Chapitre 3

Interpolation

3.1 Introduction

Le problème de l'interpolation 1d est de déterminer une fonction [1] $g : E \subset \mathbb{R} \to \mathbb{R}$, à partir de m couples de points de la forme (x_i, y_i) de manière à respecter *les conditions d'interpolation* :

$$g(x_i) = y_i,\ i = 1, \ldots, m$$

g est prise dans une famille de fonctions facilement calculables comme des polynômes ou des splines (polynômes par morceaux) des fonctions trigonométriques, des fonctions rationnelles, etc. En général la famille de fonctions est définie à partir de m fonctions $\varphi_1, \ldots, \varphi_m$ données et g dépend linéairement de ces fonctions, soit $g \in Vect\{\varphi_1, \ldots, \varphi_m\}$ c'est à dire :

$$g = \alpha_1\varphi_1 + \ldots \alpha_m\varphi_m = \sum_{j=1}^{m} \alpha_j\varphi_j$$

Avec cette forme les conditions d'interpolation s'écrivent :

$$\begin{array}{rcrcl}
g(x_1) = y_1 & \iff & \alpha_1\varphi_1(x_1) + \alpha_2\varphi_2(x_1) + \ldots \alpha_m\varphi_m(x_1) & = & y_1 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
g(x_i) = y_i & \iff & \alpha_1\varphi_1(x_i) + \alpha_2\varphi_2(x_i) + \ldots \alpha_m\varphi_m(x_i) & = & y_i \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
g(x_m) = y_m & \iff & \alpha_1\varphi_1(x_m) + \alpha_2\varphi_2(x_m) + \ldots \alpha_m\varphi_m(x_m) & = & y_m
\end{array}$$

ce qui est en fait un système linéaire $M\alpha = y$ avec :

$$M = \begin{bmatrix} \varphi_1(x_1) & \varphi_2(x_1) & \ldots & \varphi_m(x_1) \\ \varphi_1(x_2) & \varphi_2(x_2) & \ldots & \varphi_m(x_2) \\ \vdots & \vdots & \vdots & \vdots \\ \varphi_1(x_m) & \varphi_2(x_m) & \ldots & \varphi_m(x_m) \end{bmatrix},\ y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \text{ et } \alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix}.$$

Vu sous cet angle, résoudre un problème d'interpolation revient à résoudre un système linéaire, en particulier il faut que la matrice précédente (parfois appelée, par extension du cas polynomial, *matrice de Vandermonde*) soit inversible. Une condition nécessaire pour cela est que les fonctions φ_j soient linéairement indépendantes, c'est à dire que :

$$\sum_{j=1}^{m} \alpha_j\varphi_j = 0(\iff \sum_{j=1}^{m} \alpha_j\varphi_j(x) = 0, \forall x \in E) \iff \alpha_j = 0, \forall j \in [\![1, m]\!]$$

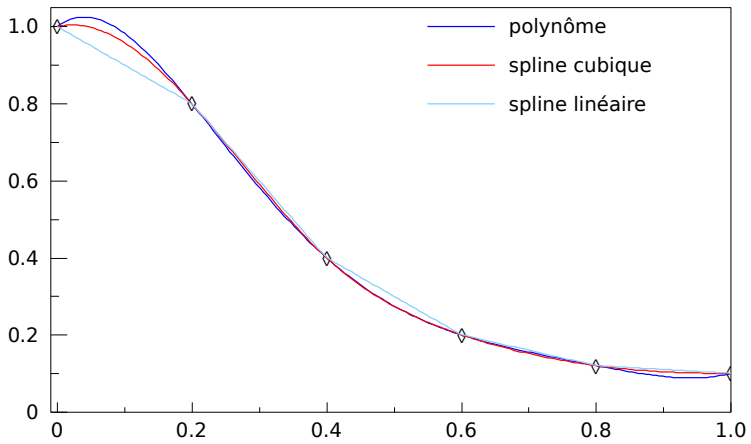
1. E est le plus souvent un intervalle $[a, b]$ et de plus $x_i \in E, \forall i$.

mais cette condition n'est pas suffisante pour que la matrice M soit bien inversible. L'aspect existence et unicité des coefficients α_j n'est qu'une partie des choses :

- Que choisit-on comme fonctions φ_j ?
- Peut-on jouer sur le choix des abscisses d'interpolation (leur nombre et leur position) ?
- Quel est le but du problème d'interpolation posé (on en parle tout de suite après) ? Et va-t-on l'atteindre avec les choix fait pour les fonctions φ_j, et (peut être) pour les x_i, etc.

Quelques besoins pratiques de l'interpolation

- Approcher très précisément des fonctions transcendantes comme sin, cos, arcsin, exp, log, etc. Vous êtes vous déjà demandé comment fait votre calculatrice ou les bibliothèques standards des langages de programmation pour les calculer ? Dans la plupart des cas cela finit, après certaines opérations [2], par le calcul d'un polynôme d'interpolation.
  Dans ce type d'application on peut en général choisir le nombre de points d'interpolation ainsi que la position des abscisses d'interpolation et le but visé est d'offrir à l'utilisateur une grande précision, en particulier si f est la fonction et p le polynôme qui l'approche, on aimerait que l'erreur relative maximale ne soit pas trop grande par rapport au epsilon machine :
  $$\forall x \in E : p(x) = f(x)(1 + \epsilon_x) \text{ avec } |\epsilon_x| \leq C\mathbf{u}$$
  avec par exemple $C \leq 10$.
  Si les fonctions transcendantes les plus utilisées ont déjà été codées et sont disponibles dans plusieurs bibliothèques, le procédé peut néanmoins être utile si vous devez calculer de nombreuses fois une fonction qui s'écrit à l'aide de telles fonctions élémentaires (comme sin, exp, etc.) et si le temps de calcul pour cette fonction est crucial (applications temps réel). On peut vouloir l'approcher directement par un polynôme d'interpolation bien choisi.
- Définir une fonction en utilisant des mesures expérimentales suffisamment précises [3]. Cela est utile si vous avez besoin d'estimer la fonction en d'autres points que les points de mesure. Voici un exemple où on a 6 points de mesure dont les abscisses sont dans l'intervalle $[0, 1]$. On veut donc définir une fonction sur l'intervale complet $[0, 1]$ par interpolation. L'exemple montre trois interpolants différents.



2. Exploitation de certaines propriétés de la fonction (symétries, périodicité, etc.) de manière à réduire l'intervalle d'approximation par un polynôme le plus possible.

3. Lorsque les mesures sont entachées d'erreur, on préfère utiliser d'autres principes d'approximation comme les moindres carrés.

Rmq : dans ce cas on ne dispose pas en général du choix des x_i et les abscisses sont souvent placées de manière régulière ($x_{i+1} - x_i = \Delta x$ constant).

Le même type de problème peut se poser en $2d$ (ou plus). Dans ce cas on dispose de points de la forme $((x_i, y_i), z_i)$ et on cherche $g : E = [a, b] \times [c, d] \to \mathbb{R}$ telle que $g(x_i, y_i) = z_i, i = 1, \ldots, m$.

De même l'espace but peut être aussi de dimension 2 (ou plus). On a alors affaire à autant de problèmes d'interpolation différents.

3.2 Interpolation polynomiale 1d

3.2.1 Fonctions polynomiales et base canonique

On note $\mathcal{P}_n$ l'ensemble des fonctions polynomiales de degré $\leq n$:

$$\mathcal{P}_n := \{f \in \mathcal{F}(\mathbb{R}, \mathbb{R}) : \exists (a_0, \ldots, a_n) \in \mathbb{R}^{n+1} \text{ avec } f(x) = a_0 + a_1 x + \ldots a_n x^n \ \forall x \in \mathbb{R}\}$$

En admettant que l'ensemble $\mathcal{F}(\mathbb{R}, \mathbb{R})$ des fonctions de $\mathbb{R}$ dans $\mathbb{R}$ est un espace vectoriel réel (ce qui est vrai) il est alors évident que $\mathcal{P}_n$ en est un sous-espace vectoriel [4] car $\forall p, q \in \mathcal{P}_n$ et $\forall \alpha, \beta \in \mathbb{R}$ il est clair que $\alpha p + \beta q \in \mathcal{P}_n$ (si on suppose que les "coefficients" de p sont les $(a_0, \ldots, a_n)$ et ceux de q les $(b_0, \ldots, b_n)$ alors les "coefficients" de $\alpha p + \beta q$ sont les $(\alpha a_0 + \beta b_0, \ldots, \alpha a_n + \beta b_n)$).

On voit aussi que $\forall p \in \mathcal{P}_n$, p peut s'écrire :

$$p = \sum_{i=0}^{n} a_i e_i \text{ avec } e_i : x \mapsto x^i$$

où $\mathcal{E}_n := (e_0, \ldots, e_n)$ forme la base canonique de $\mathcal{P}_n$. C'est la plus connue mais cette base ne se prête pas bien aux calculs avec les flottants (dès que n est un peu grand (disons $n > 3$)) et d'autres bases sont utilisées.

En fait avec la définition donnée ci-avant $\mathcal{E}_n$ apparait seulement comme une famille génératrice de $\mathcal{P}_n$. Pour qu'elle soit une base il faut montrer qu'elle est libre : est-ce que $\sum_{i=0}^{n} a_i e^i = 0 \Rightarrow a_0 = \cdots = a_n = 0$? Réponse : oui. En effet :

$$\sum_{i=0}^{n} a_i e^i = 0 \iff \sum_{i=0}^{n} a_i e^i(x) = \sum_{i=0}^{n} a_i x^i = 0 \ \forall x \in \mathbb{R}$$

En prenant $x = 0$ on obtient donc que $a_0 = 0$. Puis en dérivant on obtient que :

$$\sum_{i=1}^{n} i a_i x^{i-1} = 0 \ \forall x \in \mathbb{R}$$

De nouveau en prenant $x = 0$ on obtient que $a_1 = 0$, puis en réitérant cette astuce on obtient que tous les coefficients sont nuls. Conclusion : $\mathcal{E}_n$ est bien une base de $\mathcal{P}_n$ et ce dernier est donc un espace vectoriel réel de dimension $n + 1$.

4. Par contre l'ensemble des fonctions polynômes de degré exactement n (ie $a_n \neq 0$) n'est pas un espace vectoriel.

3.2.2 Solution du problème d'interpolation dans la base canonique

Calcul des coefficients

On se donne donc $n+1$ points (x_i, y_i) avec des abscisses toutes distinctes ($x_i \neq x_j$ si $i \neq j$) et on cherche un polynôme p de $\mathcal{P}_n$ vérifiant les $n+1$ conditions d'interpolation :

$$p(x_i) = y_i, \quad i \in [\![0, n]\!]$$

Si on cherche p dans la base canonique on obtient donc les $n+1$ équations linéaires :

$$a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n = y_i, \quad i \in [\![0, n]\!]$$

qui constituent un système linéaire dont les inconnues sont les coefficients a_j :

$$Va = y \text{ avec } V = \begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & x_i & x_i^2 & \dots & x_i^n \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix}, \; a = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_i \\ \vdots \\ a_n \end{bmatrix} \text{ et } y = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_i \\ \vdots \\ y_n \end{bmatrix}.$$

On a alors existence et unicité si la matrice V est inversible. On peut montrer que le déterminant d'une telle matrice (dite de "Vandermonde") est égal à :

$$det(V) = \prod_{0 \leq i < j \leq n} (x_j - x_i)$$

et une telle matrice est donc inversible avec la condition des abscisses toutes distinctes. Cependant on peut aussi montrer que le conditionnement de cette matrice croît très rapidement avec n et ne convient donc pas en arithmétique flottante. D'autre part, si on utilise un algorithme standard comme Gauss, le coût du calcul des coefficients est de $\frac{1}{3}(n+1)^3$.

Evaluation d'un polynôme en un point

Un algorithme pour évaluer p en x consisterait à écrire :

```
px ← a_0
pour i de 1 à n
    xi ← x^i
    px ← px + a_i × xi
```

mais il faut préciser comment calculer x^i... On a l'algorithme de la puissance mais ici comme on a besoin de toutes les puissances de x il suffit d'utiliser $x^i = x \times x^{i-1}$ ce qui donne finalement :

```
eval_poly(x, [a_0, ..., a_n]) :
    px ← a_0
    xi ← 1
    pour i de 1 à n
        xi ← xi × x
        px ← px + a_i × xi
    retourner px
```

La complexité de cet algorithme est $2n$ multiplications et n additions. Or on peut faire mieux en utilisant le parenthésage suivant, explicité ci-dessous pour un polynôme de degré 4 :

$$p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 = a_0 + x(a_1 + x(a_2 + x(a_3 + x(a_4))))$$

ce qui conduit aux opérations algorithmiques suivantes :

$px \leftarrow a_4$
$px \leftarrow a_3 + x \times px$
$px \leftarrow a_2 + x \times px$
$px \leftarrow a_1 + x \times px$
$px \leftarrow a_0 + x \times px$

et en utilisant une boucle et en généralisant à un polynôme de degré inférieur ou égal à n, on obtient l'algorithme d'*Horner* :

eval_poly_horner$(x, [a_0, \ldots, a_n])$:
    $px \leftarrow a_n$
    pour i de $n-1$ à 0 par pas de -1
        $px \leftarrow a_i + x \times px$
    retourner px

dont la complexité est de n multiplications et n additions, on a donc gagné n multiplications.

3.2.3 Solution du problème d'interpolation dans la base de Lagrange

La base de Lagrange associée aux abscisses $x_0, \ldots, x_n$ est constituée des $n+1$ polynômes (de degré exactement n) :

$$\mathcal{L}_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right), \ i \in [\![0, n]\!]$$

On montre facilement qu'ils possèdent la propriété suivante :

$$\mathcal{L}_i(x_j) = \delta_{i,j} = \begin{cases} 1 & \text{pour } i = j \\ 0 & \text{pour } i \neq j \end{cases}$$

Il est assez facile de montrer que $(\mathcal{L}_0, \ldots, \mathcal{L}_n)$ est bien une base de $\mathcal{P}_n$ car c'est une famille libre [5]. En effet :

$$\sum_{i=0}^{n} \alpha_i \mathcal{L}_i = 0 \iff \sum_{i=0}^{n} \alpha_i \mathcal{L}_i(x) = 0, \forall x \in \mathbb{R}$$

et en prenant $x = x_j$ on obtient que $\alpha_j \underbrace{\mathcal{L}_j(x_j)}_{=1} = 0$, d'où $\alpha_j = 0$. Ainsi $\sum_{i=0}^{n} \alpha_i \mathcal{L}_i = 0 \Rightarrow \alpha_i = 0, \forall i \in [\![0, n]\!]$.

On peut maintenant répondre à la question posée :

Théorème 1 : *Il existe un unique polynôme $p \in \mathcal{P}_n$ tel que $p(x_i) = y_i, \forall i \in [\![0, n]\!]$ et :*

$$p = \sum_{i=0}^{n} y_i \mathcal{L}_i$$

5. Une famille libre à m vecteurs dans un e.v. de dimension m est aussi génératrice et est donc une base.

Preuve : Comme $\mathcal{L}_i \in \mathcal{P}_n \ \forall i$, il est clair que $p \in \mathcal{P}_n$ et p vérifie bien les $n+1$ conditions d'interpolation, en effet :

$$p(x_k) = \sum_{i=0}^{n} y_i \mathcal{L}_i(x_k) = \sum_{i=0}^{n} y_i \delta_{i,k} = y_k, \ \forall k \in [\![0, n]\!].$$

Finalement un tel polynôme est bien unique : tout polynôme q de $\mathcal{P}_n$ s'écrit dans la base de Lagrange associée aux x_i :

$$q = \sum_{i=0}^{n} q(x_i)\mathcal{L}_i$$

et donc si q vérifie les mêmes conditions d'interpolation ($q(x_i) = y_i$), il vient $q = p$ (deux vecteurs qui se décomposent de manière identique dans une même base sont égaux). □

Remarque : dans la bonne base de Lagrange (i.e. celle associée aux abscisses x_i), il n'y a pas de coût associé à l'obtention des coefficients du polynôme d'interpolation dans cette base puisque ce sont directement les y_i (la matrice du système linéaire correspondant est la matrice identité I_{n+1}) !

Evaluation d'un polynôme écrit dans la base de Lagrange

Cette partie est vue en TD. On remarque alors que si le coût calcul des coefficients est nul, celui d'une évaluation basique en un point est en $O(n^2)$. Malgré le désavantage de ce coût en $O(n^2)$ plus important que le coût d'évaluation dans la base canonique, on peut montrer que le calcul est assez stable en arithmétique flottante.

On peut obtenir une évaluation en $O(n)$ en factorisant certains termes [6] et aboutir à la formule barycentrique de Lagrange (cf TD). Cette formulation reste aussi stable en arithmétique flottante sous certaines conditions qui sont vérifiées lorsqu'on utilise les abscisses de Tchebychev (voir plus loin). La formule barycentrique permet aussi d'utiliser des facteurs d'échelle pour éviter les problèmes d'underflow et d'overflow.

Ainsi l'interpolation de Lagrange avec la formule barycentrique (et en utilisant les abscisses de Tchebichev) constitue une méthode très efficace et assez simple pour approcher des fonctions assez régulières sur un intervalle donné $[a, b]$.

3.2.4 Base de Newton

Une autre base que celle de Lagrange est aussi couramment utilisée, il s'agit de la base de Newton. Dans cette base (associée aux abscisses x_0, x_1, x_2, …, x_{n-1}) un polynôme de $\mathcal{P}_n$ s'écrit :

$$\begin{aligned} p(x) &= c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \cdots + c_n(x - x_0)(x - x_1)\dots(x - x_{n-1}) \\ &= c_0\mathcal{N}_0(x) + c_1\mathcal{N}_1(x) + c_2\mathcal{N}_2(x) + \cdots + c_n\mathcal{N}_n(x) \\ &= \sum_{i=0}^{n} c_i\mathcal{N}_i(x) \end{aligned}$$

où $\mathcal{N}_0(x) = 1$, $\mathcal{N}_1(x) = x - x_0$, $\mathcal{N}_2(x) = (x - x_0)(x - x_1)$, etc…, soit :

$$\begin{aligned} \mathcal{N}_i(x) &= \prod_{j=0}^{i-1}(x - x_j) \\ &= (x - x_{i-1})\mathcal{N}_{i-1}(x), \quad \text{si } i \geq 1. \end{aligned}$$

6. Cette factorisation est en $O(n^2)$ et est donc "amortie" si on évalue le polynôme en de nombreux points.

La base de newton comporte de certains avantages :

— Evaluation rapide en un point x : on peut généraliser l'algorithme d'Horner et évaluer très rapidement $p(x)$ mais aussi $p'(x)$, $p''(x)$, etc.

— On peut rajouter successivements des points d'interpolation sans refaire tous les calculs.

Néanmoins l'évaluation dans cette base est moins stable que dans celle de Lagrange et il faut se limiter à des polynômes de degré raisonnable.

3.2.5 Evaluation directe d'un polynôme d'interpolation

Lorsqu'un polynôme d'interpolation doit être évalué en un point (ou quelques points), on peut éviter de passer par le calcul des coefficients dans une base donnée en utilisant le résultat suivant :

Théorème 2 : *(Relation d'Aïtken) Soient α, β et $x_0, x_1, \ldots, x_{n-1}$, $n+2$ réels tous deux à deux distincts. On considère les deux polynômes d'interpolation q et r de degré $\leq n$ associés respectivement aux points :*

$$\{(x_0,\, y_0), \ldots, (x_{n-1},\, y_{n-1}), (\alpha,\, a)\} \text{ et } \{(x_0,\, y_0), \ldots, (x_{n-1},\, y_{n-1}), (\beta,\, b)\}$$

alors le polynôme d'interpolation p de degré $\leq n+1$ associé aux points :

$$\{(x_0,\, y_0), \ldots, (x_{n-1},\, y_{n-1}), (\alpha,\, a), (\beta,\, b)\}$$

est donné par :

$$p(x) = \frac{(\beta - x)q(x) - (\alpha - x)r(x)}{\beta - \alpha}.$$

Preuve : cf TD. A partir de cette relation on peut construire divers algorithmes (Aïtken, Neville,...) qui permettent une évaluation en $O(n^2)$ opérations (cf TD).

3.2.6 Convergence de l'interpolation

L'espace des fonctions continues sur un intervalle $[a, b]$

L'ensemble des fonctions continues sur un intervalle $[a, b]$ noté $C([a, b], \mathbb{R})$ est un espace vectoriel que l'on peut voir comme un s.e.v. de l'e.v. des fonctions de $[a, b]$ dans $\mathbb{R}$. En effet si f et g sont deux telles fonctions et α, β deux réels quelconques, alors la fonction $h := \alpha f + \beta g$ est aussi continue.

Comme pour tout e.v. il est interessant de le munir d'une norme et on peut montrer assez facilement que :

$$\|f\|_\infty := \max_{x \in [a,b]} |f(x)|$$

est bien une norme sur $C([a, b], \mathbb{R})$. De plus l'e.v.n. ainsi obtenu est complet : toute suite de Cauchy (f_k) de fonctions de $C([a, b], \mathbb{R})$ converge bien vers une fonction continue. Ainsi on pourra mesurer la distance entre deux fonctions de $C([a, b], \mathbb{R})$ par $d(f, g) := \|f - g\|_\infty$ et c'est ce que nous utiliserons dans la suite.

Remarque : on peut aussi montrer que :

$$\|f\|_2 := \left(\int_a^b (f(x))^2 \, dx \right)^{\frac{1}{2}}$$

est une norme mais $C([a,b],\mathbb{R})$ muni de celle-ci n'est pas complet.

Un autre résultat intéressant est que toute fonction de $C([a,b],\mathbb{R})$ peut être approchée d'aussi près que l'on veut par un polynôme que l'on peut même construire explicitement :

Théorème 3 : $\forall f \in C([0,1],\mathbb{R})$, la suite de polynômes $(B_n(f))$ définis par :

$$B_n(f)(x) := \sum_{j=0}^{n} f\left(\frac{j}{n}\right) \mathcal{B}_{n,j}(x), \text{ où } \mathcal{B}_{n,j}(x) = C_n^j x^j (1-x)^{n-j}$$

vérifie :

$$\lim_{k\to\infty} \|B_n(f) - f\|_\infty = 0$$

Preuve : admise. Remarques :

— Pour construire un polynôme qui approche une fonction de $C([a,b],\mathbb{R})$ il suffit d'utiliser la transformation affine bijective suivante :

$$\begin{array}{rcl} C([a,b],\mathbb{R}) & \to & C([0,1],\mathbb{R}) \\ g & \mapsto & f \text{ avec } f(x) := g(a(1-x)+bx), \forall x \in [0,1] \end{array}$$

— Le défaut de cette approche est que pour obtenir une précision ϵ (c'est à dire trouver le polynôme $B_n(f)$ vérifiant $\|B_n(f) - f\|_\infty \leq \epsilon$) il faut généralement que n soit grand, voire très grand et ceci même pour une fonction f très régulière. Ainsi ce beau résultat mathématique est décevant en pratique, du moins pour l'approximation de fonctions (voir remarque suivante).

— Pour n fixé, les $n+1$ polynômes $\mathcal{B}_{n,j}, j = 0,\ldots,n$ forment une base de $\mathcal{P}_n$ appelée base de *Bernstein*. Ces polynômes sont utilisés pour construire les fameuses courbes de *Bézier* utilisées en CAO.

Erreur d'interpolation

On va à présent analyser l'erreur que l'on commet lorsqu'on remplace une fonction f par son polynôme d'interpolation p associé aux points $\{x_0, \ldots, x_n\}$ distincts. Le théorème suivant fournit une estimation de la différence.

Théorème 4 : *Erreur d'interpolation : soit f une fonction $(n+1)$ fois dérivable et soit p son polynôme d'interpolation dans $\mathcal{P}_n$ associé aux points $x_0, \ldots, x_n$ distincts. Alors pour tout $x \in \mathbb{R}$, il existe un réel $\theta_x \in]\min(x,x_i), \max(x,x_i)[$ tel que*

$$f(x) - p(x) = \frac{1}{(n+1)!}\Pi_{n+1}(x)\, f^{(n+1)}(\theta_x), \qquad \text{avec} \qquad \Pi_{n+1}(x) = \prod_{i=0}^{n}(x - x_i).$$

Preuve : Si $x = x_i$, alors $\Pi_{n+1}(x) = 0$ et la formule est juste. Fixons à présent $x \neq x_i\ \forall i$ et considérons q le polynôme d'interpolation de f en $x, x_0, \ldots, x_n$. On a évidemment $f(x) - p(x) = q(x) - p(x)$ et $q - p$ est un polynôme de degré $\leq n+1$ qui s'annule aux $(n+1)$ points $x_0, \ldots, x_n$. Par conséquent, il existe une constante α telle que $q(t) - p(t) = \alpha \prod_{i=0}^{n}(t - x_i) = \alpha\Pi_{n+1}(t)$, pour tout $t \in \mathbb{R}$. Il reste à montrer que $\alpha = \frac{1}{(n+1)!} f^{(n+1)}(\theta_x)$. Posons $r(t) = f(t) - q(t) = f(t) - p(t) - \alpha\Pi_{n+1}(t)$. On remarque que la fonction r s'annule $(n+2)$ fois en $x, x_0, x_1, \ldots, x_n$. En appliquant $(n+1)$ fois le théorème de Rolle, on en déduit que la dérivée r' s'annule $(n+1)$

fois dans l'intervalle $I =]\min(x, x_i), \max(x, x_i)[$. On peut à nouveau appliquer n fois le théorème de Rolle et ainsi de suite ... De cette façon, en appliquant par récurrence le théorème de Rolle, on obtient que la $(n+1)$-ième dérivée $r^{(n+1)}$ s'annule une fois dans I, c'est-à-dire qu'il existe $\theta_x \in I$ tel que $r^{(n+1)}(\theta_x) = 0$. On en déduit que

$$0 = r^{(n+1)}(\theta_x) = f^{(n+1)}(\theta_x) - p^{(n+1)}(\theta_x) - \alpha \Pi_{n+1}^{(n+1)}(\theta_x).$$

Or $p^{(n+1)} \equiv 0$ car $\deg(p) \leq n$ et $\Pi_{n+1}^{(n+1)} \equiv (n+1)!$. On obtient donc $\alpha = \dfrac{1}{(n+1)!} f^{(n+1)}(\theta_x)$ ce qui prouve le théorème. □

Problème de la convergence de l'interpolation

On a vu précédemment qu'on pouvait estimer l'erreur entre une fonction f et son polynôme d'interpolation. La question qu'on se pose à présent est de savoir si, sur un intervalle $[a, b]$ donné l'erreur d'interpolation diminue lorsqu'on augmente le nombre n de points d'interpolation et à la limite quand n tend vers l'infini, est-ce-que p converge (en un sens à préciser) vers f ? Commençons par étudier le cas de points d'interpolation répartis (presque [7]) de manière quelconque à savoir :

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

D'après le théorème précédent, on a :

$$\max_{x\in[a,b]} |f(x) - p(x)| \leq \frac{1}{(n+1)!} \max_{x\in[a,b]} |\Pi_{n+1}(x)| \max_{x\in[a,b]} \left|f^{(n+1)}(x)\right|.$$

On peut alors montrer (cf futur exercice ?) l'estimation suivante sur $|\Pi_{n+1}(x)|$:

$$\max_{x\in[a,b]} |\Pi_{n+1}(x)| \leq h^{n+1} n! \text{ où } h = \max_i h_i,\ h_i = x_{i+1} - x_i$$

On obtient donc alors la majoration de l'erreur d'interpolation suivante :

$$\max_{x\in[a,b]} |f(x) - p(x)| \leq \frac{h^{n+1}}{n+1} \max_{x\in[a,b]} \left|f^{(n+1)}(x)\right|.$$

En particulier si on choisit une répartition uniforme ($h = (b-a)/n$) ce qui semble être un choix "raisonnable", on a :

$$\max_{x\in[a,b]} |f(x) - p(x)| \leq \frac{1}{(n+1)} \left(\frac{b-a}{n}\right)^{n+1} \max_{x\in[a,b]} \left|f^{(n+1)}(x)\right|.$$

Il est clair que $\dfrac{1}{(n+1)} \left(\dfrac{b-a}{n}\right)^{n+1} \longrightarrow 0$ quand $n \to +\infty$ et cette convergence vers 0 est même très rapide. Par exemple, prenons un intervalle de longueur $b - a = 10$, on obtient :

n	5	10	15	20	25	30
$\frac{1}{(n+1)}(\frac{b-a}{n})^{n+1}$	$1.067\ 10^{1}$	$9.091\ 10^{-2}$	$9.515\ 10^{-5}$	$2.271\ 10^{-8}$	$1.732\ 10^{-12}$	$5.223\ 10^{-17}$

On serait donc tenté de penser que l'erreur tend vers 0 quand $n \to +\infty$. Mais attention car la $(n+1)$-ième dérivée de f dépend de n et en fait peut croître très rapidement avec n et pour

7. On impose juste que les deux extrémités correspondent à des points d'interpolation.

certaines fonctions, p ne converge pas vers f lorsque n tend vers $+\infty$. L'exemple suivant illustre une telle situation de non-convergence.

Exemple : *(Runge)*
On considère la fonction $f(x) = \dfrac{1}{1+x^2}$ sur l'intervalle $[-5,5]$. On note p_n le polynôme d'interpolation de f aux $n+1$ points équidistants dans l'intervalle $[-5,5]$. On observe alors (cf. Figure 3.1) quand n augmente, des problèmes aux extrémités de l'intervalle. En fait $\left|f^{(n)}(5)\right|$ devient rapidement grand avec n. On montre que pour $|x| \geq 3.83\cdots$, on a $|f(x) - p(x)| \to +\infty$ quand $n \to +\infty$. □

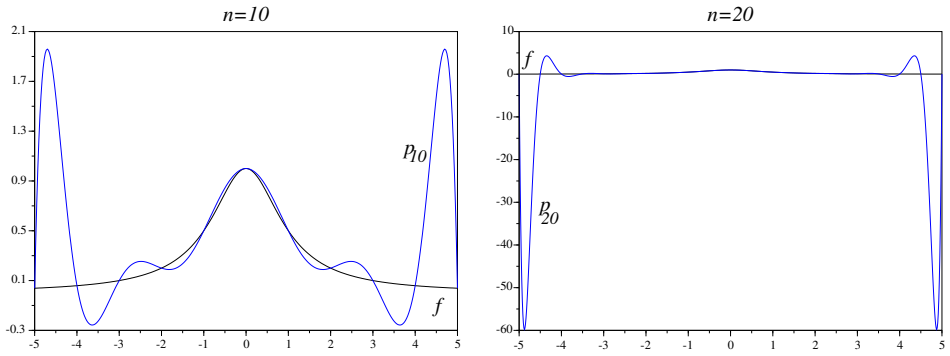


FIGURE 3.1 – Phénomène de Runge : "explosion" de l'interpolation polynomiale avec points équidistants

Nous venons de voir que la convergence du polynôme d'interpolation vers une fonction f n'est pas assurée lorsqu'on choisit les points d'interpolation répartis de façon uniforme dans un intervalle fermé borné. Mais existe-t-il une répartition (évidemment non uniforme) des points d'interpolation pour laquelle il y aurait convergence ? Une réponse est fournie par les abscisses de Tchebichev. Sans rentrer dans les détails et en considérant que l'intervalle sur lequel on veut approcher une fonction est $[-1,1]$, le polynôme de $\mathcal{P}_{n+1}$ (de la forme de Π_{n+1}) qui minimise :

$$\|\Pi_{n+1}\|_\infty = \max_{x\in[-1,1]} |\Pi_{n+1}(x)| .$$

est appelé polynôme de Tchebichev et ses $n+1$ racines sont données par :

$$x_i = \cos\left(\frac{(2i+1)\pi}{2(n+1)}\right),\ i \in [\![0,n]\!]$$

On peut voir que celles-ci sont plus denses aux extrémités de l'intervalle (-1 et 1) et sont réparties symétriquement autour de 0 (car $x_{n-i} = -x_i$).

Pour calculer les abscisses de Tchebichev d'un intervalle quelconque $[a,b]$ il suffit d'utiliser la transformation affine :

$$[-1,1] \ni x \longmapsto X = \frac{b-a}{2}x + \frac{a+b}{2} \in [a,b]$$

En particulier l'exemple précédent avec la fonction de Runge converge si on utilise cette répartition. D'une manière générale les polynômes d'interpolation construits avec les abscisses de Tchebichev convergent vers f avec peu de restrictions :

Théorème 5 : Si f est absolument continue sur l'intervalle $[a, b]$ alors la séquence des polynômes d'interpolation p_n construits avec la répartition de Tchebichev converge uniformément vers f, c'est à dire :

$$\lim_{n\to+\infty} \underbrace{\|f - p_n\|_\infty}_{=\max_{x\in[a,b]} |f(x)-p_n(x)|} = 0$$

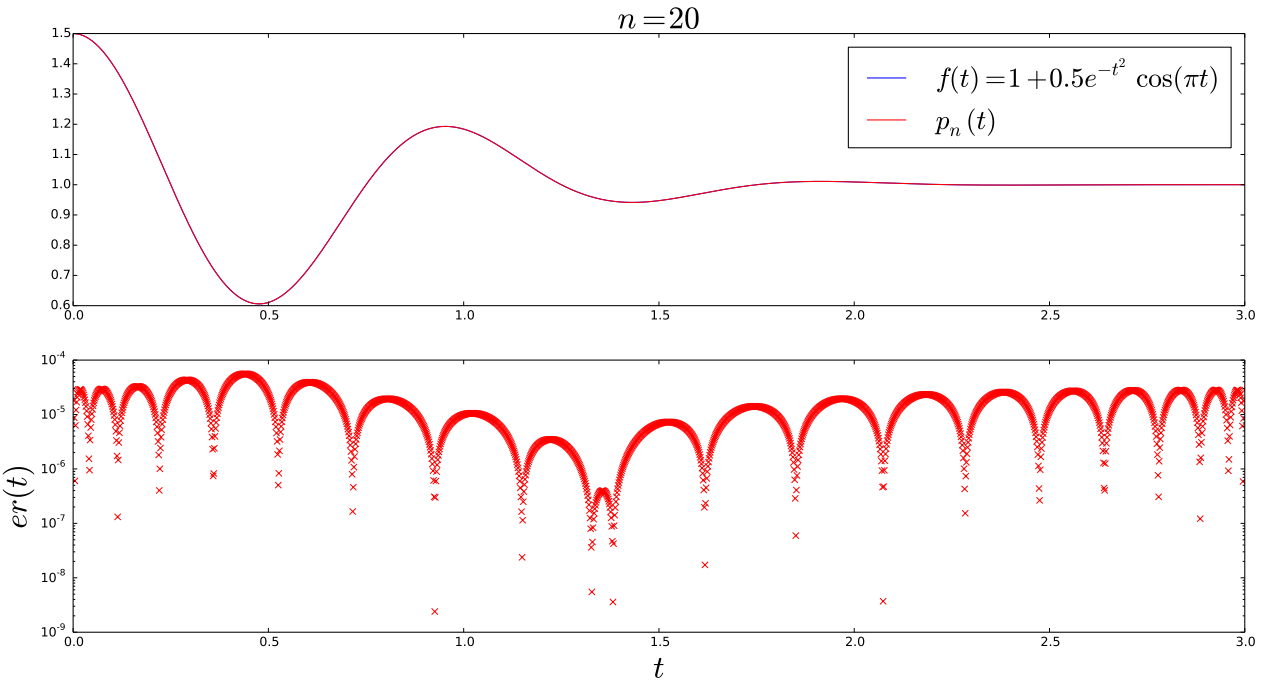
Une fonction f est absolument continue sur un intervalle $[a, b]$ si et seulement s'il existe une fonction g intégrable (au sens de Lebesgue) sur $[a, b]$ telle que :

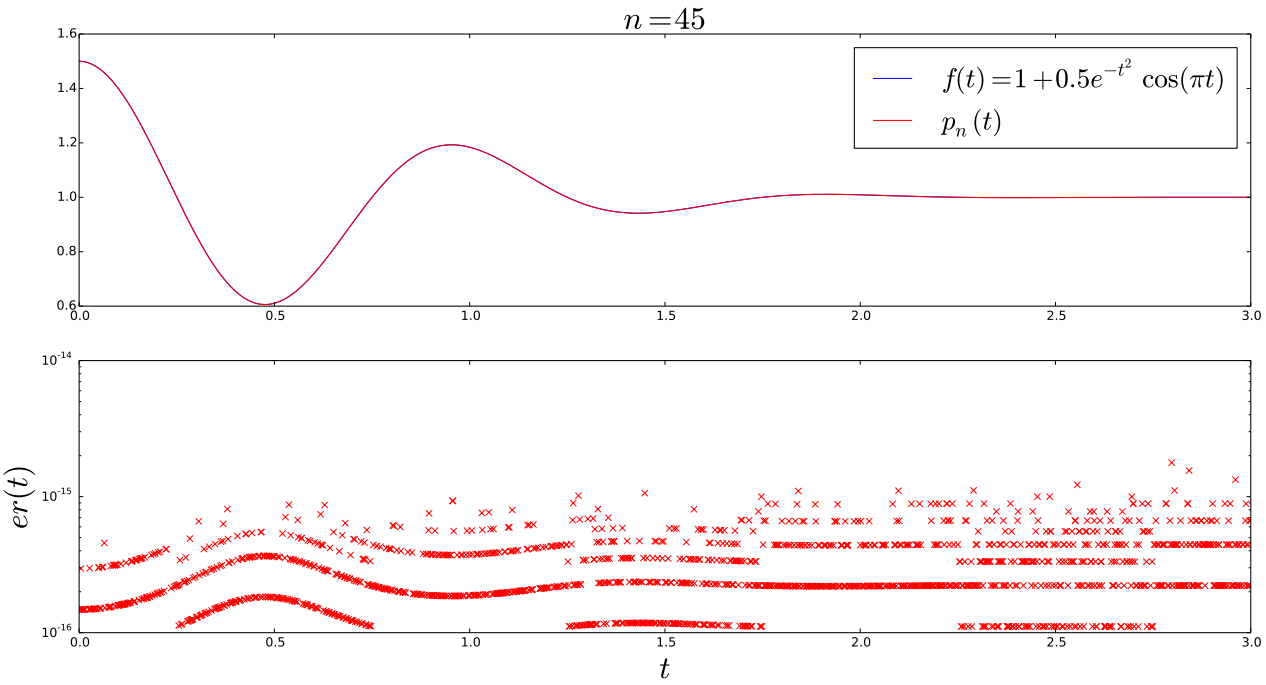
$$f(x) - f(a) = \int_a^x g(t)dt, \ \forall x \in [a, b]$$

En particulier si f est dérivable sur $[a, b]$ elle est absolument continue sur $[a, b]$.

Un exemple

On cherche à approcher la fonction par interpolation $f(t) = 1 + 0.5e^{-t^2}\cos(\pi t)$ sur $[-3, 3]$. Profitant de la parité de cette fonction, les calculs se font sur $[0, 3]$. Dans cet exemple on utilise les abscisses de Tchebichev et la formule barycentrique de Lagrange. Avec 20 points d'interpolation on obtient une précision relative inférieure à 10^{-4} et on obtient presque la précision machine avec 45 points d'interpolation.





3.2.7 Interpolation de Lagrange-Hermite

Dans certains problèmes, on est amené à chercher un polynôme qui interpole une fonction f en des points donnés (interpolation de Lagrange) ainsi que les dérivées en ces points (pentes données). C'est l'interpolation de Lagrange-Hermite. Plus précisement, on se donne des triplets $(x_i,\, y_i,\, y_i')$, pour $i = 0, \ldots, n$, où $y_i = f(x_i)$ et $y_i' = f'(x_i)$ sont connus (f' désigne la dérivée de f). On cherche alors un polynôme p (polynôme d'interpolation de Lagrange-Hermite) tel que :

$$\begin{cases} p(x_i) = y_i \\ p'(x_i) = y_i', \quad \text{pour } i = 0, \ldots, n. \end{cases}$$

On voit clairement qu'on dispose de $2(n+1)$ équations. Il faut donc $2(n+1)$ inconnues et par conséquent on cherche un polynôme de degré $\leq 2n+1$.
Le polynôme d'interpolation de Lagrange-Hermite est donné par

$$p(x) = \sum_{i=0}^{n} y_i H_i(x) + \sum_{i=0}^{n} y_i' K_i(x), \tag{3.1}$$

où les polynômes de Lagrange-Hermite H_i et K_i sont définis par

$$\begin{aligned} H_i(x) &= (1 - 2(x - x_i)L_i'(x_i))\, L_i^2(x), \\ K_i(x) &= (x - x_i)L_i^2(x) \end{aligned}$$

et ceci pour $i = 0, \ldots, n$. On vérifie que H_i et K_i sont bien des polynômes de degré $2n+1$.

On peut montrer le résultat d'erreur suivant (assez conforme à celui obtenu pour l'interpolation simple) :

Théorème 6 : *Erreur d'interpolation (Lagrange-Hermite) : soit f une fonction $(2n+2)$ fois dérivable et soit p son polynôme d'interpolation de Lagrange-Hermite dans $\mathcal{P}_{2n+1}$ associé aux*

points $x_0, \ldots, x_n$ distincts. Alors pour tout $x \in \mathbb{R}$, il existe un réel $\theta_x \in]\min(x, x_i), \max(x, x_i)[$ tel que

$$f(x) - p(x) = \frac{1}{(2n+2)!}\Pi_{n+1}^2(x)\, f^{(2n+2)}(\theta_x), \qquad \text{avec} \qquad \Pi_{n+1}(x) = \prod_{i=0}^{n}(x - x_i).$$

3.3 Interpolation polynomiale par morceaux 1d

Si l'interpolation polynomiale avec abscisses de Tchebichev est un puissant moyen d'approximation elle ne répond pas à tous les besoins car avec des données expérimentales :

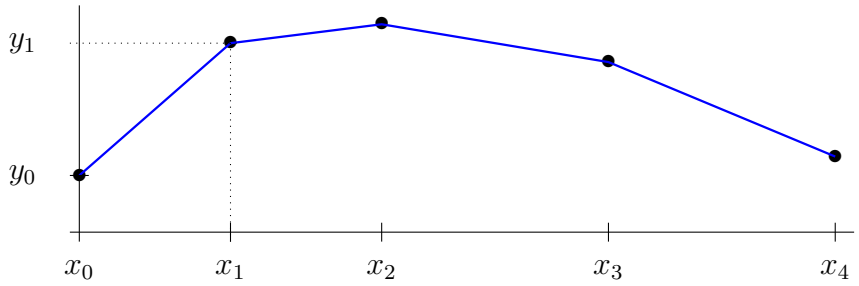
— les abscisses ne suivent généralement pas une telle répartition ;

— il y a des erreurs de mesures.

Lorsque les erreurs de mesure sont importantes il vaut d'ailleurs mieux ne pas interpoler (cf chapitre suivant) mais même avec des erreurs faibles, utiliser un même polynôme sur tout l'intervalle n'est souvent pas adapté. Une alternative est alors d'utiliser plusieurs polynômes et la plus simple d'entre-elles est la ligne brisée.

3.3.1 La ligne brisée (interpolation affine par morceaux)

Ce procédé consiste à relier deux points d'interpolation consécutifs par un segment de droite :



Sur chaque intervalle $I_k := [x_{k-1}, x_k]$ il s'agit donc d'interpoler par un polynôme $p_k \in \mathcal{P}_1$, ce qui donne :

$$p_k(x) = \begin{cases} y_{k-1}\left(\dfrac{x - x_k}{x_{k-1} - x_k}\right) + y_k\left(\dfrac{x - x_{k-1}}{x_k - x_{k-1}}\right) & \text{sous la forme “Lagrange”} \\ y_{k-1} + \left(\dfrac{y_k - y_{k-1}}{x_k - x_{k-1}}\right)(x - x_{k-1}) & \text{sous la forme “Newton”} \end{cases}$$

Etant donné x il suffit donc de repérer [8] l'intervalle $I_k = [x_{k-1}, x_k]$ qui contient x puis d'utiliser l'une de ces deux formules. Si on suppose que l'on approche une fonction f sur un intervalle $[a, b]$ par ce procédé en utilisant n sous-intervalles $a = x_0 < x_1 < \cdots < x_n = b$ (on notera $\mathcal{X}_n$ la subdivision $x_0, x_1, \ldots, x_n$ et donc $y_k = f(x_k)$) que peut-on attendre comme précision (pour une fonction suffisamment régulière) ? La réponse est assez simple. Sur l'intervalle I_k en utilisant le théorème sur l'erreur d'interpolation, on obtient que [9] :

$$\max_{x \in I_k} |f(x) - p_k(x)| \leq \frac{1}{2} \max_{x \in I_k} |(x - x_{k-1})(x - x_k)| \max_{x \in I_k} |f''(x)|$$

8. Par dichotomie par exemple, cf TD.

9. Si f est deux fois continûment dérivable sur $[a, b]$.

Il est facile de voir que le maximum de la fonction $x \mapsto |(x - x_{k-1})(x - x_k)|$ est obtenu pour $x = \frac{x_{k-1}+x_k}{2}$ ce qui nous donne :

$$\max_{x\in I_k} |f(x) - p_k(x)| \leq \frac{h_k^2}{8} \max_{x\in I_k} |f''(x)| \text{ où } h_k = x_k - x_{k-1}$$

et donc en notant $p^{f,\mathcal{X}_n}$ l'interpolant ainsi construit on obtient :

$$\max_{x\in[a,b]} |f(x) - p^{f,\mathcal{X}_n}(x)| \leq \frac{h^2}{8} \max_{x\in[a,b]} |f''(x)| \text{ où } h = \max_k h_k$$

Si on prend une subdivision régulière $h = (b - a)/n$ il est clair que le procédé converge et que lorsque l'on double le nombre d'intervalles la majoration d'erreur précédente est divisée par 4. C'est en général beaucoup moins rapide que si on interpole avec un polynôme avec abscisses de Tchebichev mais il y a une certaine souplesse, en particulier au lieu de prendre une subdivision uniforme on a tout intérêt à resserrer les abscisses d'interpolation là où la dérivée seconde de f est grande en module.

L'interpolation affine par morceaux comme un sous-espace vectoriel de fonctions

Supposons que l'on fixe les abscisses $\mathcal{X}_n = \{x_0, x_1, \ldots, x_n\}$ et que l'on définisse l'ensemble des fonctions continues sur $[a, b]$ (noté $\mathcal{C}([a, b], \mathbb{R})$) dont la restriction sur chaque sous-intervalle $I_k = [x_{k-1}, x_k], k = 1, \ldots, n$ est un polynôme de degré inférieur ou égal à 1 :

$$\mathcal{S}_1^{\mathcal{X}_n} = \{f \in \mathcal{C}([a, b], \mathbb{R}) : f_{|I_k} \in \mathcal{P}_1\}$$

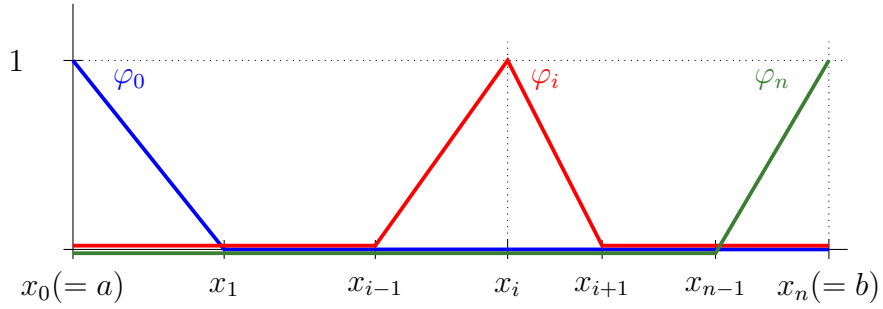
En fait il s'agit de l'ensemble des "lignes brisées" que l'on peut obtenir avec ces abscisses fixes. Une fonction s de cet ensemble est uniquement déterminée par les valeurs $(y_k)_{k\in[\![0,n]\!]}$ de s en les abscisses $(x_k)_{k\in[\![0,n]\!]}$: $y_k = s(x_k)$. Il est aisé de remarquer que c'est un sous-espace vectoriel (de $\mathcal{C}([a, b], \mathbb{R})$). En effet si on multiplie une fonction de $\mathcal{S}_1^{\mathcal{X}_n}$ par un scalaire α on reste bien dans cet ensemble et de même si on additionne deux fonctions de $\mathcal{S}_1^{\mathcal{X}_n}$. On doit donc pouvoir exhiber une base avec a priori $n + 1$ éléments. La base la plus naturelle est celle dont "les degrés de liberté" $(y_k)_{k\in[\![0,n]\!]}$ (qui fixent une fonction s donnée de $\mathcal{S}_1^{\mathcal{X}_n}$) vont en être les coordonnées. On va montrer que :

$$s(x) = \sum_{i=0}^{n} y_i \varphi_i(x), \forall x \in [a, b] \tag{3.2}$$

où les φ_i sont appelées fonctions "chapeaux" et sont données par :

$$\varphi_i(x) = \begin{cases} 0 & \text{si } x \notin [x_{i-1}, x_{i+1}] \\ \frac{x-x_{i-1}}{x_i-x_{i-1}} & \text{si } x \in [x_{i-1}, x_i] \\ \frac{x-x_{i+1}}{x_i-x_{i+1}} & \text{si } x \in [x_i, x_{i+1}] \end{cases} \tag{3.3}$$

avec deux exceptions pour $i = 0$ et $i = n$ où l'on a juste une moitiée de chapeau :



Rmq : on a légèrement décalé le zéro pour φ_j (légèrement au-dessus) et φ_n (légèrement au dessous) de manière à ne pas superposer les couleurs.

Preuve : $\forall x \in [a,b]$, $\exists k \in [\![1,n]\!]$ tel que $x \in I_k$. Si on regarde la définition des fonctions chapeaux, seules φ_{k-1} et φ_k sont non nulles sur ce sous-intervalle et la somme dans l'expression (3.2) se réduit donc aux deux termes :

$$s(x) = y_{k-1}\varphi_{k-1}(x) + y_k\varphi_k(x)$$

Toujours d'après (3.3), on obtient, pour $x \in [x_{k-1}, x_k]$:

$$\begin{aligned} \varphi_{k-1}(x) &= \frac{x - x_k}{x_{k-1} - x_k} \\ \varphi_k(x) &= \frac{x - x_{k-1}}{x_k - x_{k-1}} \end{aligned}$$

d'où finalement pour $x \in [x_{k-1}, x_k]$:

$$s(x) = y_{k-1}\frac{x - x_k}{x_{k-1} - x_k} + y_k\frac{x - x_{k-1}}{x_k - x_{k-1}}$$

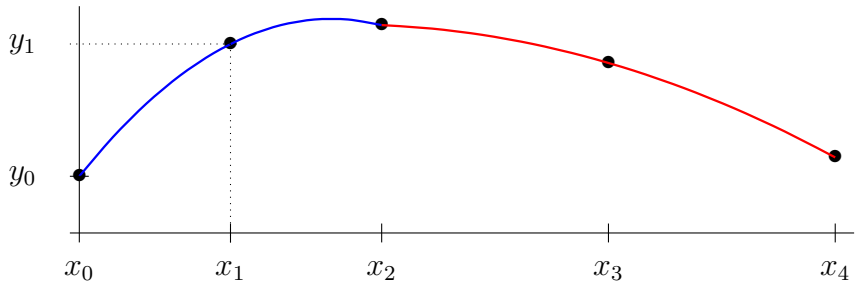
qui correspond bien l'interpolation affine des points (x_{k-1}, y_{k-1}) et (x_k, y_k) □

Une autre façon de déterminer cette base est de montrer que l'application $\mathcal{A}$ qui à $(y_k)_{k\in[\![0,n]\!]} \in \mathbb{R}^{n+1}$ associe $s \in \mathcal{S}_1^{\mathcal{X}_n}$ est un isomorphisme d'espaces vectoriels. La base des fonctions chapeaux est alors obtenue comme l'image de la base canonique de $\mathbb{R}^{n+1}$ par $\mathcal{A}$: φ_i est la fonction de $\mathcal{S}_1^{\mathcal{X}_n}$ qui vaut 1 en x_i et 0 sur les autres noeuds.

> S'il est inutile de mettre en évidence cette base des fonctions chapeaux pour faire de l'interpolation cela est par contre indispensable lorsqu'on veut approcher des données par moindres carrés (cf chapitre suivant).

3.3.2 L'interpolation de Lagrange par morceaux

Une façon de généraliser une ligne brisée est de faire passer un polynôme de $\mathcal{P}_2$ par les points $(x_0, y_0), (x_1, y_1), (x_2, y_2)$ puis un autre polynôme de $\mathcal{P}_2$ par les points $(x_2, y_2), (x_3, y_3), (x_4, y_4)$, etc...En utilisant les mêmes données que dans l'exemple de la ligne brisée on obtient :



ce qui met bien en évidence le défaut de cette approche : l'interpolant n'est pas dérivable en x_2 qui est le noeud intermédiaire entre les deux polynômes. Un autre défaut est que le nombre de points d'interpolation doit être impair [10] ou alors l'un des morceaux doit être un segment de droite. Bref ce type d'approximation n'est généralement pas utilisé pour interpoler des points donnés (mais peut être utile pour d'autres besoins).

10. Si on utilise des polynômes de $\mathcal{P}_3$ il faut que le nombre de points soit en $3m+1$, etc...

3.3.3 L'interpolation par fonctions splines

Dans ce cours paragraphe nous allons survoler ce domaine très important qui a de nombreuses applications non seulement en mathématiques appliquées mais aussi en CAO. Un point de vue qui permet d'aborder les splines est de construire des fonctions polynomiales par morceaux mais avec des raccords plus réguliers que l'interpolation de Lagrange par morceaux.

Une première méthode pour construire une telle fonction polynomiale par morceaux est d'utiliser un polynôme de Lagrange-hermite (avec $n = 1$) sur chaque segment I_k ce qui va permettre d'obtenir une fonction continûment dérivable. Ce point de vue est développé dans l'exercice 2 de la feuille 5. Ce procédé est pratique mais assez limité, les splines permettant d'aller beaucoup plus loin.

Définition des espaces de splines

On se donne un intervalle $[a, b]$ et une subdivision $\mathcal{X}_n = \{x_0, x_1, \ldots, x_n\}$ avec $a = x_0 < x_1 < \cdots < x_n = b$. L'espace $\mathcal{S}_k^{\mathcal{X}_n}$ des fonctions splines de degré k [11] définies sur $[a, b]$ muni de la subdivision $\mathcal{X}_n$ est :

$$\mathcal{S}_k^{\mathcal{X}_n} = \{f \in \mathcal{C}^{(k-1)}([a, b], \mathbb{R}) : f_{|I_j} \in \mathcal{P}_k\}$$

c'est à dire est formé sur chaque intervalle $I_j := [x_j, x_{j+1}]$ d'un polynôme de degré inférieur ou égal à k et ses polynômes se "raccordent" (au moins) jusqu'à l'ordre $k-1$ en chaque noeud interne x_1, x_2, …, x_{n-1}. Si on appelle $p_j := f_{|I_j}$ alors les conditions de raccords s'énoncent :

$$p_j^{(\ell)}(x_{j+1}) = p_{j+1}^{(\ell)}(x_{j+1}),\ j \in [\![0, n-2]\!],\ \ell \in [\![0, k-1]\!] \tag{3.4}$$

On a déjà rencontré un tel espace : les lignes brisées qui sont des splines de degré 1. Comme pour les lignes brisées il n'est pas obligatoire d'exhiber une base d'un tel espace pour faire de l'interpolation, par exemple l'exercice 4 de la feuille 5 propose une méthode pour construire une spline cubique (i.e. de degré 3) d'interpolation.

Il est assez simple de voir que $\mathcal{S}_k^{\mathcal{X}_n}$ est un espace vectoriel. On peut aussi deviner sa dimension avec les arguments suivants [12] :

- on a donc n polynômes (il y a n intervalles) de degré $\leq k$ soit $n \times (k+1)$ coefficients ;
- cependant ces $n \times (k+1)$ coefficients ne peuvent pas être choisis de manière quelconque car il faut respecter les $(n-1) \times k$ conditions de raccords.

On en déduit donc que :

$$dim(\mathcal{S}_k^{\mathcal{X}_n}) = n \times (k+1) - (n-1) \times k = n + k$$

Bases des espaces de splines

Pour l'espace des lignes brisées on a vu que la "bonne" base est celle des fonctions chapeaux. Cependant dès que $k \geq 2$ il est plus difficile de trouver une "bonne base" de $\mathcal{S}_k^{\mathcal{X}_n}$. Au tout début de l'étude de ces espaces de fonctions on a mis en évidence qu'une base de $\mathcal{S}_k^{\mathcal{X}_n}$ est formée des fonctions suivantes :

$$\underbrace{\{e_0 : x \mapsto 1, e_1 : x \mapsto x - a, \ldots, e_k : x \mapsto (x-a)^k\}}_{\text{base canonique "décalée" de } \mathcal{P}_k} \cup \{g_j : x \mapsto (x - x_j)_+^k,\ j = 1, \ldots, n-1\}$$

11. On dit aussi splines d'ordre $k+1$.
12. Ce raisonnement "empirique" peut être rendu rigoureux…

où la notation $_+$ désigne la partie positive d'une fonction ($(f(x))_+ = f(x)$ si $f(x) \geq 0$ et $(f(x))_+ = 0$ sinon) et donc :

$$(x - x_j)_+^k := \begin{cases} 0 & \text{pour } x < x_j \\ (x - x_j)^k & \text{pour } x \geq x_j \end{cases}$$

C'est un bon exercice de voir que ces $n + k$ fonctions sont bien dans l'espace $\mathcal{S}_k^{\mathcal{X}_n}$ (en particulier tout polynôme de $\mathcal{P}_k$ est bien dans $\mathcal{S}_k^{\mathcal{X}_n}$, dans ce cas les noeuds internes ne sont pas "actifs", c'est à dire qu'il n'y a pas de discontinuité dans la k ème dérivée) et qu'elles forment une famille libre (et donc une base puisque le nombre de fonctions est égal à la dimension de l'espace). Cependant cette base est très mauvaise pour les calculs en arithmétique flottante.

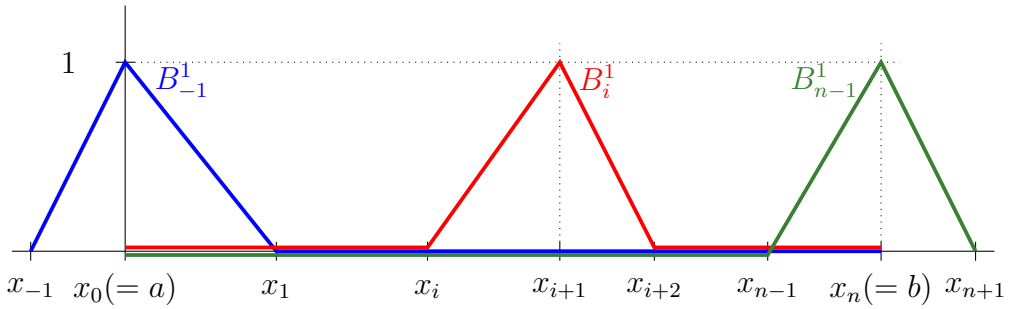
Les "bonnes" bases de ces espaces sont formées de splines particulières appelées B-splines et dans le cas de $\mathcal{S}_1^{\mathcal{X}_n}$ se sont exactement les fonctions chapeaux. Si on réexamine ces fonctions on remarque qu'elles possèdent les propriétés suivantes :

1. φ_j est non nulle uniquement sur l'intervalle $[x_{j-1}, x_{j+1}]$ sur lequel elle est positive ;
2. $\sum_{j=0}^{n} \varphi_j(x) = 1,\ \forall x \in [a, b]$.

Avant d'aller plus loin notons que s'il est naturel de "centrer" φ_j en x_j, cela ne l'est plus à partir de $k \geq 2$ et la tradition est de numéroter une fonction B-spline avec le numéro du noeud à partir duquel elle devient non nulle. Avec cette convention, on note $B_i^1 := \varphi_{i+1}$ et dans ce cas le "support" de B_i^1 est $[x_i, x_{i+2}]$. Cependant ceci ne fonctionne pas "tel que" pour les fonctions chapeaux des deux extrémités :

1. $B_{-1}^1 = \varphi_0$ a comme support $[x_0, x_1]$ (et pas $[x_{-1}, x_1]$)
2. $B_{n-1}^1 = \varphi_n$ a comme support $[x_{n-1}, x_n]$ (et pas $[x_{n-1}, x_{n+1}]$)

Et il est donc naturel de rajouter des noeuds a priori en dehors de l'intervalle $[a, b]$ de manière à avoir une sorte d'uniformité dans la définition des B-splines. Par exemple pour les fonctions chapeaux :



Ainsi la définition des B-splines de degré k se fait sur ce même modèle :

1. on rajoute k noeuds $x_{-k}, \ldots, x_{-1}$ à gauche de a et k noeuds $x_{n+1}, \ldots, x_{n+k}$ à droite de b ;
2. on définit alors $n + k$ fonctions B_i^k, $i = -k, \ldots, n - 1$; le support de B_i^k étant $[x_i, x_{i+k+1}]$, cette fonction devant être positive sur cet intervalle ;
3. elles vérifient la propriété :

$$\sum_{i=-k}^{n-1} B_i^k(x) = 1, \forall x \in [a, b]$$

On montre que les $n + k$ B-splines de degré k peuvent être définies à partir des $n + k + 1$ B-splines de degré $k - 1$ que l'on peut définir à partir des noeuds $x_{-k}, \ldots, x_{n+k}$ par la relation :

$$B_i^k(x) = \frac{x - x_i}{x_{i+k} - x_i} B_i^{k-1}(x) + \frac{x_{i+k+1} - x}{x_{i+k+1} - x_{i+1}} B_{i+1}^{k-1}(x),\ i = -k, \ldots, n - 1 \tag{3.5}$$

Par exemple pour obtenir les B-splines quadratiques ($k = 2$), on rajoute deux noeuds x_{-2} et x_{-1} à gauche de a et deux autres x_{n+1} et x_{n+2} à droite de b. On peut alors définir $n+3$ fonctions chapeaux :

$$B_i^1(x) = \begin{cases} 0 & \text{si } x \notin [x_i, x_{i+2}] \\ \frac{x-x_i}{x_{i+1}-x_i} & \text{pour } x \in [x_i, x_{i+1}] \\ \frac{x_{i+2}-x}{x_{i+2}-x_{i+1}} & \text{pour } x \in [x_{i+1}, x_{i+2}] \end{cases} \qquad i = -2, \ldots, n$$

qui permettent alors de définir à l'aide de (3.5) les $n+2$ fonctions B-splines qui forment une base de $\mathcal{S}_2$. Pour B_i^2, on utilise donc B_i^1 et B_{i+1}^1 :

1. sur $[x_i, x_{i+1}]$ seule B_i^1 est non nulle , (3.5) donne :

$$B_i^2(x) = \frac{x - x_i}{x_{i+2} - x_i} \, \frac{x - x_i}{x_{i+1} - x_i} = \frac{(x - x_i)^2}{(x_{i+2} - x_i)(x_{i+1} - x_i)}, \; x \in [x_i, x_{i+1}]$$

2. sur $[x_{i+1}, x_{i+2}]$:

$$B_i^2(x) = \frac{x - x_i}{x_{i+2} - x_i} \, \frac{x_{i+2} - x}{x_{i+2} - x_{i+1}} + \frac{x_{i+3} - x}{x_{i+3} - x_{i+1}} \, \frac{x - x_{i+1}}{x_{i+2} - x_{i+1}}$$

3. et sur $[x_{i+2}, x_{i+3}]$ seule B_{i+1}^1 est non nulle, ce qui donne :

$$B_i^2(x) = \frac{x_{i+3} - x}{x_{i+3} - x_{i+1}} \, \frac{x_{i+3} - x}{x_{i+3} - x_{i+2}} = \frac{(x_{i+3} - x)^2}{(x_{i+3} - x_{i+1})(x_{i+3} - x_{i+2})}$$