Algorithmique du theoreme fondamental de l'algebre

Xavier Gourdon

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Algorithmique du théorème fondamental de l'algèbre

Xavier GOURDON

N° 1852
Février 1993

PROGRAMME 2

Calcul Symbolique,
Programmation
et Génie logiciel

Algorithmic Aspects of the Fundamental Theorem of Algebra

Xavier Gourdon
Projet Algorithmes,
INRIA Rocquencourt,
78153 Le Chesnay Cedex
France

Abstract

Schönhage has shown that approximating complex roots of a complex polynomial has polynomial complexity. We describe several optimizations of his algorithm making it efficiently implementable in computer algebra. Tests show that the program, written in MAPLE, routinely solve polynomials of degree 100 and behaves well even on ill conditionned polynomials.

Algorithmique du Théorème Fondamental de l'Algèbre

Résumé

Schönhage a montré que le calcul des racines complexes d'un polynôme complexe a une complexité polynomiale. Ce rapport décrit plusieurs optimisations de son algorithme le rendant implémentable efficacement en calcul formel. Des tests montrent que le programme, écrit en MAPLE, traîte de manière routinère des polynômes de degré 100 et a un bon comportement même vis à vis de polynômes mal conditionnés.

Table des matières

Introduction

La plupart (sinon tous) des logiciels de calcul formel possèdent une commande de recherche des racines d'un polynôme. Sur MAPLE V par exemple, l'utilisateur qui veut les racines complexes du polynôme $P = X^6 - X + 1$ n'a qu'à taper :

```
>   fsolve(X^6-X+1,X,complex);
```

Il verra rapidement apparaître à l'écran la liste des racines :

```
    - .9454023333 + .6118366938 I,  - .9454023333 - .6118366938 I,
      .1547351445 + 1.038380754 I, .1547351445 - 1.038380754 I,
      .7906671888 - .3005069203 I, .7906671888 + .3005069203 I
```

Encouragés par ce résultat, on lance un nouveau test. Les polynômes à racines multiples étant mal conditionnés, curieux de savoir comment réagira MAPLE V, on calcule :

```
> p:=expand((X-1.)^4*(X^2+X+1));
```

$$p := X^6 - 3.X^5 + 3.X^4 - 2.X^3 + 3.X^2 - 3.X + 1.$$

puis on lance :

```
> fsolve(p,X,complex);
```

MAPLE V vous répond

```
Error, (in fsolve/polyquad)
unable to find solution, probably not enough precision
```

MAPLE V nous dit qu'il n'a pas assez de précision. Un premier problème apparaît donc. D'autres tests sur le même polynôme montrent que même en augmentant indéfiniment la valeur de Digits, (qui sur MAPLE V représente le nombre de chiffres avec lequel sont effectuées les opérations; Digits=10 par défaut) MAPLE V renvoie toujours le même message d'erreur. Ne soyons quand même pas pessimistes. Sur beaucoup d'autres polynômes, la fonction *fsolve* de MAPLE V trouve les racines [1] (il apparaît quand même que lorsque le polynôme est de degré trop élevé — variable selon le type de polynôme — la fonction *fsolve* échoue). Ce message d'erreur est en fait dû à une erreur de programmation de l'algorithme de recherche des racines, qui ici est l'algorithme de Traub et Jenkins [9]. Bruno Salvy a programmé correctement cet algorithme en MAPLE V, mais si sa version tourne bien sur cet exemple, elle est très gourmande en mémoire et sur certains polynômes de degré à peine 25, 56 Méga-octets de mémoire ne suffisent pas (voir la section 13).

Voyons maintenant ce qu'il se passe avec MATHEMATICA :

[1] Une série de tests beaucoup plus complète fait l'objet de la section 13.

```
In[1]:= p:=Expand[(X-1.)^4*(X^2+X+1)]
In[2]:= NRoots[p==0,X]
Out[2]= X == -0.5 - 0.866025 I || X == -0.5 + 0.866025 I || X == 0.999889 ||
>     X == 1. || X == 1.00006 || X == 1.00006
```

Les racines sont cette fois ci déterminées, avec néanmoins une imprécision assez importante sur la racine 1 d'ordre 4. Ceci est toutefois compréhensible, les racines multiples étant très sensibles aux perturbations (voir la section 12). Sur d'autres exemples même pathologiques, MATHEMATICA trouve les racines. Par contre, MATHEMATICA renvoie parfois des résultats surprenants. Par exemple, sur le polynôme

```
In[3]:= p :=Expand[(X^10-1)*((X-1)^10-10^(-20))]
```

dont les racines sont les $\exp(2ik\pi/10)$ et les $1+\exp(2ik\pi/10)/100$, si on fait

```
In[4]:= NRoots[p==0,X,100]
```

(*i.e.* on lui demande 100 chiffres de précision sur le résultat) MATHEMATICA renvoie parmi sa liste de racines les 4 suivantes :

```
>     X == 1.0030901699437494742410229341718281905891 ||
>     X == 1.0030901699437494742410229341718281905891 ||
>     X == 1.0080901699437494742410229341718281905886 ||
>     X == 1.0080901699437494742410229341718281905886 ||
```

Ces racines sont purement réelles et ne correspondent pas à des racines de p dont toutes (sauf 1, 0.99 et 1.01) ont des parties imaginaires non nulles. De plus, elles sont doubles et p n'a que des racines simples. Même si on sait que des racines proches sont très sensibles aux perturbations, une telle imprécision sur les racines a de quoi laisser perplexe l'utilisateur qui aurait pu espérer obtenir une bonne précision en demandant 100 chiffres au départ. Au moins, MATHEMATICA pourrait-il renvoyer avec chaque racine une borne d'erreur sur celle-ci pour éviter de tromper l'utilisateur. MAPLE V ne fait d'ailleurs pas mieux puisque même avec Digits=500, il renvoie un message d'erreur. Notons que l'algorithme utilisé par MATHEMATICA est aussi celui de Traub et Jenkins.

Citons également le logiciel PARI qui pour trouver les racines utilise une méthode de Newton améliorée, et qui est mis en échec pour des polynômes assez faciles.

La nécessité de l'implantation d'un programme de recherche de racines qui évite ces problèmes se fait donc sentir. La méthode que je me propose de mettre en place répond à cette attente [2] : Un polynôme P de degré $n > 1$ et un paramètre de précision s étant entrés, le programme détermine *dans tous les cas* n complexes $v_1, \ldots, v_n$ tels que

$$|P - \beta(X - v_1)\cdots(X - v_n)| < 10^{-s}|P|$$

(où β désigne le coefficient dominant de P et $|P|$ la somme des modules des coefficients de P). En option, il renvoie également avec chaque racine une borne d'erreur sur cette dernière.

Cette méthode repose essentiellement sur un algorithme dû à Schönhage [20]. Néanmoins, l'algorithme de Schönhage est surtout un résultat théorique et il s'agissait de le rendre applicable dans la pratique. Les sections 1 à 12 exposent de façon détaillée sa méthode et les optimisations que j'en ai faites. Enfin, la section 13 compare cette méthode avec celles déjà existantes sur plusieurs familles de polynômes.

Sauf mention explicite, tous les polynômes considérés dans la suite sont à coefficients complexes.

[2] Elle donne des résultats tout à fait satisfaisants sur les exemples cités plus haut.

Chapitre 1

Présentation générale

Pendant plusieurs siècles, le problème de la détermination des racines d'un polynôme a surtout été étudié en termes de formules. C'est au début du XVI^{ème} siècle que Cardan donna les formules pour le troisième degré puis Ferrari pour le quatrième. Pendant plus de 250 ans, les mathématiciens ont en vain tenté de trouver des formules analogues pour le degré 5. Ce n'est qu'au début du XIX^{ème} siècle, qu'Abel et Galois démontrèrent l'impossibilité de résoudre par radicaux les équations de degré supérieur ou égal à 5. On savait pourtant que tout polynôme à coefficients complexes de degré n a exactement n racines.

C'est le théorème fondamental de l'algèbre, dont les premières démonstrations constructivistes n'apparurent qu'un siècle plus tard et, avec elles, l'élaboration d'algorithmes.

Actuellement, il existe plusieurs algorithmes approchant les racines d'un polynôme. Cependant, il n'en existe pas un vraiment meilleur que les autres : chacun a ses propres avantages et inconvénients. Plusieurs critères peuvent les caractériser (cf. [8] vol.1, §6.9) :

1. **La convergence globale :** Plusieurs algorithmes n'assurent la convergence que si on leur donne une approximation initiale z_0 suffisamment proche d'un zéro du polynôme (c'est le cas par exemple de la méthode de Newton). Ces algorithmes sont dits *localement convergents* ; d'autres algorithmes n'imposent pas de conditions de ce type : ils sont *globalement convergents.*

2. **La convergence inconditionnelle :** Certains algorithmes n'assurent la détermination de zéros de polynômes que sous certaines conditions (par exemple, tous les zéros sont simples, ou ont des modules différents). Ils sont *convergents conditionnellement.* Ceux qui convergent (localement ou globalement) pour tous les polynômes sont dits *convergents inconditionnellement.*

3. **Le temps d'exécution .**

L'algorithme de Schönhage [20], sur lequel est fondé la méthode de recherche des racines que nous présentons, est un algorithme convergeant *globalement* et *inconditionnellement.* Il était intéressant de tester sa mise en œuvre pour deux raisons :

- Son temps d'exécution asymptotique est intéressant.
- Tous les algorithmes mis en œuvre jusqu'à présent échouent lorsque le degré du polynôme est trop élevé (variant de 25 à 100 selon l'algorithme et le polynôme).

1.1 Position du problème

Notation *On définit la norme* $|\cdot|$. *Si* $P = a_0 + a_1 X + \cdots + a_n X^n$, *on pose :*

$$|P| = |a_0| + |a_1| + \cdots + |a_n|,$$

Si tant d'algorithmes échouent, c'est que la recherche des racines d'un polynôme est un problème extrêmement mal conditionné. Le théorème de perturbation qui suit confirme ce fait.

Théorème (Ostrowski) *Soient P et Q deux polynômes unitaires de degré $n > 0$, avec $Q = (X - v_1) \cdots (X - v_n)$. Alors si $\epsilon = |P - Q|$, on peut numéroter les racines u_i de P telles que*

$$\forall i, \quad |u_i - v_i| < 4\rho\epsilon^{1/n}, \qquad \text{où } \rho = \sup\{1, |u_i|\}.$$

Autrement dit, les racines d'un polynôme bougent en $\epsilon^{1/n}$ lorsque ses coefficients bougent en ϵ (un exemple facile où le phénomène se produit est le cas de $P = X^n - \epsilon$ et $Q = X^n$). Ce résultat est en fait très pessimiste dans le cas général. De manière intuitive, lorsqu'un polynôme est perturbé à ϵ près, une racine u d'ordre p (i.e. p racines du polynôme sont proches de u, les autres sont éloignées de u) se déplace avec une erreur de l'ordre de $\epsilon^{1/p}$.

Ce théorème justifie le choix fait par Schönhage. Un polynôme P unitaire de degré $n > 1$ et un entier $s > 0$ étant donnés, son algorithme détermine n nombres complexes $u_1, \ldots, u_n$ tels que :

$$|P - (X - u_1) \cdots (X - u_n)| < 2^{-s} \tag{1.1}$$

en un temps d'exécution

$$O\left[(n^3 \log n + sn^2) \log(ns) \log(\log(ns))\right]. \tag{1.2}$$

C'est donc au sens de (1.1) que l'algorithme donne des approximations des racines de P. En généralisant le théorème donné plus haut, on peut alors donner *a posteriori* pour chaque racine trouvée une borne d'erreur proche de l'imprécision réelle (voir le chapitre 12). Quant à (1.2), ce résultat est surtout théorique. Il est obtenu en utilisant des méthodes de FFT pour le calcul de produit et de division de polynômes, et la constante devant (1.2) est énorme. Dans la pratique, c'est surtout le fait que les racines soient trouvées dans *tous les cas* qui est intéressant. Notons que la borne (1.2) a été améliorée par Victor Pan [16] mais son algorithme est désastreux dans la pratique.

1.2 Principe de la méthode de Schönhage

Soit P un polynôme à coefficients complexes de degré $n > 1$, $u_1, \ldots, u_n$ ses racines. L'idée est non pas de calculer directement une racine de P (comme le font en général les autres méthodes), mais de factoriser P en deux polynômes : $P = FG$. En réitérant ainsi le procédé sur F et G, on saura écrire P comme produit de facteurs du premier degré. L'algorithme détermine d'abord un cercle Γ contenant une partie des racines de P (voir figure 1a), par exemple $u_1, u_2, \ldots, u_k$ avec $1 \leq k < n$.

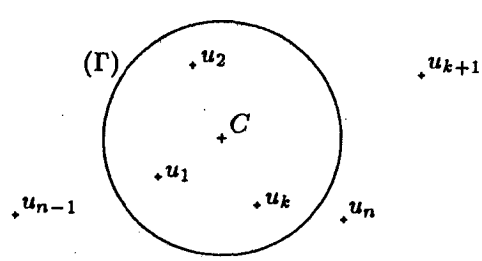


Figure 1a. Les racines de P et le cercle de séparation.

Par intégration le long de ce cercle (appelé cercle de séparation), on peut ensuite déterminer le polynôme unitaire dont les racines sont $u_1, u_2, \ldots, u_k$. On a en effet :

$$\forall p, \qquad S_p = u_1^p + u_2^p + \cdots + u_k^p = \oint_{(\Gamma)} \frac{P'(z)}{P(z)} z^p \, dz \tag{1.3}$$

d'où les coefficients de $F(z) = (z-u_1)\cdots(z-u_k)$ par les formules de Newton. Si G est le quotient de la division euclidienne de P par F, on a donc factorisé P en un produit de deux facteurs. Ceci dit, l'approximation numérique de (1.3) par discrétisation de Γ sera plus ou moins précise selon le choix de Γ. On sent bien que s'il se trouve près de Γ une racine de P, le calcul numérique de (1.3) sera mauvais. Dans cette optique, le cercle de séparation Γ sera choisi de sorte que les racines de P s'en éloignent le plus possible.

Nous présentons maintenant brièvement chacune des étapes de l'algorithme décrites plus haut.

1.3 Recherche du cercle de séparation

Notation *Si P est un polynôme complexe de degré $n > 0$, on note $r_1(P), \cdots, r_n(P)$ les modules de ses racines rangés dans l'ordre croissant.*

On veut trouver un cercle de séparation pour P, c'est à dire un cercle contenant une partie non vide de ses racines et ne les contenant pas toutes. Si $\Delta = \log(r_n(P)/r_1(P)) > 0$, il existe j tel que $\log(r_{j+1}(P)/r_j(P)) \geq \Delta/(n-1)$. On a alors trouvé un candidat pour notre cercle de séparation : le cercle de centre 0 de rayon $\sqrt{r_j \cdot r_{j+1}}$. Ceci n'est malheureusement possible que si $\Delta \neq 0$; de toutes façons, même si $\Delta \neq 0$ il se peut que Δ soit faible et alors notre cercle de séparation n'est pas intéressant puisqu'il est trop près des racines. Nous allons pallier ce problème en changeant d'origine.

1.3.1 Recherche du centre du cercle de séparation

On se ramène d'abord au cas où le barycentre des racines de P est l'origine (pour cela, si $P = X^n + a_{n-1}X^{n-1} + \cdots$, considérer $P(X - a_{n-1}/n)$). On se ramène ensuite au cas où le plus grand des rayons des racines de P est égal à 1 (voir le paragraphe suivant pour le calcul de $r_n(P)$). Le dessin suivant montre alors que si $v_0 = 2, v_1 = 2i, v_2 = -2, v_3 = -2i$, alors il existe j tel que $\Delta_j = \log(r_n(P_j)/r_1(P_j)) > 0.30$ (où $P_j = P(X + v_j)$). En calculant Δ_j pour $j = 0 \ldots 3$, puis en choisissant j maximisant Δ_j, on s'est ainsi ramené au cas où $\Delta > 0.30$.

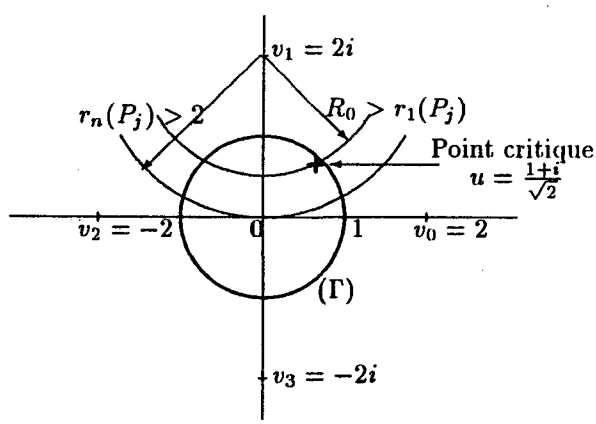


Figure 1b. Les centres v_i pour obtenir l'écartement $\Delta = \log(r_n/r_1) > 0.30$.

1.3.2 Recherche du rayon du cercle de séparation

Il nous faut savoir calculer les modules $r_i(P)$ des racines de P. Comme $\Delta = \log(r_n/r_1)$ n'est pas trop petit, on saura ensuite trouver un cercle de séparation dont les racines ne soient pas trop proches. Ceci est permis grâce à la méthode de Graeffe dont nous rappelons les grandes lignes.

L'idée est que les racines du polynôme Q défini par $Q(X^2) = P(X)P(-X)$ sont les carrés des racines de P. On note Q =Graeffe(P). On pose $P_0 = P$ puis P_m =Graeffe(P_{m-1}), de sorte que $r_k(P_m) = r_k(P)^{2^m}$, et

donc, si $r_1(P) < \cdots < r_n(P)$, en notant $P_m = a_n^{(m)} X^n + a_{n-1}^{(m)} X^{n-1} + \cdots$:

$$\lim_{m\to\infty} \left| \frac{a_{k-1}^{(m)}}{a_k^{(m)}} \right|^{2^{-m}} = r_k(P).$$

L'utilisation directe de ce procédé est difficile pour deux raisons :

- La convergence n'est assurée que si $r_1 < \cdots < r_n$. Par ailleurs, même si cette dernière condition est réalisée, la convergence peut être très lente (c'est le cas si pour un k, r_{k+1}/r_k est proche de 1).
- Les coefficients de P_m divergent très rapidement en module.

Schönhage contourne ces difficultés en utilisant une approche modifiée du problème.

1.4 Calcul d'un facteur à partir du cercle de séparation

Quitte à effectuer des transformations simples, on peut supposer que le cercle de séparation est le cercle unité. Sa recherche nous permet même de connaître le nombre k des racines se trouvant à l'intérieur du cercle et un paramètre $\delta > 0$ tel qu'aucune des racines de P ne se trouve dans la couronne $\{e^{-\delta} < |z| < e^{\delta}\}$.

Les coefficients du polynôme F dont les racines se trouvent à l'intérieur du cercle de séparation sont donnés par calcul de (1.3) suivi des formules de Newton. On approxime les valeurs S_p de (1.3) pour $1 \leq p \leq k$ par discrétisation du cercle unité en N points régulièrement répartis. On montre qu'après calculs, l'approximation F_0 de F vérifie

$$|F_0 - F| < |F| k n e^{-\delta(N-k)}.$$

C'est cette majoration qui permet de déterminer la valeur de N qu'il faut choisir pour être sûr que F_0 est déterminé avec suffisamment de précision. Seulement ce calcul peut devenir très coûteux si l'on désire avoir une bonne précision sur F_0. L'algorithme de Schönhage s'en sort comme suit. Il détermine d'abord par cette méthode une approximation F_0 de F suffisamment bonne puis il lance une méthode dite de Newton-Schönhage qui permet d'obtenir rapidement à partir de F_0 une bonne approximation de F.

1.4.1 La méthode de Newton-Schönhage

Hypothèses. $P = FG$ où les racines de F sont à l'intérieur du cercle unité, celles de G en dehors. On connaît une approximation F_0 de F, une approximation G_0 de G (déterminée par division euclidienne de P par F_0).
Problème. On veut trouver deux termes correcteurs f et g tels que $F_1 = F_0 + f$ et $G_1 = G_0 + g$ approchent F et G mieux que F_0 et G_0.

On a $F_1 G_1 = (F_0 + f)(G_0 + g) = F_0 G_0 + f G_0 + g F_0 + fg$. Le terme fg étant du second ordre, si on choisit f et g tels que

$$P = F_0 G_0 + f G_0 + g F_0 \quad \Longleftrightarrow \quad \frac{P - F_0 G_0}{F_0 G_0} = \frac{f}{F_0} + \frac{g}{G_0}, \tag{1.4}$$

on aura $|P - F_1 G_1| = |fg| \leq |f| \cdot |g|$, donc du second ordre. F_0 et G_0 sont premiers entre eux et f et g dans (1.4) sont donc uniques. On peut les déterminer par exemple grâce à l'algorithme d'Euclide. Sans rentrer dans les détails, disons que le problème est que cette méthode est peu stable numériquement et Schönhage procède autrement à partir d'une représentation intégrale de f, conséquence du théorème des résidus.

1.4.2 Description complète de la factorisation à partir du cercle de séparation

On se demande maintenant avec quelle précision on doit calculer F_0 pour assurer une convergence rapide de la méthode de Newton-Schönhage. On montre que le nombre de points N pris sur le cercle unité pour approximer F_0 doit vérifier :

$$N \geq k + \frac{1}{\delta}\left(\log(100) + \log(k^3(n-k)^{3/2}n) + \frac{5}{2}\log(\gamma) + \log\left(\frac{1}{\mu}\right) + 3\log(|F|\cdot|G|)\right),$$

où $\gamma = \frac{1}{2\pi}\oint |dt/P(t)|$ et $\mu = \inf_{|z|=1}|P(z)|$. Le calcul de γ et μ est réglé par approximations numériques. Il faut donner une majoration de $|F|\cdot|G|$. Schönhage utilise la majoration $|F|\cdot|G| < 2^n|P|$, donnée par Mignotte dans [14]. En fait, on peut donner une inégalité beaucoup plus fine qui est :

$$|F|\cdot|G| < 2^{n/2}\sqrt{\binom{n}{k}}\,[P]_2, \quad \text{où} \quad [P]_2 = \left(\sum_{i=0}^{n}\frac{|a_i|^2}{\binom{n}{i}}\right)^{1/2}.$$

Cette inégalité est une conséquence d'un résultat démontré dans [2].

Cette étape d'intégration numérique est la plus coûteuse de l'algorithme de Schönhage. Son coût est déjà considérablement diminué en utilisant une FFT partielle. Il est également important de diminuer le plus possible la valeur de N. Celle donnée plus haut améliore déjà celle donnée par Schönhage lui-même dans [20]. On peut encore l'améliorer en augmentant la valeur de δ, ce qui est rendu possible par une transformation homographique préalable : $z \mapsto (z-a)/(\bar{a}z-1)$.

Chapitre 2

Quelques résultats mathématiques

Cette section met en place le décor mathématique dont nous aurons besoin pour traiter correctement l'étude de l'algorithme de Schönhage. Les définitions étant lues, elle peut être sautée en première lecture.

2.1 Définitions

2.1.1 Normes

$\mathbb{C}[X]$ désigne l'espace vectoriel des polynômes à coefficients dans $\mathbb{C}$.

Définition 1 *Si $P = a_0 + a_1 X + \cdots + a_n X^n \in \mathbb{C}[X]$, on définit les normes suivantes ($|z|$ désigne le module du nombre complexe z) :*

$|P| = |a_0| + |a_1| + \cdots + |a_n|$, $\|P\| = \left(|a_0|^2 + |a_1|^2 + \cdots + |a_n|^2\right)^{1/2}$, $|P|_{\mathbb{T}} = \sup\{|P(z)|, \quad |z| = 1\}$ *et*

$$[P]_2 = \left(\sum_{k=0}^{n} \frac{|a_k|^2}{\binom{n}{k}} \right)^{1/2}.$$

La norme $[.]_2$ a été introduite par Bombieri [1], et elle permettra, comme nous allons le voir, de raffiner les majorations de Schönhage de manière non négligeable.

Nous nous servirons des inégalités suivantes.

Proposition 1 *Soit $P \in \mathbb{C}[X]$ de degré n. On a*

$$\|P\| \leq |P|_{\mathbb{T}} \leq |P| \leq \sqrt{n+1} \cdot \|P\|, \tag{2.1}$$

$$|P| \leq 2^{n/2} [P]_2 \tag{2.2}$$

et, si $Q \in \mathbb{C}[X]$:

$$|PQ|_{\mathbb{T}} \leq |P|_{\mathbb{T}} \cdot |Q|_{\mathbb{T}} \quad \textit{et} \quad |PQ| \leq |P| \cdot |Q|. \tag{2.3}$$

Preuve

L'inégalité $\|P\| \leq |P|_{\mathbb{T}}$ se montre grâce à la formule de Parseval (voir par exemple [7]). Les inégalités (2.2) et $|P| \leq \sqrt{n+1} \cdot \|P\|$ sont des conséquences de l'inégalité de Schwarz.

Remarque : La relation (2.3) de sous-multiplicativité n'est pas satisfaite pour la norme $\| \ \|$. Elle l'est par contre pour la norme $[.]_2$ (cf. [2] prop.5), mais nous ne l'utiliserons pas.

2.1.2 Mesure de Mahler d'un polynôme

La mesure d'un polynôme introduite par Mahler [13] nous permettra d'améliorer les bornes données par Schönhage.

Définition 2 *Soit $P(X) = \beta(X-u_1)\cdots(X-u_n) \in \mathbb{C}[X]$ un polynôme non constant. On appelle mesure de Mahler de P la valeur notée $M(P) = |\beta| \prod_{i=1}^{n} \max(1, |u_i|)$.*

De cette définition découle facilement la propriété suivante :

Proposition 2 *Soient P et $Q \in \mathbb{C}[X]$. Alors :*

$$M(PQ) = M(P)M(Q). \tag{2.4}$$

Enfin, la formule de Jensen [7] nous sera utile pour évaluer la mesure d'un polynôme :

Théorème 1 (Formule de Jensen) *Si P est un polynôme non constant à coefficients complexes, alors*

$$\log(M(P)) = \frac{1}{2\pi}\int_0^{2\pi} \log\left|P\left(e^{i\theta}\right)\right| d\theta. \tag{2.5}$$

Preuve

On peut supposer P unitaire. Écrivons $P(X) = (X-u_1)\cdots(X-u_n)$. Il suffit de montrer

$$\forall k, \quad \frac{1}{2\pi}\int_0^{2\pi} \log\left|e^{i\theta} - u_k\right| d\theta = \begin{cases} 0 & \text{si } |u_k| \leq 1, \\ \log|u_k| & \text{sinon.} \end{cases} \tag{2.6}$$

Supposons dans un premier temps que $|u_k| \neq 1$. La valeur de (2.6) est la limite, lorsque N tend vers l'infini, de la somme de Riemann

$$\frac{1}{N}\sum_{j=0}^{N-1} \log\left|e^{2ij\pi/N} - u_k\right|.$$

Cette expression n'est autre que

$$\frac{1}{N}\log\left|\prod_{j=0}^{N-1}\left(e^{2ij\pi/N} - u_k\right)\right| = \frac{1}{N}\log\left|1 - u_k^N\right|.$$

Des relations

$$\lim_{N\to\infty} \frac{1}{N}\log\left|1-u_k^N\right| = \begin{cases} 0 & \text{si } |u_k| < 1 \\ \log|u_k| & \text{si } |u_k| > 1 \end{cases}$$

on en déduit (2.6).

Si $|u_k| = 1$, il suffit dans (2.6) de faire tendre $|u_k|$ vers 1 par valeurs inférieures et d'appliquer le théorème de convergence dominée. □

Remarque. Une autre égalité remarquable satisfaite par la mesure de Mahler est la suivante (voir [12]). Celle-ci est la conséquence, à partir de (2.5), d'un théorème de Szegö [6] :

$$M(P) = \inf\{\|PQ\|, Q \in \mathbb{C}[X], Q \text{ unitaire}\}.$$

2.1.3 Polynôme réciproque

Définition 3 *Soit $P = a_0 + a_1X + \cdots + a_nX^n$ un polynôme de degré n. On appelle polynôme réciproque de P le polynôme noté :*

$$P^*(X) = X^n P\left(\frac{1}{X}\right) = a_0X^n + a_1X^{n-1} + \cdots a_{n-1}X + a_n.$$

L'intérêt du polynôme réciproque réside principalement dans le résultat suivant :

Proposition 3 *Les racines du polynôme réciproque $P^\star$ sont les inverses des racines de P.*

D'autres résultats immédiats nous seront utiles.

Proposition 4 *Soit P un polynôme à coefficients complexes. Alors*

$$|P| = |P^\star|, \qquad \|P\| = \|P^\star\| \quad \text{et} \quad |P|_{\mathbb{T}} = |P^\star|_{\mathbb{T}}, \tag{2.7}$$

et si le terme constant de P est non nul et P non constant

$$M(P) = M(P^\star). \tag{2.8}$$

2.2 Inégalités et normes

Nous nous intéresserons principalement à la norme $|\ |$. Un premier résultat est le suivant.

Théorème 2 *Soit P un polynôme à coefficients complexes. On a :*

$$|P| \leq 2^n M(P) \tag{2.9}$$

$$M(P) \leq |P|_{\mathbb{T}} \leq |P| \tag{2.10}$$

Preuve

Écrivons $P(X) = \beta(X-u_1)\cdots(X-u_n)$. De (2.3), on tire : $|P| \leq |\beta|(1+|u_1|)\cdots(1+|u_n|)$. Il suffit alors d'écrire que $\forall k, (1+|u_k|) \leq 2\max(1,|u_k|)$, d'où (2.9). (2.10) découle immédiatement de la formule de Jensen (2.5). □

Remarque : Il existe une version plus forte de (2.10) qui est $M(P) \leq \|P\|$ (inégalité de Landau, voir [14]). Nous n'en aurons pas besoin, mais elle exprime déjà que dans la pratique, $M(P)$ est souvent bien plus faible que $|P|$.

Théorème 3 *Soit P un polynôme complexe de degré n, produit des polynômes $P_1, P_2, \ldots, P_k$. Alors*

$$|P_1|\cdot|P_2|\cdots|P_k| \leq 2^n M(P), \tag{2.11}$$

et donc

$$|P_1|\cdot|P_2|\cdots|P_k| \leq 2^n |P|. \tag{2.12}$$

Preuve

D'après (2.9), $\forall i, |P_i| \leq 2^{\deg(P_i)} M(P_i)$, donc $|P_1|\cdot|P_2|\cdots|P_k| \leq 2^n\, M(P_1)M(P_2)\cdots M(P_k)$, d'où (2.11) car d'après (2.4) $M(P_1)\cdots M(P_k) = M(P_1\cdots P_k) = M(P)$. (2.12) est alors une conséquence de (2.10). □

Une autre inégalité beaucoup plus fine que (2.12) et souvent plus fine que (2.11) est la suivante :

Théorème 4 *Soit P un polynôme complexe, produit des polynômes F et G. Alors*

$$|F|\cdot|G| \leq 2^{n/2}\sqrt{\binom{n}{k}}\,[P]_2, \quad \text{où} \quad n = \deg(P) \text{ et } k = \deg(F). \tag{2.13}$$

Preuve

La norme $[.]_2$ vérifie l'inégalité suivante [1]

$$[F]_2[G]_2 \leq \sqrt{\binom{n}{k}}\,[P]_2$$

[1] Nous ne montrerons pas ce résultat qui sortirait trop du cadre de ce rapport; la norme $[.]_2$ est en effet étroitement liée aux polynômes à plusieurs variables et elle y apparaît beaucoup plus naturellement. Une démonstration est donnée dans [2] th.1 — voir aussi [1] —.

d'où le résultat grâce à la majoration (2.2) appliquée respectivement à F et à G.

Remarques

- Dans son algorithme, Schönhage utilise la borne (2.12). Nous utiliserons plutôt les bornes (2.11) ou (2.13), beaucoup plus fines. Citons enfin les inégalités (cf. [2])

$$\binom{n}{[n/2]}^{-1/2} M(P) \leq [P]_2 \leq 2^{n/2} M(P)$$

qui permettent de comparer les bornes données par $[.]_2$ et par la mesure (à ce sujet, des comparaisons intéressantes font l'objet de [22] p.53).

- Donnons également une autre inégalité du même type satisfaite par la norme $|P|_{\mathbb{V}}$ (démontrée récemment par Boyd [3]) $|F|_{\mathbb{V}} \cdot |G|_{\mathbb{V}} \leq \delta^n |P|_{\mathbb{V}}$ (où $P = FG$ et $\deg(P) = n$), où $\delta \simeq 1.792\ldots$ L'amélioration apportée par cette dernière inégalité par rapport à (2.12) est peu intéressante dans notre contexte. De plus, cette majoration est souvent moins fine que (2.13).

2.3 Inégalités sur les modules des racines

Notation 1 *Soit P un polynôme complexe de degré n. On note $r_1(P) \leq r_2(P) \leq \cdots \leq r_n(P)$ les modules des racines de P dans l'ordre croissant. Lorsqu'il n'y aura aucune ambigüité, on notera $r_i = r_i(P)$.*
On appelle le rayon des racines de P le réel $r_n(P)$.

La proposition suivante est une conséquence immédiate de l'expression des coefficients de P comme fonctions symétriques de ses racines :

Proposition 5 *Soit $P(X) = a_0 + a_1 X + \cdots + a_n X^n$ un polynôme complexe de degré n. Alors :*

$$\forall k, \quad \left|\frac{a_{n-k}}{a_n}\right| \leq \binom{n}{k} r_n^k \tag{2.14}$$

et donc $|P| \leq |a_n|(1 + r_n)^n$.

Dans [8], on trouve toute une série d'inégalités sur les rayons des racines d'un polynôme. En particulier :

Théorème 5 *Soit $P = a_0 + a_1 X + \cdots + a_{n-1} X^{n-1} + a_n X^n$ un polynôme complexe de degré $n > 0$. Alors :*

$$r_n(P) < 2\rho, \quad \text{où} \quad \rho = \max_{1 \leq m \leq n} \left|\frac{a_{n-m}}{a_n}\right|^{\frac{1}{m}} \tag{2.15}$$

Preuve
Par définition de ρ on a : $\forall m, \quad \rho^m \geq |a_{n-m}/a_n|$ i.e $|a_{n-m}|\,\rho^{n-m} \leq |a_n|\,\rho^n$.
Par conséquent, si $r := 2\rho$, on a
$|a_0| + |a_1| r + \cdots + |a_{n-1}| r^{n-1} = |a_0| + 2|a_1|\rho + \cdots + 2^{n-1}|a_{n-1}|\rho^{n-1} \leq \left(1 + 2 + \cdots + 2^{n-1}\right)|a_n|\rho^n < 2^n |a_n|\rho^n = |a_n| r^n$.
Ceci étant, soit z un complexe de module $\geq r$. Comme

$$\left|\frac{P(z)}{z^n}\right| \geq |a_n| - \left(\frac{|a_0|}{|z|^n} + \cdots + \frac{|a_{n-1}|}{|z|}\right) \geq |a_n| - \left(\frac{|a_0|}{r^n} + \cdots + \frac{|a_{n-1}|}{r}\right)$$

on en déduit que $P(z) \neq 0$, et ceci $\forall z$, $|z| \geq r$, d'où le théorème.□

Le résultat qui suit est une des pièces maîtresses de la méthode de Schönhage. Il nous permettra de calculer les $r_i(P)$ par une méthode dérivée de la méthode de Graeffe.

Théorème 6 *Soit $P = a_0 + a_1X + \cdots + a_{n-1}X^{n-1} + a_nX^n$ un polynôme complexe de degré $n > 0$. Soit un entier m, $1 \leq m \leq n$ tel que $\exists c > 0, q > 0$ tels que :*

$$\forall k \geq 1, \qquad |a_{m-k}| \leq cq^k\,|a_m|$$

Alors

$$r_m(P) \leq (n-m+1)(c+1)q.$$

Pour prouver ce théorème, nous aurons besoin du résultat suivant, dont on trouvera une démonstration dans [8] (th. 6.4n).

Théorème 7 *Soit $P = a_0 + a_1X + \cdots + a_{n-1}X^{n-1} + a_nX^n$ un polynôme complexe de degré $n > 0$. Soit un entier m, $1 \leq m \leq n$. Alors, si $\rho = r_m(P)$:*

$$|a_m|\rho^m \leq \binom{n}{m}|a_0| + \binom{n-1}{m-1}|a_1|\rho + \cdots + \binom{n-m+1}{1}|a_{m-1}|\rho^{m-1} \tag{2.16}$$

Nous sommes maintenant en mesure de démontrer le théorème 6.

Preuve du théorème 6

Posons là aussi : $\rho = r_m(P)$. Avec ce que l'on vient de voir, les hypothèses du théorème donnent (avec $x = q/\rho$) :

$$1 \leq \binom{n-m+1}{1}cx + \binom{n-m+2}{2}cx^2 + \cdots + \binom{n}{m}cx^m \tag{2.17}$$

Si $\rho \leq q$ c'est terminé. Sinon $x < 1$ et donc (2.17) entraîne :

$$1 + c \leq \frac{c}{(1-x)^{n-m+1}} \quad \text{donc} \quad \frac{1}{x} \leq \frac{\sqrt[n-m+1]{c+1}}{\sqrt[n-m+1]{c+1} - \sqrt[n-m+1]{c}} \leq (c+1)(n-m+1), \tag{2.18}$$

d'où le résultat.□

2.4 Une conséquence de la formule de Cauchy

Lors des majorations que nous effectuerons, nous aurons besoin du résultat suivant.

Théorème 8 *Soient deux polynômes F et $G \in \mathbb{C}[X]$ tels que F, de degré k, a toutes ses racines de module < 1 et G a toute ses racines de module > 1. F et G n'ont donc aucune racine commune. On peut écrire, si K est un polynôme*

$$\exists\, p, q \in \mathbb{C}[X], \text{ tels que } : \quad \frac{K}{FG} = \frac{p}{F} + \frac{q}{G} \iff K = pG + qF.$$

Si de plus $\deg(p) < k$, on a (Γ désignant le cercle unité)

$$p(z) = \frac{1}{2i\pi}\oint_\Gamma \frac{K(t)}{(FG)(t)} \cdot \frac{F(z) - F(t)}{z - t}\,dt. \tag{2.19}$$

Preuve

Nous allons en fait montrer

$$p(z) = \frac{1}{2i\pi}\oint_\Gamma \frac{p(t)}{F(t)} \cdot \frac{F(z) - F(t)}{z - t}\,dt \tag{2.20}$$

ce qui entraînera (2.19) puisque $t \mapsto q(z)/G(z) \cdot [F(z) - F(t)]/(z - t)$ n'a aucun pôle dans le disque unité. L'égalité (2.20) s'écrit aussi

$$p(z) = F(z) \cdot \frac{1}{2i\pi}\oint_\Gamma \frac{p(t)}{F(t)} \cdot \frac{dt}{z - t} \;-\; \frac{1}{2i\pi}\oint_\Gamma \frac{p(t)}{z - t}\,dt. \tag{2.21}$$

Nous allons montrer (2.21) pour $|z| > 1$. Les deux termes de (2.20) étant polynomiaux, cela prouvera le résultat pour tout complexe z. Pour $|z| > 1$, $t \mapsto p(t)/(z-t)$ est analytique dans le disque unité et le second terme du membre droit de (2.21) est nul. On se ramène donc à montrer, pour $|z| > 1$:

$$\frac{p(z)}{F(z)} = \frac{1}{2i\pi} \oint_\Gamma \frac{p(t)}{F(t)} \cdot \frac{dt}{z-t}. \tag{2.22}$$

En effectuant une décomposition en éléments simples, on se ramène à montrer (2.22) dans le cas où :

$$\frac{p(X)}{F(X)} = \frac{1}{(X-u)^\alpha}, \quad \textit{avec} \qquad |u| < 1 \quad \text{et} \quad \alpha \geq 1.$$

Il suffit alors de remarquer que :

$$\frac{1}{(t-u)^\alpha} \cdot \frac{1}{z-t} = \frac{1}{(t-u)^\alpha} \left[\frac{1}{z-u} \cdot \frac{1}{1-\left(\frac{t-u}{z-u}\right)} \right] = \frac{1}{(t-u)^\alpha} \left[\frac{1}{(z-u)} + \frac{(t-u)}{(z-u)^2} + \cdots \right]$$

et donc que le résidu de l'application $t \mapsto 1/(t-u)^\alpha/(z-t)$ se trouvant dans le disque unité est $1/(z-u)^\alpha$ d'où (2.22).

On vient donc de montrer (2.19). Cette relation peut être lue en termes de coefficients, ce qui donne (2.23).□

On obtient immédiatement le corollaire suivant (remarquer que rien n'est supposé sur le degré de q) :

Corollaire 1 *Soient F et G deux polynômes tels que F, de degré k, a toutes ses racines de module < 1 et G a toutes ses racines de module > 1. Si pour des polynômes p et K avec $\deg(p) < k$ on a*

$$pG \equiv K \ [\text{mod } F],$$

alors

$$|p| \leq \gamma \cdot |K| \cdot |F'| \leq \gamma \cdot |K| \cdot k \cdot |F| \qquad \text{où } \gamma = \oint_\Gamma |dt/(F(t)G(t))|. \tag{2.23}$$

Remarque : Si $\deg(q) < n-k = \deg(G)$ et si $qF \equiv K \ [\text{mod } G]$, on obtient en prenant les polynômes réciproques une inégalité analogue : $|q| \leq \gamma \cdot |K| \cdot (n-k) \cdot |G|$.

Chapitre 3

Recherche du cercle de séparation

On veut trouver un cercle contenant une partie non vide des racines du polynôme P (mais ne les contenant pas toutes) et tel que ses racines ne s'en approchent pas trop : c'est le cercle de séparation (voir la section 1.3). Pour cela, nous nous servirons du calcul des modules des racines de P. Ceci est permis grâce à la méthode de Graeffe et quelques variantes données par Schönhage.

3.1 La méthode de Graeffe

Soit $P = a_0 + a_1X + \cdots + a_nX^n$ un polynôme de degré $n > 0$. En accord avec 2.3 notation 1, notons $u_1, u_2, \ldots, u_n$ les racines de P telles que $r_1 \leq r_2 \leq \cdots \leq r_n$ (avec $r_i = |u_i|$). On désire approcher numériquement les r_k. L'idée de base est que les racines du polynôme P_1 de degré n défini par

$$P_1(X^2) = P(X)P(-X) = (-1)^n a_n^2 (X^2 - u_1^2) \cdots (X^2 - u_n^2)$$

sont les carrés des racines de P. On note P_1 =Graeffe(P); par récurrence, on peut ainsi définir une suite de polynômes (P_m) par P_m =Graeffe(P_{m-1}). Les racines de P_m sont alors les $(u_i)^{2^m}$. En notant $P_m(X) = a_0^{(m)} + a_1^{(m)}X + \cdots + a_n^{(m)}X^n$, on voit que

$$\forall k, 1 \leq k \leq n : \qquad \frac{a_{n-k}^{(m)}}{a_n^{(m)}} = (-1)^k \sum_{1 \leq i_1 < \cdots < i_k \leq n} (u_{i_1} \cdots u_{i_k})^{2^m},$$

donc si $r_1 < r_2 < \cdots < r_n$,

$$\left| \frac{a_{n-k}^{(m)}}{a_n^{(m)}} \right| \sim |u_{n-k+1} \cdots u_n|^{2^m} \qquad (m \to \infty)$$

et donc

$$\forall k, 1 \leq k \leq n, \qquad \lim_{m \to \infty} \left| \frac{a_{k-1}^{(m)}}{a_k^{(m)}} \right|^{2^{-m}} = |u_k| = r_k(P).$$

L'utilisation directe de ce procédé est difficile. En effet :

- La convergence n'est assurée que si $r_1 < r_2 < \cdots < r_n$. On ne peut donc rien dire si pour un k, $r_k = r_{k+1}$ (en particulier si P a des racines doubles).
- Par ailleurs, même si $r_1 < \cdots < r_n$, il se peut que la convergence soit très lente (c'est le cas par exemple si $\exists k$ tel que r_{k+1}/r_k est proche de 1).
- De plus, les coefficients des P_m divergent rapidement en module.

Arnold Schönhage utilise une approche modifiée du problème : son idée est de ne pas chercher à calculer tous les r_k en même temps, mais de se focaliser sur un r_k. Il propose plusieurs variantes, et chacune va nous servir.

1. La première (cf. 3.1.1) nous donne, $R > 0$ étant fixé, un entier k tel que, approximativement (ce terme prendra tout son sens un peu plus tard) : $r_k \leq R \leq r_{k+1}$
2. La deuxième (cf. 3.1.2) nous donne, un entier k étant fixé, une approximation de r_k.
3. Enfin la troisième (cf. 3.1.3) permet de calculer r_n, le module des racines de P, par une méthode beaucoup plus rapide que celle donnée par 3.1.2.

3.1.1 Première variante de la méthode de Graeffe

Soit P un polynôme complexe. Fixons $R > 0$ et $\tau > 0$ (avec τ petit). Cette première variante va nous donner un entier k tel que

$$r_k(P)\, e^{-\tau} < R < r_{k+1}(P)\, e^{\tau} \tag{3.1}$$

(avec, par convention, $r_0 = 0$ et $r_{n+1} = \infty$). Remarquons déjà qu'il peut y avoir plusieurs valeurs de k répondant à la question. Par homothétie (changer $P(X)$ en $P(RX)$), on se ramène immédiatement au cas où $R = 1$.

Commençons par donner une vague idée de l'algorithme. Écrivons $P_0 := P = a_0 + a_1 X + \cdots + a_n X^n$. Prenons k tel que :

$$|a_k| = \sup_{0 \leq i \leq n} |a_i|.$$

Alors, d'après le théorème 6 appliqué à P_0 et à son polynôme réciproque P_0^* avec $c = q = 1$, on obtient :

$$r_k \leq 2n \quad \text{et} \quad r_{k+1} \geq \frac{1}{2n}.$$

On aura donc trouvé l'entier k si $e^{\tau} > 2n$, *i.e.* $\tau > \log(2n)$. Une telle valeur de τ semble être déraisonnablement grande, mais l'astuce est que par itérations de Graeffe, la valeur de τ sera géométriquement augmentée de sorte qu'au bout d'un nombre fini d'itérations, la valeur $\log(2n)$ sera dépassée.

Pour résoudre le problème (3.1), on va donc effectuer des itérations de Graeffe. Comme on l'a déjà dit, les coefficients des polynômes ainsi obtenus divergent rapidement en module, et il sera donc nécessaire d'arrondir les polynômes obtenus. Nous aurons besoin du résultat de perturbation suivant.

Théorème 9 *Soit $\tau > 0$, soit P un polynôme complexe de degré $n > 0$, et $\hat{P}$ une approximation de P telle que :*

$$|P - \hat{P}| < \frac{1}{2^{6n-1}} \left(\frac{\tau}{e^{8\tau/7}}\right)^n |P|$$

Alors si

$$r_k(\hat{P})\, e^{-\tau} \leq 1 \leq r_{k+1}(\hat{P})\, e^{\tau},$$

on a

$$r_k(P)\, e^{-8\tau/7} \leq 1 \leq r_{k+1}(P)\, e^{8\tau/7}.$$

Preuve

Montrons par exemple $1 \leq r_{k+1}(P)\, e^{8\tau/7}$. Pour cela, raisonnons par l'absurde et supposons $r_{k+1}(P) \leq e^{-8\tau/7}$. Considérons la famille continue de polynômes :

$$H_t = P + t(\hat{P} - P), \qquad 0 \leq t \leq 1$$

Si $\epsilon = 1/2^{6n-1} \cdot (\tau/e^{\tau})^n$, on a $\forall t, |P - H_t| \leq |P - \hat{P}| \leq \epsilon |P|$. L'application $t \mapsto r_{k+1}(H_t)$ est continue (!) et varie de $r_{k+1}(P) \leq e^{-8\tau/7} = p$ à $r_{k+1}(\hat{P}) \geq e^{-\tau} = q$. On en déduit que pour tout r dans $[p, q]$, il existe un t tel que H_t a une racine z de module $|z| = r$, donc

$$\forall r \in [p, q], \exists z, |z| = r \quad \text{tel que} \quad |P(z)| = |P(z) - H_t(z)| \leq |P - H_t| < \epsilon |P|.$$

Ceci entraîne, si $P(z) = \beta(z-u_1)\cdots(z-u_n)$:

$$\epsilon|P| > |\beta|\prod_{j=1}^{n}|z-u_j| \geq |P|\prod_{j=1}^{n}\left(\frac{|r-r_j|}{1+r_j}\right) \geq |P|\prod_{j=1}^{n}\left(\frac{|r-\rho_j|}{1+\rho_j}\right),$$

où $\rho_j = \min\{1, r_j\}$, et par conséquent :

$$\prod_{j=1}^{n}|r-\rho_j| < 2^n\epsilon, \qquad \forall r \in [p,q] \tag{3.2}$$

Par une propriété bien connue du $n^{ième}$ **polynôme** de Tchébichev (cf. [19], VI n. 63), on peut trouver $r \in [p,q]$ tel que le membre de gauche dans (3.2) est plus grand que $(q-p)^n\, 2^{1-2n}$. De plus $q-p = e^{-\tau} - e^{-8\tau/7} > \tau/8 \cdot e^{-8\tau/7}$, donc

$$\left(\frac{\tau}{8}e^{-8\tau/7}\right)^n 2^{1-2n} \leq 2^n\epsilon,$$

absurde de par la valeur de ϵ.□

Donnons maintenant l'algorithme de recherche de k pour résoudre le problème (3.1). (On s'est ramené, rappelons le, à $R=1$).

Algorithme 1 *1. On pose $P_0 = P$ et $\tau_0 = 7/8 \cdot \tau$. Soit*

$$\epsilon_0 = \frac{1}{2^{6n-1}}\left(\frac{\tau_0}{e^{8\tau_0/7}}\right)^n.$$

On arrondit P_0 à $\epsilon_0|P_0|$ près, donnant ainsi un polynôme $\hat{P}_0$ tel que $|P_0 - \hat{P}_0| < \epsilon_0|P_0|$.

2. On calcule exactement $P_1 = Graeffe(\hat{P}_0)$. On pose $\tau_1 = 7/4 \cdot \tau_0$.

3. Soit

$$\epsilon_1 = \frac{1}{2^{6n-1}}\left(\frac{\tau_1}{e^{8\tau_1/7}}\right)^n.$$

On arrondit P_1 à $\epsilon_1|P_1|$ près, donnant ainsi un polynôme $\hat{P}_1$ tel que $|P_1 - \hat{P}_1| < \epsilon_1|P_1|$.

4. De manière générale, on calcule $P_m = Graeffe(\hat{P}_{m-1})$. On pose $\tau_m = 7/4 \cdot \tau_{m-1} = (7/4)^m \cdot 7/8 \cdot \tau$, on arrondit P_m à $\epsilon_m|P_m|$ prés, où

$$\epsilon_m = \frac{1}{2^{6n-1}}\left(\frac{\tau_m}{e^{8\tau_m/7}}\right)^n.$$

5. On s'arrête au rang m dès que

$$\tau_m = \left(\frac{7}{4}\right)^m \cdot \frac{7}{8}\tau \geq \log(2n).$$

6. Si $\hat{P}_m = a_0 + a_1X + \cdots + a_nX^n$, on prend k tel que

$$|a_k| = \max_{1\leq i\leq n}|a_i|.$$

7. L'entier k ainsi déterminé répond au problème (3.1).

Étude de l'algorithme 1 :

Le choix de k (dernière étape) entraîne, d'après le théorème 6 (appliqué à $\hat{P}_m$ puis à son polynôme réciproque $\hat{P}_m^*$ avec $c = q = 1$) que

$$r_k(\hat{P}_m) < 2n < e^{\tau_m} \quad \text{et} \quad r_{k+1}(\hat{P}_m) > \frac{1}{2n} > e^{-\tau_m},$$

et donc, d'après le théorème 9, de par le choix de ϵ_m

$$r_k(P_m) < e^{8\tau_m/7} \quad \text{et} \quad r_{k+1}(P_m) > e^{-8\tau_m/7}.$$

Comme P_m =Graeffe($\hat{P}_{m-1}$), les racines de P_m sont les carrés des racines de $\hat{P}_{m-1}$, de sorte que

$$r_k(\hat{P}_{m-1}) < e^{4\tau_m/7} = e^{\tau_{m-1}} \quad \text{et} \quad r_{k+1}(\hat{P}_{m-1}) > e^{-4\tau_m/7} = e^{-\tau_{m-1}},$$

et toujours d'après le théorème de perturbation 9

$$r_k(P_{m-1}) < e^{8\tau_{m-1}/7} \quad \text{et} \quad r_{k+1}(P_{m-1}) > e^{-8\tau_{m-1}/7}.$$

Par récurrence on obtient donc

$$r_k(P_0) < e^{8\tau_0/7} = e^{\tau} \quad \text{et} \quad r_{k+1}(P_0) > e^{-8\tau_0/7} = e^{-\tau},$$

ce qui prouve que k répond au problème posé.
Remarques :

- Cette variante de la méthode de Graeffe permet de déterminer le rayon d'un cercle de centre 0 éloigné des racines. Elle est beaucoup plus économique que la méthode de Graeffe décrite en introduction de la section 3; cette dernière impliquerait en effet de garder à chaque itération une précision ϵ_m telle que $\log(\epsilon_m) = O(2^m)$, ce qui devient très vite prohibitif (on a ici $\log(\epsilon_m) = O(n\log(1/\tau)) = O(n\log(n))$ car dans notre algorithme, on aura $1/\tau = O(n)$).
- Dans son algorithme, Schönhage obtient les même majorations sauf qu'à chaque fois, la constante 7/4 (ou 4/7) est remplacée par 3/2 (ou 2/3) (et en plus $\epsilon_k = 1/2^{5n-1}(\tau_i e^{-\tau_i})^n$ au lieu de notre ϵ_k). Expérimentalement, le temps d'exécution est légèrement inférieur avec la constante 7/4.

3.1.2 Deuxième variante de la méthode de Graeffe

On se donne un polynôme complexe $P = a_0 + a_1X + \cdots + a_nX^n$. Soit un entier k, $1 \leq k \leq n$, et un paramètre d'erreur $\tau > 0$. Cette deuxième variante va nous donner $R > 0$ tel que

$$Re^{-\tau} < r_k(P) < Re^{\tau} \tag{3.3}$$

(il sera supposé que $r_k(P) \neq 0$, ce qui sera le cas dans l'algorithme). Commençons par donner l'idée générale de cette variante. On définit $\tilde{P}(X) = P(\rho X) = \tilde{a}_0 + \tilde{a}_1X + \cdots + \tilde{a}_nX^n$, où ρ est déterminé de sorte que

$$\exists l \leq k-1 \quad \text{et} \quad \exists h \geq k \quad \text{tels que } |\tilde{a}_l| = |\tilde{a}_h| \quad \text{et} \quad \forall j, 1 \leq j \leq n, \quad |\tilde{a}_j| \leq |\tilde{a}_l|. \tag{3.4}$$

En appliquant le théorème 6 (à $\tilde{P}$ puis $\tilde{P}^*$, avec $c = q = 1$), on a donc :

$$\frac{1}{2n} < r_l(\tilde{P}) \leq r_k(\tilde{P}) \leq r_h(\tilde{P}) < 2n.$$

ρ, h et l peuvent être déterminé à partir des sommets de l'enveloppe convexe supérieure des points du diagramme de Newton (voir figure 2).

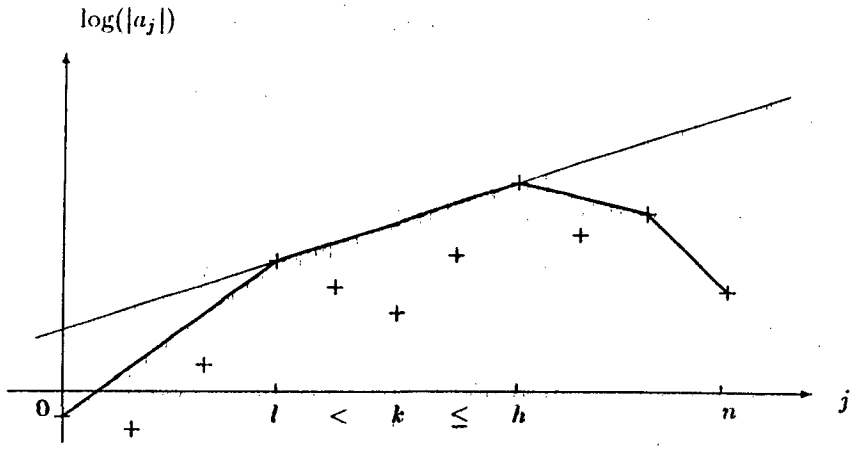


Figure 2. Les traits en gras joignent les sommets de l'enveloppe convexe supérieure des points $\log|a_j|$. Il faut déterminer ρ de sorte que la droite tracée en trait fin devienne horizontale.

Le réel $R = \rho$ répondra donc au problème (3.3) si $e^\tau > 2n$, i.e, $\tau > \log(2n)$. Cette valeur de τ peut paraître grande, mais par itérations de Graeffe, elle est à chaque pas doublée de sorte qu'au bout d'un moment, la valeur $\log(2n)$ sera dépassée.

Rentrons dans les détails; comme dans la première variante, nous aurons besoin d'un résultat de perturbation (la preuve est sensiblement identique à celle du théorème précédent),

Théorème 10 *Soit $\tau > 0$, soit P un polynôme de degré $n > 0$ et $\hat{P}$ une approximation de P telle que*

$$|P - \hat{P}| < \frac{1}{(2n)^n} \cdot \frac{1}{2^{2n-1}}(1 - e^{-\tau})^n.$$

Alors

$$\text{si } \frac{1}{2n} < r_k(P) < 2n, \quad \text{on a} \quad r_k(\hat{P}) \cdot e^{-\tau} < r_k(P) < r_k(\hat{P}) \cdot e^{\tau}.$$

Passons maintenant à l'algorithme proprement dit :

Algorithme 2 *1. On pose $P_0 = P$ et on détermine ρ_0 de sorte que $\tilde{P}_0(X) = P_0(\rho_0 X)$ vérifie les conditions de (3.4).*

2. Soit

$$\epsilon = \frac{1}{(2n)^n} \cdot \frac{1}{2^{2n-1}}(1 - e^{-\tau/4})^n.$$

On arrondit $\tilde{P}_0$ à $\epsilon|\tilde{P}_0|$ près, donnant ainsi un polynôme $\hat{P}_0$ tel que $|\hat{P}_0 - \tilde{P}_0| < \epsilon|\tilde{P}_0|$.

3. On calcule exactement $P_1 = Graeffe(\hat{P}_0)$. On détermine ensuite ρ_1 de sorte que $\tilde{P}_1(X) = P_1(\rho_1 X)$ vérifie les conditions de (3.4).

4. On arrondit $\tilde{P}_1$ à $\epsilon|\tilde{P}_1|$ prés, donnant ainsi un polynôme $\hat{P}_1$ tel que $|\hat{P}_1 - \tilde{P}_1| < \epsilon|\tilde{P}_1|$.

5. De manière générale, on calcule $P_m = Graeffe(\hat{P}_{m-1})$, puis on détermine ρ_m de sorte que $\tilde{P}_m(X) = P_m(\rho_m X)$ vérifie (3.4). On prend ensuite $\hat{P}_m$, arrondi de $\tilde{P}_m$ à $\epsilon|\tilde{P}_m|$ près.

6. On s'arrête au rang m dès que

$$(2n)^{2^{-m}} < e^{\tau/2}.$$

7. *Le réel*

$$R = \rho_0 \cdot \rho_1^{1/2} \cdot \rho_2^{1/4} \cdots \rho_m^{1/2^m}$$

répond alors au problème (3.3).

Étude de l'algorithme 2 :

On a déjà : $r_k(P_0) = \rho_0 \cdot r_k(\tilde{P}_0)$, et donc, d'après le théorème 10, vu le choix de ϵ :

$$\rho_0 \cdot r_k(\hat{P}_0) \cdot e^{-\tau/4} < r_k(P_0) < \rho_0 \cdot r_k(\hat{P}_0) \cdot e^{\tau/4}.$$

Comme P_1 =Graeffe($\hat{P}_0$), on a $r_k(P_1) = r_k(\hat{P}_0)^2$ de sorte que :

$$\rho_0 \cdot r_k(P_1)^{1/2} \cdot e^{-\tau/4} < r_k(P_0) < \rho_0 \cdot r_k(P_1)^{1/2} \cdot e^{\tau/4}.$$

Or, de même que pour P_0 :

$$\rho_1 \cdot r_k(\hat{P}_1) \cdot e^{-\tau/4} < r_k(P_1) < \rho_1 \cdot r_k(\hat{P}_1) \cdot e^{\tau/4},$$

d'où on tire :

$$\rho_0 \cdot \rho_1^{1/2} \cdot r_k(\hat{P}_1)^{1/2} \cdot e^{-\tau/4-\tau/8} < r_k(P_0) < \rho_0 \cdot \rho_1^{1/2} \cdot r_k(\hat{P}_1) \cdot e^{\tau/4+\tau/8}.$$

En procédant par récurrence, on obtient ainsi

$$\rho_0 \cdots \rho_m^{1/2^m} \cdot r_k(\hat{P}_m)^{1/2^m} \cdot e^{-\tau/4-\cdots-\tau/2^m} < r_k(P_0) < \rho_0 \cdots \rho_m^{1/2^m} \cdot r_k(\hat{P}_m)^{1/2^m} \cdot e^{\tau/4+\cdots+\tau/2^m}.$$

C'est à dire, si $R = \rho_0 \cdot \rho_1^{1/2} \cdots \rho_m^{1/2^m}$:

$$R \cdot r_k(\hat{P}_m)^{1/2^m} \cdot e^{-\tau/2} < r_k(P_0) < R \cdot r_k(\hat{P}_m)^{1/2^m} \cdot e^{\tau/2}.$$

Comme $1/2n < r_k(\hat{P}_m) < 2n$, on en déduit, étant donnée la valeur de m (cf étape 6), que R répond au problème (3.3).

Remarques

- Ici, contrairement à l'algorithme précédent, la même précision ϵ est utilisée à chaque itération.
- Pour les mêmes raisons que dans 3.1.1, cette variante est beaucoup plus économique que la méthode de Graeffe usuelle : c'est le fait de borner $r_k(\tilde{P}_m)$ à chaque itération qui permet d'utiliser une précision ϵ raisonnable.
- La précision ϵ est ici beaucoup plus faible que dans la méthode précédente; cette variante est donc relativement coûteuse et on veillera dans l'algorithme à s'en servir le moins possible (voir 3.2.2).

3.1.3 Troisième variante : calcul du module des racines de P

On aura souvent besoin d'une valeur approchée de $r_n(P)$ (i.e le plus grand des modules des racines de P). Dans ce cas particulier, il existe une méthode beaucoup plus économique que la précédente que nous décrivons maintenant.

Commençons par esquisser le principe de l'algorithme. Soit $\tau > 0$ un paramètre d'erreur. On veut trouver $R > 0$ tel que :

$$R \cdot e^{-\tau} < r_n(P) < R \cdot e^{\tau} \tag{3.5}$$

Partant de P, on détermine $\rho > 0$ et β tels que le polynôme $\tilde{P}(X) = \beta \cdot P(\rho X)$ soit unitaire et vérifie, si on note $\tilde{a}_i$ ses coefficients :

$$\forall j, 1 \le j \le n, \quad |\tilde{a}_j| \le \binom{n}{j} \quad \text{et} \quad \exists h, 1 \le h \le n, \text{tel que} : \quad |\tilde{a}_h| = \binom{n}{h} \tag{3.6}$$

Comme dans 3.1.2, ρ et h peuvent être déterminés à partir du diagramme de Newton des points $\log(|a_j|/\binom{n}{j}))$. D'après la proposition 5 et le théorème 5, on a alors

$$1 \leq r_n(\widetilde{P}) < 2n,$$

et donc $R := \rho$ répondra à la question si $\tau > \log(2n)$. Si tel n'est pas le cas, on effectue des itérations de Graeffe jusqu'à ce que la valeur $\log(2n)$ soit dépassée.

On ne voit pas encore l'avantage de cette méthode. Il réside en fait dans le résultat de perturbation qui suit et qui nous permettra de coder les polynômes avec moins de chiffres significatifs que dans les variantes précédentes.

Théorème 11 *Soit $\tau > 0$, soit P un polynôme unitaire de degré $n > 0$, et soit $\widehat{P}$ une approximation de P, unitaire de degré n telle que*

$$|P - \widehat{P}| < (1 - e^{-\tau})^n.$$

Alors, si $r_n(P) \geq 1$, on a

$$r_n(\widehat{P})e^{-\tau} \leq r_n(P) \leq r_n(\widehat{P})e^{\tau}.$$

Preuve

Supposons par exemple $r_n(\widehat{P}) > r_n(P)$. Soit z une racine de $\widehat{P}$ telle que $|z| = r_n(\widehat{P}) > 1$. Alors

$$|P(z)| = |P(z) - \widehat{P}(z)| \leq |z|^n |P - \widehat{P}| < r_n(\widehat{P})^n \cdot (1 - e^{-\tau})^n$$

et

$$|P(z)| \geq \prod_{j=1}^{n} (|z| - r_j(P)) \geq (r_n(\widehat{P}) - r_n(P))^n,$$

d'où on déduit le résultat.□

Passons maintenant à la description précise de l'algorithme. Ici, contrairement à la variante précédente, la précision utilisée change à chaque itération.

Algorithme 3

1. *Partant de $P_0 = P$, on calcule ρ_0 et β_0 tels que le polynôme $\widetilde{P}_0(X) = \beta_0 \cdot P_0(\rho_0 X)$ vérifie (3.6).*
2. *Soit $\tau_0 = \tau/6$ et $\epsilon_0 = (1 - e^{-\tau_0})^n$. On arrondit $\widetilde{P}_0$ à ϵ_0 près, donnant ainsi un polynôme $\widehat{P}_0$ tel que $|\widetilde{P}_0 - \widehat{P}_0| \leq \epsilon_0$.*
3. *On calcule exactement $P_1(X) = Graeffe(\widehat{P}_0)$, puis on détermine ρ_1 et β_1 tel que le polynôme $\widetilde{P}_1(X) = \beta_1 \cdot P_1(\rho_1 X)$ vérifie (3.6).*
4. *Soit $\tau_1 = 3/2 \cdot \tau_0$ et $\epsilon_1 = (1 - e^{-\tau_1})^n$. On arrondit $\widetilde{P}_1$ à ϵ_1 près, donnant ainsi un polynôme $\widehat{P}_1$ tel que $|\widetilde{P}_1 - \widehat{P}_1| \leq \epsilon_1$.*
5. *De manière générale, au rang m, on calcule $P_m(X) = Graeffe(\widehat{P}_{m-1})$, puis ρ_m et β_m tel que le polynôme $\widetilde{P}_m(X) = \beta_m \cdot P_m(\rho_m X)$ vérifie (3.6). On pose $\tau_m = 3/2 \cdot \tau_{m-1}$ et*

   $$\epsilon_m = (1 - e^{-\tau_m})^n,$$

   puis on arrondit $\widetilde{P}_m$ à ϵ_m près, donnant ainsi $\widehat{P}_m$.
6. *On s'arrête au rang m lorsque*

   $$(2n)^{2^{-m}} < e^{\tau/3}$$

7. Le réel

$$R = \rho_0 \cdot \rho_1^{1/2} \cdot \rho_2^{1/4} \cdots \rho_m^{1/2^m}$$

répond alors au problème (3.5).

Étude de l'algorithme 3

De la même manière que dans 3.1.2, on montre, si $R = \rho_0 \cdot \rho_1^{1/2} \cdots \rho_m^{1/2^m}$:

$$R \cdot r_n(\widehat{P}_m)^{1/2^m} \cdot e^{-\tau_0 - \tau_1/2 - \cdots - \tau_m/2^m} < r_k(P) < R \cdot r_n(\widehat{P}_m)^{1/2^m} \cdot e^{\tau_0 + \tau_1/2 + \cdots + \tau_m/2^m}.$$

Or

$$\tau_0 + \frac{\tau_1}{2} + \cdots + \frac{\tau_m}{2^m} = \frac{\tau}{6} \cdot \left(1 + \frac{3/2}{2} + \cdots + \frac{(3/2)^m}{2^m}\right) < \frac{2\tau}{3}$$

donc, d'après le choix de m (cf. étape 6), R vérifie (3.5).

Remarques

- Si $k \geq 1$, β_k, au signe près, est égal à $(1/\rho_k)^n$ (en effet, si $k \geq 1$, P_k est unitaire au signe près).
- Schönhage, dans son algorithme, utilise une même précision $\epsilon_k = \epsilon = (1 - e^{-\tau/4})^n$ à chaque étape. Expérimentalement, il s'avère que la méthode décrite ici (avec un ϵ_k qui varie) est plus rapide (d'un facteur 2/3 environ). Cela n'était pas le cas dans la variante précédente.

3.2 Recherche du cercle de séparation

On sait maintenant calculer le module des racines du polynôme $P = \beta(X - u_1) \cdots (X - u_n)$. On veut déterminer un cercle de séparation, c'est à dire un cercle contenant une partie non vide des racines (notées $u_1, \ldots, u_k$ avec $k > 0$) et ne les contenant pas toutes (donc $k < n$). On veut aussi (voir 1.3) que ce cercle (de centre c de rayon $R > 0$) soit loin des racines, autrement dit qu'il existe une couronne $\{z \in \mathbb{C}, Re^{-\delta} < |z - c| < Re^{\delta}\}$ ne contenant aucune des racines, avec $\delta > 0$ pas trop petit (voir figure 3).

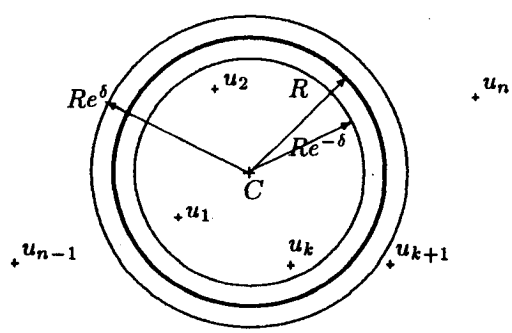


Figure 3

Une possibilité serait par exemple la suivante :

On calcule le module de toutes les racines du polynôme : $r_1 \leq r_2 \leq \cdots \leq r_n$, puis on choisit k tel que r_{k+1}/r_k soit maximal. Le cercle de centre 0 de rayon $R = \sqrt{r_k r_{k+1}}$ convient alors, avec $\delta = 1/2 \cdot \log(r_{k+1}/r_k)$. Mais il se pourrait que les r_i soient tous égaux, où qu'ils soient proches; δ serait alors très faible (voire nul !), ce qui n'est pas intéressant. Pour déterminer le cercle de séparation, on va donc procéder autrement.

3.2.1 Recherche du centre du cercle de séparation

Soit P un polynôme de degré $n > 0$. On cherche le centre d'un cercle de séparation pour P; celui ci devra être déterminé de sorte que $\Delta = \log(r_n/r_1)$ ne soit pas trop petit.

Algorithme 4 *1. On commence par effectuer une translation de sorte que l'origine soit le barycentre des racines de P (i.e. on considère $P\left[X - \frac{a_{n-1}}{n\,a_n}\right]$).*

2. Si le coefficient constant du polynôme obtenu est nul, on a trouvé un facteur; sinon R, le plus grand des modules des racines de ce polynôme est non nul. On le calcule grâce à l'algorithme 3 avec une erreur relative de 0.02. En considérant le polynôme $P(RX)$, on se ramène ainsi à un polynôme dont le rayon des racines est 1 à 0.02 près.

3. On pose $v_0 = 2$, $v_1 = 2i$, $v_2 = -2$, $v_3 = -2i$. Pour $i = 0 \ldots 3$, on calcule le polynôme $Q_i(X) = P(X + v_i)$. On cherche ensuite le plus grand des rayons de ses racines $r_n(Q_i)$ et le plus petit $r_1(Q_i)$ grâce à l'algorithme 3 ($r_1(Q_i) = 1/r_n(Q_i^*)$). On choisit alors i tel que $r_n(Q_i)/r_1(Q_i)$ soit maximal. (On peut montrer que $\exists i$ tel que $\log(r_n(Q_i)/r_1(Q_i)) > 0.30$, voir figure 4).

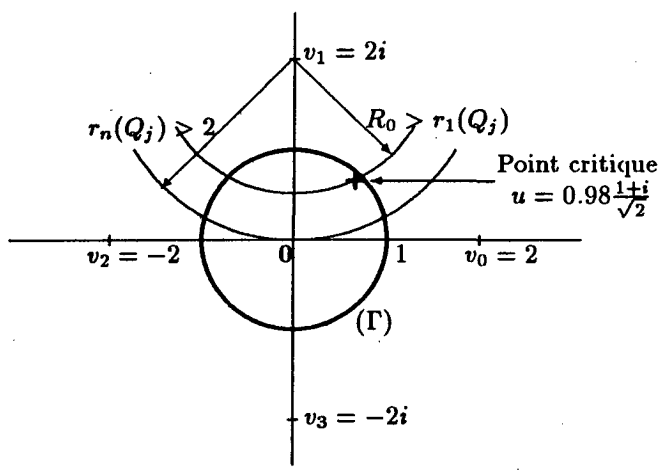


Figure 4. Les centres $v_i \in \{2, 2i, -2, -2i\}$ pour obtenir l'écartement $\Delta = \log(r_n/r_1) > 0.30$.

4. Le centre est alors translaté en v_i : En considérant le polynôme $P(X + v_i)$, on se ramène au cas où ce centre est l'origine.

3.2.2 Recherche du rayon du cercle de séparation

On s'est ramené au cas où $\Delta = \log(r_n/r_1) > 0.30$. De la relation $\sum_{i=1}^{n-1} \log\left(\frac{r_{i+1}}{r_i}\right) = \Delta$, on tire

$$\exists k, \qquad \log\left(\frac{r_{k+1}}{r_k}\right) > \frac{\Delta}{n-1} > \frac{0.30}{n-1}.$$

Comme on l'a déjà dit, on pourrait alors calculer tous les r_k et choisir k tel que $\log(r_{k+1}/r_k)$ soit maximal (le rayon du cercle serait alors $R = \sqrt{r_k \cdot r_{k+1}}$ avec $\delta = \frac{1}{2}\log(r_{k+1}/r_k) > 0.15/(n-1)$). Ceci est faisable lorsque n est faible, mais devient trop coûteux lorsque n augmente.

Une autre solution serait alors de procéder par dichotomie : partant de $i = 1, j = n$, on maximise à chaque fois l'un des termes :

$$\frac{1}{k-i}\log\left(\frac{r_k}{r_i}\right), \quad \frac{1}{j-k}\log\left(\frac{r_j}{r_k}\right), \qquad \text{avec } k = \lfloor\frac{i+j}{2}\rfloor$$

puis on remplace i ou j par k selon que le second ou le premier terme est maximal. On ne calcule ainsi plus que $O[\log_2(n)]$ (au lieu de $O(n)$) modules de racines (en utilisant l'algorithme 2).

En fait, l'algorithme 2 est plus coûteux que l'algorithme 1, et on va procéder autrement en utilisant ce dernier. De plus, à chaque étape de factorisation $P = F \cdot G$, on aimerait bien avoir $\deg(F) \simeq \deg(G)$ (pour minimaliser le nombre de factorisations à effectuer). Le fait est que, souvent, les modules des racines r_k sont concentrés lorsque $k \sim n/2$. Les méthodes décrites précédemment mèneraient donc à des valeurs de k se trouvant vers l'une des extrémités $k = 1$ ou $k = n$, ce qui n'est pas satisfaisant. Il faut donc trouver un compromis; en d'autres termes, pour avoir k suffisamment loin des extrémités $k = 1$, $k = n$, on est prêt à sacrifier la valeur de δ en en admettant une plus faible (pas trop quand même). Beaucoup d'algorithmes respectant cette idée pourraient être construits. En voici un, différent de celui de Schönhage et qui dans la pratique est assez satisfaisant.

Commençons par donner une idée de l'algorithme. Grâce à l'algorithme 1, on va, en procédant par dichotomie, déterminer des suites :

$$(i_p), \quad (j_p), \quad (m_p), \quad (M_p)$$

satisfaisant aux conditions initiales

$$i_0 = 0, \quad j_0 = n, \quad m_0 = r_1(P), \quad M_0 = r_n(P),$$

telles que

$$\forall p, \quad r_{i_p}(P) \leq m_p < M_p < r_{j_p}(P)$$

et

$$\forall p, \quad (j_{p+1} - i_{p+1}) \leq \frac{j_p - i_p}{2}.$$

Ainsi, au bout de $p \leq \lceil \log_2(n) \rceil$ itérations, on a $i_p = j_p$ et donc aucun des modules $r_i(P)$ ne se trouve dans l'intervalle $]m_p, M_p[$. Le rayon $R = \sqrt{m_p \cdot M_p}$ convient alors avec $\delta = 1/2 \cdot \log(M_p/m_p)$ et $k = i_p$. Cette valeur de δ sera souvent moindre que la valeur $1/2 \cdot \log(r_{k+1}/r_k)$. En conséquence, on détermine ensuite r_k et r_{k+1} grâce à l'algorithme 2, puis on prend :

$$R = \sqrt{r_k \cdot r_{k+1}}, \qquad \delta = \frac{1}{2} \log\left(\frac{r_{k+1}}{r_k}\right).$$

Disons également que les inégalités données plus haut ne seront pas strictes dans le sens où elles dépendront d'un paramètre d'erreur judicieusement choisi.

Algorithme 5 *1. On commence par calculer $m := r_1$ et $M := r_n$ à $\Delta/(10n)$ près (grâce à l'algorithme 3). On pose $i=0$ et $j=n$.*

2. On pose $\rho = \sqrt{m \cdot M}$. Grâce à l'algorithme 1 appliqué à ρ avec une erreur $\tau = \Delta/10$, on détermine un entier k tel que

$$r_k(P)e^{-\tau} \leq \rho \leq r_{k+1}(P)e^{\tau}.$$

Si $k < n/5$ ou $n/2 < k < 4n/5$, on fait $M := \rho$ et $j := k$, sinon on fait $m := \rho$ et $i := k$.

3. Les itérations suivantes sont analogues sauf que le choix de ρ essaiera de satisfaire le discours tenu plus haut :

On part de i, j, m et M qui par construction satisfont (à un terme d'erreur prés)

$$r_i(P) \leq m < M \leq r_j(P).$$

On choisit un ρ dans l'intervalle $]m, M[$ (voir étape suivante) puis on détermine grâce à l'algorithme 1 un entier k tel que :

$$r_k(P)e^{-\tau} \leq \rho \leq r_{k+1}e^{\tau}, \qquad \text{avec } \tau = \frac{1}{10} \cdot \frac{1}{(j-i)} \cdot \log(r_j/r_i).$$

Si $(k-i) < (j-k)$ *(i.e s'il y a moins de modules de racines dans l'intervalle* $]m, \rho[$ *que dans* $]\rho, M[$*) alors on prend*

$$M := \rho, \quad j := k, \quad \text{et on laisse } m \text{ et } i \text{ inchangés;}$$

si au contraire $(k-i) > (j-k)$*, alors on prend*

$$m := \rho, \quad i := k, \quad \text{et on laisse } M \text{ et } j \text{ inchangés;}$$

Si maintenant $(k-i) = (j-k)$*, on se ramène au premier cas si* $k > n/2$*, au second sinon (ceci encore pour essayer de se rapprocher du milieu de l'échelle).*

4. *Choix de* ρ*. On calcule un terme de poids :*

$$\kappa = \alpha \left(1 - \frac{\log(d_i + 1)}{\log(d_e + 1)}\right),$$

où $d_i := \min\{i, n-j\}$ *(distance intérieure du couple* (i,j) *au bord de l'échelle),* $d_e := \min\{j, n-i\}$ *(distance extérieure de* (i,j) *au bord de l'échelle) et* α *est un paramètre que l'on peut prendre par exemple égal à* 1*. (Le terme de poids* κ *est choisi de telle sorte qu'il soit important lorsque l'on est prés du bord de l'échelle et faible lorsque l'on se trouve vers son milieu). Si* $j \leq n/2$ *on choisit alors*

$$\rho := \left(m \cdot M^{1+\kappa}\right)^{1/(2+\kappa)},$$

sinon

$$\rho := \left(m^{1+\kappa} \cdot M\right)^{1/(2+\kappa)}.$$

5. *On s'arrête lorsque* $i = j$*. On pose alors* $k := i$ *puis on calcule, grâce à l'algorithme 2,* $r_k(P)$ *et* $r_{k+1}(P)$*. On prend alors le rayon* $R = \sqrt{r_k \cdot r_{k+1}}$ *avec* $\delta = 1/2 \cdot \log(r_{k+1}/r_k)$.

Remarque : C'est à partir de cette valeur de δ que Schönhage calcule le polynôme F dont les racines sont dans le cercle de séparation. En fait, en effectuant une transformation sur le polynôme, on va voir (cf. section 9) qu'on peut améliorer la valeur de δ (d'un facteur variant de 2 à 4).

Chapitre 4

Factorisation initiale à partir du cercle de séparation

Nous avons trouvé un cercle de séparation Γ pour le polynôme complexe P de degré $n > 0$. Quitte à effectuer un changement de repère, on peut supposer que ce cercle de séparation est le cercle unité. Par ailleurs, la recherche de Γ nous a fourni un réel $\delta > 0$ tel qu'aucune des racines de P ne se trouve dans la couronne $\{z \in \mathbb{C}, e^{-\delta} \leq |z| \leq e^{\delta}\}$. Conformément aux notations précédentes, nous noterons $u_1, \ldots, u_k$ les racines de P se trouvant dans le cercle de séparation, $u_{k+1}, \ldots, u_n$ celles se trouvant en dehors (voir figure 3).

Le problème que l'on se pose est le suivant : on note $F = (X - u_1) \cdots (X - u_k)$ le polynôme unitaire dont les racines se trouvent dans le cercle de séparation, $G = \beta(X - u_{k+1}) \cdots (X - u_n)$ l'autre facteur de P dont les racines se trouvent en dehors du cercle, de sorte que $P = F \cdot G$. $\epsilon > 0$ étant donné, on veut calculer numériquement une approximation F_0 de F telle que :

$$|F_0 - F| \leq \epsilon.$$

Ceci sera effectué à partir des approximations numériques (par discrétisation du cercle unité) des intégrales de Cauchy (1.3) et des formules de Newton. Le premier résultat qui suit nous éclaire déjà quant à la majoration d'erreur que l'on a sur l'approximation numérique.

Théorème 12 *Avec les hypothèses précédentes, si $N \in \mathbb{N}, N > k$ et si*

$$\forall m, \qquad W_m = \frac{1}{N} \sum_{j=0}^{N-1} \frac{P'(w^j)}{P(w^j)} w^{mj}, \quad \text{où} \quad \omega = e^{2i\pi/N}, \tag{4.1}$$

alors :

$$\forall m, 1 \leq m \leq k, \qquad |W_{m+1} - s_m| \leq \frac{ne^{-\delta(N-m)}}{1 - e^{-\delta N}}, \tag{4.2}$$

où

$$\forall m, \qquad s_m = u_1^m + \cdots + u_k^m = \frac{1}{2i\pi} \oint_\Gamma \frac{P'(z)}{P(z)} z^m \, dz. \tag{4.3}$$

Preuve

La dernière relation est une conséquence immédiate du théorème des résidus et de l'écriture :

$$\frac{P'(z)}{P(z)} = \sum_{j=1}^{n} \frac{1}{z - u_i}. \tag{4.4}$$

Pour montrer (4.2), écrivons le développement de P'/P en série de Laurent au voisinage du cercle unité :

$$\frac{P'(z)}{P(z)} = \sum_{j \in \mathbb{Z}} c_j z^j.$$

D'après (4.4), si s_m est défini par (4.3) et si on note

$$\forall m \in \mathbb{N}, \qquad S_m = \left(\frac{1}{u_{k+1}}\right)^m + \cdots + \left(\frac{1}{u_n}\right)^m,$$

on a

$$c_j = -S_{j+1} \quad \text{si } j \geq 0 \qquad \text{et} \qquad c_j = s_{-j-1} \quad \text{si } j < 0. \tag{4.5}$$

On obtient, en remplaçant dans la série de Laurent

$$\forall p, \qquad W_p = \sum_{j \in \mathbb{Z}} c_{-p+jN},$$

et compte tenu de (4.5)

$$\forall m, 1 \leq m \leq k, \qquad |W_{m+1} - s_m| \leq \sum_{j=1}^{\infty} |s_{jN+m}| + \sum_{j=1}^{\infty} |S_{jN-m}|.$$

Les hypothèses sur les modules des racines de P entraînent que

$$|s_m| \leq k e^{-\delta m} \quad \text{et} \quad |S_m| \leq (n-k) e^{-\delta m}$$

et donc

$$\forall m, 1 \leq m \leq k, \qquad |W_{m+1} - s_m| \leq \frac{k e^{-\delta(N+m)} + (n-k) e^{-\delta(N-m)}}{1 - e^{-\delta N}},$$

d'où le théorème.□

Nous savons donc calculer des approximations W_{m+1} des s_m. L'identité bien connue :

$$F^*(z) = \exp\left(-\sum_{m=1}^{\infty} \frac{s_m}{m} z^m\right)$$

(où F^* est le polynôme réciproque de F), donne, si $F(z) = z^k + \varphi_1 z^{k-1} + \cdots + \varphi_k$, les formules de Newton :

$$\varphi_m = -\frac{1}{m}(s_1 \varphi_{m-1} + \cdots + s_{m-1}\varphi_1 + s_m) \qquad (1 \leq m \leq k).$$

Nous allons donc calculer successivement les $\hat{\varphi}_k$ par :

$$\hat{\varphi}_m = -\frac{1}{m}(W_2 \hat{\varphi}_{m-1} + \cdots + W_m \hat{\varphi}_1 + W_{m+1}) \qquad (1 \leq m \leq k), \tag{4.6}$$

ce qui nous donnera une approximation $F_0 = z^k + \hat{\varphi}_1 z^{k-1} + \cdots + \hat{\varphi}_{k-1} z + \hat{\varphi}_k$ de F.

On se fixe $\epsilon > 0$. On veut savoir jusqu'à quel point les W_{m+1} doivent approximer s_m pour avoir, après calcul de (4.6) $|F_0 - F| < \epsilon$. Ceci repose sur le théorème suivant.

Théorème 13 *Si $F_0 = z^k + \hat{\varphi}_1 z^{k-1} + \cdots + \hat{\varphi}_{k-1} z + \hat{\varphi}_k$ est calculé à partir des relations (4.6), alors :*

$$|F_0 - F| \leq |F| \cdot \left(|D| + \frac{|D|^2}{2!} + \cdots + \frac{|D|^k}{k!}\right), \tag{4.7}$$

où

$$D = \sum_{m=1}^{k} \frac{1}{m}(s_m - W_{m+1}) z^m. \tag{4.8}$$

Preuve

Rappelons l'identité :

$$F^*(z) = \exp\left(-\sum_{m=1}^{\infty} \frac{s_m}{m} z^m\right). \tag{4.9}$$

Posons :

$$A(z) = \exp\left(-\sum_{m=1}^{k} \frac{W_{m+1}}{m} z^m - \sum_{m=k+1}^{\infty} \frac{s_m}{m} z^m\right).$$

Les premiers termes du développement de $A(z)$ correspondent aux coefficients de F_0^*, et, comme $A(z) = F^*(z) \cdot \exp(D(z))$, on peut écrire

$$|F_0^* - F^*| \leq |F^*| \left(|D| + \frac{|D|^2}{2!} + \cdots + \frac{|D|^k}{k!}\right),$$

d'où le résultat.□

Les majorations (4.2) entraînent

$$|D| \leq \frac{k \cdot n \cdot e^{-\delta(N-k)}}{1 - e^{-\delta N}}.$$

Vue la grande taille de N (c'est le cas dans la pratique), on peut légitimement approximer $1 - e^{\delta N}$ à 1. $|D|$ est également faible, de sorte qu'également, on peut lire la majoration (4.7) comme : $|F_0 - F| \leq |F| \cdot |D|$. En couplant ces résultats, on tombe sur :

$$|F_0 - F| \leq |F| \cdot k \cdot n \cdot e^{-\delta(N-k)}. \tag{4.10}$$

C'est cette majoration qui nous permettra de donner une valeur de N pour laquelle on soit sûr qu'après calculs $|F_0 - F| < \epsilon$.

Remarques :

- La majoration (4.10), comme on l'avait pressenti, exprime le fait que plus δ est petit (i.e. plus les racines de P sont proches du cercle de séparation), plus il faudra considérer un grand nombre N de points sur le cercle Γ pour avoir, après les calculs précédents, une bonne approximation F_0 de F. Le choix de Γ est donc important pour minimiser le coût des calculs.
- C'est cette étape qui est coûteuse dans l'algorithme de Schönhage (surtout lorsque n est grand). Une FFT (transformée de Fourier rapide) partielle permet de réduire le coût du calcul des relations (4.1) (c.f. [20] et [11]).

Chapitre 5

L'algorithme d'itération de Newton

Les calculs précédents permettent, $\epsilon > 0$ étant donné, de calculer une approximation F_0 de F telle que $|F_0 - F| < \epsilon$. Ces calculs sont coûteux, surtout lorsque ϵ est petit! Pour cette raison, Schönhage, à partir d'une approximation F_0 de F suffisamment bonne (calculée par la méthode précédente), implémente un algorithme d'itération de Newton qui lui permet de déterminer rapidement une bonne approximation de F. Ce n'est pas à proprement parler une véritable méthode de Newton, au sens usuel en mathématiques; l'algorithme est totalement différent. Nous allons d'abord présenter cet algorithme d'itération de Newton, puis étudier la précision avec laquelle il faut calculer l'approximation initiale F_0 de F par la méthode précédente pour assurer la convergence de cet algorithme.

5.1 Principe de la méthode de Newton-Schönhage

Commençons par donner une idée de l'algorithme d'itération de Newton.

Hypothèses : $P = F \cdot G$ où F est unitaire, les racines de F sont toutes dans le cercle unité, celles de G en dehors (remarquer qu'alors $M(G) = M(P)$). On dispose d'une approximation F_0 de F et G_0 de G (G_0 est obtenu par division euclidienne de P par F_0). On suppose que ces approximations sont telles que les racines de F_0 restent encore dans le cercle unité, celles de G_0 en dehors du cercle unité.

Problème : On veut déterminer f et g tels que les polynômes $F_0 + f$ et $G_0 + g$ soient de meilleures approximations de F et G que F_0 et G_0. Il s'agit donc pour nous de minimiser $|P - (F_0 + f) \cdot (G_0 + g)|$.

Écrivons

$$P - (F_0 + f)(G_0 + g) = P - F_0G_0 - fG_0 - gF_0 - fg.$$

Le terme fg étant du second ordre, on va chercher f et g tels que :

$$P - F_0G_0 = fG_0 + gF_0,$$

ce qui s'écrit encore :

$$\frac{P - F_0G_0}{F_0G_0} = \frac{f}{F_0} + \frac{g}{G_0}.$$

On verra par la suite comment trouver f et g. Supposons que l'on sache les déterminer. On a alors :

$$|P - (F_0 + f)(G_0 + g)| = |fg| \leq |f| \cdot |g|.$$

Or, d'après le corollaire du théorème 8, si $\gamma = \oint |dz/F_0G_0(z)|$, on a :

$$|f| \leq |P - F_0G_0| \cdot \gamma \cdot k \cdot |F_0|.$$

De même, en considérant les polynômes réciproques (voir la remarque du corollaire utilisé) :

$$|g| \leq |P - F_0G_0| \cdot \gamma \cdot (n-k) \cdot |G_0|.$$

Finalement, si $\epsilon_0 = |P - F_0G_0|$ et $\epsilon_1 = |P - (F_0 + f)(G_0 + g)|$:

$$\epsilon_1 \leq \epsilon_0^2 \cdot \gamma^2 \cdot k(n-k) \cdot |F_0| \cdot |G_0|.$$

En vue d'une convergence rapide de l'algorithme d'itération de Newton, on aimerait avoir $\epsilon_1 \leq \epsilon_0^{1.5}$, ce qui sera réalisé si :

$$\epsilon_0 \leq \frac{1}{\gamma^4 \cdot k^2(n-k)^2 \cdot (|F_0||G_0|)^2}. \tag{5.1}$$

Si F_0 est une approximation suffisante de F telle que la borne (5.1) soit réalisée, on sera alors assuré d'une convergence rapide de la méthode de Newton-Schönhage. La relation $P - F_0G_0 \equiv P - F_0G \equiv G(F - F_0)$ [mod F_0] s'écrit aussi $(P - F_0G_0)G_0 \equiv GG_0(F - F_0)$ [mod F_0] et donc, toujours d'après le fameux corollaire :

$$|P - F_0G_0| \leq |G| \cdot |G_0| \cdot \gamma \cdot k \cdot |F_0| \cdot |F - F_0|. \tag{5.2}$$

Si les approximations initiales F_0 et G_0 sont suffisamment bonnes, on peut légitimement écrire $|F_0| \simeq |F|$ et $|G_0| \simeq |G|$, de sorte que, en remplaçant (5.2) dans (5.1), on voit que pour réaliser (5.1) il faut

$$|F_0 - F| \leq \frac{1}{\gamma^5 \cdot k^3(n-k)^2 \cdot (|F||G|)^3 \cdot |G|}. \tag{5.3}$$

Pour que cette dernière relation soit vérifiée, on utilise alors les majorations de $|F||G|$ données dans la section 2.2.

Remarques :

- Il reste encore à donner des approximations de γ et $M(P)$. Ces problèmes seront traités plus tard.
- En fait, les majorations données ici ne seront pas reprises telle quelles; nous verrons qu'en s'y prenant autrement, on peut obtenir une borne plus fine que (5.3). Un autre problème est que f et g ne seront pas déterminés exactement.

5.2 Le polynôme auxiliaire H

Dans le paragraphe précédent, un problème avait été laissé en suspens : celui de la détermination des termes correcteurs f et g. Rappelons que sous les hypothèses précédentes, on veut trouver f et g tels que :

$$P - F_0G_0 = fG_0 + gF_0 \iff \frac{P - F_0G_0}{F_0G_0} = \frac{f}{F_0} + \frac{g}{G_0}.$$

Un moyen de faire serait d'utiliser l'algorithme d'Euclide. Schönhage écarte cette hypothèse, estimant à juste titre que l'approche numérique serait peu fiable et demanderait un temps d'exécution important. Il contourne la difficulté en introduisant un polynôme auxiliaire H tel que $HG_0 \equiv 1$ [mod F_0] et $\deg(H) < k$. f est alors donné par $f \equiv HP$ [mod F_0] et g pris tel que $G_0 + g$ est le quotient de la division euclidienne de P par $F_0 + f$.

H se retrouve dans la décomposition partielle en éléments simples :

$$\frac{1}{F_0G_0} = \frac{H}{F_0} + \frac{L}{G_0} \qquad (\text{avec } \deg(L) < n-k).$$

Comment calculer H ? Là aussi, on pourrait trouver H en utilisant l'algorithme d'Euclide. Pour les raisons évoquées plus haut, on ne procédera pas comme cela; on utilisera plutôt la représentation intégrale donnée par le théorème 8

$$H(z) = \frac{1}{2i\pi} \oint_\Gamma \frac{1}{(F_0G_0)(t)} \frac{F_0(z) - F_0(t)}{z - t}\, dt. \tag{5.4}$$

5.2.1 Détermination initiale du polynôme auxiliaire

On va donc utiliser la relation (5.4) pour calculer numériquement une approximation H_0 de H. Si

$$F_0 = X^k + \varphi_1 X^{k-1} + \cdots + \varphi_{k-1} X + \varphi_k,$$

on obtient, en insérant les coefficients dans (5.4) :

$$H = \sum_{m=0}^{k-1} \left(\sum_{l=1+m}^{k} \varphi_{k-l} v_{l-m-1} \right) X^m, \tag{5.5}$$

où

$$\forall m, \quad 0 \leq m \leq k-1, \qquad v_m = \frac{1}{2i\pi} \oint_\Gamma \frac{t^m}{(F_0 G_0)(t)}\, dt. \tag{5.6}$$

Les v_m seront approximées par les quantités :

$$V_p = \frac{1}{N} \sum_{j=0}^{N-1} \frac{1}{(F_0 G_0)(\omega^j)} \omega^{pj}, \qquad \text{avec } N \in \mathbb{N}^* \text{ et } \omega = e^{2i\pi/N}. \tag{5.7}$$

Arrêtons nous ici pour faire la remarque suivante (que Schönhage ne semble pas avoir faite) : Lors du calcul de l'approximation F_0 de F, on avait à évaluer les quantités :

$$W_m = \frac{1}{N} \sum_{j=0}^{N-1} \frac{P'(\omega^j)}{P(\omega^j)} \omega^{mj}.$$

Il fallait donc évaluer les $1/P(\omega^j)$, ce qui ressemble fortement à $1/(F_0G_0)(\omega^j)$. L'idée est donc d'utiliser les approximations $1/P(\omega^j)$ de $1/(F_0G_0)(\omega^j)$, déjà calculées, pour approcher les v_m. On veut alors savoir jusqu'à quel point l'approximation H_0 de H ainsi calculée sera bonne. La réponse est donnée par le théorème suivant :

Théorème 14 *Avec* $F_0 = X^k + \varphi_1 X^{k-1} + \cdots + \varphi_{k-1} X + \varphi_k$, *si*

$$\forall p, \quad 1 \leq p \leq k, \qquad U_p = \frac{1}{N} \sum_{j=0}^{N-1} \frac{\omega^{pj}}{P(\omega^j)} \quad \text{avec } \omega = e^{2i\pi/N}, \tag{5.8}$$

et si

$$H_0 = \sum_{m=0}^{k-1} \left(\sum_{l=1+m}^{k} \varphi_{k-l} U_{l-m} \right) X^m, \tag{5.9}$$

alors :

$$|H_0 - H| < k|F_0| \cdot \gamma \left[2\left(1 + e^{\delta/2}\right)^n e^{-\delta(N-k)/2} + \frac{|P - F_0 G_0|}{\mu} \right],$$

où $\gamma = 1/2\pi \cdot \oint_\Gamma |dt/P(t)|$ *et* $\mu = \inf_{|z|=1} |P(z)|$.

Preuve

Posons

$$H_1(z) = \frac{1}{2i\pi} \oint_\Gamma \frac{1}{P(t)} \frac{F_0(z) - F_0(t)}{z - t}\, dt. \tag{5.10}$$

Comme pour H, on a la relation :

$$H_1 = \sum_{m=0}^{k-1} \left(\sum_{l=1+m}^{k} \varphi_{k-l} u_{l-m-1} \right) X^m, \tag{5.11}$$

où

$$\forall m, \qquad u_m = \frac{1}{2i\pi} \oint_\Gamma \frac{t^m}{P(t)}\, dt. \tag{5.12}$$

Écrivons le développement de Laurent de $1/P$ dans la couronne $\{e^{-\delta} < |z| < e^{\delta}\}$:

$$\frac{1}{P(t)} = \sum_{j\in\mathbb{Z}} d_j t^j.$$

En remplaçant dans la série, on a :

$$U_p = \sum_{j\in\mathbb{Z}} d_{-p+jN},$$

et donc, comme $u_p = d_{-p-1}$:

$$|U_{m+1} - u_m| = |\sum_{j\neq 0} u_{m+jN}| \leq \sum_{j=1}^{\infty} |u_{m+jN}| + \sum_{j=1}^{\infty} |u_{m-jN}|. \tag{5.13}$$

Le lemme 1 (voir plus loin) nous dit que :

$$\text{si } |t| = 1, \qquad |P(e^{\pm\delta/2}t)| \geq \frac{|P(t)|}{(1+e^{\delta/2})^n},$$

ce qui entraîne, par intégration le long des cercles $|z| = e^{\pm\delta/2}$:

$$|u_p| \leq (1+e^{\delta/2})^n \cdot e^{-|p|\delta/2} \cdot \frac{1}{2\pi} \oint_\Gamma \left|\frac{dt}{P(t)}\right|.$$

En remplaçant dans (5.13), on obtient donc :

$$\forall m, 0 \leq m \leq k-1, \qquad |U_{m+1} - u_m| \leq (1+e^{\delta/2})^n \cdot \gamma \cdot \left(\frac{e^{-\delta(m+N)/2}}{1-e^{-\delta N/2}} + \frac{e^{-\delta(N-m)/2}}{1-e^{-\delta N/2}}\right).$$

δN sera grand, de sorte que légitimement, cette majoration entraîne :

$$\forall m, 0 \leq m \leq k-1, \qquad |U_{m+1} - u_m| \leq 2(1+e^{\delta/2})^n \cdot \gamma \cdot e^{-\delta(N-m)/2}. \tag{5.14}$$

La convolution (5.9) pouvant amplifier les erreurs $|U_{m+1} - u_m|$ par un facteur $\sum_m \sum_{l\geq 1+m} |\varphi_{k-l}| \leq k|F_0|$ au plus, on voit alors que :

$$|H_1 - H_0| \leq 2k|F_0| \cdot \left(1+e^{\delta/2}\right)^n \cdot \gamma \cdot e^{\delta(N-m)/2}.$$

Ceci étant, on a facilement :

$$|H_1 - H| < |P - F_0G_0| \cdot \gamma \cdot \frac{1}{\mu} \cdot k|F_0|,$$

(où on a utilisé l'approximation $\inf_{|z|=1} |F_0G_0(t)| \simeq \inf_{|z|=1} |P(t)| = \mu$) d'où le résultat.□

Lemme 1 *Si P est un polynôme complexe de degré $n > 0$ n'ayant aucune racine dans la couronne $\{e^{-\delta} < |z| < e^{\delta}\}$, alors*

$$\forall z, |z| = 1, \qquad |P(e^{\pm\delta/2}z)| \geq \frac{|P(z)|}{(1+e^{\delta/2})^n}.$$

Preuve

Montrons par exemple le résultat sur le cercle $|z| = e^{-\delta/2}$. Soient $u_1, \ldots, u_n$ les racines de P numérotées telles que :

$$|u_1| \leq \cdots \leq |u_k| \leq e^{-\delta/2} < e^{\delta/2} \leq |u_{k+1}| \leq \cdots \leq |u_n|.$$

Le fait que :

$$\frac{P'(z)}{P(z)} = \sum_{j=1}^{n} \frac{1}{z-u_i}$$

entraîne :

$$\forall z,\ e^{-\delta/2} \leq |z| \leq 1, \qquad \left|\frac{P'(z)}{P(z)}\right| \leq \frac{n}{|z| - e^{-\delta}}.$$

Fixons z, $|z| = 1$. Si $\psi(t) = \log(|P(t \cdot z)|)$, on a donc

$$|\psi'(t)| = \left|\frac{P'(t \cdot z)}{P(t \cdot z)}\right| \leq \frac{n}{t - e^{-\delta}},$$

d'où par intégration :

$$\left|\log\left|\frac{P(z)}{P(e^{-\delta/2}z)}\right|\right| = \left|\log\left(\frac{\psi(1)}{\psi(e^{-\delta/2})}\right)\right| \leq 1 + e^{\delta/2},$$

d'où le lemme.□

5.2.2 Raffinement du calcul du polynôme auxiliaire

Comme lors du calcul de l'approximation de F, on ne va pas directement calculer l'approximation finale H_0 de H par la méthode précédente (ce serait trop coûteux). Si l'approximation initiale H_0 de H est suffisamment bonne, on saura par un algorithme d'itérations raffiner autant qu'on le voudra cette approximation. Plus précisément :

Hypothèses : Supposons calculé H_0 (à l'aide de la méthode précédente) tel que :

$$\epsilon_0 = |G_0||H - H_0| < 1. \tag{5.15}$$

Problème : Soit $\epsilon > 0$. On veut trouver $\hat{H}$ de degré $< k$ tel que :

$$\hat{H}G_0 \equiv 1 - D \ [\text{mod } F_0], \quad \text{(avec } D \text{ un polynôme) et} |D| < \epsilon.$$

Nous allons déterminer une suite (H_k) de polynômes approximants qui converge très vite vers H.

Algorithme 6 *1. On calcule le produit H_0G_0 puis on détermine (par division euclidienne) l'unique polynôme D_0 de degré $< k$ tel que $H_0G_0 \equiv 1 - D_0$ [mod F_0].*

2. On calcule le produit $\tilde{H}_1 = H_0(1 + D_0)$.

3. On détermine l'unique polynôme H_1 de degré $< k$ tel que $H_1 \equiv \tilde{H}_1$ [mod F_0].

4. De manière générale, H_m étant connu, on détermine D_m de degré $< k$ tel que $H_mG_0 \equiv 1 - D_m$ [mod F_0]. On détermine ensuite H_{m+1} de degré $< k$ tel que $H_{m+1} \equiv H_m(1 + D_m)$ [mod F_0].

5. On s'arrête au rang m dès que $|D_m| < \epsilon$.

Étude de la convergence de l'algorithme.
Posons $D = (H - H_0)G_0$. On a $H_0G_0 \equiv 1 - D$ [mod F_0], donc $D_0 \equiv D$ [mod F_0]. On en déduit :

$$H_1G_0 \equiv H_0G_0(1 + D_0) \equiv H_0G_0(1 + D) \equiv (1 - D)(1 + D) \equiv 1 - D^2 \ [\text{mod } F_0],$$

donc

$$D_1 \equiv D^2 \ [\text{mod } F_0].$$

De manière générale, on pourrait montrer que :

$$D_m \equiv D^{2^m} \ [\text{mod } F_0].$$

En multipliant par G_0 et en appliquant le corollaire du théorème 8 ($\deg(D_m) < k$), on a, avec les notations précédentes :

$$|D_m| \leq k \cdot \gamma \cdot |F_0| \cdot |G_0 D^{2^m}|,$$

donc :

$$|D_m| \leq k \cdot \gamma \cdot |F_0| \cdot |G_0| \cdot \epsilon_0^{2^m}. \tag{5.16}$$

Comme $\epsilon_0 < 1$, la convergence est assurée.

Remarque :

- On peut montrer que chaque calcul, au rang m, peut être effectué avec une précision variable $\epsilon_m \sim |D_m|^2/(k\gamma|F_0| \cdot |G_0|)$.
- Schönhage utilise lui, à chaque itération, la majoration

  $$|D_m| \leq k \cdot \gamma \cdot |F_0| \cdot |G_0| \cdot |D_{m-1}|^2.$$

  Cette majoration prise telle quelle au rang m est moins bonne que (5.16) : pour assurer la convergence (par exemple $|D_m| < |D_{m-1}|^{1.5}$), elle impose que

  $$|D_{m-1}| < \frac{1}{k^2 \cdot \gamma^2 \cdot |F_0|^2 \cdot |G_0|^2}.$$

  Cette inégalité doit donc être vérifiée dès la première itération. Ceci impose un calcul de l'approximation initiale H_0 beaucoup plus fin que le notre, entraînant ainsi des calculs plus coûteux.

Chapitre 6

Description complète de la factorisation à partir du cercle de séparation

Dans les deux sections précédentes, nous avons donné les bases d'une méthode permettant de calculer une factorisation de P à partir du cercle de séparation. Il nous reste à combiner ces techniques pour construire proprement un algorithme qui saura nous factoriser P avec une précision arbitraire fixée à l'avance. Résumons la situation.

Hypothèses : P est un polynôme à coefficients complexes de degré $n > 0$. La recherche d'un cercle de séparation Γ (on peut supposer que Γ est le cercle unité, quitte à effectuer une transformation affine) nous a donné $k \in \mathbb{N}, 0 < k < n$ et $\delta > 0$ tels que les racines de P peuvent s'écrire $u_1, \ldots, u_n$ avec :

$$|u_1| \leq \cdots \leq |u_k| \leq e^{-\delta} < e^{\delta} \leq |u_{k+1}| \leq \cdots \leq |u_n|.$$

Problème : Si $F = (X - u_1)\cdots(X - u_k)$ et $G = \beta(X - u_{k+1})\cdots(X - u_n)$ tels que $P = FG$, on veut, $\epsilon > 0$ étant donné, trouver deux approximations $\widehat{F}$ et $\widehat{G}$ de F et G telles que :

$$|P - \widehat{F}\widehat{G}| < \epsilon |P|.$$

Remarquons que ces hypothèses entraînent l'égalité des mesures : $M(G) = M(P)$. Quitte à diviser P par $|P|$, on peut supposer $|P| = 1$. Enfin, quitte à considérer les polynômes réciproques, on peut supposer $k \leq n - k$.

- La première étape (cf. 6.1) consiste à déterminer, grâce aux méthodes de la section 4, une approximation "assez grossière" F_0 de F. Par division euclidienne de P par F_0, on détermine une approximation G_0 de G. En même temps est déterminé, grâce aux techniques de la section 5.2.1, une approximation initiale H_0 de H (cf. 6.2).
- La factorisation sera ensuite raffinée (cf. 6.3) par la méthode de Newton-Schönhage dont les bases sont décrites dans la section précédente.

6.1 Factorisation initiale

Pour assurer la convergence de l'algorithme de Newton-Schönhage, nous allons voir qu'il suffit de déterminer une approximation de F_0 de F telle que :

$$|F_0 - F| \leq \frac{\mu}{100 \cdot k^2(n-k)^{3/2} \cdot \gamma^{5/2} \cdot (|F| \cdot |G|)^2 |G|} \tag{6.1}$$

où $\mu = \inf_{|z|=1} |P(z)|$, et $\gamma = \frac{1}{2\pi} \oint_\Gamma |dt/P(t)|$. (Le calcul de μ et γ sera discuté ultérieurement). Nous ne connaissons pas *a priori* les valeurs de $|F|$ et $|G|$, mais les résultats de la section 2.2 nous permettront d'en donner des majorations.

Algorithme 7 (Calcul de F_0 et G_0) *1. Si*

$$\alpha = \min\left(2^{n/2}\sqrt{\binom{n}{k}}\,[P]_2, 2^n M(P)\right)$$

(voir section 2.1.1 pour la définition de $[P]_2$, 2.1.2 pour celle de $M(P)$; le calcul de $M(P)$ sera discuté plus tard), on choisit un entier N tel que :

$$N \geq k + \frac{1}{\delta}\left(\log 100 + \log(k^3(n-k)^{3/2}n) + \frac{5}{2}\log\gamma + \log\left(\frac{1}{\mu}\right) + 3\log\alpha\right).$$

2. Si $\omega = e^{2i\pi/N}$, on calcule ensuite :

$$W_m = \frac{1}{N}\sum_{j=0}^{N-1} \frac{P'(\omega^j)}{P(\omega^j)}\omega^{mj} \qquad (2 \leq m \leq k+1)$$

puis on calcule successivement les φ_j par :

$$\varphi_m = -\frac{1}{m}(W_2\varphi_{m-1} + \cdots + W_m\varphi_1 + W_{m+1}) \qquad (1 \leq m \leq k).$$

En vue de (6.1), tout ces calculs peuvent être menés avec une précision

$$\epsilon_0 \sim \frac{\mu^2}{100k^2(n-k)^{3/2}\gamma^{5/2}\cdot\alpha^3}.$$

3. $F_0 = X^k + \varphi_1 X^{k-1} + \cdots + \varphi_k$ vérifie alors (6.1). On détermine ensuite G_0 par division euclidienne de P par F_0.

Le fait que cet algorithme nous donne bien la majoration (6.1) provient de l'étude effectuée à la section 4 et de la majoration $|F|\cdot|G| \leq \alpha$ (voir 2.2, théorèmes 3 et 4).

Par la suite, nous aurons besoin d'avoir des majorations de $|P - F_0G_0|$ et de $|G_0 - G|$. Ces dernières sont données par le lemme qui suit :

Lemme 2 *Si F_0 vérifie (6.1) et si G_0 est déterminé par division euclidienne de P par F_0 alors :*

$$|P - F_0G_0| \leq \frac{\mu}{100\cdot k(n-k)^{3/2}\cdot\gamma^{3/2}\cdot|F|\cdot|G|} \tag{6.2}$$

et

$$|G - G_0| \leq \frac{\mu}{100\cdot(n-k)^{1/2}\cdot\gamma^{1/2}\cdot|F|}. \tag{6.3}$$

Preuve

On a $P - F_0G_0 \equiv P - F_0G \pmod{F_0}$, donc $G(P - F_0G_0) \equiv G^2|F - F_0| \pmod{F_0}$. Nous voulons appliquer le corollaire du théorème 8. Pour cela, il faut que tous les zéros de F_0 soient de module < 1. Ceci est le cas car

$$\forall z, |z| = 1, \quad |F(z)| \geq \frac{|P(z)|}{|G|} \geq \frac{\mu}{|G|},$$

$$\text{si } |z| = 1, \quad |F(z) - F_0(z)| \leq |F - F_0| < \frac{\mu}{|G|}$$

(d'après (6.1)) et d'après le théorème de Rouché. Comme de plus $\deg(P - F_0G_0) < k$, nous pouvons appliquer le corollaire du th.8 qui nous donne (6.2) (on a utilisé les approximations $\oint_\Gamma |dt/(F_0G)(t)| \simeq \gamma$ et $|F_0| \simeq |F|$, légitimes puisque l'on a (6.1)). Maintenant, on a $F_0(G - G_0) \equiv P - F_0G_0 \ [\mathrm{mod}\ G]$ et toujours d'après le corollaire (et sa remarque !), on obtient (6.3).

Remarques :

- Outre l'approximation $|F_0| \simeq |F|$ déjà utilisée, (6.3) nous montre que l'approximation $|G_0| \simeq |G|$ est également valable.
- On a vu dans la démonstration du lemme que F_0 avait toutes ses racines de module < 1. (6.3) entraîne :

$$\forall z, |z| = 1, \quad |G(z) - G_0(z)| \leq |G - G_0| < \frac{\mu}{|F|} \leq |G(z)|$$

et d'après le théorème de Rouché, G_0 a toutes ses racines de module > 1. Pour F_0 comme pour G_0, on pourra donc en toute quiétude appliquer le corollaire du théorème 8.

6.2 Détermination initiale d'une approximation du polynôme auxiliaire H

Rappelons, pour assurer la convergence de l'algorithme de 5.2.2, que H_0 doit être déterminé tel que $\epsilon_0 = |G_0| \cdot |H - H_0| < 1$. Pour que la convergence soit assez rapide au début, on va calculer H_0 de sorte que $\epsilon_0 < 1/100$.

Algorithme 8 **Calcul de H_0**

1. *Si comme précédemment*

$$\alpha = \min\left(2^{n/2}\sqrt{\binom{n}{k}}\,[P]_2, 2^n M(P)\right),$$

on choisit un entier N' tel que

$$N' \geq k + \frac{2}{\delta}\left(\log(200) + n\log(1 + e^{\delta/2}) + \log(k) + \log(\gamma) + \log(\alpha)\right).$$

2. *On calcule ensuite H_0 par les formules données dans le théorème 14, avec $w = e^{2i\pi/N'}$. En vue de la borne $|G_0| \cdot |P - P_0| < 1/100$, ces calculs pourront être effectués avec une précision*

$$\epsilon \sim \frac{\mu}{200 \cdot k \cdot \alpha}.$$

3. *H_0 ainsi déterminé vérifie $|G_0| \cdot |H - H_0| \leq 1/100$*

On a bien, aprés calculs, $|G_0| \cdot |H - H_0| < 1/100$ d'aprés le théorème 14 et la majoration (6.2).

Remarque.

- Dans la pratique, N (voir l'algorithme précédent) et N' sont proches. Les deux algorithmes peuvent alors être effectués avec le plus grand des deux entiers; ainsi, lors du calcul des formules (5.8), les valeurs de $1/P(\omega^j)$, déjà calculées lors de l'algorithme précédent, peuvent être reprises telles quelles ici.

6.3 L'algorithme de Newton-Schönhage

Tous les paramètres précédents avaient été choisis pour assurer la convergence de l'algorithme qui suit :

Hypothèses : On connait F_0 vérifiant (6.1),G_0 vérifiant (6.2) et H_0 vérifiant $|G_0| \cdot |H - H_0| < 1/100$.

Problème : $\epsilon > 0$ étant donné, on veut trouver $\widehat{F}$ et $\widehat{G}$ tels que

$$|P - \widehat{F}\widehat{G}| < \epsilon |P|.$$

Algorithme 9 *1. On évalue $\epsilon_0 = |P - F_0G_0|$. Grâce à l'algorithme de raffinement du calcul du polynôme H, on détermine H_1 tel que*

$$G_0H_1 \equiv 1 - D \ [\mathrm{mod}\ F_0], \qquad \text{avec } |D| < \epsilon_0.$$

2. On calcule l'unique polynôme f_0 de degré $< k$ tel que

$$f_0 \equiv H_1P \equiv H_1(P - F_0G_0) \ [\mathrm{mod}\ F_0].$$

3. On pose $F_1 = F_0 + f_0$. On détermine G_1 par division euclidienne de P par F_1.

4. On recommence, en partant cette fois de F_1, G_1, H_1 et $\epsilon_1 = |P - F_1G_1|$.

5. On s'arrête au rang m lorsque $|P - F_mG_m| < \epsilon|P|$.

Étude de la convergence de l'algorithme :

Les majorations d'erreurs que nous allons effectuer sont différentes selon que l'on effectue la première ou les étapes suivantes (et c'est celà qui nous permet d'utiliser des paramètres moins couteux que ceux donnés par Schönhage, en particulier les majorations (6.6) et (6.7), plus fines que celles données par Schönhage à la première étape).

- Première étape : Si $g_0 = G_1 - G_0$, de sorte que $G_1 = G_0 + g_0$, on a déterminé g_0 tel que :

$$(P - F_0G_0) - f_0G_0 = g_0F_0 + R, \qquad \text{avec } \deg(R) < k \tag{6.4}$$

On en déduit donc

$$\epsilon_1 = |P - F_1G_1| \leq |R| + |f_0| \cdot |g_0|. \tag{6.5}$$

Donnons maintenant une majoration de f_0. Le fait que

$$f_0G_0 \equiv (H_1G_0)P \equiv (1 - D)(P - F_0G) \equiv G(F - F_0)(1 - D) \ [\mathrm{mod}\ F_0]$$

entraîne, d'après le corollaire du théorème 8 :

$$|f_0| \leq k\gamma|F_0| \cdot |G| \cdot |F - F_0| \cdot (1 + |D|) \simeq k\gamma|F_0| \cdot |G| \cdot |F - F_0|. \tag{6.6}$$

De manière similaire, (6.4) donne $R \equiv -DP$ [mod F_0], et donc

$$RG_0 \equiv -D(P - F_0G)G_0 \equiv -DG(F - F_0)G_0 \ [\mathrm{mod}\ F_0],$$

ce qui donne, grâce au même corollaire :

$$|R| \leq k\gamma|F_0| \cdot |D| \cdot |G| \cdot |G_0| \cdot |F - F_0| < k\gamma|F_0|\epsilon_0|G| \cdot |G_0| \cdot |F - F_0|. \tag{6.7}$$

Reste à majorer $|g_0|$. (6.4) entraîne

$$g_0F_0 \equiv P - F_0G_0 \ [\mathrm{mod}\ G_0],$$

et donc, toujours grâce au corollaire du théorème 8 (et sa remarque) :

$$|g_0| \leq (n - k)\gamma|G_0| \cdot |P - F_0G_0| \simeq (n - k)\gamma|G|\epsilon_0.$$

En combinant toutes ces majorations, on a grâce à (6.5) :

$$\epsilon_1 \leq k\gamma|F| \cdot |G|^2(1 + (n - k)\gamma)\epsilon_0|F - F_0|.$$

(Les approximations $|F_0| \simeq |F|$ et $|G_0| \simeq |G|$ ont été utilisées.)
Avec (6.1) et la majoration de ϵ_0 donnée par (6.2), ceci donne :

$$\epsilon_1 < \frac{\mu^2}{5000k^2(n-k)^2\gamma^2(|F|\cdot|G|)^2}. \tag{6.8}$$

Ce n'est pas fini ! Il nous reste à vérifier que H_1 est bien adapté à F_1, G_1, i.e que $H_1G_1 \equiv 1 - D'$ [mod F_1] avec $|D'| < 1/100$ (ceci pour assurer la convergence de l'algorithme de raffinement de H_1 pour l'étape suivante). Rappelons que H_1 avait été déterminé tel que

$$H_1G_0 + QF_0 = 1 - D, \qquad \text{avec } Q \text{ un polynôme et } |D| < \epsilon_0.$$

Ceci entraîne $H_1G_0 \equiv 1 - D$ [mod F_0] et par application du corollaire du théorème 8 :

$$|H_1| \leq k\gamma(1 + |D|)|F_0| \simeq k\gamma|F|.$$

De même, de $QF_0 \equiv 1 - D$ [mod G_0], on tire

$$|Q| \leq (n-k)\gamma|G|.$$

Il reste alors à remarquer que si $D' = D - H_1g_0 - Qf_0$, on a $H_1G_1 \equiv 1 - D'$ [mod F_1], et que les majorations précédentes entraînent :

$$|D'| \leq |D| + |H_1|\cdot|g_0| + |Q|\cdot|f_0|$$
$$\leq \epsilon_0 + \frac{\gamma^{1/2}\mu}{100(n-k)^{1/2}} + \frac{\mu}{100k(n-k)^{1/2}\gamma^{1/2}|F|\cdot|G|}.$$

Les remarques du type $1/\gamma < \mu < 1$ et $|F|\cdot|G| \geq |P| = 1$ montrent alors que $|D'| < 1/100$.

- Étapes suivantes : (par exemple pour le calcul de $F_2 = F_1 + f_1$ et $G_2 = G_1 + g_1$.) On ne peut pas s'y prendre de la même manière puisque l'on ne connaît pas *a priori* de majoration sympathique de $|F - F_1|$. On va donc tout majorer en fonction de $\epsilon_1 = |P - F_1G_1|$. Comme précédemment, si R désigne le reste de la division euclidienne de P par F_1, on a la majoration

$$\epsilon_2 = |P - F_2G_2| \leq |R| + |f_1|\cdot|g_1|.$$

Pour majorer f_1, la méthode est légèrement différente. On écrit, si $H_2G_1 \equiv 1 - D$ [mod F_1] (avec $|D| < \epsilon_1$) : $f_1G_1 \equiv (P - F_1G_1)(1 - D)$ [mod F_0], ce qui entraîne, en appliquant le fameux corollaire :

$$|f_1| \leq k\gamma|F_1|\epsilon_1(1 + |D|) \simeq k\gamma|F|\epsilon_1.$$

Pour R, de $RG_1 \equiv -D(P - F_1G_1)G_1$ [mod F_1] on tire :

$$|R| \leq k\gamma\epsilon_1^2|F_1|\cdot|G_1|.$$

La majoration de $|g_1|$ est identique à celle de $|g_0|$:

$$|g_1| \leq (n-k)\gamma|G|\epsilon_1.$$

En combinant tout ces résultats, en utilisant (6.8), on montre que :

$$\epsilon_1 < \epsilon_0^{1.5}.$$

Enfin, par des méthodes strictement analogues aux précédentes, on montre également que H_2 est un bon candidat pour l'itération suivante.

- L'algorithme de Newton-Schönhage converge donc.□

Remarques :

- Typiquement, 2 ou 3 itérations de Newton suffiront pour obtenir la factorisation précise désirée.
- En toute rigueur, il faudrait également vérifier que comme pour F et G, les racines de F_1 sont de module < 1, celles de G_1 de module > 1. Les mêmes méthodes que celles utilisées pour F_0 et G_0 confirment ce fait.

Chapitre 7

Calcul approché de μ, γ et $M(P)$

Dans la section précédente, les paramètres utilisés sont exprimés (entre autres) en fonction de

$$\mu = \inf_{|z|=1} |P(z)|, \quad \gamma = \frac{1}{2\pi} \oint_\Gamma \left| \frac{dt}{P(t)} \right|,$$

et $M(P)$, la mesure de Mahler de P (voir 2.1.2). (On rappelle que Γ désigne le cercle unité.) Il nous faut donc donner des valeurs approchées de ces valeurs. Le principe de la méthode sera de calculer la valeur de P en des points régulièrement répartis sur le cercle unité, sous les hypothèses décrites au début de la section précédente. Pour majorer l'erreur, nous aurons besoin d'un résultat qui fait l'objet du paragraphe ci dessous.

7.1 Un lemme utile

Lemme 3 *Soient $e^{i\theta_0}$ et $e^{i\theta}$ deux nombres complexes sur le cercle unité. Alors, si P est un polynôme de degré $n > 0$ n'ayant aucune racine dans la couronne $\{e^{-\delta} < |z| < e^{\delta}\}$, on a :*

$$\left| \log \left| \frac{P(e^{i\theta_0})}{P(e^{i\theta})} \right| \right| \leq |\theta - \theta_0| \frac{ne^{\delta}}{\delta}.$$

Preuve

Les hypothèses sur P entraînent (si k désigne le nombre de racines de P dans le disque unité)

$$\forall z,\ |z| = 1, \qquad \left| \frac{P'(z)}{P(z)} \right| \leq \frac{k}{1 - e^{-\delta}} + \frac{n-k}{e^{\delta} - 1} \leq \frac{ne^{\delta}}{\delta}.$$

Si $\psi(t) = \log(|P(e^{it})|$, on a donc :

$$|\psi'(t)| \leq \left| \frac{P'(e^{it})}{P(e^{it})} \right| \leq \frac{ne^{\delta}}{\delta},$$

d'où par intégration

$$\left| \log \left| \frac{P(e^{i\theta_0})}{P(e^{i\theta})} \right| \right| = \left| \log \left(\frac{\psi(\theta_0)}{\psi(\theta)} \right) \right| \leq |\theta - \theta_0| \frac{ne^{\delta}}{\delta},$$

d'où le lemme.□

7.2 Calcul approché de $\mu = \inf_{|z|=1} |P(z)|$

Il nous faut en fait avoir une minoration de μ. Les hypothèses sur P entraînent, si β est le coefficient dominant de P :

$$\mu \geq |\beta|(1 - e^{-\delta})^k (e^{\delta} - 1)^{n-k}. \tag{7.1}$$

Cependant, cette minoration est souvent très pessimiste. Il est donc intéressant de calculer une valeur approchée μ_0 de μ. La quantité décisive dans les paramètres est $\log(\mu)$, et dans cette optique, nous allons calculer un μ_0 de sorte que $|\log(\mu_0/\mu)| < 1$. La méthode à adopter est contenue dans le théorème suivant :

Théorème 15 *Soit un entier N tel que $N > \pi n e^{\delta}/\delta$. Alors si*

$$\mu_0 = \inf_{0 \le k \le N-1} |P(\omega^k)|, \qquad (\text{où } \omega = e^{2ik\pi/N}),$$

on a :

$$\left|\log\left(\frac{\mu_0}{\mu}\right)\right| < 1.$$

Une minoration de μ est donc donnée en considérant μ_0/e.

Preuve

La compacité de $\Gamma = \{z,\ |z| = 1\}$ entraîne l'existence de $z_0 = e^{i\theta_0}$ tel que $|P(e^{i\theta_0})| = \mu$. Or il existe k, $0 \le k \le N-1$, tel que $|2\pi k/N - \theta_0| \le \pi/N$. D'après le lemme 2, on a donc :

$$0 \le \log\left(\frac{\mu_0}{\mu}\right) \le \log\left|\frac{P(\omega^k)}{P(e^{i\theta_0})}\right| \le \frac{\pi}{N}\frac{ne^{\delta}}{\delta},$$

d'où le résultat vue la minoration vérifiée par N.□

Remarque : En vue de la minoration (7.1), les calculs de $P(\omega^k)$ pourront être menés avec une précision $\epsilon \sim |\beta|\delta^n|P|$.

7.3 Calcul approché de $\gamma = \frac{1}{2\pi}\oint_\Gamma |dt/P(t)|$

L'idée est toute simple. Avec la valeur précédente de N, on calcule une valeur approchée γ_0 de γ donnée par

$$\gamma_0 = \frac{1}{N}\sum_{k=0}^{N-1}\frac{1}{|P(\omega^k)|}.$$

Les mêmes techniques que celles utilisées précédemment montrent que l'on a $|\log(\gamma_0/\gamma)| < 1$. Une majoration de γ est donc obtenue en considérant $\gamma_0 \cdot e$.

7.4 Calcul approché de M(P)

On cherche à donner une majoration de la mesure de Mahler de P, $M(P)$. On utilise pour cela la formule de Jensen (c.f 2.1.2). Avec les paramètres déjà utilisés précédemment, on calcule ainsi une approximation M_0 de $M(P)$ par la relation :

$$M_0 = \exp\left(\frac{1}{N}\sum_{k=0}^{N-1}\log|P(\omega^k)|\right).$$

On peut montrer, de manière analogue à ce qui a déjà été fait précédemment, que $|\log(M_0/M(P))| < 1$. Une majoration de $M(P)$ est donc obtenue en considérant $M_0 \cdot e$.

Remarques.

- La même valeur N est utilisée pour chacune des trois étapes de calcul de μ, γ et $M(P)$. Les valeurs $P(\omega^k)$ (calculées une seule fois) sont donc utilisées pour ces trois calculs.
- Comme dans le calcul de μ, on peut utiliser dans les calculs de γ et $M(P)$ une précision $\epsilon \sim |\beta|\delta^n|P|$.

Chapitre 8

Homothéties et translations

Dans les algorithmes précédents, (et en particulier lors de la recherche du cercle de séparation), nous avions à effectuer sur le polynôme P des transformations homothétiques ($P(X) \mapsto P(\rho X)$) et des translations ($P(X) \mapsto P(X+c)$). Lors de ces opérations, il faut prendre garde à la propagation des erreurs. Pour étudier ce problème, introduisons les notations suivantes :

$$\forall n \in \mathbb{N}^*, \qquad \text{on note } \mathbb{C}_n[X] = \{P \in \mathbb{C}[X], \deg(P) \leq n\};$$

$\forall \rho > 0$, on note H_ρ l'opérateur :

$$H_\rho : \mathbb{C}_n[X] \to \mathbb{C}_n[X] \quad P(X) \mapsto P(\rho X);$$

enfin, $\forall c \in \mathbb{C}$, on note T_c l'opérateur :

$$T_c : \mathbb{C}_n[X] \to \mathbb{C}_n[X] \quad P(X) \mapsto P(X+c).$$

Les normes de ces opérateurs sur $\mathbb{C}_n[X]$ sont les suivantes :

$$|||H_\rho||| = \sup_{|P|\leq 1, P\in\mathbb{C}_n[X]} |H_\rho(P)| = \sup\{1, \rho^n\}$$

et

$$|||T_c||| = \sup_{|P|\leq 1, P\in\mathbb{C}_n[X]} |T_c(P)| = (1+|c|)^n.$$

Examinons maintenant en détail la manière de calculer ces opérations lors de la recherche du cercle de séparation (cette section peut être sautée en première lecture).

On se donne un polynôme $P = P_0$ de degré $n > 1$ et $\epsilon_0 > 0$. On veut trouver deux facteurs approximatifs F_0 et G_0 de P tels que :

$$|P_0 - F_0 G_0| < \epsilon |P|.$$

Algorithme 10 *1. Si le coefficient constant de P est inférieur en module à $\epsilon|P|$, on a trouvé une factorisation de P avec $F_0 = X$ et $G_0 = (P - P(0))/X$.*

2. Si ce n'est pas le cas, on commence par calculer avec une erreur relative de 0.05 la valeur du rayon des racines de P : $R \simeq r_n(P)$.

3. Si $R < 1.9$, on se rend à l'étape 4; sinon $R \geq 1.9$. On calcule alors avec une erreur relative de 0.05 le rayon des racines du polynôme réciproque P^* : $R' \simeq r_n(P^*) = 1/r_1(P)$. Si $R' < 1.9$, on se rend à l'étape 4 en partant cette fois du polynôme réciproque P^*. Sinon, c'est que $R' \geq 1.9$. Donc :

$$r_1(P) \leq \frac{1.05}{R'} \leq 0.55 < 1.8 \leq \frac{R}{1.05} \leq r_n(P).$$

$\Delta = \log(r_n/r_1)$ *est suffisamment grand. On va donc choisir l'origine comme centre du cercle de séparation. Le rayon* ρ *du cercle est déterminé grâce à l'algorithme 5. On calcule alors le polynôme* $P_1(X) = P_0(\rho X) = H_\rho(P_0)$, *de sorte que le cercle de séparation est le cercle unité. On embraye sur les algorithmes donnés à la section 6 qui nous donnent une factorisation* $P_1 \simeq F_1 G_1$. *Il suffit alors de prendre* $F_0 = H_{1/\rho}(F_1)$ *et* $G_0 = H_{1/\rho}(G_1)$. *Compte tenu de la norme de l'opérateur* $H_{1/\rho}$, *la factorisation à partir des algorithmes de la section précédente devra être effectuée de sorte que*

$$|P_1 - F_1 G_1| < \epsilon_1 |P_1|, \qquad \text{avec } \epsilon_1 = \frac{\epsilon |P_0|}{|H_\rho(P_0)|} \sup\{1, \rho^n\}.$$

Enfin, les calculs de F_0 *et* G_0 *à partir de* F_1 *et* G_1 *pourront être menés avec une précision :*

$$\epsilon' \sim \epsilon \frac{|P_0|}{|F_1| \cdot |G_1|} \sup\{1, 1/\rho^n\}.$$

4. *Pour cette étape et les suivantes, on est en présence d'un polynôme* P_0 *dont le module des racines vérifie* $r_n(P_0) < 1.05 * R < 2$. *On calcule ensuite* $P_1(X) = P_0(4X) = H_4(P_0)$ *de sorte que* $r_n(P_1) < 0.5$. *Ce calcul peut être mené avec une précision* $\epsilon'_1 = \epsilon 4^n$.

5. *Le barycentre b des racines de* P_1 *vérifie donc* $|b| < 0.5$. *On calcule alors* $P_2(X) = P_1(X + b) = T_b(P_1)$. *On a* $|||T_b||| = (1 + |b|)^n$, *les calculs peuvent donc être effectués avec une précision* $\epsilon'_2 \sim \epsilon |P|/|P_1| \cdot (1 + |b|)^n$.

6. *Si le terme constant de* P_2 *est en module inférieur à* $\epsilon |P_0|/(1 + |b|)^n$, *alors on a trouvé une factorisation de* P_2 *avec* $F_2 = X$ *et* $G_2 = (P_2 - P_2(0))/X$. *On calcule alors* $F_0 = H_{1/4}(T_{-b}(F_2))$ *et* $G_0 = H_{1/4}(T_{-b}(G_2))$. *Ces derniers calculs peuvent être menés avec une précision* $\epsilon' \sim \epsilon/(1 + |b|)^n$. *Sinon :*

7. *On calcule le rayon des racines de* P_2 *à 0.01 prés en erreur relative :* $r \simeq r_n(P_2)$, *puis on calcule* $P_3(X) = 1/r^n P_2(rX) = 1/r^n H_r(P_2)$. *Une étude technique montre que ces calculs peuvent être menés avec une précision* $\epsilon'_3 \sim \epsilon |P|/|P_2|/2^n$.

8. *Si* $v_0 = 2, v_1 = 2i, v_2 = -2$ *et* $v_3 = -2i$, *on calcule, pour tout* j, $0 \leq j \leq 3$, *le rapport* $r_n[P_3(X + v_j)]/r_1[P_3(X + v_j)]$ *(où le calcul des rayons est effectué grâce à l'algorithme 2 avec une erreur relative de 0.05), puis on choisit* j *tel que ce rapport soit maximum. Nous avons ainsi déterminé le centre* v_j *du cercle de séparation. On calcule alors* $P_4(X) = P_3(X + v_j) = T_{v_j}(P_3)$. *Les calculs peuvent être menés avec une précision* $\epsilon'_4 \sim \epsilon |P|/|P_3|/(3(1 + |b|))^n$.

9. *On recherche le rayon* R_0 *du cercle de séparation grâce à l'algorithme 5. On calcule ensuite* $P_5 = H_{R_0}(P_4)$ *(avec une précision* $\epsilon'_4 \sim \epsilon |P|/|P_4|/(3R_0(1 + |b|))^n$*). Ainsi transformé, le cercle de séparation devient le cercle unité.*

10. *Si* $j > n/2$ *on considère le polynôme réciproque* P_5^* *(pour avoir moins de points dans le cercle unité qu'en dehors). Grâce aux algorithmes décrits dans la section 6, on calcule ensuite deux polynômes* F_5 *et* G_5 *tels que* $|P_5 - F_5 G_5| < \epsilon' |P_5|$, *avec*

$$\epsilon' \sim \frac{|P|}{|P_5| 3^n (1 + |b|)^n}.$$

11. *Il ne nous reste plus qu'à remonter, en traitant là aussi convenablement les problèmes de précision. On calcule* $F_4 = H_{1/R_0}(F_5)$ *et* $G_4 = H_{1/R_0}(G_5)$, *(où les calculs sont menés avec une précision* $\hat{\epsilon_4} \sim |P|/(|F_5| \cdot |G_5| \cdot 3^n (1 + |b|)^n)$.

12. *On calcule* $F_3 = T_{-v_j}(F_4)$ *et* $G_3 = T_{-v_j}(G_4)$ *(avec ici une précision* $\hat{\epsilon_3} \sim |P|/(|F_4| \cdot |G_4| \cdot 3^n (1 + |b|)^n)$.

13. *On calcule* $F_2 = r^j H_{1/r}(F_3)$ *et* $G_2 = r^{n-j} H_{1/r}(G_3)$ *(ici, la précision utilisée dans les calculs est* $\hat{\epsilon_2} \sim |P|/(|F_3| \cdot |G_3| \cdot (1+|b|)^n)$.*)*

14. *On calcule* $F_1 = T_{-b}(F_2)$ *et* $G_1 = T_{-b}(G_2)$ *(ici,* $\hat{\epsilon_1} \sim |P|/(|F_2| \cdot |G_2| \cdot (1+|b|)^n)$.*)*

15. *On calcule enfin* $F_0 = H_{1/4}(F_1)$ *et* $G_0 = H_{1/4}(G_1)$ *(précision utilisée :* $\hat{\epsilon_1} \sim |P|/(|F_1| \cdot |G_1|)$.*)*

16. *On a alors* $|P - F_0 G_0| < \epsilon |P|$.

Le fait que l'on a bien la majoration voulue résulte d'une étude pénible sur les amplifications d'erreurs dues aux opérateurs H et T.

Remarque :

- Les opérations d'homothéties et de translation sont coûteuses. Il est donc très intéressant (voire indispensable lorsque le degré du polynôme est élevé) de mettre le polynôme sous sa forme de Horner, avant de lui appliquer l'un des opérateurs H ou T.
- L'étape 8 également est coûteuse. Il n'est en fait pas toujours nécessaire de calculer tous les rapports r_n/r_1; si pour un v_j, ce rapport est plus grand qu'une certaine valeur (par exemple 2), alors $\Delta = \log(r_n/r_1)$ n'est pas trop petit et on peut directement passer à l'étape suivante sans calculer les autres rapports.
- Nous verrons à la section suivante un algorithme qui s'insère entre les étapes 9 et 10, et qui aura pour effet de diminuer le coût de l'étape 10.

Chapitre 9

Une transformation homographique sympathique

Le problème est le suivant : le cercle de séparation Γ étant déterminé et ramené au cercle unité, peut-on effectuer sur le polynôme P une transformation qui écarte les racines du cercle ? Une possibilité serait de considérer le polynôme P_1 =Graeffe(P) dont les racines sont les carrés des racines de P. Le problème est que l'on ne sait pas correctement déterminer l'opérateur inverse de Graeffe. Cette méthode est donc inutilisable. Il se trouve que dans le cas où le centre du cercle de séparation est déterminé par l'algorithme 4 (i.e si dans l'algorithme précédent, à l'étape 3, on a $r_n(P) < 2$ ou $r_1(P) > 1/2$), on peut utiliser une transformation homographique qui écarte les racines de Γ (on sait déterminer l'inverse d'un tel opérateur). Ainsi, une amélioration non négligeable est donnée à la méthode que Schönhage décrit dans [20].

Hypothèses :

Le centre du cercle de séparation est déterminé grâce à l'algorithme 4. Dans l'étape 3 de cet algorithme, on a choisi un centre $v \in \{2, 2i, -2, -2i\}$, par exemple $v = -2$. À l'étape 4, ce centre est translaté en 0. On est ainsi en présence d'un polynôme P dont toutes les racines se trouvent dans le disque $|z - 2| \leq 1$, donc dans le secteur

$$\mathcal{S} = \left\{ \rho e^{i\theta}, \quad \rho > 0 \text{ et } -\frac{\pi}{6} \leq \theta \leq \frac{\pi}{6} \right\}.$$

Le rayon R du cercle est ensuite déterminé grâce à l'algorithme 5. Après la transformation homothétique qui ramène ce rayon égal à 1, les racines se trouvent toujours dans le secteur $\mathcal{S}$. Finalement, la distribution des racines u_i de P est typiquement représentée par la figure suivante :

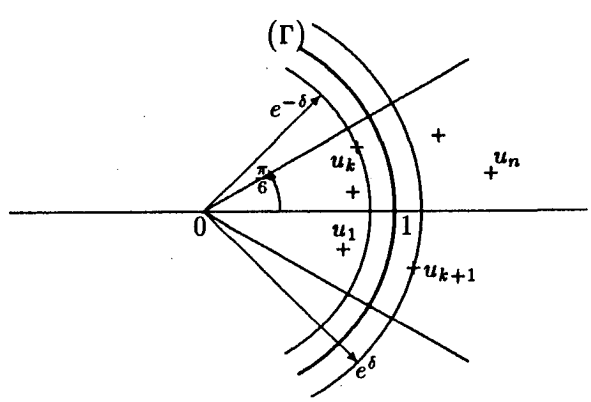


Figure 5. La position des racines après les algorithmes 4 et 5

Problème :

On veut trouver une transformation homographique

$$z \mapsto \frac{z-a}{az-1} \qquad \text{avec } 0 < a < 1$$

telle que le polynôme transformé $Q(X) = P[(X-a)/(aX-1)]$ ait ses racines plus loin du cercle de séparation que P. La démarche à suivre est déjà partiellement donnée dans le résultat suivant :

Proposition 6 *Soit P un polynôme de degré $n > 0$, ayant toutes ses racines dans le secteur*

$$\mathcal{S} = \left\{ \rho e^{i\theta}, \quad \rho > 0 \text{ et } -\frac{\pi}{6} \leq \theta \leq \frac{\pi}{6} \right\}$$

et tel que ses racines $u_1, \ldots, u_n$ peuvent être numérotées telles que :

$$|u_1|, \ldots, |u_k| \leq e^{-\delta} < e^{\delta} \leq |u_{k+1}|, \ldots, |u_n| \qquad \text{avec } \delta > 0.$$

Si Q est le polynôme défini par :

$$Q(X) = (aX-1)^n P\left(\frac{X-a}{aX-1}\right), \qquad \text{avec} \quad 0 < a < 1,$$

alors :

1. *les racines de Q sont les $v_i = (u_i - a)/(au_i - 1)$.*
2. *Pour $1 \leq i \leq k$, les v_i sont à l'intérieur du cercle unité, pour $k+1 \leq i \leq n$ à l'extérieur.*
3. *Si de plus $\delta > 0$ est petit, elles vérifient (avec $a = 1/\sqrt{3} \simeq 0.577$) :*

$$|v_1|, \ldots, |v_k| \leq e^{-2\delta} + O(\delta^2) < e^{2\delta} + O(\delta^2) \leq |v_{k+1}|, \ldots, |v_n|.$$

Preuve

On a

$$Q(v) = 0 \iff P\left(\frac{v-a}{av-1}\right) = 0 \iff \exists i, \frac{v-a}{av-1} = u_i \iff \exists i, \ v = \frac{u_i - a}{au_i - 1},$$

d'où la partie (1) de la proposition. Ceci étant, soit $z = re^{i\theta} \in \mathbb{C}$. Si $T(z) = (z-a)/(az-1)$, on trouve, après calculs :

$$|T(z)|^2 - 1 = \frac{(1-a^2)(r^2-1)}{|az-1|^2} = \frac{(1-a^2)(r^2-1)}{a^2r^2 - 2ar\cos\theta + 1}. \tag{9.1}$$

Cette égalité, compte tenu de (1), prouve la partie (2) de la proposition. Montrons maintenant (3). Soit $v_i = T(u_i)$ une racine de Q. Comme $u_i \in \mathcal{S}$, on peut écrire $u_i = re^{i\theta}$ avec $-\pi/6 \leq \theta \leq \pi/6$ Or $\cos\theta \geq \sqrt{3}/2$ et par conséquent

$$a^2r^2 - 2a\cos\theta r + 1 \leq a^2r^2 - \sqrt{3}ar + 1.$$

Compte tenu de (9.1), on a donc, si $a = 1/\sqrt{3}$:

$$|v_i|^2 - 1 = |T(u_i)|^2 - 1 \geq \frac{2/3}{r^2/3 - r + 1}(r^2 - 1). \tag{9.2}$$

Supposons $k+1 \leq i \leq n$, de sorte que $r = |u_i| \geq e^{\delta}$. Si δ est petit, une petite étude montre que le terme de droite de (9.2) est plus grand que ce même terme où r est remplacé par e^{δ}. Donc, si δ est petit :

$$|v_i|^2 - 1 \geq \frac{2(e^{2\delta}-1)}{e^{2\delta} - 3e^{\delta} + 3} = 4\delta + O(\delta^2),$$

d'où

$$|v_i| \geq \sqrt{1 + 4\delta + O(\delta^2)} = 1 + 2\delta + O(\delta^2) = e^{2\delta} + O(\delta^2).$$

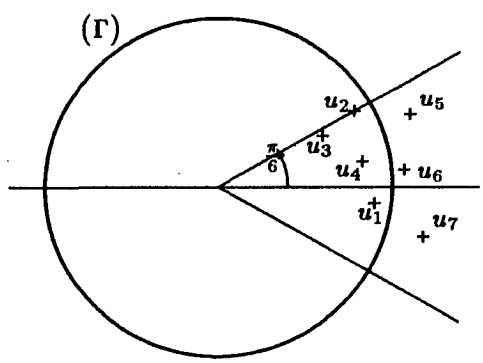
On montrerait de même que si $1 \leq i \leq k$, $|v_i| \leq e^{-2\delta} + O(\delta^2)$.□

Remarques :

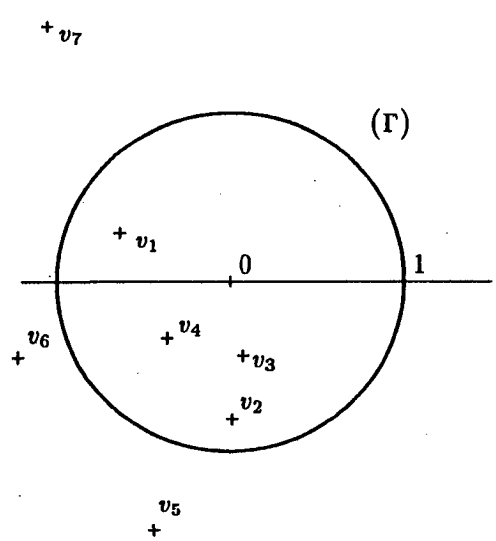
- Dans la pratique, c'est le cas δ petit qui est préocuppant. Cette transformation homographique $z \mapsto (z-a)/(az-1)$ double donc la valeur de δ au moins (elle peut même jusqu'à la quadrupler si les racines proches du cercle Γ sont aussi proches de l'axe réel). Elle présente de plus l'avantage d'étaler les racines en argument (voir la figure 6). Ainsi, dans le paramètre N de l'algorithme 7, non seulement δ est augmenté, mais en plus les paramètres γ et $[P]_2$ sont diminués, μ augmenté (c.f étape 1 de l'algorithme 7).
- Pourquoi avoir choisi $a = 1/\sqrt{3}$? C'est tout simplement parce que c'est le paramètre qui donne les meilleures bornes dans la proposition.
- Si le centre $v = v_i$ déterminé lors de l'algorithme 4 n'est plus $v = -2$, la transformation à utiliser devient :

$$z \mapsto \frac{z-a}{\bar{a}z-1}, \qquad \text{avec } a = -\frac{v}{2\sqrt{3}}.$$

(Attention; si de plus, dans l'algorithme 10, à l'étape 10, on considère le polynôme réciproque P^*, il faut en plus changer a en $\bar{a}$.)



Les racines u_i de P avant la transformation homographique



Les racines du polynôme transformé Q : $v_i = (u_i - a)/(au_i - 1)$ avec $a = 1/\sqrt{3}$.
Figure 6.

Ceci étant, avec les hypothèses décrites précédemment, on veut trouver deux facteurs F et G tels que

$$|P - FG| < \epsilon |P|. \tag{9.3}$$

On veut donc savoir avec quelle précision on doit effectuer la factorisation sur le polynôme transformé $\tilde{P} = Q$ pour avoir (9.3) après transformation inverse. Toutes ces questions trouvent une réponse dans le résultat suivant et son corollaire :

Proposition 7 *Soit K un polynôme de $\mathbb{C}_n[X]$ (avec $n > 0$) et a, $0 < a < 1$. On définit le polynôme $\tilde{K}$ de degré $\leq n$ par :*

$$\tilde{K}(X) = (aX - 1)^n K\left(\frac{X - a}{aX - 1}\right).$$

Alors :

1. *La norme de l'opérateur $\mathcal{H} : \mathbb{C}_n[X] \to \mathbb{C}_n[X]$ $K \mapsto \tilde{K}$ est $|||\mathcal{H}||| = (1 + a)^n$.*
2. *On a l'identité :*

$$(1 - a^2)^n K(X) = \tilde{\tilde{K}} = (aX - 1)^n \tilde{K}\left(\frac{X - a}{aX - 1}\right).$$

Preuve

Si $K = a_0 + a_1 X + \cdots + a_n X^n$, on a

$$\tilde{K}(X) = a_0(aX - 1)^n + a_1(X - a)(aX - 1)^{n-1} + \cdots + a_{n-1}(X - a)^{n-1}(aX - 1) + a_n(X - a)^n,$$

d'où

$$|\tilde{K}| \leq \sum_{j=0}^{n} |X - a|^j |aX - 1|^{n-j} |a_j| = (1 + a)^n |K|,$$

donc $|||\mathcal{H}||| \leq (1 + a)^n$. Le cas d'un monôme montre qu'il y a égalité, d'où la partie 1 de la proposition. Ceci étant, posons $T(X) = (X - a)/(aX - 1)$. Calculons $\tilde{\tilde{K}}$:

$$\tilde{\tilde{K}} = (aX - 1)^n [(aT(X) - 1)^n K[T(T(X))]].$$

Le relations $T(T(X)) = X$ et $(aX - 1)(aT(X) - 1) = (1 - a^2)$ entraînent donc la partie 2 de la proposition.□

Corollaire 2 *Soit P un polynôme de degré $n > 0$ et a, $0 < a < 1$. On considère le polynôme $\tilde{P}$ défini par*

$$\tilde{P}(X) = (aX - 1)^n P\left(\frac{X - a}{aX - 1}\right).$$

Alors, si on trouve deux polynômes $\tilde{F}$ et $\tilde{G}$ de degré k et $n - k$ tels que :

$$|\tilde{P} - \tilde{F}\tilde{G}| < \epsilon,$$

avec

$$F(X) = \left(\frac{aX - 1}{1 - a^2}\right)^k \tilde{F}\left(\frac{X - a}{aX - 1}\right) \quad \text{et} \quad G(X) = \left(\frac{aX - 1}{1 - a^2}\right)^{n-k} \tilde{G}\left(\frac{X - a}{aX - 1}\right)$$

on a

$$|P - FG| < \frac{\epsilon(1 + a)^n}{(1 - a^2)^n}.$$

Preuve

Posons $K = (\tilde{P} - \tilde{F}\tilde{G})/(1-a^2)^n$. Ainsi, on a $\tilde{K} = P - FG$, donc, toujours grâce à la proposition précédente :

$$|P - FG| = |\tilde{K}| \leq (1+a)^n |K| < \frac{(1+a)^n \epsilon}{(1-a^2)^n},$$

d'où le résultat.□

Sous les hypothèses décrites au début de cette section, on est donc en mesure de donner un algorithme qui s'insère entre les étapes 9 et 10 de l'algorithme de la section précédente et qui écarte les racines du cercle de séparation. ϵ étant donné on veut trouver F et G tels que $|P - FG| < \epsilon |P|$.

Algorithme 11 *1. Si $a = 1/\sqrt{3}$, on calcule le polynôme*

$$\tilde{P}(X) = (aX-1)^n P\left(\frac{X-a}{aX-1}\right).$$

Ce calcul peut être mené avec une précision $\epsilon' \sim \epsilon(1-a^2)^n/(1+a)^{2n}$.

2. On connaît k (le nombre de racines de P se trouvant dans le cercle unité). D'après la proposition, le nombre de racines de $\tilde{P}$ se trouvant dans le cercle unité est encore égal à k. On calcule $r_k(\tilde{P})$ et $r_{k+1}(\tilde{P})$ grâce à l'algorithme 2, puis $R = \sqrt{r_k r_{k+1}}$ et $\delta = \log(r_{k+1}/r_k)/2$ (voir la remarque qui suit pour la précision avec laquelle on veut r_k et r_{k+1}). On change alors $\tilde{P}$ en $\tilde{P}(RX)$ (il est clair que $R \sim 1$; ce dernier calcul peut être effectué avec une précision de l'ordre de ϵ_0, où ϵ_0 est donné plus bas).

3. Grâce aux algorithmes décrits dans la section 6, on détermine deux polynômes $\tilde{F}$ et $\tilde{G}$ tels que

$$|\tilde{P} - \tilde{F}\tilde{G}| < \epsilon_0 |\tilde{P}|,$$

avec

$$\epsilon_0 = \frac{\epsilon(1-a^2)^n |P|}{(1+a)^n |\tilde{P}|}.$$

4. On calcule ensuite $\tilde{F}_0 = \tilde{F}(X/R)$ et $\tilde{G}_0 = \tilde{G}(X/R)$, puis

$$F(X) = \frac{1}{(1-a^2)^k}(aX-1)^k \tilde{F}_0\left(\frac{X-a}{aX-1}\right)$$

et

$$G(X) = \frac{1}{(1-a^2)^{n-k}}(aX-1)^{n-k} \tilde{G}_0\left(\frac{X-a}{aX-1}\right).$$

Ces calculs peuvent être menés avec une précision

$$\epsilon_0' \sim \frac{\epsilon |\tilde{P}| (1-a^2)^n}{|\tilde{F}| \cdot |\tilde{G}| \cdot (1+a)^n}.$$

5. Ainsi déterminés, F et G vérifient $|P - FG| < \epsilon |P|$.

Remarques :

- Si le centre du cercle trouvé lors de l'algorithme 4 n'est plus $v = -2$, la transformation homographique utilisée est différente (voir les remarques précédentes).
- Cet algorithme est à mettre en œuvre après l'algorithme 5. L'étape 5 de ce dernier devient alors inutile; elle est remplacée ici par l'étape 2. À ce sujet, lors de cette étape, le calcul de r_k et r_{k+1} devra être effectué avec une incertitude relative de $2\log(M/m)/10$, où M et m sont les variables qui entrent en jeu dans l'algorithme 5. δ sera alors déterminé à 20% près. En toute rigueur, il faut donc ensuite utiliser la minoration 0.8δ de δ au lieu de δ lui même.

Chapitre 10

Factorisation complète

On sait maintenant, $\epsilon > 0$ étant donné, déterminer deux facteurs approximatifs F et G de P, avec $\deg(F) \geq 1$ et $\deg(G) \geq 1$, tels que $|P - FG| < \epsilon|P|$. Rappelons que notre but est de factoriser P en un produit de facteurs du premier degré. Il faut donc effectuer des factorisations successives. Le problème que l'on se pose est le suivant : Étant donné un polynôme complexe P de degré $n \geq 2$ et $\epsilon > 0$, avec quelle précision doit-on effectuer les factorisations successives (en deux facteurs) pour avoir, en fin de compte

$$|P - L_1 L_2 \cdots L_n| < \epsilon|P|$$

où $L_1, \ldots, L_n$ sont des polynômes de degré 1 ? Schönhage résout ce problème en procédant comme suit. On veut qu'à chaque factorisation en k facteurs $P_1, \ldots, P_k$ on ait

$$|P - P_1 \cdots P_k| < \frac{k}{n}\epsilon|P|. \tag{10.1}$$

Supposons donc (10.1) vérifié avec $k < n$, de sorte que l'un des P_i est de degré > 1, par exemple $m = \deg(P_1) > 1$. On va ensuite factoriser P_1 en deux facteurs F et G tels que :

$$|P_1 - FG| < \epsilon_k|P_1|.$$

Ceci entraînera :

$$|P - FGP_2 \cdots P_k| < \frac{k}{n}\epsilon|P| + \epsilon_k|P_1| \cdot |P_2 \cdots P_k|.$$

On veut que ce dernier terme soit inférieur à $(k+1)/n \cdot \epsilon|P|$, ce qui sera vérifié si

$$\epsilon_k < \frac{\epsilon}{n} \frac{|P|}{|P_1| \cdot |P_2 \cdots P_k|}.$$

Comme d'après le théorème 4, on a :

$$|P_1| \cdot |P_2 \cdots P_k| \leq 2^{n/2} \sqrt{\binom{n}{m}}\, [P]_2 < 2^n [P]_2,$$

ceci sera donc vérifié si

$$\epsilon_k = \frac{\epsilon}{n} \cdot \frac{|P|}{2^n [P]_2}.$$

Résumons :

Si chaque factorisation en deux facteurs à partir du cercle de séparation est effectuée avec une précision absolue de

$$\frac{\epsilon}{n} \cdot \frac{|P|}{2^n [P]_2},$$

alors on aura en fin de compte trouvé n facteurs du premiers degré $L_1, \ldots, L_n$ tels que :

$$|P - L_1 \cdots L_n| < \epsilon |P|.$$

Remarques :

- Une autre solution serait, à chaque étape de factorisation, de choisir

$$\epsilon_k = \frac{\epsilon}{n} \frac{|P|}{|P_1| \cdot |P_2 \cdots P_k|}.$$

  Cependant le calcul du produit $P_2 \cdots P_k$ n'est pas nécéssaire dans les autres parties de l'algorithme, et il est un peu coûteux. On peut alors prendre :

$$\epsilon_k = \frac{\epsilon}{n} \frac{|P|}{|P_1| \cdot |P_2| \cdots |P_k|}.$$

  L'avantage de notre choix est qu'il permet d'utiliser une fois pour toute un même ϵ_k sans se préoccuper des $|P_i|$.

- On aurait également pu utiliser la majoration

$$|P_1| \cdot |P_2 \cdots P_k| \leq 2^n M(P).$$

  Le problème est que la mesure de P, $M(P)$, est longue à évaluer.

Chapitre 11

Un algorithme rapide pour les bas degrés

Si la méthode de Schönhage est asymptotiquement rapide, elle n'en n'est pas moins très lente lorsque l'on cherche les racines de polynômes de degré faible (par rapport à des méthodes plus classiques comme la méthode de Newton, ou plus évoluée comme la méthode de Traub et Jenkins). Il est donc intéressant de disposer d'un algorithme rapide qui trouve les racines de polynômes de bas degré, lorsque le polynôme n'est pas trop mal conditionné; qu'importe si celui-ci échoue pour certains polynômes pathologiques, puisqu'il suffit alors d'embrayer sur l'algorithme de Schönhage. En voici un qui repose essentiellement sur la méthode de Newton :

Lorsque le degré du polynôme est égal à 2, il suffit pour trouver ses racines de calculer les formules correspondantes.

Sinon nous allons procéder à l'aide de la méthode de Newton. Soit $P = a_0 + a_1 X + \cdots + a_n X^n$ un polynôme de degré $n > 2$.

Lorsque le polynôme P est un polynôme obtenu après des factorisations successives grâce au cercle de séparation, ses racines seront souvent proches les unes des autres. Il est donc intéressant de les regrouper autour de l'origine, en ramenant le barycentre des racines à 0 (considérer $P(X - \frac{a_{n-1}}{n\, a_n})$).

Le problème majeur est de déterminer une valeur initiale z_0 pour commencer les itérations de Newton.

- Une première méthode est de calculer grâce à l'algorithme 3 le plus grand des modules R des racines de P (à 0.05 prés par exemple), puis de changer P en $P(RX)$, se ramenant ainsi au cas où la plus grande des racines est sur le cercle unité. On calcule ensuite la valeur de P en plusieurs points du cercle unité (par exemple on calcule les $P(\exp(ik\pi/n))$, $0 \leq k \leq 2n-1$) puis on choisit z_0 minimisant ces valeurs.

- Cependant, si cette première manière de faire donne de bons résultats dans la pratique, le calcul de R est relativement long. Dans des cas simples (par exemple pour un degré ≤ 15), on peut procéder autrement pour trouver z_0 : la valeur de z_0 sera moins bonne que la précédente mais sera trouvée plus rapidement. L'idée, classique, est d'utiliser la majoration du plus grand des modules de P donnée par l'unique racine positive de l'équation :

$$|a_n|R^n = |a_{n-1}|R^{n-1} + \cdots + |a_1|R + |a_0|.$$

  (on peut montrer que $(2^{1/n}-1)R \leq r_n(P) \leq R$, voir [8]). On considère ensuite un point pris au hasard sur le cercle de rayon R, par exemple $z_0 = Re^i$.

A partir de z_0, on effectue ensuite des itérations de Newton :

$$z_{n+1} = z_n - \frac{P(z_n)}{P'(z_n)}.$$

Si z_0 est proche d'une racine, on sait que (z_n) converge très vite (quadratiquement). Cependant, surtout lors des premiers pas d'itération, la suite (z_n) peut se mettre à diverger. On contrôle alors le processus comme suit :

- Si $P'(z_n) = 0$ (et $P(z_n) \neq 0$), on perturbe un tout petit peu z_n.
- Si $|P(z_{n+1})|$ est supérieur à $|P(z_n)|/\alpha$, où α est une constante > 1 (par exemple $\alpha = 1.1$) on considère un nouveau point z_{n+1} défini par [1] :

$$z_{n+1} = \frac{z_n + z_{n+1}}{2},$$

puis on recommence jusqu'à ce que $|P(z_{n+1})| < |P(z_n)|/\alpha$ (Un cas typique de la situation est représenté par la figure 7).

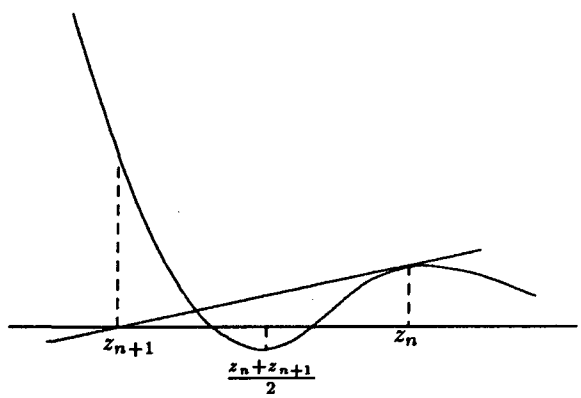


Figure 7. Un cas où $|P(z_{n+1})| > |P(z_n)|/\alpha$.

Si au bout d'un nombre déterminé d'opérations de ce type (par exemple 5 itérations), on n'a toujours pas la minoration escomptée, on recherche z_{n+1} en résolvant l'équation $P(z_{n+1}) = 0$ non plus à l'ordre 1 (c'est la méthode de Newton), mais à l'ordre 2; on recherche si l'une des deux racines z de

$$P(z_n) + (z - z_n)P'(z_n) + \frac{(z - z_n)^2}{2}P''(z_n) = 0$$

vérifie $|P(z_{n+1})| < |P(z_n)|/\alpha$. Si c'est le cas, nous avons notre candidat pour z_{n+1}, sinon la méthode de Newton est considérée comme divergente et on lance la méthode de Schönhage.

On s'arrête lorsque l'on a effectué trop d'itérations (par exemple 40) ou lorsque la valeur du polynôme initial P en z_n est inférieure en module à $\epsilon|P|$ (où ϵ est tel que l'on veut $|P - FG| < \epsilon|P|$). Si $F = X - z_n$ et G est la division euclidienne de P par F, on a alors facilement $P - FG = P(z_n)$, donc

$$|P - FG| < \epsilon|P|.$$

Remarques.

- Cette méthode donne dans la pratique de bons résultats lorsque le polynôme est de bas degré et pas trop mal conditionné.

[1] Cette méthode est celle utilisée par Cohen dans PARI.

- Lorsque le polynôme est à coefficients réels, il est intéressant, à la fin de la méthode de Newton, de tester la valeur de la partie imaginaire de z_n. Si celle-ci est faible, on teste alors si sa partie réelle est une racine satisfaisante de P (avec le test déjà utilisé plus haut). Si c'est le cas, on remplace alors la racine trouvée par sa partie réelle. Sinon, on teste si son conjugué $\bar{z_n}$ est tel que la division euclidienne G de P par $F = X^2 - (z_n + \bar{z_n})X + z_n\bar{z_n}$ vérifie le désormais classique $|P - FG| < \epsilon|P|$. Si c'est le cas, tant mieux (F et G sont des facteurs satisfaisants), sinon on itère encore une fois par Newton. Si les tests précédents échouent encore (c'est rarissime !) on lance Schönhage.

  L'avantage de chercher des facteurs réels à un polynôme P réel est double. L'utilisateur sera certainement satisfait de savoir qu'une racine est purement réelle ou de voir que ses racines sont conjuguées. De plus, on gagne en facteur temps puisque les polynômes déflatés étant eux aussi réels, les calculs sur ces polynômes seront plus rapides que s'ils étaient à coefficients complexes.

Chapitre 12

Étude de la précision obtenue sur les racines

Certes, un polynôme complexe P de degré $n > 0$ étant donné, l'algorithme sait, $\forall \epsilon > 0$, trouver n complexes $v_1, \ldots, v_n$ tels que :

$$|P - \beta(X - v_1) \cdots (X - v_n)| < \epsilon |P|.$$

Un utilisateur sera certainement intéressé de savoir à quel point les v_i approximent les racines de P. Un premier résultat à ce sujet est dû à Ostrowski :

Théorème 16 *Soit un polynôme P unitaire de degré $n > 0$ et n complexes $v_1, \ldots, v_n$ tels que*

$$|P - (X - v_1) \cdots (X - v_n)| < \epsilon.$$

Alors on peut écrire

$$P = (X - u_1) \cdots (X - u_n)$$

avec

$$\forall i, \qquad (1 - 4\rho \sqrt[n]{\epsilon})|u_i - v_i| < 4\rho \sqrt[n]{\epsilon}, \quad \text{où } \rho = \sup\{1, |v_1|, \cdots, |v_n|\}.$$

Preuve :

L'argument utilisé ressemble à celui de la démonstration du théorème 9. Soit v une racine de $\hat{P} = (X - v_1) \cdots (X - v_n)$. Considérons la famille continue de polynômes $H_t = tP + (1-t)\hat{P}$, $t \in [0,1]$. v est racine de $H_0 = \hat{P}$; un argument de "continuité des racines" des H_t en fonction de $t \in [0,1]$ montre qu'il existe un chemin continu $t \mapsto z_t$ tel que $z_0 = 0$ et $\forall t$, $v + z_t$ est racine de H_t (en particulier, $u = v + z_1$ est racine de $H_1 = P$). Donc

$$\begin{aligned} \forall t \in [0,1], \qquad |\hat{P}(v + z_t)| &= |H_t(v + z_t) + t(\hat{P} - P)(v + z_t)| = t|(\hat{P} - P)(v + z_t)| \\ &\leq (\sup\{1, |v + z_t|\})^n |\hat{P} - P| < (\sup\{1, |v| + |z_t|\})^n \epsilon. \end{aligned} \tag{12.1}$$

Or, on a aussi :

$$\forall t \in [0,1], \qquad |\hat{P}(v + z_t)| = \prod_i |v_i - v - z_t| \geq \prod_i ||z_t| - |v - v_i||.$$

En combinant ce dernier résultat avec (12.1), on obtient :

$$\forall t \in [0,1], \qquad \prod_i ||z_t| - |v - v_i|| < (\sup\{1, |v| + |z_t|\})^n \epsilon.$$

Lorsque t parcourt $[0,1]$, la continuité de $t \mapsto z_t$ entraîne que $|z_t|$ prend (au moins) toutes les valeurs entre 0 et $|z_1| = |u - v|$, donc

$$\forall r \in [0, |u - v|], \qquad \prod_i |r - |v - v_i|| < (\sup\{1, |v| + r\})^n \epsilon \leq (\rho + r)^n \epsilon.$$

D'après une propriété bien connue des polynômes de Tchébichev, il existe un r dans $[0, |u-v|]$ tel que le terme de gauche soit plus grand que $|u-v|^n/2^{2n-1}$. On en déduit donc :

$$\frac{|u-v|^n}{2^{2n-1}} < (\rho + |u-v|)^n \epsilon$$

d'où le résultat puisque u est une racine de P.□

Remarques.

- L'erreur sur les racines est donc de l'ordre de $\sqrt[n]{\epsilon}$. Le cas évident des polynômes $P = X^n$ et $\hat{P} = X^n - \epsilon$ montre qu'il en est bien ainsi. (D'autres cas de polynômes mal conditionnés seront traités plus tard. Les polynômes de Tchébichev dont la propriété mini-max nous a servi sont également des cas pathologiques intéressants).
- Cependant, dans le cas général, la borne donnée par le théorème est très pessimiste. On peut en fait donner *a posteriori* une majoration d'erreur sur les racines beaucoup plus fine que celle du théorème. Ce type de problème est traité par exemple dans [18]. Schönhage fait une analyse beaucoup plus fine de la situation grâce à l'introduction des polynômes de Tchébichev. Nous suivrons son idée. Grossièrement, si v_i est une racine d'ordre m (au sens où m racines sont proches de v_i et les autres éloignées), on a une majoration d'erreur de l'ordre de $\sqrt[m]{\epsilon}$. Plus précisément, voyons comment donner un algorithme permettant de donner une majoration plus fine que celle du théorème.

Nous sommes toujours sous les hypothèses du théorème. Reprenons la démonstration du théorème. On avait montré (v désigne l'une des racines v_i trouvée par l'algorithme, u une racine du polynôme de départ P) :

$$\forall r \in [0, |u-v|], \qquad \prod_i |r - |v - v_i|| < (\sup\{1, |v| + r\})\, r^n \epsilon$$

En fait, $\exists i$ tel que $v = v_i$, par exemple $v = v_1$, et ceci s'écrit :

$$\forall r \in [0, |u-v|], \qquad r \prod_{i=2}^{n} ||v - v_i| - r| < (\sup\{1, |v| + r\})^n . \tag{12.2}$$

Supposons que les v_i soient numérotés de telle sorte que les $d_i = |v - v_i|$ soient rangés dans l'ordre croissant :

$$d_2 \leq d_3 \leq \cdots \leq d_n.$$

Une première borne sur $|u-v|$ est donnée par :

$$|u-v| \leq \epsilon_1 = \frac{4 \sup\{1, |v|\} \sqrt[n]{\epsilon}}{1 - 4\sqrt[n]{\epsilon}}.$$

Soit m_1, $2 \leq m_1 \leq n$ choisi tel que

$$d_{m_1} \leq \frac{5}{4}\epsilon_1 < d_{m_1+1}.$$

Alors d'après (12.2) :

$$\forall r \in [0, |u-v|], \qquad r \prod_{i=2}^{m_1} |d_i - r| \prod_{i=m_1+1}^{n} (d_i - \epsilon_1) < (\sup\{1, |v| + \epsilon_1\})^n \epsilon.$$

Comme précédemment, le terme $\prod_{i=2}^{m_1} |d_i - r|$ est plus grand que $|u-v|^{m_1-1}/2^{2m_1-3}$; on obtient finalement :

$$|u-v| < \epsilon_2 = \frac{2^{2-3/m_1} \rho_1^{n/m_1}}{\left(\prod_{i=m_1+1}^{n} (d_i - \epsilon_1)\right)^{1/m_1}} \sqrt[m_1]{\epsilon}, \qquad \text{où } \rho_1 = \sup\{1, |v| + \epsilon_1\}.$$

On peut bien sûr recommencer l'opération en partant cette fois de la majoration $|u-v| \leq \epsilon_2$. Si

$$d_{m_2} \leq \frac{5}{4}\epsilon_2 < d_{m_2+1},$$

on a :

$$|u-v| < \epsilon_3 = \frac{2^{2-3/m_2}\rho_2^{n/m_2}}{\left(\prod_{i=m_2+1}^{n}(d_i-\epsilon_2)\right)^{1/m_2}}\sqrt[m_2]{\epsilon}, \qquad \text{où } \rho_2 = \sup\{1, |v|+\epsilon_2\},$$

et on peut ainsi réitérer l'opération pour obtenir des majorations $\epsilon_3, \epsilon_4, \cdots$ de $|u-v|$ de plus en plus fines. (Dans la pratique, on se limitera a un nombre fixé d'itérations déterminé à l'avance. Une autre possibilité est de s'arrêter lorsque le rapport $\epsilon_k/\epsilon_{k+1}$ devient plus faible qu'une valeur fixée, par exemple 2).

Remarque :

- Un moyen de faire, également, serait de calculer les valeurs de P aux points v_i. Le problème est que si le polynôme P n'est pas déterminé exactement (par exemple si P n'est pas à coefficients dans $\mathbb{Z}$), cette méthode est trompeuse ! En procédant ainsi, on évalue les erreurs par rapport aux racines du polynôme entré $\hat{P}$. Le polynôme $\hat{P}$ étant un arrondi du véritable polynôme P, les résultats donnés ne sont pas ceux que l'on veut ! La méthode que nous avons décrite est donc, de ce point de vue, bien meilleure. Elle donne de plus dans la pratique des résultats tout à fait satisfaisants.

Chapitre 13

Une série de tests

Pour évaluer les performances de la méthode de Schönhage par rapport à des méthodes déjà existantes, nous présentons une série de tests sur plusieurs familles de polynômes. La méthode décrite dans les sections précédente a été programmée en MAPLE V. Le programme correspondant est comparé à trois autres programmes :

1. La fonction *fsolve* de MAPLE V. (Implantation boguée de la méthode de Traub et Jenkins, voir [9], pour les polynômes à coefficients complexes).
2. La fonction *fsolve/traubjen*. (Implantation non boguée, par Bruno Salvy, de la méthode de Traub et Jenkins, en MAPLE V).
3. La fonction *NRoots* de MATHEMATICA. (Utilisant également l'algorithme de Traub et Jenkins).

La comparaison est effectuée sur trois paramètres :

- La précision obtenue sur les racines
- Le temps d'exécution (donné en temps équivalent sur une même machine).
- La taille mémoire utilisée (pour les polynômes de haut degré)

La fonction *NRoots* utilise du C compilé alors que les autres programmes tournent en interprété en MAPLE V. Ceci explique (en partie peut-être) le gain de temps considérable obtenu avec *NRoots*.

13.1 Des polynômes mal conditionnés

Dans [10],Traub et Jenkins donnent toute une série de polynômes mal conditionnés. Nous effectuerons dans cette sous partie des tests analogues aux leurs [1].

13.1.1 Les polynômes à racines proches ou multiples

La section 11 nous a montré que les racines proches ou multiples étaient très sensibles aux perturbations. Voyons ce que cela donne dans la pratique.

$$P_1 = \left(X - \frac{\sqrt{2}}{10}\right)^3 \left(X - \frac{4}{10}\right)\left(X - \frac{5}{10}\right)\left(X - \frac{\pi}{5}\right)$$

Précision demandée : 10 chiffres

[1] Les polynômes tests donnés par Traub et Jenkins présentent la particularité d'avoir des racines décimales. Il m'a paru plus représentatif de prendre des racines non rationnelles et parfois transcendantes, mais déterminées exactement (comme π, e ...)

Méthode	Δ_{racines}	Temps d'exécution
Schönhage	10^{-5}	4.5 s
fsolve	10^{-10}	1.5 s
fsolve/traubjen	0.2×10^{-3}	1.2 s
NRoots	10^{-6}	0.1 s

Chacun des quatre programmes détermine donc les racines correctement. Remarquons ici le temps important mis par la méthode de Schönhage par rapport aux autres. C'est normal puisque sur les bas degrés (même si la méthode de Newton est utilisée), la méthode de Schönhage n'est pas concurrentielle.

Autre remarque : À part la fonction *fsolve* de MAPLE V, les programmes renvoient dans la liste deux racines à partie imaginaire non nulle (celles proches de $\sqrt{2}/10$). Les programmes sont donc dans ce cas ci incapables de distinguer si une racine est réelle ou non.

Enfin, la fonction *NRoots* renvoie les racines avec 6 chiffres significatifs et c'est pour cette raison que l'on ne peut pas s'attendre à avoir sur les racines données une précision supérieure à 6 chiffres.

$$P_2 = \left(X - \frac{e}{10}\right)\left(X - \frac{2e}{10}\right)^2\left(X - \frac{3e}{10}\right)^3\left(X - \frac{4e}{10}\right)^4$$

Précision demandée : 10 chiffres

Méthode	Δ_{racines}	Temps d'exécution
Schönhage	0.5×10^{-3}	18. s
fsolve	non trouvées (message d'erreur au bout de 3.3 s)	
fsolve/traubjen	0.2×10^{-1}	5.6 s
NRoots	10^{-3}	0.07 s

Lorsque l'on augmente la précision du calcul, une curieuse réaction de NRoots sur MATHEMATICA apparaît : Avec 100 chiffres de précision, *NRoots* boucle indéfiniment et est donc incapable de déterminer les racines avec une trop grande précision.

Enfin *fsolve* est incapable de déterminer les racines même lorsque l'on augmente indéfiniment la valeur de Digits (Digits=500 par exemple).

Un phénomène amusant apparaît lorsque l'on change l'ordre des puissances du polynôme P_2 :

$$P_3 = (X - \frac{e}{10})^4(X - \frac{2e}{10})^3(X - \frac{3e}{10})^2(X - \frac{4e}{10})$$

Précision demandée : 10 chiffres

Méthode	Δ_{racines}	Temps d'exécution
Schönhage	10^{-3}	16. s
fsolve	0.4×10^{-4}	7.5 s.
fsolve/traubjen	0.5×10^{-2}	11. s
NRoots	10^{-5}	0.07 s

Cette fois ci, la fonction *fsolve* de MAPLE V renvoie les racines. Remarquons également qu'à part la méthode de Schönhage, la précision sur les racines est meilleure. Ceci s'explique par le fait que dans la méthode de Traub et Jenkins, on commence par trouver les racines de plus petit module. Ainsi, la perte de précision par déflation a peu de conséquences puisque les racines de grand module sont d'ordre de multiplicité inférieur a celles de petit module.

$$P_4 = (X - 1)(X - 1.001)(X - 0.998)(X - 1.00002)(X - 0.99999)$$

La forme développée de P_3 s'écrit avec 17 décimales. Nous effectuerons donc le calcul avec 20 décimales.
Précision demandée : 20 chiffres

Méthode	Δ_{racines}	Temps d'exécution
Schönhage	0.5×10^{-3}	18. s
fsolve	Non trouvées (message d'erreur au bout de 0.5 s)	
fsolve/traubjen	10^{-10}	1.450 s
NRoots	2×10^{-5}	0.5 s.

Une remarque intéressante qui n'apparaît pas dans le tableau est que *NRoots* trouve la liste

```
{ X == 0.998 || X == 1. || X == 1. || X == 1. || X == 1.001}.
```

Autrement dit, 1 est vu comme une racine triple ! Le polynôme donné par Traub et Jenkins tendait ce piège. Certains programmes, ayant des problèmes avec les racines proches, considèrent que celles-ci sont multiples ce qui facilite leur tâche. NRoots fait partie de ceux-ci. Même en augmentant indéfiniment la précision (jusqu'à 500 chiffres), la réponse donnée par MATHEMATICA est la même. Par contre, MATHEMATICA trouve les racines correctement (pour une grande précision) lorsqu'on lui entre le polynôme sous forme fractionnaire. Enfin, la fonction *fsolve* de MAPLE V échoue quelle que soit la précision entrée au départ.

13.1.2 Les polynômes à zéros d'ordre de grandeur différents

Lorsque les zéros d'un polynôme sont d'un ordre de grandeur différents, l'imprécision sur l'une des racines (par exemple sur la plus grande) entraîne par déflation des imprécisions beaucoup plus grandes sur les petites racines.

$$P_5 = \left(X - \frac{e}{100}\right)\left(X - \frac{e}{10}\right)(X - e)\left(X - e + i\frac{\pi}{10^6}\right)\left(X - e - i\frac{\pi}{10^6}\right)(X - 10e)(X - 100e)$$

Précision demandée : 30 chiffres (L'imprécision sur la 'racine presque triple' e étant de l'ordre de $10^{-s/3}$ si s est la précision demandée, on prend $s = 30$ pour avoir 10 chiffres de précision sur cette racine)

Méthode	Δ_{racines}	Temps d'exécution
Schönhage	0.5×10^{-19}	14. s
fsolve	10^{-25}	6 s
fsolve/traubjen	10^{-17}	2.5 s
NRoots	0.5×10^{-5}	0.45 s

Un fait étonnant est à signaler : La fonction *fsolve* de MAPLE V ne parvient plus à déterminer les racines lorsque Digits$\geq$ 33. Par ailleurs, la fonction *NRoots* ne trouve ici que des racines purement réelles (d'où l'imprécision 0.5×10^{-5}). Elle ne parvient à déterminer la partie imaginaire des racines que lorsque la précision demandée est supérieure à 44.

Jusqu'à maintenant, la fonction *NRoots* semble (la méthode de Schönhage mise à part) donner des résultats pas trop inquiétants en un temps très court. En perturbant (beaucoup !) le polynôme P_5, la fonction *NRoots* s'en sort bien plus mal :

$$P_6 = P_5 - 100$$

Précision demandée : 30 chiffres

Méthode	Δ_{racines}	Temps d'exécution
Schönhage	0.5×10^{-30}	6. s
fsolve	10^{-30}	3. s
fsolve/traubjen	10^{-25}	3. s
NRoots	0.12	1. s

Le polynôme possède 2 racines non réelles qui sont, à 10^{-5} prés :

```
2.6546 + .1181 I,   2.6546 - .1181 I.
```

Au lieu de celles ci, *NRoots* renvoie :

```
X == 2.6546944316807 || X == 2.6546944316807 ||
```

Autrement dit, *NRoots* renvoie la partie réelle de ces racines! L'utilisateur est alors trompé sur deux plans : il croit n'avoir que des racines réelles et une racine double. Même en lui demandant d'effectuer la calcul avec 500 chiffres significatifs, *NRoots* renvoie toujours la même chose.

$$P_7 = \left[(X+e)^5 - 10^{10}\right]\left[(X+\pi)^5 - 10^{-10}\right]$$

Ce polynôme a de grandes racines autour de $-e$ et de petites racines autour de $-\pi$. De plus, la régularité de la position de ses racines est un argument à son comportement mal conditionné.

Précision demandée : 16 chiffres

Méthode	Δ_{racines}	Temps d'exécution
Schönhage	10^{-11}	23. s
fsolve	Non trouvées (message d'erreur au bout de 5. s)	
fsolve/traubjen	10^{-5}	5. s
NRoots	Erreur machine	

NRoots sur MATHEMATICA plante la session (sur IBM RS 6000 et DecStation). Au moins devrait-elle renvoyer un message d'erreur... Un fait étonnant est que lorsque les racines sont demandées avec 20 chiffres de précision, *NRoots* renvoie une réponse correcte.

Autre fait bizarre, la fonction *fsolve* de MAPLE V parvient à déterminer les racines lorsque Digits$\neq$ 16.

13.2 Des familles de polynômes classiques

Les polynômes envisagés précédemment sont très particuliers et ont été choisis pour leur mauvais comportement vis-à-vis de la recherche des racines. Un utilisateur aura en général peu à rechercher les racines de ce type de polynômes. Il est donc intéressant de comparer l'efficacité des différents programmes vis à vis de polynômes plus "banals", par exemple sur des familles classiques de polynômes (polynômes de Bernoulli, Laguerre ...).

13.2.1 Une famille de polynômes orthogonaux : les polynômes de Laguerre

Les polynômes orthogonaux (vis à vis d'une mesure sur $\mathbb{R}$) ont leur racines toutes réelles et simples. Il est donc clair que pour ce type de polynômes, on pourrait mettre en œuvre un programme beaucoup plus performant de recherche des racines. Il est quand même intéressant de voir ce que donne la méthode de Schönhage (et celle de Traub et Jenkins) sur de tels polynômes.

L'exemple choisi de polynômes orthogonaux est celui des polynômes de Laguerre (L_n). Ces polynômes sont *a priori* peu sympathiques puisqu'ils possèdent de grandes et de petites racines. (Par exemple les racines de L_{30} s'étalent de $0.047\ldots$ à $104.157\ldots$). Pour mémoire, rappelons que les polynômes de Laguerre sont orthogonaux au sens suivant :

$$\forall n, m > 0, \qquad \int_0^{+\infty} L_n(t) L_m(t) e^{-t} dt = \delta_n^m.$$

Les racines de L_n sont donc toutes sur $\mathbb{R}^+$. On obtient une forme explicite des L_n sous la forme

$$L_n(X) = \sum_{k=0}^{n} \frac{(-1)^k}{k!} \binom{n}{m} X^k.$$

On voit que le coefficient dominant de L_n est tout petit. Pour obtenir des résultats raisonnables, nous calculerons les racines de L_n avec une précision (en nombre de chiffres) au moins égale à n.

Polynôme	Précision	Schönhage		fsolve		fsolve/traubjen		NRoots	
		Δ_r	tps	Δ_r	tps	Δ_r	tps	Δ_r	tps
L_5	10	10^{-10}	0.8 "	$2 \cdot 10^{-9}$	0.5 "	$2 \cdot 10^{-8}$	0.7"	10^{-5}	0.03"
L_{10}	10	10^{-10}	13"	10^{-8}	2"	10^{-5}	3"	10^{-4}	0.07"
L_{20}	20	10^{-19}	2'	10^{-18}	10"	10^{-10}	16"	10^{-10}	1.7"
L_{30}	30	10^{-30}	11'30"	10^{-28}	45"	10^{-12}	41"	10^{-15}	4"
L_{40}	40	10^{-40}	23'20"	10^{-39}	6'50"	10^{-22}	1'40"	10^{-20}	12"

Figure 8. Tableau de tests effectués sur les polynômes de Laguerre.
La colonne précision indique le nombre de chiffres demandés pour le calcul.
Δ_r représente l'erreur maximale sur les racines données par l'algorithme.
Enfin, les colonnes tps indiquent le temps de calcul.

Remarques.

- Tous les programmes renvoient des racines purement réelles (sur les polynômes testés).
- En fait, les trois programmes mis en concurrence avec celui implantant la méthode de Schonhage possèdent une procédure de recherche de racines réelles; le test est donc fortement faussé car on ne compare pas des programmes de recherche de racines complexes.
- Ici, l'algorithme de Schönhage est trés lent. Les racines des polynômes de Laguerre sont éparses et il y a toujours peu de racines (souvent une ou deux) dans le cercle de séparation (*cf.* section 3). Il faut donc effectuer beaucoup d'itérations avant d'obtenir la factorisation complète du polynôme. Cette faiblesse de l'algorithme de Schönhage est mentionnée par Schönhage lui même dans [21] (p. 142).

13.2.2 Polynômes de Bernoulli

Les polynômes $B_n(X)$ de Bernoulli ont leur fonction génératrice égale à :

$$\frac{te^{xt}}{e^t - 1} = \sum_{n=0}^{\infty} B_n(x)\frac{t^n}{n!}.$$

La formule $\sum_{k=0}^{n-1} \binom{n}{m} B_k(X) = nX^{n-1}$ permet par récurrence de calculer les $B_n(X)$. La famille des polynômes de Bernoulli n'étant pas orthogonale, les racines de $B_n(X)$ ne sont *a priori* pas toutes situées sur l'axe réel. Des calculs de racines montrent qu'il en est bien ainsi.

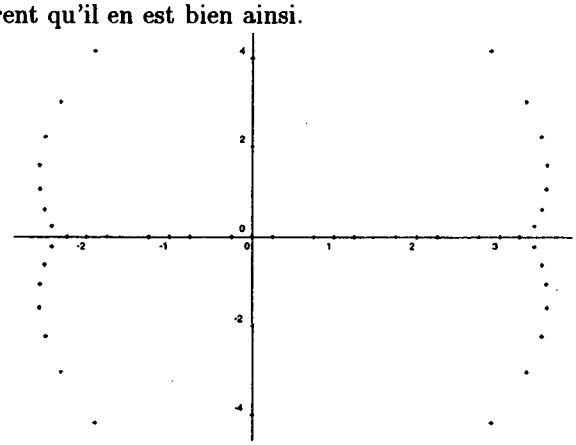


Figure 9. Les racines du polynôme $B_{40}(X)$
Les racines des autres polynômes de Bernoulli ont une localisation du même type.

La relation bien connue $B_n(1-X) = (-1)^n B_n(X)$ montre que les racines de B_n sont symétriques par rapport à l'axe vertical $x = \frac{1}{2}$, résultat concordant avec la figure 9. (Une étude approfondie de la localisation des racines des polynômes de Bernoulli est possible et peut se trouver dans [5].)

Voyons maintenant comment se comportent nos programmes de résolution des racines face à ces polynômes :

Polynôme	Précision	Schönhage		fsolve		fsolve/traubjen		NRoots	
		Δ_r	tps	Δ_r	tps	Δ_r	tps	Δ_r	tps
B_5	10	10^{-10}	0.9 "	10^{-9}	0.35 "	$2 \cdot 10^{-8}$	0.8"	10^{-5}	0.01"
B_{10}	10	10^{-10}	7.1"	10^{-10}	1.4"	10^{-9}	4"	10^{-5}	0.07"
B_{20}	20	10^{-20}	1'	10^{-19}	20"	10^{-19}	30"	10^{-12}	1.7"
B_{30}	30	10^{-30}	5'10"	10^{-29}	1'20"	1	20'	10^{-24}	4.5"
B_{40}	40	10^{-40}	10'40"	10^{-39}	3'15"	/	>45'	10^{-32}	9"
B_{50}	50	10^{-50}	43'	10^{-49}	6'30"	/	/	10^{-39}	17"

Figure 10. Tableau de tests effectués sur les polynômes de Bernoulli

Remarques.

- Il apparaît clairement ici que la fonction *fsolve/traubjen* est mise en échec. Elle met plus de vingt minutes à renvoyer une liste de racines pour le polynôme de Bernoulli de degré 30, et de plus les racines renvoyées sont totalement fausses (imprécision de 1). Pour le degré 40, les racines ne sont toujours pas trouvées au bout de 45 minutes et de toutes façons, même avec suffisamment de temps on s'attend à ce que le résultat soit faux. Le test n'a pas été effectué pour le degré 50.
- Comme dans le test précédent sur les polynômes de Laguerre, la curieuse répartition des racines des $B_n(X)$ fait que l'algorithme de Schönhage effectue à chaque itération une factorisation $P = FG$ déséquilibrée (en d'autres termes, il ne se trouve que peu de racines dans le cercle de séparation). Il lui faut donc effectuer beaucoup de factorisations avant d'obtenir les racines, d'où le temps important mis par l'algorithme.

13.3 Tests sur les polynômes de Curtz

Ces polynômes ont été introduit par Paul Curtz dans [4] et sont définis par la récurrence

$$P_0(x) = 1, \qquad P_n(x) = x \sum_{q=0}^{n-1} \frac{(-1)^q}{q+1} P_{n-q-1}(x) + \frac{(-1)^n}{n+1}.$$

Ces relations donnent une formule explicite pour la série génératrice des $P_n(x)$

$$P(z,x) := \sum_{n=0}^{\infty} P_n(x) z^n = \frac{\log(1+z)}{z(1-x\log(1+z))}.$$

On peut, grâce à cette formule déduire de nombreuses propriétés sur les polynômes P_n [15]. Notamment, une étude asymptotique des coefficients de z^n de la série génératrice des $P_n(x)$ est rendu possible par sa forme explicite, et permet de montrer que, pour n grand, les racines de P_n s'accumulent régulièrement autour de la courbe limite

$$z = \frac{1}{\log(1+e^{i\theta})} \quad \text{avec} \quad \theta \in]-\pi, +\pi[.$$

La figure 11 confirme le bien fondé de cette assertion. Notons également que les P_n n'ont pas de racine réelle si n est pair et une unique racine réelle si n est impair, résultat démontré en appendice et qui permet de confirmer (partiellement bien sûr) la validité des valeurs numériques obtenues.

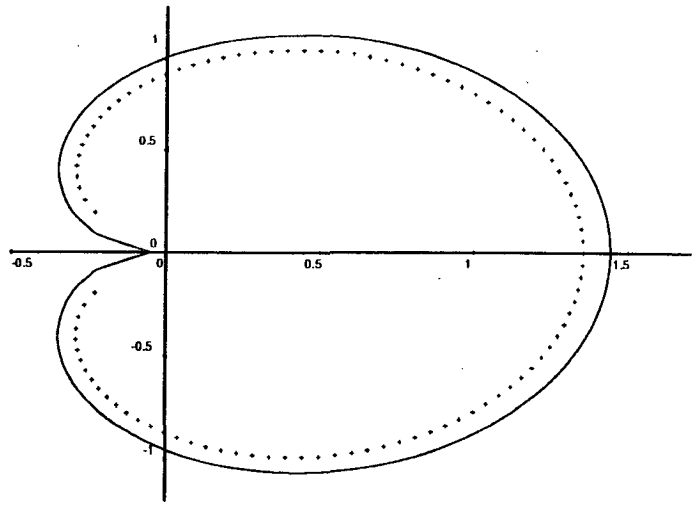


Figure 11. Les racines de $P_{101}(x)$ et la courbe limite des racines.

Ces polynômes fournissent donc une famille de tests intéressante puisque l'on a déjà une idée de la forme du résultat. De plus, ces polynômes possèdent une curieuse propriété de platitude (à l'intérieur de la courbe limite des racines) et à ce titre, ils possèdent un mauvais conditionnement. De fait, nous allons voir que les programmes déjà existants (*fsolve*, fsolve/traubjen et *NRoots*) échouent lorsque le degré du polynôme de Curtz devient trop élevé.

Polynôme	Précision	Schönhage		fsolve		fsolve/traubjen		NRoots	
		Δ_r	tps	Δ_r	tps	Δ_r	tps	Δ_r	tps
P_5	10	10^{-10}	1.5 "	10^{-10}	0.9 "	$2 \cdot 10^{-8}$	1"	$2 \cdot 10^{-6}$	0.01"
P_{10}	10	10^{-10}	5"	10^{-10}	3.5"	10^{-7}	5.5"	10^{-5}	0.02"
P_{20}	20	10^{-20}	1'30"	10^{-20}	30"	10^{-11}	47"	10^{-13}	2"
P_{30}	30	10^{-30}	6'	10^{-30}	2'20"	/	/	10^{-24}	4.5"
P_{40}	40	10^{-40}	20'	/	/	/	/	10^{-32}	10"
P_{50}	50	10^{-50}	40'	/	/	/	/	/	> 1h.
P_{60}	60	10^{-60}	1h08'	/	/	/	/	/	/
P_{70}	70	10^{-70}	3h	/	/	/	/	/	/
P_{80}	80	10^{-80}	9h	/	/	/	/	/	/
P_{101}	40	10^{-36}	24h	/	/	/	/	/	/

Figure 12. Tableau de tests effectués sur les polynômes de Curtz

Remarques.

- Lors de la recherche des racines de P_{30}, *fsolve/traubjen* provoque au bout d'une dizaine de minutes un arrêt de la session pour manque de mémoire (test effectué avec 56 Mo). Ce phénomène se produit déjà lors de la recherche des racines de P_{25}.
- La fonction *fsolve* a quant à elle un curieux comportement. Elle parvient à déterminer les racines de P_{30} et de P_{32}, mais pas celles de P_{29} ou de P_{31} (et ce quelle que soit la précision demandée). Les polynômes de Curtz de degré impair seraient donc plus mal conditionnés que ceux de degré pair (?) Toujours est-il que pour $n \geq 36$, *fsolve* est incapable de donner des racines de P_n.
- Au bout d'une heure de calcul, la fonction *NRoots* ne parvient pas à déterminer les racines de P_{50} lorsqu'on lui demande 50 chiffres de précision (d'autres tests montrent qu'elle n'y parvient pas non plus si on lui demande plus de 17 chiffres). Elle y parvient cependant lorsque le nombre de chiffres

demandés est égal à 16, mais comme on s'y attend, l'imprécision sur les racines est énorme (de l'ordre de 0.3).

- Lorsque le degré du polynôme est élevé, la mémoire utilisée par l'algorithme de Schönhage est grande (40 Mo pour P_{101}).

13.4 Polynômes issus de l'étude de l'ensemble de Mandelbrot

Nous sommes en possession d'un programme efficace, amusons nous ! Rappelons que l'ensemble de Mandelbrot [17] est l'ensemble des points c du plan complexes tels que la suite définie par

$$z_0 = 0, \qquad z_{n+1} = c + z_n^2$$

ne diverge pas en module vers l'infini. En fait, cet ensemble est lié de très prés aux cycles de l'itération $z_{n+1} = c + z_n^2$. On se propose ici de déterminer les cycles d'ordre $k > 0$ (ou d'ordre divisant k), autrement dit on veut trouver les points c tels que dans la suite $z_{n+1} = c + z_n^2$, on ait $z_k = z_0 = 0$. On s'aperçoit que de tels points c sont les racines du polynôme $P_k(X)$, où la suite $P_n(X)$ est définie par :

$$P_0(X) = 0, \qquad P_{n+1}(X) = X + P_n(X)^2.$$

Tous les polynômes $P_n(X)$ s'annulent en 0. Si on pose $P_n(X) = XQ_n(X)$, les $Q_n(X)$ vérifient la récurrence :

$$Q_0 = 0, \qquad Q_{n+1}(X) = 1 + XQ_n(X)^2.$$

Les racines des Q_k donnent donc les points c de l'ensemble de Mandelbrot, autres que 0, engendrant un cycle d'ordre k. On retrouve également ces polynômes Q_n lors de l'étude des arbres binaires.

Le degré de Q_n est $2^n - 1$. Pour $n = 1, \ldots, 5$, tous les programmes testés précédemment trouvent les racines de Q_n. Pour $n = 6$, seuls l'algorithme de Schönhage et *NRoots* déterminent les racines. Pour $n = 7$, on a $\deg(Q_n) = 127$ et la fonction *NRoots* est mise en échec : la zone des racines se trouvant prés de la valeur -2 (voir figure 13) est identifiée à une racine multiple d'ordre 22 en -2. Seul l'algorithme de Schönhage donne des résultats satisfaisants (les racines ont été déterminées en 11 heures à 10^{-40} prés). Un bon nombre des racines s'accumulent apparemment le long de la frontière de l'ensemble de Mandelbrot comme on peut le voir sur la figure 13.

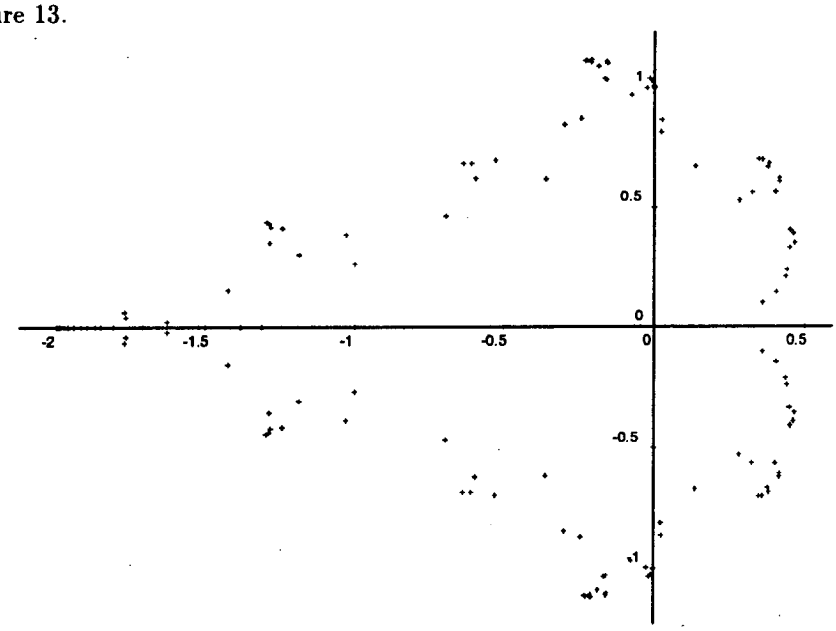


Figure 13. Les racines du polynôme Q_7 déterminées par l'algorithme de Schönhage

Conclusion

Les optimisations que nous avons faites de la méthode de Schönhage ont permis par rapport à l'algorithme initial de gagner un facteur temps de l'ordre de 10. En fait, c'est essentiellement sur l'intégration numérique autour du cercle de séparation que le gain est le plus important. D'autres optimisations sont à mon avis possibles, en particulier sur la recherche du cercle de séparation et permettraient de gagner encore considérablement.

Néanmoins, nous disposons maintenant d'un programme infaillible de recherche des racines d'un polynôme (ce qui n'avait encore jamais été réalisé). Certes, le temps d'exécution est encore un peu élevé par rapport aux autres algorithmes. Une implantation soignée en C utilisant des bibliothèques de calcul sur grands entiers permettraient de le rendre tout à fait utilisable.

La garantie des erreurs données par l'algorithme sur les racines autorise l'introduction de moyens numériques dans le calcul formel. Cette idée est nouvelle et permettrait parfois de diminuer considérablement le coût de certaines opérations formelles (comme par exemple la comparaison du module de deux racines de polynômes à coefficients entiers, opération très lourde à réaliser avec des moyens formels seuls).

Appendice

Les polynômes de Curtz

Dans la section 13.3, nous avons recherché les racines des polynômes de Curtz, définis par la récurrence

$$P_0(x) = 1, \qquad P_n(x) = x \sum_{q=0}^{n-1} \frac{(-1)^q}{q+1} P_{n-q-1}(x) + \frac{(-1)^n}{n+1}.$$

Ces relations donnent une formule explicite pour la série génératrice des $P_n(x)$

$$P(z,x) := \sum_{n=0}^{\infty} P_n(x) z^n = \frac{\log(1+z)}{z(1-x\log(1+z))}.$$

Comme il a été brièvement dit à la section 13.3, on a le résultat suivant.

Théorème 17 *Les polynômes de Curtz $P_n(x)$ n'ont aucune racine réelle lorsque n est pair. Ils possèdent une unique racine réelle lorsque n est impair, qui de plus est positive.*

Notre propos est de démontrer ce théorème. Une autre approche de la démonstration fait l'objet de [15].

Les singularités de $P(z,x)$ considérée comme fonction de z sont en

$$z_0 = -1 \quad \text{et} \quad z_1 = z_1(x) = e^{1/x} - 1.$$

Ces singularités dictent le comportement des coefficients $[z^n]P(z,x)$ (notation qui signifie : coefficient de z^n dans $P(z,x)$),*i.e* des $P_n(x)$. Plus précisément, si on intègre le long d'un contour de Hankel qui a la forme de celui de la figure 14, un calcul de résidus de l'intégrale de Cauchy donne :

$$P_n(x) - \frac{e^{1/x}}{x^2 z_1(x)^{n+2}} = \frac{1}{2i\pi} \oint P(z,x) \frac{dz}{z^{n+1}} = \frac{1}{2i\pi} \oint \frac{\log(1+z)}{1-x\log(1+z)} \frac{dz}{z^{n+2}}. \tag{.1}$$

En effet, les seuls résidus à prendre en compte se trouvent en $z = 0$ (d'où le terme $P_n(x)$) et en $z = z_1(x)$ (d'où le terme $-e^{1/x}/(x^2 z_1(x)^{n+2})$).

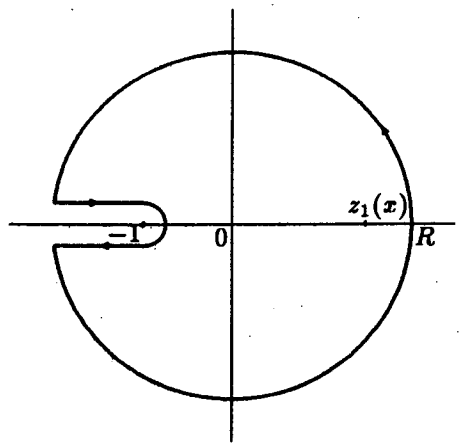


Figure 14. Le contour de Hankel utilisé dans l'analyse des polynômes $P_n(x)$

Ceci étant, on fait tendre le rayon du cercle extérieur R vers l'infini, puis l'on rapproche infinitésimalement les deux bord voisins de l'axe réel négatif. La comparaison des deux déterminations du logarithme fournit alors une représentation intégrale qui précise (.1). Après avoir effectué le changement de variable supplémentaire $z \mapsto -z$, on obtient la représentation intégrale exacte

$$P_n(x) = \frac{e^{1/x}}{x^2} \frac{1}{z_1(x)^{n+2}} + (-1)^n \int_1^{+\infty} \frac{1}{(1 - x\log(x-1))^2 + x^2\pi^2} \frac{dz}{z^{n+2}}. \tag{.2}$$

Lorsque n est pair, la représentation (.2) a tout ses termes strictement positifs et donc $\forall x, P_n(x) > 0$, d'où la première partie du théorème.

Lorsque n est impair, c'est plus délicat. Si $x < 0$, la représentation (.2) a ses deux termes strictement négatifs, et donc $P_n(x) < 0$. Si $x > 0$, la même représentation entraîne :

$$\forall x \geq \lambda = \frac{1}{\log 2}, \qquad P_n(x) \geq \frac{1}{x^2} - \int_1^{\infty} \frac{dz}{x^2\pi^2 z^{n+2}} = \frac{1}{x^2}\left[1 - \frac{1}{(n+1)\pi^2}\right] > 0.$$

Le théorème sera donc montré si on prouve que P_n croît strictement sur l'intervalle $]0, \lambda[$. Par dérivation de (.2) par rapport à x, on trouve, après arrangements

$$P_n'(x) = \frac{d}{dx}\left[\frac{e^{1/x}}{x^2 z_1(x)^{n+2}}\right] + \int_1^{\infty} \left(\frac{d}{dz}[-\frac{1}{D}]\frac{(z-1)\log(z-1)}{x} + \frac{2x\pi^2}{D^2}\right) \frac{dz}{z^{n+2}},$$

où $D = (1 - x\log(z-1))^2 + x^2\pi^2$. Le premier terme est la dérivée d'une fonction croissante, donc positif. Il reste à montrer que l'intégrale est positive pour $x \in]0, \lambda[$. Un peu de chirurgie règle le problème. On remarque déjà que l'intégrale est supérieure à

$$\int_1^{\infty} \frac{d}{dz}[-\frac{1}{D}]\frac{(z-1)\log(z-1)}{x} \frac{dz}{z^{n+2}}, \tag{.3}$$

puis on décompose cette dernière en deux morceaux

$$\int_1^{+\infty} = \int_1^2 + \int_2^{+\infty}.$$

L'intégrande est positive pour $z \in [1,2]$, donc

$$\int_1^2 \geq \int_{5/4}^{3/2} \geq \frac{1}{x}\int_{5/4}^{3/2} \frac{d}{dz}[\frac{1}{D}]\frac{\log 2}{4}\frac{dz}{z^{n+2}}$$

$$\geq \frac{1}{x}\frac{\log 2}{4}\left(\frac{2}{3}\right)^{n+2}\int_{5/4}^{3/2}\frac{d}{dz}[\frac{1}{D}] = \frac{1}{x}\frac{\log 2}{9}\left(\frac{2}{3}\right)^{n}\left(\frac{1}{D(3/2)} - \frac{1}{D(5/4)}\right).$$

Après calcul de ce dernier terme, on trouve donc

$$\int_1^2 \geq c_1\left(\frac{2}{3}\right)^n, \qquad \text{où } c_1 = \frac{2(\log 2)^2}{9(4+\pi^2/\log 4)(9+\pi^2/\log 4)}.$$

Majorons maintenant la partie négative de l'intégrale de 2 à l'infini de l'intégrande de (.3) :

$$\int_2^\infty \frac{2\log(z-1)}{D^2}\frac{dz}{z^{n+2}} \leq 2\int_2^\infty \frac{1}{D^2}\frac{dz}{z^{n+1}}.$$

On découpe la dernière intégrale de 2 à $s(x)$ puis de $s(x)$ à l'infini, où $s(x) = 1 + e^{1/2x}$. De 2 à $s(x)$, comme $D \geq 1/4$, on a

$$\int_2^{s(x)} \frac{1}{D^2}\frac{dz}{z^{n+1}} \leq 16\int_2^{s(x)}\frac{dz}{z^{n+1}} = \frac{16}{n2^n}.$$

Par ailleurs, comme $D \geq x^2\pi^2$,

$$\int_{s(x)}^\infty \frac{1}{D^2}\frac{dz}{z^{n+1}} \leq \int_{s(x)}^\infty \frac{1}{x^4\pi^4}\frac{dz}{z^{n+1}} = \frac{1}{x^4\pi^4}\frac{1}{ns(x)^n}.$$

Comme on le vérifie facilement, $xs(x) \geq 1/2$, donc ce dernier terme est majoré par

$$\frac{16}{\pi^4 n s(x)^{n-4}} \leq \frac{16^2}{\pi^4 n 2^n}.$$

Finalement, on obtient que la partie négative de l'intégrale (.3) de 2 à l'infini est majorée en valeur absolue par $c_2/(n2^n)$, où

$$c_2 = 2\left(16 + \frac{16^2}{\pi^4}\right).$$

Un calcul numérique montre que $c_1 \geq 5.10^{-4}$ et $c_2 \leq 38$, et comme on a montré que

$$\forall x \in]0,\lambda[, \quad P_n'(x) \geq c_1\left(\frac{2}{3}\right)^n - c_2\frac{1}{n2^n},$$

on vérifie facilement que ceci est positif pour $n \geq 29$. Les cas restant de n sont réglés à la main en utilisant les séquences de Sturm. D'où notre théorème.

Bibliographie

[1] P. Enflo B. Beauzamy, E. Bombieri and H. Montgomery. Products of polynomials in many variables. *Journal of Number Theory*, 36(2):219–245, 1990.

[2] B. Beauzamy. Products of polynomials and a priori estimates for coefficients in poynomial decompositions. *Journal of Symbolic Computation*, 1992. To appear.

[3] D. W. Boyd. Two sharp inequalities for the norm of a factor of a polynomial. *Mathematika*, 1992. To appear.

[4] P. Curtz. Problème soumis dans *La Gazette des Mathématiciens*, 52:44, Avril 1992.

[5] K. Dilcher. Zeros of Bernoulli, generalized Bernoulli and Euler polynomials. *Memoirs of the American Mathematical Society*, 73(386), May 1988.

[6] H. Dim and H. P. Mc Kean. *Fourier series and Integrals.* Academic Press, 1972.

[7] P. Glesser. *Bornes pour les algorithmes de factorisation des polynômes.* PhD thesis, Publication de l'I.R.M.A., 1990.

[8] P. Henrici. *Applied and computational complex analysis.* J. Wiley, New York, 1974.

[9] M. A. Jenkins and J. F. Traub. A three-stage variable-shift iteration for polynomial zeros and its relation to generalized Rayleigh iteration. *Numer. Math.*, 14:252–263, 1970.

[10] M. A. Jenkins and J. F. Traub. Principles for testing polynomial zerofinding programs. *ACM Transaction on Mathematical Software*, 1(1):26–34, March 1975.

[11] D. E. Knuth. *The art of computer programming*, volume 2: Seminumerical Algorithms. Addison-Wesley, 1973.

[12] M. Mignotte L. Cerlienco and F. Piras. Computing the measure of a polynomial. *Journal of Symbolic Computation*, 4:21–23, 1987.

[13] K. Mahler. An application of Jensen's formula to polynomials. *Mathematika*, 7:98–100, 1960.

[14] M. Mignotte. An inequality about factors of polynomials. *Mathematics of Computation*, 28(128), 1974.

[15] X. Gourdon et B. Salvy P. Flajolet. Sur une famille de polynômes issus de l'analyse numérique. *La Gazette des Mathématiciens*, 1992. à paraître.

[16] V. Pan. Algebraic complexity of computing polynomial zeros. *Computers and Mathematics with Applications*, 4(14):285–304, 1987.

[17] H.-O. Peitgen and P.-H. Richter. *The beauty of fractals.* Springer-Verlag, Berlin, 1986.

[18] G. Peters and J. H. Wilkinson. Practical problems arising in the solution of polynomial equations. *J. Inst. Maths Applics*, 8:16–35, 1971.

[19] G. Polya and G. Szego. *Problems and Theorems in analysis*, volume II. Springer-Verlag, New York, Heidelberg, Berlin, 4th edition, 1970.

[20] Schönhage. The fundamental theorem of algebra in terms of computational complexity. Technical report, Mathematisches Institut der Universität Tübingen, 1982. preliminary report.

[21] Schönhage. Equation solving in terms of computational complexity. In *Proceedings of the International Congress of Mathematicians*, pages 131–153, 1987. Berkeley, California, 1986.

[22] V. Trevisan. ***Univariate polynomial factorisation.*** **PhD thesis, Kent State University Graduate College,** May 1992.

Unité de Recherche INRIA Rocquencourt
Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)

Unité de Recherche INRIA Lorraine Technopôle de Nancy-Brabois - Campus Scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 VILLERS LES NANCY Cedex (France)
Unité de Recherche INRIA Rennes IRISA, Campus Universitaire de Beaulieu 35042 RENNES Cedex (France)
Unité de Recherche INRIA Rhône-Alpes 46, avenue Félix Viallet - 38031 GRENOBLE Cedex (France)
Unité de Recherche INRIA Sophia Antipolis 2004, route des Lucioles - B.P. 93 - 06902 SOPHIA ANTIPOLIS Cedex (France)

EDITEUR
INRIA - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)

ISSN 0249 - 6399