Calcul Scientifique et Symbolique *via* l'initiation à des logiciels de calcul (MHT304), Automne 2010

Alain Yger

27 septembre 2011

Table des matières

Chapitre 1

Introduction

1.1 Deux logiciels distincts pour deux tâches distinctes

Ce cours a un double objectif :

- d'une part vous familiariser avec la prise en main de deux logiciels, l'un dévolu au calcul symbolique (MAPLE12), l'autre au calcul scientifique (MATLAB10) et à la programmation sous ces environnements ;
- d'autre part, présenter les bases d'une première initiation au calcul symbolique et au calcul scientifique.

Si le second objectif puise ses sources dans vos acquis de L1 (en particulier en arithmétique, en analyse, en algèbre des polynômes ou des fractions rationnelles, ainsi qu'en algèbre linéaire) et dans les bases mathématiques que vous allez engranger tout au long du semestre (cours de MHT301 et MHT302), il arrivera aussi que ce cours anticipe certaines notions que vous ne serez amenés à voir qu'ultérieurement : ainsi certains concepts dont vous verrez plus tard une présentation théorique, tels la résolution numérique des équations ou systèmes différentiels, l'orthogonalité et ses conséquences, la mise en route d'algorithmes basés sur les méthodes de « point fixe » ou bien encore certains aspects du calcul matriciel, en particulier du point de vue spectral (valeurs propres, vecteurs propres).

Ce cours d'initiation au calcul scientifique et symbolique sera constamment illustré *via* l'utilisation d'un logiciel de calcul. Le mieux adapté au calcul symbolique (axé sur la manipulation des expressions formelles et par voie de conséquence le calcul « sans pertes » relevant plutôt de l'arithmétique) sera MAPLE12. Le mieux adapté au calcul scientifique (axé cette fois sur le calcul dans le champ des réels au service de la modélisation en mathématiques appliquées) sera MATLAB10. Ces deux logiciels seront alternativement utilisés tout au long de ce cours dans les scéances de TP depuis leur prise en main jusqu'à l'illustration des acquis de cours et à une initiation à la programmation sous leur environnement.

Le logiciel MAPLE12 est en libre accès à l'espace ALPHA ainsi qu'au CREMI et vous l'avez sans doute déjà testé en L1 au moins comme une grosse calculette. L'une des premières fonctions de base est la fonction *eval* permettant l'évaluation d'une expression $P(x)$ dépendant d'un symbole x lorsque l'on assigne à ce symbole une valeur. Pour vous convaincre que MAPLE12 est un logiciel de calcul symbolique, faites (comme on l'a fait en cours) le test d'évaluer $\sqrt{x}$ par exemple en $x = 346$. Le

logiciel vous retournera

$$\sqrt{346}$$

Mais si vous avez pris la précaution de déclarer 346 comme un nombre décimal (et non plus comme un entier traité de fait comme un symbole), par exemple si vous demandez l'évaluation de $\sqrt{x}$ en $x = 346.0$, MAPLE12 vous retournera

```
18.60107524
```

Le logiciel MAPLE12, conçu pour faire du calcul symbolique, c'est-à-dire de la manipulation d'expresions mathématiques formelles, aura ici été détourné de sa fonction pour traiter ce calcul comme un calcul scientifique (la valeur fournie ici pour $\sqrt{346}$ n'étant bien sûr qu'une valeur tronquée à certaines décimales après la virgule). On reviendra plus loin sur la représentation des réels en *virgule flottante* et le codage machine des nombres ou des symboles.

1.2 Les boucles logiques (*if/else, for, while*) sur 2 exemples

1.2.1 La recherche du PGCD de deux entiers a, b, $b > 0$

Rappelons ici la définition du plus grand diviseur commun (PGCD) de deux entiers positifs ou nuls non tous les deux nuls ainsi que celle du petit multiple commun (PPCM) de deux entiers a et b strictement positifs.

Définition 1.1 *Si a et b sont deux entiers positifs non tous les deux nuls, on appelle plus grand diviseur commun (en abrégé PGCD) de a et b le plus grand de tous les nombres entiers positifs divisant à la fois a et b. Si $b = 0$, le PGCD de a et b vaut donc a. On dit que a et b sont premiers entre eux si leur PGCD est égal à 1. Le PPCM de deux entiers strictement positifs a et b est le plus petit entier strictement positif multiple à la fois de a et de b et l'on a la relation*

$$\mathrm{PPCM}(a, b) \times \mathrm{PGCD}(a, b) = a \times b\,,\ \forall\, a, b \in \mathbb{N}^*\,.$$

Le calcul du PGCD de deux nombres entiers positifs a et b avec b non nul fait apparaître une démarche mathématique constructive (donc implémentable sur une machine) que l'on appelle un *algorithme*. Ce terme vient du surnom Al-Khwarizmi du mathématicien ouzbek Abu Ja'far Mohammed Ben Musa, 780-850, (dont le début du titre d'un des ouvrages fournit d'ailleurs aussi le mot *algèbre*).

Notons (comme sous la syntaxe du logiciel de calcul MATLAB que nous utilisons ici)

```
[q,r] = div(a,b)
```

l'instruction qui calcule, étant donnés deux nombres entiers tels que $b \neq 0$, l'unique couple (q, r) avec $r \in \{0, ..., b-1\}$ tel que $a = bq + r$ donné par la division euclidienne de a par b. On pourrait tout aussi bien utiliser les commandes de MAPLE12. Considérons la suite d'instructions qui conduit au calcul du PGCD de deux entiers a et b ; le nombre à calculer est noté PGCD dans la suite d'instructions ci-dessous :

```
function PGCD=PGCD(a,b);

x=a ;
y=b ;
while y>0
    [q,r] = div(x,y);
    if r==0
        PGCD = y;
        y = 0 ;
    else
        [q1,r1] = div(y,r);
        x = r;
        PGCD = x ;
        y=r1 ;
    end
end
```

Si l'on traduit ceci en français, voici la lecture :

```
fonction PGCD=PGCD(a,b);

x=a;
y=b;
tant que y est non nul, faire
   [q,r] = div(x,y);
   si r=0
     PGCD = y;
     y=0;
   sinon
     [q1,r1]= div(y,r);
     x=r;
     PGCD = x;
     y=r1;
   fin
fin
```

Si maintenant, on lit ceci en langage « mathématique » et de manière exhaustive (toutes les instructions sont listées et l'on n'effectue pas de ré-assignement des variables), la démarche que traduit cette routine est :

$$\begin{aligned} a &= bq + r \\ b &= rq_1 + r_1 \\ r &= r_1 q_2 + r_2 \\ \vdots &= \vdots \\ r_{N-2} &= q_N r_{N-1} + r_N \\ r_{N-1} &= r_N q_{N+1} + 0 \end{aligned}$$

(comme les restes successifs $r, r_1, ...$ décroissent strictement et qu'il y a un nombre fini d'entiers entre 0 et b, il vient forcément un moment où r_N divise r_{N-1}, r_N étant

le dernier reste obtenu non nul). Le PGCD de a et b est aussi celui de b et r, de r et r_1, et ainsi, en cascade, de r_N et 0 ; il vaut donc r_N.

CONCLUSION : *Le PGCD de a et b est donc égal au dernier reste non nul r_N dans ce très célèbre algorithme de division dit algorithme d'Euclide.*

Par exemple, pour $a = 13$ et $b = 4$,

$$\begin{aligned} 13 &= 4 \times 3 + 1 \\ 4 &= 4 \times 1 + 0 \end{aligned}$$

le dernier reste non nul étant ici $r_0 = 1$. On a donc $\mathrm{PGCD}\,(a, b) = 1$.

La routine MATLAB que l'on a mis en place pour implémenter la fonction PGCD inclut une boucle de calcul

```
while y > 0
  ...
  ...
  ...
end
```

Tant que la variable y garde une valeur strictement positive, une certaine opération est répétée en boucle. Cette variable y, ré-initialisée après chaque retour de boucle, matérialise le dernier reste atteint dans la division euclidienne (tant que se reste reste non nul). A l'intérieur même de cette boucle, figure un noeud de décision :

```
if  r = 0
    ...
    ...
    ...
else
    ...
    ...
    ...
end
```

La première alternative conduit à l'arrêt du processus précisément lors de la boucle en cours (puisque on assigne à y la valeur 0 sous cette alternative) tandis que la seconde appelle la boucle suivante avec des variables x et y qui entre temps ont été ré-initialisées.

Avec cet exemple de routine PGCD, on a un exemple concret tant de la commande « *while* » que de l'alternative « *if/else* » .

1.2.2 La résolution de l'identité de Bézout (le *lemme chinois* pour deux éléments $a, b \in \mathbb{Z}$)

Si a est un entier relatif, on pose $|a| = a$ si $a \in \mathbb{N}$, $|a| = a'$ si $a = -a'$ avec $a' \in \mathbb{N}$.

Si a et b sont deux entiers relatifs non tous les deux nuls, on appelle *plus grand diviseur commun* de a et b (ou encore $\mathrm{PGCD}(a, b)$) le PGCD des entiers positifs $|a|$ et $|b|$. Les nombres a et b sont premiers entre eux si ce PGCD vaut 1. On attribue au

mathématicien français Etienne Bézout (1730-1783) le résultat suivant, dont l'intérêt pratique tant en mathématiques pures qu'appliquées est aujourd'hui devenu capital (il conditionne ce que l'on appelle le *lemme des restes chinois* que vous verrez plus tard)[1].

Théorème 1.1 *Soient a et b deux nombres entiers relatifs non tous les deux nuls et d leur PGCD ; il existe au moins un couple $(u_0, v_0) \in \mathbb{Z}^2$ tel que*

$$au_0 + bv_0 = d$$

(une telle relation est appelée identité de Bézout lorsque $d = 1$).

Preuve. La construction de u_0 et v_0 est algorithmique et se fait en remontant depuis l'avant-dernière ligne les calculs faits dans l'algorithme de division euclidienne de $|a|$ par $|b|$ (on suppose ici $b \neq 0$).

$$\begin{aligned} |a| &= |b|q_0 + r_0 \\ |b| &= r_0 q_1 + r_1 \\ r_0 &= r_1 q_2 + r_2 \\ \vdots &= \vdots \\ r_{N-3} &= q_{N-1} r_{N-2} + r_{N-1} \\ r_{N-2} &= q_N r_{N-1} + d \\ r_{N-1} &= d q_{N+1} + 0 \end{aligned}$$

On écrit

$$\begin{aligned} d &= r_{N-2} - q_N r_{N-1} \\ &= r_{N-2} - q_N(r_{N-3} - q_{N-1} r_{N-2}) \\ &= -q_N r_{N-3} + (1 + q_N q_{N-1}) r_{N-2} \\ \vdots &= \vdots \\ &= u_0 a + v_0 b \end{aligned}$$

Plus que l'énoncé du théorème 2.1 lui même, ce qui est très important (parce que très utile) est qu'il s'agisse d'une assertion dont la démonstration est constructive, c'est-à-dire s'articule sur un algorithme. Voici, en quelques lignes de code, la fonction qui calcule, étant donnés deux entiers relatifs a et b avec $b \neq 0$ à la fois le PGCD d de a et b, mais aussi une paire d'entiers relatifs (u, v) telle que $d = au + bv$ (il s'agit, on le remarquera, d'un algorithme inductif qui s'auto-appelle) :

```
fonction [PGCD,u,v]=bezout(a,b);
x=a ;
```

[1] En voici un exemple d'application parlant : c'est grâce à ce type de résultat que l'on peut combiner deux prises de vue à des temps d'exposition premiers entre eux (disons par exemple a et b secondes) d'un mobile se déplacant à vitesse constante pour réaliser un cliché instantané net de ce mobile en mouvement (imaginez deux photos d'une avenue la nuit avec les traînées lumineuses des phares de voitures lorsque l'on veut précisément voir se fixer sur la pellicule le flux de véhicules immobile).

```
y= abs(b) ;
[q,r]=div(x,y);
if r==0
    PGCD = y;
    u=0 ;
    v=1;
else
 [d,u1,v1]=bezout(y,r);
 PGCD=d;
 u=v1;
 v=sign(b)*(u1- q*v1);
end
```

On retrouve ici une alternative

```
if
  ...
  ...
else
  ...
  ...
end
```

mais cette fois non plus une boucle

```
while  ...
   ...
   ...
   ...
end
```

mais (dans un des volets de l'alternative) un ré-appel au programme (ici appelé bezout) comme sous-programme de lui-même. Il faut prendre garde ici que la présence précisément de l'alternative interdit que la procédure ne boucle sur elle même. Cette alternative devient à une certaine étape un *test d'arrêt*. La notion dégagée ici est celle de *récursivité* en programmation.

1.3 Le calcul machine est-il fiable ?

Commençons par soumettre la machine à un test très simple, celui consistant à effectuer le calcul itératif du nombre

$$x = (10^n + 1.5) - 10^n\,,\ n = 0, 1, 2, ...$$

Le code de calcul que nous soumettons à la machine (ici sous MATLAB) est donc :

```
function test1
temp=1;
for i=1:30
    temp=10*temp;
    x=temp+1.5;
```

```
    y=temp;
    z=x-y;
    [i z]
end
```

Notons ici encore la présence d'une boucle de calculs :

```
for i=1:30
    ...
    ...
    ...
end
```

et la réinitialisation de la variable temp à chaque étape.

Voici la réponse obtenue au niveau de l'affichage des sorties (le numéro i de l'itération, puis la valeur correspondante z du résultat) :

```
1.0000    1.5000
2.0000    1.5000
3.0000    1.5000
4.0000    1.5000
5.0000    1.5000
6.0000    1.5000
7.0000    1.5000
8.0000    1.5000
9.0000    1.5000
10.0000    1.5000
11.0000    1.5000
12.0000    1.5000
13.0000    1.5000
14.0000    1.5000
15.0000    1.5000
16     2
17     0
18     0
19     0
20     0
21     0
22     0
23     0
24     0
25     0
26     0
27     0
28     0
29     0
30     0
```

Le constat est clair : au delà de la 16-ème itération, la machine ne rend plus le résultat escompté (qui bien sûr devrait être 1.5000). En fait, ce qui se passe ici est que la

taille des nombres impliqués a nécessité (au niveau de leur codage) un espace de travail dont le volume dépasse le seuil fixé par la machine : il faut savoir en effet que pour coder un nombre réel en double précision (ce que fait MATLAB par défaut), la machine dispose de 64 bits (seulement 32 bits pour le codage en *simple précision*). Nous verrons au paragraphe suivant comment ces 64 bits sont « organisés » pour le codage des réels en *virgule flottante*. Mais d'ores et déjà, nous nous rendons compte que la défaillance du calcul ici est manifestement liée à des erreurs d'arrondi, la capacité de discernement de la machine ne s'avérant plus suffisante lorsque les entrées sont des expressions faisant intervenir trop de *digits*, comme des entiers positifs de trop grande taille (comme c'est le cas ici lorsque l'exposant de 10 se met à dépasser 16).

1.4 Le codage des réels en *virgule flottante* et les erreurs d'arrondi

Tout nombre réel x (non nul, la machine n'aime pas en général le nombre zéro) se représente de manière unique sous la forme :

$$x = \pm\, 0.\underline{?}\,?\,?\,?\,?\,?\,?\,?\,?\,?\,?\ ...\ \times 10^{p}\,,$$

où la suite des points d'interrogation désigne une suite de chiffres pris entre 0 et 9, étant convenu que le premier de ces chiffres (celui qui est souligné) est pris dans la liste $\{1, ..., 9\}$, et où p désigne un entier relatif. Par exemple :

$$\begin{aligned} 2834 &= 0.2834 \times 10^{4} \\ -0.003756 &= -0.3756 \times 10^{-2}\,. \end{aligned}$$

Ce moyen de « coder »les nombres réels est le *codage en virgule flottante* (ici en décimal). Le nombre décimal positif

$$m := 0.\underline{?}\,?\,?\,?\,?\,?\,?\,?\,?\,?\,?\ ...$$

(toujours dans $[1/10, 1]$) s'appelle la *mantisse*, l'entier p s'appelle l'*exposant*.

C'est cette idée que la machine reprend à son compte pour coder les nombres réels, à la nuance (très importante près) que les nombres entiers naturels $n \in \mathbb{N}^*$ se doivent d'être écrits dans le système binaire (en base 2), c'est-à-dire sous la forme

$$n = a_0 2^N + a_1 2^{N-1} + \cdots + a_N\,,$$

avec $a_j \in \{0, 1\}$, $a_0 = 1$, au lieu de

$$n = \tilde{a}_0 10^N + \tilde{a}_1 10^{N-1} + \cdots + \tilde{a}_N\,,$$

avec $\tilde{a}_j \in \{0, 1, ..., 9\}$, $a_0 \in \{1, ..., 9\}$; par exemple l'entier 19 s'écrit

$$19 = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1$$

et se code donc en binaire

$$1\ 0\ 0\ 1\ 1$$

Si nous utilisons le codage des entiers en binaire (c'est-à-dire en base 2) au lieu du codage en décimal, on peut encore parler de codage des réels en virgule flottante, mais le codage doit être pensé sous la forme

$$x = \pm\, 0.\underline{?}\,?\,?\,?\,?\,?\,?\,?\,?\,?\,?\ \ldots\ \times 2^p\,,$$

où la mantisse

$$m := 0.\underline{?}\,?\,?\,?\,?\,?\,?\,?\,?\,?\,?\ \ldots$$

est maintenant un nombre réel de $[1/2, 1]$ que l'on décide d'approcher en binaire en ne conservant que s chiffres représentatifs (ce qui correspondrait, si l'on était en décimal, à s chiffres significatifs après la virgule) ; le premier chiffre se doit d'être un 1 (puisque le nombre et entre 1/2 et 1).

Suivant que l'on code un nombre réel en *double précision* ou *simple précision*, on utilise 64 bits ou 32 bits. En double précision, on conserve un des 64 bits pour coder le signe $\pm$, on réserve 11 de ces 64 bits pour coder (en binaire) l'exposant p et les 52 bits restants servent à coder (avec 52 chiffres « après la virgule », le codage étant toujours pensé en binaire) la mantisse m. En simple précision, on réserve 8 bits pour l'exposant p et 23 bits pour la mantisse, le dernier des 32 bits disponibles étant occupé par le codage du signe.

Si s est le nombre de chiffres codés dans le codage de la mantisse ($s = 52$ en double précision, $s = 23$ en simple précision), l'erreur d'arrondi entre une vraie mantisse m et la mantisse « codée »$\overline{m}$ est donc majorée en module par

$$|m - \overline{m}| \leq 2^{-s-1}$$

(on « arrondit »au nombre codable avec s chiffres le plus proche, qu'il soit plus petit ou plus grand que m, exactement comme on fait en décimal). L'erreur relative commise lorsque l'on arrondit $x = \pm m \times 2^p$ en $\overline{x} = \pm\overline{m} \times 2^p$ est donc majorée par

$$\frac{|x - \overline{x}|}{|x|} = \frac{|m - \overline{m}|}{m} \leq \frac{|m - \overline{m}|}{\frac{1}{2}} \leq 2 \times 2^{-s-1} = 2^{-s}\,.$$

Cette erreur relative 2^{-s} (s est le nombre de bits utilisés pour coder la mantisse, soit 52 en double précision, 23 en simple précision) est appelée *erreur machine* ; c'est elle qui est responsable pour les erreurs d'arrondi telles celles que nous avons observé dans le calcul lancé dans la section 1.3.

Pour donner un exemple sous MATLAB, considérons la suite récurrente (de nombres rationnels) initiée à $x_0 = -1$ et régie par la relation inductive :

$$x_{k+1} = x_k - \frac{x_k^3 + x_k^2 + 1}{3x_k^2 + 2x_k}\,.$$

On verra dans les cours à venir que cette suite converge vers l'unique racine réelle de l'équation algébrique

$$x^3 + x^2 + 1 = 0$$

qui appartient au segment $[-2, -1]$ sur lequel la fonction $f\ : x \longmapsto x^3 + x^2 + 1$ est strictement croissante avec $f(-2) = -3 < 0$ et $f(-1) = 1 > 0$ (l'existence de la racine résulte donc du théorème des valeurs intermédiaires). Pour continuer à vous familiariser avec les boucles de calcul, voici la routine (écrite ici sous MATLAB) fournissant x_N :

```
function x=Newton(init,N);
init=-1;
x=init;
for i=1:N
    x = x - (x^3 + x^2 +1)./(3*x^2 + 2*x);
end
```

Le résultat sous MATLAB, obtenu pour x_{50} est le suivant :

```
>> format long
>> x=Newton(50);
>> x

x =

  -1.465571231876768

>> single(x)

ans =

  -1.4655713
```

On constate que, si on ne lui précise rien, le logiciel travaille en double précision (« double ») et affiche (en format « long ») les nombres réels avec 15 décimales ; en simple précision (« single »), seules 7 décimales sont affichées. On pourra comparer avec la résolution numérique de

$$x^3 + x^2 + 1 = 0$$

sous MAPLE12 générée par la commande « fsolve » :

```
> fsolve (x^3+ x^2 + 1 =0)

                          -1.465571232
```

L'affichage fournit ici 9 décimales.

1.5 La force du calcul symbolique

Ce que nous venons de remarquer dans la section précédente à propos des limites (simple ou double précision pour le codage des nombres réels) ne vaut plus lorsqu'il s'agit de représenter sous forme décimale un nombre rationnel du type N/D où N et D sont deux entiers strictement positifs sous un logiciel de calcul symbolique tel MAPLE12 ou Mathematica. En effet, ce que peut faire le logiciel dans ce cas, c'est effectuer l'algorithme de division euclidienne de N par D (celui que vous connaissez depuis l'école primaire), ce autant de fois qu'on le veut, et afficher donc un nombre arbitrairement grand de décimales du nombre N/D. On obtient ici une valeur numérique de N/D avec une précision arbitraire (bien sûr, la seule limite est la capacité de stockage mémoire de la machine).

Illustrons cela avec une formule d'analyse (que vous prouverez dans le cours de MHT401) permettant d'obtenir le nombre π comme limite commune des deux suites adjacentes

$$u_n = 4\sum_{k=0}^{2n-1} \frac{(-1)^k}{2k+1}\,,\ n \geq 1$$

(cette suite croît) et

$$v_n = 4\sum_{k=0}^{2n} \frac{(-1)^k}{2k+1}\,,\ n \geq 1$$

(cette suite décroît), avec $\lim\limits_{n\to+\infty}(v_n - u_n) = 0$; ces deux suites sont des suites de nombres rationnels et l'on peut par exemple implémenter (ici sous Mathematica, mais vous pouvez tout aussi bien le faire sous MAPLE12 avec la commande « sum ») les calculs (sous formes d'expressions rationnelles, il s'agit donc ici de calcul symbolique et non scientifique) de u_{100} et v_{100} et de leurs développements décimaux avec au moins 100 décimales exactes après la virgule N[u,105] et N[v,105] (générés par l'algorithme de division euclidienne, donc par un algorithme de nature arithmétique). Voici le résultat (entraînez vous à le retrouver sous MAPLE12) :

```
u = Sum[ 4*(-1)^k/(2*k+1), {k,0, 2*100-1}]

731070388038968202720510198283781775 45
472107045424764412183032623507735395 74
264808462771826385227569137313742699 36
715193263560910895530282361264868908 26
6204946805311497184/
233077884665326398150824103062920077 32
931920841802367839890768331984845911 01
069034917139195411721198713640327975 68
478630600311502003066966840474629472 26
7552661106168111125

v = Sum[ 4*(-1)^k/(2*k+1), {k,0, 2*100}]

294091537142287554883527885924048172 26
666042608582540000644959155354541277 33
684464333240059162123140018917372134 34
896707021089171277119911094229110950 10
3818394313354582815284/
934642317507958856584804653282309510 09
057002575627495037961981011259232103 15
286830017728173601002006841697715182 49
599308707249123032298537030303264183 79
288617103573412561125

N[u,105]

3.13659268483881675041496970507761296671529135173151 39
56134844085589009291979003690727634551795494746897 96
```

```
N[v,105]

3.14656774718295640128778766018983241808686242405071595114731291351918385557002613461790886600995858729696
```

Ce résultat n'est guère convaincant ici car les deux suites adjacentes $(u_n)_n$ et $(v_n)_n$ ne convergent pas assez vite vers leur limite commune. Mais il existe (et on en reparlera beaucoup plus loin dans le cours) des méthodes pour modifier les suites (u_n) et (v_n) en des suites $(\tilde{u}_n)_n$ et $(\tilde{v}_n)_n$ convergeant toujours vers la même limite π, mais cette fois plus vite (on parle d'*accélération de convergence*). La transformation des suites peut reposer sur une manipulation mathématique subtile. John Machin, mathématicien anglais du XVIII-ème siècle, a proposé une transformation bien loin d'être évidente en remplaçant les suites $(u_n)_n$ et $(v_n)_n$ par

$$\begin{aligned}\tilde{u}_n &= 4\sum_{k=0}^{2n-1}\frac{(-1)^k}{2k+1}\Big(4(1/5)^{2k+1}-(1/239)^{2k+1}\Big)\,,\ n\geq 1\\ \tilde{v}_n &= 4\sum_{k=0}^{2n}\frac{(-1)^k}{2k+1}\Big(4(1/5)^{2k+1}-(1/239)^{2k+1}\Big)\,,\ n\geq 1\end{aligned}$$

La première est toujours croissante, la seconde décroissante, et l'on a toujours

$$\lim_{n\to+\infty}(\tilde{v}_n-\tilde{u}_n)=0\,.$$

La limite commune de ces deux suites adjacentes est égale à π (on l'admettra ici tout comme on a admis que la limite commune des deux suites adjacentes $(u_n)_n$ et $(v_n)_n$ valait aussi π). Voici ce que donnent cette fois les calculs de $\tilde{u}_{100}$ et $\tilde{v}_{100}$ et de leurs développements décimaux avec au moins 100 décimales exactes après la virgule N[u,102] et N[v,102]

```
u = Sum[ (4*(-1)^k/(2*k+1))* (4* (1/5)^(2k+1)- (1/239)^(2k+1)),
{k,0, 2*100-1}]

467561866566185148432408850447197460290967186627031513422
161029919334642681200697519716281767414189733920411006524
925696811973351366968145202980746983910098443854499455555
719434856692083198193735427362907458902670151859745516640
261696708310490427348665685933335496500332951027074386349
373775839012923580222892281745722142440284663511820122845
336718328487767480940774412351517155024342986534208406398
989542671751401239271107986155849849063593817922435934567
292378169366712429549542613330229871984838286463972620199
226090583960779228887168252364327332085197996854626231200
342664726052281632467797365178274532651668961720455996605
464283884611759976535766252994412831930906572925500903807
721535697113788273084489782069887453836731315212427319236
610402699924804825104345362671233177864459599272760453509
980009280145585767525099587354917225629726418842222505172
```

```
665636354128805865133743573449359637620184131015440418786
247970114280454361583884425758011879365626105530283474742
702820312132070722115881955305568253142433365109410777931
475152312401724132566399517696411020786919866076179344724
720718780468272681890911962567525682309704655552854842702
686646997387743541107238477695666373068436030908721790266
111139973429621862517073999016422217089645873567696617422
794982235545367543559438664793141793028412326848629305930
755885364069176395607183982084828140674792088627569899159
7003844858590736608/
148829564530563115494064362815452016776262864278439994987
481445930774830721823989247803333887324816997285418774951
362541482345828742672830532139012074558290891685987880447
788946310038423290406928149877404030087961215011047852595
477421679375460350285361332810541601349357211331494999076
736937138114363596861223799090702962539960018785087717036
408791939084488739759823788895768251762743748569746318016
146038682538279558400186629525722012153756312128600092781
562415632946743734722173781324535280476761689446064646255
413079270114819387448773407993422598890412893346915315488
332968808447105377619905436448241108238141236175673661828
673201828323116003865670410179411208130242815578314518421
194529452512404848936191000196410218773751044885514074428
963319023562787243638304051618460987596778141512556315483
823628411729748115013360945155336169200346783453979913063
278972377610896529339628555273333397045662112737889819714
383924258178412697568356473887096424457189233778644024975
145934657322693831175352598733297178653327338378486282 46
456648456319995398930414038959087477210089292980297222234
197453712424495475301380456015935730572083605635632055798
808465038846149565679211370840044566622768119344685911474
520865355289479109035319196236316916591646646952151713453
835287682790449335872452431336682286302697804940048183534
101153465215801892059481113525303778294395229409019520971
9240665435791015625
```

```
v = Sum[ (4*(-1)^k/(2*k+1))* (4* (1/5)^(2k+1)- (1/239)^(2k+1)),
 {k,0, 2*100}]
```

```
267743703835773795182666450112603520446035375089909737458
827283404973699099043472056312730151635570966207459665964
695564294157216279411688906948119065890985550444413955593
002454630452312555749344222497626285523793702987414749651
484093486195920325098345904778135520378345072918565598302
812072962451260743255583014501742533879736638572756642347
228632903080739546863628805522764583222128188824213127687
986591386221904050857243235062084434373723740293774506655
886746897623095430921629843094946240456482041714530353672
717826555694912913986777194612358288838474125977513720037
```

```
0190371261958159712732028326334434142293042324068010266324229595192835706067398960798073750517099002046494270516946113390714516853387925021348131669923557775628305469951676814383709148959598014276303685484043402123891524303592578077628562195823311181490267488046613803269962807823215068119332254175352249777820542203888096984212241050814984873591068853299248444746206495064179905878906595293238114685083803078740515831284510642763674276299307453283761601282160936544866542219152712080900312575501152875284035818254751767225515026213330089567568574725651416414978694743084375209014597281993361662496096698303889798037373091101619599112166058420643046167253288111013984126103644160436860532845083629962024874567894811565747150472912392055073840030892247680275977544390776149847418107621481428669938913737721039406523290771780468/8522546789439171459436791594552388236902603348124892881064127492194318578425461360048333670565325573881195256858606771681343108773819240289708378289982621244359055302003355797783181144038713262478206270063591661068644052679549076531468256974684845321750003199624177703324851637491656867946312731896139423850739539429694033053168873605552973548798370863326194007080446882128966254712034876207610629978778570274232393220155031163016139738753010908605103145250658102279748262432429517844463767055033872240057803054023391272406991697260375393025872730018973230680713226103490901800855753740407141849814483802945300427154696318754089310665397338210299529868986077929886566188683205258657538953486491034800695243765375246959768418720184927130690704557803097922950692232246468293553476968531318920761199933682022026809794328723860245885022141952411392189735806227770766344658007040399450125324971650768787055552569028850874478952876344770465953702670969485726539882266001639818242108948319184276104984081725279812211352672756308728010148420146380933252599407701804908751818042805883765253176917353911681050598018784106564805281730076066067210059319476773126407366145151382955385711542842855894505234592094466158343890620276744839777351952391130267408552426692570167813979949396556562812187085854480861423192704435223805204670543082994260204447373930743127540352738202855675808677915483713150024414062 5

N[u,105]

3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706798215

N[v,105]
```

```
3.14159265358979323846264338327950288419716939937510582097494459230781640628620899
86280348253421170679821
```

On constate que les 100 premières décimales de $\tilde{u}_{100}$ sont aussi les 100 premières décimales de $\tilde{v}_{100}$. Comme π est « pincé » entre ces deux nombres, on vient de trouver les 100 premières décimales de π.

1.6 A propos de stabilité : un exemple troublant !

Continuons cette introduction aux difficultés inhérentes au calcul (et pour l'instant non aux mathématiques !) par un exemple troublant.

Rappelons tout d'abord quelques bases concernant la résolution des systèmes linéaires de n équations à n inconnues

$$M \cdot X = B\,,$$

où M est une matrice $n \times n$ à coefficients réels et B un vecteur colonne à entrées réelles ; si le rang de la matrice M est égal à n (ce qui signifie $\det M \neq 0$), alors l'application

$$X \in \mathbb{R}^n \longmapsto M \cdot X$$

est bijective et l'unique vecteur colonne

$$X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

de l'équation $M \cdot X = B$ est donné par

$$X = M^{-1} \cdot B\,,$$

où

$$M^{-1} = \frac{1}{\det M}[\text{cofacteurs}\,(M)]^t \tag{1.1}$$

(A^t désignant la transposée d'une matrice A).

On verra au chapitre suivant pourquoi la méthode (algorithmique) du pivot s'avère plus judicieuse pour résoudre un tel système linéaire que la résolution *via* le calcul de la matrice M^{-1} (que nous utiliserons ici). Nous allons dans cette section utiliser MATLAB qui, comme son nom l'indique, un logiciel s'articulant essentiellement autour du *calcul matriciel*. On part d'une matrice M très simple que nous déclarons sous MATLAB (ce cours est une première initiation à cet environnement, initiation que vous allez approfondir en TP) :

```
>> M= [10  7  8  7 ; 7 5 6 5 ; 8 6 10 9 ; 7 5 9 10] ;
M =

    10     7     8     7
     7     5     6     5
     8     6    10     9
     7     5     9    10
```

Le calcul du déterminant de M montre que $\det M = 1$. On déclare un vecteur colonne B par

```
>> B = [32 ; 23 ; 33 ; 31];

B =

    32
    23
    33
    31
```

La résolution immédiate du système donne :

```
>> M^(-1)*B

ans =

    1.0000
    1.0000
    1.0000
    1.0000
```

Si ce problème numérique correspondait à un calcul numérique sollicité par le résultat d'une exprérience, il est vraisemblable que les entrées de l'appareil (représenté par l'action de la matrice M) sont connues non pas exactement, mais avec une marge d'erreur. Il en est de même pour les coordonnées de la « sortie »B. Pertubons donc légèrement les entrées de M et tentons à nouveau de résoudre le système. Les nouvelles matrices M et B perturbées sont les suivantes :

```
>> Mperturb=[10 7 8.05 7.05 ; 7.04 5.02 6 5 ;8 5.99 9.98 9 ; 7 5 9 10]

Mperturb =

   10.0000    7.0000    8.0500    7.0500
    7.0400    5.0200    6.0000    5.0000
    8.0000    5.9900    9.9800    9.0000
    7.0000    5.0000    9.0000   10.0000
```

Le résultat du calcul de $X = \widetilde{M}^{-1} \cdot B$ avec ces données (légèrement) perturbées est troublant :

```
>> Mperturb^(-1)*B

ans =

    1.1693
    0.6593
    1.1329
    0.9322
```

On voit que l'on est très loin de la solution

$$X = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

obtenue lorsque l'entrée M n'est pas perturbées !

On peut envisager de perturber aussi B, par exemple en

```
>> Bperturb=[32.01 ; 22.99 ; 33.01 ; 30.997]

Bperturb =

   32.0100
   22.9900
   33.0100
   30.9970
```

Le résultat du calcul de $\widetilde{M}^{-1} \cdot \widetilde{B}$ est encore pire (on y voit apparaître une entrée négative) !

```
>> Mperturb^(-1)*Bperturb

ans =

    1.9353
   -0.6112
    1.4540
    0.7420
```

Ce n'est pas ici la machine qui est en jeu, mais les mathématiques elles mêmes ! On peut deviner le problème en réalisant que

```
>> M^(-1)

ans =

   25.0000  -41.0000   10.0000   -6.0000
  -41.0000   68.0000  -17.0000   10.0000
   10.0000  -17.0000    5.0000   -3.0000
   -6.0000   10.0000   -3.0000    2.0000
```

L'inverse de M a des coefficients de taille significative, ce qui est une des raisons pour cette instabilité. La notion sous-jacente est en fait celle de *conditionnement* d'une matrice, mais l'on y viendra que lorsque vous aurez avancé dans le cours de MHT301 (la notion de valeur propre jouant un rôle important). On comprend en tout cas l'intérêt de développer des méthodes algorithmiques, telles le *pivot* de Gauss que vous avez rencontré en MAT201 ou les méthodes itératives basées sur le théorème du point fixe (Jacobi, Gauss-Seidel) dont nous parlerons dans ce cours (en les implémentant sous MAPLE12 ou MATLAB), plutôt que d'attaquer la résolution en inversant la matrice M. Notons d'ailleurs au passage que l'expression développée d'un déterminant d'ordre N implique $N!$ termes, ce qui fait du problème du calcul d'un gros déterminant un problème très vite d'une excessive complexité algorithmique ! Le calcul de M^{-1} directement suivant (1.1) est donc à déconseiller du point de vue complexité et de plus risque de mauvais conditionnement de matrice.

Chapitre 2

Résolution numérique des équations

2.1 Introduction : « *solve* » et « *fsolve* »

Soit $f(x) = 0$ une équation algébrique, par exemple

$$x^5 + \frac{x^3}{2} + 1 = 0\,.$$

Nous avons délibérément choisi ici une équation algébrique de degré supérieur ou égal à 5 dont on sait, d'après le théorème de Galois, que, du fait de la non résolubilité du sous groupe $\mathcal{A}_5$ de $\mathcal{S}_5$[1], elle ne saurait être résoluble par radicaux (sauf évidemment en cas de présence de racine évidente, ce qui ne semble pas le cas ici). Voici d'ailleurs, pour s'en convaincre, le retour que propose MAPLE12 à la commande « *solve* » appliquée à une telle équation :

```
{x = RootOf(2*_Z^5+_Z^3+2, index = 1)},
{x = RootOf(2*_Z^5+_Z^3+2, index = 2)},
{x = RootOf(2*_Z^5+_Z^3+2, index = 3)},
{x = RootOf(2*_Z^5+_Z^3+2, index = 4)},
{x = RootOf(2*_Z^5+_Z^3+2, index = 5)}
```

On constate que le logiciel de calcul symbolique n'a ici strictement rien pu faire ! Il aurait pu en revanche attaquer une équation de degré 3 ou de degré 4 car l'on sait, depuis les travaux de Cardan (degré 3) et Riccatti (degré 4) datant de la Renaissance, que ces équations sont explicitement résolubles par radicaux. Par exemple, pour l'équation

$$x^3 + \frac{x^2}{2} + x + 1 = 0\,,$$

voici la réponse de MAPLE12 à la commande « *solve* » :

```
{x = -1/6*(91+6*sqrt(267))^(1/3)+11/6/(91+6*sqrt(267))^(1/3)-1/6},
{x = 1/12*(91+6*sqrt(267))^(1/3)-11/12/(91+6*sqrt(267))^(1/3)-1/6
+1/2*I*sqrt(3)*(-1/6*(91+6*sqrt(267))^(1/3)-11/6/(91+6*sqrt(267))^(1/3))},
{x = 1/12*(91+6*sqrt(267))^(1/3)-11/12/(91+6*sqrt(267))^(1/3)-1/6
-1/2*I*sqrt(3)*(-1/6*(91+6*sqrt(267))^(1/3)-11/6/(91+6*sqrt(267))^(1/3))}
```

[1] Il s'agit ici de résultats algébriques de la théorie des groupes finis que vous aurez l'occasion de voir ultérieurement dans votre cursus, on les admet ici.

En revanche, la commande « Solve Numerically » (ou « *fsolve* ») fournit ici (par exemple sur l'exemple de $x^5 + x^3/2 + 1 = 0$, la solution réelle (unique) de cette équation de manière approchée, soit

```
{x = -.9098248906}
```

Ce sont les méthodes conduisant à ces résolutions numériques (elles aussi s'appuyant sur des algorithmes itératifs dont la convergence est assurée par le théorème de point fixe) que nous allons décrire dans ce chapitre.

2.2 Présentation des méthodes : Newton, fausse position, sécante et dichotomie

2.2.1 La méthode de Newton (présentation)

Considérons la fonction

$$x \longmapsto f(x) = x^5 + x^3/2 + 1$$

sur $[-1, -1/2]$. Nous savons qu'elle est croissante (strictement), que sa valeur en -1 vaut $-1/2$ et que sa valeur en $-1/2$ vaut $-1/32 - 1/16 + 1 > 0$. La fonction doit donc (d'après le théorème des valeurs intermédiaires) s'annuler en un unique point de $[-1, -1/2]$. Lançons l'algorithme itératif qui consiste, partant de $x_0 = -1$ à calculer de manière itérative les x_n suivant la règle

$$x_{k+1} = x_k - \frac{x_k^5 + x_k^3/2 + 1}{5x_k^4 + 3x_k^2/2} = x_k - \frac{f(x_k)}{f'(x_k)}\,.$$

En fait x_k est l'abcisse du point où la tangente en $(x_{k-1}, f(x_{k-1}))$ rencontre l'axe des x. Voici ici la routine itérative sous MATLAB :

```
function x=Newton2(init,N);
x=init;
for i=1:N
    x = x - (x^5+x^3/2+1)/(5*x^4+3*x^2/2);
end
```

Si l'on prend comme valeur initiale init=-1, on constate la convergence de la suite $(x_k)_k$:

```
>> Newton2(-1,3)
  -0.909825093948150
>> Newton2(-1,5)
  -0.909824890637916
>> Newton2(-1,10)
  -0.909824890637916
```

On constate que l'on récupère bien (ici en partant de $x_0 = -1$ et ce dès après 5 itérations, les 14 premières décimales de l'unique racine réelle ξ de l'équation $f(x) = 0$ (en double précision, MATLAB ne peut pas plus) appartenant à $]-1, -1/2[$. Le résultat est concordant avec celui que nous donnait la commande « *fsolve* » sous

MAPLE12 (voir l'introduction). C'est ce qu'il nous reste maintenant à clarifier et à comprendre.

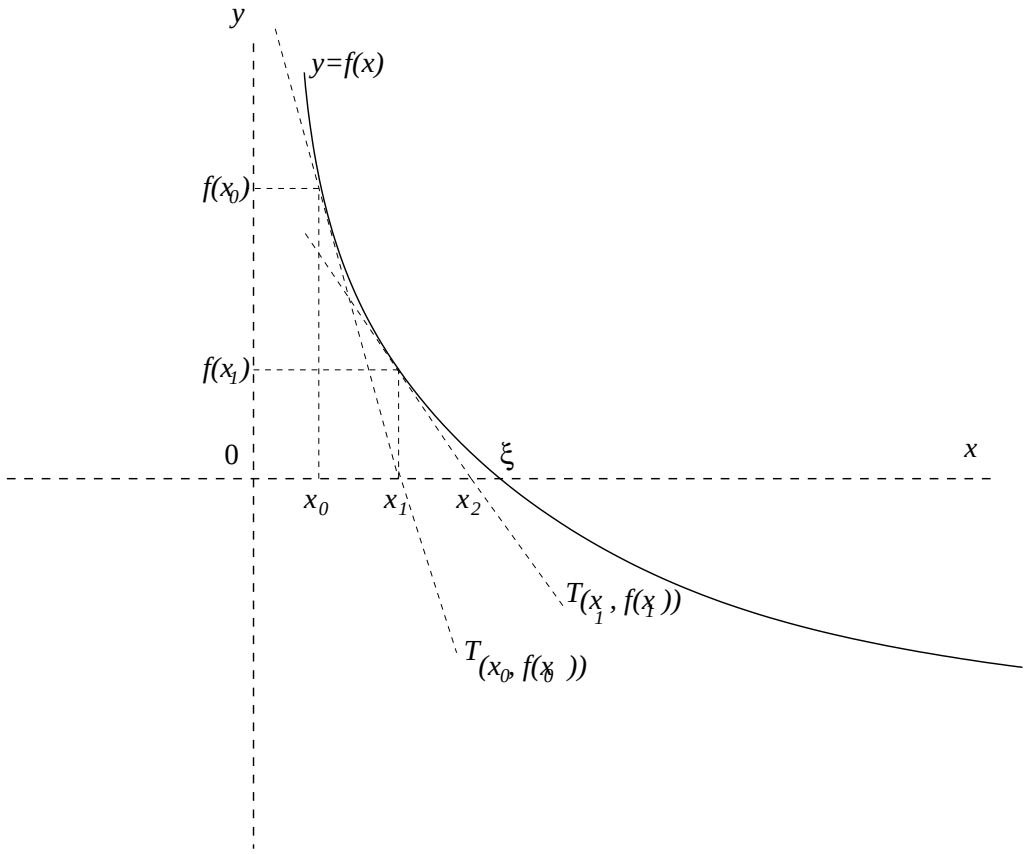

FIG. 2.1 – Méthode de Newton

Supposons que f soit une fonction de classe C^2 sur un intervalle ouvert I de $\mathbb{R}$, à valeurs réelles, et que a et b soient deux points de I tels que :

- $f(a)f(b) < 0$;
- $f' \neq 0$ sur $[a, b]$.

La seconde condition implique que f' reste soit strictement positive, soit strictement négative sur $[a, b]$. Nous avons ici fait la figure en supposant $f' < 0$ sur $[a, b]$ (il faut prendre $-f$ pour raccrocher à l'exemple précédent, où f était strictement croissante sur $[a, b]$ avec $a = -1$ et $b = -1/2$).

Pour faciliter la compréhension de la figure 2.1 sur laquelle nous nous appuyons, nous supposerons de plus (mais, on le verra, cette hypothèse n'a rien d'essentiel) que f'' ne s'annule pas sur $[a, b]$, c'est-à-dire que le graphe de f ne présente aucun changement de sens de concavité sur $[a, b]$: soit ce graphe « regarde toujours vers le haut » ($f'' > 0$ sur $[a, b]$, ce que nous supposerons), soit il regarde toujours vers le bas ($f'' < 0$ sur $[a, b]$).

Comme $f(a)f(b) < 0$ et que $f' < 0$ sur $[a, b]$, la fonction f est strictement décroissante sur $[a, b]$ et, d'après le théorème des valeurs intermédiaires [2] s'annule en un

[2]Revoir le cours de MIS101 (pour l'énoncé) et celui de MAT202 (pour la preuve et une étude plus poussée).

unique point $\xi \in]a, b[$. Notre but ici est de calculer ξ *via* une méthode itérative s'appuyant sur le théorème du point fixe.

On part de $x_0 = a$ et l'on construit la tangente T_0 au point $(x_0, f(x_0))$, tangente dont l'équation est

$$y - f(x_0) = f'(x_0)(x - x_0)\,.$$

Cette tangente coupe l'axe des abscisses au point $(x_1, 0)$ tel que

$$y(x_1) = 0 = f(x_0) + f'(x_0)(x_1 - x_0)\,,$$

soit

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \geq x_0\,.$$

Du fait de la propriété de concavité ($f'' > 0$ sur $[a, b]$), la tangente T_0 au point $(x_0, f(x_0))$ reste sous le graphe de f et le nombre x_1 (voir la figure) se trouve entre x_0 et l'abscisse inconnue ξ. On peut donc recommencer (car $x_1 \in [a, b]$). On construit donc la tangente T_1 au point $(x_1, f(x_1))$; cette tangente coupe l'axe des abscisses au point $(x_2, 0)$ avec

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} \in [x_1, \xi]\,.$$

Le procédé itératif consistant, partant de $x_0 = a$, à calculer de proche en proche les x_k, $k \geq 0$, *via* la relation de récurrence à un pas

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}\,,\ k \in \mathbb{N}\,,$$

génère une suite

$$x_0 \leq x_1 \leq x_2 \leq \cdots \leq \xi\,.$$

Cette suite est une suite croissante majorée, donc convergente (vers un point x_∞ de $[x_0, \xi] \subset [a, b]$). Comme

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}\,,\ k \in \mathbb{N}\,,$$

on a, en passant à la limite lorsque k tend vers $+\infty$ et en utilisant le fait que f et f' sont continues (avec $f' \neq 0$ sur $[x_0, \xi] \subset [a, b]$) :

$$x_\infty = x_\infty - \frac{f(x_\infty)}{f'(x_\infty)}\,,$$

d'où $f(x_\infty) = 0$, ce qui implique nécessairement $x_\infty = \xi$. La suite $(x_k)_{k\geq 0}$ approche donc le point ξ (ici en croissant) [3].

Il faut noter que, si l'algorithme converge sous les hypothèses de stricte croissance et de stricte convexité pour la fonction f sous lesquelles nous nous sommes placés[4], il n'en est pas de même en général, même si f' garde un signe fixe sur $[a, b]$; des changements de convavité du graphe peuvent nous faire sortir du cadre $[a, b]$. Même

[3] Vérifiez que si, suivant le même procédé, on était parti du point b, on aurait de même approché le point ξ, mais cette fois en décroissant.

[4] Même si, comme nous le verrons dans la section 2.3.2, ces hypothèzs de stricte concavité ou convexité peuvent être allégées.

si d'ailleurs la fonction f est de classe C^2 et strictement monotone (avec $f' > 0$ ou $f' < 0$) sur $\mathbb{R}$ tout entier, de tels changements de concavité peuvent aisément induire la non convergence de l'algorithme lorsqu'il est initié en un point arbitraire de $\mathbb{R}$; l'exemple de la fonction

$$x \in [-1, 1] \mapsto \frac{5x - x^3}{4},$$

pour lequel la méthode, initiée à -1 ou 1, ne fait que « rebondir » entre ces deux valeurs, en est un exemple (notons qu'il y a ici changement de concavité en $x = 0$).

Cette démarche algorithmique pour résoudre numériquement l'équation $f = 0$, est une démarche attribuée à Isaac Newton ; c'est l'*algorithme de Newton* [5]. Nous verrons à la section suivante (section 2.3) que le théorème du point fixe est impliqué dans la justification (si elle s'avère licite) de la convergence et que la méthode de Newton est une méthode dite d'ordre 2.

2.2.2 Les méthodes de *fausse position* et de la *sécante* (présentation)

Décrivons maintenant une méthode à deux pas, la *méthode de fausse position*. On se place sous les mêmes hypothèses que précédemment (f est une fonction de classe C^2 dans un intervalle ouvert I de $\mathbb{R}$, a et b sont deux points de I tels que $a < b$ et $f(a)f(b) < 0$, $f' \neq 0$ sur $[a, b]$). On suppose ici (par exemple) $f' < 0$ sur $[a, b]$ de plus (pour simplifier les choses) que f'' reste strictement positive sur $[a, b]$ (comme sur la figure 2.2). La fonction f s'annule (comme dans la section précédente) en un unique point ξ de $]a, b[$ que l'on se propose d'approcher. On part cette fois du couple de points $x_0 = a$, $x_1 = b$ (notre méthode sera cette fois une méthode à deux pas). On construit la droite (dite sécante) joignant les points $(x_0, f(x_0))$ et $(x_1, f(x_1))$; cette droite a pour équation

$$y - f(x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}\,(x - x_0)\,;$$

elle coupe l'axe des abscisses en un point c tel que

$$0 - f(x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}\,(c - x_1)\,,$$

soit

$$c = x_1 - f(x_1)\frac{x_1 - x_0}{f(x_1) - f(x_0)}\,.$$

Sur notre figure (du fait des propriétés de concavité), c reste à droite du point ξ, mais ceci n'est pas toujours le cas. Ici intervient un processus de décision : on calcule $f(c)$ et l'on opère comme suit :

– soit $f(c)$ est du signe de $f(x_1)$, auquel cas, on pose

$$\begin{aligned} x_2 &= x_0 \\ x_3 &= c \end{aligned}$$

[5] Philosophe, mathématicien, physicien, esprit « universel », l'anglais Isaac Newton (1642-1727) fut, avec le mathématicien prussien Leibniz, l'un des pères incontestés du calcul différentiel moderne.

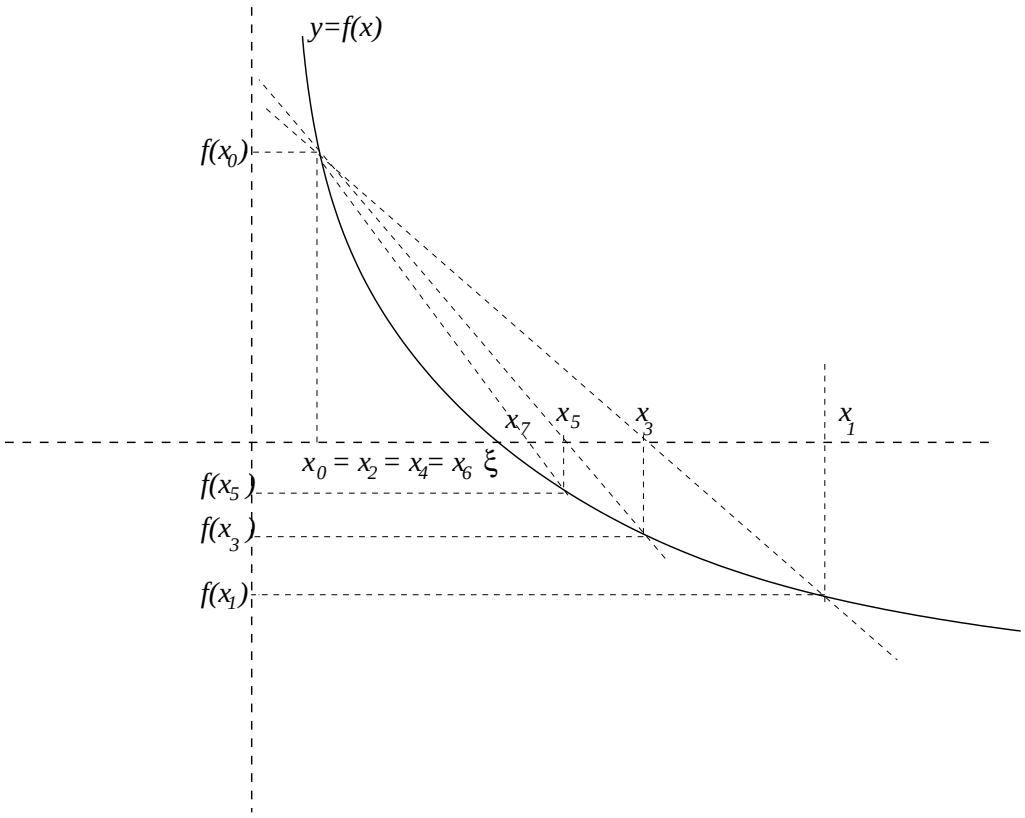

FIG. 2.2 – Méthode de « *fausse position* »

(de manière à ce que x_2 et x_3 continuent, comme x_0 et x_1, à encadrer le point ξ) ;

– soit $f(c)$ est du signe de $f(x_0)$, auquel cas, on pose

$$\begin{aligned} x_2 &= c \\ x_3 &= x_1 \end{aligned}$$

(de manière à ce que x_2 et x_3 continuent, comme x_0 et x_1, à encadrer le point ξ) .

On continue suivant ce processus pour construire un processus itératif à deux pas convergent vers le point ξ. Cette méthode est dite méthode de « *fausse position* » [6] ou aussi des « *regula falsi* ».

La *méthode de la sécante* est basée sur la même idée, excepté que l'on oublie le processus de choix basé sur le souci de constamment encadrer la limite potentielle ξ. La démarche algorithmique est plus simple ; on la lance à partir des valeurs initiales $x_0 = a$, $x_1 = b$. Le procédé inductif consiste ensuite à calculer de proche en proche

$$x_{k+1} = x_k - f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}\,,\ k = 1, 2, ... \tag{2.1}$$

On remarque que ce calcul ne nécessite pas (au contraire de l'algorithme itératif fondant la méthode de Newton) la connaissance de f', mais simplement celle de

[6] La valeur x_k est la « fausse position » du zéro ξ, fausse position que l'algorithme itératif conduit ici s'emploie à corriger.

f (comme pour la méthode de « *fausse position* » d'ailleurs). On pourra d'ailleurs préférer à la formule (2.1) la formulation plus « symétrique » :

$$x_{k+1} = \frac{x_{k-1}f(x_k) - x_k f(x_{k-1})}{f(x_k) - f(x_{k-1})}\,,\ k = 1, 2, ... \tag{2.2}$$

ce sera d'ailleurs cette seconde formulation itérative (2.2) que nous exploiterons dans la section 2.3 pour préciser la vitesse de convergence de la méthode et comparer les performances de cet algorithme avec celui de Newton.

Voici le type de programme à implémenter pour cet algorithme à deux pas :

```
function x=secante1(init1,init2,N);
x1=init1;
x2=init2;
for i=1:N
   y=x1;
   x1=x2;
   x2=x2 - f(x2)*(x2-y)/(f(x2)-f(y));
end
x=x2
```

(les valeurs initiales x_0 et x_1 sont ici init1 et init2 et N est le nombre d'itérations). La méthode de la sécante, initiée à partir de $x_0 = -1$, $x_1 = -1/2$ sur l'exemple où

$$f(x) = x^5 + \frac{x^3}{2} + 1$$

fournit, pour les approximations de l'unique racine ξ entre -1 et $-1/2$, les approximations successives suivantes :

```
>> secante4(-1,-.5,3)
  -0.891825801886447
>> secante4(-1,-.5,5)
  -0.909947528796009
>> secante4(-1,-.5,10)
  -0.909824890637916
>> secante4(-1,-.5,11)
Warning: Divide by zero.
   NaN
```

On constate qu'une fois que l'écart entre $f(x_k)$ et $f(x_{k+1})$ devient strictement inférieur à l'erreur machine, la division par 0 dans l'algorithme itératif devient inéluctable dès le cran suivant et l'on doit alors nécessairement s'arrêter. La vitesse de convergence de l'*algorithme de la sécante* semble ici comparable à celle de l'algorithme de Newton. De fait, l'ordre[7] de la méthode est inférieur à celui de la méthode de Newton. On verra en effet dans la section 2.3 à venir que ces méthodes de fausse position ou de la sécante sont des méthodes d'ordre le *nombre d'or* $\frac{1+\sqrt{5}}{2} \simeq 1.618$ (que l'on retrouve, on verra pourquoi, dans l'explicitation des termes de la *suite de Fibonacci* régie par la relation inductive $\alpha_{k+1} = \alpha_k + \alpha_{k-1}$ pour $k \geq 1$).

[7]Cette notion sera précisée dans la section 2.3 à venir.

2.2.3 La méthode de « *dichotomie* » (présentation)

La méthode de dichotomie est aussi une méthode à deux pas, mais cette fois (on le verra dans la section 2.3 à venir) d'ordre $q = 1$ [8] ; on dira aussi que c'est une méthode *linéaire*.

Son principe est très simple : on part encore d'un segment $[a, b]$ sur laquelle f est dérivable, telle que $f(a)f(b) < 0$, avec $f' \neq 0$ sur $[a, b]$. On part de $x_0 = a$, $x_1 = b$ et on calcule

$$c = \frac{x_0 + x_1}{2}\,.$$

Si $f(x_0)f(c) < 0$, on pose $x_2 = x_0$ et $x_3 = c$ et on continue ; si $f(x_0)f(c) \geq 0$, on pose $x_2 = c$ et $x_3 = x_1$ et on poursuit ainsi, suivant le processus itératif décrit dans la routine suivante :

```
function x=dichot(init1,init2,N);
x1=init1;
x2=init2;
for i=1:N
   y=(x1+x2)/2;
   u=f(x1)*f(y);
   if u <= 0
   x1=x1;
   x2=y;
   else
       x1=y;
       x2=x2;
   end
end
x=x2
```

Cette méthode (seulement linéaire, comme l'est, ce qui justifie l'épithète, le calcul de $y = (x_k + x_{k-1})/2$ en fonction de x_k et x_{k-1}) peut être néanmoins exploitée pour déterminer les nombres a et b à partir lesquels on s'autorisera à lancer une méthode plus rapide, offrant ainsi un premier « dégrossissage » de la situation avec une localisation grossière du zéro éventuel ξ entre $x_0 = a$ et $x_1 = b$. Sur notre exemple

$$x \longmapsto f(x) = x^5 + \frac{x^3}{2} + 1\,,$$

la méthode de dichotomie initiée avec $x_0 = -1$ et $x_1 = -1/2$ fournit par exemple :

```
>> dichot(-1,-.5,3)
  -0.875000000000000
>> dichot(-1,-.5,5)
  -0.906250000000000
>> dichot(-1,-.5,10)
  -0.909667968750000
>> dichot(-1,-.5,20)
  -0.909824848175049
>> dichot(-1,-.5,50)
```

[8] La notion d'*ordre* d'une méthode itérative y sera alors précisée.

```
  -0.909824890637916
>> dichot(-1,-.5,100)
  -0.909824890637916
```

On constate que la vitesse de convergence ici est loin d'être comparable à celle donnée par l'algorithme de Newton. Ceci sera clarifié dans la section suivante 2.3.

2.3 Retour sur les méthodes : comparaison *via* la notion d'ordre

2.3.1 Nombre de pas et *ordre* d'une méthode itérative

Une notion importante, celle d'*ordre*, rend compte de la rapidité de convergence, donc de l'efficacité, d'une méthode itérative (lorsque celle ci génère une suite $(x_k)_k$ convergente), tandis qu'une autre notion (celle de *nombre de pas*) rend compte de la capacité de stockage mémoire nécessaire pour implémenter la méthode récursive :

Définition 2.1 *Soit p un entier positif non nul. Une méthode itérative basée sur l'utilisation d'une relation inductive*

$$x_{k+1} = F(x_k, x_{k-1}, ..., x_{k-(p-1)}) \,,\ k \geq p-1, ,$$

et démarrant donc avec des données initiales $x_0, ..., x_{p-1}$ est dite méthode itérative à p pas. On dit que cette méthode est d'ordre $q \in]0, +\infty[$ si et seulement si q est la borne supérieure de l'ensemble des nombres réels strictement positifs q tels que, dès que la suite $(x_k)_{k\geq 0}$ converge vers une limite finie x_∞, on a

$$\limsup_{k\to+\infty} \frac{|x_{k+1} - x_\infty|}{|x_k - x_\infty|^q} < \infty \,.$$

Remarque 3.1. Souvent, on se contente d'une minoration q_0 de q, ce qui nous permet de dire que l'ordre de la méthode est « au moins » égal à q_0. Pour prouver que l'ordre est exactement égal à q_0, il convient en plus d'exhiber une suite $(x_k)_{k\geq 0}$ générée par l'algorithme itératif à partir des p données initiales $x_0, ..., x_{p-1}$, convergent vers une limite x_∞ (qu'il convient donc de connaître, ce qui complique le problème), et telle que, pour cette suite

$$\liminf_{k\to+\infty} \frac{|x_{k+1} - x_\infty|}{|x_k - x_\infty|^{q_0}} = l \in]0, \infty[\ ;$$

ceci n'est pas toujours si facile ! En revanche, un minorant de l'ordre est donné par tout entier q_0 tel que, pour toute suite $(x_k)_{k\geq 0}$ générée par l'algorithme et convergent vers une limite finie x_∞, il existe une constante C (dépendant *a priori* de la suite $(x_k)_{k\geq 0}$) et telle que

$$|x_{k+1} - x_\infty| \leq C|x_k - x_\infty|^{q_0} \,,\ \forall k \geq 0 \,.$$

2.3.2 La méthode de Newton est au moins d'ordre 2

Reprenons la méthode de Newton générée à partir de $x_0 \in [a, b]$ (on suppose f de classe C^2 sur $[a, b]$ avec $f' \neq 0$ sur cet intervalle) régie par la relation inductive

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \,,\ k \geq 0 \,.$$

On suppose ici que tous les x_k, $k \geq 0$ restent dans le segment $[a, b]$ et que la suite $(x_k)_{k\geq 0}$ converge vers un zéro ξ de f, ce qui est par exemple le cas si f est décroissante et strictement convexe sur $[a, b]$ ($f'' > 0$) et si $f(a)f(b) < 0$ (comme on l'a vu à la section 2.2.1 ci-dessus).

On peut préciser la vitesse de convergence de $(x_k)_{k\geq 0}$ vers ξ en utilisant la formule de Taylor-Lagrange au second ordre. Soit $m \in \mathbb{N}$. Rappelons que si f est une fonction m fois dérivable dans un intervalle ouvert I de $\mathbb{R}$, alors, si $\alpha < \beta$ sont deux points de I et si f est $m+1$ fois dérivable sur l'intervalle ouvert $]\alpha, \beta[$, il existe $\theta \in]\alpha, \beta[$ tel que

$$f(\beta) = \sum_{k=0}^{m} \frac{f^{(k)}(\alpha)}{k!}(\beta - \alpha)^k + \frac{f^{(m+1)}(\theta)}{(m+1)!}(\beta - \alpha)^{m+1}\,.$$

Lorsque $m = 0$, cette formule est la *formule des accroissements finis* [9]. Si l'on utilise ici ce résultat avec $\alpha = x_k$, $\beta = \xi$, $m = 1$, il vient

$$\exists \theta_k \in]x_k, \xi[\,,\ f(\xi) = 0 = f(x_k) + f'(x_k)(\xi - x_k) + \frac{f''(\theta_k)}{2}(\xi - x_k)^2\,.$$

En divisant par $f'(x_k) \neq 0$, il vient

$$0 = \frac{f(x_k)}{f'(x_k)} + (\xi - x_k) + \frac{f''(\theta_k)}{2f'(x_k)}(\xi - x_k)^2\,,$$

ce qui s'écrit encore

$$\xi - \Big(x_k - \frac{f(x_k)}{f'(x_k)}\Big) = -\frac{f''(\theta_k)}{2f'(x_k)}(\xi - x_k)^2\,,$$

c'est-à-dire

$$\xi - x_{k+1} = -\frac{f''(\theta_k)}{2f'(x_k)}(\xi - x_k)^2\,.$$

On constate donc que

$$\lim_{k\to+\infty} \Big(\frac{|\xi - x_{k+1}|}{|\xi - x_k|^2}\Big) = \frac{|f''(\xi)|}{2|f'(\xi)|} \neq 0\,.$$

Si M est un majorant de $|f''|/(2|f'|)$ sur $[a, b]$ et si p est le premier cran tel que $|\xi - x_p| < 1$, on a, au niveau des majorations d'erreur, pour $k \geq p$:

$$|\xi - x_k| \leq M|\xi - x_{k-1}|^2 \leq M^2|\xi - x_k|^4 \leq \cdots \leq M^{k-p}|\xi - x_p|^{2^{k-p}}\,.$$

On constate donc que la convergence de $(x_k)_{k\geq 0}$ vers ξ s'inscrit bien dans le cadre d'un théorème de point fixe et que cette convergence est ici très rapide ($10^{-2^{10}}$ vaut déjà 10^{-1024} !).

On peut aussi remarquer, compte tenu de la définition de l'ordre, que *la méthode de Newton est une méthode d'ordre au moins* 2. Appliquée dans un intervalle $[a, b]$ sur lequel $f'' \neq 0$, c'est une méthode d'ordre exactement 2. Quand bien même f'' s'annule, nous pouvons énoncer la Proposition suivante :

[9] On a énoncé le résultat ici sous les hypothèses les plus faibles ; on pourra se contenter du résultat lorsque f est supposée de classe C^{m+1} sur l'intervalle ouvert I, ce qui est probablement l'énoncé dont vous disposez depuis le cours de MHT202.

Proposition 2.1 *Soit f une fonction de classe C^2 sur l'intervalle $]\xi - r, \xi + r[$, s'annulant en $x = \xi$, et telle que f' ne s'annule pas dans $]\xi - r, \xi + r[$. Soit*

$$\gamma = \sup_{x,u \in]\xi - r, \xi + r[} \frac{|f''(u)|}{|f'(x)|}.$$

Si $\gamma r < 2$, la suite de Newton $(x_n)_{n\in\mathbb{N}}$ initiée en un point x_0 de $]\xi - r, \xi + r[$ converge vers ξ lorsque n tend vers l'infini et l'on a même

$$|x_k - \xi| \leq \Big(\frac{\gamma r}{2}\Big)^{2^k - 1} |x_0 - \xi| \quad \forall\, k \in \mathbb{N}. \tag{2.3}$$

Preuve. Prenons $x \in]\xi - r, \xi + r[$ et supposons que

$$N_f(x) = x - \frac{f(x)}{f'(x)}$$

appartienne encore à $]\xi - r, \xi + r[$. En utilisant la formule de Taylor avec reste intégral (deux fois, une fois avec f, une fois avec f'), il vient

$$\begin{aligned}
N_f(x) - \xi &= x - \xi - \frac{f(x) - f(\xi)}{f'(x)} \\
&= \frac{1}{f'(x)}\Big(f'(x)(x - \xi) - (f(x) - f(\xi))\Big) \\
&= \frac{1}{f'(x)}\Big((x - \xi)\Big(f'(\xi) + (x - \xi)\int_0^1 f''(tx + (1 - t)\xi)\, dt\Big) \\
&\qquad -\Big(f'(\xi)(x - \xi) + (x - \xi)^2 \int_0^1 (1 - t) f''(tx + (1 - t)\xi)\, dt\Big)\Big) \\
&= (x - \xi)^2 \int_0^1 t \frac{f''(tx + (1 - t)\xi)}{f'(x)}\, dt.
\end{aligned}$$

On en déduit l'estimation

$$|N_f(x) - \xi| \leq \frac{\gamma}{2}|x - \xi|^2. \tag{2.4}$$

Si nous supposons que $x_0, x_1, ..., x_k$ restent dans $]\xi - r, \xi + r[$ et que

$$|x_k - \xi| \leq \Big(\frac{\gamma r}{2}\Big)^{2^k - 1} |x_0 - \xi|,$$

on constate grâce à (2.4) que

$$|x_{k+1} - \xi| \leq \frac{\gamma}{2}|x_k - \xi|^2 \leq \frac{\gamma}{2}\Big(\frac{\gamma r}{2}\Big)^{2^{k+1} - 2} |x_0 - \xi|^2 \leq \Big(\frac{\gamma r}{2}\Big)^{2^{k+1} - 1} |x_0 - \xi|,$$

ce qui prouve l'inégalité (2.3) par récurrence. Comme $\gamma r/2 < 1$, il y a bien convergence vers 0 de la suite géométrique de raison $\gamma r/2$, donc convergence de la suite de Newton $(x_k)_k$ vers la racine ξ de f. $\diamondsuit$

2.3.3 Ordre de la méthode de la sécante (ou de la méthode de « fausse position »)

La méthode de la sécante (ou celle de « fausse position ») sont des méthodes à deux pas, car le calcul de x_{k+1} (que l'on y intègre ou non comme dans la méthode de « fausse position » un processus de décision) nécessite de disposer au préalable des deux entrées x_k et x_{k-1}.

Concernant l'ordre, nous nous contenterons ici de donner une minoration de l'ordre de la méthode de la sécante (ce serait la même chose pour la méthode parente de « fausse position »). L'ordre de ces méthodes est au moins égal au nombre d'or

$$q = \frac{1+\sqrt{5}}{2} \simeq 1.618...$$

Essayons de le montrer en supposant pour simplifier que le point cherché ξ (limite de la suite $(x_k)_{k\geq 0}$) est $\xi = 0$ et toujours que la fonction f est de classe C^2 dans un voisinage (ouvert) segment $[a, b]$ voisinage (ouvert) de 0 et contenant tous les x_k, $k \geq 0$.

On peut écrire, on l'a vu, la relation entre x_{k-1}, x_k et x_{k+1} sous la forme plus « symétrique » (2.2) :

$$x_{k+1} = \frac{x_{k-1}f(x_k) - x_k f(x_{k-1})}{f(x_k) - f(x_{k-1})}\,.$$

Ainsi a t'on, si $e_k := x_k - \xi = x_k$,

$$|e_{k+1}| = |e_{k-1}|\,|e_k|\,|\varphi(x_{k-1}, x_k)|\,,$$

avec

$$\varphi(u, v) := \frac{\frac{f(u)}{u} - \frac{f(v)}{v}}{f(u) - f(v)}\,.$$

La fonction de deux variables

$$(u, v) \longmapsto \varphi(u, v)$$

est en fait bornée sur $[a, b]$: on utilise en effet au numérateur et au dénominateur la formule des accroissements finis, applicable tant au numérateur qu'au dénominateur car les hypothèses faites sur f (de classe C^2 avec $f(0) = 0$) impliquent que

$$t \longmapsto \frac{f(t)}{t}$$

(prolongée par $f'(0)$ en 0) est de classe C^1 sur $[a, b]$. Si l'on désigne par M un majorant de φ dans l'intervalle $[a, b]^2$ où l'on travaille et que l'on pose $\rho_k = M|e_k|$, $k \in \mathbb{N}$, on constate que

$$\rho_{k+1} \leq \rho_k \rho_{k-1}\,,\ k \geq 1\,.$$

En prenant le logarithme, il vient

$$\log \rho_{k+1} \leq \log \rho_k + \log \rho_{k-1}\,.$$

Si l'on introduit la suite récurrente définie par $\alpha_0 = \log \rho_0 < 0$, $\alpha_1 = \log \rho_1 < 0$ (quitte à supposer que x_0 et x_1 sont déjà dans $]-1/M, 1/M[$, on peut supposer

$\rho_0 < 1$ et $\rho_1 < 1$) et par la relation inductive $\alpha_{k+1} = \alpha_k + \alpha_{k-1}$ pour $k \geq 1$ [10], on constate (par récurrence sur k) que

$$\forall k \in \mathbb{N}\,,\ \log \rho_k \leq \alpha_k\,.$$

Mais l'on sait que

$$\alpha_k = \lambda\xi^k + \mu\eta^k = \lambda\Big(\frac{1+\sqrt{5}}{2}\Big)^k + \mu\Big(\frac{1-\sqrt{5}}{2}\Big)^k,$$

où ξ et η sont les racines de $X^2 - X - 1 = 0$ (voir le cours de MHT202 pour l'étude des suites récurrentes à deux pas) ; la constante μ (calculée à partir des conditions initiales $\alpha_0 < 0$ et $\alpha_1 < 0$) étant strictement négative (car tous les α_k le sont par récurrence), on a

$$\alpha_k \leq -C\Big(\frac{1+\sqrt{5}}{2}\Big)^k$$

pour k assez grand, donc, en passant aux exponentielles, toujours pour k assez grand,

$$|e_k| \leq (e^{-C})^{\left(\frac{1+\sqrt{5}}{2}\right)^k},$$

avec $C > 0$.

Ce petit raisonnement nous montre que *l'ordre de la méthode de la sécante est au moins égal au nombre d'or*. En le complétant, on prouverait aussi, si $f'' \neq 0$ sur $[a, b]$, que la méthode de la sécante est exactement d'ordre le nombre d'or.

2.3.4 Ordre de la méthode de dichotomie

Comme la méthode repose sur la relation inductive

$$x_{k+1} = \frac{x_k + x_{k-1}}{2}$$

(x_{k+1} est une fonction linéaire des deux entrées x_k et x_{k-1}), la méthode de dichotomie (qui est aussi une méthode à deux pas) est dite *linéaire*. On vérifie de plus par récurrence que

$$x_{k+1} - \xi = \frac{1}{2}(x_k - \xi) + \frac{1}{2}(x_{k-1} - \xi)$$

et donc que

$$e_{k+1} = \frac{e_k + e_{k-1}}{2}\,.$$

Si $e_{k+1} = O(e_k^q)$ avec $q > 1$, on aurait

$$e_k = -e_{k-1} + O(e_k^q)\,,$$

donc $e_k \simeq -e_{k-1}$, ce qui imposerait $q \leq 1$. *L'ordre de la méthode de dichotomie est donc exactement égal à* 1. C'est la méthode certes d'ordre le plus mauvais (ordre 1) parmi les ordres des quatre méthodes envisagées, mais, néanmoins, c'est elle dont on est le plus souvent assuré (sans restrictions du type convexité ou autre) de la convergence [11]. C'est donc une méthode à ne jamais oublier !

[10] La même que celle qui donne les nombres de Fibonacci.

[11] Par exemple, si l'on prend $f(x) = \frac{5x-x^3}{4}$ (il y a un unique zéro ξ de f dans l'intervalle $[-1, 1]$ sur lequel $f' > 0$), la méthode de Newton initiée à $x = 1$ ne converge pas (on trouve $x_k = \pm 1$, ce à cause de la présence de l'inflexion en $\xi = x = 0$), tandis que la méthode de dichotomie, démarrant avec $x_0 \in [-1, 0[$ et $x_1 = 1$ est ici, certes lente, mais infaillible, approchant bien sûr le zéro 0.

Chapitre 3

Polynômes ; élimination, interpolation, approximation

Ce chapitre est consacré aux polynômes ; l'anneau $\mathbb{K}[X]$ sera l'anneau des polynômes à coefficients dans un corps commutatif $\mathbb{K}$, tel qu'il a été introduit par exemple dans l'UE MHT201 ; dans ce cours, on prendra essentiellement les trois exemples $\mathbb{K} = \mathbb{Q}, \mathbb{R}, \mathbb{C}$, mais l'on pourra aussi envisager de remplacer $\mathbb{K}$ par le corps $\mathbb{K}(Y)$ des fractions rationnelles en une variable Y (jouant le rôle de paramètre) sur $\mathbb{K}$.

3.1 Quelques rappels sur l'anneau des polynômes $\mathbb{K}[X]$ et la division euclidienne

On utilisera ici essentiellement la structure d'anneau euclidien de $\mathbb{K}[X]$, liée au fait qu'existe dans $\mathbb{K}[X]$ un *algorithme de division euclidienne* : étant donnés $H \in \mathbb{K}[X]$ et $P \in \mathbb{K}[X]$, avec $\deg P = N \geq 0$, il existe un unique couple (Q, R) de $\mathbb{K}[X] \times \mathbb{K}[X]$ avec

$$H(X) = P(X)Q(X) + R(X)$$

et $\deg R < N$ (le polynôme identiquement nul étant considéré comme de degré $-\infty$).

Si P_1 et P_2 sont deux polynômes non tous les deux nuls de $\mathbb{K}[X]$, l'ensemble des polynômes H de $\mathbb{K}[X]$ de la forme

$$H(X) = P_1(X)A_1(X) + P_2(X)A_2(X)$$

coïncide avec l'ensemble des polynômes multiples du dernier reste non nul Δ dans l'algorithme de division euclidienne de P_1 par P_2 (si $\deg P_2 \geq 0$) ou de P_2 par P_1 (si $\deg P_1 \geq 0$). Le polynôme non nul Δ de degré inférieur à $\max(\deg P_1, \deg P_2)$ ainsi défini est appelé PGCD (*plus grand diviseur commun*) de P_1 et P_2 ; il est défini à la multiplication par un élément non nul de $\mathbb{K}$ près.

C'est en remontant les calculs conduits dans l'algorithme de division euclidienne (de P_1 par P_2 ou de P_2 par P_1) que l'on réalise une identité polynômiale (dite *identité de Bézout* [1]) de la forme

$$\Delta(X) = U(X)P_1(X) + V(X)P_2(X)$$

[1] Si les travaux du mathématicien et professeur français Etienne Bézout (1730-1783) traitent tant de la résolution des équations algébriques que de l'élimination (où il fit réellement œuvre de précurseur, ses travaux devant être repris au siècle suivant par exemple par Cayley et Kronecker),

avec $\deg U \leq \deg P_2 - 1$ et $\deg V \leq \deg P_1 - 1$. En particulier, si $P_1 \equiv 0$, on a $\Delta = P_2$ et l'identité de Bézout devient simplement $\Delta = P_2 = 1 \times P_2$.

Lorsque $\mathbb{K} = \mathbb{C}$, le fait que $\deg(\mathrm{PGCD}(P_1, P_2) > 0$ équivaut à ce que P_1 et P_2 aient une racine commune dans $\mathbb{C}$. Ceci résulte du théorème fondamental de l'algèbre, à savoir le théorème de Jean Lerond d'Alembert, suivant lequel tout polynôme à coefficients complexes et de degré N se factorise dans $\mathbb{C}[X]$ sous la forme

$$P(X) = a_0 \prod_{j=1}^{N} (X - \lambda_j)\,,$$

où $\lambda_1, ..., \lambda_N$ sont des nombres complexes ; si λ_j est répété exactement m_j fois dans cette liste, on dit que l'entier strictement positif m_j est la multiplicité de λ_j comme zédu polynôme P ; dire que $\lambda_j \in \mathbb{C}$ est zéro de multiplicité $m_j \in \mathbb{N}^*$ équivaut à dire

$$P(\lambda_j) = P'(\lambda_j) = \cdots P^{(m_j-1)}(\lambda_j) = 0\,,$$

où $P^{(k)}$ désigne, pour $k \in \mathbb{N}$, le k-ème polynôme dérivé de P.

3.2 La division suivant les puissances croissantes

Tout aussi important, sinon plus, que l'algorithme de division euclidienne (qui consiste à traiter la division en présentant les polynômes de manière à ce que les monômes soient rangés dans l'ordre décroissant, ce qui correspond à la présentation de Hörner, voir la section 3.3 suivante), l'algorithme de division suivant les puissances croissantes est initié avec deux entrées polynomiales A et B dont les monômes sont rangés cette fois dans l'ordre croissant :

$$A(X) = a_0 + a_1 X + \cdots + a_N X^N\,,\ B(X) = b_0 + b_1 X + \ldots + b_0 X^M.$$

Pour démarrer, il faut (ceci est essentiel) que $b_0 \neq 0$. On écrit

$$A(X) = \frac{a_0}{b_0} B(X) + A_1(X)$$

avec

$$A_1(X) = \alpha_{11} X + \alpha_{12} X^2 + \cdots$$

(les termes constants sont asbsents de A_1). On peut donc recommencer et diviser le « reste » A_1 par B :

$$A_1(X) = \frac{\alpha_{11}}{b_0} X B(X) + A_2(X),$$

avec

$$A_2(X) = \alpha_{22} X^2 + \alpha_{23} X^3 + \ldots$$

(les termes de degré 0 et 1 sont absents de A_2). On continue de la sorte et l'on trouve après k opérations (ce processus est sans fin, contrairement à celui de la division euclidienne)

$$A(X) = B(X)(\gamma_0 + \gamma_1 X + \cdots + \gamma_k X^k) + B_k(X),$$

on n'y trouve nulle trace de cette fameuse identité ! On peut néanmoins penser que l'influence des travaux de Bézout dans l'enseignement (écoles d'artillerie ou de marine) à travers ses divers traités didactiques explique cette terminologie.

avec

$$B_k(X) = \alpha_{k+1,k+1}X^{k+1} + \alpha_{k+1,k+2}X^{k+2} + \cdots$$

Le polynôme B_k est appelé *reste à l'ordre k* dans le processus de division suivant les puissances croissantes, tandis que

$$Q_k(X) = \sum_{j=0}^{k} \gamma_j X^j$$

est appelé *quotient à l'ordre k* dans ce processus.

Vous avez rencontré cet algorithme très important en analyse, par exemple dans la recherche de développements limités pour des quotients. Il est tout aussi aisément implémentable que celui de la division euclidienne.

3.3 Le « codage » d'un polynôme ; l'algorithme de Hörner

Un polynôme

$$P(X) = a_0X^N + a_1X^{N-1} + \cdots + a_N$$

avec coefficients dans un corps commutatif $\mathbb{K}$ ou même un anneau commutatif unitaire $\mathbb{A}$ se code par son degré N et par la liste des coefficients

$$[a_0, a_1, \cdots, a_N]$$

présentée comme une matrice ligne à $N+1$ entrées. Les coefficients sont rangés dans l'ordre ci dessus, en commencant par le coefficient (non nul) du monôme de degré maximal (ce degré vaut ici N) ; si un monôme tel X^k ne figure pas, on code l'entrée a_{N-k} correspondante comme un zéro. Par exemple, sous MATLAB, on déclare par

```
>> P = [-3 4 1 0 2 -6];
```

le polynôme de degré 5 :

$$P(X) = -3X^5 + 4X^4 + X^3 + 2X - 6\,.$$

Les $N+1$ entrées a_k, $k = 0, ..., N$, qui sont les coefficients du polynôme, sont soit des éléments de $\mathbb{Z}$ ou $\mathbb{Q}$ (que l'on code alors « sans pertes » en travaillant sous un logiciel de calcul symbolique, tel MAPLE12), soit des entrées réelles ou complexes (dont on code des approximations en virgule flottante sous un logiciel de calcul scientifique tel MATLAB). Les entrées $a_0, ..., a_N$ peuvent aussi être des expressions formelles ou des mots fabriqués à partir d'un nombre fini de symboles, ce qui se passe lorsque l'on envisage de travailler avec le polynôme P sous l'angle du *calcul symbolique*.

L'évaluation (on dit aussi la « spécification ») du polynôme en un élément ou un symbole x (qui peut être un nombre ou une expression formelle pourvu que l'on sache donner un sens aux expressions :

$$x \cdot a + \tilde{a}$$

lorsque $a, \tilde{a}$ sont deux éléments de $\mathbb{A}$ et à l'itération de ce processus), se fait par l'algorithme itératif décrit par la formule

$$a_N + \left[x \cdot \left[a_{N-1} + \cdots + \left[x \cdot \left[x \cdot [x \cdot a_0 + a_1] + a_2 \right] + a_3 \right] \cdots \right] \right]$$

et par le synopsis algorithmique suivant :

```
P=a(0);
pour i=1:N
      P = x * P + a(i);
      i = i+1 ;
fin
```

Cet algorithme « consommme » N multiplications et N additions (en fait N opérations du type $x*(\cdot)+(\cdot)$) ; la méthode naïve consistant à calculer toutes les puissances x^k, $k = 1, ..., N$ nécessite aussi N multiplications mais en plus de l'espace mémoire (ce que ne nécessite pas la démarche algorithmique présentée ci dessus) ; il reste ensuite (toujours si l'on suit cette méthode naïve) N multiplications à effectuer pour évaluer

$$\sum_{k=0}^{N} a_k \cdot x^{N-k} = P(x) \ ;$$

cette méthode naïve s'avère donc plus coûteuse (en temps et en espace mémoire). L'algorithme décrit dans le synopsis présenté plus haut (synopsis synthétisant une boucle, un test d'arrêt et un branchement) est dit *algorithme de Hörner*[2]. C'est lui que les commandes telles que

```
>> P = [-3 4 1 0 2 -6];
>> y = polyval (P,x);
```

(sous MATLAB) exploitent pour le calcul du vecteur ligne de scalaires

$$(P(x_0), ..., P(x_M))$$

si $x_0, x_1, ..., x_M$ sont $M + 1$ valeurs (ici nombres réels ou complexes car on travaille avec un logiciel de calcul scientifique) spécifiées.

Sous MAPLE12, ce sont les commandent du type

```
> eval (P(x),x)
> eval (f(x,y),x)
```

qui réalisent l'évaluation d'un polynôme $P(X)$ (dont les coefficients peuvent tout aussi bien être des expression symboliques, par exemple des polynômes en une nouvelle variable Y indépendante de X) sur un scalaire x ou une expression symbolique (par exemple $f(X, Y)$, où f est une expression rationnelle en X et Y).

[2]C'est au mathématicien anglais William George Hörner (1786-1837) que l'on attribue cet algorithme, même si probablement l'origine remonte bien au delà dans l'histoire des mathématiques et du calcul.

3.4 Une opération banale mais coûteuse : la multiplication des polyômes

Multiplier deux polynômes conduit (pour le calcul des coefficients) aux formules de Cauchy :

$$\Big(a_0 + a_1X + \cdots + a_NX^N\Big)\Big(b_0 + b_1X + \cdots + b_MX^M\Big) = \sum_{k=0}^{N+M} c_kX^k\,,$$

avec

$$c_k = \sum_{k_1+k_2=k} a_{k_1}b_{k_2},\ k = 0, ..., N+M.$$

Multiplier deux polyômes de degré N et M consomme donc $(N+1)(M+1)$ multiplications. Il suffit de penser à la multiplication de deux polynômes de degré 10000 pour entrevoir le coût du calcul !

Le problème de la multiplication rapide des polynômes (c'est-à-dire de leur multiplication « à moindre coût ») a été résolu par les ingénieurs informaticiens américains J.W. Cooley et J.W. Tukey vers 1965. On y reviendra dans la section 3.7 (sous-section 3.7.2), une fois évoquée l'interpolation par des polynômes trigonométriques.

Il était utile de signaler dès à présent cette difficulté. L'algorithme de Cooley-Tukey déclencha, on le verra, ce qui fut vers les années 1970 la « révolution numérique ».

3.5 L'interpolation de Lagrange

3.5.1 Définition et formule

Soient $x_0 = a, ..., x_N = b$, $N+1$ nombres réels ou complexes distincts. Lorsque l'on pense à des nombres réels, on pensera à $x_0 = a < x_1 < \cdots < x_{N-1} < x_N = b$ comme des points intermédiaires d'un intervalle $[a, b]$ fermé borné donné. Dans notre exemple, le vecteur

$$X = (x_0, ..., x_N)$$

est le vecteur (ici à $21 = N+1$ entrées)

```
>> x=-8:16/20:8;
```

des 21 points régulièrement espacés entre -8 et 8

$$x_k := -8 + \frac{16k}{20}\ , ...,\ k = 0, ..., 20\,.$$

Associons à ces valeurs x_k, $k = 0, ..., N$, des valeurs numériques $(y_0, ..., y_N)$, réelles ou complexes. Dans l'exemple que nous traitons ici, nous prendrons par exemple

```
>> y = 1./ (1+ x.^2);
```

ce qui signifie

$$y = (y_0, ..., y_N)$$

et

$$y_k = \frac{1}{1+x_k^2}\,,\ k = 0, ..., N\,.$$

Les points (x_k, y_k) affichés ici avec des croix sont des points du graphe de la fonction

$$f \ : x \in [-8, 8] \longmapsto \frac{1}{1+x^2} \, .$$

Comme on le voit sur la figure ci-dessous, le graphe de la fonction affine par morceaux interpolant ces points (en plein sur la figure) ne rend pas compte (par exemple sur $[-2, 2]$) de la forme du graphe de f, en pointillés sur la figure, qui présente par exemple une tangente horizontale au point $(0, 1)$.

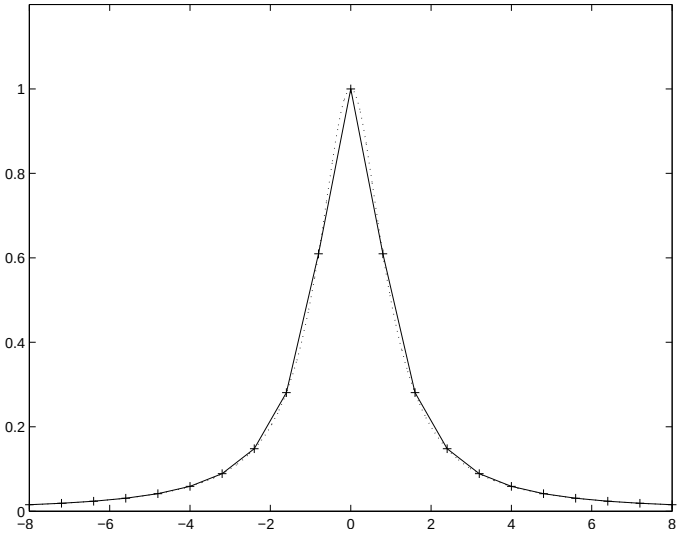

FIG. 3.1 – Le graphe de $x \longmapsto 1/(1+x^2)$ sur $[-8, 8]$ et les points (x_k, y_k) « marqués »

Le $\mathbb{R}$ (*resp.* $\mathbb{C}$)- espace vectoriel des polynômes à coefficients réels (*resp.* complexes) de degré au plus N est un $\mathbb{R}$ (*resp.* $\mathbb{C}$)-espace vectoriel de dimension $N+1$ dont une base est constituée de l'ensemble des monômes $\{1, X, X^2, ..., X^N\}$. Ecrire qu'un tel polynôme

$$P(X) = \sum_{k=0}^{N} a_k X^{N-k}$$

prend des valeurs spécifiéees $y_0, ..., y_N$ (réelles ou complexes) aux points distincts $x_0, ..., x_N$ revient à écrire les $N+1$ contraintes

$$P(x_k) = \sum_{k=0}^{N} a_k x_j^k = y_j \, , \ j = 0, ..., N \, .$$

Ces contraintes correspondent à un système linéaire

$$M(x_0, ..., x_N) \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_N \end{pmatrix} \tag{3.1}$$

où $M(x_0, ..., x_N)$ est une matrice carrée dont on vérifie que le déterminant (dit de Vandermonde [3]) est égal à

$$\begin{vmatrix} x_0^N & x_0^{N-1} & \dots & x_0 & 1 \\ x_1^N & x_1^{N-1} & \dots & x_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \\ x_{N-1}^N & x_{N-1}^{N-1} & \dots & x_{N-1} & 1 \\ x_N^N & x_N^{N-1} & \dots & x_N & 1 \end{vmatrix} = \prod_{0 \le k < l \le N} (x_k - x_l) \neq 0 \,.$$

Ceci prouve que le système (3.1) est un système de Cramer qui admet une unique solution $(a_0, ..., a_N)$. Le polynôme

$$P_N(X\,; Y) = a_0 X^N + a_1 X^{N-1} + \cdots + a_N$$

correspondant est l'unique polynôme de degré N prenant les valeurs prescrites des coordonnées du vecteur ligne Y

$$Y = (y_0, ..., y_N)\,,$$

(dans cet ordre) respectivement aux points $x_0, ..., x_N$. Ce polynôme est dit *polynôme d'interpolation de Lagrange*[4] attaché aux données

$$\{x_0, y_0\}\,,\ \{x_1, y_1\}\,,\ \dots\,,\ \{x_N, y_N\}\,.$$

La matrice $M(x_0, ..., x_N)$ est en général mal conditionnée et l'inverser n'est pas la bonne stratégie !

Certes, on vérifie immédiatement, si l'on pose

$$P(X) = (X - x_0) \cdots (X - x_N)\,,$$

que notre polynôme cherché s'écrit

$$\begin{aligned} P_N(X; Y) &= \sum_{k=0}^{N} \frac{P_k(X) - P_k(x_k)}{P_k'(x_k)(X - x_k)}\, y_k \\ &= \sum_{k=0}^{N} \frac{P_k(X)}{P_k'(x_k)(X - x_k)}\, y_k \\ &= \sum_{k=0}^{N} \frac{\prod\limits_{\substack{l=0 \\ l \neq k}}^{N} (X - x_l)}{\prod\limits_{\substack{l=0 \\ l \neq k}}^{N} (x_k - x_l)}\, y_k \,. \end{aligned} \tag{3.2}$$

Si l'on pose en effet $X = x_k$, on trouve bien $P_N(x_k; Y) = y_k$ pour $k = 0, ..., N$; le polynôme $P_N(\cdot\,; Y)$ étant l'unique polynôme ayant cette propriété, c'est bien lui qui est donné explicitement par la formule (3.2) ci-dessus.

[3] Mathématicien français contemporain d'Etienne Bézout, Alexandre-Théophile Vandermonde (1735-1796) s'est intéressé à la résolution des équations algébriques en même temps qu'à la combinatoire ; le dernier des quatre articles qu'il a rédigé est sans doute l'article fondateur de la théorie des déterminants.

[4] Mathématicien franco-italien (1736-1813), il marqua profondément tant l'algèbre que l'analyse (en particulier le calcul différentiel) et la mécanique ; ce fut aussi un astronome.

Tout ce que nous avons dit dans cette section s'adapte au cas où $x_0, ..., x_N$ sont $N+1$ nombres complexes distincts et $y_0, ..., y_N$ nombres complexes. Dans ce nouveau contexte, avant de clôre cette section, il n'est pas inintéressant de remarquer que la construction de polynômes d'interpolation de Lagrange fournit, lorsque $\mathbb{K} = \mathbb{C}$, une résolution directe de l'identité de Bézout. On a en effet la

Proposition 3.1 *Soient P et Q deux polynômes à coefficients complexes, de degrés respectifs $p > 0$ et $q > 0$, n'ayant tous les deux que des racines simples ($\lambda_1, ..., \lambda_p$ pour P, $\mu_1, ..., \mu_q$ pour Q)[5]. Si P et Q n'ont aucune racine commune et si A désigne le polynôme d'interpolation de Lagrange (de degré exactement $q-1$) interpolant les valeurs $1/P(\mu_j)$, avec $j = 1, ..., q$ aux points $\mu_1, ..., \mu_q$ et B le polynôme d'interpolation de Lagrange (de degré exactement $p-1$) interpolant les valeurs $1/Q(\lambda_j)$, avec $j = 1, ..., p$, aux points $\lambda_1, ..., \lambda_p$, on a l'identité de Bézout*

$$1 = A(X)P(X) + B(X)Q(X)\,.$$

Preuve. Il suffit de remarquer que le polynôme

$$1 - A(X)P(X) - B(X)Q(X)$$

s'annule en les $p+q$ points distincts $\lambda_1, ..., \lambda_p, \mu_1, ..., \mu_q$ et est de degré au plus $p+q-1$. C'est donc le polynôme identiquement nul et la proposition est ainsi démontrée. ♢

Remarque 4.1. Si cette proposition est d'un intérêt mineur lorsque l'on travaille avec des polynômes en une variable (l'algorithme de division euclidienne étant dans ce cas un outil algorithmique de complexité optimale), il n'en est plus de même lorsque l'on recherche (comme c'est le cas dans nombre de problèmes pratiques surgis par exemple de la robotique) une identité de Bézout

$$1 = A_1(X_1, ..., X_n)P_1(X_1, ..., X_n) + \cdots + A_m(X_1, ..., X_n)P_m(X_1, ..., X_n)\,,$$

où $P_1, ..., P_m$ sont m polynômes en n variables sans zéros communs dans $\mathbb{C}^n$[6]. Dans ce cas, le recours à des identités obtenues directement suivant le mécanisme suggéré par l'interpolation Lagrange peut s'avèrer dans bien des cas algorithmiquement moins complexe.

[5]Cette condition de simplicité des zéros peut être levée, mais il faut disposer de la notion de polynôme d'interpolation de Lagrange à des points éventuellement multiples, notion que nous n'avons pas développé en cours, mais qui pourra être introduite par exemple en TP. En fait, on a l'identité de Bézout dès que l'on pose (racines multiples ou non pour P ou Q)

$$\begin{aligned} A(X) &= \sum_{j=1}^{q} \mathrm{Res}_{Y=\mu_j}\Big[\frac{Q(X)}{(X-Y)P(Y)Q(Y)}\Big] \\ B(X) &= \sum_{j=1}^{p} \mathrm{Res}_{Y=\lambda_j}\Big[\frac{P(X)}{(X-Y)P(Y)Q(Y)}\Big] \end{aligned}$$

si le résidu en $u = 0$ d'une fraction rationnelle $R(u) = N(u)/u^m D(u)$ (avec $D(0) \neq 0$) est le coefficient de u^{-1} dans le développement de R obtenu après division suivant les puissances croissantes de N par D. En effet, on constate que $A(X)P(X) + B(X)Q(X)$ s'écrit aussi

$$\sum_{j=1}^{q} \mathrm{Res}_{Y=\mu_j}\Big[\frac{P(X)Q(X) - P(Y)Q(Y))}{(X-Y)P(Y)Q(Y)}\Big] + \sum_{j=1}^{p} \mathrm{Res}_{Y=\lambda_j}\Big[\frac{P(X)Q(X) - P(Y)Q(Y))}{(X-Y)P(Y)Q(Y)}\Big] \equiv 1\,.$$

[6]C'est à David Hilbert que l'on doit dans ce cas le résultat impliquant l'existence de tels polynômes $A_1, ..., A_m$.

3.5.2 Différences divisées et méthode récursive d'Aitken

Utiliser la formule (3.2) n'est pas non plus la meilleure stratégie pour calculer le polynôme d'interpolation de Lagrange (comme d'ailleurs le fait de tenter de résoudre le système (3.1), ce à cause des probèmes de conditionnement mentionnés auparavant). On préfère utiliser la méthode (due à Newton) des *différences divisées* consistant à chercher à exprimer $P(\cdot\,;Y)$ non dans la base usuelle

$$\{1, X, ..., X^N\}$$

des $\mathbb{R}$ (*resp.* $\mathbb{C}$)-polynômes de degré au plus N à coefficients réels (*resp.* complexes), mais à l'exprimer dans la base

$$\{1\,,\,(X-x_0)\,,\,(X-x_0)(X-x_1)\,,\,...\,,\,(X-x_0)\cdots(X-x_{N-1})\}$$

(on peut prendre N points parmi les $N+1$ points x_k proposés et choisir un ordre arbitraire). Le polynôme $P_N(\cdot\,;Y)$ s'écrit

$$P_N(X\,;Y) = u_0 + u_1(X-x_0) + \cdots + u_N(X-x_0)\cdots(X-x_{N-1})\,.$$

En spécifiant les valeurs $x_0, x_1, ..., x_N$, on trouve :

$$\begin{array}{rcl} y_0 & = & u_0 \\ y_1 & = & u_0 + u_1(x_1-x_0) \\ y_2 & = & u_0 + u_1(x_2-x_0) + u_2(x_2-x_0)(x_2-x_1) \\ \vdots & & \vdots \\ y_N & = & u_0 + u_1(x_N-x_0) + \cdots + u_N(x_N-x_0)\cdots(x_N-x_{N-1})\,. \end{array}$$

Ce système (en les inconnues $u_0, ..., u_N$) se résoud « en cascade » de proche en proche très facilement :

$$\begin{array}{rcl} u_0 & = & y_0 \\ u_1 & = & \dfrac{y_1-u_0}{x_1-x_0} = \dfrac{y_1-y_0}{x_1-x_0} \\ u_2 & = & \dfrac{(y_2-u_0)-u_1(x_2-x_0)}{(x_2-x_0)(x_2-x_1)} \\ & = & \dfrac{\frac{y_2-y_0}{x_2-x_0} - \frac{y_1-y_0}{x_1-x_0}}{x_2-x_1} \\ u_3 & = & \ldots \ \textit{etc.} \end{array}$$

Il s'agit d'un système linéaire dont la matrice est triangulaire supérieure. On présentera ici cette démarche sous forme d'un tableau présentant certaines analogies de structure avec le triangle de Pascal. Nous aurons besoin pour cela du lemme suivant, dû à A. Aitken [7] :

Lemme 3.1 (Lemme d'Aitken) *Soient $x_0, ..., x_N$ $N+1$ nombres réels distincts et $y_0, ..., y_N$ $N+1$ nombres réels. Si Q désigne le polynôme d'interpolation de Lagrange*

[7] Mathématicien néo-zélandais, Alexander Craig Aitken, 1895-1967, est aussi connu comme un calculateur mental « prodige » ; le lemme cité ici lui est certainement bien antérieur, mais non sans relation avec les méthodes d'accélération de convergence qu'il développa.

interpolant les valeurs $y_0, ..., y_{N-1}$ respectivement aux points $x_0, ..., x_{N-1}$, R le polynôme d'interpolation de Lagrange interpolant $y_1, ..., y_N$ respectivement aux points $x_1, ..., x_N$, on a, si P désigne le polynôme d'interpolation de f aux points $x_0, ..., x_N$, la formule d'Aitken :

$$P(X) = \frac{(X - x_0)R(X) - (X - x_N)Q(X)}{x_N - x_0}\,.$$

Preuve. On note

$$\widetilde{P}(X) = \frac{(X - x_0)R(X) - (X - x_N)Q(X)}{x_N - x_0}\,.$$

Il suffit de remarquer que, pour $k = 1, ..., N-1$,

$$\begin{aligned}\widetilde{P}(x_k) &= \frac{(x_k - x_0)R(x_k) - (x_k - x_N)Q(x_k)}{x_N - x_0} \\ &= \frac{(x_k - x_0)y_k - (x_k - x_N)y_k}{x_N - x_0} = y_k\end{aligned}$$

(puisqu'à la fois Q et R interpolent les valeurs y_k aux points $x_1, ..., x_N$) et que d'autre part

$$\begin{aligned}\widetilde{P}(x_0) &= \frac{-(x_0 - x_N)Q(x_0)}{x_N - x_0} = Q(x_0) = y_0 \\ \widetilde{P}(x_N) &= \frac{(x_N - x_0)R(x_N)}{x_N - x_0} = R(x_N) = y_N\end{aligned}$$

puisque Q interpole y_0 au point x_0 et que R interpole y_N au point x_N. Comme le degré de $\widetilde{P}$ est exactement égal à N (puisque Q et R sont deux polynômes unitaires de degré exactement $N-1$), $\widetilde{P}$ coïncide bien avec le polynôme d'interpolation de Lagrange aux points $x_0, ..., x_N$. $\diamondsuit$

L'algorithme récursif déduit du lemme d'Aitken. Ce lemme établi, voici un procédé algorithmique permettant de conduire de manière récursive le calcul du polynôme d'interpolation d'un nombre fini de valeurs en un nombre fini de points. Soient $x_0, ..., x_N$ $N+1$ nombres réels distincts et $y_0, ..., y_N$ $N+1$ nombres réels. On introduit la collection de polynômes $P_{k,j}$, $j = 0, ..., N$, $k = j, ..., N$ définis comme suit :

- $P_{k,0} \equiv y_k$, $k = 0, ..., N$;
- $P_{k,j}$, $k = j, ..., N$, est le polynôme d'interpolation de Lagrange interpolant les valeurs $y_{k-j}, ..., y_k$ respectivement aux points $x_{k-j}, ..., x_k$.

La relation de récurrence permettant de calculer $P_{k,j}$ à partir de $P_{k,j-1}$ et $P_{k-1,j-1}$ lorsque $j \geq 1$ et $k \geq j$ s'obtient aisément à partir du lemme 3.1 : le polynôme $P_{k,j-1}$ interpole les valeurs $y_{k-j+1}, y_{k-j}, ..., y_{k-1}, y_k$ aux points

$$x_{k-j+1}, ..., x_{k-1}, x_k$$

(c'est le polynôme constant y_k si $j = 1$), tandis que le polynôme $P_{k-1,j-1}$ interpole les valeurs $y_{k-j}, ..., y_{k-1}$ aux points $x_{k-j}, ..., x_{k-1}$ (c'est le polynôme constant y_{k-1} si $j = 1$) ; le rôle de Q dans le lemme d'Aitken est tenu par $P_{k,j-1}$, les points $x_0, ..., x_{N-1}$ étant maintenant les points $x_{k-j+1}, ..., x_{k-1}$; le rôle de R est tenu par $P_{k-1,j-1}$; le

polynôme $P_{k,j}$ qui interpole les valeurs $y_{k-j}, ..., y_k$ aux points $x_{k-j}, ..., x_k$ est donc donné, si l'on utilise le lemme, par

$$P_{k,j}(X) = \frac{(x_{k-j} - X)P_{k,j-1}(X) - (x_k - X)P_{k-1,j-1}(X)}{x_{k-j} - x_k}\,,\ 1 \leq j \leq k\,.$$

Le calcul des $P_{k,j}$ peut ainsi être présenté sous la forme du tableau triangulaire suivant :

$$\begin{array}{llllllllll}
x_0 & P_{0,0} = y_0 & & & & & & & & \\
x_1 & P_{1,0} = y_1 & P_{1,1} & & & & & & & \\
x_2 & P_{2,0} = y_2 & P_{2,1} & P_{2,2} & & & & & & \\
\vdots & \vdots & \vdots & \vdots & & & & & & \\
x_k & P_{k,0} = y_k & P_{k,1} & P_{k,2} & \dots & P_{k,j} & \dots & P_{k,k} & & \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \\
x_N & P_{N,0} = y_N & P_{N,1} & P_{N,2} & \dots & P_{N,j} & \dots & P_{N,k} & \dots & P_{N,N}
\end{array} \tag{3.3}$$

La première colonne de cette matrice figure les polynômes $P_{k,0} \equiv y_k$. La démarche algorithmique consiste à calculer les colonnes de gauche à droite en utilisant la formule inductive (3.3) permettant de calculer l'entrée $P_{k,j}$ à partir des deux entrées $P_{k-1,j-1}$ et $P_{k,j-1}$ de la colonne précédente. Le polynôme cherché $P_N = P_{N,N}$ est l'entrée figurant dans le coin inférieur droit de ce tableau triangulaire (dit *tableau d'Aitken*).

La procédure récursive ainsi décrite se schématise ainsi :

```
function F=proc(N,f,a,b)
if N>1
   F = [(X-a)*proc(N-1,f, a+(b-a)/N,b)
                        -(X-b)*proc(N-1,f,a,b-(b-a)/N)]/(b-a)
if N=1
   F = [(X-a)*f(a) - (X-b)*f(b)]/(b-a)
```

lorsqu'il s'agit de retourner, N, f, a, b étant donnés, le polynôme d'interpolation $P_N[f]$ de f aux $N+1$ points régulièrement espacés

$$x_k := a + k\,\frac{b-a}{N}\,, \quad k = 0, ..., N$$

d'un segment $[a, b]$ sur lequel est définie la fonction f.

3.5.3 L'interpolation de Lagrange à l'épreuve des « effets de bord »

Reprenons maintenant la suite du cours, avec notre exemple de la fonction

$$f\ : x \in [-8, 8] \longmapsto \frac{1}{1+x^2}$$

échantillonnée aux 21 points x_k régulièment espacés entre $x_0 = -8$ et $x_{20} = 8$. Calculons (sous MATLAB) le polynôme d'interpolation de Lagrange correspondant aux données

$$\{x_0, y_0\}\,,\ \{x_1, y_1\}\,,\ ...\,,\ \{x_N, y_N\}$$

(où $y_k = f(x_k)$, $k = 0, ..., N$). Le calcul effectué sous MATLAB pour trouver ce polynôme correspond de fait à la recherche du polynôme p de degré N tel que l'*erreur quadratique* :

$$\sum_{j=0}^{N} |p(x_j) - y_j|^2$$

soit minimale (dans ce cas, ce minimum est atteint précisément par le polynôme $P_N(\cdot\,; Y)$ et vaut 0). On reviendra ultérieurement sur ce procédé (méthode d'approximation au sens des moindres carrés) dans une section ultérieure de ce chapitre ; on cherchera alors, n étant fixé entre 0 et N, le « meilleur » polynôme p_n de degré n, au sens suivant : celui qui rend minimale l'erreur quadratique

$$\sum_{j=0}^{N} |p_n(x_j) - y_j|^2 \,,$$

les données $\{x_k, y_k\}$, $k = 0, ..., N$, ayant été fixées. Notre méthode de calcul (dans le cas $n = N$) du polynôme d'interpolation P_N est basée sur la résolution du système (3.1), mais auparavant, pour éviter l'écueil du mauvais conditionnement, on normalise les x_k en les divisant par l'écart type σ de la distribution des x_k.

```
>> sigma=std(x);
>> xx=x/sigma;
>> P=polyfit(xx,y,20);
```

Pour afficher le graphe[8] du polynôme $P_N(\cdot\,; Y)$ sur $[-8, 8]$, il faut donc prescrire les commandes

```
>> xxx = -8:.0001:8 ;
>> yyy = polyval (P, xxx/sigma) ;
```

Voici sur la figure ci-dessous le graphe obtenu :

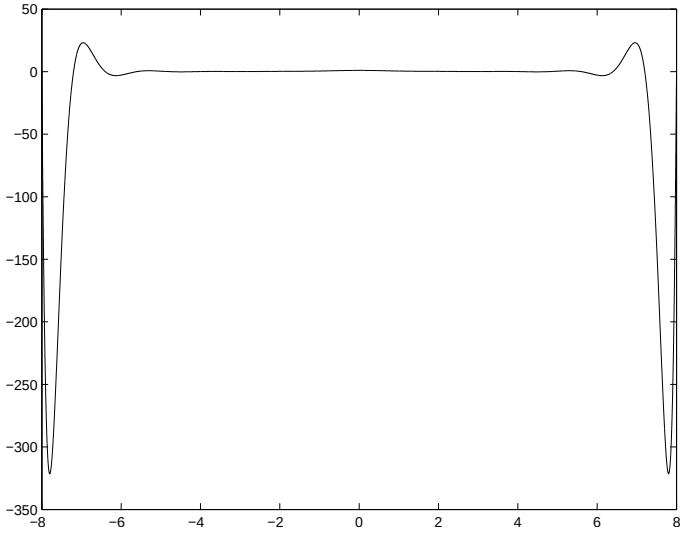

FIG. 3.2 – Les « effets de bord » de l'interpolation de Lagrange

[8] Avec un pas d'échantillonnage de l'axe des abscisses égal par exemple à 10^{-4}.

On constate que le graphe de $P_N(\cdot; Y)$ présente des oscillations amplifiées au niveau des extrémités de l'intervalle $[-8, 8]$, phénomène que l'on tentera d'expliquer dans la section à venir[9], lorsque nous analyserons le contrôle d'erreur dans le mécanisme d'interpolation de Lagrange. Ces phénomènes sont réellement des « effets de bord[10] » car si l'on trace le graphe de $x \longmapsto P_N(x\,;Y)$ sur $[-5, 5]$ (avec toujours le pas d'échantillonnage égal à 10^{-4}), on constate que ce graphe approche celui de la fonction

$$f\ : x \longmapsto \frac{1}{1+x^2}$$

sur lequel se trouvaient les points « marqués » (x_k, y_k) à partir desquels à été menée l'interpolation de Lagrange. On a figuré ce graphe en pointillés sur la figure et en plein le graphe de $x \longmapsto P_N(\cdot\,;Y)$ et marqué les points (x_k, y_k) par des croix. On constate déjà sur $[-5, 5]$ l'apparition des oscillations appelées à s'amplifier à l'approche des extrémités -8 et 8 pour le graphe de la fonction polynômiale

$$x \longmapsto P_N(x\,;Y)\,.$$

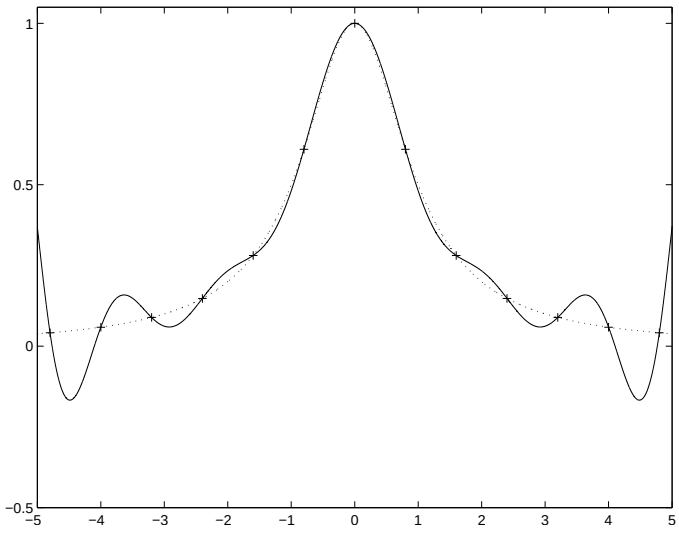

FIG. 3.3 – Approximation de f par $x \longmapsto P_N(x\,;Y)$ sur $[-5, 5]$

Concernant l'erreur commise lorsque l'on « approche » sur un intervalle $[a, b]$ une fonction f par le polynôme de Lagrange P assignant, $x_0, ..., x_N$ étant $N+1$ points distincts de $[a, b]$, à x_k le nombre $f(x_k)$, elle peut être estimée lorsque f est assez régulière. On a en effet la

Proposition 3.2 *Supposons que f soit de classe C^{N+1} sur $[a, b]$, que $x_0, ..., x_N$ soient $N+1$ points distincts de $[a, b]$ et que P soit le polynôme d'interpolation*

[9]Il ne faut pas oublier qu'un polynôme de degré N a une dérivée $(N+1)$-ème identiquement nulle ; essayez de calculer par récurrence la dérivée $N+1$-ème de la fonction f pour comprendre pourquoi penser « approcher » f par $P_N(\cdot\,;Y)$ lorsque l'on s'éloigne de l'origine était désespéré ! On y reviendra dans la section suivante.

[10]La raison est que les x_k sont espacés d'un pas fixe, pas qui ne se « resserre » nullement lorsque l'on se rapproche du bord.

de Lagrange (de degré N) tel que $P(x_k) = f(x_k)$, $k = 0, ..., N$. Pour tout $x \in [a, b]$, il existe $\xi_x \in]a, b[$ tel que

$$f(x) - P(x) = \frac{(x - x_0) \cdots (x - x_N)}{(N + 1)!} f^{(N+1)}(\xi_x) \, .$$

Preuve. On suppose que x n'est pas l'un des points $x_0, ..., x_N$ (sinon, n'importe quel point ξ de $[a, b]$ fait l'affaire car les deux membres sont nuls). On pose ensuite

$$K(x) = \frac{f(x) - P(x)}{(x - x_0) \cdots (x - x_N)}$$

et on considère la fonction

$$F \ : t \in [a, b] \longmapsto f(t) - P(t) - K(x)(t - x_0) \cdots (t - x_N) \, .$$

Cette fonction s'annule aux $N+2$ points (distincts) $x_0, ..., x_N, x$. Grâce au théorème de Rolle, sa dérivée F' s'annule en $N + 1$ points de $]a, b[$; on continue ainsi, F'' s'annule en N points,..., finalement $F^{(N+1)}$ s'annule en un point $\xi_x \in]a, b[$. Mais on voit, comme P est de degré N, que

$$f^{(N+1)}(\xi_x) = (N + 1)! K(x) \, ,$$

d'où le résultat en remplacant $K(x)$ par cette expression. $\diamondsuit$

3.6 Quelques rudiments autour de l'élimination

Si P et Q sont deux polynômes non nuls de degrés respectifs $p > 0$ et $q > 0$ à coefficients dans un corps $\mathbb{K}$, dire que P et Q sont premiers entre eux dans $\mathbb{K}[X]$, *i.e* ont un PGCD égal à 1 (ce que l'on teste en conduisant dans $\mathbb{K}[X]$ l'algorithme d'Euclide) équivaut à affirmer qu'il existe des polynômes A, B, de degrés respectifs $q - 1$ et $p - 1$, tels que

$$A(X)P(X) + B(X)Q(X) \equiv 1 \, .$$

On constate d'ailleurs puisque tout autre solution $(\widetilde{A}, \widetilde{B})$ de $\widetilde{A}P + \widetilde{B}Q \equiv 1$ s'écrit alors (à cause du lemme de Gauss[11])

$$(\widetilde{A}, \widetilde{B}) = (A, B) + H \times (Q, -P) \, ,$$

où H est un polynôme arbitraire dans $\mathbb{K}[X]$, que la solution (A, B) avec ces degrés $(q - 1, p - 1)$ précisés est unique.

On conclut donc de ce qui précède que le fait que P et Q soient premiers entre eux dans $\mathbb{K}[X]$ équivaut à ce que le système de $p + q$ équations en $p + q$ indéderminées (les coefficients des polynômes A et B que l'on cherche) obtenu en écrivant l'égalité polynômiale

$$\Big(\sum_{j=0}^{q-1} u_j X^j \Big) P(X) + \Big(\sum_{j=0}^{p-1} v_j X^j \Big) Q(X) \equiv 1 \tag{3.4}$$

[11] Voir le cours de MHT201 : si P et Q sont premiers entre eux dans $\mathbb{K}[X]$ et si P divise un produit du type $B(X) \times Q(X)$, alors P divise B.

(la différence est un polynôme de degré $p+q-1$, il y a donc $p+q$ équations affines en les inconnues à écrire) est un système de Cramer.

En résumé, si P et Q sont de degrés respectifs exactement $p>0$ et $q>0$, dire que P et Q sont premiers entre eux dans $\mathbb{K}[X]$ équivaut à dire que le déterminant de ce système (que l'on appelle *résultant de Sylvester*[12]) est non nul. On peut écrire explicitement ce déterminant $R_{p,q}(P,Q)$ si

$$\begin{aligned} P(X) &= a_0X^p+\cdots+a_p,\ a_0\neq 0\\ Q(X) &= b_0X^q+\cdots+b_q,\ b_0\neq 0 \end{aligned}$$

et l'on obtient :

$$R_{p,q}[P,Q]=\begin{vmatrix} a_0 & a_1 & \dots & a_p & 0 & \dots & 0 & 0\\ 0 & a_0 & \dots & a_{p-1} & a_p & \dots & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \dots & \dots & \dots & \dots & a_p & 0\\ 0 & 0 & \dots & \dots & \dots & \dots & a_{p-1} & a_p\\ b_0 & b_1 & \cdots & \cdots & \cdots & \cdots & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & b_{q-1} & b_q \end{vmatrix} \tag{3.5}$$

Les formules de Cramer (utilisées pour chercher les inconnues $u_0,...,u_{q-1},v_0,...,v_{p-1}$ du système de Cramer correspondant à la formule (3.4) traduite sur les coefficients des puissances de X : $1,X,...,X^{p+q-1}$) nous montrent d'ailleurs que $R_{p,q}[P,Q]$ s'exprime sous la forme

$$A_{p,q}(a_0,...,a_p,b_0,...,b_q;X)\,P(X)+B_{p,q}(a_0,...,a_p,b_0,...,b_q;X)\,Q(X),$$

où les expressions $A_{p,q}$ et $B_{p,q}$ sont à coefficients des expressions polynomiales (ne dépendant que de p et q) à coefficients entiers en les « entrées » $a_0,...,a_p,b_0,...,b_q$ correspondant aux coefficients des polynômes P et Q.

Cette dernière remarque a son importance : si P et Q sont par exemple des polynômes en deux variables (X,Y) à coefficients dans $\mathbb{K}$:

$$\begin{aligned} P(X,Y) &:= a_0(Y)X^p+\cdots+a_p(Y)\\ Q(X,Y) &:= b_0(Y)X^q+\cdots+b_q(Y) \end{aligned}$$

et que l'on forme le résultant de Sylvester $R_{p,q}[P,Q](Y)$ de ces deux polynômes considérés comme polynômes en la variable X, on obtient un polynôme $R(Y)$ tel que

$$\Big(P(X,Y)=0 \text{ et } Q(X,Y)=0\Big)\Longrightarrow R(Y)=0\,,$$

ce qui permet (à condition toutefois que le polynôme $R_{p,q}[P,Q](Y)$ que l'on construit ne soit pas identiquement nul) d'« éliminer » la variable X dans le système de deux équations à deux inconnues (mais non linéaire !)

$$P(X,Y)=Q(X,Y)=0\,.$$

Sous MAPLE12, le calcul de résultant se fait sous une routine du type

[12] James Sylvester (1814-1897), avocat, musicologue et mathématicien anglais, ami de Cayley (vous connaissez depuis le cours de MHT301 le célèbre théorème de Cayley-Hamilton) ; Sylvester fut l'un des pionniers de la théorie des déterminants ; c'est à lui d'ailleurs que l'on doit le qualificatif « matrice ».

```
> resultant (P(x), Q(x), x)
> resultant (f(x,y), g(x,y),x)
```

Ce procédé se généralise : si par exemple $F_1, ..., F_M$ sont une liste d'expressions polynomiales en n symboles $X_1, ..., X_n$, alors, en formant le résultant de Sylvester de F_1 et $F_1 + u_2F_2 + ... + u_MF_M$ (où $u_2, ..., u_M$ sont des paramètres), tous les deux considérés comme des polynômes en X_1 (on suppose que X_1 figure explicitement dans F_1), on obtient une expression polynômiale en $u_2, ..., u_M$ et $X_2, ..., X_n$; la liste des coefficients $G_1(X_2, ...X_n), ..., G_{M'}(X_2, ..., X_n)$ de cette expression (considérée comme polynôme en $u_2, ..., u_M$) fournit une liste d'expressions polynomiales en $X_2, ..., X_n$, toutes s'exprimant sous la forme

$$G_j(X_2, ..., X_n) = \sum_{k=1}^{M} A_{j,k}(X_1, ..., X_n)F_k(X_1, ..., X_n)\,,\ j = 1, ..., M'\,.$$

Si l'on s'intéresse à résoudre le système d'équations

$$F_1(X_1, ..., X_n) = \cdots = F_M(X_1, ..., X_n) = 0\,,$$

le système

$$G_1(X_2, ..., X_n) = \cdots = G_{M'}(X_2, ..., X_n) = 0$$

fournit un nouveau système où la variable X_1 a été « éliminée ». On peut ainsi continuer et c'est là la pierre d'angle de la *théorie de l'élimination* que l'on a fait qu'esquisser ici [13].

Si $\mathbb{K} = \mathbb{C}$, dire que $R_{p,q}[P, Q] = 0$ équivaut à dire que P et Q (de degrés exactement $p > 0$ et $q > 0$) ont une racine commune dans $\mathbb{C}$.

Si P est un polynôme de $\mathbb{C}[X]$ de degré exactement $p > 1$, le résultant $R_{p,p-1}[P, P']$ de P et P' est un déterminant de taille $2\deg P - 1$ tel que $R_{p,p-1}[P, P'] = 0$ équivaut (toujours si $a_0 \neq 0$) au fait que le polynôme

$$a_0X^p + \cdots + a_p = 0$$

ait une racine double. On pourra par exemple calculer le résultant $R[P, P']$ si

$$P(X) = aX^2 + bX + c$$

(en fonction des paramètres a, b, c), ce qui donne

$$R_{2,1}(P, P') = \begin{vmatrix} a & b & c \\ 2a & b & 0 \\ 0 & 2a & b \end{vmatrix} = -ab^2 + 4a^2c = -a(b^2 - 4ac)\ ;$$

si

$$P(X) = X^3 + \beta X + \gamma\,,$$

[13] Il faut toutefois souligner que ce procédé devient très coûteux en temps et espace de calcul dès que le nombre de variables $X_1, ..., X_n$ augmente ! D'autres méthodes, inspirées de l'algorithme d'Euclide (la plus importante est celle qui passe par la construction de bases de Gröbner), ont été introduites depuis les années 1970 (à l'aube de la révolution numérique) pour gérer à moindre coût les systèmes d'équations en beaucoup de paramètres que fait par exemple surgir la robotique.

(autre cas important), on trouve

$$R_{3,2}(P,P') = \begin{vmatrix} 1 & 0 & \beta & \gamma & 0 \\ 0 & 1 & 0 & \beta & \gamma \\ 3 & 0 & \beta & 0 & 0 \\ 0 & 3 & 0 & \beta & 0 \\ 0 & 0 & 3 & 0 & \beta \end{vmatrix} = 4\beta^3 + 27\gamma^2 \,.$$

Le résultant $R_{d,d-1}(P,P')$ de P et P' (le degré de P étant précisé) est un déterminant $(2d-1)\times(2d-1)$ que l'on appelle le *discriminant* du polynôme P ; il s'exprime polynomialement en les coefficients de P et est nul si et seulement P a une racine multiple dans $\mathbb{C}$ (attention, seulement si le coefficient dominant a_0 de P doit être non nul, comme en témoigne l'exemple du polynôme $P(X) = aX^2 + bX + c$ dont le discriminant n'est pas $b^2 - 4ac$ mais $-a(b^2 - 4ac)$!).

3.7 Interpolation et calcul numérique d'intégrales

Soit $[a,b]$ un segment de $\mathbb{R}$ et $f : [a,b] \longmapsto \mathbb{C}$ une fonction continue. À toute subdivision

$$x_0 = a < x_1 < \ldots < x_{N-1} < x_N = b \,,$$

on peut associer une formule de calcul approché de l'intégrale

$$\int_a^b f(t)\,dt$$

en remplaçant cette intégrale par

$$\int_a^b \left(\sum_{k=0}^{N} f(x_k) \frac{\prod_{j\neq k}(t - x_j)}{\prod_{j\neq k}(x_k - x_j)} \right) dt \,, \tag{3.6}$$

où

$$\sum_{k=0}^{N} f(x_k) \frac{\prod_{j\neq k}(t - x_j)}{\prod_{j\neq k}(x_k - x_j)}$$

représente ici le polynôme d'interpolation de Lagrange de f, interpolant les valeurs $f(x_k)$, $k = 0, ..., N$, respectivement aux points $x_0, ..., x_N$; l'erreur entre la valeur exacte $\int_a^b f(t)\,dt$ et la valeur approchée (3.6) est nulle lorsque f est une fonction polynômiale de degré au plus N. La subdivision

$$x_0 = a < x_1 < \ldots < x_{N-1} < x_N = b$$

étant donnée, on obtient donc en résolvant le système de $N+1$ équations en $N+1$ inconnues $u_0, ..., u_N$

$$\int_a^b t^j\,dt = \sum_{k=0}^{N} u_k x_k^j \,,\ j = 0, ..., N \,, \tag{3.7}$$

(qui est un système de Cramer car le déterminant du système est un déterminant de Vandermonde non nul), le calcul des coefficients $u_0, ..., u_N$ impliqués dans la formule approchée

$$\int_a^b f(t)\,dt \simeq \sum_{k=0}^{N} u_k f(x_k)$$

de manière à ce que cette formule soit exacte pour les fonctions polynômiales de degré au plus N(c'est-à-dire pour $f(t) \equiv 1$,..., $f(t) \equiv t^N$).

Dans le cas où, pour $N \in \mathbb{N}^*$ fixé, on part de la subdivision à pas constant

$$x_{N,0} = a < x_{N,1} = a + \frac{b-a}{N} < ... < x_{N,N} = a + N\,\frac{b-a}{N} = b\,,$$

on obtient les *formules de quadrature* attribuées à Isaac Newton et Roger Cotes [14] :

$$\int_a^b f(t)\,dt \simeq \sum_{k=0}^{N} u_{N,k}\, f(x_{N,k})\,.$$

Les premiers cas sont les cas $N = 1$ (formule des *trapèzes*), $N = 2$ (formule de *Simpson*[15]), $N = 3$:

$$\begin{aligned}
\int_a^b f(t)\,dt &\simeq (b-a)\,\frac{f(a)+f(b)}{2} \qquad (N=1\,;\,2 \text{ points})\\
\int_a^b f(t)\,dt &\simeq (b-a)\,\Big(\frac{f(a)}{6} + \frac{4}{6} f\Big(\frac{a+b}{2}\Big) + \frac{f(b)}{6}\Big) \qquad (N=2\,;\,3 \text{ points})\\
\int_a^b f(t)\,dt &\simeq (b-a)\,\Big(\frac{f(a)}{8} + \frac{3}{8}\Big[f\Big(a+\frac{b-a}{3}\Big) + f\Big(a+\frac{2(b-a)}{3}\Big)\Big] + \frac{f(b)}{8}\Big)\\
&\qquad (N=3\,;\,4 \text{ points})
\end{aligned}$$

On rappelle (voir la proposition 3.2) que, si f est de classe C^{N+1} sur $[a,b]$, l'erreur $e_N(t)$ entre $f(t)$ et sont polynôme d'interpolation de Lagrange P_N en $N+1$ points disctincts $x_0, ..., x_N$ du segment $[a,b]$ s'écrit

$$e_N(t) = f(t) - P_N(t) = \frac{(t-x_0)...(t-x_N)}{(N+1)!}\, f^{(n+1)}(\xi_t)\,,$$

où ξ_t est un point de $]a,b[$. En particulier, si $N = 1$, $x_0 = a$, $x_1 = b$, on a

$$e_1(t) = \frac{(t-a)(t-b)}{2}\, f''(\xi_t)\,.$$

L'erreur commise dans la formule des trapèzes est donc contrôlée en

$$\Big|\int_a^b f(t)\,dt - \frac{f(a)+f(b)}{2}\Big| \le \sup_{[a,b]} |f''| \times \int_a^b (t-a)(b-t)\,dt = \sup_{[a,b]} |f''| \times \frac{(b-a)^3}{12}\,,$$

ce que l'on interprète en disant que *l'ordre de la formule des trapèzes est égal à* 3.

[14] Ces formules sont apparues à l'occasion du travail de relecture par le mathématicien anglais Roger Cotes (1682-1716) des *Principia* d'Isaac Newton.

[15] Ainsi dénommée en référence au mathématicien et astrologue britannique Thomas Simpson (1710-1761) ; de fait, elle avait été déjà introduite par Johannes Kepler deux siècles auparavant.

Pour la formule de Simpson ($N = 2$), on pose $h = (b-a)/2$ et $\alpha = (a+b)/2$ et, pour des raisons évidentes de symétrie, on écrit l'erreur

$$\int_a^b f(t)\,dt - (b-a)\Big(\frac{f(a)}{6} + \frac{4}{6}f\Big(\frac{a+b}{2}\Big) + \frac{f(b)}{6}\Big) = \int_{a-h}^{a+h} f(t)\,dt - \frac{h}{3}[f(\alpha-h) + 4f(\alpha) + f(\alpha+h)]\,. \tag{3.8}$$

Si l'on pose

$$\varphi(u) := \int_{\alpha-u}^{\alpha+u} f(t)\,dt - \frac{u}{3}[f(\alpha-u) + 4f(\alpha) + f(\alpha+u)]$$

et que f est supposée de classe C^5 sur $[a,b]$, on vérifie que la première dérivée non nulle de φ en $u=0$ est la dérivée d'ordre 5, qui vaut à $\varphi^{(5)}(0) = -(4/3)f^{(4)}(\alpha)$; en effet

$$\begin{aligned}
\varphi'(u) &= \frac{2}{3}(f(\alpha+u) + f(\alpha-u)) - \frac{u}{3}[f'(\alpha+u) - f'(\alpha-u)] - \frac{4f(\alpha)}{3}\\
\varphi''(u) &= \frac{1}{3}(f'(\alpha+u) - f'(\alpha-u)) - \frac{u}{3}[f''(\alpha+u) + f''(\alpha-u)]\\
\varphi'''(u) &= -\frac{u}{3}[f'''(\alpha+u) - f'''(\alpha-u)]\\
\varphi^{(4)}(u) &= -\frac{1}{3}[f'''(\alpha+u) - f'''(\alpha-u)] - \frac{u}{3}[f^{(4)}(\alpha-u) + f^{(4)}(\alpha+u)]\\
\varphi^{(5)}(u) &= -\frac{2(f^{(4)}(\alpha+u) + f^{(4)}(\alpha-u))}{3} - \frac{u}{3}[f^{(5)}(\alpha+u) - f^{(5)}(\alpha-u)]\,.
\end{aligned}$$

Grâce à la formule de Taylor-Young, on peut donc écrire, au voisinage de $u=0$,

$$\varphi(u) = -\frac{h^5}{90}\,f^{(4)}(\alpha) + o(h^5)\,,$$

ce qui prouve que *la formule de Simpson est d'ordre 5* (le contrôle de cette erreur étant en h^5 si $h = (b-a)/2$ est suffisamment petit).

Dans le cas général, l'ordre de l'erreur est déterminé par l'ordre de la première dérivée non nulle de φ en 0. Si $h = \frac{b-a}{N}$, cette erreur est ainsi en h^{N+3} si N est pair (il y a donc un nombre impair de points x_k et possibilité d'utiliser un point « médian » comme α dans la méthode des trapèzes), en h^{N+2} si N est impair (pas de point « médian » car il y a un nombre pair de points x_k dans ce cas). Utiliser un nombre N pair est donc préférable ; cependant, plus N augmente, plus la méthode devient instable car les coefficients $u_{N,0}, ..., u_{N,N}$ dans

$$\int_a^b f(t)\,dt \simeq \sum_{k=0}^{N} u_{N,k}\,f(x_{N,k})$$

« explosent » avec N, pour la même raison en fait que celle impliquant les effets de bord dans l'interpolation de Lagrange sous-jacente aux méthodes de Newton-Cotes : on ne peut approcher sans risque une fonction dont les dérivées successives s'amplifie près des extrémités de $[a,b]$ par des fonctions polynômiales dont les dérivées d'ordre assez grand sont automatiquement indentiquement nulles !

L'idée pratique pour pallier à ces difficultés est d'utiliser des *méthodes composites*. On découpe $[a,b]$ en N segments égaux $[a_k,b_k]$ (donc de longueur $h=(b-a)/N$) et, sur chacun de ces segments, on utilise une méthode de Newton-Cotes d'ordre p. L'ordre de la méthode composite est alors $p-1$ car il faut multiplier h^p (contrôlant l'erreur sur chaque segment) par $N=(b-a)/h$, ce qui donne une erreur en h^{p-1}.

3.8 L'interpolation par des polynômes trigonométriques et la FFT

3.8.1 Un problème d'interpolation

Interpoler une fonction f sur un intervalle $[a,b]$ de $\mathbb{R}$ par des polynômes de degré N fixé (comme nous permet de le faire par exemple l'interpolation de Lagrange) aux fins de travailler ensuite avec cette fonction (par exemple de l'intégrer de manière approchée sur $[a,b]$), c'est chercher à trouver LE polynôme de degré N dont l'évaluation coïncide avec celle de f aux points

$$x_0 = a < x_1 \cdots < x_N = b$$

d'une subdivision de $[a,b]$.

Mais il se peut fort bien arriver que la classe des fonctions polynômiales ne soit pas la classe de fonctions la mieux adaptée à la modélisation de f ! Si par exemple, l'objectif est de représenter la restriction à $[0,2\pi]$ d'une fonction 2π-périodique sur $\mathbb{R}$ (ce sont ces fonctions que l'on retrouve en théorie des télécommunications ou, en deux dimensions, en imagerie), il est plus naturel, si l'on a en tête la théorie la décomposition en harmoniques fondamentales $\theta \longmapsto e^{ik\theta}$, $k \in \mathbb{Z}$, de toute fonction périodique (formalisée au XIX-ème siècle sous l'impulsion de l'optique, de l'électromagnétisme ou de la mécanique ondulatoire tant par des physiciens comme James Clerk Maxwell (1831-1879) que des mathématiciens tels, bien avant, Jean Baptiste Fourier, 1768-1830), pour interpoler f aux N valeurs $\theta = 0,...,\theta = 2(N-1)\pi/N$, de chercher à réaliser cette interpolation avec le polynôme trigonométrique

$$\theta \in \mathbb{R} \longmapsto \sum_{k=0}^{N-1} a_k e^{ik\theta}\,.$$

Notons que cela n'a pas de sens d'envisager des fréquences $k \geq N$ car un maillage avec un pas de $2\pi/N$ ne saurait suffire pour rendre compte d'une fonction aussi oscillante que $\theta \longmapsto e^{ik\theta}$, où $k \geq N$, fonction dont la partie réelle s'annule sur un réseau de points de pas $\pi/k < \frac{1}{2}\,(2\pi/N)$ dès que $N \geq 2$.

Chercher $a_0,...,a_N$ revient à résoudre le système de N équations à N inconnues

$$\sum_{k=1}^{n} a_k W_N^{kj} = f(2\pi j/N)\,,\ j = 0,...,N-1\,,$$

système dont la matrice est la matrice hermitienne

$$\left[\overline{W_N}^{jk}\right]_{0\leq j,k\leq N-1}\,,$$

où

$$W_N := \exp(-2i\pi/N)\,.$$

La matrice

$$A_N := \left[W_N^{jk}\right]_{0\leq j,k\leq N-1}$$

est une matrice inversible, car l'on vérifie immédiatement que

$$A_N\,.\,\overline{A_N} = N\,I_N\,,$$

où I_N est la matrice de l'identité ; pour vérifier cela, il suffit de se rappeler que si ζ est un nombre complexe différent de 1 et tel que $\zeta^N = 1$, alors on a

$$1 + \zeta + \cdots + \zeta^{N-1} = 0$$

du fait de l'identité remarquable

$$(X^N - 1) = (X - 1)(1 + X + X^2 + \cdots + X^{N-1})\,.$$

L'opération linéaire de $\mathbb{C}^N$ dans $\mathbb{C}^N$ correspondant à la multiplication par la matrice A_N s'appelle la *transformation de Fourier discrète* d'ordre N[16] ; son inverse (à savoir la multiplication par la matrice $\overline{A_N}/N$) s'appelle *transformation de Fourier discrète inverse* d'ordre N [17]. On a donc

$$\begin{aligned}\begin{pmatrix} a_0 \\ \vdots \\ a_{N-1}\end{pmatrix} &= \frac{1}{N}\,A_N\,.\,\begin{pmatrix} f(0) \\ \vdots \\ f(2\pi(N-1)/N)\end{pmatrix} \\ &= \frac{1}{N}\,\mathrm{DFT}_N\left[\begin{pmatrix} f(0) \\ \vdots \\ f(2\pi(N-1)/N)\end{pmatrix}\right]\,.\end{aligned}$$

La seule DFT_N réelle est celle qui correspond à $N = 2$ et dont la matrice est la matrice dite *matrice papillon*

$$A_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1\end{pmatrix}\,.$$

C'est aux ingénieurs informaticiens américains J.W. Cooley et J.W. Tukey que revient en 1965 l'idée du calcul algorithmique de l'action de la DFT lorsque l'ordre N est une puissance de 2, $N = 2^k$. Le nombre de multiplications nécessité par la multiplication d'un vecteur colonne X de longueur 2^k par la matrice A_{2^k} (ce calcul correspond précisément à l'action de la DFT d'ordre 2^k sur le vecteur colonne X correspondant) se trouve ainsi réduit à $k \times N/2$ au lieu de ce qu'il devait être *à priori*, c'est-à-dire N^2. C'est cette réduction drastique qui rend aujourd'hui possible l'usage intensif de la transformation de Fourier discrète, non seulement en analyse et traitement du signal, mais dans des pans entiers des mathématiques appliquées ou de l'informatique. L'algorithme de Cooley-Tukey marque le début de ce que l'on pourrait qualifier de « révolution numérique ».

[16] En anglais, DFT_N pour *Discrete Fourier Transform* d'ordre N.

[17] Ou encore, dans la terminologie anglo-saxonne IDFT_N pour *Inverse Discrete Fourier Transform* d'ordre N.

L'idée clef de Cooley-Tukey est de remarquer que, si N est une puissance de 2, $N = 2^k$, alors $N/2$ aussi ($N/2 = 2^{k-1}$), et par conséquent

$$W_N^{N/2} = e^{-i\pi} = -1\,.$$

C'est sur la cellule de calcul élémentaire dite *cellule papillon* que s'articule toute l'architecture de l'algorithme de Cooley-Tukey ; il s'agit d'une cellule nécessitant deux additions (et aucune multiplication, hormis la multiplication triviale par 1) et transformant le vecteur (z_1, z_2) en le vecteur

$$\begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix},$$

cellule que l'on peut représenter (c'est de là que vient la terminologie) comme sur la figure 4.4 ci-dessous :

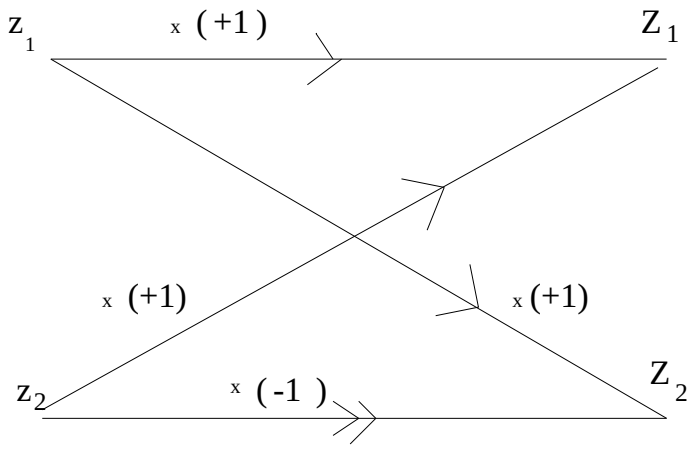

FIG. 3.4 – Cellule papillon

Cette opération correspond d'ailleurs à l'opération de transformée de Fourier discrète lorsque $N = 2$. Exprimons sur un diagramme (figure 4.5 ci-dessous) l'architecture de l'algorithme correspondant à $N = 2^k$ en fonction de celle correspondant à $N = 2^{k-1}$.

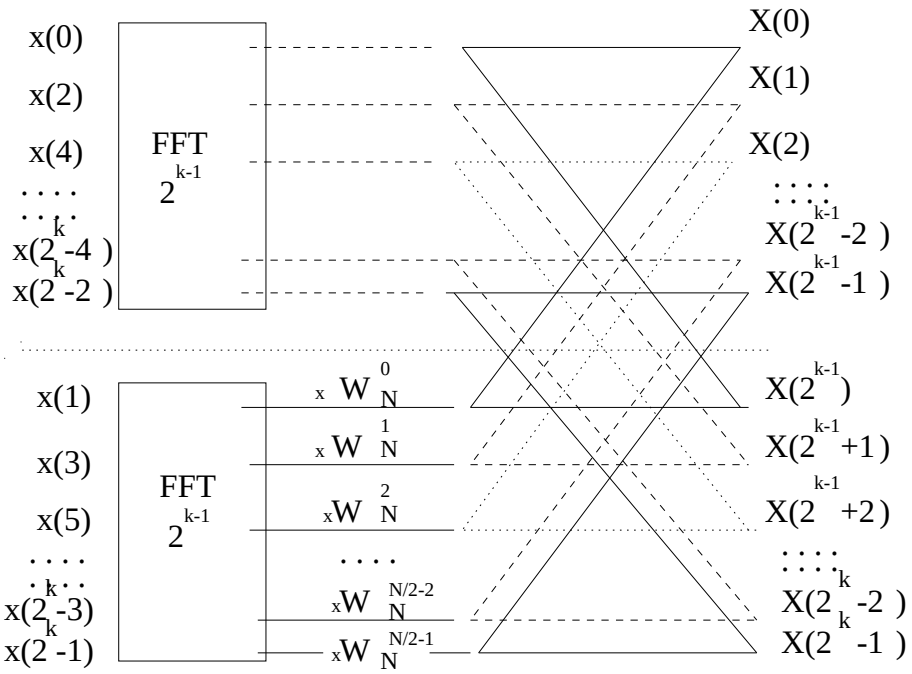

FIG. 3.5 – *Algorithme de Cooley Tukey* $N/2 = 2^{k-1} \rightarrow N = 2^k$

On voit que l'algorithme doit commencer par une phase de ré-agencement des entrées (qui se traduit de fait par un renversement des bits dans l'écriture de l'indice d'entrée en base 2) ; ensuite les mécanismes s'enchaînent comme indiqué sur le diagramme ci-dessus et la consommation au niveau des opérations arithmétiques significatives est, comme on le voit immédiatement, de

$$\frac{N}{2} + 2\frac{N}{4} + 4\frac{N}{8} + \cdots 2^{k-1}\frac{N}{2^k} = \frac{kN}{2}$$

comme annoncé. Cet algorithme est implémenté dans l'environnement **MATLAB** sous l'une ou l'autre des commandes

```
>>X=fft(x);
>>X=fft(x,2^k);
```

(on parle de *Fast Fourier Transform*, d'où la terminologie FFT). Dans le second cas, le signal est soit tronqué, soit complété trivialement par des zéros suivant que sa longueur est supérieure ou inférieure ou égale à 2^k. La transformation inverse est un algorithme exactement du même type, mis à part le fait que W_N est remplacé par $\overline{W}_N$ et qu'il y a une division finale par N. Sous l'environnement **MATLAB**, ce sont l'une ou l'autre des commandes

```
>>X=ifft(x);
>>X=ifft(x,2^k);
```

qui réalisent l'inverse de la transformation de Fourier discrète lorsque N est une puissance de 2. Les versions bidimensionnelles sont données par les commandes

```
>>A=fft2(I);
>>A=fft2(I,2^(k1),2^(k2));
>>I=ifft2(A);
>>I=ifft2(A,2^(k1),2^(k2));
```

Notons que les transformations tronquées ne sont pas inverses l'une de l'autre (les tailles ne sont pas respectées), tandis que l'on a bien toujours

```
>>X=ifft(fft(X));
>>A=ifft2(fft2(A);
```

3.8.2 Utilisation algorithmique de la FFT dans $\mathbb{K}[X]$

L'algorithme de Cooley-Tukey s'avère aussi un outil très précieux dans l'algorithmique des polynômes, pour calculer de manière rapide le produit de deux polynômes. La remarque clef est la suivante : si P et Q sont deux polynômes de degrés M et que $N > 2M$ [18], le produit $P(X)Q(X)$ coïncide avec le reste de ce polynôme dans la division par le polynôme $X^N - 1$. Si

$$\begin{aligned} P(X) &= u_0 + u_1 X + \cdots + u_M X^M \\ Q(X) &= v_0 + v_1 X + \cdots + v_M X^M \,, \end{aligned}$$

le produit PQ s'écrit donc

$$P(X)Q(X) = \sum_{k=0}^{N-1} \widetilde{w}_k X^k$$

où

$$\widetilde{w}_k := \sum_{l=0}^{N} \widetilde{u}_l \widetilde{v}_{N-l} = \sum_{l=0}^{N} \widetilde{u}_{N-l} \widetilde{v}_l \,,$$

$(\widetilde{u}_l)_{l\in\mathbb{Z}}$ (*resp.* $(\widetilde{v}_l)_{l\in\mathbb{Z}}$) étant la version N périodisée de la suite $\widetilde{u}_0 = u_0, ..., \widetilde{u}_M = u_M, \widetilde{u}_k = 0$ si $k = M+1, ..., N-1$ (*resp.* $(\widetilde{v}_l)_{l\in\mathbb{Z}}$) étant la version N périodisée de la suite $\widetilde{v}_0 = u_0, ..., \widetilde{v}_M = u_M, \widetilde{v}_k = 0$ si $k = M+1, ..., N-1$). On remarque immédiatement que le vecteur $\mathrm{DFT}_N[\widetilde{w}]$ s'obtient comme le produit, coordonnée par coordonnée, des deux vecteurs de longueur N.

$$\begin{aligned} \mathrm{DFT}_N(\widetilde{u}) &= \mathrm{DFT}_N[(\widetilde{u}_0, ..., \widetilde{u}_{N-1})] \\ \mathrm{DFT}_N(\widetilde{v}) &= \mathrm{DFT}_N[(\widetilde{v}_0, ..., \widetilde{v}_{N-1})] \end{aligned}$$

et que son calcul, une fois les deux transformées de Fourier discrètes calculées, nécessite juste N multiplications. On calcule ensuite le vecteur $\widetilde{w}$ (et donc les calculs des coefficients du produit PQ) par transformation de Fourier discrète inverse. Si N est une puissance de 2, ces calculs s'implémentent de manière rapide comme des calculs de FFT.

3.9 Approximation et moindres carrés

Soient $(x_0, y_0), ..., (x_N, y_N)$ $N+1$ points de $\mathbb{R}^2$, avec $x_0, ..., x_N$ distincts. Plutôt que de chercher une fonction polynômiale P (de degré N) telle que $P(x_k) = y_k$ (exactement)

[18] Le choix de $N = 2^p$, où p est premier et $M_p = N - 1$ est un nombre de Mersenne premier, s'avère, pour des raisons algorithmiques, un choix privilégié.

pour $k = 0, ..., N$, on peut chercher, si $0 \leq M \leq N$, quelle est la fonction polynômiale p de degré M telle que

$$\sum_{j=0}^{N} (y_j - p(y_j))^2$$

soit minimale (parmi tous les choix possibles de telles fonctions polynômiales p). Le polynôme p recherché est de la forme

$$p(X) = a_0 + a_1 X + \cdots + a_M X^M \,,$$

les inconnues de notre problème sont les coefficients $a_0, ..., a_M$.

3.9.1 Le cas $M = 1$: la droite de régression d'un « nuage » de points

Cherchons dans cette section la meilleure fonction polynômiale de degré $M = 1$. Il s'agit donc de trouver des réels a et b tels que

$$F(a, b) := (y_0 - ax_0 - b)^2 + \cdots + (y_N - ax_N - b)^2$$

soit minimale. Introduisons les moyennes m_x et m_y des valeurs respectives des x_j et y_j, soit

$$m_x := \frac{x_0 + \cdots + x_N}{N+1}$$

$$m_y := \frac{y_0 + \cdots + y_N}{N+1}$$

et posons $x'_j = x_j - m_x$ et $y'_j = y_j - m_y$ pour $j = 0, ..., N$. On a $\sum_{j=0}^{N} x'_j = \sum_{j=0}^{N} y'_j = 0$.
Si nous trouvons a' et b' minimisant la fonction

$$G(a', b') := (y'_0 - a'x'_0 - b')^2 + \cdots + (y'_N - a'x'_N - b')^2 \,,$$

on en déduira que les valeurs de a et b minimisant $F(a, b)$ sont

$$a = a' \,, \quad b = b' + m_y - a'm_x \,.$$

Un calcul simple montre que

$$\begin{aligned} G(a', b') &= a'^2 \sum_{j=0}^{N} x'^2_j - 2a \sum_{j=0}^{N} x'_j y'_j + (N+1)b'^2 + \sum_{j=0}^{N} y'^2_j \\ &= \Big(a' \sqrt{\sum_{j=0}^{N} x'^2_j} - \frac{\sum_{j=0}^{N} x'_j y'_j}{\sqrt{\sum_{j=0}^{N} x'^2_j}} \Big)^2 + (N+1)b'^2 + \sum_{j=0}^{N} y'^2_j - \frac{\Big(\sum_{j=0}^{N} x'_j y'_j \Big)^2}{\sum_{j=0}^{N} x'^2_j} \,; \end{aligned}$$

le minimum de cette fonction est donc atteint pour

$$a' = \frac{\sum_{j=0}^{N} x'_j y'_j}{\sum_{j=0}^{N} x'^2_j} \,, \quad b' = 0$$

et vaut

$$\min G(a',b') = \min F(a,b) = \sum_{j=0}^{N} y_j'^2 - \frac{\Big(\sum\limits_{j=0}^{N} x_j' y_j'\Big)^2}{\sum\limits_{j=0}^{N} x_j'^2}$$

(qui, remarquons-le, est forcément une quantité positive ou nulle, ce qui donne une inégalité très importante dans toutes les mathématiques, dite *inégalité de Cauchy-Schwarz* [19]).

La droite affine réalisant le meilleur compromis concernant la dépendance linéaire de l'information y à partir de l'information x au sein du nuage de points est donc la droite affine d'équation

$$y - m_y = a'(x - m_x)\,;$$

cette droite importante est dite *droite de régression linéaire* et le nombre

$$\rho := \frac{\sum\limits_{j=0}^{N} x_j' y_j'}{\sqrt{\sum\limits_{j=0}^{N} x_j'^2}\,\sqrt{\sum\limits_{j=0}^{N} y_j'^2}}$$

est dit *coefficient de corrélation entre les x_j et les y_j* au sein du nuage [20]. Ainsi la droite de régression linéaire a-t-elle pour équation cartésienne

$$\frac{y - m_y}{\sqrt{\sum\limits_{j=0}^{N} y_j'^2}} = \rho \frac{x - m_x}{\sqrt{\sum\limits_{j=0}^{N} x_j'^2}}\,.$$

Ces notions jouent un rôle très important en théorie des probabilités et plus particulièrement dans l'étude des modèles statistiques (comme les enquêtes d'opinion).

Remarquons que la seule hypothèse dont nous avons eu besoin pour définir la droite de régression est que

$$\sum_{k=0}^{N} x_k'^2 > 0\,,$$

ce qui équivaut à dire que les points (x_k, y_k) ne sont pas tous alignés sur une droite verticale. Si tel était le cas, la droite de régression du nuage serait bien sûr cette droite verticale.

3.9.2 Un regard géométrique sur le problème

Si $x_0, ..., x_N$ sont $N+1$ nombres réels distincts, l'ensemble des fonctions définies sur l'ensemble $\{x_0, ..., x_N\}$ et à valeurs réelles a une structure de $\mathbb{R}$-espace vectoriel $\mathcal{V}$ de dimension $N+1$. La somme $f+g$ de deux fonctions est la fonction définie par

$$(f+g)(x_k) := f(x_k) + g(x_k)\,,\ k = 0, ..., M\,;$$

[19] Si X et Y sont deux vecteurs de $\mathbb{R}^{N+1}$, si $\langle\,,\,\rangle$ désigne le produit scalaire canonique et $\|\ \|_2$ la norme euclidienne sur $\mathbb{R}^{N+1}$, on a $|\langle X, Y\rangle| \leq \|X\|_2\,\|Y\|_2$, l'inégalité ne devenant une égalité que si X et Y sont liés.

[20] D'après le cas d'égalité dans l'inégalité de Cauchy-Schwarz, ce coefficient de corrélation est égal à 1 si et seulement si les points (x_k, y_k) sont sur une droite du plan $\mathbb{R}^2$.

la multiplication d'une fonction f par un scalaire $\lambda \in \mathbb{R}$ est la fonction définie par

$$(\lambda \cdot f)(x_k) = \lambda f(x_k)\,,\ k = 0, ..., N\,.$$

L'ensemble des restrictions à $\{x_0, ..., x_N\}$ des fonctions polynômiales de degré M ($0 \leq M \leq N$) est donc un $\mathbb{R}$-sous-espace vectoriel $\mathcal{W}_M$ de $\mathcal{V}$ de dimension $M+1$: ceci résulte du fait que, si $x_0, ..., x_N$ sont $N+1$ nombres réels distincts, les vecteurs

$$\begin{pmatrix} 1 \\ x_j \\ x_j^2 \\ \vdots \\ x_j^N \end{pmatrix}\,,\ j = 0, ..., N\,,$$

sont linéairement indépendants, ce qui implique que la matrice constituée avec leurs colonnes est de rang $N+1$ (son déterminant est un déterminant de Van der Monde, voir les cours d'Algèbre MHT201 et MHT301). Une base de $\mathcal{W}_M$ est constituée des restrictions à $\{x_0, ..., x_N\}$ des fonctions

$$Y_k\ : x \mapsto x^k\,,\ k = 0, ..., M\,.$$

Sur l'espace $\mathcal{E}$, on peut définir un produit scalaire par

$$\langle f\,,\, g\rangle := \sum_{j=0}^{N} f(x_j)g(x_j)\,.$$

On peut donc définir une opération de projection orthogonale sur le sous-espace vectoriel $\mathcal{W}_M$ (voir le cours d'algèbre MHT301). La fonction polynômiale p de degré M telle que

$$\sum_{j=0}^{N}(y_j - p(x_j))^2$$

soit minimale est l'élément de $\mathcal{W}_M$ le plus proche de la fonction

$$Y\ : x_k \mapsto y_k,\quad k = 0, ..., N\,,$$

relativement à la distance associée à ce produit scalaire. On a donc, grâce au théorème de Pythagore

$$p = \mathrm{Proj}_{\mathcal{V}_M}[Y]\,.$$

Cette projection se calcule aisément si l'on dispose d'une base orthonormée de $\mathcal{W}_M$; malheureusement, la construction d'une telle base *via* le procédé d'orthonormalisation de Gram-Schmidt [21] s'avère souvent délicate numériquement car impliquant des problèmes de sous-conditionnement ; on peut aussi calculer cette projection en écrivant

$$\langle Y - p\,,\, Y_k\rangle = 0\,,\quad k = 0, ..., M\,,$$

[21] Si on l'attribue au mathématicien allemand Erhard Schmidt (1876-1959), spécialiste d'analyse fonctionnelle, et au mathématicien danois Jørgen Pedersen Gram (1853-1916), qui s'est penché sur la méthode des moindres carrés, ce procédé algorithmique (que vous avez rencontré dans le cours d'algèbre de MHT301) est certainement bien plus ancien et connu de Cauchy et Laplace au début du XIX-ème siècle (Cauchy le manipula expressément vers 1830).

soit encore

$$\left\langle Y - \sum_{j=0}^{M} a_j x^j \, , x^k \right\rangle = 0 \, , \;\; k = 0, ..., M \, .$$

Ceci s'écrit

$$\sum_{j=0}^{N} \Big(y_j - \sum_{l=0}^{M} a_l x_j^l \Big) x_j^k = 0 \, , \;\; k = 0, ..., M \, ,$$

c'est à dire comme un système linéaire en $a_0, ..., a_M$. La matrice d'un tel système (dite *matrice de Gram*) s'avère être souvent mal conditionnée. Sur l'exemple ci-dessous, on a représenté un nuage de points (20 points de coordonnées x, y avec $-1 \leq x \leq 2$ et $4 \leq y \leq 10$) sur lequel on a fait agir les routines MATLAB :

```
>> P= polyfit(x,y,1);
>> PP= polyfit(x,y,10);
>> PPP=polyfit(x,y,14);
>> t=-0.5:.01:1.5 ;
>> z=polyval (P,t);
>> zz=polyval (PP,t);
>> zzz=polyval (PPP,t);
>> plot(x,y,'+');
>> hold
>> plot(t,z,'r');
>> plot(t,zz,'-.');
>> plot(t,zzz,'k');
```

On a ainsi affiché les graphes de la droite de régression et des polynômes optimaux de degré 1, 10 et 14.

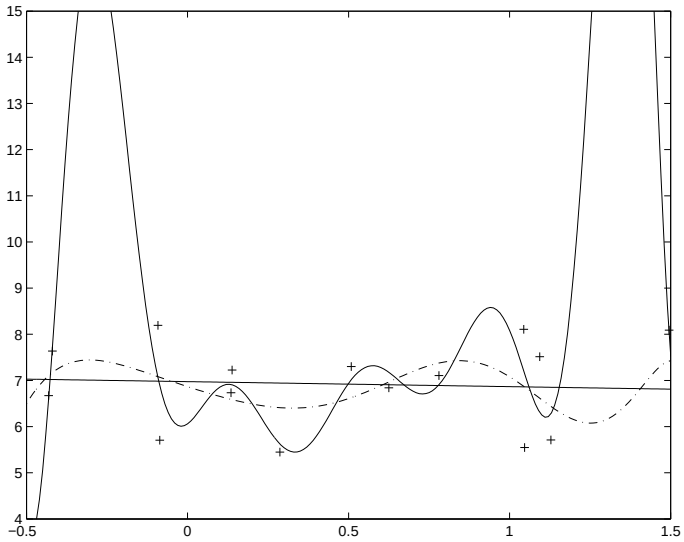

FIG. 3.6 – Approximation polynômiale au sens des moindres carrés

3.9.3 Retour aux calculs approchés d'intégrales : les familles de polynômes orthogonaux

Soit $[a,b]$ un intervalle de $\mathbb{R}$ et

$$w \ : [a,b] \longmapsto [0,\infty]$$

une fonction numérique telle que

$$\int_a^b w(t)\,dt < +\infty$$

(on dit que w est un *poids*).

Si nous appliquons un procédé inspiré de Gram-Schmidt, nous construisons, pour chaque entier $N \geq 0$, un unique polynôme unitaire

$$P_{w,N+1}(X) = X^{N+1} + a_{N,1}X^N + \ldots + a_{N,N}$$

de degré exactement $N+1$ tel que

$$\int_a^b P_{w,N+1}(t)\,t^k\,w(t)\,dt = 0\,,\ \ k = 0,...,N\,.$$

Les constructions les plus célèbres sont celles où
- $[a,b] = [-1,1]$ et $w \equiv 1$;
- $[a,b] = [-1,1]$ et $w \equiv (\sqrt{1-t^2})^{-1}$.

Dans le premier cas, on voit que, pour tout $N \geq 0$,

$$P_{w,N+1}(t) = \frac{(-1)^{N+1}(N+1)!}{(2(N+1))!}\Big(\frac{d}{dt}\Big)^{N+1}[(1-t^2)^{N+1}]\,.$$

C'est la formule d'intégration par parties qui permet de vérifier (faites l'exercice) que

$$\int_{-1}^1 P_{w,N+1}(t)\,t^k\,dt = 0\,,\quad k = 0,...,N\,,$$

pour tout $N \in \mathbb{N}$ et donc que $P_{w,N+1}$ est l'unique polynôme unitaire de degré exactement $N+1$ qui convient. Ces polynômes $L_1, L_2, ..., ...$ sont attachés au nom d'Adrien-Marie Legendre [22] et sont en général plutôt normalisés par

$$L_N(t) = \frac{1}{2^N N!}\Big(\frac{d}{dt}\Big)^N[(1-t^2)^N]\,.$$

Dans le second cas, on voit que, toujours pour $N \geq 0$, $P_{w,N+1} = T_{N+1}$ est le polynôme de degré $N+1$ donné par la relation

$$\cos((N+1)\theta) = 2^N P_{w,N+1}(\cos\theta)\,;$$

[22] Géomètre et analyste français (1752-1833), c'est lui qui formalisa (à l'occasion de ses calculs en mécanique céleste) la « méthode des moindres carrés » présentée dans cette section.

ceci se voit grâce à la formule de changement de variables dans les intégrales, qui nous donne, si $M_1 > M_2 \geq 1$,

$$\begin{aligned}\int_{-1}^{1} \frac{P_{w,M_1}(t)P_{w,M_2}(t)}{\sqrt{1-t^2}}\,dt &= \int_0^{\pi} P_{w,M_1}(\cos\theta)P_{w,M_2}(\cos\theta)\,d\theta \\ &= \frac{1}{2^{M_1+M_2-2}}\int_{-\pi}^{\pi}\cos(M_1\theta)\cos(M_2\theta)\,d\theta = 0\,.\end{aligned}$$

Ces polynômes $P_{w,N+1} = T_{N+1}$, $N = 0, 1, ...,$, sont dans ce cas les polynômes de Tchebychev [23].

On pourrait vérifier (ce que conforte ces deux exemples) grâce au théorème de Rolle utilisé de manière répétitive que $P_{w,N+1}$ s'annule exactement $N+1$ fois sur $[a, b]$. Si l'on examine les graphes par exemple de L_{30} et de T_{100} (convenablement normalisés) et que nous avons représenté sur les figures ci-dessous (les graphes ont été traçés sous MATLAB), on constate que dans le second cas les zéros s'accumulent au bord, tandis que dans le premier cas, les oscillations sont drastiquement accentuées près des extrémités de l'intervalle tandis que les zéros se répartissent plus uniformément dans l'intervalle.

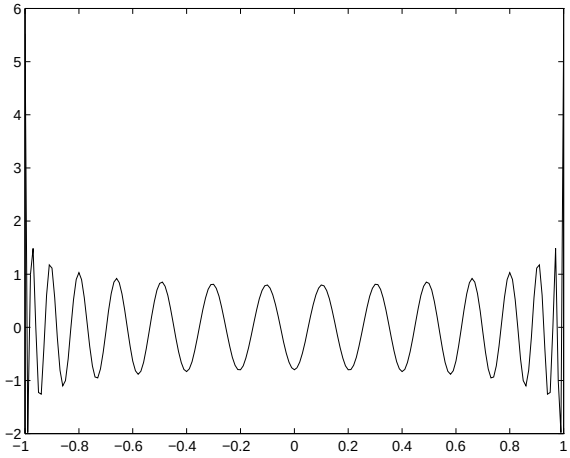

FIG. 3.7 – Le polynôme de Legendre L_{30} normalisé

[23] Le mathématicien russe Pafnouti Tchebychev (1821-1894) est l'un des pères de ce qui deviendra la théorie moderne des probabilités.

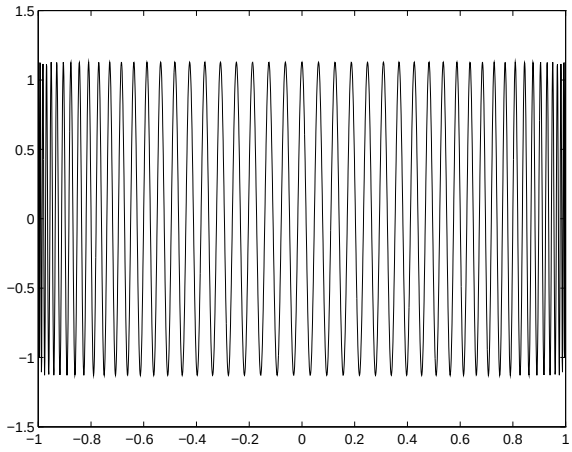

FIG. 3.8 – Le polynôme de Tchebychev T_{100} normalisé

Si l'on utilise les zéros $x_0, ..., x_N$ de $P_{w,N+1}$ pour générer (comme dans les formules de Newton-Cotes de la section 3.7) des coefficients $u_0, ..., u_N$ de manière à ce que la formule de calcul approché d'intégrale

$$\int_a^b f(t)w(t)\,dt \simeq \sum_{k=0}^{N} u_k f(x_k) \tag{3.9}$$

proposée dans la section 3.7 soit exacte pour toute fonction polynômiale de degré au plus N, on constate que cette formule est même exacte pour toute fonction polynômiale de degré au plus $2N+1$. En effet, un polynôme P de degré au plus $2N+1$ se divise par $P_{w,N+1}$ suivant la division euclidienne et s'écrit

$$P(X) = P_{w,N+1}(X)Q(X) + R(X)$$

avec $\deg Q \leq 2N+1-N-1 = N$ et $\deg R \leq N$. Or, on a

$$\int_a^b R(t)\,w(t)\,dt = \sum_{k=0}^{N} u_k R(x_k)$$

grâce au choix des coefficients u_k et

$$\int_a^b P_{w,N+1}(t)Q(t)w(t)\,dt = 0$$

(car $\deg Q \leq N$) par construction de $P_{w,N+1}$; on a aussi $P_{w,N+1}(x_k) = 0$ pour $k = 0, ..., N$. Au final, on a bien

$$\int_a^b P(t)\,w(t)\,dt \;=\; \int_a^b R(t)\,w(t)\,dt = \sum_{k=0}^{N} u_k R(x_k) = \sum_{k=0}^{N} u_k P(x_k)\,,$$

ce qui prouve l'exactitude de la formule (3.9) pour toute fonction polynômiale de degré au plus $2N+1$ (alors que ceci n'était prévu *a priori* que pour les fonctions polynômiales de degré au plus N !).

Définition 3.1 *Les points $x_0, ..., x_N$, zéros de $P_{w,N+1}$, sont dits points de Gauss* [24] *et la formule approchée*

$$\int_a^b f(t)\, w(t)\, dt \simeq \sum_{k=0}^{N} u_k f(x_k)\,,$$

exacte pour les fonctions polynômiales de degré au plus $2N+1$ est dite formule de quadrature de Gauss.

3.10 Peut-on « accélérer » la convergence des approximations ?

Nous allons profiter de ce chapitre (et en particulier du matériel introduit dans les sections 3.7 et 3.9.3) pour introduire quelques méthodes simples d'accélération de convergence dans ces méthodes numériques (par exemple de calcul approché d'intégrales) présentant un terme d'erreur de la forme

$$\varphi(h) = Ah^k + o(h^k)\,,$$

h désignant par exemple un pas de découpage de l'intervalle d'intégration.

3.10.1 Le procédé d'« extrapolation » de L.W. Richardson

On a vu (section 3.7) que le calcul approché d'une intégrale

$$\int_a^b f(t)\, dt$$

se traite en général par un procédé composite consistant à subdiviser $[a, b]$ en N intervalles égaux de longueur $h = (b-a)/N$ et à utiliser pour le calcul approché de l'intégrale sur chacun des intervalles de la subdivision une méthode de Newton-Cotes (trapèzes, Simpson, *etc.*). Si la méthode de Newton-Cotes utilisée est d'ordre p (par exemple $p = 3$ pour la méthode des trapèzes, $p = 5$ pour la méthode de Simpson, *etc.*), le calcul approché $I(h)$ de l'intégrale par le procédé composite associé se présentera, on l'a vu en fin de la section 3.7, sous la forme

$$I(h) = \int_a^b f(t)\, dt + \alpha h^{p-1} + o(h^{p-1})\,. \tag{3.10}$$

Connaître explicitement α n'est pas évident, mais si l'on refait le calcul approché en divisant le pas par 2, on trouve, remplaçant h par $h/2$ dans (3.10),

$$I(h/2) = \int_a^b f(t)\, dt + \alpha (h/2)^{p-1} + o(h^{p-1}) \tag{3.11}$$

(tous les $o(h^{p-1})$ ont ici été notés de manière identique pour alléger les notations). En soustrayant (3.11) à (3.10), il vient

$$\alpha (h/2)^{p-1} \sim \frac{I(h) - I(h/2)}{2^{p-1} - 1}$$

[24] On retrouve ici le mathématicien, astronome et philosophe allemand Carl Friedrich Gauss (1777-1855), certainement l'un de ceux qui ont le plus contribué à guider l'évolution des mathématiques tous domaines confondus (algèbre, analyse, théorie des nombres et géométrie).

au voisinage de $h = 0$, ce qui implique donc que l'étude du comportement de

$$h \longmapsto \frac{I(h) - I(h/2)}{(2^{p-1} - 1)(h/2)^{p-1}}$$

lorsque h tend vers 0 nous permet de déterminer explicitement α (au moins de manière approchée) et d'être par là même capable de contrôler l'ordre de grandeur (et le signe si $\alpha \neq 0$) de l'erreur numérique

$$\int_a^b f(t)dt - I(h)\,.$$

Il y a aussi un autre moyen d'exploiter pareille idée, aux fins cette fois d'en tirer une approximation plus efficace (c'est-à-dire convergeant plus rapidement) de l'intégrale inconnue. Ce procédé à été initié assez récemment par L.F. Richardson [25] et est aujourd'hui très utilisé dans le calcul approché des limites de suites ou des sommes de séries numériques.

On multiplie cette fois la relation (3.11) par 2^{p-1} et l'on soustrait la relation obtenue à la formule (3.10), ce qui donne

$$(1 - 2^{p-1}) \int_a^b f(t)\,dt = I(h) - 2^{p-1} I(h/2) + o(h^{p-1})\,,$$

ou encore

$$\int_a^b f(t)\,dt = \frac{I(h) - 2^{p-1} I(h/2)}{1 - 2^{p-1}} + o(h^{p-1})\,.$$

Si l'on pose

$$\widetilde{I}(h) := \frac{I(h) - 2^{p-1} I(h/2)}{1 - 2^{p-1}}\,, \tag{3.12}$$

on constate que l'erreur commise en remplaçant l'intégrale

$$\int_a^b f(t)\,dt$$

par $\widetilde{I}(h)$ est cette fois en $o(h^{p-1})$, alors que l'erreur commise en la remplaçant par $I(h)$ était en $O(h^{p-1})$!

3.10.2 La méthode de Romberg

Reprenons le calcul approché d'intégrale évoqué dans la sous-section précédente en relation avec le procédé d'extrapolation de Richardson.

Les choses peuvent être même plus précises si l'on sait *a priori* que

$$\int_a^b f(t)\,dt = I(h) + \alpha_0 h^{p-1} + \alpha_1 h^p + \cdots + \alpha_k h^{p-1+k} + o(h^{p-1+k})\,. \tag{3.13}$$

[25] Les travaux du physicien et mathématicien anglais Lewis Fry Richardson (1881-1953) ont été pour une grande part tournés vers les prévisions météorologistes ; c'est dans cette optique qu'a surgi la technique d'extrapolation (et d'accélération de convergence) que nous mentionnons ici.

En écrivant aussi

$$\int_a^b f(t)\,dt = I(h/2) + \alpha_0(h/2)^{p-1} + \alpha_1(h/2)^p + \cdots + \alpha_k(h/2)^{p-1+k} + o(h^{p-1+k})$$

et en combinant avec la relation précédente, on trouve

$$\begin{aligned}\int_a^b f(t)\,dt &= \frac{I(h) - 2^{p-1}I(h/2)}{1-2^{p-1}} \\ &\quad +\alpha_1\frac{1-1/2}{1-2^{p-1}}\,h^p + \alpha_2\frac{1-1/4}{1-2^{p-1}}\,h^{p+1} + \cdots + \alpha_k\frac{1-1/2^k}{1-2^{p-1}}\,h^{p-1+k} \\ &\quad +o(h^{p-1+k})\,,\end{aligned}$$

relation que l'on peut encore écrire

$$\int_a^b f(t)\,dt = \widetilde{I}(h) + \widetilde{\alpha}_0 h^{\tilde{p}-1} + \widetilde{\alpha}_1 h^{\tilde{p}} + \cdots + \widetilde{\alpha}_{\tilde{k}} h^{\tilde{p}-1+\tilde{k}} + o(h^{\tilde{p}-1+\tilde{k}}) \tag{3.14}$$

avec $\tilde{p} := p+1$, $\tilde{k} := k-1$ et

$$\widetilde{I}(h) := \frac{I(h) - 2^{p-1}I(h/2)}{1-2^{p-1}}\,.$$

On remarque que la nouvelle relation obtenue (3.14) est exactement du type de la relation (3.13), ce qui nous permet de réitérer le processus en écrivant

$$\int_a^b f(t)\,dt = \check{I}(h) + \check{\alpha}_0 h^{\check{p}-1} + \check{\alpha}_1 h^{\check{p}} + \cdots + \check{\alpha}_{\check{k}} h^{\check{p}-1+\check{k}} + o(h^{\check{p}-1+\check{k}}) \tag{3.15}$$

avec $\check{p} := \tilde{p}+1 = p+2$, $\check{k} := \tilde{k}-1 = k-2$ et

$$\check{I}(h) := \frac{\widetilde{I}(h) - 2^{\tilde{p}-1}\widetilde{I}(h/2)}{1-2^{\tilde{p}-1}}\,.$$

On est ainsi en position de recommencer, et ainsi de suite jusqu'à ce que l'on ait épuisé le développement limité. L'opération peut ainsi être itérée k fois et conduit à une approximation de l'intégrale

$$\int_a^b f(t)\,dt$$

avec une erreur en $o(h^{p-1+k})$, tous les termes de la partie principale développement jusqu'à cet ordre ayant été « aspirés ».

Ceci est d'autant plus important que l'on sait, d'après une formule sommatoire dite formule d'Euler-MacLaurin [26], que le terme d'erreur dans la méthode des trapèzes composite se présente précisément sous la forme

$$\int_a^b f(t)\,dt - I(h) = \alpha_0 h^2 + \alpha_2 h^4 + \cdots + \alpha_{2(p-1)} h^{2(K-1)} + O(h^{2K})\,, \tag{3.16}$$

[26] Cette formule, établie par Leonhard Euler et le mathématicien écossais Colin MacLaurin autour

ce qui permet de lui appliquer le processus d'extrapolation de Richardson de manière itérative comme ci-dessus. Le procédé décrit précédemment et transposé à cet exemple particulier est ce que l'on appelle aujourd'hui la *méthode de Romberg*[27]. C'est une méthode très utilisée du fait de son efficacité.

Remarque 4.2. On peut se poser la question suivante : pourquoi parler d'« extrapolation » à propos de la méthode de L. F. Richardson (ou de celle de W. Romberg qui s'en déduit) ? La raison en est que la quantité

$$\frac{I(h) - 2^{p-1} I(h/2)}{1 - 2^{p-1}}$$

introduite dans (3.12) s'interprète comme une « extrapolation[28] » de la fonction I à partir de ses valeurs en h et $h/2$. Cette idée n'est pas sans rapport avec l'identité de Bézout rappelée dans la section 3.1 : si N_1 et N_2 (par exemple $N_1 = 2$ et $N_2 = 3$) sont deux nombres premiers entre eux, les polynômes

$$1 + X + \cdots + X^{N_1 - 1} \quad \text{et} \quad 1 + X + \cdots + X^{N_2 - 1}$$

sont premiers entre eux et l'on peut trouver deux polynômes U_1 et U_2 tels que

$$U_1(X)(1 + \cdots + X^{N_1-1}) + U_2(X)(1 + \cdots + X^{N_2-1}) = 1\,,$$

soit aussi, en multipliant par $X - 1$,

$$U_1(X)(X^{N_1} - 1) + U_2(X)(X^{N_2} - 1) = X - 1\,.$$

Si $(u_n)_{n\in\mathbb{Z}}$ est une suite de nombres complexes indexée par $\mathbb{Z}$, cette relation permet de déduire (essayez d'expliciter comment) un procédé de calcul de la suite $(u_{n+1} - u_n)_{n\in\mathbb{Z}}$ à partir des deux suites $(u_{n+N_1} - u_n)_{n\in\mathbb{Z}}$ et $(u_{n+N_2} - u_n)_{n\in\mathbb{Z}}$, permettant ainsi d'extrapoler les différences successives entre les u_n depuis les différences prises entre les mêmes u_n, mais avec des sauts de $N_1 - 1$ indices, et celles prises avec des sauts de $N_2 - 1$ indices.

---

de 1735, stipule en effet que, si f est de classe C^{2K} sur le segment $[0, N]$,

$$\begin{aligned}\int_0^N f(t)\,dt &= \Big[\frac{f(0)}{2} + \sum_{j=1}^{N-1} f(j) + \frac{f(N)}{2}\Big] - \sum_{k=1}^{K} \frac{b_{2j}}{(2j)!}\Big(f^{(2j-1)}(N) - f^{(2j-1)}(0)\Big) \\ &\quad + \int_0^N f^{(2K)}(t)\frac{B_{2K}(t - [t])}{(2K)!}\,dt\,,\end{aligned}$$

où les $b_2, ..., b_{2K}$ sont des nombres rationnels indépendants de f (les *nombres de Bernouilli* d'indices $2, 4, ..., 2K$) et B_{2K} un certain polynôme (lui aussi indépendant de f), dit polynôme de Bernouilli d'indice $2K$. Il suffit d'appliquer cette formule à la fonction

$$t \longmapsto f(a + ht)\,, \quad h = \frac{b-a}{N}$$

(définie sur $[0, N]$) pour en déduire le résultat (3.16) voulu. Vous démontrerez plus tard la formule importante d'Euler-MacLaurin ; il s'agit d'une formule dans le même esprit que la *formule de Taylor avec reste intégral*.

[27] Du nom du mathématicien allemand Werner Romberg (1909-2003) qui l'introduisit dans ses travaux en intégration numérique.

[28] « Extrapoler » une fonction à partir de ses valeurs en des points donnés x_k, c'est pouvoir reconstituer ses valeurs en d'autres points que les points x_k où la fonction est *a priori* donnée ; ce n'est pas la même chose qu'« interpoler » une fonction f aux points x_k par une fonction plus simple $\widetilde{f}$ appartenant à une classe connue (par exemple une fonction polynômiale), même si la connaissance des $\widetilde{f}(x)$ pour x différent des x_k fournit une manière d'extrapoler f depuis ses valeurs aux points x_k.

Chapitre 4

Initiation aux méthodes itératives

4.1 Le théorème du point fixe : un énoncé de théorème « constructif »

Un résultat majeur pour nous soutendra tout ce chapitre basé sur l'utilisation de méthodes itératives. C'est le résultat suivant, l'important théorème du point fixe, que nous énoncerons ici dans un cas très particulier (on en donnera la preuve un peu plus tard dans ce cours).

Théorème 4.1 *Soit T une application de $\mathbb{R}^n$ dans lui-même, contractant strictement les distances, c'est-à-dire telle qu'il existe une constante $\kappa \in [0,1[$ telle que*

$$\forall X, Y \in \mathbb{R}^n\,,\ \|T(X) - T(Y)\| \leq \kappa \|X - Y\| \tag{4.1}$$

(la norme étant ici par exemple la norme euclidienne, ce n'est pas important). L'application T a un unique point fixe (sous l'action de T) dans $\mathbb{R}^n$ et ce point s'atteint en partant de n'importe quel point X_0 de $\mathbb{R}^n$ comme la limite de la suite $(X_k)_{k\geq 0}$ définie récursivement par

$$X_k = T(X_{k-1})\,,\ \forall\, k \geq 1\,.$$

4.2 Comment résoudre un système linéaire devient un problème de point fixe ?

Pour illustrer les démarches algorithmiques inspirées de la méthode du point fixe, nous donnerons ici une piste d'attaque pour envisager la résolution d'un système linéaire de Cramer

$$M \cdot X = B$$

(M étant une matrice inversible $n \times n$ à entrées réelles et B un vecteur de $\mathbb{R}^n$). On sait que ce système admet une solution unique dans $\mathbb{R}^n$, donnée par

$$X = M^{-1} \cdot B\,.$$

Notre démarche pour résoudre ce système va contourner le problème du calcul de l'inverse de la matrice M, au moins lorsque celle ci aura une forme particulière,

dite *à diagonale dominante*. C'est la méthode proposée par Carl-Gustav Jacobi[1], à la base de ce que l'on appelle l'*algorithme itératif de Jacobi*, implémentable pour résoudre un système $M \cdot X = B$ sous certaines hypothèses sur la matrice M, parmi lesquelles l'hypothèse essentielle que les termes diagonaux de cette matrice soient tous non nuls.

La matrice M que nous allons envisager ici sera *à diagonale dominante*, au sens de la définition suivante :

Définition 4.1 *Une matrice $A = [a_{i,j}]_{1 \leq i,j \leq n}$, i pour l'indice de ligne, j pour l'indice de colonne (à coefficients réels ou complexes) est dite à diagonale dominante si et seulement si*

$$\forall\, i = 1, ..., n\,,\ |a_{i,i}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{i,j}|\,.$$

Exemple 2.1. Les matrices

$$\begin{pmatrix} 5 & 2 & 2 \\ 1 & 6 & 3 \\ 3 & 4 & -8 \end{pmatrix} \qquad \begin{pmatrix} 5+2i & 3 & 2 \\ 1 & 3(1+i) & 3 \\ 3+i & 4 & -8-i \end{pmatrix}$$

sont à diagonale dominante (la première en tant que matrice à entrées réelles, la seconde en tant que matrice à entrées complexes).

Remarque 2.1. Considérons ($\mathbb{K}$ désignant $\mathbb{R}$ ou $\mathbb{C}$) la norme

$$\|X\|_\infty = \max_j |x_j|$$

sur les vecteurs X de $\mathbb{K}^n$ (de coordonnées $x_1, ..., x_n$ et représentés comme des vecteurs colonnes). Cette norme induit au niveau des matrices $n \times n$ à entrées dans $\mathbb{K}$ la norme

$$\|A\|_\infty := \sup_{X \in \mathbb{R}^n \setminus \{0\}} \frac{\|A \cdot X\|_\infty}{\|X\|_\infty} = \sup_i \Big(\sum_j |a_{i,j}| \Big)$$

On verra que, si M est à diagonale dominante, D désignant la matrice (diagonale inversible) obtenue en extrayant de A les termes diagonaux et en complétant par des zéros, que, si $M = D - E$, la matrice $D^{-1} \cdot E$ a précisément la propriété

$$\|D^{-1}.E\|_\infty = \sup_i \Big(\frac{\sum_{j \neq i} |a_{i,j}|}{|a_{i,i}|} \Big) < 1\,.$$

Cette remarque sera capitale par la suite.

Nous prendrons ici comme exemple la matrice (à diagonale dominante et entrées réelles)

```
>> M=[30 7 8 7 ; 7 20 6 5 ; 8 6 40 9 ;7 5 9 30]

M =

    30     7     8     7
     7    20     6     5
     8     6    40     9
     7     5     9    30
```

[1] Algébriste et géomètre allemand (1804-1851), il marqua l'essor des mathématiques au XIX-ème siècle (déterminants, fonctions elliptiques, algèbre linéaire, problèmes géométriques d'intersection,...).

Nous allons envisager la résolution du système de Cramer $M \cdot X = B$ avec ici

```
>> B=[32 ; 23 ; 33; 31]

B =

    32
    23
    33
    31
```

La solution immédiate proposée sous MATLAB pour la résolution directe de ce système linéaire (avec les erreurs numériques que cela comporte) est

```
>> M^(-1)*B

ans =

   0.649375600384246
   0.625452420067077
   0.457215765882871
   0.640405560134302
```

Nous allons, plutôt que cette résolution directe, la transformer en la recherche d'un point fixe d'une certaine application affine de $\mathbb{R}^n$ dans lui-même (ici $n = 4$). Posons pour cela (c'est la démarche générale proposée dans l'algorithme itératif de Jacobi)

$$M = D - E$$

en « isolant »les termes diagonaux de M dans la matrice D. Ici

```
>> D=[30 0 0 0 ; 0 20 0 0 ; 0 0 40 0 ; 0 0 0 30]

D =

    30     0     0     0
     0    20     0     0
     0     0    40     0
     0     0     0    30

>> E=[0 -7 -8 -7 ; -7 0 -6 -5 ; -8 -6 0 -9 ; -7 -5 -9 0]

E =

     0    -7    -8    -7
    -7     0    -6    -5
    -8    -6     0    -9
    -7    -5    -9     0
```

Résoudre $M \cdot X = B$ revient à chercher X tel que

$$D \cdot X = E \cdot X + B\,,$$

ou encore, puisque D est immédiatement inversible (et d'inverse immédiat à calculer car inverse d'une matrice diagonale à coefficients diagonaux tous non nuls)

$$X = U \cdot X + V$$

avec $U := D^{-1} \cdot E$ et $V \ : D^{-1} \cdot B$. Résoudre notre système $M \cdot X = B$ revient donc à chercher (si tant est qu'il existe et est unique) le point fixe de l'application affine $T \ : \mathbb{R}^4 \longrightarrow \mathbb{R}^4$ qui à X associe $U \cdot X + B$. Nous écrivons donc la routine MATLAB calculant, initiée avec un vecteur arbitraire X, la suite des itérés :

$$X_0 = X,\ X_1 = T(X),\ X_2 = T(T(X)\ , ...,\ X_k = T(X_{k-1})\,,\ k \geq 1$$

en imposant un nombre N d'itérations.

```
function f=jacobi(X,U,V,N);

for j=1:N
    X=U*X+V;
end
f=X;
```

Le résultat est le suivant, testé pour 50, 100, 150, 200 itérations à partir d'un vecteur X arbitraire.

```
>> X= [1; -7; -4; 10]

X =

     1
    -7
    -4
    10

>> jacobi(X,U,V,50)

ans =

   0.649375579812054
   0.625452396020122
   0.457215748919845
   0.640405540658039

>> jacobi(X,U,V,100)

ans =

   0.649375600384245
   0.625452420067076
   0.457215765882871
   0.640405560134301
```

```
>> jacobi(X,U,V,150)

ans =

   0.649375600384246
   0.625452420067077
   0.457215765882871
   0.640405560134302

>> jacobi(X,U,V,200)

ans =

   0.649375600384246
   0.625452420067077
   0.457215765882871
   0.640405560134302
```

et, si l'on part d'un autre vecteur

```
>> X= [168; -754; -432; 1218]

X =

         168
        -754
        -432
        1218

>> jacobi(X,U,V,50)

ans =

   0.649378833791283
   0.625456199615299
   0.457218432024114
   0.640408621290100

>> jacobi(X,U,V,100)

ans =

   0.649375600384397
   0.625452420067254
   0.457215765882996
   0.640405560134445

>> jacobi(X,U,V,150)

ans =
```

```
   0.649375600384246
   0.625452420067077
   0.457215765882871
   0.640405560134302

>> jacobi(X,U,V,200)

ans =

   0.649375600384246
   0.625452420067077
   0.457215765882871
   0.640405560134302
```

On retrouve bien la convergence, ce vers manifestement la solution du système de Cramer $M \cdot X = B$. Restera à justifier ce résultat (en invoquant le théorème du point fixe) et à estimer l'erreur entre l'itéré X_N et l'unique point fixe de l'application affine $T : X \longmapsto U \cdot X + V$. Dans notre cas particulier, on remarque que, si l'on prend une norme dans $\mathbb{R}^n$, par exemple, la norme

$$\|x\|_\infty := \sup_j |x_j| \,, \; x = (x_1, ..., x_n)$$

(ici $n = 4$), on a

$$T(X) - T(Y) = U \cdot (X - Y) \,,$$

et donc

$$\|T(X) - T(Y)\|_\infty \leq \|U\|_\infty \, \|X - Y\|_\infty \,,$$

où

$$\|U\|_\infty = \sup_i \Big(\sum_j |u_{i,j}| \Big)$$

(voir la remarque 2.1).

On a constaté (voir la remarque 2.1) que le fait que M soit à diagonale dominante impliquait bien

$$\|U\|_\infty = \sup_i \Big(\frac{\sum_{j \neq i} |m_{i,j}|}{|m_{i,i}|} \Big) < 1 \,.$$

Nous sommes donc bien sous les conditions d'application du théorème du point fixe (théorème 4.1).

4.3 La notion de rayon spectral d'une matrice

Il est tentant, pour conforter l'intuition selon laquelle $T : X \longmapsto U \cdot X + V$ est strictement contractante, de chercher les valeurs propres de la matrice U, c'est-à-dire le spectre (*a priori* complexe) de la matrice U. La routine eig sous MATLAB (basée sur l'algorithme QR que nous verrons plus tard) fournit immédiatement ce spectre, dans notre cas en fait réel :

```
>> eig(U)

ans =

  -0.713530725646165
   0.165135250128611
   0.289325132808473
   0.259070342709081
```

Le maximum des modules des valeurs propres de U est appelé *rayon spectral* de U (noté $\rho(U)$) et c'est, on le verra, la condition $\rho(U) < 1$ qui est nécessaire et suffisante pour que $T : X \longmapsto U \cdot X + V$ soit une stricte contraction de $\mathbb{R}^n$ dans lui-même (ici $n = 4$), ce, bien sûr, quelque soit le vecteur V.

Une matrice $n \times n$ à entrées réelles A prise au hasard a une probabilité nulle de ne pas être telle que sont polynôme caractéristique soit scindé dans $\mathbb{C}[X]$, donc une probabilité nulle de ne pas être diagonalisable sur $\mathbb{C}$. Toujours avec une probabilité 1, l'algorithme itératif suivant (nous le justifierons), initié à n'importe vecteur X_0 de $\mathbb{R}^n$, fournit une suite convergeant (avec la rapidité exponentielle d'une suite géométrique) vers le rayon spectral $\rho(A)$ de la matrice A.

Proposition 4.1 *Pour presque toute matrice A à entrées réelles (resp. complexes), l'algorithme itératif initié à X_0 et régi ensuite par*

$$X_{k+1} = \frac{A \cdot X_k}{\|A \cdot X_k\|}, \ k \geq 0$$

(une norme ayant été arbitrairement choisie sur $\mathbb{R}^n$ (resp. $\mathbb{C}^n$)) est tel que

$$\lim_{k \to +\infty} \|A \cdot X_k\| = \rho(A)$$

et fournit donc un moyen numérique d'approcher le rayon spectral de A. La vitesse de convergence est de plus exponentielle.

Nous allons prouver cette proposition un peu plus loin. Mettons la en œuvre (par exemple sous MATLAB) avec ici comme norme sur $\mathbb{R}^n$ la norme euclidienne. Voici la routine donnant $\|A \cdot X_N\|$ au bout de N itérations, initiées à partir d'un vecteur arbitraire $X_0 = X$.

```
function r=rayonspectral1(A,X,N)

x=X;
for i=1:N
    y=A*x;
    x=y/norm(y);
end
   r=norm(y);
```

Si nous l'implémentons ici sur notre exemple à partir d'un vecteur X arbitraire, voici les résultats :

```
>> X= [1; -7; -4; 10]

X =

     1
    -7
    -4
    10

>> rayonspectral1(U,X,10)

ans =

   0.713428623400969

>> rayonspectral1(U,X,20)

ans =

   0.713530696402975

>> rayonspectral1(U,X,50)

ans =

   0.713530725646165

>> rayonspectral1(U,X,70)

ans =

   0.713530725646165
```

On observe la convergence (rapide,complète relativement à la précision possible certainement au delà de 50 itérations) de l'algorithme. On retrouve bien le rayon spectral de la matrice U.

4.4 Le pourquoi : la preuve du théorème du point fixe

C'est le fait que toute suite de Cauchy de nombres réels (donc aussi de vecteurs de nombres réels) dans $\mathbb{R}^n$ est convergente[2] qui soutend la preuve de ce résultat.

Que le point fixe soit unique est une évidence car s'il y avait deux ($T(Y_1) = Y_1$ et

[2]C'est un résultat que vous connaissez depuis le cours de MAT202 : toute suite de Cauchy de nombres réels est convergente dans $\mathbb{R}$; on rappelle qu'une suite $(X_k)_k$ de $\mathbb{R}^n$ est de Cauchy si et seulement si la distance entre deux points de la suite X_p et X_q peut être rendue arbitrairement petite pourvu que p et q soient assez grands. Pour passer du cas $n = 1$ au cas n quelconque, il convient de raisonner coordonnée par coordonnée.

$T(Y_2) = Y_2$ avec $Y_1 \neq Y_2$), on aurait

$$\|T(Y_1) - T(Y_2)\| = \|Y_1 - Y_2\| \leq \kappa \|Y_1 - Y_2\| \,,$$

ce qui est impossible si $\kappa < 1$. Si k est un entier strictement positif,

$$X_{k+1} - X_k = T(X_k) - T(X_{k-1}) \,,$$

donc

$$\|X_{k+1} - X_k\| \leq \kappa \|X_k - X_{k-1}\| \leq \cdots \leq \kappa^k \|X_1 - X_0\| \,.$$

Ainsi, si $p \geq 1$,

$$\|X_{k+p} - X_k\| \leq \sum_{l=k}^{k+p-1} \|X_{l+1} - X_l\| \leq \kappa^k (1 + \kappa + \cdots + \kappa^{p-1}) \|X_1 - X_0\| \leq \frac{\kappa^k}{1-\kappa} \|X_1 - X_0\|$$

puisque la série géométrique $[\kappa^k]_{k \geq 1}$ est convergente (de somme $1/(1-\kappa)$). La suite $(X_k)_{k \geq 0}$ est donc bien de Cauchy, donc converge vers un point Y, l'erreur d'approximation de Y par X_k au cran k de l'itération étant majorée par

$$\|X_k - Y\| \leq \kappa^k \frac{\|X_1 - X_0\|}{1 - \kappa} \,.$$

C'est une convergence très rapide (exponentielle puisque κ peut s'écrire $\kappa = e^{-u}$ avec $u > 0$). Comme T est continue (car contractant, même strictement, les distances), on déduit de $T(X_{k-1}) = X_k$ que $T(Y) = Y$, donc que Y est le point fixe de T. $\diamondsuit$

4.5 Le pourquoi (suite) ; normes de vecteurs et de matrices, conditionnement

On rappelle qu'une *norme* sur $\mathbb{R}^n$ est une application

$$X \in \mathbb{R}^n \longmapsto \|X\| \in [0, \infty[$$

telle que

$$\|\lambda X\| = |\lambda| \, \|X\| \,, \ \forall \, X \in \mathbb{R}^n \,, \ \forall \lambda > 0 \,,$$

que

$$\|X + Y\| \leq \|X\| + \|Y\| \quad \forall \, X, Y \in \mathbb{R}^n \,,$$

et qu'enfin

$$\|X\| = 0 \Longleftrightarrow X = 0 \,.$$

On le sait, deux normes sur $\mathbb{R}^n$ (par exemple $\| \ \|$ et $|\|| \ |\|$) sont toujours *équivalentes* au sens suivant : il existe deux constantes strictement positives K_1 et K_2 (dépendant bien sûr de ces deux normes) telles que

$$K_1 \|X\| \leq |\|X|\| \leq K_2 \|X\| \quad \forall \, X \in \mathbb{R}^n \,.$$

Cependant, changer de norme affecte la manière de quantifier la « taille » des vecteurs. Les normes les plus importantes sont :

– la norme euclidienne $\| \ \|_2$

$$\|(x_1, ..., x_n)\|_2 := (x_1^2 + \cdots + x_n^2)^{1/2} ,$$

importante car liée au produit scalaire dans $\mathbb{R}^n$, outil nous permettant d'y faire de la géométrie « à la Pythagore » ;

– la norme $\| \ \|_\infty$ définie par

$$\|(x_1, ..., x_n)\|_\infty := \sup_{j=1,...,n} |x_j| ;$$

– la norme $\| \ \|_1$ définie par

$$\|(x_1, ..., x_n)\|_1 := \sum_{j=1}^{n} |x_j| ;$$

– plus généralement, si p désigne un nombre réel supérieur ou égal à 1, la norme $\| \ \|_p$ définie par [3]

$$\|X\|_p := \Big(\sum_{j=1}^{n} |x_j|^p \Big)^{1/p} .$$

Ces normes se définissent de manière identique sur $\mathbb{C}^n$ en remplaçant la valeur absolue $|x|$ d'un nombre réel par le module $|z|$ d'un nombre complexe.

Faisons le choix d'une norme dans $\mathbb{R}^n$, celui de la norme euclidienne

$$\|(x_1, ..., x_n)\| := (x_1^2 + \cdots + x_n^2)^{1/2} .$$

Le choix d'une norme sur $\mathbb{R}^n$ induit le choix d'une norme sur les applications linéaires $L \ : \mathbb{R}^n \longmapsto \mathbb{R}^n$ par

$$\|L\| := \sup_{Y \in (\mathbb{R}^n)^*} \frac{\|L(Y)\|}{\|Y\|} .$$

On peut aussi envisager le point de vue matriciel et définir la norme d'une matrice $n \times n$ comme la norme de l'application linéaire que cette matrice représente. Si A est une matrice $n \times n$, on a donc

$$\|A\| := \sup_{Y \in (\mathbb{R}^n)^*} \frac{\|A \cdot Y\|}{\|Y\|} .$$

La norme de toute matrice représentant la même application linéaire que celle que représente A est donc égale à la norme de A. Bien sûr, cette définition de la norme d'une matrice (ou d'une application linéaire) dépend du choix de la norme qui a été fait dans $\mathbb{R}^n$ (ou $\mathbb{C}^n$ si l'on travaille avec des matrices de nombres complexes). On a fait ici le choix de la norme euclidienne dans $\mathbb{R}^n$ ou $\mathbb{C}^n$, mais on aurait pu faire un autre choix : par exemple, si l'on choisit la norme $\| \ \| = \| \ \|_\infty$, la norme $\|A\|_\infty$ correspondante d'une matrice A est

$$\|[a_{i,j}]\|_\infty = \sup_i \sum_{j=1}^{n} |a_{i,j}|$$

[3] Qu'il s'agisse d'une norme est ici moins immédiat ; l'inégalité triangulaire est une célèbre inégalité due au mathématicien russe, géomètre tant des espaces que des nombres, Hermann Minkowski (1864-1909).

(i indice de ligne, j indice de colonne) ; si par contre, on prend comme norme sur $\mathbb{R}^n$ la norme $\| \ \| = \| \ \|_1$, la norme $\|A\|_1$ correspondante d'une matrice A est

$$\|[a_{i,j}]\|_\infty = \sup_j \sum_i |a_{i,j}| \, .$$

C'est donc très différent ! On définit de même la norme d'une application linéaire de $\mathbb{R}^p$ dans $\mathbb{R}^q$ en équipant en général ces deux espaces de la même norme $\| \ \|$ à l'arrivée et au départ. Ce que nous avons dit pour les matrices carrées vaut donc aussi pour les matrices rectangulaires.

Étant données deux matrices carrées et un choix de norme dans $\mathbb{R}^n$, on a toujours :

$$\|A \cdot B\| \leq \|A\| \times \|B\|$$

(ce pour le choix de la norme dérivé du choix de norme qui a été fait dans $\mathbb{R}^n$ ou $\mathbb{C}^n$, disons par exemple ici la norme euclidienne, mais on peut très bien avoir fait un autre choix).

Revenons un instant à la résolution de notre système

$$M \cdot X = B \tag{4.2}$$

de la section 1.6. Notons $M + \Delta M$ la matrice M perturbée (Mperturb) et $X + \Delta X$ la solution du système (que l'on suppose toujours de Cramer, la perturbation étant assez petite pour que $\det(M + \Delta M) \neq 0$)

$$(M + \Delta M) \cdot (X + \Delta X) = B \tag{4.3}$$

(ici, on ne pertube pas B). En mettant ensemble les deux relations (4.2) et (4.3), on trouve immédiatement (par différence)

$$\Delta M \cdot X + M \cdot \Delta X + \Delta M \cdot \Delta X = 0 \, ,$$

ce que l'on réécrit

$$M \cdot \Delta X = -\Delta M \cdot (X + \Delta X) \, . \tag{4.4}$$

On peut transformer (4.4) en

$$\Delta X = -M^{-1} \cdot \Delta M \cdot (X + \Delta X)$$

et en déduire

$$\|\Delta X\| \leq \|M^{-1}\| \times \|\Delta M\| \times \|X + \Delta X\| \, ,$$

ce que l'on écrit (un peut artificiellement !)

$$\frac{\|\Delta X\|}{\|X + \Delta X\|} \leq \Big(\|M\| \times \|M^{-1}\| \Big) \times \frac{\|\Delta M\|}{\|M\|} \, ,$$

ou encore

$$\frac{\|(X + \Delta X) - X\|}{\|X + \Delta X\|} \leq \Big(\|M\| \times \|M^{-1}\| \Big) \times \frac{\|\Delta M\|}{\|M\|} \, .$$

Le membre de gauche

$$\frac{\|(X + \Delta X) - X\|}{\|X + \Delta X\|}$$

peut s'interpréter comme une « erreur relative »sur X tandis qu'au membre de droite, on voit apparaître

$$\frac{\|\Delta M\|}{\|M\|},$$

qui correspond (en norme) à l'erreur relative commise sur M. La quantité

$$\|M\| \times \|M^{-1}\|$$

qui « contrôle » en un certain sens la stabilité de la résolution du système est appelée à jouer un rôle majeur.

Définition 4.2 *Si M est une matrice carrée inversible de nombres réels ou complexes, on appelle conditionnement de M (relativement au choix d'une norme $\|\ \|$) la quantité $\|M\| \times \|M^{-1}\|$.*

Certes, le conditionnement d'une matrice $n \times n$ inversible (à entrées réelles ou complexes) dépend de la norme choisie sur $\mathbb{R}^n$ ou $\mathbb{C}^n$ mais, du fait de l'équivalence des normes, une matrice qui aura un grand condtionnement relativement à une norme en aura aussi un grand relativement au choix d'une autre norme.

C'est la taille du conditionnement qui est responsable pour les questions d'instabilité évoquées sur un exemple dans la section 1.6.

Ce serait un cercle vicieux que de prétendre calculer le conditionnement d'une matrice pour une certaine norme sur $\mathbb{R}^n$ en calculant A^{-1}, puis $\|A^{-1}\|$. En effet, si A est par un malencontreux hasard mal conditionnée, le calcul numérique de A^{-1} pose déjà problème ! Il faut donc trouver une parade pour calculer le conditionnement (relatif à une norme donnée sur $\mathbb{R}^n$) d'une matrice réelle inversible A de taille (n, n). Il en existe une lorsque la norme choisie sur $\mathbb{R}^n$ est la norme euclidienne $\|\ \|_2$ (induisant une norme notée ici $\|\ \|_2$ sur les matrices $n \times n$ réelles). Dans ce cas, si

$$\rho_1 \geq \rho_2 \geq \cdots \geq \rho_n > 0$$

sont les racines positives des valeurs propres de la matrice symétrique positive $A^t A$ (on les appele les *valeurs singulières* de A), on a

$$\mathrm{cond}_{\|\ \|_2}(A) = \frac{\rho_1}{\rho_n}.$$

Les valeurs singulières de A forment la diagonale de la matrice diagonale D impliquée dans la décomposition en valeurs singulières (« Singular Value Decomposition » dans la terminologie anglo-saxonne)

```
[U,D,V] = svd(A)
```

(ici sous l'environnement `MATLAB`). Les matrices U et V sont ici des matrices orthogonales réelles telles que

$$A = U \cdot D \cdot {}^tV$$

Pour obtenir cette décomposition, on part d'une diagonalisation de la matrice symétrique positive réelle $A \cdot {}^tA$ dans une base orthonormée :

$$\begin{aligned} A \cdot {}^tA &= U \cdot \mathrm{Diag}(\rho_1^2, ..., \rho_n^2) \cdot {}^tU \\ &= (U \cdot \mathrm{Diag}(\rho_1, ..., \rho_n)) \cdot {}^t(U \cdot \mathrm{Diag}(\rho_1, ..., \rho_n)). \end{aligned}$$

On pose ensuite

$$V = {}^tA \cdot \left({}^t(U \cdot \mathrm{diag}(\rho_1, ..., \rho_n)) \right)^{-1}$$

et l'on vérifie que V est orthogonale et que

$$A = U \cdot \mathrm{Diag}(\rho_1, ..., \rho_n) \cdot {}^tV$$

comme on le voulait.

4.6 Le pourquoi (suite) : où les valeurs propres entrent en jeu.

Soit A une matrice $n \times n$ à coefficients réels ou complexes.

Définition 4.3 *Le rayon spectral de la matrice A est par définition le maximum des modules des racines du polynôme caractéristique* $\det[A - XI_n]$ *dans* $\mathbb{C}$.

La propostion suivante sera pour nous capitale :

Proposition 4.2 *Si A est une matrice $n \times n$ à coefficients complexes de rayon spectral $\rho(A)$ et si $\| \ \|$ est une norme arbitraire sur $\mathbb{C}^n$, on a toujours*

$$\rho(A) \leq \|A\| .$$

Preuve. Prenons une valeur propre (par exemple λ_j) de module maximum et V un vecteur propre associé. On a

$$\|A \cdot V\| = \|\lambda_1 V\| = |\lambda_1| \times \|V\| \leq \|A\| \, \|V\|$$

par définition de la norme de A :

$$\|A\| = \sup_{X \in \mathbb{C}^n \setminus \{0\}} \frac{\|A \cdot X\|}{\|X\|} .$$

En divisant par $\|V\| \neq 0$, on trouve bien $\rho(A) \leq \|A\|$. $\diamondsuit$

Supposons le polynôme caractéristique de A scindé (c'est-à-dire ayant n racines distinctes). Ceci n'est de fait pas vraiment restrictif[4] car si l'on perturbe très légèrement aléatorement les coefficients de A, on peut toujours supposer qu'il en soit ainsi. Dans ce cas A est diagonalisable (sur $\mathbb{C}$) et il existe donc une base de $\mathbb{C}^n$ constituée de vecteurs propres $e_1, ..., e_n$ relatifs respectivement aux valeurs propres $\lambda_1, ..., \lambda_n$. Comme norme d'un vecteur sur $\mathbb{C}^n$, on peut envisager le sup des modules des coordonnées y_j du vecteur exprimé dans la base $(e_1, ..., e_n)$. Comme il s'agit d'une base de vecteurs propres

$$A \cdot (y_1 e_1 + \cdots + y_n e_n) = \lambda_1 y_1 e_1 + \cdots + \lambda_n y_n e_n .$$

Pour ce choix particulier de norme, on a

$$\|A \cdot (y_1 \cdot e_1 + \cdots + y_n \cdot e_n)\| = \sup_j |\lambda_j| \, |y_j| \leq \rho(A) \sup_j |y_j| \leq \rho(A) \, \|X\| .$$

[4] Du moins, pour les besoins du calcul scientifique.

Reprenons maintenant la démarche de Jacobi de la section 4.2 pour résoudre le système

$$M \cdot X = B$$

en écrivant (lorsque M est à diagonale dominante)

$$M = [m_{ij}]_{i,j} = D - E$$

(i indice de ligne, j indice de colonne) avec D la matrice diagonale et $D^{-1} \cdot E$ diagonalisable sur $\mathbb{C}$. Alors l'application affine

$$T \ : X \in \mathbb{R}^n \longmapsto U \cdot X + V \, , \ U = D^{-1} \cdot E \, , \ V = D^{-1} \cdot B \, ,$$

est strictement contractante : en effet $\rho(U) \leq \|U\|_\infty$ d'après la proposition 4.2. Or on a vu (remarque 2.1) que

$$\|U\|_\infty = \sup_i \left(\frac{\sum\limits_{j \neq i} |m_{i,j}|}{|m_{i,i}|} \right) < 1$$

Si l'on choisit maintenant comme norme particulière sur $\mathbb{R}^n$ la norme définie par

$$\|X\| = \|y_1 e_1 + \cdots + y_n e_n\| = \sup_j |y_j| \, ,$$

où $y_1, ..., y_n$ sont les coordonnées (complexes) du vecteur $(x_1, ..., x_n)$ exprimé dans la base (dans $\mathbb{C}^n$) de vecteurs propres $e_1, ..., e_n$ pour U [5], alors, on constate que

$$\|U\| = \rho(U) < 1 \, .$$

On a donc

$$\|T(X) - T(Y)\| \leq \rho(U) \|X - Y\|$$

avec $\kappa = \rho(U) < 1$ et il était donc licite que le théorème du point fixe s'applique (dans la section 4.2) et que notre algorithme converge.

En fait, on peut s'affranchir de l'hypothèse selon laquelle $D^{-1} \cdot E$ est diagonalisable et énoncer le résultat (que nous avons, nous, prouvé, sous une hypothèse supplémentaire de fait inutile) :

Proposition 4.3 *S'il existe un choix de norme sur $\mathbb{C}^n$ tel que, pour ce choix, la norme induite $\|D^{-1} \cdot E\|$ de la matrice $D^{-1} \cdot E$ soit strictement inférieure à 1 (ce qui est par exemple le cas si M est à diagonale dominante pour la norme $\| \ \|_\infty$), alors le système $M \cdot X = B$ est de Cramer et l'algorithme de Jacobi*

$$X_{k+1} = (D^{-1} \cdot E) \cdot X_k + D^{-1} \cdot B \, , \ k \in \mathbb{N} \, ,$$

initié en un vecteur X_0 quelconque de $\mathbb{R}^n$, converge vers l'unique solution X de ce système linéaire $M \cdot X = (D - E) \cdot X = B$, la vitesse de convergence étant exponentielle.

[5] Nous supposons ici U diagonalisable sur $\mathbb{C}$ pour simplifier (mais, on le verra, ce n'est pas réeellement nécessaire).

Outre l'algorithme proposé par C. G. Jacobi, un second algorithme itératif (toujours si les $a_{i,i}$ sont tous non nuls) peut être conduit : il consiste à écrire

$$M = T_{\text{inf}} - F\,,$$

où T_{inf} reprśente la matrice triangulaire inférieure extraite de M (diagonale incluse) et $-F$ cette fois la matrice triangulaire supérieure constituée des entrées strictement au dessus de la diagonale, les autres étant mises à 0. Par exemple

$$\begin{pmatrix} 5 & 2 & 2 \\ 1 & 6 & 3 \\ 3 & 4 & -8 \end{pmatrix} = \begin{pmatrix} 5 & 0 & 0 \\ 1 & 6 & 0 \\ 3 & 4 & -8 \end{pmatrix} - \begin{pmatrix} 0 & -2 & -2 \\ 0 & 0 & -3 \\ 0 & 0 & 0 \end{pmatrix}\,.$$

Cette méthode est la méthode de Gauss-Seidel[6]. Comme la méthode de Jacobi, on a la proposition suivante :

Proposition 4.4 *S'il existe un choix de norme sur $\mathbb{C}^n$ tel que, pour ce choix, la norme induite $\|T_{\text{inf}}^{-1} \cdot F\|$ de la matrice $T_{\text{inf}}^{-1} \cdot F$ soit strictement inférieure à 1 (ce qui est par exemple le cas si M est à diagonale dominante, pour la norme $\| \ \|_\infty$), alors le système $M \cdot X = B$ est de Cramer et l'algorithme de Gauss-Seidel*

$$X_{k+1} = (T_{\text{inf}}^{-1} \cdot F) \cdot X_k + T_{\text{inf}}^{-1} \cdot B\,,\ k \in \mathbb{N}\,,$$

initié en un vecteur X_0 quelconque de $\mathbb{R}^n$, converge vers l'unique solution X du système linéaire $M \cdot X = (T_{\text{inf}} - F) \cdot X = B$, la vitesse de convergence étant exponentielle.

L'intérêt de l'algorithme de Gauss-Seidel par rapport à celui de Jacobi réside dans le fait que l'encombrement mémoire (ainsi que le temps de calculs) nécessité par $T_{\text{inf}}^{-1} \cdot F$ est moindre que pour $D^{-1} \cdot E$ car E est (avec seulement *a priori* des zéros sur la diagonale) est « deux fois plus pleine » que F (elle, triangulaire supérieure stricte).

4.7 Le calcul du rayon spectral par une méthode itérative

Soit A une matrice à coefficients complexes, inversible et diagonalisable sur $\mathbb{C}$ (c'est le cas, avec une probabilité 1, pour une matrice dont les coefficients sont pris au hasard), telle que les valeurs propres (distinctes ou confondues) aient des modules s'organisant comme suit :

$$|\lambda_1| > |\lambda_{\mu+1}| \geq |\lambda_{\mu+2}| \geq ... \geq |\lambda_n| \geq 0$$

(c'est aussi le cas pour une matrice « générique » avec une probabilité 1). Soit $e_1, ..., e_n$ une base de vecteurs propres telle que $e_1, ..., e_\mu$ soit une base[7] du sous espace propre associé à λ_1. Soit V_0 un vecteur

$$X_0 = x_1 e_1 + \cdots + x_n e_n\,,$$

[6] Au nom de Gauss, se trouve ajouté celui Philipp Ludwig von Seidel (1821-1896), élève de Jacobi, astronome, géomètre et probabiliste allemand.

[7] De fait, si A est générique comme indiqué ici, nous pouvons assurer $\mu = 1$; nous donnerons toutefois la preuve de la proposition ce cas un peu plus général, où A est simplement supposée, outre le fait d'être diagonalisable, avoir une unique valeur propre, ici λ_1, de module maximal.

où l'un au moins des x_k, $k = 1, ..., \mu$, est non nul (c'est encore le cas d'un X_0 pris au hasard, avec une probabilité 1). Voici la proposition qui précise la proposition 4.1 dont les hypothèses n'avaient pas été explicitées. Nous sommes aussi maintenant en mesure de prouver cet énonçé résultant (encore une fois) du théorème du point fixe.

Proposition 4.5 (proposition 4.1 précisée) *Sous les hypothèses ci dessus (portant sur A et sur le choix de X_0), l'algorithme itératif initié à X_0 et régi ensuite par*

$$X_{k+1} = \frac{A \cdot X_k}{\|A \cdot X_k\|}\,,\ k \geq 0$$

(une norme sur $\mathbb{C}^n$ ayant été arbitrairement choisie) est tel que

$$\lim_{k \to +\infty} \|A \cdot X_k\| = |\lambda_1|$$

et fournit donc un moyen numérique d'approcher le rayon spectral de A. La vitesse de convergence est de plus exponentielle.

Le choix d'une norme doit être fait au préalable ; sous MATLAB, la norme que l'on choisit en priorité est la norme euclidienne (ou $\| \ \|_2$), mais l'on pourrait prendre aussi n'importe laquelle des normes $\| \ \|_p$, $p \in [1, \infty]$. Rappelons ici la routine correspondant à l'algorithme itératif très simple (déjà mentionné dans le cours) :

```
function r=rayonspectral1(A,X,k)

x=X;
for i=1:k
    y=A*x;
    x=y/norm(y);
end
   r=norm(y);
```

Preuve. Montrons d'abord par récurrence sur k que, pour tout $k \geq 1$,

$$X_k = \frac{A^k \cdot X_0}{\|A^k \cdot X_0\|}\,.$$

Ceci est vrai pour $k = 1$ par définition de X_1 et on a, pour $k \geq 1$,

$$X_{k+1} = \frac{A \cdot X_k}{\|A \cdot X_k\|} = A \cdot \Big(\frac{A^k \cdot X_0}{\|A^k \cdot X_0\|}\Big) \times \Big(\frac{\|A^{k+1} \cdot X_0\|}{\|A^k \cdot X_0\|}\Big)^{-1} = \frac{A^{k+1} \cdot X_0}{\|A^{k+1} \cdot X_0\|}\,,$$

ce qui prouve le résultat au cran $k+1$. Or

$$\begin{aligned} A^k \cdot X_0 &= \sum_{j=1}^{n} \lambda_j^k x_j e_j = \lambda_1^k \Big[\Big(\sum_{j=1}^{\mu} x_j e_j\Big) + \sum_{j=\mu+1}^{n} \Big(\frac{\lambda_j}{\lambda_1}\Big)^k x_j e_j\Big] \\ &= \lambda_1^k \Big(\sum_{j=1}^{\mu} x_j e_j + \vec{\epsilon}_k\Big) \end{aligned}$$

avec

$$\|\vec{\epsilon}_k\| < \|X_0\|(|\lambda_{\mu+1}|/|\lambda_1|)^k$$

On a donc

$$|\lambda_1|^k \left\| \sum_{j=1}^{\mu} x_j e_j \right\| (1-\eta_k) \leq \|A^k \cdot X_0\| \leq |\lambda_1|^k \left\| \sum_{j=1}^{\mu} x_j e_j \right\| (1+\eta_k) \tag{4.5}$$

avec

$$|\eta_k| < (|\lambda_{\mu+1}|/|\lambda_1|)^k \frac{\|X_0\|}{\left\| \sum_{j=1}^{\mu} x_j e_j \right\|} .$$

On a d'autre part

$$A \cdot X_k = \frac{\lambda_1^k (\lambda_1 \sum_{j=1}^{\mu} x_j e_j + A \cdot \vec{\epsilon}_k)}{\|A^k . X_0\|}$$

et, en prenant les normes

$$\frac{|\lambda_1|^{k+1} \left\| \sum_{j=1}^{\mu} x_j e_j \right\| (1-\tilde{\eta}_k)}{\|A^k . X_0\|} \leq \|A \cdot X_k\| \leq \frac{|\lambda_1|^{k+1} \left\| \sum_{j=1}^{\mu} x_j e_j \right\| (1+\tilde{\eta}_k)}{\|A^k . X_0\|} \tag{4.6}$$

avec

$$\tilde{\eta}_k \leq (|\lambda_{\mu+1}|/|\lambda_1|)^k \|A\| \frac{\|X_0\|}{|\lambda_1| \left\| \sum_{j=1}^{\mu} x_j e_j \right\|} .$$

On achève donc la preuve en combinant les encadrements (4.5) et (4.6). Comme $(|\lambda_{\mu+1}|/|\lambda_1|)^k$ (qui gouverne la décroissance vers 0 de η_k et $\tilde{\eta}_k$ est en $e^{-\rho k}$ avec $\rho > 0$ (puisque $|\lambda_{\mu+1}| < |\lambda_1|$), on a bien une vitesse exponentielle de convergence de $\|A \cdot X_k\|$ vers $|\lambda_1|$. $\diamondsuit$

L'algorithme présenté ci dessus permet (dans bien des cas) de calculer le rayon spectral d'une matrice et donc de tester si les algorithmes itératifs de Jacobi ou de Gauss-Seidel (lorsque les termes diagonaux de A sont tous non nuls) sont convergeant ou non pour tout choix de vecteur initial. Des méthodes plus élaborées permettent de calculer (pour une matrice réelle symétrique comme par exemple ${}^tA \cdot A$ lorsque A est réelle), les modules des autres valeurs propres $|\lambda_{\mu+1}|, ..., |\lambda_n|$, comme vous le verrez plus tard.

Notons pour terminer ici que l'algorithme ci-dessus fournit pour le rayon spectral d'une matrice A une approximation meilleure que celle que fournissent les algorithmes (basés sur la méthode QR que vous verrez plus tard) de recherche des vecteurs propres. Pour s'en convaincre, on a généré en cours une matrice aléatoire[8] A de taille 20×20 et on a calculer le spectre (c'est-à-dire la liste des valeurs propres) suivant la routine MATLAB « eig » , puis on a cherché de manière approchée le rayon spectral suivant l'algorithme itératif « rayonspecral1 »proposé plus haut. Voici les résultats :

```
>> A=rand(20,20);
>> eig(A)
```

[8]Les coefficients sont générés aléatoirement dans $[0, 1]$ suivant la loi de répartition uniforme sur $[0, 1]$, ce de manière indépendante les uns des autres.

```
ans =

 10.271312778421446
 -1.031285813538273 + 0.506241943544177i
 -1.031285813538273 - 0.506241943544177i
 -1.112001388367636 + 0.088837166100393i
 -1.112001388367636 - 0.088837166100393i
 -0.521656094147787 + 0.929049363231960i
 -0.521656094147787 - 0.929049363231960i
  1.179747238775334
  1.055919092604295 + 0.471484263061214i
  1.055919092604295 - 0.471484263061214i
  0.524701754063551 + 0.819471866612696i
  0.524701754063551 - 0.819471866612696i
  0.307110228375588 + 0.829276506915428i
  0.307110228375588 - 0.829276506915428i
 -0.153371702125838 + 0.657379179230264i
 -0.153371702125838 - 0.657379179230264i
 -0.494267101028723 + 0.225970680397249i
 -0.494267101028723 - 0.225970680397249i
  0.545885781424375
  0.119148059786589

>> X0=rand(20,1);
>> r=rayonspectral1(A,X0,100);
>> r

r =

  10.271312778421443

>> r=rayonspectral1(A,X0,500);
>> r

r =

  10.271312778421443
```

La dernière décimale proposée pour le rayon spectral approché est ici 3, tandis que 6 est la dernière décimale affichée pour la plus grande valeur propre. Notons ici que cette valeur propre, au vu du choix aléatoire particulier fait sur les coefficients de A (lois uniformes indépendantes X sur $[0,1]$) approche en fait $N \times E[X] = N/2$, où N est la taille de la matrice (ici $N = 20$) et $E[X]$ l'espérance (ou « moyenne » de la variable X), ici $E[X] = 1/2$ car la distribution de probabilité est uniforme sur $[0,1]$).

4.8 Un exemple « actuel » d'application du théorème du point fixe : Pagerank

L'algorithme Pagerank, brique de base de la version primitive du moteur de recherche Google sur la toile, constitue aussi une illustration du champ applicatif qu'entr'ouvre le théorème du point fixe[9].

L'ensemble des sites web peut être considéré comme l'ensemble E des sommets d'un *graphe orienté*. Il s'agit d'un ensemble de cardinal gigantesque (on évoque une trentaine de milliards), mais fini, et dont on peut indexer les éléments de 1 à N. Pour compléter la définition de graphe orienté, il faut ajouter la donnée d'un sous-ensemble fini V de $E \times E$, un couple de sommets (i, j) de $E \times E$ étant appelé une *arête* du graphe orienté (E, V) lorsque $(i, j) \in V$. Les arêtes du graphe orienté correspondant à la toile internet sont ici les liens $i \to j$ pointant d'une page i sur une page j ; $(i, j) \in V$ si et seulement si il existe un tel lien.

On associe à cette configuration une matrice $\mathbb{G}$ de taille $N \times N$ définie comme suit : l'entrée g_{ij} de $\mathbb{G}$ (i indice de ligne, j indice de colonne) est nulle si et seulement si il n'existe aucun lien pointant de i vers j (*i.e.* $(i, j) \notin V$). S'il existe un lien de i vers j, on définit g_{ij} comme l'inverse $1/L_i$ du nombre de liens de la page i vers une autre page web (dont, bien sûr, la page j, puisque $(i, j) \in V$ dans ce cas). Cette matrice $\mathbb{G}$ a l'intéressante propriété suivante : la somme des coefficients de chaque ligne vaut 1 ; de plus toutes les entrées de cette matrice sont positives ou nulles ; une telle matrice est dite *matrice stochastique* (on peut considérer chaque ligne comme une distribution de probabilité sur l'ensemble fini $\{1, ..., N\}$). On voit que 1 est valeur propre de cette matrice : en effet le vecteur colonne $\texttt{ones}(N, 1)$ dont toutes les coordonnées valent 1 est vecteur propre associé à la valeur propre 1. Si l'on choisit comme norme sur $\mathbb{R}^N$ la norme $\| \ \|_\infty$, on voit immédiatement que

$$\forall\, X \in \mathbb{R}^N,\ \|\mathbb{G} \cdot X\| \leq \|X\|,$$

donc $\|\mathbb{G}\| \leq 1$. De fait, on a même $\|\mathbb{G}\| = 1$ et le rayon spectral de $\mathbb{G}$ vaut 1.

Supposons un instant qu'aucune page du réseau ne soit une impasse, i.e. ne pointe sur aucune autre page. Si un robot dépourvu de la moindre capacité de discernement se déplace sur la toile (à partir d'un instant initial noté $k = 0$) et que l'on note $\mu_{k,j}$ la probabilité que notre robot se trouve sur la page j au clic k, on a

$$\mu_{k+1,i} = \sum_{j=1}^{N} \mathbb{P}\Big(\text{clic sur i} \,\Big|\, \text{le robot est en } j\Big)\, \mu_{k,j} \quad \forall\, i = 1, ..., N. \tag{4.7}$$

Comme le robot est idiot, il clique au hasard et la probabilité qu'il fasse son $(k+1)$-clic vers la page i depuis la page j (où il est arrivé après k clics) vaut $1/L_j = g_{ji}$. Les relations (4.7) se lisent matriciellement

$$[\mu_{k+1,1}, ..., \mu_{k+1,N}] = [\mu_{k,1}, ..., \mu_{k,N}] \cdot \mathbb{G}.$$

Pour prendre en compte le fait que certaines pages puissent se révéler être des impasses, on suppose qu'avec une probabilité $1 - \kappa$ (on prend couramment .15),

[9] On pourra aussi consulter avec profit (pour plus de détails) le texte rédigé par Michael Eisermann dont je me suis inspiré pour cette section :
http ://www.igt.uni-stuttgart.de/eiserm/enseignement/google.pdf

le robot, au moment de décider où aller au bout de k clics, décide d'aller avec la probabilité $1/N$ vers une page arbitraire du réseau. Ceci revient à modifier la matrice stochastique $\mathbb{G}$ en posant

$$\mathbb{G}_\kappa = \frac{1-\kappa}{N}\,\texttt{ones}(N,N) + \kappa\,\mathbb{G}\,,$$

où $\texttt{ones}(N,N)$ est la matrice $N \times N$ dont les entrées sont toutes des 1. Les relations matricielles deviennent

$$[\mu_{k+1,1},...,\mu_{k+1,N}] = [\mu_{k,1},...,\mu_{k,N}]\cdot\mathbb{G}_\kappa.$$

Une « mesure d'équilibre » $[\mu_1,...,\mu_N]$ (au seuil de tolérance $1-\kappa$) est par définition une distribution de probabilité sur $\{1,...,N\}$ telle que

$$[\mu_1,...,\mu_N] = [\mu_1,...,\mu_N]\cdot\mathbb{G}_\kappa.$$

En fait, on voit que ceci est équivalent à dire que $[\mu_1,...,\mu_N]$ (traité comme vecteur ligne) est un point fixe de l'application affine T_κ de $\mathbb{R}^N$ dans $\mathbb{R}^N$ (les vecteurs étant ici traités en ligne) définie par

$$T_\kappa\ : [x_1,...,x_N] \mapsto \frac{1-\kappa}{N}\,\texttt{ones}(1,N) + \kappa\,[x_1,...,x_N]\cdot\mathbb{G}.$$

Comme le rayon spectral de $\mathbb{G}$ vaut 1, cette application T_κ est $\kappa \simeq .85 < 1$- contractante et le théorème du point fixe assure l'existence et l'unicité de la « mesure d'équililibre » (au seuil de tolérance $1-\kappa$), en même temps que la possibilité d'approcher asymptotiquement (avec une erreur décroissant exponentiellement) cette « mesure d'équilibre » en partant d'une distribution de probabilité arbitraire

$$[\mu_{\text{init},1},...,\mu_{\text{init},N}] = \mu_{\text{init}}$$

(correspondant à $k=0$) et en considérant la démarche itérative

$$[\mu_{k+1,1},...,\mu_{k+1,N}] = T_\kappa([\mu_{k,1},...,\mu_{k,N}]),\quad k=0,1,2,...$$

Cette mesure d'équilibre (accessible *via* l'algorithme du point fixe) traduit les « poids » relatifs des diverses pages `web` sur la toile.

Évidemment, la matrice $\mathbb{G}$ est de taille colossale et demande à être réactualisée en permanence. D'où la difficulté de la maintenir stable le temps d'un calcul lourd (de par la complexité inhérente à la taille des matrices en jeu), donc forcément consommateur en temps ! Ce qui nous sauve cependant est l'aspect « creux » de la matrice (une page `web` pointe en moyenne seulement vers une dizaine de pages sur la toile, mais il faut encore beaucoup travailler pour passer du théorique à l'opérationnel[10]). Ce que nous venons de présenter correspond à une version primitive de l'algorithme `Pagerank`, pièce maitresse du dispositif du moteur de recherche `Google` élaboré vers 1997 par Serguey Brin et Larry Page à l'université de Stanford. On dispose ainsi d'une formidable application (combien actuelle !) du théorème du point fixe « en situation ».

[10] Voir l'article de Michael Eisermann mentionné précédemment.

Chapitre 5

Calcul numérique et équations différentielles

5.1 Une esquisse de théorie ; pourquoi le point fixe ?

Le $\mathbb{R}$-espace vectoriel des applications continues de $[a,b] \subset \mathbb{R}$ dans $[c,d] \subset \mathbb{R}$, équipé de la norme « uniforme »

$$\|f\|_\infty := \sup_{x\in[a,b]} |f(x)|$$

n'est plus un $\mathbb{R}$-espace vectoriel de dimension finie (comme $\mathbb{R}^n$) ; pourtant, on voit facilement qu'il s'agit d'un $\mathbb{R}$-espace vectoriel équipé d'une norme et dans lequel toute suite de Cauchy est convergente : en effet, si $(f_n)_{n\geq 0}$ est une telle suite de Cauchy, on a

$$\forall \epsilon > 0\,,\ \exists N(\epsilon) \in \mathbb{N}\,,\ \forall n,m \geq N(\epsilon)\,,\ \forall x \in [a,b]\,,\ |f_n(t) - f_m(t)| \leq \epsilon \qquad (5.1)$$

(attention à l'ordre des quantificateurs, c'est très important ici !). Pour chaque $x \in [a,b]$, la suite numérique $(f_n(x))_{n\geq 0}$ est de Cauchy dans $\mathbb{R}$, donc convergente vers un nombre réel $f(x) \in [c,d]$. En gelant n dans (5.1) et en faitant « courir » m vers $+\infty$, on voit que

$$\forall \epsilon > 0\,,\ \exists N(\epsilon) \in \mathbb{N}\,,\ \forall n \geq N(\epsilon)\,,\ \forall x \in [a,b]\,,\ |f_n(t) - f(t)| \leq \epsilon\,,$$

ce qui montre que la suite de fonctions $(f_n)_{n\geq 0}$ converge bien vers f dans l'espace des fonctions continues de $[a,b]$ dans $[c,d]$ équipé de la norme uniforme.

Nous allons nous intéresser dans cette section à la résolution sur un intervalle $[a,b]$ de $\mathbb{R}$, $f\ : [a,b] \times \mathbb{R} \longrightarrow \mathbb{R}$ désignant une fonction de deux variables x, y) continue en ces deux variables [1], de l'équation différentielle

$$\forall x \in [a,b]\,,\ y'(x) = f(x, y(x)) \qquad (5.2)$$

avec la condition initiale $y(a) = y_0$, c'est-à-dire à la recherche d'une fonction de classe C^1 sur $[a,b]$ [2] solution de l'équation différentielle du premier ordre (5.2).

[1] Voir le cours de MHT302 pour l'introduction au maniement des fonctions de plusieurs variables.

[2] Ceci signifie de classe C^1 sur $]a,b[$, avec en plus une dérivée à droite en a et en gauche en b de manière à ce que $x \longmapsto y'(x)$ soit continue sur $[a,b]$.

On peut voir cette recherche comme la recherche d'un point fixe d'une certaine application T de l'espace des fonctions continues de $[a,b]$ dans $\mathbb{R}$ dans lui-même :

$$h \longmapsto T(h)\,,$$

où

$$\forall x \in [a,b]\,,\ T(h)(x) := \int_a^x f(x,h(x))\,dx\,.$$

On voit en effet immédiatement que si $T(h) \equiv h$ (comme fonction), alors h est de classe C^1 sur $[a,b]$ et (à cause du théorème fondamental de l'analyse) que h est solution de l'équation différentielle (5.2). Le problème malheureuseusement ici est que l'on se place dans l'espace des fonctions continues de $[a,b]$ dans $\mathbb{R}$ alors qu'il faudrait se placer dans l'espace des fonctions continues de $[a,b]$ dans un intervalle fermé borné $[c,d]$ pour que notre théorème du point fixe s'applique.

Nous allons donc faire une hypothèse supplémentaire suivante sur f qui sera la suivante :

$$\exists K \geq 0\,,\ \forall x \in [a,b]\,,\ \forall y_1, y_2 \in \mathbb{R}\,,\ |f(x,y_1) - f(x,y_2)| \leq K|y_1 - y_2|\,. \tag{5.3}$$

On admettra ici que sous cette hypothèse, on a le résultat suivant :

Théorème 5.1 (théorème de Cauchy-Lipschitz [3]) *Soit f une fonction continue de $[a,b]\times\mathbb{R}$ dans $\mathbb{R}$, satisfaisant l'hypothèse (2.4) et y_0 un nombre réel ; il existe alors une unique solution y de l'équation différentielle*

$$\forall x \in [a,b]\,,\ y'(x) = f(x,y(x))$$

satisfaisant à la condition initiale $y(a) = y_0$.

Remarque 5.1. On connaît ce résultat depuis le cours de MIS101 lorsque $f(x,y) = \alpha(x)y + \beta(x)$, α et β étant des fonctions continues (cas particulier où l'hypothèse (5.3) est remplie). C'est le cas de équations dites « linéaires » du premier ordre. Si l'on connaît « explicitement » la solution après une double quadrature sous la forme

$$y(x) = \exp(A(x)) \times \Big(\int_a^x \beta(t)\exp(-\alpha(t))\,dt + y_0 \Big)$$

avec

$$A(x) := \int_a^x \alpha(t)\,dt\,,$$

on sait bien que cette solution est inexploitable (autrement que numériquement) si l'on ne dispose pas d'une primitive simple A de la fonction a, puis (pire encore !) ensuite d'une primitive de $t \longmapsto e^{-A(t)}\beta(t)$. Cependant, on sait que le théorème est bien valide dans ce cas particulier.

Ce théorème se prouve de proche en proche : on construit d'abord la solution sur un petit intervalle $[a,\epsilon]$ comme le point fixe d'une application strictement contractante de l'espace $\mathcal{C}([a,a+\epsilon],[y_a-\eta,y_a+\eta])$ dans lui-même (avec un choix judicieux – et couplé– de ϵ et η), puis on continue à partir de $a+\epsilon$ avec une nouvelle condition initiale $y_1 = y(a+\epsilon)$, *etc.* C'est sur ce principe qu'est fondée la méthode numérique due au Leonhard Euler [4] que nous allons présenter.

[3] Au nom du mathématicien français Augustin Cauchy (1789-1857) est ici associé celui de l'analyste allemand Rudolph Lipschitz (1832-1903), à qui l'on doit la mise en évidence de l'importance de la condition (5.3) ; une fonction satisfaisant la condition (5.3) est d'ailleurs appelée fonction *localement lipschitzienne* en la seconde variable y.

[4] Mathématicien suisse du siècle des lumières (1707-1783), qui marqua profondément et dans tous les domaines (analyse, arithmétique, astronomie,...) la pensée mathématique.

5.2 Le principe de la méthode

Fixons $p \in \mathbb{N}$, $N = 10^p$, posons $\tau_p = \frac{b-a}{10^p} = \frac{b-a}{N}$ et découpons $[a,b]$ avec la subdivision

$$a = x_{p,0} < x_{p,1} < \cdots < x_{p,N} = b\,,$$

où

$$x_{p,k} \ : a + k\tau_p \ , k = 0, 1, ..., N\,.$$

Nous allons expliter le fait que, pour h petit, on peut, si $x \longmapsto y(x)$ est une fonction dérivable au point x de $[a,b]$, assimiler

$$y'(x) \simeq \frac{y(x+h) - y(x)}{h}\,.$$

Cela nous suggère de définir de proche en proche la suite

$$\begin{aligned} y_{p,0} &= y_0 \\ \frac{y_{p,1} - y_{p,0}}{\tau_p} &= f(x_{p,0}, y_{p,0}) \\ \vdots &= \vdots \\ \frac{y_{p,k+1} - y_{p,k}}{\tau_p} &= f(x_{p,k}, y_{p,k}) \\ \vdots &= \vdots \\ \frac{y_{p,N} - y_{p,N-1}}{\tau_p} &= f(x_{p,N-1}, y_{p,N-1})\,. \end{aligned}$$

La relation de récurrence (à un pas) est donc

$$y_{p,k+1} = y_{p,k} + \tau_p f(x_{p,k}, y_{p,k})\,,\ k = 0, ..., N-1 \tag{5.4}$$

et le processus est initialisé avec $y_{p,0} = y_0$.

Nous allons montrer plus loin que si y_p désigne la fonction affine par morceaux sur $[a,b]$ et dont le graphe interpole les $N+1$ points $(x_{p,k}, y_{p,k})$, $k = 0, ..., N$, alors la suite de fonctions $(y_p)_{p\geq 0}$ converge uniformément sur $[a,b]$ vers la solution de notre problème, c'est-à-dire l'unique fonction $y \ : [a,b] \longmapsto \mathbb{R}$ solution de l'équation différentielle $y' = f(x,y)$ et satisfaisant à la condition initiale $y(a) = y_0$.

Voici un exemple de routine pour calculer les nombres $y_{p,k}$, $k = 0, ..., N$ lorsque p est fixé. Cette fois (au contraire de ce que l'on faisait dans les méthodes itératives précédentes), il ne faut plus « écraser » la valeur de $y_{p,k}$ avec celle de $y_{p,k+1}$ mais stocker toutes ces valeurs pour afficher en suite les points

$$(x_{p,k}, y_{p,k}),\ k = 0, ..., N\,,$$

donc le graphe de la fonction y_p (qui va nous donner une approximation de y). Cela tient au fait que l'on travaille avec une méthode reposant sur le théorème du point fixe, mais dans un espace de fonctions cette fois, et non plus dans $\mathbb{R}$ ou $\mathbb{R}^n$!

Voici pour illustrer cette démarche deux exemples.

- Le premier (exemple 5.1) est celui de l'équation différentielle $y' = y$ avec la condition initiale $y(0) = 1$, dont on connaît bien la solution, à savoir la fonction $x \longmapsto \exp(x)$ sur $\mathbb{R}$. Voici la routine utilisée dans le cours.

```
tau=1/N;
x=a:tau:b;
M=length(x);
y=1;
yy=1;
for i=1:M-1
    yy= yy + tau*yy;
    y=[y yy];
end
plot(x,y)
```

– Le second exemple (exemple 5.2) est celui d'une équation linéaire plus complexe (la résolution explicite par double quadrature est ici impossible car les primitives ne s'expriment pas en termes de fonctions simples)

$$y'(x) = \Big(1 - \rho\, x\, \cos(\alpha x^2)\sin(\beta x)\Big)\, y(x) + \cos(\gamma x)$$

avec la condition initiale $y(a) = y_0$, les nombres $\rho, \alpha, \beta, \gamma$ étant des paramètres. Voici la routine utilisée dans le cours :

```
tau=1/N;
x=a:tau:b;
M=length(x);
A=1-rho*x.*cos(alpha*x.^2).*sin(beta*x);
B=cos(gamma*x);
y=y0;
yy=y0;
for k=1:M-1
    yy = yy + tau*(yy*A(k)+ B(k));
    y = [y yy];
end
plot(x,y)
```

On constate immédiatement en utilisant la première routine que l'on approche bien le graphe de la fonction exponentielle (dès $N = 10$) sur $[0, 2]$ (voir la figure ci-dessous).

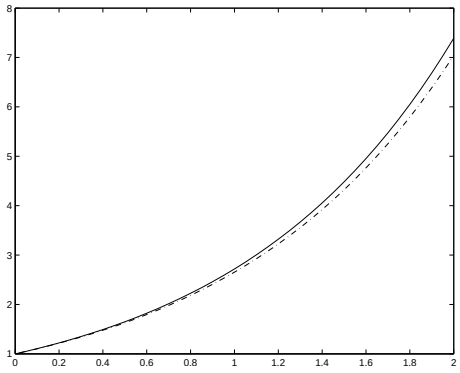

FIG. 5.1 – L'approximation de exp (exemple 5.1) par la méthode d'Euler sur $[0, 2]$

Nous avons testé le second exemple (exemple 5.2) avec les choix numériques suivants : $a = 0$, $b = 3$, $y_0 = 1$, $\rho = 3$, $\alpha = 2.36$, $\beta = 1.56$, $\gamma = 2.28$ et $N = 100$; voici sur

la figure ci-dessus le graphe de la fonction y_p correspondante (pour des valeurs de N entre 10 et 1000). On constate bien sur la figure l'approximation uniforme de la solution.

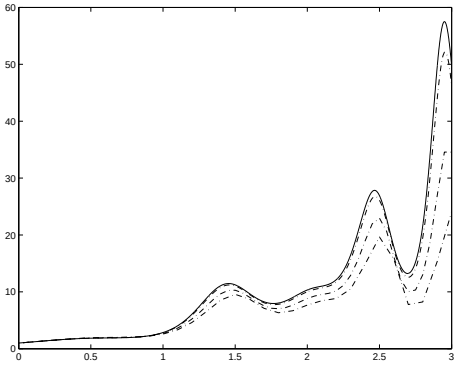

FIG. 5.2 – Illustration de l'approximation par la méthode d'Euler (exemple 5.2)

5.3 La méthode est d'ordre 1

Nous allons estimer en fonction de τ_p l'erreur commise entre $y_{p,k}$, $k = 0, ..., N$ et $y(a + k\tau_p)$, où $y : x \longmapsto y(x)$ est LA solution de classe C^1 sur $[a, b]$ de l'équation différentielle

$$y'(x) = f(x, y(x)) \,, \ \forall x \in [a, b]$$

assujettie à la condition initiale

$$y(a) = y_0 = y_{p,0} \,, \ \forall p \geq 1 \,.$$

(une telle solution existe en vertu du théorème 5.1). Nous ferons pour cela l'hypothèse supplémentaire que cette solution y est de classe C^2 sur $[a, b]$[5]. D'après la formule de Taylor-Lagrange à l'ordre 1[6], on peut écrire, pour tout $k = 0, ..., N - 1$:

$$y(x_{p,k+1}) = y(x_{p,k}) + \tau_p f(x_{p,k}) + \frac{\tau_p^2}{2} y''(\xi_{p,k}) \,, \tag{5.5}$$

où $\xi_{p,k} \in]x_{p,k}, x_{p,k+1}[$. On écrit donc l'une sous l'autre les deux relations (5.4) et (5.5), soit :

$$\begin{aligned} y_{p,k+1} &= y_{p,k} + \tau_p f(x_{p,k}, y_{p,k}) \,, \ k = 0, ..., N - 1 \\ y(x_{p,k+1}) &= y(x_{p,k}) + \tau_p f(x_{p,k}) + \frac{\tau_p^2}{2} y''(\xi_{p,k}) \,, \ k = 0, ..., N - 1 \,. \end{aligned}$$

Si l'on note

$$e_{p,k} := y_{p,k} - y(x_{p,k}) \,, \ k = 0, ..., N \,,$$

[5]Cette hypothèse est automatiquement vérifiée si la fonction f est de classe C^1 sur $[a, b] \times \mathbb{R}$ d'après le théorème sur la dérivation des fonctions composées ; on a d'ailleurs dans ce cas, d'après la règle de Leibniz :

$$y''(x) = \frac{d}{dx}[f(x, y(x))] = \frac{\partial f}{\partial x}(x, y(x)) + y'(x)\frac{\partial f}{\partial y}(x, y(x)) = \Big(\frac{\partial f}{\partial x} + f\frac{\partial f}{\partial y}\Big)(x, y(x)) \,.$$

[6]Il suffirait d'ailleurs que y soit deux fois dérivable sur $]a, b[$ pour appliquer cette formule.

on obtient en soustrayant ces deux relations

$$e_{p,k+1} = e_{p,k} + \tau_p(f(x_{p,k}, y_{p,k}) - f(x_{p,k}, y(x_{p,k})) + \frac{\tau_p^2}{2}\, y''(\xi_{p,k})\,,\ k = 0, ..., N-1\,.$$

En prenant les valeurs absolues et en utilisant la condition (5.3) sur f, il vient

$$|e_{p,k+1}| \leq |e_{p,k}| \times (1 + K\tau_p) + \frac{\tau_p^2}{2}\, M\,,$$

où

$$M := \sup_{[a,b]} |y''|\,.$$

Par récurrence sur k, on en déduit, puisque $e_{p,0} = 0$,

$$\begin{aligned} |e_{p,k}| &\leq M\frac{\tau_p^2}{2}\,\frac{(1+K\tau_p)^k - 1}{1 + K\tau_p - 1} \\ &\leq \frac{M\Big[(1+K\tau_p)^k - 1\Big]}{2K} \times \tau_p \\ &\leq \frac{M(e^{kK\tau_p} - 1)}{2K}\,\tau_p \\ &\leq \frac{M(e^{K(b-a)} - 1)}{2K}\,\tau_p\,,\ k = 0, ..., N-1\,. \end{aligned}$$

Du fait qu'il existe (pourvu que y soit C^2 ou au moins deux fois dérivable sur $]a, b[$ et de dérivée seconde bornée) une constante $C > 0$ telle que

$$|y_{p,k} - y(x_{p,k})| \leq C\tau_p\,,\ k = 0, ..., N-1,\ p \in \mathbb{N}\,, \tag{5.6}$$

on dit que la méthode d'Euler est *une méthode à un pas d'ordre* 1.

5.4 Un exemple de méthode à un pas d'ordre 2 : Euler modifiée

L'ordre 1 n'étant en général pas suffisant pour le contrôle d'erreur dans les problèmes pratiques (la constante C dans (5.6) pouvant être très grande et non « absorbée » par τ), nous allons modifier la méthode d'Euler pour générer une méthode d'ordre $\rho = 2$, c'est à dire telle qu'il puisse exister une constante $\widetilde{C}$ telle que

$$|\tilde{e}_{p,k}| = |\tilde{y}_{p,k} - y(x_{p,k})| \leq \widetilde{C}_\rho \times \tau^\rho$$

(les $\tilde{y}_{p,k}$, $k = 0, ..., N$, étant les nouvelles valeurs « approchées » aux points $x_{p,k}$, $k = 0, ..., N$, générées par cette nouvelle méthode, toujours bien sûr en prenant en compte la condition initiale $\tilde{y}_{p,0} = y_0$, soit $\tilde{e}_{p,0} = 0$).

Voici cette méthode, dite d'*Euler modifiée*. Elle consiste à définir les valeurs approchées $\tilde{y}_{p,k}$, $k = 0, ..., N$, de proche en proche avec la relation récurrente :

$$\tilde{y}_{p,k+1} = \tilde{y}_{p,k} + \tau_p f\Big(x_{p,k} + \frac{\tau_p}{2}\,,\, \tilde{y}_{p,k} + \frac{\tau_p}{2} f(x_{p,k}, \tilde{y}_{p,k})\Big)\,,\ k = 0, ..., N-1\,, \tag{5.7}$$

toujours à partir de la condition initiale $\tilde{y}_{p,0} = y_0$. Si l'on introduit la fonction

$$(x, y, h) \in [a, b] \times \mathbb{R}^2 \longmapsto F(x, y, h) = f\Big(x + \frac{h}{2}, y + \frac{h}{2} f(x, y)\Big)\,,$$

la relation (5.7) s'écrit sous forme abrégée :

$$\tilde{y}_{p,k+1} = \tilde{y}_{p,k} + \tau_p\, F(x_{p,k}, \tilde{y}_{p,k}, \tau_p)\,. \tag{5.8}$$

On note que si $x \in [a, b]$ et $h \in [0, 1]$

$$|F(x, y_1, h) - F(x, y_2, h)| \leq K(1 + K/2)|y_1 - y_2| = \tilde{K}|y_1 - y_2|\,. \tag{5.9}$$

On va supposer f de classe C^2 sur $[a, b] \times \mathbb{R}$, ce qui implique, puisque

$$y'(x) = f(x, y(x))\,,$$

(y désigne toujours la solution de $y' = f(x, y)$ de classe C^1 sur $[a, , b]$ et valant y_0 en a, dont l'existence est assurée par le théorème 5.1), que y' est de classe C^2, donc y de classe C^3. Il existe donc une constante M_1 telle que

$$\forall\, \xi \in [a, b]\,,\ |y'''(\xi)| \leq M_1\,.$$

On suppose aussi que la norme $\|D^2 f(\xi, \eta)\|$ est majorée par une constante M_2 sur $[a, b] \times [-2m, 2m]$, où $m := \sup_{[a,b]}(|y| + |y'|)$.

Si $h \in [0, 1]$ et $[x, x + h] \subset [a, b]$, on peut écrire cette fois la formule de Taylor-Lagrange à l'ordre 2 pour y en x et en déduire que

$$\Big|y(x + h) - y(x) - f(x, y(x))h - \frac{(f'_x + f f'_y)(x, y(x))}{2} h^2\Big| \leq \frac{M_1}{6}\, h^3$$

puisqu'on rappelle que

$$\begin{aligned} y''(x) &= \frac{d}{dx}[f(x, y(x))] \\ &= \frac{\partial f}{\partial x}(x, y(x)) + y'(x)\frac{\partial f}{\partial y}(x, y(x)) = \Big(\frac{\partial f}{\partial x} + f\frac{\partial f}{\partial y}\Big)(x, y(x))\,, \end{aligned}$$

ce que l'on écrit en abrégé

$$y''(x) = (f'_x + f f'_y)(x, y(x))\,.$$

En divisant par h, il vient

$$\Big|\frac{y(x + h) - y(x)}{h} - f(x, y(x)) - \frac{h}{2}(f'_x + f f'_y)(x, y(x))\Big| \leq \frac{M_1}{6}\, h^2\,.$$

On constate aussi, en utilisant la formule de Taylor Lagrange à l'ordre 1 au point $(x, y(x))$ pour la fonction f (de deux variables cette fois) que, toujours si $[x, x + h]$ est inclus dans $[a, b]$, on a

$$\begin{aligned} &\Big|f\Big(x + \frac{h}{2}, y(x) + \frac{h}{2}\, f(x, y(x))\Big) - f(x, y(x)) - \frac{h}{2}(f'_x + f f'_y)(x, y(x))\Big| \\ &\qquad \leq \frac{M_2}{8}\, h^2(1 + m^2)\,. \end{aligned} \tag{5.10}$$

En combinant les relations (5.8) et (5.10), on a donc

$$\Big|\frac{y(x+h)-y(x)}{h} - F(x,y(x),h)\Big| \leq Ch^2 \tag{5.11}$$

pour une certaine constante $C > 0$.

Ecrivons maintenant, pour $k = 0, ..., N-1$,

$$\begin{aligned} y(x_{p,k+1}) &= y(x_{p,k}) + \tau_p\Big(\frac{y(x_{p,k+1})-y(x_{p,k})}{\tau_p} - F(x_{p,k}, y(x_{p,k}), \tau_p\Big) \\ &\quad +\tau_p\, F(x_{p,k}, y(x_{p,k}), \tau_p) \end{aligned} \tag{5.12}$$

et soustrayons cette relation de la relation (5.8). On obtient, si l'on pose

$$\tilde{e}_{p,k} = \tilde{y}_{p,k} - y(x_{p,k})\,,\ k = 0, ..., N-1\,,$$

$$\begin{aligned} \tilde{e}_{p,k+1} &= \tilde{e}_{p,k} + \tau_p\Big(F(x_{p,k}, \tilde{y}_{p,k}, \tau_p) - F(x_{p,k}, y(x_{p,k}), \tau_p)\Big) \\ &+\tau_p\Big[\frac{y(x_{p,k+1})-y(x_{p,k})}{\tau_p} - F(x_{p,k}, y(x_{p,k}), \tau_p)\Big]\,. \end{aligned}$$

En utilisant (5.11) avec $x = x_{p,k}$ et $h = \tau_p$ ainsi que (5.9), on déduit, pour $k = 0, ..., N-1$,

$$|e_{p,k+1}| \leq |e_{p,k}|(1 + \widetilde{K}\tau_p) + C\tau_p^3\,.$$

En raisonnant comme pour la méthode d'Euler classique, il en résulte puisque l'on a au cran initial $\tilde{e}_{p,0} = 0$,

$$|\tilde{e}_{p,k}| \leq C\frac{e^{\widetilde{K}(b-a)}-1}{\widetilde{K}}\,\tau_p^2\,,\ k = 0, ..., N-1\,.$$

ce qui prouve bien que la méthode est d'ordre cette fois 2.

5.5 Pour aller plus loin ...

Les schémas numériques que nous avons proposé ici pour résoudre $y' = f(x,y)$ avec la condition initiale $y(a) = y_0$ étaient des schémas *explicites*, au sens où la relation de récurrence proposée pour modéliser l'équation différentielle, à savoir ici

$$y_{p,k+1} = y_{p,k} + \tau_p f(x_{p,k}, y_{p,k})$$

dans le cas d'Euler ou

$$y_{p,k+1} = y_{p,k} + \tau_p F(x_{p,k}, y_{p,k}, \tau_p)$$

avec

$$F(x,y,h) := f(x + h/2, y + h/2 f(x,y))$$

pour Euler modifiée, permettait d'exprimer <u>explicitement</u> $y_{p,k+1}$ en fonction de $y_{p,k}$; si l'on avait (par exemple dans le schéma d'Euler) modélisé la dérivée $y'(t_{p,k})$ comme une dérivée à gauche et non plus à droite, on aurait eu le schéma

$$y_{p,k} = y_{p,k-1} + \tau_p f(x_{p,k}, y_{p,k})$$

qui, lui, est un schéma *implicite*, au sens où il faut résoudre une certaine équation en $y_{p,k}$, à savoir l'équation

$$y_{p,k} - \tau_p f(x_{p,k}, y_{p,k}) - y_{p,k-1} = 0$$

pour trouver $y_{p,k}$ à partir de $y_{p,k-1}$; ce dernier schéma conduit à la méthode dite d'*Euler rétrograde*, qui, elle, est donc une méthode implicite.

Ce qu'on a vu avec la méthode d'Euler modifiée est que, si f est de classe C^∞, dans une méthode explicite où le schéma numérique est

$$y_{p,k+1} = y_{p,k} + \tau_p F(x_{p,k}, y_{p,k}, \tau_p)\,,$$

la méthode est d'ordre q dès que les développements limités en $h = 0$ de

$$h \mapsto \frac{y(x+h) - y(x)}{h}$$

($x \mapsto y(x)$ étant la solution théorique du problème de Cauchy $y' = f(x, y(x))$ sur $[a, b]$,, $y(a) = y_0$) coïncide jusqu'à l'ordre $q - 1$ (inclus) avec celui de

$$x \mapsto F(x, y(x), h)\,.$$

On voit ainsi que $q = 1$ pour la méthode d'Euler, $q = 2$ pour la méthode d'Euler modifiée. On peut construire une méthode d'ordre 4 (dite de Runge-Kutta[7]) en prenant

$$F(x, y, h) = \frac{1}{6}(K_1(x, y, h) + 2K_2(x, y, h) + 2K_3(x, y, h) + K_4(x, y, h))$$

avec

$$\begin{aligned} K_1(x, y, h) &= f(x, y) \\ K_2(x, y, h) &= f\Big(x + \frac{h}{2}, y + \frac{h}{2}K_1(x, y, h)\Big) \\ K_3(x, y, h) &= f\Big(x + \frac{h}{2}, y + \frac{h}{2}K_2(x, y, h)\Big) \\ K_4(x, y) &= f(x + h, y + hK_3(x, y, h))\,. \end{aligned}$$

Pour avoir une idée de la méthode (sur un exemple plus simple), regardons comment on peut choisir a_1, a_2, p_1, p_2 pour qu'en posant

$$\Phi(x, y, h) = a_1 f(x, y) + a_2 f(x + p_1 h, y + p_2 h f(x, y))\,,$$

on obtienne une méthode du second ordre : le développement de Taylor à l'ordre 1 en h donne

$$\Phi(x, y(x), h) = (a_1 + a_2) f(x, y(x)) + a_2(p_1 f'_x(x, y(x)) + p_2 f(x, y(x)) f'_y(x, y(x)))h + O(h^2)\,.$$

Comme

$$\frac{y(x+h) - y(x)}{h} = f(x, y(x) + \frac{h}{2}(f'_x(x, y(x)) + f(x, y(x) f'_y(x, y(x)) + O(h^2)\,,$$

[7] Carl Runge, mathématicien et physicien allemand, 1856-1927 ; Martin Kutta, mathématicien allemand, 1867-1944, connu également pour ses travaux en aérodynamique.

les conditions d'égalité des deux développements à l'ordre $2-1=1$ sont $a_1+a_2=1$ et $p_1=p_2=1/(2a_2)$. On effectue un calcul identique pour Runge-Kutta en prenant F de la forme $a_1K_1+a_2K_2+a_3K_3+a_4K_4$ et en cherchant les conditions sur a_1, a_2, a_3, a_4 pour que les développements limités de $h \mapsto \Phi(x, y, h)$ et $h \mapsto (y(x+h)-y(x))/h$ coïncident à l'ordre 3 cette fois (et non plus 1). Les calculs sont bien plus compliqués mais l'idée est la même ! C'est une très bonne révision sur les développements limités et la dérivation des fonctions composées.

Pour aller (bien) au delà de ces notes de cours, brève initiation aux mathématiques « en situation » sous leurs divers aspects, on renvoie ici aux divers chapitres concernés de *Mathématiques L2*, Pearson Education, 2007, ainsi que de *Mathématiques appliquées L3*, Pearson Education, 2009 (à paraître), par exemple (concernant le chapitre 5 de ces notes) au chapitre 16 du premier ouvrage ainsi qu'au chapitre 3 du second ou (concernant cette fois les chapitres 1,2,3,4) aux chapitres 13 du premier et 1 et 2 du second ouvrage.

FIN