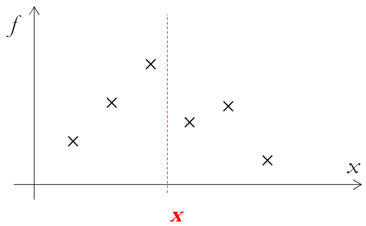
Chapitre 5
Interpolation et Approximation

1.1.

Problématique

à partir d'un signal discret, on souhaite obtenir des informations sur la valeur du signal à des endroits où il n'est pas connu



(x_i, f_i) : donnés

$\tilde{f}(x)$?

N.B. :
- (x_i, f_i) : « mesures »
- x : espace, temps, …
- Δx : régulier ou non
- le signal f peut être 1D, 2D, 3D, …, nD

surfaces (2D), volumes (3D)

1.2.

Méthodes

il existe plusieurs types de méthodes d'interpolation :

interpolation polynomiale globale :
interpolée de Lagrange

en tout point x, la fonction interpolée $f(x)$ dépend de **toutes** les mesures (x_i, f_i)

interpolation polynomiale par morceaux :
interpolée de type spline

en tout point x, la fonction interpolée $f(x)$ dépend seulement des mesures (x_i, f_i) pour x_i **« suffisamment proche »** de x

interpolation « trigonométrique » :
cf. transformée de Fourier

1. INTERPOLATION

1.3.

Interpolation de Lagrange

principe 1.3.1.

on **interpole** les n données connues (x_i, f_i) par un polynôme de degré (n–1) :

$$p(x) = \sum_{k=0}^{n-1} c_k x^k$$

on **recherche** donc : les coefficients c_k du polynôme $p(x)$ tels que :

$$\forall i,\ p(x_i) = f_i$$

1. INTERPOLATION

1.3.

Interpolation de Lagrange

principe 1.3.1.

théorème :

il existe un unique polynôme p de degré (n–1) tel que $\forall\ i, p(x_i) = f_i$.
il s'agit du polynôme d'interpolation de Lagrange qui s'écrit :

$$p(x) = \sum_{m=0}^{n-1} f_m L_m(x)$$

où $L_m(x)$ est le $m^{ième}$ polynôme élémentaire de Lagrange : $L_m(x) = \prod_{j=0, j\neq m}^{n-1} \frac{x - x_j}{x_m - x_j}$

preuve :

(i) il est évident que p est un polynôme et qu'il convient
(ii) il est unique car si il existait un autre polynôme $q(x)$ solution, alors p–q serait encore un polynôme de degré (n–1) possédant n racines (tous les x_i). Ce serait donc le polynôme nul

1. INTERPOLATION

1.3.

Interpolation de Lagrange

erreur d'interpolation 1.3.2.

pour caractériser l'erreur d'interpolation, on suppose que les valeurs f_i correspondent aux valeurs d'une fonction f C^n, aux points x_i : de sorte que l'on a $\forall\ i, f(x_i) = f_i$

N.B. : en pratique, cette fonction f est souvent inconnue et elle est, en fait, « approchée » par le polynôme d'interpolation de Lagrange

théorème :

– si $f \in C^n[a,b]$, et si p est son polynôme d'interpolation de Lagrange aux points x_j tels que $a \le x_1 < x_2 < \ldots < x_{n-1} < x_n \le b$
– alors l'erreur $e(x) = f(x) - p(x)$ est telle que :

$$\forall x \in [a,b], \exists \xi \in [a,b]:\ e(x) = \frac{1}{n!} \prod_{j=0}^{n-1} (x - x_j) f^{(n)}(\xi)$$

1. INTERPOLATION

1.3.

Interpolation de Lagrange

erreur d'interpolation 1.3.2.

preuve :

(i) si $\exists\, i$ tel que $x = x_i$, alors $e(x_i)$=0 qui est bien égal à $\frac{1}{n!}\prod_{j=0}^{n-1}(x-x_j)f^{(n)}(\xi)$

(ii) sinon, on définit la fonction suivante pour $t \neq x_i$:

$$\forall t \in [a,b],\ g(t) = f(t) - p(t) - e(x)\prod_{j=0}^{n-1}\frac{t-x_j}{x-x_j}$$

on a : • $g \in C^n[a,b]$
• $g(x) = 0$
• $g(x_i) = 0,\ \forall\, i \in [1..\, n]$

donc g est C^n et s'annule n+1 fois sur $[a,b]$

en appliquant le théorème de Rolle {si $f \in C^1[a,b]$ et $f(a) = f(b)$, alors $\exists\, c \in\,]a,b[$ t.q. $f'(c) = 0$} n fois, on montre que :

p : degré (n–1)

$$\exists \xi \in [a,b]:\ g^{(n)}(\xi) = 0 \qquad \text{or : } g^{(n)}(t) = f^{(n)}(t) - \cancel{p^{(n)}}(t) - e(x)\prod_{j=0}^{n-1}\frac{1}{x-x_j}\,n!$$

d'où : $e(x) = \frac{1}{n!}\prod_{j=0}^{n-1}(x-x_j)f^{(n)}(\xi)$

1. INTERPOLATION

1.3.

Interpolation de Lagrange

limitations : phénomène de Runge 1.3.3.

on cherche à interpoler sur [–5, +5] la fonction f suivante :

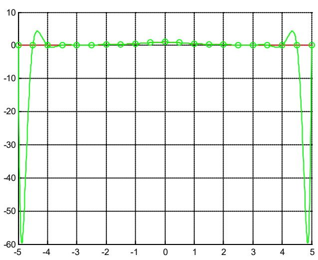
par le polynôme d'interpolation de Lagrange p_n de degré n

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

aux points $x_i = -5 + 10\frac{i}{n}$, avec i = 0, 1, …, n

et on estime l'erreur au point $x_{n-1/2} = 5 - \frac{5}{n}$

n = 20



n	$f(x_{n-1/2})$	$p_n(x_{n-1/2})$	$e(x_{n-1/2})$
2	0.138	0.760	-0.622
10	0.047	1.579	-1.532
20	0.042	-39.952	39.995

on observe que :

$$\lim_{n\to+\infty}\left\| f(x_{n-1/2}) - p_n(x_{n-1/2})\right\|_\infty = +\infty$$

1. INTERPOLATION

1.3.

Interpolation de Lagrange

limitations : phénomène de Runge 1.3.3.

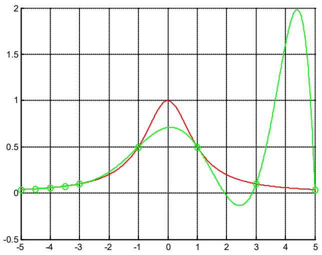
interprétation :

rappel : $e(x) = \frac{1}{n!} \prod_{j=0}^{n-1} (x - x_j) f^{(n)}(\xi)$

l'augmentation rapide de l'erreur au point $x_{n-1/2}$ est liée au terme $\prod_{j=0}^{n-1} (x_{n-1/2} - x_j)$

solution :

concentrer les points d'interpolation aux bornes de l'intervalle



$\Rightarrow$ une façon d'améliorer l'interpolation est de choisir les points d'interpolation de Tchebychev

1. INTERPOLATION

1.3.

Interpolation de Lagrange

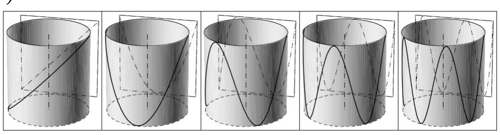
points de Tchebychev 1.3.4.

on montre que pour **minimiser** l'**erreur d'interpolation** (*cf.* phénomène de Runge), il faut choisir les **racines** des **polynômes de Tchebychev** comme **points d'interpolation**

les **polynômes de Tchebychev** sont définis par : $T_n(\cos\theta) = \cos(n\theta)$

i.e. : $T_n(x) = \cos(n \operatorname{acos} x)$ avec $-1 \le x \le 1$

illustration : guirlandes $z = \cos(n\theta)$ autour d'un tambour $x = \cos(\theta)$, $y = \sin(\theta)$ et leur projections $T_n(x)$ dans le plan (x, z)



propriétés :

– on obtient les coefficients de $T_n(x)$ en développant $\cos(n\theta)$ en puissances de $\cos\theta$ et en remplaçant $\cos\theta$ par x

– on obtient la formule de récurrence suivante : $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$

en utilisant la relation : $\cos p + \cos q = 2\cos\frac{p+q}{2}\cos\frac{p-q}{2}$, avec : $\begin{cases} p = (n+1)\theta \\ q = (n-1)\theta \end{cases}$

1. INTERPOLATION

1.3.

1. INTERPOLATION

Interpolation de Lagrange

points de Tchebychev 1.3.4.

remarque : le maximum de $T_n(x) = \cos(n\theta)$, vaut 1, avec $x = \cos(\theta)$

si on choisit des points x_i tels que : $\prod_{j=0}^{n-1}(x - x_j) = \alpha T_n(x)$

alors l'erreur $e(x) = \frac{1}{n!}\prod_{j=0}^{n-1}(x - x_j) f^{(n)}(\xi)$ sera limitée $\forall$ n

dans ce cas, les x_i sont nécessairement racines de $T_n(x)$, d'où :

abscisses des points de Tchebychev :

$$x_i = \cos\left(\frac{(2(n-i)-1)\pi}{2n}\right) \text{ , avec i = 0, 1, …, n–1}$$

pour travailler sur un intervalle quelconque, on effectue la transformation affine suivante :

$$x_i = \lambda + \mu\cos\left(\frac{(2(n-i)-1)\pi}{2n}\right) \text{ , avec i = 0, 1, …, n–1 et } \begin{cases} \lambda = \frac{1}{2}(a+b) \\ \mu = \frac{1}{2}(b-a) \end{cases}$$

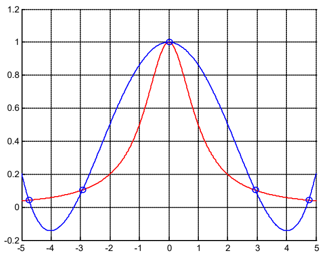
1.3.

1. INTERPOLATION

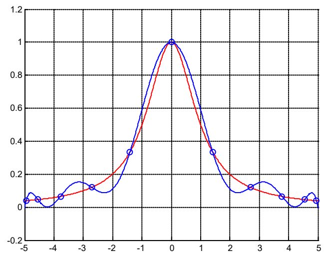
Interpolation de Lagrange

points de Tchebychev 1.3.4.

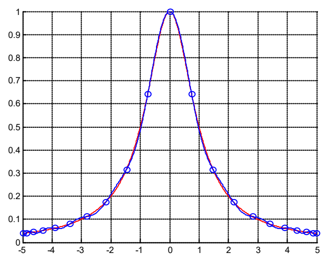
n = 5



n = 10



n = 20



1.3.

Interpolation de Lagrange

algorithme des différences divisées 1.3.5.

définition :

notion de complexité arithmétique : ordre de grandeur du nombre d'opérations nécessaires pour effectuer un calcul, en fonction de la « taille » du problème à traiter

illustration pour le calcul de $p(x)=\sum_{m=0}^{n-1} f_m L_m(x)$, avec : $L_m(x)=\prod_{j=0,\, j\neq m}^{n-1} \frac{x-x_j}{x_m-x_j}$

calcul de L_m : 2(n–1) soustractions + (n–1) divisions + (n–1) multiplications $\Rightarrow O(n)$

calcul de p : n calculs de L_m + n multiplications + n additions $\Rightarrow O(n^2)$

existe-t-il une méthode plus performante pour calculer p ?

1. INTERPOLATION

1.3.

Interpolation de Lagrange

algorithme des différences divisées 1.3.5.

définition :

soit (n+1) couples de valeurs (x_i , f_i), avec tous les x_i distincts (pas forcément ordonnés)
on définit les n différences divisées de la manière suivante :

$$f[x_i]=f_i$$

$$\delta f[x_i,x_j]=\frac{f[x_j]-f[x_i]}{x_j-x_i}$$

$$\delta^2 f[x_i,x_j,x_k]=\frac{\delta f[x_j,x_k]-\delta f[x_i,x_j]}{x_k-x_i}$$

$$\delta^n f[x_{i_0},x_{i_1},\dots,x_{i_n}]=\frac{\delta^{n-1} f[x_{i_1},x_{i_2},\dots,x_{i_n}]-\delta f^{n-1}[x_{i_0},x_{i_1},\dots,x_{i_{n-1}}]}{x_{i_n}-x_{i_0}}$$

1. INTERPOLATION

1.3.

Interpolation de Lagrange

algorithme des différences divisées 1.3.5.

théorème (formule de Newton) :

le polynôme d'interpolation (de degré n) passant par les n+1 couples de points (x_i, f_i) avec $i = 0, 1, \dots, n$, est unique et est donné par :

$$p(x) = f[x_0] + (x - x_0)\delta f[x_0, x_1] + (x - x_0)(x - x_1)\delta^2 f[x_0, x_1, x_2]$$
$$+ \dots + (x - x_0)(x - x_1)\dots(x - x_{n-1})\delta^n f[x_0, x_1, x_2, \dots, x_n]$$

preuve : par récurrence …

(*i*) pour n = 1 : $p(x) = f_0 + (x - x_0)\dfrac{y_1 - y_0}{x_1 - x_0} \quad \Rightarrow$ O.K.

(*ii*) si on suppose que p_1 est le polynôme de degré (n–1) associé aux n premiers couples :

$$p_1(x) = f[x_0] + (x - x_0)\delta f[x_0, x_1] + (x - x_0)(x - x_1)\delta^2 f[x_0, x_1, x_2] + \dots + (x - x_0)(x - x_1)\dots(x - x_{n-2})\delta^{n-1} f[x_0, x_1, \dots, x_{n-1}]$$

alors on peut écrire p sous la forme : $p(x) = \underbrace{p_1(x)}_{\text{degré n–1}} + \underbrace{a(x - x_0)(x - x_1)\dots(x - x_{n-1})}_{\text{degré n}}$

où a est déterminé de telle sorte que $p(x_n) = f_n$

1. INTERPOLATION

1.3.

Interpolation de Lagrange

algorithme des différences divisées 1.3.5.

preuve (suite) :

il reste donc à montrer que : $a = \delta^n f[x_0, x_1, x_2, \dots, x_n]$

pour cela on considère le polynôme de degré (n–1) suivant, passant par les n derniers couples (x_i, f_i) avec i = 1, 2, …, n :

$$p_2(x) = f[x_1] + (x - x_1)\delta f[x_1, x_2] + (x - x_1)(x - x_2)\delta^2 f[x_1, x_2, x_3] + \dots + (x - x_1)(x - x_2)\dots(x - x_{n-1})\delta^{n-1} f[x_1, x_2, \dots, x_n]$$

on pose ensuite : $q(x) = \dfrac{1}{x_n - x_0}\left[(x_n - x)p_1(x) - (x_0 - x)p_2(x)\right]$

qui est un polynôme de degré n tel que : $q(x_0) = p_1[x_0] = f[x_0] = f_0$

$$q(x_n) = p_2(x_n) = f_n$$

$q(x_i) = f_i$, car $p_1(x_i) = p_2(x_i) = f_i$, $\forall$ i = 1, 2, …, n–1

q vérifie la condition recherchée : il est donc solution
ce polynôme est unique (de par l'unicité du polynôme d'interpolation)

1. INTERPOLATION

1.3.

Interpolation de Lagrange

algorithme des différences divisées 1.3.5.

preuve (suite) :

$$\text{on a : } q(x)=\frac{1}{x_n-x_0}\left[(x_n-x)p_1(x)-(x_0-x)p_2(x)\right] \quad \text{, avec :}$$

$$p_1(x)=f[x_0]+(x-x_0)\delta f[x_0,x_1]+(x-x_0)(x-x_1)\delta^2 f[x_0,x_1,x_2]+\ldots+(x-x_0)(x-x_1)\ldots(x-x_{n-2})\delta^{n-1}f[x_0,x_1,\ldots,x_{n-1}]$$

$$p_2(x)=f[x_1]+(x-x_1)\delta f[x_1,x_2]+(x-x_1)(x-x_2)\delta^2 f[x_1,x_2,x_3]+\ldots+(x-x_1)(x-x_2)\ldots(x-x_{n-1})\delta^{n-1}f[x_1,x_2,\ldots,x_n]$$

$$\text{et on a aussi : } q(x)=p_1(x)+a(x-x_0)(x-x_1)\ldots(x-x_{n-1})$$

$$\text{soit : } q(x)=\frac{1}{x_n-x_0}\left[(x_n-x_0+x_0-x)p_1(x)-(x_0-x)p_2(x)\right]$$

$$\text{c.à.d. : } q(x)=\underbrace{p_1(x)}_{n-1}+\frac{1}{x_n-x_0}\left[\underbrace{(x-x_0)p_2(x)}_{n}-\underbrace{(x-x_{n-1})p_1(x)}_{n}-\underbrace{(x_{n-1}-x_0)p_1(x)}_{n-1}\right]$$

le coefficient de degré n de q s'écrit donc :

$$\frac{(x-x_0)(x-x_1)(x-x_2)\ldots(x-x_{n-2})(x-x_{n-1})}{(x_n-x_0)}\left[\delta^{n-1}f[x_1,x_2,\ldots,x_n]-\delta^{n-1}f[x_0,x_1,\ldots,x_{n-1}]\right]$$

$$\text{d'où : } a=\frac{1}{(x_n-x_0)}\left[\delta^{n-1}f[x_1,x_2,\ldots,x_n]-\delta^{n-1}f[x_0,x_1,\ldots,x_{n-1}]\right]=\delta^n f[x_0,x_1,\ldots,x_{n-1},x_n]$$

C.Q.F.D.

1. INTERPOLATION

1.3.

Interpolation de Lagrange

algorithme des différences divisées 1.3.5.

exemple :

x_i	$f[x_i]=f_i$	δf	$\delta^2 f$	$\delta^3 f$	$\delta^4 f$	$\delta^5 f$
0	−1	1	3/8	−77/120	167/960	−287/9600
2	1	5/2	−17/6	3/4	−1/8	
4	6	−6	5/3	−1/4		
5	0	2/3	1/6			
8	2	3/2				
10	5					

$$p(x)=-1+(x-0)1+(x-0)(x-2)\frac{3}{8}+(x-0)(x-2)(x-4)\left(-\frac{77}{120}\right)+(x-0)(x-2)(x-4)(x-5)\frac{167}{960}+(x-0)(x-2)(x-4)(x-5)(x-8)\left(-\frac{287}{9600}\right)$$

i.e. : $p(x)=-1+(x-0)\left(1+(x-2)\left(\frac{3}{8}+(x-4)\left(-\frac{77}{120}+(x-5)\left(\frac{167}{960}+(x-8)\left(-\frac{287}{9600}\right)\right)\right)\right)\right)$

1. INTERPOLATION

1.4.

Interpolation par des polynômes composites

principe 1.4.1.

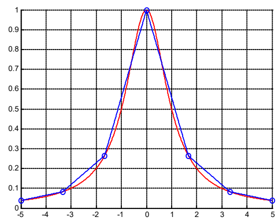
pour limiter le phénomène de Runge, on peut aussi découper l'intervalle d'étude et interpoler le signal par des polynômes sur chaque sous-intervalle

– on obtient ainsi une fonction d'interpolation continue par morceaux
– par contre les dérivées ne sont pas forcément continues d'un sous-intervalle à l'autre

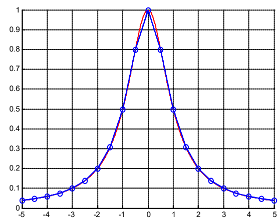
exemple :

interpolation linéaire par morceaux

n = 7



n = 21



1. INTERPOLATION

Chap. 5 Interpolation et Approximation 18

1.4.

Interpolation par des polynômes composites

interpolation d'Hermite 1.4.2.

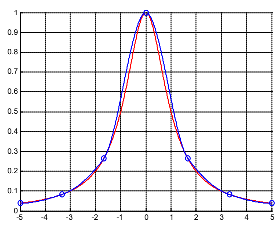
on peut aussi rajouter des conditions sur le polynôme d'interpolation pour assurer la continuité de la dérivée de la fonction d'interpolation

sur chaque intervalle $[x_i, x_{i+1}]$, on cherche un polynôme p de degré 3 tel que :

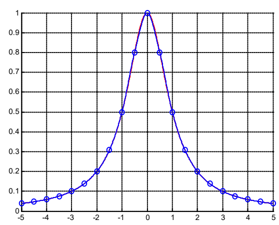
$$p(x_i) = f(x_i)$$
$$p'(x_i) = f'(x_i)$$
$$p(x_{i+1}) = f(x_{i+1})$$
$$p'(x_{i+1}) = f'(x_{i+1})$$

on montre que ce polynôme est **unique** et que l'on peut **modifier** l'algorithme des **différences divisées** pour calculer ce polynôme pour chaque intervalle

n = 7



n = 21



1. INTERPOLATION

Chap. 5 Interpolation et Approximation 19

1.5.

Interpolation par splines

principe 1.5.1.

spline : latte souple en bois utilisée en menuiserie pour tracer des courbes lisses et « quelconques »

pour des points (x_i , f_i) avec des abscisses ordonnées, on cherche une fonction s
$s : [a, b\,] \rightarrow \mathbb{R}$, avec $a = x_0$ et $b = x_n$, satisfaisant les 3 conditions suivantes :

(*i*) $s(x_i) = f_i$, $\forall$ i = 0, …, n
(*ii*) $s(x)$ est 2 fois continûment dérivable (*i.e.* de classe C^2)

(*iii*) $\int_a^b [s''(x)]^2 \mathrm{d}x$ minimale

N.B. :
l'intégrale (*iii*) représente « l'énergie de déformation » de la latte déformée en flexion (qui doit être minimale, *cf.* cours R.d.M.)

1. INTERPOLATION

1.5.

Interpolation par splines

définition spline cubique 1.5.2.

– soit $a = x_0 < x_1 < \ldots < x_n = b$ une suite d'indices ordonnés
– une fonction $s : [a, b\,] \rightarrow \mathbb{R}$ s'appelle une spline (cubique) si :
elle est 2 fois continûment différentiable (*i.e.* C^2)
elle est un polynôme de degré 3 sur chaque intervalle $[x_i , x_{i+1}]$

N.B. : il existe **plusieurs catégories** de **splines cubiques** :
– les splines naturels, tels que :
$s''(a) = 0$ et $s''(b) = 0$
– les splines scellés, tels que :
$s'(a) = f\,'(x_0)$ et $s'(b) = f\,'(x_n)$
– les splines périodiques, tels que :
$s'(a) = s'(b)$ et $s''(a) = s''(b)$

il existe aussi de **nombreux types** de **splines** :
quintique, B-spline (Bézier), …

1. INTERPOLATION

1. INTERPOLATION

1.6.

Interpolation par TF

principe 1.6.1.

on utilise la **propriété de translation** de la **transformée de Fourier** :

soit $f \in L^1(\mathbb{R})$ et $a \in \mathbb{R}$, on définit $\tau_a[f](x) = f(x - a)$

$$\text{alors : } \mathcal{F}\big[\tau_a[f]\big](\xi) = e^{-2i\pi\xi a}\,\mathcal{F}\big[f\big](\xi)$$

que l'on applique à la **transformée de Fourier discrète** :

$$\tau_a[f] = f(x-a) = \mathrm{TFD}^{-1}\Big[e^{-2i\pi\xi a}\,\mathrm{TFD}[f]\Big]$$

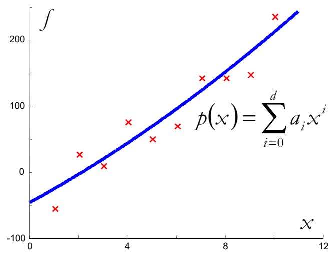
2. APPROXIMATION

2.1.

Problématique

on a vu qu'en présence de **nombreuses données**, l'**interpolation** (et particulièrement l'interpolation de Lagrange) **peut générer** des **oscillations « parasites »** importantes (*cf.* ex. Runge)

– pour **limiter ces oscillations**, on a déjà proposé d'utiliser : des polynômes composites, des splines, …
– on peut aussi chercher un **polynôme** de degré fixé (donc indépendant du nombre de données) qui **approche « au mieux »** les données



N.B. :
dans certaines situations (présence de **bruit** sur les données, …), il peut aussi être nécessaire d'**assouplir** les **contraintes** liées à l'**interpolation** (passage par tous les points de contrôle) en approchant « au mieux » les données

2.2.

Définition

– on suppose que l'on dispose de (n+1) jeux de données (x_i, y_i) distincts

– on appelle polynôme aux moindres carrés $p_d(x)$ le polynôme de degré d tel que :

ensemble des polynômes de degré d

$$\sum_{i=0}^{n} \left| y_i - p_d(x_i) \right|^2 \leq \sum_{i=0}^{n} \left| y_i - q_d(x_i) \right|^2 \qquad , \forall q_d(x) \in P_d$$

c'est le polynôme de degré d qui, parmi tous les polynômes de degré d, minimise la distance avec les données

2. APPROXIMATION

2.3.

Détermination

– on écrit le polynôme p_d de la façon suivante :

$$p_d(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_d x^d = \sum_{j=0}^{d} a_j x^j$$

– on définit la fonctionnelle F par :

$$F(a_0, a_1, \ldots, a_d) = \sum_{i=0}^{n} \left| y_i - \left(a_0 + a_1 x_i + a_2 x_i^2 + \ldots + a_d x_i^d \right) \right|^2$$

soit : $$F(a_0, a_1, \ldots, a_d) = \sum_{i=0}^{n} \left| y_i - \left(\sum_{j=0}^{d} a_j x_i^j \right) \right|^2$$

– F est continue, dérivable : elle est minimale si son gradient est nul. Soit :

$$\frac{\partial F}{\partial a_k} = 0$$

ce qui correspond à la résolution d'un système linéaire :
de (d+1) équations (1 pour chaque valeur de k)
de (d+1) inconnues (tous les j)

$\Rightarrow$ **solution unique**

2. APPROXIMATION

2.4.

2. APPROXIMATION

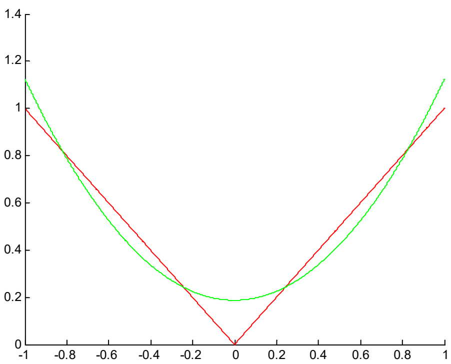
Illustration

– on va illustrer le calcul du polynôme d'approximation sur un exemple où les données sont continues (on remplace simplement la somme discrète sur les données par une intégrale)

– calcul … tion $|x|$ sur l'inter…

x

$$\begin{cases} \dfrac{\partial F}{\partial a} = -2\displaystyle\int_{-1}^{1} x^2 \big[\,|\ \ldots\ {}^4 + bx^3 + cx^2\big)\mathrm{d}x \\ \dfrac{\partial F}{\partial b} = -2\displaystyle\int_{-1}^{1} x \big[\,|x\ \ldots\ + bx^2 + cx\big)\mathrm{d}x \\ \dfrac{\partial F}{\partial c} = -2\displaystyle\int_{-1}^{1} \big[\,|x|\ \ldots\ bx + c\big)\mathrm{d}x \end{cases}$$



soit : $\begin{pmatrix} 2/\ldots & & \\ 0 & 2/3 & 0 \\ 2/3 & 0 & 2 \end{pmatrix}\begin{pmatrix} \ldots \\ b \\ c \end{pmatrix} = \begin{pmatrix} \ldots \\ 0 \\ 1 \end{pmatrix}$ d'où : $\begin{pmatrix} \ldots \\ b \\ c \end{pmatrix} = \begin{pmatrix} \ldots \\ 0 \\ 9/48 \end{pmatrix}$

2.5.

2. APPROXIMATION

Généralisation

on peut **généraliser** la notion d'approximation à d'autres fonctions que les polynômes

on distingue :

– les **approximations linéaires** : les fonctions d'approximation sont linéaires par rapport aux coefficients d'approximation

$\Rightarrow$ elles se traduisent par une résolution de système linéaire (toujours possible)

ex :

interpolations trigonométrique, logarithmique, exponentielle … du type :

$$a\sin(2x) + b\cos^2(5x)\ ,\quad a\ln\left(\frac{2x}{x^2+8}\right) + be^{-x^2}\ ,\ \ldots$$

– les **approximations non linéaires** : les fonctions d'approximation ne sont pas linéaires par rapport aux coefficients d'approximation

$\Rightarrow$ elles se traduisent par une résolution de système non linéaire (pas toujours simple …)

ex :

$$a\sin(bx+1) + ce^{d\cos 3x}$$