TP 1 : Interpolation polynomiale

Remarque. Cette fiche correspond à la fusion d'une fiche de J.-B. Bardet et d'une très vieille fiche d'O. Guibé.

L'interpolation polynomiale consiste en la résolution de deux problèmes légèrement distincts mais liés :

(1) Étant donné $n+1$ points du plan $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, trouver une fonction « simple » qui passe par ces n points ;

(2) Étant donné une fonction $f : I \longrightarrow \mathbf{R}$ « compliquée » (non-intégrable explicitement, ou connue seulement en certains points), trouver une fonction « simple » qui est une bonne approximation de f.

Un choix naturel de fonction « simple » est une fonction polynomiale, considérée comme étant d'autant plus simple que son degré est bas.

Exercice 1 (Interpolation de Lagrange)**.**
Soit $n \in \mathbf{N}$ et $n+1$ points du plan $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, d'abscisses x_i deux-à-deux distinctes. En introduisant les polynômes

$$L_i(x) = \prod_{\substack{0 \le j \le n \\ j \neq i}} \frac{x - x_j}{x_i - x_j},$$

montrer qu'il existe un et un seul polynôme de degré (au plus) n passant par ces $n+1$ points du plan. On l'appelle le **polynôme d'interpolation de Lagrange** de ces points. Si f est une fonction définie sur un intervalle contenant les points $x_0, \ldots, x_n$ on appelle polynôme d'interpolation de Lagrange de f, l'unique polynôme de degré au plus n passant par les points $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$.

Exercice 2 (Différences divisées de Newton)**.** L'expression du polynôme dans la base L_i ne permet pas réellement d'obtenir un algorithme efficace du calcul du polynôme d'interpolation de Lagrange. Les inconvénients sont l'instabilité numérique, un calcul assez long (même si on ne développe pas p le calcul de $p(x)$ prend du temps) et le fait qu'ajouter un point nécessite de refaire tous les calculs. Usuellement on utilise les différences divisées de Newton qui consistent à écrire le polynôme d'interpolation de Lagrange dans une base différente.

Soient f une fonction et $x_0, \ldots, x_n$ $n+1$ réels distincts (mais non nécessairement ordonnés). On « rappelle » que les différences divisées sont définies par

$$f[x_i] = f(x_i) \quad \forall i \in \{0, \ldots, n\}, \tag{1}$$

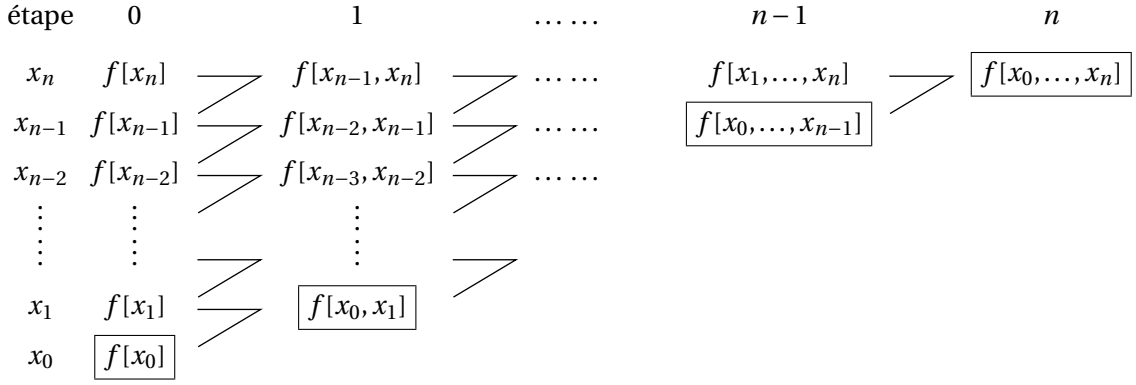
$$f[x_i, x_{i+1}, \ldots, x_j] = \frac{f[x_{i+1}, x_{i+2}, \ldots, x_j] - f[x_i, x_{i+1}, \ldots, x_{j-1}]}{x_j - x_i} \quad \text{si } 0 \le i < j \le n. \tag{2}$$

Démontrer que le polynôme, noté p, d'interpolation de Lagrange de f aux points $x_0, \ldots, x_n$ s'écrit

$$\begin{aligned} p(x) = {} & f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) \\ & + \cdots + f[x_0, x_1, \ldots, x_n](x - x_0)(x - x_1) \cdots (x - x_{n-1}) \end{aligned} \tag{3}$$

Exercice 3 (Avec Scilab)**.** On utilise (1) et (2) pour calculer toutes les différences divisées. Il est important d'éviter une méthode récursive (donc lente) ou gourmande en espace mémoire (stockage d'un tableau de taille $(n+1)$) alors qu'un seul vecteur de taille $n+1$ est nécessaire sans algorithme récursif. Nous utiliserons le tableau ci-dessous. Comme (d'après (1) et (2)) $f[x_i, \ldots, x_j]$ intervient au plus dans le calcul de $f[x_{i-1}, \ldots, x_j]$ et $f[x_i, \ldots, x_{j+1}]$ on procède à chaque nouvelle étape de façon « descendante » (i.e. à l'étape i, on calcule $f[x_i, \ldots, x_n]$ puis

$f[x_{i-1},\dots,x_{n-1}]$, etc, et $f[x_0,\dots,x_{n-i}]$; le tableau a été ordonné de façon à utiliser comme variable de stockage informatique un vecteur de taille $n+1$.



Suivant cette méthode on écrira la fonction `diffdiv(xpt,ypt)` avec en entrée `xpt` le vecteur ligne contenant les $n+1$ réels $x_0, x_1, \dots, x_n$ (attention, le vecteur a donc des indices allant de 1 à $n+1$) et `ypt` le vecteur ligne contenant les valeurs $f(x_0)$, $f(x_1), \dots, f(x_n)$, qui renvoie le vecteur des différences divisées `dpt`. On pourra utiliser la fonction `length` qui renvoie la longueur du vecteur.

Évaluation du polynôme par la méthode de Horner. C'est une méthode fine permettant d'évaluer un polynôme en un réel x. L'avantage de la méthode d'Horner par rapport à la méthode naïve est la stabilité numérique et le nombre d'opérations Soient $x_0,\dots,x_n$ et $a_0,\dots,a_n$ deux fois $n+1$ réels. Soit le polynôme p défini par

$$p(x) = a_0 + (x-x_0)a_1 + (x-x_0)(x-x_1)a_2 + \cdots + (x-x_0)\cdots(x-x_{n-1})a_n$$

Plutôt que de développer p (ce qui est long et peut mener à des désastres numériques) on utilise la méthode de Horner qui consiste à « voir » que

$$p(x) = a_0 + (x-x_0)\Big(a_1 + (x-x_1)\big(\cdots + (x-x_{n-2})(a_{n-1} + (x-x_{n-1})a_n)\cdots\big)\cdots\Big)$$

On définit alors pour évaluer p en $y \in \mathbf{R}$ la suite b_k par

$$b_n = a_n, \quad b_k = a_k + (y-x_k)b_{k+1} \quad \text{pour } k \text{ variant de } n-1 \text{ à } 0.$$

On obtient alors que $b_0 = p(y)$.

Suivant cette méthode on écrira `phorner(xpt,dpt,y)` qui calcule $p(y)$ avec la convention : le vecteur `xpt` contient $x_0,\dots,x_n$, `dpt` contient $a_0,\dots,a_n$ et y contient la valeur de y

Vous avez désormais la possibilité d'évaluer le polynôme d'interpolation et de tester vos procédures!!! La vectorisation étant l'un des points forts de Scilab, la procédure `phorner` devra accepter en argument un vecteur `y` pas seulement un réel, ceci accélérera la production des graphiques.

Premier test. Tapez
`xpt=[1,2,3,4]; deff('y=f(x)','y=sin(x)'); dpt=diffdiv(xpt,f(xpt));`
`phorner(xpt,dpt,1); phorner(xpt,dpt,2); phorner(xpt,dpt,3);`
`phorner(xpt,dpt,4)` et comparez avec les valeurs de sin(1), sin(2), sin(3), sin(4).

On peut aussi définir `deff('y=p(x)','y=phorner(xpt,dpt,x)')` et ainsi `p` est le polynôme d'interpolation de Lagrange de la fonction sinus aux points 1, 2, 3 et 4 au sens de l'évaluation et non pas au sens polynomiale.

a) Écrire une fonction `interpol(n)` qui trace sur un même graphe la fonction $f(x) = \exp(x)$ et son interpolation polynomiale construite à partir de $n+1$ points équirépartis dans l'intervalle $[-1,1]$.
Mettre en évidence la convergence des polynômes d'interpolation vers la fonction f lorsque n tend vers l'infini.

b) Écrire la procédure `einterp(n,f,a,b)`, avec en entrée $n \in \mathbf{N}^*$, f une fonction définie sur $[a,b]$ avec $a<b$, qui trace sur un même graphe, la fonction f, le polynôme d'interpolation de Lagrange de f aux points équidistants $a+k(b-a)/n$, $k \in \{0,\dots,n\}$ et les points d'interpolation avec un motif (voir Figure 1 pour la fonction $\sin(x)$ et quelques points). Il faut bien sûr utiliser `diffdiv` et `phorner`.

Exercice 4 (Erreur d'interpolation ; phénomène de Runge)**.** Soit I un intervalle de $\mathbf{R}$, f une fonction C^{n+1} sur I, et $n+2$ points distincts de I, $x_0, x_1, \dots, x_n$ et x. On note $\Pi_n f$ le polynôme d'interpolation de f construit sur les points d'interpolation $x_0, x_1, \dots, x_n$, et on introduit

$$E_n(t) = f(t) - \Pi_n(t) \qquad \text{l'erreur d'interpolation au point } t$$
$$\omega_{n+1}(t) = \prod_{0\le i\le n} (t - x_i)$$
$$G(t) = E_n(t) - \omega_{n+1}(t)\frac{E_n(x)}{\omega_{n+1}(x)}$$

a) Montrer que G s'annule aux $n+2$ points distincts $x_0, x_1, \dots, x_n$ et x.

b) En déduire qu'il existe $\xi \in I$ tel que $G^{(n+1)}(\xi) = 0$.

c) Démontrer à l'aide de la question précédente que, pour tout $x \in I$, il existe $\xi \in I$ tel que

$$E_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x).$$

d) Confirmer théoriquement les observations numériques de la question a) de l'Exercice 1, en montrant que, pour $I=[-1,1]$, et pour $f(x)=\exp(x)$,

$$\sup_{x\in I} |E_n(x)| \xrightarrow[n\to+\infty]{} 0.$$

e) De nouveau à l'aide de la fonction `interpol1`, illustrer graphiquement le fait que, pour $I=[-5,5]$ et $f(x)=\frac{1}{1+x^2}$, la suite des polynômes d'interpolation de Lagrange construits sur des points d'interpolation équirépartis ne converge pas uniformément vers f.
Comment expliquer ce phénomène (qu'on appelle le **phénomène de Runge**) ?

Exercice 5 (Interpolation de Chebychev)**.** On complique et on prend les points de Tchebychef définis par

$$x_k = \frac{a+b}{2} + \frac{b-a}{2}\cos\Big(\frac{2k+1}{2(n+1)}\pi\Big), \qquad k \in \{0,\dots,n\}.$$

a) Procédure `tinterp(n,f,a,b)`. Idem que `einterp` mais aux points de Tchebychef.

b) Testez votre procédure sur les deux exemples précédents.

c) Observez-vous un phénomène similaire ?

d) Pour ce type de fonctions quel choix de points d'interpolation vous paraît préférable ?

e) Prendre $f(x)=|x|$ sur $[-1,1]$ et constater un phénomène similaire. Pour information si f est dérivable (sauf en un nombre fini de points) alors il est rassurant de choisir les points de Tchebychef.

Remarque. Dans tous les cas (points équiréparties et points de Tchebychev) il est important de savoir distinguer l'instabilité numérique du aux limites de la précision de la machine et l'effet de Runge.

Exercice 6 (Perturbation de l'interpolation de Lagrange)**.** « Rappelons » quelques éléments sur les polynômes de Lagrange. Soient $a<b$ deux réels et $x_0,\dots,x_n$, $n+1$ points distincts dans $[a,b]$. Pour $0\le i\le n$, on note l_i les polynômes de base de l'interpolation de Lagrange (l_i est de degré au plus n et $l_i(x_j)=\delta_i^j$). Notons $\lambda_n(x)=\sum_{i=0}^n |l_i(x)|$ et $\Lambda_n = \sup_{a\le x\le b}\lambda_n(x) = \|\lambda_n\|_\infty$ (Λ_n est appelée constante de Lebesgue associée aux points $x_0,\dots,x_n$). Considérons une fonction f définie sur $[a,b]$. Posons $y_i=f(x_i)$ et $\widehat{y}_i = f(x_i)+\varepsilon_i$ (ε_i représente l'erreur commise sur

la valeur y_i). Soient p_n et $\widehat{p}_n$ les polynômes d'interpolation de Lagrange aux points (x_i, y_i) et $(x_i, \widehat{y}_i)$ respectivement. Alors on a

$$\|p_n - \widehat{p}_n\|_\infty \leq \Lambda_n \max_{0\leq i\leq n} |\varepsilon_i|.$$

Avec Scilab.

a) Soient $a = 0$, $b = 5$ et $f(x) = \sin(x)$. Le théorème de l'erreur nous permet de montrer facilement que p_n le polynôme d'interpolation de Lagrange aux points équidistants $5i/n$ ($0 \leq i \leq n$) converge uniformément vers $f(x)$ sur $[0,5]$ (une estimation grossière donne $\|p_n - f\|_\infty \leq 5^n/n!$). Pour les points équidistants nous savons que Λ_n tend vers $+\infty$.

Fixons n un entier naturel. Pour simuler une perturbation sur les données $\sin(5i/n)$, il suffit d'utiliser la fonction `rand`. Le schéma est le suivant, soient les points $x_i = 5i/n$, `xpt` |

- `xpt` contient $x_0, \ldots, x_n$.
- générer le vecteur `ypt` contenant les valeurs de f aux points x_i.
- soit `zpt` une perturbation d'ordre 10^{-5} du vecteur `ypt` à l'aide de `rand`.
- on obtient deux polynômes d'interpolation, p_n valant `ypt` en `xpt` et $\widehat{p}_n$ valant `zpt` en `xpt`.
- Tracer, sur le même graphique, p_n, $\widehat{p}_n$ et $\sin(x)$.

Tester pour $n = 10$, $n = 15$, $n = 20$, $n = 25$, $n = 30$, $n = 35$, $n = 40$. Tester aussi pour des pertubations plus petites, plus grandes…

b) Faire de même avec les points de Tchebychef,

c) Comparer les résultats entre points équidistants et points de Tchebychev avec une même perturbation. Qu'observez-vous ?

d) Montrer les limites de Scilab en traçant p_n pour les points équidistants et pour les points de Tchebychev pour n grand avec la fonction $\sin(x)$ sur l'intervalle $[0,5]$. En théorie, p_n converge uniformément vers f quand n tend vers l'infini. Qu'observez-vous ? Expliquez.