000206	9,9962832
temps de calcul : 1,8 secondes	temps de calcul : 3 secondes

Si les 4 premières racines sont meilleures avec la méthode de Bernoulli, par contre la méthode des Réduites semblent plus stable et donne une précision supérieure pour les racines suivantes.

Notons enfin que, la procédure REDUITE ITER concerne les équations à coefficients réels, mais que rien ne s'oppose, ni au point de vue théorique, ni au point de vue programmation à ce que la méthode des Réduites produise, dans les mêmes conditions, les NB racines de plus petit module d'une équation à coefficients complexes.

DEUXIEME PARTIE

-:-:-:-

ETUDE DE L'EMPLOI DE L'ALGORITHME DE ROUTH POUR RESOUDRE LES EQUATIONS ALGEBRIQUES

-:-:-:-

INTRODUCTION

Le théorème (et l'algorithme) de Routh permettent de déterminer le nombre de racines d'un polynôme situées dans le demi-plan $Re(z) > 0$. Nous présentons dans ce travail une façon originale, et nous l'espérons rentable numériquement, d'employer cet algorithme à la résolution des équations algébriques.

Le chapitre I reprend l'exposé de l'auteur Soviétique F. R. Gantmacher sur l'algorithme [1]. Signalons que cet algorithme a déjà été employé dans une méthode de résolution des équations algébriques, en particulier par E. Aparo de l'Institut de Calcul Numérique de Rome [2]. Le procédé de calcul utilisé ici débute comme celui qu'a employé Aparo, mais s'en distingue ensuite dans la façon d'affiner la recherche des parties réelles des racines. Il sera présenté au chapitre II, ainsi qu'un programme ALGOL de la méthode qui se limite au traitement des équations à coefficients réels.

CHAPITRE I

L'ALGORITHME DE ROUTH

I - INDICE DE CAUCHY.

Définition :

L'indice de Cauchy d'une fraction rationnelle R(x) entre les limites a et b (notation $I_a^b\ R(x)$, a et b étant des nombres réels ou $\pm\infty$) est la différence entre le nombre de sauts que fait R(x) depuis $-\infty$ jusqu'à $+\infty$ et le nombre de sauts depuis $+\infty$ jusqu'à $-\infty$ lorsque l'argument réel x varie de a à b (*)

Exemples:

1. $$R(x) = \sum_{i=1}^{p} \frac{A_i}{x-\alpha_i} + R_1(x)$$

où A_i et α_i (i = 1, 2, ..., p) sont des nombres réels et $R_1(x)$ une fraction rationnelle sans pôles réels ; on a alors:

$$I_{-\infty}^{+\infty}\ R(x) = \sum_{i=1}^{p} \text{signe } A_i \quad (**) \ ,$$

et en général :

$$I_a^b\ R(x) = \sum_{a<\alpha_i<b} \text{signe } A_i \quad (a < b)$$

2. Si $f(x) = a_0\ (x-\alpha_1)^{n_1}\ x \ldots x (x-\alpha_m)^{n_m}$ est un polynôme à coefficients réels et si parmi ses racines $\alpha_1 \ldots \alpha_m$, seulement les p premières sont réelles alors :

(*) en comptant le nombre de sauts, on ne tient pas compte des valeurs extrêmes R(a) et R(b) ; x varie dans l'ouvert]a, b[

(**) Signe A (où A est un nombre réel)=+1,-1,0 selon que A est > 0 , < 0 , = 0

$$\frac{f'(x)}{f(x)} = \sum_{j=1}^{m} \frac{n_j}{x-\alpha_j} = \sum_{i=1}^{p} \frac{n_i}{x-\alpha_i} + R_1(x) \quad ,$$

où $R_1(x)$ est une fraction rationnelle sans pôles réels ; doù :

<u>l'indice I_a^b $f'(x)/f(x)$ $(a < b)$ est égal au nombre de racines réelles distinctes de $f(x)$ dans l'intervalle $]a,b[$</u>

II - <u>THEOREME DE STURM</u>

Une méthode pour calculer l'indice I_a^b $R(x)$ repose sur le théorème de Sturm. Considérons la suite des polynômes à coefficients réels

$$f_1(x) \ , \ f_2(x) \ , \ldots \ , \ f_m(x) \quad ,$$

qui a les 2 propriétés suivantes dans l'intervalle $]a,b[$ (*) :

1. pour tout $x_0, x_0 \in]a,b[$, si $f_k(x_0)$ $(k = 2,\ldots, m-1)$ s'annule, les 2 polynômes adjacents f_{k-1} et f_{k+1} sont $\neq 0$ et de signes opposés pour cette valeur x_0 .

2. le dernier polynôme $f_m(x)$ ne s'annule pas dans $]a,b[$.

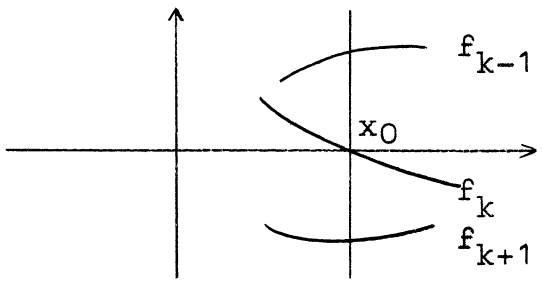
<u>Une telle suite de polynômes est appelée chaîne de Sturm simple dans l'intervalle $]a,b[$</u> .

Notons $V(x)$ la fonction égale au nombre de variations de signes dans la chaîne pour une valeur fixée de x. (**)

(*) a peut être $-\infty$ et b $+\infty$

(**) Si a est fini, alors $V(a)$ doit être interprété comme $V(a+\varepsilon)$ où ε est un nombre > 0 assez petit pour que dans l'intervalle $]a, a+\varepsilon]$ demi-fermé aucune des fonctions $f_i(x)$ ne s'annule. Définition semblable pour $V(b)$.

La valeur de V(x) , quand x varie dans]a,b[, ne peut changer que lorsqu'un des polynômes de la suite s'annule. Mais, d'après la propriété 1, quand un des polynômes f_k (pour k = 2,...,m-1) passe par 0, la valeur de V(x) ne change pas.



Seuls les passages de $f_1(x)$ par 0 font donc varier V(x)

V(x) augmente de 1 quand $\frac{f_2(x)}{f_1(x)}$ passe de $+\infty$ à $-\infty$

V(x) diminue de 1 quand $\frac{f_2(x)}{f_1(x)}$ passe de $-\infty$ à $+\infty$

d'où :

Théorème de Sturm :

Si $f_1(x)$, $f_2(x)$,..., $f_m(x)$ est une chaîne de Sturm dans l'intervalle]a,b[et si V(x) est le nombre de variations de signes de la chaîne en x, on a :

$$\textcircled{1} \qquad I_a^b \ \frac{f_2(x)}{f_1(x)} = V(a) - V(b)$$

Remarque :

Si on multiplie tous les polynômes d'une chaîne de Sturm par un même polynôme d(x) ayant ou non des racines réelles dans]a,b[, aucun membre de ① n'est altéré. On obtient une chaîne de Sturm généralisée et le théorème de Sturm

reste valable pour cette chaîne.

Construction d'une chaîne de Sturm à l'aide de l'algorithme d'Euclide :

Soient deux polynômes $f_1(x)$ et $f_2(x)$ tels que degré de $f_1 \geqslant$ degré de f_2 .
Posons :

- $f_3(x)$ = reste de la division de f_1 par f_2
- $f_4(x)$ = reste de la division de f_2 par f_3 ... etc

$$f_1(x) = q_1(x) \times f_2(x) - f_3(x)$$
$$f_2(x) = q_2(x) \times f_3(x) - f_4(x)$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$f_{k-1}(x) = q_{k-1}(x) \times f_k(x) - f_{k+1}(x)$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$f_{m-2}(x) = q_{m-2}(x) \times f_{m-1}(x) - f_m(x)$$
$$f_{m-1}(x) = q_{m-1}(x) \times f_m(x) - f_{m+1}(x)$$

C'est l'algorithme d'Euclide qui s'arrête à la rencontre d'un reste ($f_{m+1}(x)$) identiquement nul. $f_m(x)$ est le dernier reste non identiquement nul et c'est aussi le PGCD de tous les polynômes $f_1 \ldots f_m$ de la suite ainsi construite.

Deux cas se présentent :

1) $f_m(x)$ n'a pas de racines réelles dans]a,b[: $f_m(x)$ étant le PGCD de tous les polynômes de la suite, ces derniers

n'ont pas de racines réelles communes et $f_k(x_0) = 0$ entraîne $f_{k-1}(x_0) = -f_{k+1}(x_0)$ $(k = 2, \ldots, m-1)$
ou bien

$$f_{k-1}(x_0) \times f_{k+1}(x_0) \lneq 0$$

dans ce cas, la suite $f_1 \ldots . f_m$ est une chaîne de Sturm simple.

2) $f_m(x)$ a des racines réelles dans $]a,b[$:
Si alors on divise tous les polynômes de la suite par $f_m(x)$ on obtient une chaîne de Sturm généralisée.

Dans tous les cas, le théorème de Sturm (formule ①) s'applique à la suite construite à l'aide de l'algorithme d'Euclide.

Application directe :

Le nombre de racines réelles distinctes d'un polynôme étant égal à $I_{-\infty}^{+\infty} f'(x)/f(x)$, construisons à l'aide de l'algorithme d'Euclide une chaîne de Sturm à partir de f et f' .

On a : nombre de racines réelles distinctes = $V(-\infty) - V(+\infty)$ de la chaîne.

III - L'ALGORITHME DE ROUTH.

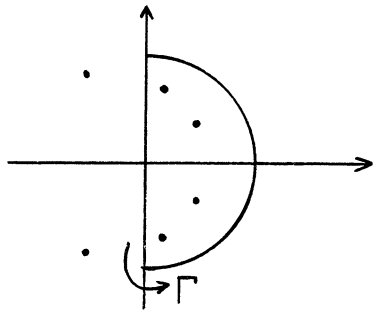
Formule de départ :

Le problème de Routh consiste à déterminer le nom-

bre k de racines d'un polynôme situées dans le demi-plan défini par $Re(z) > 0$. Deux possibilités sont à envisager :

1) Le polynôme f(z) n'a pas de racines imaginaires pures

Prenons le nombre R suffisamment grand pour que toutes les racines de f situées dans le demi-plan $Re(z)>0$ se trouvent dans le demi-cercle de rayon R centré à l'origine.



Considérons le domaine limité par Γ = demi-cercle + partie axe imaginaire. $f(z) = a_0 \prod_{i=1}^{n} (z-z_i)$; z décrivant le contour Γ une seule fois dans le sens positif, on a

$$\Delta \arg f(z) = \sum_{i=1}^{n} \Delta \arg (z-z_i) = 2 k \pi$$

car si z_i est à l'intérieur du domaine, $\Delta \arg (z-z_i) = 2\pi$, si z_i est à l'extérieur $\Delta \arg (z-z_i) = 0$. D'autre part, si on fait tendre R vers l'infini, z étant sur le demi-cercle, f(z) se comporte comme $a_0 z^n$, son monôme de degré le plus élevé. Donc, z décrivant tout le demi-cercle,

$$\Delta \arg f(z) = \Delta \arg a_0 z^n = n \pi .$$

D'où, lorsque f n'a pas de racines imaginaires pures :

$$\Delta_{-\infty}^{+\infty} \arg f(iw) = (n-2k)\,\pi$$

Posons $f(z) = a_0 z^n + b_0 z^{n-1} + a_1 z^{n-2} + b_1 z^{n-3} + \ldots \quad (a_0 \neq 0)$

On a

$$f(iw) = U(w) + i\,V(w) \quad \text{avec}$$

pour n pair $\begin{cases} U(w) = (-1)^{n/2}(a_0 w^n - a_1 w^{n-2} + a_2 w^{n-4} - \ldots) \\ V(w) = (-1)^{n/2-1}(b_0 w^{n-1} - b_1 w^{n-3} + b_2 w^{n-5} - \ldots) \end{cases}$

pour n impair $\begin{cases} U(w) = (-1)^{(n-1)/2}(b_0 w^{n-1} - b_1 w^{n-3} + b_2 w^{n-5} - \ldots) \\ V(w) = (-1)^{(n-1)/2}(a_0 w^n - a_1 w^{n-2} + a_2 w^{n-4} - \ldots) \end{cases}$

On a d'autre part :

$$\arg f(iw) = \text{arc tgte } \frac{V(w)}{U(w)}$$

et on vérifie facilement que

$$\frac{1}{\pi}\,\Delta_{-\infty}^{+\infty} \arg f(iw) = \begin{cases} I_{-\infty}^{+\infty} \dfrac{U(w)}{V(w)} & n \text{ impair} \\ -I_{-\infty}^{+\infty} \dfrac{V(w)}{U(w)} & n \text{ pair} \end{cases}$$

En combinant les trois groupes de formules précédents, on obtient la formule suivante valable quel que soit n, pourvu que f n'ait pas de racines imaginaires pures :

② $$I_{-\infty}^{+\infty} \frac{b_0 w^{n-1} - b_1 w^{n-3} + b_2 w^{n-5} - \ldots}{a_0 w^n - a_1 w^{n-2} + a_2 w^{n-4} - \ldots} = n - 2k$$

2) Le polynôme f(z) a des racines imaginaires pures.

Nous allons généraliser la formule ②. Supposons que f ait k racines dans le 1/2 plan Re(z) > 0 et s racines imaginaires pures. Posons

$$f(z) = d(z) \times f^{*}(z) \quad ,$$

où $d(z) = z^{s} + \ldots$ a s racines imaginaires pures et $f^{*}(z)$ n'en a pas.

$$f(iw) = U(w) + iV(w) = d(iw) \left[U^{*}(w) + iV^{*}(w) \right]$$

Supposons que s soit pair. d(z) est alors un produit de facteurs de la forme $(z - iw_1)(z+iw_1)$ ou de la forme z^2. Donc d(iw) prend des valeurs réelles.

Par suite $\begin{cases} U(w) = d(iw)\ U^{*}(w) \\ V(w) = d(iw)\ V^{*}(w) \end{cases}$

et

$$\frac{U(w)}{V(w)} = \frac{U^{*}(w)}{V^{*}(w)}$$

d'où, en posant

$$f_2(w) = b_0 w^{n-1} - b_1 w^{n-3} + b_2 w^{n-5} - \ldots$$
$$f_1(w) = a_0 w^{n} - a_1 w^{n-2} + a_2 w^{n-4} - \ldots \quad ,$$

f_1 et f_2 étant respectivement les polynômes dénominateurs

et numérateurs du 1er membre de la formule ② pour f, f_1^* et f_2^* étant ces mêmes polynômes pour f^* , et puisque n et n^* ont des parités égales :

$$\frac{f_2(w)}{f_1(w)} = \frac{f_2^*(w)}{f_1^*(w)}$$

On peut appliquer la formule ② à $f^*(z)$ qui n'a pas de racines imaginaires pures. Il vient :

$$I_{-\infty}^{+\infty} \frac{f_2(w)}{f_1(w)} = I_{-\infty}^{+\infty} \frac{f_2^*(w)}{f_1^*(w)} = n^* - 2k = n - s - 2k \ .$$

Donc, lorsque f(z) a s racines imaginaires pures :

$$③ \quad I_{-\infty}^{+\infty} \frac{b_0 w^{n-1} - b_1 w^{n-3} + b_2 w^{n-5} - \ldots.}{a_0 w^n - a_1 w^{n-2} + a_2 w^{n-4} - \ldots.} = n - 2k - s \ .$$

(étude semblable dans le cas s impair).

Application du théorème de Sturm dans le cas régulier :

Notre problème est de déterminer k. Pour cela, nous nous servirons du théorème de Sturm pour calculer la partie gauche des formules ② ou ③ . Construisons par le schéma d'Euclide la chaîne de Sturm généralisée (ou non) sur f_1 et f_2

$$f_1(w) \ , \ f_2(w) \ , \ \ldots \ , \ f_m(w)$$

Nous appellerons cas régulier le cas où m = n+1. Dans ce cas, nécessairement le degré de chaque polynôme de

la chaîne est inférieur de 1 au degré du précédent et $f_m(w)$ est de degré 0.

Le cas régulier conduit donc à une chaîne de Sturm simple. Posons :

$$f_3(w) = c_0\, w^{n-2} - c_1\, w^{n-4} + c_2\, w^{n-6} - \dots$$
$$f_4(w) = d_0\, w^{n-3} - d_1\, w^{n-5} + d_2\, w^{n-7} - \dots$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots .$$

En effectuant les divisions successives, il vient :

$$f_3(w) = \frac{a_0}{b_0}\, w\, f_2(w) - f_1(w)$$
$$f_4(w) = \frac{b_0}{c_0}\, w\, f_3(w) - f_2(w)$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots$$

et comme dans ce cas régulier, aucun des coefficients de tête b_0, c_0, d_0,... ne s'annule , ces coefficients et les suivants sont égaux à :

$$c_0 = a_1 - \frac{a_0}{b_0}\, b_1 \qquad c_1 = a_2 - \frac{a_0}{b_0}\, b_2 \dots$$
$$d_0 = b_1 - \frac{b_0}{c_0}\, c_1 \qquad d_1 = b_2 - \frac{b_0}{c_0}\, c_2 \dots$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots ,$$

ainsi de suite, les coefficients de $f_5 \dots f_{n+1}$ étant déterminés de la même manière.

Nous obtenons le tableau des coefficients de ces polynômes (au signe près) sous la forme du schéma de Routh :

$$\begin{array}{cccc} a_0 & a_1 & a_2 & a_3 \cdots \\ b_0 & b_1 & b_2 & b_3 \cdots \\ c_0 & c_1 & c_2 & c_3 \cdots \\ d_0 & d_1 & d_2 & d_3 \cdots \\ \cdots & \cdots & \cdots & \cdots \end{array}$$

à l'aide de la règle suivante :

Règle :

Une ligne du schéma de Routh est déterminée à partir des 2 précédentes en retranchant à la 1ère, la 2ème multipliée par le nombre qui annule la 1ère différence et on déplace la ligne obtenue de 1 coefficient vers la gauche.

On déduit du fait que $f_1(w)$ et $f_2(w)$ n'ont pas de racines communes, que $U(w)$ et $V(w)$ (qui en sont déduits par multiplication par des constantes) ne s'annulent pas simultanément pour w réel. Par suite, le polynôme donné $f(z)$ n'a pas, dans ce cas régulier, de racines imaginaires pures.

Ce cas régulier est donc inclus dans celui de la formule ② qui s'y applique.

Pour calculer le membre de gauche de ② , utilisons la théorème de Sturm. Il vient:

$$V(-\infty) - V(+\infty) = n - 2k$$

où V se calcule sur la chaîne de Sturm ordinaire construite à partir de f_1 et f_2 au moyen du schéma de Routh.

Soit :

$$f_k(w) = m_0\, w^{n-k+1} - m_1\, w^{n-k-1} + \ldots$$

un des polynômes de la chaîne.

Si $w \to +\infty$, le signe de f_k coïncide avec le signe de son coefficient de plus haut degré.

Si $w \to -\infty$, le signe de $f_k(w)$ est égal à signe $(m_0 \times (-1)^{n-k+1})$, donc

$$V(+\infty) = V(a_0,\ b_0,\ c_0 \ \ldots\ )$$

$$V(-\infty) = V((-1)^n\, a_0,\ (-1)^{n-1}\, b_0, \ldots, (-1)^2 l_0, (-1)^1 m_0, n_0)$$

et quelle que soit la parité de n

$$V(-\infty) = V(a_0,\ -b_0,\ c_0,\ -d_0\ ,\ \ldots)\ ,$$

et comme

$$V(a_0,\ b_0,\ c_0,\ \ldots) + V(a_0,\ -b_0,\ c_0,\ -d_0\ \ldots) = V\ \max$$

et que dans le cas régulier V max = n, on obtient la formule suivante, valable seulement dans ce cas

④ $$k = V(a_0,\ b_0,\ c_0,\ d_0,\ \ldots)$$

Théorème de Routh :

> Dans le cas où le schéma de Routh se détermine normalement , c'est-à-dire dans le cas où aucun élément de la première colonne ne s'annule, le nombre de racines d'un polynôme à coefficients réels, situées dans le demi-plan $Re(z) > 0$ est égal au nombre de variations de signes de cette première colonne.

Exemple :

Considérons le polynôme $z^5 + z^4 + 2z^3 + z^2 + 2z - 1$

On a :

$$U(w) = (-1)^2 f_2(w)$$

$$V(w) = (-1)^2 f_1(w)$$

avec

$$f_1(w) = w^5 - 2w^3 + 2w$$

$$f_2(w) = w^4 - w^2 - 1$$

Le schéma de Routh associé à la suite de Sturm sur f_1 et f_2 est le suivant :

w^5	1	2	2
w^4	1	1	-1
w^3	1	3	
w^2	-2	-1	
w	5/2		
w^0	-1		

Aucun élément de la 1ère colonne ne s'annule : on est dans le cas régulier.

$$V(1,\ 1,\ 1,\ -2,\ \frac{5}{2},\ -1) = 3$$

Ce polynôme a donc 3 racines à droite de l'axe imaginaire.

Cas où toutes les racines de f ont des parties réelles négatives.

La formule

$$I_{-\infty}^{+\infty} \frac{f_2}{f_1} = V(-\infty) - V(+\infty) = n - 2k$$

s'applique dans tous les cas où f n'a pas de racines imaginaires pures, donc ici. Comme $k = 0$, elle devient :

$$V(-\infty) - V(+\infty) = n$$

Mais

$$0 \leqslant V(-\infty) \leqslant m-1 \leqslant n$$

$$0 \leqslant V(+\infty) \leqslant m-1 \leqslant n$$

donc

$$V(-\infty)-V(+\infty) = n \text{ n'est possible que si } \begin{cases} V(-\infty)=n \\ V(+\infty)=0 \end{cases}$$

ce qui entraîne qu'on est ici nécessairement dans le cas régulier et réciproquement, si on est dans le cas régulier et si $V(+\infty) = 0$, alors $k = 0$ d'où :

Critère de Routh :

Toutes les racines d'un polynôme à coefficients réels f(z) ont des parties réelles négatives si, et seulement si, dans l'algorithme de Routh, tous les éléments de la 1ère colonne sont $\neq 0$ (cas régulier) et de même signe.

Remarque concernant le cas où f a des racines imaginaires pures.

$$f(iw) = U(w) + iV(w)$$

Toutes les racines imaginaires pures de f sont les racines réelles communes à U et V (à un coefficient multiplicatif i près) et réciproquement. Si f a des racines imaginaires pures, le PGCD de f_1 et f_2 a donc au moins comme racines les racines réelles correspondantes et il est de degré > 0 , au moins égal au nombre des racines imaginaires pures.

On est certainement alors conduit à un cas singulier.

La réciproque n'est pas vraie.

IV - LES CAS SINGULIERS DE L'ALGORITHME DE ROUTH.

Nous savons jusqu'à présent calculer le nombre de racines de f(z) situées dans le demi-plan $Re(z)>0$ dans le

cas régulier qui est lui-même inclus dans le cas où f n'a pas de racines imaginaires pures. Nous avons vu que le schéma de Routh est un moyen de calculer les restes $-f_3$, $-f_4$ ... des divisions de f_1 par f_2, f_2 par f_3 ... . Seulement ce schéma ne fonctionne qui si aucun des éléments de la 1ère colonne ne s'annule (donc si le degré des différents polynômes décroît régulièrement de 1 à la fois).

Les cas singuliers sont donc de deux sortes :

1) Un des éléments h_0 de la 1ère colonne s'annule, mais dans la ligne correspondante existent des éléments non nuls.

2) Toute une ligne du schéma s'annule à la fois.

Remarque :

Il peut évidemment se produire que dans la chaîne de Sturm existent successivement le 1er et le 2ème cas singuliers. La chaîne aura alors été construite avec un autre moyen que l'algorithme de Routh.

1° - Exemples de cas singuliers.

Exemple 1 :

$$f(z) = z^6 + z^5 + 3z^4 + 3z^3 + 3z^2 + 2z + 1$$

Commençons le schéma de Routh

1	3	3	1
1	3	2	
0	1	1	

Nous sommes arrêtés par un cas singulier du 1er type. En effectuant les divisions successives, nous obtenons :

$$f_1 = z^6 - 3z^4 + 3z^2 - 1$$
$$f_2 = z^5 - 3z^3 + 2z$$
$$f_3 = -z^2 + 1$$
$$f_4 = 0$$

Donc, bien que nous obtenions en fait un cas singulier du 1er type, f_1 et f_2 ont malgré tout un PGCD de degré>0 (degré 2). Ce PGCD a 2 racines réelles opposées ± 1, ce qui signifie que f a 2 racines imaginaires pures $\pm i$. Pour calculer k, la formule à employer est le formule ③ avec s=2 , $I_{-\infty}^{+\infty} \frac{f_2}{f_1} = 6 - 2k - 2$.

Et pour calculer $I_{-\infty}^{+\infty} \frac{f_2}{f_1}$, nous ne pouvons faire autrement que d'appliquer la formule générale ① $I_{-\infty}^{+\infty} \frac{f_2}{f_1} = V(-\infty) - V(+\infty)$ sur la chaîne de Sturm généralisée f_1, f_2, f_3

$$\left.\begin{array}{l} V(-\infty) = 1 \\ V(+\infty) = 1 \end{array}\right\} \text{ d'où } k = 2$$

Exemple 2 :

$$f(z) = z^{10} + z^9 - z^8 - 2z^7 + z^6 + 3z^5 + z^4 - 2z^3 - z^2 + z + 1$$

Construisons le schéma de Routh :

1	-1	1	1	-1	1
1	-2	3	-2	1	
1	-2	3	-2	1	
0	0	0	0		

Nous sommes arrêtés par un cas singulier du 2ème type, ce qui signifie que le PGCD est de degré > 0 et que f a comme racines imaginaires pures, les racines réelles du FGCD (s'il y en a) multipliées par i.

La chaîne de Sturm est :

$$f_1 = z^{10} + z^8 + z^6 - z^4 - z^2 - 1$$
$$f_2 = z^9 + 2z^7 + 3z^5 + 2z^3 + z$$
$$\text{PGCD} = f_3 = z^8 + 2z^6 + 3z^4 + 2z^2 + 1$$

Il est évident que le PGCD n'a pas de racines réelles. Donc f n'a pas de racines imaginaires pures. Cet exemple montre bien que la réciproque dont nous parlions à la fin du § précédent est fausse : un cas singulier du 2ème type n'entraîne pas que f ait des racines imaginaires pures.

Pour calculer k, nous pouvons employer la formule ② $I_{-\infty}^{+\infty} \frac{f_2}{f_1} = n - 2k = V(-\infty) - V(+\infty)$ sur la chaîne f_1, f_2, f_3 précédente

$$\left.\begin{array}{l} V(-\infty) = 2 \\ V(+\infty) = 0 \end{array}\right\} \Longrightarrow k = 4$$

Exemple 3 :

$$f(z) = z^4 + z^3 + 2z^2 + 2z + 1 \ .$$

Les premières lignes du schéma de Routh sont :

1	2	1
1	2	
0	1	

Nous sommes arrêtés par un cas singulier du 1er type.

En effectuant les divisions successives, nous obtenons :

$$f_1 = z^4 - 2z^2 + 1$$
$$f_2 = z^3 - 2z$$
$$f_3 = -1$$

Il manque le monôme de degré 2 dans f_3 .

Nous avons ici un cas singulier du 1er type et un PGCD de degré 0 . f n'a pas de racines imaginaires pures et nous pouvons employer la formule ② $I_{-\infty}^{+\infty}\frac{f_2}{f_1} = n - 2k$ mais pas ④ car nous ne sommes pas dans le cas régulier.

$n - 2k = V(-\infty) - V(+\infty)$ de la chaîne construite sur f_1 f_2

$$\left.\begin{array}{l} V(-\infty) = 1 \\ V(+\infty) = 1 \end{array}\right\} \Longrightarrow k = 2$$

Exemple 4 :

$$f(z) = z^4 + z^3 - z^2 + z - 2$$

Nous pouvons commencer le schéma de Routh.

1	-1	-2
1	1	
-2	-2	
0		

Nous sommes arrêtés par un cas singulier du 2ème type

$f_1 = z^4 + z^2 - 2$

$f_2 = z^3 - z$

$f_3 = -2z^2 + 2$ PGCD de degré 2 qui a 2 racines réelles ± 1 . f a donc 2 racines imaginaires pures $\pm i$. Nous emploierons la formule ③ $I_{-\infty}^{+\infty} \frac{f_2}{f_1} = n-2k-s$ avec s = 2 associée à la formule ① car le 1 cas est singulier.

pour la chaîne $f_1\ f_2\ f_3$: $V(-\infty) = 1$, $V(+\infty) = 1$ $\Longrightarrow k = 1$

Ces 4 exemples nous montrent :

1° que nous sommes théoriquement capables de calculer k dans tous les cas singuliers à condition de savoir trouver le nombre de racines réelles du PGCD.

2° que la rencontre d'un cas singulier du 2ème type ne signi-

fie pas nécessairement que f ait des racines imaginaires pures, mais que, par contre, si f a des racines imaginaires pures, il est très vraisemblable que dans le schéma de Routh, nous ayions un cas singulier du 2ème type (à moins que le schéma ait été arrêté prématurément par un cas singulier du 1er type comme dans l'exemple 1).

Cette 2ème remarque nous amène à nous demander s'il n'y aurait pas une condition nécessaire et suffisante pour que le PGCD de f_1 et f_2 soit de degré > 0.

2° - Etude du cas où le PGCD de f_1 et f_2 a un degré positif.

Dans ce cas, f_1 et f_2 ont des racines communes. Nous allons chercher d'où proviennent ces racines.

Remarque :

$$f(z) = a_0 z^n + b_0 z^{n-1} + a_1 z^{n-2} + b_1 z^{n-3} + \ldots.$$

$$\begin{cases} f_1(z) = a_0 z^n - a_1 z^{n-2} + a_2 z^{n-4} - \ldots.. \\ f_2(z) = b_0 z^{n-1} - b_1 z^{n-3} + b_2 z^{n-5} - \ldots.. \end{cases}$$

Si z_0 est une racine commune à f_1 et f_2, chaque polynôme n'ayant que des monômes de même parité, $-z_0$ est racine de f_1 et f_2.

Si z_0 est une racine complexe du PGCD, $-z_0$, $\overline{z_0}$, $-\overline{z_0}$ le sont aussi ; le PGCD est alors au moins de degré 4.

Si z_0 est racine réelle du PGCD
Si z_0 est racine imaginaire pure du PGCD
} Celui-ci est au moins de degré 2

Il y a un seul cas où le PGCD peut être de degré 1 c'est lorsque f_1 et f_2 ont en commun la racine 0.

Soit donc z_0 une racine du PGCD

$$f_1(z_0) = a_0 z_0^n - a_1 z_0^{n-2} + a_2 z_0^{n-4} - \ldots = 0$$
$$f_2(z_0) = b_0 z_0^{n-1} - b_1 z_0^{n-3} + b_2 z_0^{n-5} - \ldots = 0$$

Nous supposerons n impair pour fixer les idées

Considérons les polynômes
$$\begin{cases} g_1(z) = a_0 z^n + a_1 z^{n-2} + a_2 z^{n-4} + \ldots \\ g_2(z) = b_0 z^{n-1} + b_1 z^{n-3} + b_2 z^{n-5} + \ldots \end{cases}$$

Calculons $g_1(i z_0)$

$$g_1(i z_0) = a_0 i^n z_0^n + a_1 i^{n-2} z_0^{n-2} + a_2 i^{n-4} z_0^{n-4} + \ldots$$

Suivant la valeur de n impair, la suite i^n , i^{n-2}, $i^{n-4}, \ldots$ est la suite alternée $+i$, $-i$, $+i$, $-i \ldots$ ou la suite alternée $-i$, $+i$, $-i$, $+i \ldots$

Si on a la 1ère suite, $g_1(i z_0) = i(a_0 z_0^n - a_1 z_0^{n-2} + a_2 z_0^{n-4} - \ldots$

$$g_1(i z_0) = i \; f_1(z_0) = 0$$

Si on a la 2ème suite, $g_1(i z_0) = -i \; f_1(z_0) = 0$

De même $g_2(i z_0) = b_0 i^{n-1} z_0^{n-1} + b_1 i^{n-3} z_0^{n-3} + b_2 i^{n-5} z_0^{n-5} + \ldots$

où la suite i^{n-1}, i^{n-3}, $i^{n-5}, \ldots$ est soit la suite alternée $+1$, -1, $+1$, $-1, \ldots$ soit la suite alternée -1, $+1$, -1, $+1, \ldots$

donc, suivant n, $g_2(i\ z_0) = \pm(b_0\ z_0^{n-1} - b_1\ z_0^{n-3} + b_2\ z_0^{n-5} - \ldots)$

$$g_2(i\ z_0) = \pm\ f_2(z_0) = 0$$

Or $f(i\ z_0) = g_1(i\ z_0) + g_2(i\ z_0) \Longrightarrow f(i\ z_0) = 0$

Donc, si z_0 est racine du PGCD de f_1 et f_2, iz_0 est racine de f

Si le PGCD a 2 racines réelles opposées f a 2 racines imaginaires pures opposées

Si le PGCD a 2 racines imaginaires pures opposées, f a 2 racines réelles opposées

Si le PGCD a 2 couples de racines complexes opposées, f a 2 couples de racines complexes opposées obtenues par rotation de $\pi/2$.

La démonstration serait semblable pour n pair.

Voyons la réciproque :

Supposons que f ait z_0 et $-z_0$ comme racines. Nous prendrons n pair.

$$f(z_0) = a_0 z_0^n + b_0 z_0^{n-1} + a_1 z_0^{n-2} + b_1 z_0^{n-3} + \ldots$$
$$f(-z_0) = a_0 z_0^n - b_0 z_0^{n-1} + a_1 z_0^{n-2} - b_1 z_0^{n-3} + \ldots$$

Par addition et soustraction de $f(z_0)$ et $f(-z_0)$, nous obtenons

$$\begin{cases} a_0\ z_0^n + a_1\ z_0^{n-2} + a_2\ z_0^{n-4} + \ldots = 0 \\ b_0\ z_0^{n-1} + b_1\ z_0^{n-3} + b_2\ z_0^{n-5} + \ldots = 0 \end{cases}$$

Nous avons :

$$f_1(i\ z_0) = a_0\ i^n\ z_0^n - a_1\ i^{n-2}\ z_0^{n-2} + a_2\ i^{n-4}\ z_0^{n-4} - \ldots$$

$$f_1(i\ z_0) = \pm\ (a_0\ z_0^n + a_1\ z_0^{n-2} + a_2\ z_0^{n-4} + \ldots)$$

$$f_1(i\ z_0) = 0$$

De même

$$f_2(i\ z_0) = b_0\ i^{n-1}\ z_0^{n-1} - b_1\ i^{n-3}\ z_0^{n-3} + b_2\ i^{n-5}\ z_0^{n-5} - \ldots$$

$$f_2(i\ z_0) = \pm\ i\ (b_0\ z_0^{n-1} + b_1\ z_0^{n-3} + b_2\ z_0^{n-5} + \ldots)$$

$$f_2(i\ z_0) = 0.$$

$i\ z_0$ est alors racine du PGCD.

D'où :

Théorème :

Une condition nécessaire et suffisante pour que le PGCD de f_1 et f_2 possède des racines est que f possède des racines opposées.

L'ensemble des racines du PGCD est alors l'ensemble des racines de f 2 à 2 opposées (l'ordre de multiplicité étant conservé), chacune étant multipliée par i.

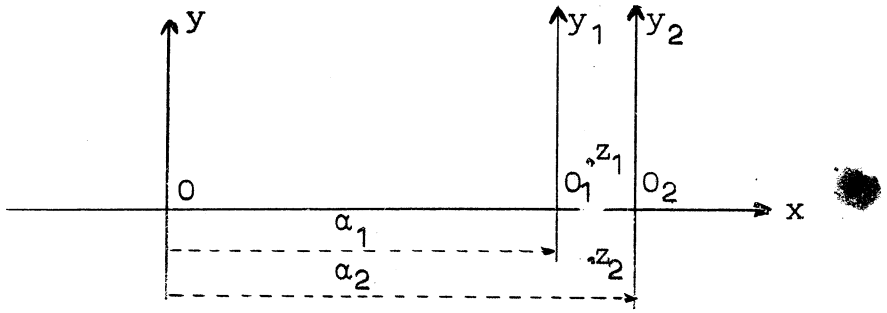
CHAPITRE II

METHODE NUMERIQUE DE RESOLUTION DES EQUATIONS ALGEBRIQUES BASEE SUR L'ALGORITHME DE ROUTH

I - INTRODUCTION

L'utilisation de l'algorithme de Routh comme méthode de résolution des équations algébriques est simple, au moins dans sa description générale. On peut toujours, étant donné un système d'axes x O y et une équation $f(z) = 0$ (de racines z_i , $i = 1,\ldots,n$), faire le changement de variable $Z = z-\alpha$. Cela revient à appliquer une translation α sur l'axe imaginaire. On obtient une nouvelle équation $g(Z) = 0$ dont les racines Z_i vérifient :

$$Z_i = z_i - \alpha \quad (i=1,\ldots,n)$$



Si on procède à différentes translations de Oy et si on applique à chaque fois le théorème de Routh qui donne le nombre de racines à droite du nouvel axe imaginaire, il est clair qu'on doit pouvoir localiser les racines de f correspondant à la même partie réelle dans une "bande" de largeur aussi réduite qu'on veut. Par exemple, les 2 tranlations α_1 et α_2 permettant de localiser les racines z_1 et z_2 dans une bande de largeur $|\alpha_1 - \alpha_2|$.

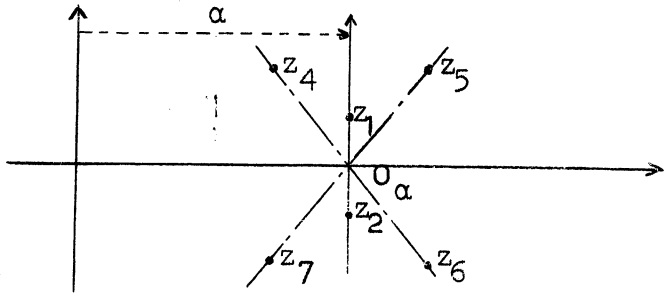
II - L'ALGORITHME NORMAL

1°) Description de l'algorithme.

Il consiste à conduire l'algorithme précédent jusqu' à une largeur de bande petite et fixée à l'avance. Si on prend alors un point réel α dans cette bande, et si on applique la translation α à Oy, on obtient un polynôme f sur lequel le schéma de Routh présente un cas singulier (ou pratiquement tel). f possède des racines imaginaires pures qui sont les parties imaginaires des racines de polynôme initial enfermées dans la bande. Le PGCD de f_1 et f_2 (pour reprendre les notations du chapitre I), possède au moins comme racines (et ce sont des racines réelles) ces parties imaginaires.

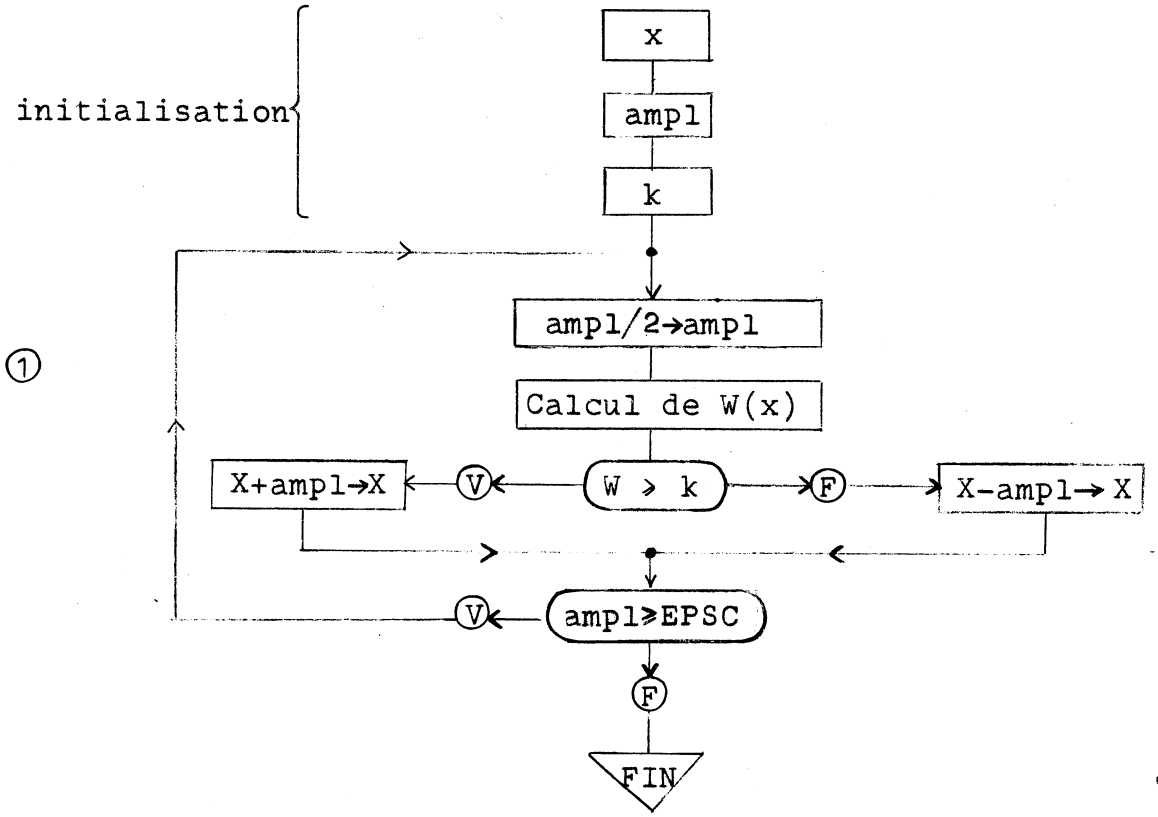
L'obtention des parties réelles conduit en même temps aux parties imaginaires qui s'identifient aux racines réelles du PGCD.

Pratiquement, quand on a atteint la partie réelle, on considère que le PGCD de f_1 et f_2 est le polynôme de la chaîne de Sturm dont le degré correspond au nombre de racines dans la bande. Ceci est toujours le cas, sauf si les racines présentent certains caractères de symétrie vus au théorème de la fin du chapitre I.



Ainsi, z_4 et z_5 étant respectivement symétriques de z_6 et z_7 par rapport à O_α , le PGCD ne sera pas de degré 2 mais 6.

Pour obtenir une partie réelle, on procéde par dichotomie, comme la méthode de bissection pour les racines réelles. Si W(x) est le nombre de racines dont la partie réelle est à droite de x, si n est le degré du polynôme et si on a déjà trouvé n-k racines, l'organigramme de recherche de la partie réelle est le suivant :



A la sortie, on obtient la partie réelle x à EPSC près. On peut, quand on a trouvé les racines correspondant à une par-

tie réelle, les éliminer par déflation ou ne pas les éliminer: c'est un avantage de la méthode que de ne pas exiger de déflation. Un autre avantage : elle ne s'embarasse pas des racines multiples. On obtient les différentes parties réelles successivement. Comme nous l'avons décrit, cet algorithme normal exige les différentes procédures suivantes :

1) Calcul d'un majorant des modules des racines pour initialiser ampl la première fois.

2) Procédure de calcul des coefficients du polynôme quand on applique à l'axe imaginaire une translation réelle.

3) Procédure d'obtention de la chaîne de Sturm construite sur f_1 et f_2 par l'algorithme de Routh et du nombre de racines dans le demi-plan $Re(z) > 0$.

4) éventuellement une procédure de division par $x-\alpha$ et par un facteur quadratique.

2°) Les différentes procédures.

a) Calcul d'un majorant ([6] p. 234)

Soit le polynôme

$$f(x) = a_0 x^n + a_1 x^{n-1} + \ldots\ldots + a_{n-1} x + a_n$$

un majorant appelé Sup, des modules des racines est donné par

$$\text{Sup} = 2 \times \underset{k=1,2,..,n}{\text{Maximum}} \left[\left| \frac{a_k}{a_0} \right|^{1/k} \right] \qquad (2)$$

démonstration : prenons $|x| > 1$. Posons $A = \underset{k=1,2,..,n}{\text{Maximum}} \left(\left| \frac{a_k}{a_0} \right| \right)$

$$f(x) = a_0 x^n \left(1 + \frac{a_1}{a_0 x} + \frac{a_2}{a_0 x^2} + \ldots + \frac{a_n}{a_0 x^n}\right)$$

$$\left|f(x)\right| = \left|a_0 x^n\right| \times \left|1 + \frac{a_1}{a_0 x} + \ldots + \frac{a_n}{a_0 x^n}\right|$$

$$\left|f(x)\right| \geq \left|a_0 x^n\right| \times \left|1 - \left|\frac{a_1}{a_0 x} + \ldots + \frac{a_n}{a_0 x^n}\right|\right|$$

$$\text{donc} \qquad f(x) \geqslant \left| a_0 x^n \right| x \left(1 - \left| \frac{a_1}{a_0 x} + \ldots + \frac{a_n}{a_0 x^n} \right|\right)$$

$$\text{par suite } f(x) \geqslant \left| a_0 x^n \right| x \left(1 - A\left(\frac{1}{|x|} + \frac{1}{|x|^2} + \ldots + \frac{1}{|x|^n}\right)\right)$$

$$\text{et à fortiori } \left|f(x)\right| \geqslant \left| a_0 x^n \right| x \left(1 - A \, x \sum_{n=1}^{\infty} \frac{1}{|x|^n}\right)$$

$$\text{donc} \qquad \left|f(x)\right| \geqslant \left| a_0 x^n \right| x \left[1 - \frac{A}{|x| - 1}\right]$$

$$\left|f(x)\right| \geqslant \left| a_0 x^n \right| x \, \frac{|x| - 1 - A}{|x| - 1}$$

pour que x soit un majorant des modules des racines, il faut s'assurer que $\left|f(x)\right| > 0$. Ce sera vérifié en particulier si

$$|x| - 1 - A > 0$$

$$|x| > 1 + A$$

un majorant des modules des racines est donc donné par

$$1 + \underset{k=1,2,\ldots,n}{\text{Maximum}} \left| \frac{a_k}{a_0} \right|$$

Prenons maintenant $\rho > 0$

$$\frac{1}{\rho^n} f(x) = a_0 \left(\frac{x}{\rho}\right)^n + \frac{a_1}{\rho}\left(\frac{x}{\rho}\right)^{n-1} + \frac{a_2}{\rho^2}\left(\frac{x}{\rho}\right)^{n-2} + \ldots + \frac{a_n}{\rho^n}$$

pour ce polynôme en $\frac{x}{\rho}$ le majorant précédent est

$$1 + \underset{k=1,2,\ldots,n}{\text{Maximum}} \left| \frac{a_k}{a_0 \, \rho^k} \right|$$

soit

$$|x| > \rho + \underset{k=1,2,\ldots,n}{\text{Maximum}} \left| \frac{a_k}{a_0 \rho^{k-1}} \right|$$

Soit alors $\rho = \underset{k=1,2,\ldots,n}{\text{Maximum}} \sqrt[k]{\left|\frac{a_k}{a_0}\right|}$

Cela entraîne $\left|\frac{a_k}{a_0}\right| \leqslant \rho^k$

donc $\left|\frac{a_k}{a_0\ \rho^{k-1}}\right| \leqslant \rho$

par suite $\underset{k=1,2,\ldots,n}{\text{Maximum}} \left|\frac{a_k}{a_0\ \rho^{k-1}}\right| \leqslant \rho$

donc un autre majorant sera donné par

$$\rho + \rho = 2\rho = 2 \underset{k=1,2,\ldots,n}{\text{Maximum}} \sqrt[k]{\left|\frac{a_k}{a_0}\right|}$$

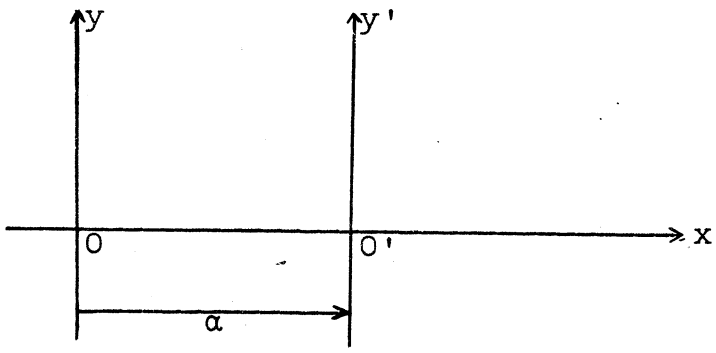
d'où la formule ② . Nous avons appelé MAXMOD la procédure ALGOL de ce calcul.

b) Translation de l'axe imaginaire

Soit le polynôme

$$f(x) = a_0\ x^n + a_1\ x^{n-1} + \ldots + a_{n-1}\ x + a_n$$

et Γ sa représentation dans le système d'axe x 0 y



Translatons Oy d'une valeur réelle $\alpha \longrightarrow O'y'$. Cela revient à faire le changement de variable $X = x-\alpha$

Dans x O' y' , Γ représente le polynôme

③ $$g(X) = A_0 X^n + A_1 X^{n-1} + \ldots + A_{n-1} X + A_n$$

Posons :

$$\begin{aligned} f(x) = f_n(x) &= (x-\alpha) f_{n-1}(x) + Rn \\ f_{n-1}(x) &= (x-\alpha) f_{n-2}(x) + R_{n-1} \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ f_2(x) &= (x-\alpha) f_1(x) + R_2 \\ f_1(x) &= (x-\alpha) \underbrace{f_0(x)}_{R_0} + R_1 \end{aligned}$$

Cette suite de divisions successives nous permet d'évaluer f(x) :

④ $$f(x) = R_n + R_{n-1} (x-\alpha) + R_{n-2}(x-\alpha)^2 + \ldots + R_1 (x-\alpha)^{n-1} + R_0 (x-\alpha)^n$$

De ③ et ④ on tire

⑤ $$A_p = R_p \qquad p = 0,1,\ldots,n$$

Pour obtenir les coefficients A_p, il suffit de calculer les différents restes R_p des divisons successives. Ceci s'effectue à l'aide du schéma de Horner. Si nous posons

$$f_p(x) = b_0 x^p + b_1 x^{p-1} + \ldots + b_{p-1} x + b_p$$

$$f_{p-1}(x) = c_0 x^{p-1} + c_1 x^{p-2} + \ldots + c_{p-2} x + c_{p-1} ,$$

la relation

$$f_p(x) = (x-\alpha)\, f_{p-1}(x) + R_p$$

donne, en identifiant les coefficients des puissances égales :

$$\begin{aligned}
c_0 &= b_0 \\
c_1 - \alpha c_0 &= b_1 \\
c_2 - \alpha c_1 &= b_2 \\
&\cdots\cdots\cdots\cdots \\
c_{p-1} - \alpha c_{p-2} &= b_{p-1} \\
R_p - \alpha c_{p-1} &= b_p
\end{aligned}
\qquad \text{ou bien } ⑥ \qquad
\begin{aligned}
\boxed{c_0} &= b_0 \\
\boxed{c_1} &= b_1 + \alpha c_0 \\
\boxed{c_2} &= b_2 + \alpha c_1 \\
&\cdots\cdots\cdots\cdots \\
\boxed{c_{p-1}} &= b_{p-1} + \alpha c_{p-2} \\
R_p &= b_p + \alpha c_{p-1}
\end{aligned}$$

Le schéma de Horner produit, non seulement le reste, mais encore les coefficients du quotient de la division de $f_p(x)$ par $x-\alpha$. La procédure consistera donc en n schémas de Horner consécutifs, sur des polynômes dont le degré décroît de 1 à chaque fois. Le calcul effectif en machine nécessite bien entendu de conserver à chaque fois les coefficients encadrés dans la formule ⑥. Cette façon d'obtenir les coefficients de g(X) est la plus économique. On l'emploiera à chaque translation de l'axe imaginaire. La procédure ALGOL de cet algorithme a reçu le nom de TRANSLREEL.

Nous aurons besoin ultérieurement de la remarque suivante :

$$f(x) = f(\alpha+X) = f(\alpha) + f'(\alpha)X + \frac{f''(\alpha)}{2!}X^2 + \ldots + \frac{f^{(n-1)}(\alpha)}{(n-1)!}X^{n-1} + \frac{f^{(n)}(\alpha)}{n!}X^n$$

en comparant avec ③ on déduit :

$$⑦ \qquad A_p = \frac{f^{(n-p)}(\alpha)}{(n-p)!} \qquad p = 0,1,\ldots,n$$

c) L'algorithme de Routh.

Il consiste à déterminer les différents polynômes de la chaîne de Sturm à partir de f_1 et f_2. En reprenant les notations du chapitre I ; on obtient successivement f_3, f_4,...:

f_1	a_0	a_1	a_2	$a_3 \cdots$
f_2	b_0	b_1	b_2	$b_3 \cdots$
f_3	c_0	c_1	c_2	$c_3 \cdots$
....	.....			

Nous avons appelé SCHEMA la procédure ALGOL de cet algorithme. On se reportera au § III du chapitre I. Précisons cependant que, par exemple, il est préférable de calculer c_1 à l'aide de

$$c_1 = a_2 - \frac{a_0}{b_0} b_2$$

plutôt qu'avec

$$c_1 = \frac{a_2 b_0 - a_0 b_2}{b_0}$$

car on gagne une multiplication.

d) Procédures de Déflation.(qui pourront être employées ou pas).

- division par $x-\alpha$: c'est ce que nous venons de voir
- division par $x^2 - sx + p$

Soit le polynôme

$$f(x) = a_0 x^n + a_1 x^{n-1} + \ldots + a_{n-1} x + a_n$$

qu'on désire diviser par

$$g(x) = x^2 - sx + p$$

Posons

$$f(x) = g(x) \times q(x) + r(x)$$

avec

$$q(x) = b_0 x^{n-2} + b_1 x^{n-3} + \ldots + b_{n-3} x + b_{n-2}$$

et mettons le reste sous la forme

$$r(x) = b_{n-1} (x-s) + b_n$$

On obtient alors, en identifiant les coefficients des puissances égales

⑥

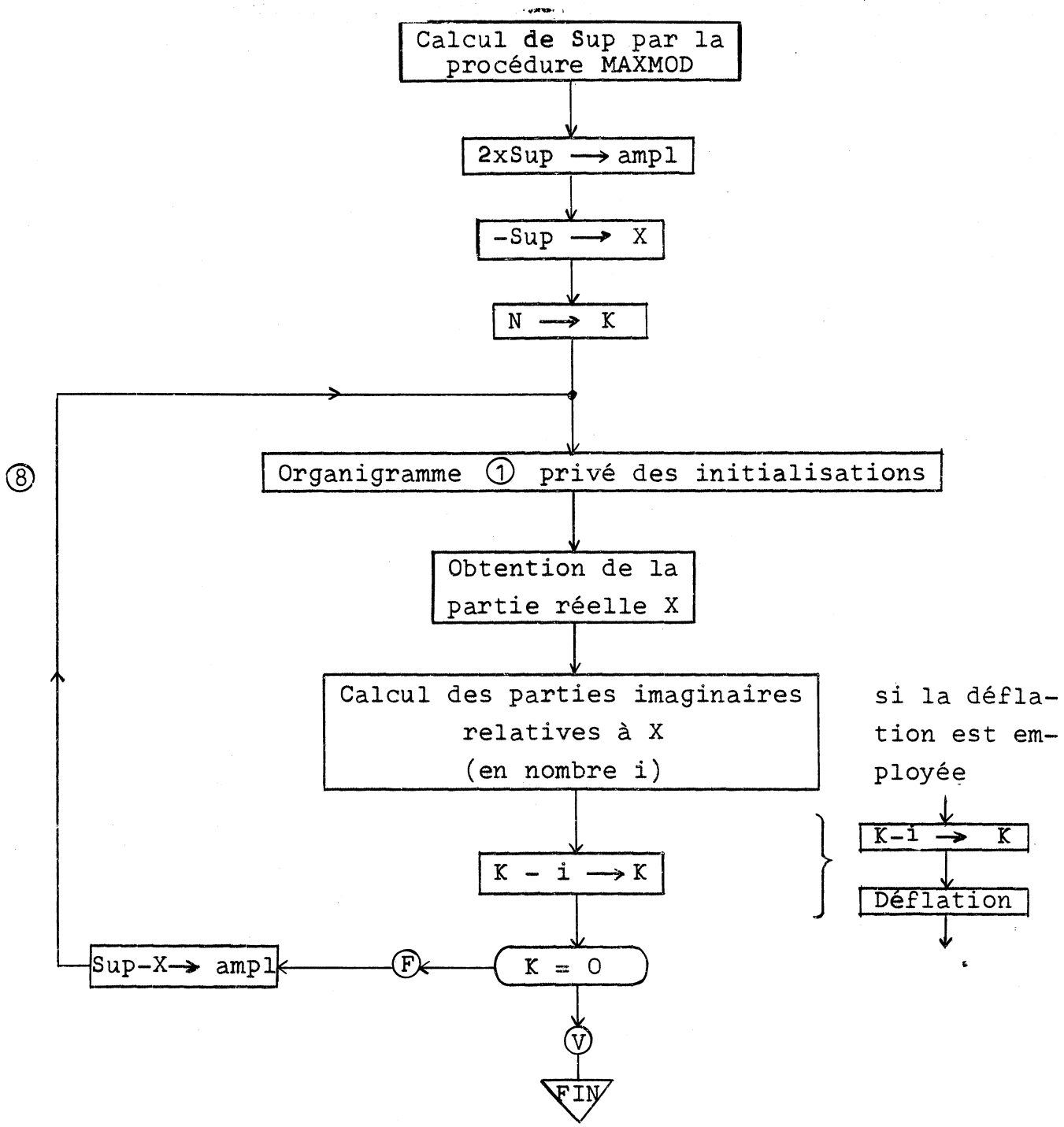
$$\begin{aligned}
b_0 &= a_0 \\
b_1 &= a_1 + sb_0 \\
b_2 &= a_2 + sb_1 - pb_0 \\
b_3 &= a_3 + sb_2 - pb_1 \\
&\cdots\cdots\cdots\cdots\cdots\cdots \\
b_q &= a_q + sb_{q-1} - pb_{q-2} \\
&\cdots\cdots\cdots\cdots\cdots\cdots \\
b_{n-2} &= a_{n-2} + sb_{n-3} - pb_{n-4} \\
b_{n-1} &= a_{n-1} + sb_{n-2} - pb_{n-3} \\
b_n &= a_n + sb_{n-1} - pb_{n-2}
\end{aligned}$$

Ces deux procédures ALGOL ont été appelées respectivement DEFLAT1 et DEFLAT2.

3°) Organigramme complet de l'algorithme normal.

La procédure consiste à employer autant de fois qu'il y a de parties réelles la procédure partielle décrite par l'organigramme ① . L'axe réel est balayé de gauche à droite à partir de -Sup. Dans l'organigramme suivant, ampl permet de pro-

céder par dichotomie, N est le degré du polynôme, K le nombre de racines qu'il reste à calculer.



III - L'ALGORITHME MODIFIE.

1°) Principe.

La modification envisagée ne prétend pas améliorer la précision du calcul des racines, mais espère diminuer le temps de calcul.

Supposons que nous ayions à rechercher p racines de même partie réelle x. D'après ce que nous avons vu, cela revient à chercher le point x présentant un cas singulier pour le schéma de Routh. A ce point x, l'algorithme de Routh conduit à un PGCD de degré p (sauf cas malheureux), la ligne suivante dans le schéma étant une ligne de zéros.

⑨

	f_1	a_0	a_1	a_2	a_3 ...
	f_2	b_0	b_1	b_2	b_3 ...
	f_3	c_0	c_1	c_2	c_3 ...
	.....				
PGCD	$f_{(n+1-p)}$	h_0	h_1	h_2	
ligne de zéros	$f_{(n+2-p)}$	$k_0=0$	$k_1=0$	$k_2=0$ ...	

L'algorithme normal 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".

Une autre façon de procéder consiste à rechercher directement le point x qui annule la $(n+2-p)^{ième}$ ligne. C'est le principe de l'algorithme modifié.

Nous avons vu (formule ⑦ § II) que les coefficients des deux premières lignes du schéma de Routh sont des fonctions de α , paramètre de translation de l'axe imaginaire. Dans la disposition ⑨ :

$$a_0(\alpha) = \frac{f^{(n)}(\alpha)}{n!} \qquad a_1(\alpha) = \frac{f^{(n-2)}(\alpha)}{(n-2)!} \qquad a_2(\alpha) = \frac{f^{(n-4)}(\alpha)}{(n-4)!} \cdots$$

⑩

$$b_0(\alpha) = \frac{f^{(n-1)}(\alpha)}{(n-1)!} \qquad b_1(\alpha) = \frac{f^{(n-3)}(\alpha)}{(n-3)!} \qquad b_2(\alpha) = \frac{f^{(n-5)}(\alpha)}{(n-5)!} \cdots$$

avec

$$a_0(0) = a_0 \qquad a_1(0) = a_1 \qquad a_2(0) = a_2 \ \ldots\ldots$$

$$b_0(0) = b_0 \qquad b_1(0) = b_1 \qquad b_2(0) = b_2 \ \ldots\ldots$$

Ces différentes fonctions sont des polynômes en α. Par conséquent, d'après l'algorithme qui les génère, les lignes suivantes du schéma sont des fractions rationnelles et l'on peut essayer de déterminer le point α tel que (disposition ⑨)

$$k_0(\alpha) = 0 \qquad k_1(\alpha)=0 \qquad k_2(\alpha)=0.$$

Ces fonctions $k(\alpha)$ ne sont pas explicitement connues, mais on peut obtenir leurs valeurs discrètement au moyen de l'algorithme de Routh. Nous allons montrer qu'on peut employer la méthode de Newton pour déterminer leurs zéros réels. Cela va nécessiter la connaissance de leurs dérivées, au moins d'une façon discrète.

2°) <u>Algorithme donnant les dérivées des coefficients du schéma de Routh</u>.

Soient 3 lignes quelconques du schéma

$m_0(\alpha).....m_i(\alpha)$ $\quad$ $m_{i+1}(\alpha).....$

$n_0(\alpha).....n_i(\alpha)$ $\quad$ $n_{i+1}(\alpha).....$

$p_0(\alpha).....p_i(\alpha)$ $\quad$ $.....$

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

⑪
$$p_i'(\alpha) = m'_{i+1}(\alpha) - \frac{\left[m'_0(\alpha)n_{i+1}(\alpha)+m_0(\alpha)n'_{i+1}(\alpha)\right] n_0(\alpha)-m_0(\alpha)n_{i+1}(\alpha)n'_0(\alpha)}{\left[n_0(\alpha)\right]^2}$$

La connaissance des 2 premières lignes et des 2 premières lignes dérivées permet d'obtenir la 3ème ligne dérivée. Or, on connaît les dérivées des coefficients des 2 premières lignes du schéma (disposition ⑩)

⑫
$$a'_0(\alpha) = \frac{f^{(n+1)}(\alpha)}{n!} \equiv 0 \qquad a'_1(\alpha) = \frac{f^{(n-1)}(\alpha)}{(n-2)!} \qquad a'_2(\alpha) = \frac{f^{(n-3}(\alpha)}{(n-4)}$$

$$b'_0(\alpha) = \frac{f^{(n)}(\alpha)}{(n-1)!} \qquad b'_1(\alpha) = \frac{f^{(n-2)}(\alpha)}{(n-3)!} \qquad b'_2(\alpha) = \frac{f^{(n-4}(\alpha)}{(n-5)}$$

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

Pratiquement, ce sont les fonctions et les dérivées au point 0 qui nous sont accessibles; pour $\alpha=0$ ⑩ devient

$$a_0 \quad a_1 \quad a_2 \quad a_3 \cdots$$
$$b_0 \quad b_1 \quad b_2 \quad b_2 \cdots$$

tandis que ⑫ se réduit à :

⑬
$$0 \qquad (n-1)b_0 \qquad (n-3)b_1 \qquad (n-5)b_2 \cdots$$
$$na_0 \qquad (n-2)a_1 \qquad (n-4)a_2 \qquad (n-6)a_3 \cdots$$

et l'algorithme ⑪ donne successivement les dérivées au point 0 des lignes suivantes. Nous pourrons ainsi avoir les dérivées en un point quelconque d'abscisse α en adoptant le processus :

polynôme initial $\xrightarrow{\text{translation } \alpha \text{ de l'axe imaginaire}}$ nouveau polynôme $\longrightarrow$

$\longrightarrow$ schéma de Routh de ce polynôme $\longrightarrow$ schéma dérivé au point 0 pour ce nouveau polynôme

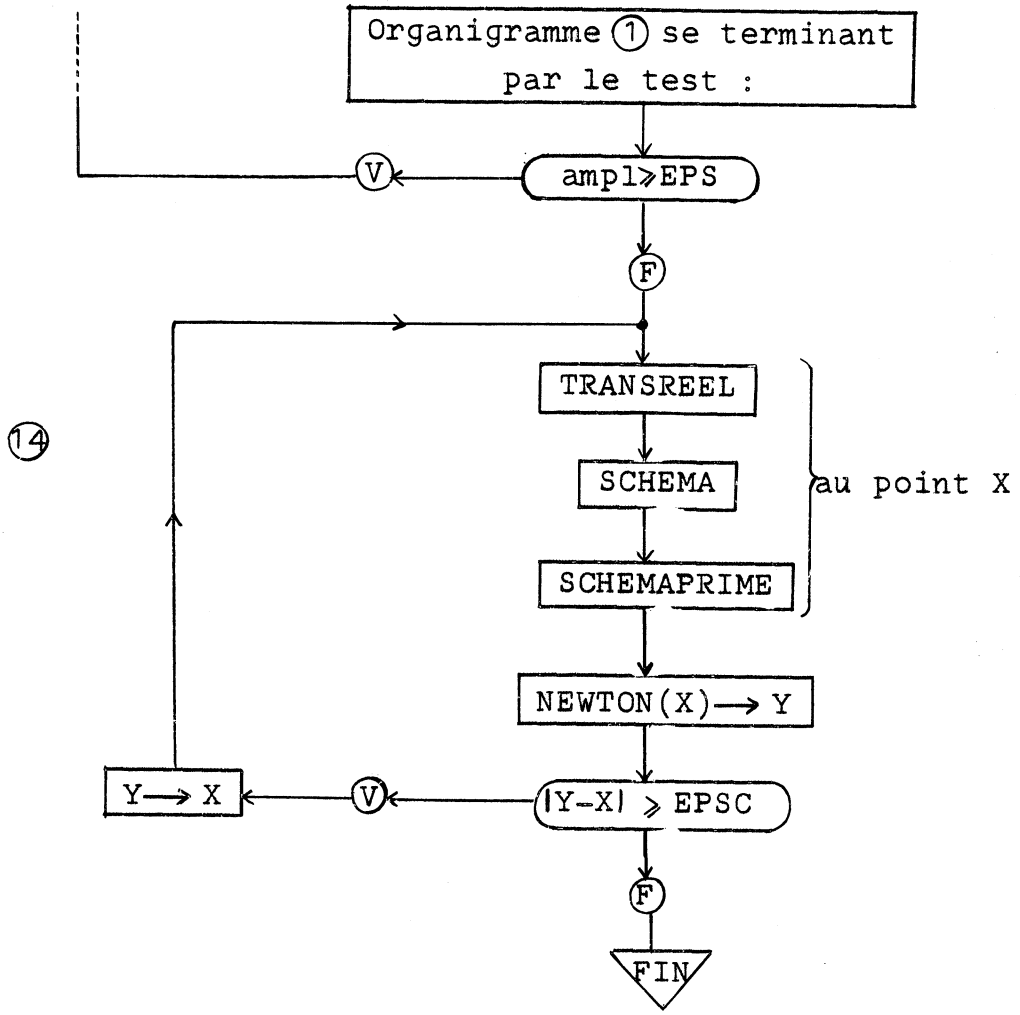
3°) Organigramme de l'algorithme modifié.

Nous appellerons provisoirement SCHEMAPRIME la procédure donnant les coefficients dérivés ; elle nécessite la connaissance préalable du schéma de Routh et dans l'emploi, nous aurons toujours SCHEMA précédant SCHEMAPRIME. Elle nécessite aussi l'initialisation des deux premières lignes aux valeurs ⑬ .

Pratiquement, nous conduirons l'algorithme normal jusqu'à une distance EPS de la racine (c'est-à-dire lorsque ampl devient < EPS) , distance jugée suffisamment petite pour "enclencher" l'algorithme modifié. A partir de là, les nouveaux

points x sont déterminés non plus par dichotomie mais par la méthode de Newton appliquée aux coefficients $k_0(\alpha)$ $k_1(\alpha)$... (disposition ⑨). Un cas particulièrement intéressant est celui des racines complexes simples. Le PGCD est alors de degré 2 et la ligne suivante se réduit à un coefficient : $k_0(\alpha)$. L'algorithme modifié réclame, outre les procédures nécessaires à l'algorithme normal, la procédure SCHEMAPRIME et une procédure de la méthode de Newton que nous appelons NEWTON dans le programme ALGOL.

Donnons les compléments à apporter à l'organigramme ① pour le calcul d'une racine.



On remplace dans l'organigramme complet ⑧ , ① par ⑭ pour obtenir toutes les racines.

4°) Evaluation du nombre d'opérations nécessité par les deux algorithmes.

a) procédure TRANSLREEL.

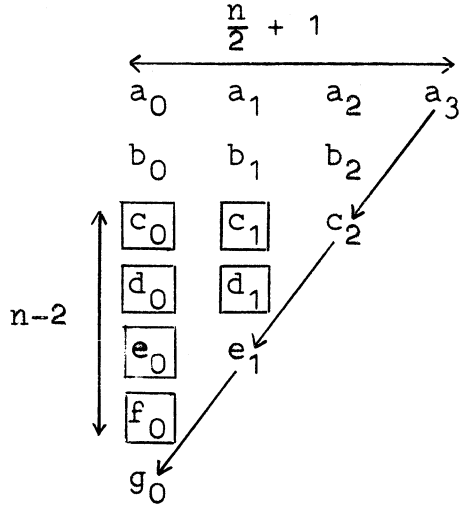
On a vu que c'est n schéma de Horner sur des polynômes de degrés n, n-1,.....,1. Chaque schéma coûte p multiplications et p additions (p=n,n-1,...,1) ; donc TRANSLREEL demande $n+(n-1)+\ldots+1$ soit

⑮
$$\begin{cases} \frac{n(n+1)}{2} \text{ ou } \frac{n^2}{2} + \frac{n}{2} \text{ multiplications} \\ \frac{n(n+1)}{2} \text{ ou } \frac{n^2}{2} + \frac{n}{2} \text{ additions} \end{cases}$$

b) procédure SCHEMA.

Pour les deux procédures SCHEMA et SCHEMAPRIME, nous ne ferons le calcul que dans le cas n pair. Nous donnerons ensuite les résultats pour n impair. Le schéma de Routh est alors à (n+1) lignes et $(\frac{n}{2} + 1)$ colonnes et exige le calcul de $\frac{n(n-2)}{4}$ coefficients (n-2 lignes à calculer comportant $(\frac{n}{2} - 1)$ termes pour les deux premières, $\frac{n}{2} - 2$ termes pour les deux suivantes,..., 1 terme pour les 2 dernières, soit $2\left[\frac{n}{2} - 1 + \frac{n}{2} - 2 + \ldots + 1\right]$ termes).

Pour fixer les idées, avec n=6 on a le schéma



On a encadré les termes à calculer.
On a $g_0=e_1=c_2=a_3$

Le calcul se faisant selon l'algorithme

$$P_i = m_{i+1} - \frac{m_0}{n_0} n_{i+1} ,$$

chaque ligne demande 1 division et chaque terme 1 multiplication et 1 addition. SCHEMA nécessite donc

⑯ $\begin{cases} n-2 \text{ divisions} \\ \frac{n^2}{4} - \frac{n}{2} \text{ multiplications} \\ \frac{n^2}{4} - \frac{n}{2} \text{ additions} \end{cases}$ et pour n impair $\begin{cases} n-2 \text{ divisions} \\ \frac{n-1}{2}(\frac{n-3}{2}+1) \text{ mulplication} \\ \frac{n-1}{2}(\frac{n-3}{2}+1) \text{ additions} \end{cases}$

c) <u>procédure donnant les coefficients et les coefficients dérivés</u>.

Nous allons revenir sur l'algorithme ⑪ donnant les coefficients dérivés. Cette formule peut s'écrire :

$$P'_i = m'_{i+1} - \left(\left[(m'_0 - \boxed{\frac{m_0}{n_0}} \times n'_0)/n_0 \right] \times n_{i+1} + \boxed{\frac{m_0}{n_0}} \times n'_{i+1} \right)$$

Les termes entourés d'un rectangle étant déjà calculés par SCHEMA, si l'on combine les deux procédures, on doit gagner un certain nombre d'opérations. De plus, le terme souligné est à calculer une seule fois par ligne.

Nous avons, comme dans SCHEMA, (n-2) lignes à calculer (en fait la 1ère ligne à calculer est plus simple car $a'_0 = 0$, mais nous n'en tiendrons pas compte), et $\frac{n(n-2)}{4}$ coefficients.

Par ligne il faut { 1 division ; 1 multiplication ; 1 addition } → en tout { (n-2) divisions ; (n-2) multiplications ; (n-2) additions }

Par coefficients il faut { 2 multiplications ; 2 additions } → en tout { $\frac{n(n-2)}{2}$ multiplications ; $\frac{n(n-2)}{2}$ additions }

A cela il faut ajouter les opérations d'initialisation des 2 premières lignes. Soit n termes à calculer à raison de 1 multiplication et 1 addition par terme :

{ n multiplications ; n additions }

SCHEMAPRIME simplifié exige donc :

⑰ { n-2 divisions ; $n^2/2$+ n-2 multiplications ; $n^2/2$+ n-2 additions } et pour n impair { n-2 divisions ; $\frac{n-1}{2}(n+3)$ multiplications ; $\frac{n-1}{2}(n+3)$ additions. }

par exemple, pour n=10, il faut

- pour TRANSLREEL { 55 multiplications ; 55 additions }

- pour SCHEMA seul { 8 divisions ; 20 multiplications ; 20 additions }

- pour SCHEMAPRIME simplifié { 8 divisions ; 58 multiplications ; 58 additions }

Ces dernières opérations seront pratiquement le coût supplémentaire d'une itération de l'algorithme modifié et nous espérons qu'il sera compensé par un nombre d'itérations beaucoup moins grand. Sur cet exemple, 1 itération de l'algorithme modifié coûte un peu moins que 2 itérations de l'algorithme normal.

Remarque :

Cette évaluation du nombre d'opérations nécessaire au calcul des coefficients du schéma dérivé est faite dans le cas d'un PGCD de degré 2. Si le PGCD est de degré k supérieur à 2 (s'il y a k racines de même partie réelle par exemple), il n'y a pas lieu de calculer les k-1 dernières lignes et le nombre d'opérations s'en trouve diminué. Donc, on a un gain supérieur encore à employer l'algorithme modifié plutôt que l'algorithme normal. Comme nous le verrons par la suite, nous n'avons pas considéré ce cas qui est l'exception et nous nous sommes limités

à employer l'algorithme modifié à la recherche des PGCD de degré 2. Il est d'autre part évident que pour une racine réelle, l'algorithme modifié, et encore plus l'algorithme normal, ne sont pas rentables. Ce dernier sera seulement utilisé pour localiser la racine réelle qui sera ensuite affinée par une méthode plus appropriée.

5°) Conclusion.

Il serait faux de prétendre que l'algorithme modifié réduise obligatoirement le temps de calcul. Dans le décompte précédent, nous n'avons pas considéré un certain nombre de procédures auxilliaires comme la procédure MAXMOD, les 2 procédures DEFLAT1 et DEFLAT2, communes aux 2 algorithmes, la procédure de la méthode de Newton (dont la procédure ALGOL porte le nom) pour l'algorithme modifié, ni la procédure qui calcule le nombre de variations de signes dans la première colonne du schéma de Routh pour l'algorithme normal. D'une part le nombre d'opérations mis en jeu par ces procédures est faible et d'autre part, il s'équilibre pratiquement dans les deux algorithmes.

Cherchons à partir de quelle valeur du degré n de l'équation on peut se permettre, en employant l'algorithme modifié, la moitié du nombre des itérations nécessaires à l'algorithme normal. Cela revient à chercher à partir de quel n "TRANSREEL plus SCHEMA moins SCHEMAPRIME" conduit à un nombre d'opérations positif.

n pair : $\begin{cases} \text{divisions} & 0 \\ \text{multiplications} & \frac{n^2}{2} + \frac{n}{2} + \frac{n^2}{4} - \frac{n}{2} - \frac{n^2}{2} - n + 2 \\ \text{additions} & \frac{n^2}{2} + \frac{n}{2} + \frac{n^2}{4} - \frac{n}{2} - \frac{n^2}{2} - n + 2 \end{cases}$

Soit $n^2 - 4n + 8 \geqslant 0$.

ou $(n-2)^2 + 4 \geqslant 0$, ce qui est vérifié quel que soit n.

n impair : le calcul nous conduit à

$n^2 - 4n + 7 \geqslant 0$

ce qui est vérifié encore quel que soit n.

Donc, quel que soit le degré de l'équation, l'algorithme modifié peut se permettre un nombre d'itérations égal à la moitié du nombre d'itérations nécessaire à l'algorithme normal.

Par dichotomie, pour affiner la recherche des parties réelles depuis 10^{-2} jusqu'à 10^{-8} il faut 20 itérations (il en faut 17 pour passer de 10^{-2} à 10^{-7}). Donc, l'algorithme modifié peut se permettre jusqu'à 10 itérations dans ce cas pratique.

Or, on peut avoir une valeur approchée de l'erreur à chaque pas pour la méthode de Newton ([3] p. 51). Si $k(\alpha)$ est la fonction, α_0 la racine, e_n l'erreur au pas n , on a :

⑱ $$e_{n+1} = K\, e_n^2 ,$$

avec

$$|K| = \frac{1}{2} \frac{|k''(\alpha_0)|}{|k'(\alpha_0)|}$$

nous obtenons

$$|e_{11}| = |K|^{1023}\ e_1^{1024}$$

en prenant $|e_1| = 10^{-2}$, $|e_{11}|$ sera inférieur ou égal à 10^{-8} si

$$|K|^{1023} \quad (10^{-2})^{1024} \leqslant 10^{-8}$$

donc pratiquement si

$$|K| \leqslant 10^{2040/1023}$$

$$|K| \leqslant 100$$

Si $\frac{|k''(\alpha_0)|}{|k'(\alpha_0|} \leqslant 200$, on passera de 10^{-2} à 10^{-8} en moins de 10 itérations de l'algorithme modifié. On pourrait essayer de décider à l'avance, au niveau du test "ampl $\geqslant$ EPS" dans l'algorithme ⑭ si on va choisir l'algorithme normal ou l'algorithme modifié. Il faudrait, pour cela, qu'on puisse avoir en ce point α situé au plus à EPS de α_0 une approximation suffisante de K. Mais d'une part cela entraîne le calcul de $k''(\alpha)$, soit par une procédure calculant les dérivées secondes des coefficients du schéma de Routh, soit par une formule approchée, d'autre part, il semble que la valeur de K au point α ne soit pas une image suffisamment précise. Nous avons abandonné cette idée. Pour conclure, nous ferons le parallèle avec la recherche des racines réelles. Dans ce cas, la méthode de Newton nécessitant, à chaque itération, pratiquement deux fois plus d'opérations que la méthode de bissection, la première méthode sera plus rentable que la seconde si la racine est atteinte en deux fois moins d'itérations.

IV - PROCEDURE ALGOL DE L'ALGORITHME MODIFIE - RESULTATS NUMERIQUES

1°) Procédure de l'algorithme modifié: Procédure ROUTH NEWTON.

La procédure décrite ci-après n'est pas une procédure de résolution des équations algébriques absolument générale. Nous avons voulu utiliser l'algorithme modifié à la recherche des PGCD de degré 2 pour pouvoir faire des comparaisons de temps de calcul avec l'algorithme normal. Cette procédure ROUTH NEWTON traite en fait principalement les racines simples. Un certain nombre de cas lui échappent : les racines multiples autres que les racines réelles doubles, le cas de plus de 2 racines de même partie réelle, le cas de plus de 2 racines à parties réelles trop voisines et les cas de symétries dans les racines dont nous avons parlé.

Cette procédure permet, soit d'éliminer les racines par déflation après leur obtention, soit de conserver tout au long du calcul les coefficients du polynôme initial. Un paramètre de la procédure permet de choisir l'une ou l'autre option : c'est le paramètre booléen DEFLAT.

a) Option déflation : Le paramètre d'entrée DEFLAT prend la valeur vrai.

Toutes les procédures de l'algorithme normal sont employées, à savoir MAXMOD, TRANSLREEL, SCHEMA, DEFLAT1 et DEFLAT2. Pour le calcul des coefficients dérivés, on a vu qu'on a intérêt à grouper leur calcul avec celui des coefficients du schéma de Routh (§ III 4°) c). Ceci est fait par la procedure DEUX SCHEMAS. On a enfin besoin d'une procédure de l'algorithme de Newton pour affiner les racines réelles simples et surtout pour l'algorithme modifié lui-même :

procedure NEWTON . Deux phases distinctes se succèdent :

1ère phase : l'algorithme normal produit un intervalle [BI, BS] encadrant la partie réelle, et donnant le nombre de racines dans cette bande (BI à gauche de la partie réelle).

2ème phase :

1er cas . racine réelle simple dans la bande BI, BS, détectée par l'algorithme normal : celle-ci est affinée par la méthode de Newton utilisant BI comme valeur de départ ; une fois la racine trouvée, elle est éliminée par la procédure DEFLAT1, le calcul se poursuivant sur le polynôme quotient.

2ème cas. 2 racines complexes conjuguées ou 2 racines réelles confondues : la partie réelle est affinée à l'aide de l'algorithme modifié qui effectue, dans le schéma de Routh, la recherche du PGCD de degré 2 et qui utilise BI comme valeur de départ. Les parties imaginaires sont les 2 racines (réelles symétriques) du PGCD. Il est procédé ensuite à l'élimination de ces deux racines par DEFLAT2.

3ème cas. L'algorithme normal trouve plus de 2 racines dans l'intervalle BI, BS: c'est le cas des racines complexes multiples, celui de plus de 2 racines à même partie réelle, celui de plus de 2 racines à parties réelles très voisines. ROUTH NEWTON donne alors BI, BS et le nombre de racines dans la bande. Il n'est procédé à aucune déflation.

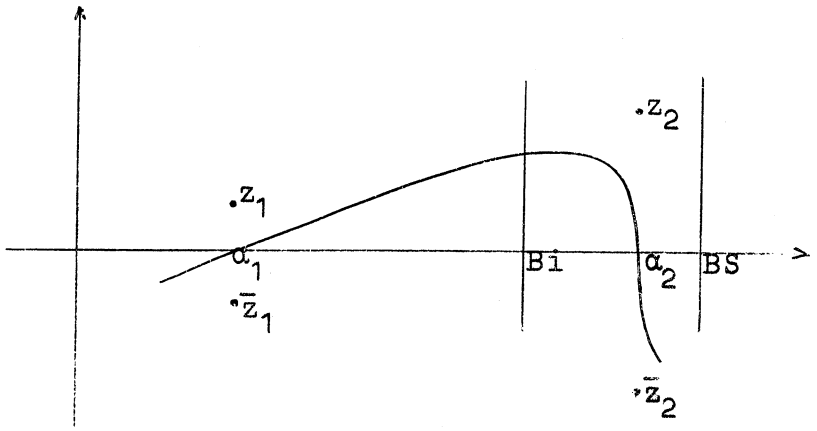
Enfin, si pour une raison quelconque, l'algorithme modifié conduit hors de [BI, BS] , ce cas est assimilé au 3ème cas précédent.

Les avantages de l'emploi de la déflation sont les suivants :

1°) le temps de calcul est diminué car le degré de l'équation est abaissé à chaque extraction.

2°) la plupart des cas de symétrie qui produisent un PGCD de degré supérieur au nombre de parties imaginaires (voir chapitre II § II 1°) sont éliminés car la partie réelle en cours de recherche est la plus à gauche.

3°) la précision à obtenir avec l'algorithme normal lors de la première phase peut être nettement diminuée. En effet, dans le cas suivant (sans déflation)



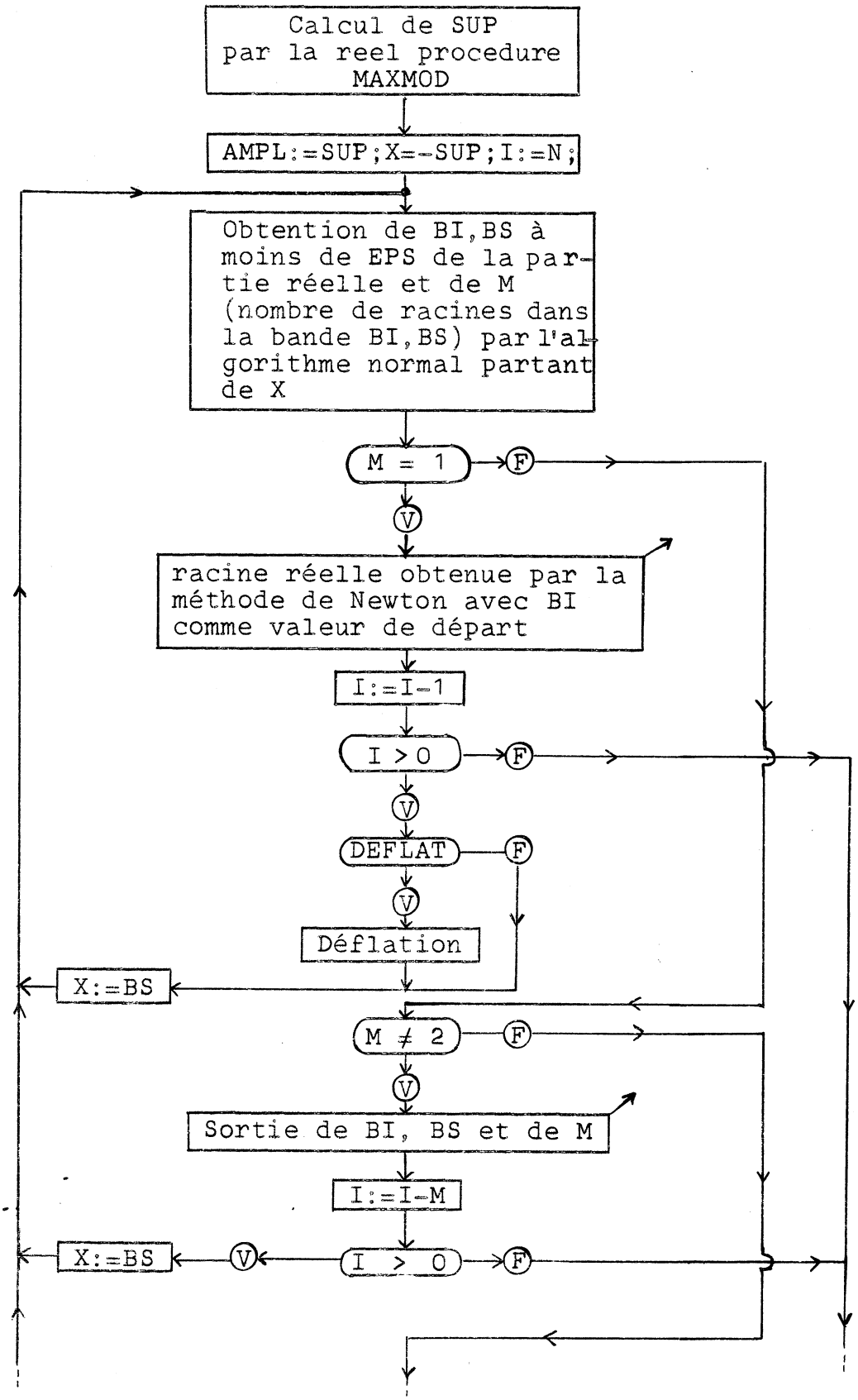
l'algorithme modifié sera "aspiré", dans la recherche de α_2, vers α_1 tant que [BI, BS] ne sera pas très petit, phénomène qui aura moins de chances d'avoir lieu si on prend soin d'éliminer les racines z_1 et $\bar{z}_1$ avant la recherche de α_2.

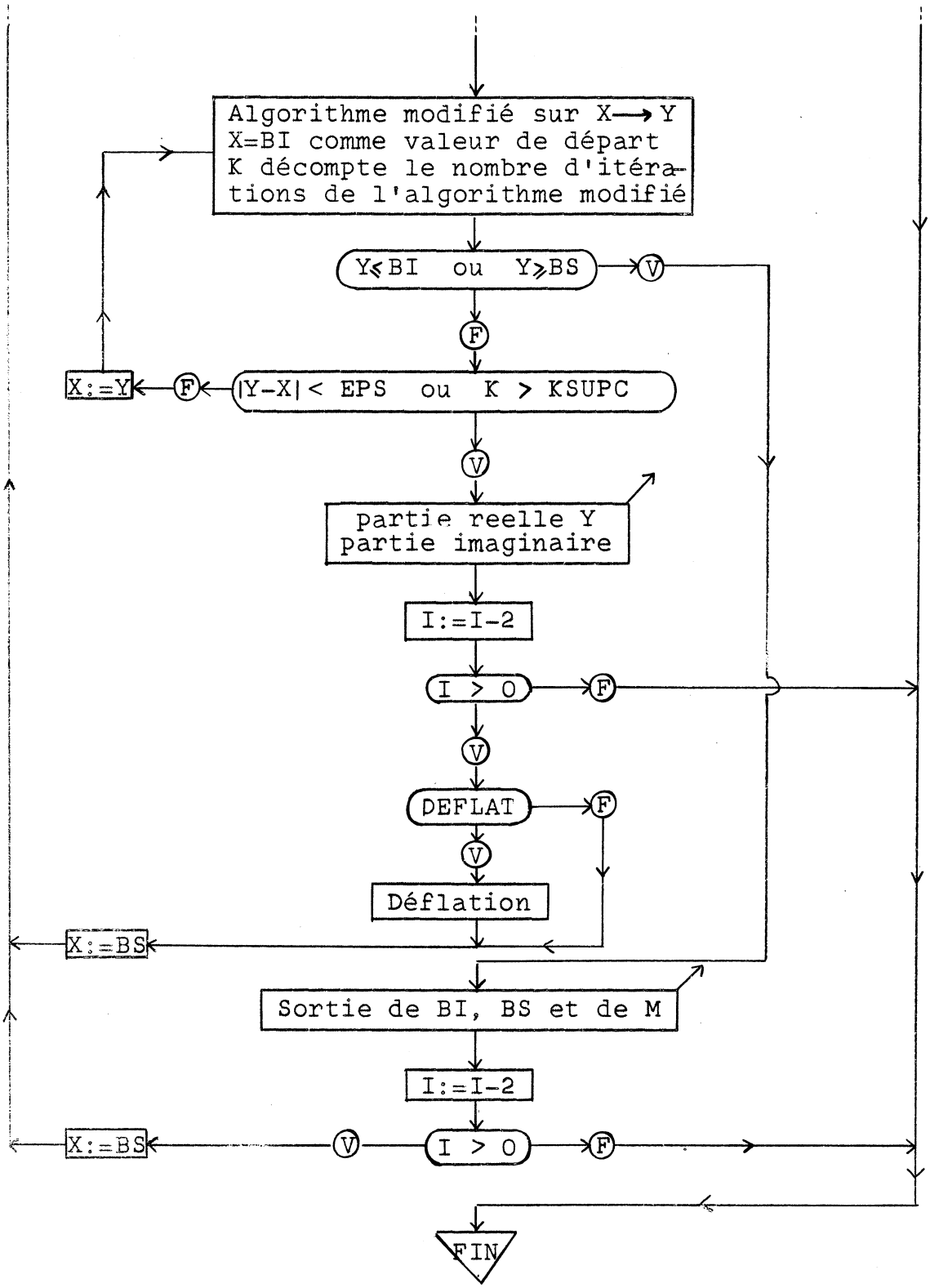
Les inconvénients de l'emploi de la déflation se retrouvent dans toutes les méthodes qui l'utilisent, à savoir que les erreurs commises sur le calcul des racines se répercutent sur les racines suivantes et que les polynômes successifs sur lesquels on travaille risquent de voir leurs racines s'éloigner de plus en plus des racines du polynôme initial.

b) Option non déflation.

DEFLAT prend la valeur faux. On n'effectue aucune déflation. Pour le reste, l'algorithme est le même.

Voici l'organigramme simplifié de cette procédure ROUTH NEWTON.


Calcul de SUP par la reel procedure MAXMOD
AMPL:=SUP;X=-SUP;I:=N;
Obtention de BI,BS à moins de EPS de la partie réelle et de M (nombre de racines dans la bande BI,BS) par l'algorithme normal partant de X
M = 1
F
V
racine réelle obtenue par la méthode de Newton avec BI comme valeur de départ
I:=I-1
I > 0
F
V
DEFLAT
F
V
Déflation
X:=BS
M ≠ 2
F
V
Sortie de BI, BS et de M
I:=I-M
X:=BS
V
I > 0
F


Algorithme modifié sur X⟶Y
X=BI comme valeur de départ
K décompte le nombre d'itérations de l'algorithme modifié
Y≤BI ou Y≥BS
V
F
X:=Y
F
|Y-X| < EPS ou K > KSUPC
V
partie reelle Y
partie imaginaire
I:=I-2
I > 0
F
V
DEFLAT
F
V
Déflation
X:=BS
Sortie de BI, BS et de M
I:=I-2
X:=BS
V
I > 0
F
FIN

Décrivons maintenant les différents paramètres de ROUTH NEWTON.

Les paramètres d'entrée sont :

EPS, N et DEFLAT cités précédemment.

C tableau réel contenant les coefficients de l'équation rangés par ordre de puissances décroissantes depuis C [1] jusqu'à C[N+1]. Le tableau C n'est pas altéré par l'emploi de la procédure.

EPSR et EPSC précision sur les racines réelles et les parties réelles des racines complexes.

KSUPR nombre d'itérations à ne pas dépasser dans l'amélioration des racines réelles depuis EPS jusqu'à EPSR.

KSUPC nombre d'itérations maximum de l'algorithme modifié. D'après ce qu'on a vu, la valeur de KSUPC est au moins égale à la partie entière de K/2, K étant le plus petit entier tel que $2^K \geqslant$ EPS/EPSC.

Les paramètres de sortie sont :

l'entier K1 ; l'entier tableau T et le tableau RAC. Pour ces 2 tableaux à une dimension, l'indice varie depuis 1 jusqu'à 2N.

Le 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 inférieure puis

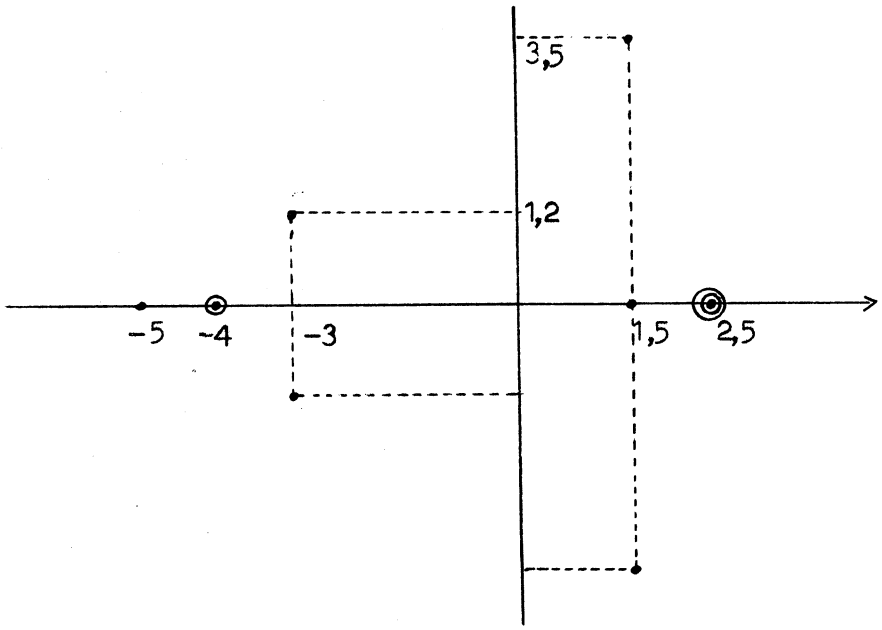
la borne supérieure de la bande.

Le 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 une racine réelle, de l'algorithme modifié pour 2 racines complexes.
- pour une localisation le nombre de parties imaginaires puis 0. C'est ce dernier zéro qui indique formellement que la procédure n'a fourni qu'une localisation de la partie réelle.

Enfin K1 est le double du nombre de parties réelles calculées ou localisées.

Soit l'exemple, défavorable pour la procédure, du polynôme qui a les racines suivantes :



Les résultats seraient alors : $K_1 = 10$

partie réelle ou borne inférieure	partie imaginaire ou borne supérieure	nombre de parties imaginaires	nombre d'itérations
RAC [1] = -5	RAC [2] = 0	T [1] = 1	T [2] = 6
RAC [3] = -4	RAC [4] = 0	T [3] = 2	T [4] = 7
RAC [5] = -3	RAC [6] = 1,2	T [5] = 2	T [6] = 4
RAC [7] = 1,4	RAC [8] = 1,6	T [7] = 3	T [8] = 0
RAC [9] = 2,4	RAC [10] = 2,6	T [9] = 3	T [10] = 0

```
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 ;
       debut  tableau  B [1 : N] ;
              entier I ; reel  MAX ;
              pour I:=1 pas 1 jusqua N faire
              B[I]:= si  A[I+1] ≠ 0.0  alors
              ABS (A[I+1]/A[1])↑(1.0/I) sinon  0.0 ;
              MAX:=0.0 ;
              pour I:=1 pas 1 jusqua N faire
              si  B[I]>MAX  alors  MAX := B[I];
              MAXMOD := 2.0 x MAX
       fin MAXMOD ;
```

```
procedure TRANSLREEL (C,CT,N,ALFA) ;
tableau C, CT ; entier N ; reel ALFA ;
debut  entier H, I ;
       pour I:=1 pas 1 jusqua N+1 faire
       CT[I]:= C[I];
       pour H:=N+1 pas -1 jusqua 2 faire
       pour I:=2 pas 1 jusqua H faire
       CT[I]:= ALFA x CT[I-1] + CT[I]
fin  TRANSLREEL ;

reel procedure NEWTON (XO, FO, DFO) ; valeur FO, DFO ;
reel XO, FO, DFO ;
NEWTON := XO - FO/DFO ;

procedure DEFLAT1 (C,N,X) ;
tableau C ; entier N ; reel X ;
debut  entier  I,K ; K := N+1 ;
       pour I:=2 pas 1 jusqua K faire
       C[I]:= C[I]+ X x C[I-1]
fin DEFLAT1 ;

procedure DEFLAT2 (C,N,A,B) ; valeur B ;
tableau  C  ; entier N ; reel  A, B ;
debut  entier I,K ; reel  S,P ;
       S:=2.0 x A ;
       P:=A x A + B x B ; K:=N+1 ;
       C[2]:= C[2]+ S x C[1];
       pour I:=3 pas 1 jusqua K faire
       C[I]:= C[I]+ S x C[I-1] - P x C[I-2]
fin DEFLAT2 ;
```

```
procedure Horner (X,C,F,DF,N) ;
          reel X,F,DF ; tableau C ; entier N ;
          debut tableau D[1:N+1] ; entier I,H ;
                H:=N+1 ;
                pour I:=1 pas 1 jusqua H faire
                D[I]:=C[I];
                pour I:=2 pas 1 jusqua H faire
                D[I]:= D[I]+ X x D[I-1];
                F:=D[H]; H:=H-1 ;
                pour I:=2 pas 1 jusqua H faire
                D[I]:=D[I]+ X x D[I-1] ;
                DF := D[H]
          fin HORNER ;
```

```
procedure SCHEMA (T1, C, N, P, V) ;
tableau T1, C ; entier N, P, V ;
debut  entier I, H ; reel  R ;
       procedure calcul (M,K) ; valeur M ;
       entier M, K ;
       debut reel S ; entier J ;
             S:=T1[M-2,1]/T1[M-1,1] ;
             pour  J:=1 pas 1 jusqua K faire
             T1[M,J]:=T1[M-2,J+1]-S x T1[M-1,J+1]
       fin calcul ;

       pour I:=1 pas 1 jusqua P faire T1[1,I]:=C[2xI-1];
       H:=si N ÷ 2 x 2 = N  alors P-1 sinon P ;
       pour I:=1 pas 1 jusqua H faire
       T1[2,I]:= C[2xI] ;
       H:=P-1 ;
       si N ÷ 2 x 2 = N alors
```

```
        debut R:=T1[1,P];
              pour I:=3 pas 2 jusqua N-1 faire
              debut T1[I,H]:= R ;
                    H := H-1 ;
                    calcul (I, H) ;
                    calcul (I+1, H)
              fin ;
              T1[N+1,1]:= R
        fin  sinon
        debut R := T1[2,P];
              pour I:= 3 pas 2 jusqua N faire
              debut calcul (I, H) ;
                    T1[I+1,H]:= R ;
                    si I = N  alors  allera  SORT ;
                    H := H-1 ; calcul (I+1, H )
              fin ; SORT :
        fin ;

        V := 0 ;
        pour I := 1 pas 1 jusqua N faire
        si ABS (SIGNE (T1[I,1]) - SIGNE (T1[I+1,1]))= 2
        alors V := V+1
fin SCHEMA ;

procedure DEUXSCHEMAS (C,T1,T2,N,P) ;
tableau C, T1, T2 ;
entier N, P ;
```

```
debut  entier I,H,J ; reel R1, R2 ;
       procedure CALCUL (M,K) ; valeur M ;
       entier M,K ;
       debut  reel  S1, S2 ; entier J ;
              S1:=T1[M-2,1]/T1[M-1,1];
              S2:=(T2[M-2,1] -S1 x T2[M-1,1])/T1[M-1,1] ;
              pour J:=1 pas 1 jusqua K faire
              debut T1[M,J]:= T1[M-2,J+1]-S1xT1[M-1,J+1];
                    T2[M,J]:= T2[M-2,J+1]-S2xT1[M-1,J+1]
                    -S1 x T2[M-1,J+1]
              fin
       fin calcul ;
       pour I:=1 pas 1 jusqua P faire T1[1,I]:=C[2xI-1];
       H:=si N÷2 x 2 = N alors P-1 sinon P ; J:=N ;
       pour I:=1 pas 1 jusqua H faire
       debut T1[2,I]:=C[2xI];
             T2[2,I]:=T1[1,I]x J ; J:=J-2
       fin ;
       J:=N-1 ;
       pour I:=2 pas 1 jusqua P faire
       debut T2[1,I]:= T1[2,I-1]x J ;
             J:=J-2
       fin ;
       T2[1,1]:= 0.0 ;
       H:=P-1 ;
```

```
        si N÷2 x 2 = N alors
        debut   R1 := T1[1,P];
                R2 := T2[1,P];
                pour I := 3 pas 2 jusqua N-1 faire
                debut   T1[I,H]:= R1 ;
                        T2[I,H]:= R2 ;
                        H := H-1 ;
                        Calcul (I,H) ;
                        Calcul (I+1,H)
                fin
        fin sinon
        debut   R1 := T1[2,P];
                R2 := T2[2,P];
                pour I := 3 pas 2 jusqua N faire
                debut   Calcul (I,H) ;
                        si I=N alors allera SORT ;
                        T1[I+1,H]:= R1 ;
                        T2[I+1,H]:= R2 ;
                        H := H-1 ;
                        Calcul (I+1,H)
                fin ; SORT :
        fin
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 ;
```

2°) Résultats numériques.

Cette série d'essais est destinée à comparer les temps de calcul de l'algorithme modifié et de l'algorithme normal. Nous avons toujours constaté avec l'un ou l'autre la même précision sur les racines. Les coefficients sont donnés par ordre de puissances décroissantes, les résultats de la procédure Routh Newton en 4 colonnes.

première colonne : partie réelle ou borne inférieure
deuxième colonne : module des parties imaginaires ou borne supérieure
troisième colonne : nombre de racines par partie réelle
quatrième colonne : nombre d'itérations

Nous donnons aussi les résultats de la procédure normale, les calculs étant effectués à la même précision EPS.

Essai 1

Polynôme de degré 10

Coefficients	Racines exactes
1	
-10	$-5 \pm 2i$
26	
-296	$-3 \pm 7i$
3 430	
6 372	$\pm 5i$
-85 892	
-181 816	$6 \pm i$
-230 215	
-9 246 650	$7 \pm 4i$
101 130 250	

Résultats algorithme normal sans déflation

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 algorithme modifié sans déflation

EPS = 10^{-1} EPSC = 10^{-7}

- 5,0000000	2,0000000	2	1
- 2,9999999	6,9999959	2	3
$- 0,58605494_{10}-7$	5,0000013	2	5
6,0000000	1,0000000	2	4
6,9999999	4,0000027	2	6

temps de calcul : 3,8 secondes

EPS = 10^{-3}	EPSC = 10^{-7}		
- 5,0000000	2,0000000	2	3
- 2,9999999	7,0000032	2	3
$0,95662456_{10}-9$	4,9999999	2	3
6,0000000	1,0000000	2	1
6,9999999	3,9999985	2	4

temps de calcul : 6 secondes

Résultats algorithme normal avec déflation

EPSC = 10^{-7}

parties réelles	modules des parties imaginaires
- 4,9999999	2,0000000
- 2,9999999	6,9999992
$0,30885530\ 10^{-6}$	5,0000027
5,9999972	0,99997249
7,0000021	4,0000039

temps de calcul : 4,4 secondes

Résultats algorithme modifié avec déflation

EPS = 10^{-1}	EPSC = 10^{-7}		
- 5,0000000	2,0000000	2	1
- 2,9999999	6,9999959	2	3
$- 0,18928195\ 10^{-4}$	5,0000094	2	6
6,0000324	1,0001191	2	4
6,9999855	3,9999903	2	3

temps de calcul : 2,4 secondes.

Essai 2

Polynôme de degré 8

Coefficients	Racines exactes
1	
- 6	$-3 \pm 2i$
18	
- 134	$-1 \pm 5i$
285	
416	$2 \pm 3i$
1 352	
- 17 576	$5 \pm i$
114 244	

Résultats algorithme normal sans déflation

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

parties réelles	modules des parties imaginaires
- 3,0000000	2,0000000
- 0,99999999	5,0000011
2,0000000	3,0000000
4,9999999	1,0000000

Temps de calcul : 4,8 secondes

Résultats algorithme modifié sans déflation

EPS = 10^{-2}	EPSC = $5\ 10^{-8}$		
- 3,0000000	2,0000000	2	3
- 0,9999996	5,0000018	2	3
2,00000000	3,0000001	2	3
5,0000000	1,0000000	2	3

Temps de calcul : 2,6 secondes

EPS = 10^{-4}	EPSC = 10^{-7}		
- 3,0000000	2,0000000	2	2
- 0,9999999	5,0000000	2	2
2,0000000	3,0000000	2	2
5,0000000	1,0000000	2	2

Temps de calcul : 4,5 secondes
L'algorithme modifié est peu utilisé le temps de calcul se rapproche de celui de l'algorithme normal.

Résultats algorithme normal avec déflation

EPSC = $5\ 10^{-8}$ parties réelles	modules des parties imaginaires
- 3,0000000	2,0000000
- 0,99999993	5,0000001
2,0000000	2,9999997
4,9999998	1,0000004

Temps de calcul : 2,7 secondes

Résultats algorithme modifié avec déflation

EPS = 10^{-2}	EPSC = $5\ 10^{-8}$		
- 3,0000000	2,0000000	2	3
- 0,99999996	5,0000018	2	3
2,0000037	3,0000006	2	3
4,9999992	0,99999793	2	3

temps de calcul : 1,9 secondes.

Essai 3

Polynôme de degré 18

Coefficients	Racines exactes
1	
-30	$-5 \pm 2i$
426	
-4 016	$-3 \pm 7i$
31 258	
-211 788	$\pm 5i$
1 091 076	
-3 232 424	$1 \pm i$
-973 359	
44 118 466	$2 \pm 2i$
51 368 970	
-3 355 235 304	$3 \pm 3i$
30 625 943 540	
-164 037 000 248	$4 \pm 4i$
585 915 735 528	
-1 403 841 908 256	$6 \pm i$
2 198 019 018 560	
-2 026 917 926 400	$7 \pm 4i$
932 016 384 000	

Résultats algorithme normal sans déflation

EPSC = 10^{-7}

parties réelles	modules de parties imaginaires
-4,9999997	2,0000002
-2,9999998	6,7625210
$0,16029059_{10-6}$	4,9982066
1,0000000	1,0000000
2,0000011	2,0000007
3,0000025	2,9999623
3,9999989	3,9998201
5,9999989	0,99999797
6,9999980	4,0006645

temps de calcul : 42,5 secondes

Résultats algorithme modifié sans déflation.

EPS = 10^{-4}	EPSC = 10^{-7}		
-4,9999999	2,0000000	2	3
-3,0000001	7,0164713	2	4
$0,61297515_{10-7}$	5,0007093	2	5
0,9999990	1,0000001	2	2
2,0000010	2,0000008	2	3
3,0000023	2,9999980	2	7
3,9999988	4,0000372	2	4
5,9999975	1,0000010	2	7
6,9999995	3,9999869	2	7

temps de calcul : 36,7 secondes. On a dû prendre EPS très petit car pour les parties réelles -3 et 0 l'algorithme modifié condui-

sait vers -5 et 1 respectivement.

Résultats algorithmes normal et modifié avec déflation.

A partir de la 3ème partie réelle calculée les résultats sont complètement faussés, à cause de la 2ème partie imaginaire trop imprécise.

Essai 4

Polynôme de degré 8

Coefficients	Racines exactes
1	
-4	$-4 \pm 3i$
40	
-20	$1 \pm 6i$
294	
-1 196	$2 \pm i$
23 720	
-89 500	$3 \pm 4i$
115 625	

Résultats algorithme normal.

EPSC = 10^{-7}

parties réelles	module des parties imaginaires
-3,9999999	3,0000001
0,9999993	5,9999566
1,9999999	1,0000001
2,9999999	4,0000004

temps de calcul : 4,6 secondes

Résultats algorithme modifié sans déflation.

EPS = 10^{-2}	EPSC = 10^{-7}		
-4,0000000	3,0000001	2	3
0,99999999	6,0000017	2	12
2,0000000	1,0000000	2	3
2,9999998	4,0000007	2	3

temps de calcul : 3 secondes

Résultats algorithme normal avec déflation.

EPSC = 10^{-7}

parties réelles	modules des parties imaginaires
-3,9999999	3,0000001
0,99999957	5,9999741
2,0000525	0,99988368
2,9999478	4,0000547

temps de calcul : 2,7 secondes

Résultats algorithme modifié avec déflation.

EPS = 10^{-2}	EPSC = 10^{-7}		
-4,0000000	3,0000001	2	3
0,99999979	6,0000064	2	5
1,9999873	1,0000235	2	3
3,0000129	3,9999879	2	2

temps de calcul : 1,5 secondes.

Essai 5

Polynôme de degré 6

Coefficients	Racines exactes
1	
34	$-30 \pm 10i$
549	
14 720	$-2 \pm 30i$
-146 400	
-8 192 000	$15 \pm 10i$
293 800 000	

Résultats algorithme normal (poussé jusqu'à une précision de 10^{-6})

$EPSC = 10^{-6}$

parties réelles	module des parties imaginaires
-30.000000	10.000000
-2.0000005	30.000001
14.999999	10.000000

temps de calcul : 2,3 secondes

Résultats algorithme modifié sans déflation.

EPS = 2.0	EPSC = 10^{-6}		
-30.000000	10.000000	2	4
-1.9999998	30.000000	2	8
15.000000	10.000000	2	4

temps de calcul : 1,1 secondes

Résultats algorithme normal avec déflation.

EPSC = 10^{-6}

parties réelles	modules des parties imaginaires
-30,000000	10,000000
-2,0000005	29,999998
15,000001	10,000007

temps de calcul : 1,4 secondes

Résultats algorithme modifié avec déflation.

EPS = 2.0 EPSC = 10^{-6}

-30,000000	10,000000	2	4
-2,0000001	30,000000	2	6
15,000000	9,9999999	2	2

temps de calcul : 0,6 secondes

Essai 6

Polynôme de degré 6

Coefficients	Racines avec 8 chiffres exact
1	
-8	-2,2209395 $\pm$ 5,0143688i
56	
-336	1,5863875 $\pm$ 4,8397364i
1 680	
-6 720	4,6345518 $\pm$ 2,0883787i
20 160	

Résultats algorithme normal

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

partie réelle	module des parties imaginaires
-2.2209394	5.0143688
1.5863875	4.8397365
4.6345519	2.0883787

temps de calcul : 2,5 secondes

Résultats algorithme modifié sans déflation.

EPS = 10^{-1}	EPSC = $5\ 10^{-8}$		
-2.2209394	5.0143687	2	5
1.5863875	4.8397363	2	4
4.6345519	2.0883787	2	4

temps de calcul : 1,2 secondes

Résultats algorithme normal avec déflation.

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

parties réelles	modules des parties imaginaires
-2,2209394	5,0143688
1,5863877	4,8397365
4,6345515	2,0883784

temps de calcul : 1,6 secondes

Résultats algorithme modifié avec déflation

EPS = 10^{-1}	EPSC = $5\ 10^{-8}$		
-2,2209394	5,0143687	2	5
1,5863873	4,8397367	2	4
4,6345521	2,0883787	2	2

temps de calcul : 0,7 secondes.

3°) Conclusion.

L'examen de ces différents essais nous laisse espérer en l'algorithme modifié. La précision semble la même pour les deux algorithmes, il ne fallait d'ailleurs pas s'attendre à autre chose et le temps de calcul est généralement divisé par 2 au moins. La déflation ne semble pas extrêmement heureuse et l'on peut vérifier que l'emploi de l'algorithme modifié sans déflation donne sur ces exemples des temps de calcul encore inférieurs à ceux de l'algorithme normal utilisant la déflation, pour une précision bien supérieure. Plus l'algorithme modifié débute loin de la partie réelle, plus la diminution du temps de calcul semble importante ; ceci confirme notre opinion. Mais justement, le problème se pose de savoir à partir de quel EPS maximum "enclencher" l'algorithme modifié. Il semble difficile de répondre à cette question : si on prend EPS trop petit, l'algorithme modifié ne sert plus à rien, si on le prend trop grand, on risque de se diriger vers une autre partie réelle. Pratiquement, pour des racines à parties réelles suffisamment séparées, comme celles que nous avons étudiées, EPS = 10^{-1} ou 10^{-2} semble être une bonne valeur. Une méthode consisterait à employer systématiquement la déflation et à "enclencher" l'algorithme modifié à partir d'un point situé à gauche de la partie réelle (dans un balayage gauche droite) ; la partie réelle recherchée étant la plus à gauche, il n'y aurait pas de risques de se diriger vers une autre. Ceci a semblé assez théorique car d'une part la déflation augmente les erreurs, d'autre part, dans certains essais effectués de la sorte, nous avons enregistré malgré tout des cas de non-

convergence qui reposaient le problème de la grandeur de EPS.

On voit donc ici les limites de l'emploi de l'algorithme modifié : pour des racines à parties réelles trop voisines il ne pourra se subtituer à l'algorithme normal.

Bien qu'extérieure à ce travail qui se limitait à l'étude de l'algorithme de Routh, l'idée vient nécessairement d'effectuer des comparaisons avec les méthodes existantes, principalement avec la méthode de Bairstow. Cette dernière semble beaucoup plus rapide (temps de calcul 2 à 3 fois ou 3 à 5 fois moins grand selon qu'on emploie ou non la déflation) et d'une précision sensiblement supérieure, même si la déflation n'est pas employée. Est-ce à dire que la méthode de Bairstow range tous les avantages de son côté ? D'après les études expérimentales faites par Messieurs Lassert et Paquet [7] au Laboratoire de Calcul de Grenoble, la convergence de la méthode de Bairstow vers un couple de racines ne semble pas assurée, selon la choix des valeurs de départ. Au contraire, nous n'avons ici, aucun cas de divergence. Nous conclurons sur cette remarque optimiste et nous soulignerons encore le fait que nous avons voulu, plutôt qu'établir une méthode de résolution complète, qui aurait nécessité un travail de programmation plus important, vérifier et mettre en pratique l'idée de l'algorithme modifié dans le cas, cependant général, des racines simples.

BIBLIOGRAPHIE PREMIERE PARTIE

[1] F. RIESZ - Les systèmes d'équations linéaires à une infinité d'inconnues. Collection de monographies sur la théorie des fonctions publiée sous la direction de E. Borel. Paris Gauthiers-Villars (1952). p.9-10-11.

[2] FÜRSTENAU - Ouvrage cité par F. RIESZ

[3] NAEGELSBACH - Ouvrage cité par F. RIESZ

[4] E. DURAND - Solutions numériques des équations algébriques. Tome 1. Masson et Cie éditeurs (1960).

[5] P. HENRICI - The Quotient-Difference Algorithm. Further contributions to the solution of simultaneous linear equations and the determination of eigenvalue. US Department of Commerce. National Bureau of Standards Applied Mathematics Series. 49 (January 15, 1958). p. 23 à 36.

[6] M. GOLLOMB - Ouvrage cité par P. HENRICI

[7] KÖNIG - Ouvrage cité par P. HENRICI

[8] A.C. AITKEN - Ouvrage cité par P. HENRICI

[9] J.H. WILKINSON - Rounding errors in algebraic processes. National Physical Laboratory. Notes on Applied Science N° 32.

BIBLIOGRAPHIE DEUXIEME PARTIE

[1] F.R. GANTMACHER - The theory of matrices (traduit du Russe). Volume two. Chelsea Publishing Company New York.

[2] E. APARO - Un Criterio di Routh e sua applicazione al calcolo degli zeri di un polinomio. Consiglio Nazionale delle Ricerche Publicazioni dell' instituto per le applicazioni del calcolo. N. 536. Roma 1958 .

[3] E. DURAND - Solutions numériques des équations algébriques. Tome 1. Masson et Cie éditeurs (1960).

[4] J. PELTIER - Résolution numérique des équations algébriques. Volume III. Gauthier-Villars éditeur (1957)

[5] J.H. WILKINSON - Rounding errors in algebraic processes. National Physical Laboratory. Notes on Applied Science N° 32.

[6] Д.К. ФАДДЕЕВ и И.С. СОМИНСКИЙ - Сборник задач по высшей алгебре. Editions d'Etat. Moscou 1956.

[7] M. LASSERT, J. PACQUET - Résolution d'équations algébriques à coefficients réels. Institut de Mathématiques Appliquées Grenoble.

TABLE DES MATIERES

PREMIERE PARTIE

APPLICATION DE LA METHODE DES REDUITES A LA RESOLUTION DES EQUATIONS ALGEBRIQUES

-:-:-:-:-

DEUXIEME PARTIE

ETUDE DE L'EMPLOI DE L'ALGORITHME DE ROUTH POUR RESOUDRE LES EQUATIONS ALGEBRIQUES.

-:-:-:-:-

VU,

Grenoble, le

Le Président de la Thèse

VU,

Grenoble, le

Le Doyen de la Faculté des Sciences

VU et permis d'imprimer,

Le Recteur de l'Académie de Grenoble