



HAL
open science

Fonctions splines avec conditions de forme

Julio Medina

► **To cite this version:**

Julio Medina. Fonctions splines avec conditions de forme. Modélisation et simulation. Université Joseph-Fourier - Grenoble I, 1985. Français. NNT: . tel-00316060

HAL Id: tel-00316060

<https://theses.hal.science/tel-00316060>

Submitted on 2 Sep 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THESE

présentée par

Julio MEDINA

pour obtenir le titre de DOCTEUR

de l'UNIVERSITE SCIENTIFIQUE ET MEDICALE DE GRENOBLE

Mathématiques Appliquées

FONCTIONS SPLINES AVEC

CONDITIONS DE FORME

Thèse soutenue le 9 Octobre 1985 devant la Commission d'Examen:

Monsieur	F. ROBERT	: Président
Messieurs	C. CARASSO	} Examineurs
	M. DUC-JACQUET	
	B. LACOLLE	
	P.J. LAURENT	

UNIVERSITE SCIENTIFIQUE ET MEDICALE DE GRENOBLE

Année universitaire 1982-1983

Président de l'Université : M. TANCHE

MEMBRES DU CORPS ENSEIGNANT DE L'U.S.M.G.

(RANG A)

SAUF ENSEIGNANTS EN MEDECINE ET PHARMACIE

PROFESSEURS DE 1ère CLASSE

ARNAUD Paul	Chimie organique
ARVIEU Robert	Physique nucléaire I.S.N.
AUBERT Guy	Physique C.N.R.S.
AYANT Yves	Physique approfondie
BARBIER Marie-Jeanne	Electrochimie
BARBIER Jean-Claude	Physique expérimentale C.N.R.S. (labo de magnétisme)
BARJON Robert	Physique nucléaire I.S.N.
BARNOUD Fernand	Biosynthèse de la cellulose-Biologie
BARRA Jean-René	Statistiques - Mathématiques appliquées
BELORISKY Elie	Physique
BENZAKEN Claude (M.)	Mathématiques pures
BERNARD Alain	Mathématiques pures
BERTRANDIAS Françoise	Mathématiques pures
BERTRANDIAS Jean-Paul	Mathématiques pures
BILLET Jean	Géographie
BONNIER Jean-Marie	Chimie générale
BOUCHEZ Robert	Physique nucléaire I.S.N.
BRAVARD Yves	Géographie
CARLIER Georges	Biologie végétale
CAUQUIS Georges	Chimie organique
CHIBON Pierre	Biologie animale
COLIN DE VERDIERE Yves	Mathématiques pures
CRABBE Pierre (détaché)	C.E.R.M.O.
CYROT Michel	Physique du solide
DAUMAS Max	Géographie
DEBELMAS Jacques	Géologie générale
DEGRANGE Charles	Zoologie
DELOBEL Claude (M.)	M.I.A.G. Mathématiques appliquées
DEPORTES Charles	Chimie minérale
DESRE Pierre	Electrochimie
DOLIQUE Jean-Michel	Physique des plasmas
DUCROS Pierre	Cristallographie
FONTAINE Jean-Marc	Mathématiques pures
GAGNAIRE Didier	Chimie physique

.../...

GASTINEL Noël	Analyse numérique - Mathématiques appliquées
GERBER Robert	Mathématiques pures
GERMAIN Jean-Pierre	Mécanique
GIRAUD Pierre	Géologie
IDELMAN Simon	Physiologie animale
JANIN Bernard	Géographie
JOLY Jean-René	Mathématiques pures
JULLIEN Pierre	Mathématiques appliquées
KAHANE André (détaché DAFCO)	Physique
KAHANE Josette	Physique
KOSZUL Jean-Louis	Mathématiques pures
KRAKOWIAK Sacha	Mathématiques appliquées
KUPTA Yvon	Mathématiques pures
LACAZE Albert	Thermodynamique
LAJZEROWICZ Jeannine	Physique
LAJZEROWICZ Joseph	Physique
LAURENT Pierre	Mathématiques appliquées
DE LEIRIS Joël	Biologie
LLIBOUTRY Louis	Géophysique
LOISEAUX Jean-Marie	Sciences nucléaires I.S.N.
LOUP Jean	Géographie
MACHE Régis	Physiologie végétale
MAYNARD Roger	Physique du solide
MICHEL Robert	Minéralogie et pétrographie (géologie)
MOZIERES Philippe	Spectrométrie - Physique
OMONT Alain	Astrophysique
OZENDA Paul	Botanique (biologie végétale)
PAYAN Jean-Jacques (détaché)	Mathématiques pures
PEBAY PEYROULA Jean-Claude	Physique
PERRIAUX Jacques	Géologie
PERRIER Guy	Géophysique
PIERRARD Jean-Marie	Mécanique
RASSAT André	Chimie systématique
RENARD Michel	Thermodynamique
RICHARD Lucien	Biologie végétale
RINAUDO Marguerite	Chimie CERMAV
SENGEL Philippe	Biologie animale
SERGERAERT Francis	Mathématiques pures
SOUTIF Michel	Physique
VAILLANT François	Zoologie
VALENTIN Jacques	Physique nucléaire I.S.N.
VAN CUTSEN Bernard	Mathématiques appliquées
VAUQUOIS Bernard	Mathématiques appliquées
VIALON Pierre	Géologie

PROFESSEURS DE 2^{ème} CLASSE

ADIBA Michel	Mathématiques pures
ARMAND Gilbert	Géographie

.../...

AURIAULT Jean-Louis	Mécanique
BEGUIN Claude (M.)	Chimie organique
BOEHLER Jean-Paul	Mécanique
BOITET Christian	Mathématiques appliquées
BORNAREL Jean	Physique
BRUN Gilbert	Biologie
CASTAING Bernard	Physique
CHARDON Michel	Géographie
COHENADDAD Jean-Pierre	Physique
DENEUVILLE Alain	Physique
DEPASSEL Roger	Mécanique des fluides
DOUCE Roland	Physiologie végétale
DUFRESNOY Alain	Mathématiques pures
GASPARD François	Physique
GAUTRON René	Chimie
GIDON Maurice	Géologie
GIGNOUX Claude (M.)	Sciences nucléaires I.S.N.
GUITTON Jacques	Chimie
HACQUES Gérard	Mathématiques appliquées
HERBIN Jacky	Géographie
HICTER Pierre	Chimie
JOSELEAU Jean-Paul	Biochimie
KERCKOVE Claude (M.)	Géologie
LE BRETON Alain	Mathématiques appliquées
LONGUEUEU Nicole	Sciences nucléaires I.S.N.
LUCAS Robert	Physiques
LUNA Domingo	Mathématiques pures
MASCLE Georges	Géologie
NEMOZ Alain	Thermodynamique (CNRS - CRTBT)
OUDET Bruno	Mathématiques appliquées
PELMONT Jean	Biochimie
PERRIN Claude (M.)	Sciences nucléaires I.S.N.
PFISTER Jean-Claude (détaché)	Physique du solide
PIBOULE Michel	Géologie
PIERRE Jean-Louis	Chimie organique
RAYNAUD Hervé	Mathématiques appliquées
ROBERT Gilles	Mathématiques pures
ROBERT Jean-Bernard	Chimie physique
ROSSI André	Physiologie végétale
SAKAROVITCH Michel	Mathématiques appliquées
SARROT REYNAUD Jean	Géologie
SAXOD Raymond	Biologie animale
SOUTIF Jeanne	Physique
SCHOOL Pierre-Claude	Mathématiques appliquées
STUTZ Pierre	Mécanique
SUBRA Robert	Chimie
VIDAL Michel	Chimie organique
VIVIAN Robert	Géographie

à flaqlts

**porque sós mi amor ,
mi cómplice y todo**



Je remercie Monsieur le Professeur F. ROBERT pour l'honneur qu'il me fait en présidant le Jury de cette thèse.

J'exprime toute ma reconnaissance à Monsieur le Professeur P.J. LAURENT qui a dirigé mes recherches depuis que j'étais étudiant au DEA. Son attention, ses conseils et ses encouragements constants ont contribué énormément à l'accomplissement de ce travail. Son aide m'a été précieuse et je lui dois beaucoup.

Monsieur le Professeur C. CARASSO a eu l'amabilité de se déplacer depuis Saint Etienne pour juger cette thèse, qu'il en soit vivement remercié.

Que soient aussi remerciés Messieurs M. DUC-JACQUET, Professeur, et B. LACOLLE, Maître de Conférences, pour avoir accepté de faire partie de ce Jury.

Je remercie également les membres de l'Equipe d'Algorithmique Mathématique; de manière particulière P. KLEIN, J.M. MULLER et D. PELLEGRIN, mes camarades du bureau 58 de la Tour de l'IRMA, pour tous les bons moments partagés : *gracias muchachos!*

Mme. C. RIVET a frappé ce texte, Mlle. R. RUIZ m'a aidé avec patience dans les corrections, Mr. D. IGLESIAS et le service reprographie de l'IMAG ont assuré la réalisation matérielle de ce travail. Qu'ils trouvent ici l'expression de ma gratitude.

Une amicale pensée pour les joyeux drilles de la "colonie" équatorienne de Grenoble.

TABLE DES MATIERES

INTRODUCTION GENERAL	1
CHAPITRE I . Splines polynomiales pour l'interpolation des données monotones et convexes	7
I.1. Introduction	9
I.2. Notations et définitions	10
I.3. Interpolation monotone et convexe par splines quadratiques	11
I.4. Interpolation cubique par morceaux des données monotones et convexes	27
I.5. Courbes paramétrées planes qui conservent la forme	36
I.6. Essais numériques	42
I.7. Conclusions	55
CHAPITRE II. Interpolation rationnelle pour données monotones et convexes	57
II.1. Introduction	59
II.2. Fonctions splines homographiques monotones et convexes	60
II.3. Fonctions splines rationnelles quadratiques monotones	73
II.4. Fonctions splines rationnelles (2/1) convexe	77
II.5. Essais numériques	85
II.6. Conclusions	98
CHAPITRE III. Fonctions splines pénalisées et sous tension	99
III.1. Introduction	101
III.2. Fonctions splines hyperboliques sous tension	103
III.3. Fonctions splines rationnelles sous tension	117
III.4. Fonctions ν -splines	119
III.5. Fonctions splines cubiques avec poids	130

III.6. Fonctions splines cubiques pénalisées	136
III.7. Le choix des paramètres de tension	157
III.8. Le choix du paramètre de lissage	159
III.9. Essais numériques	165
III.10. Conclusions	186

**CHAPITRE IV. Fonctions splines avec contraintes de
forme via la programmation
mathématique** **187**

IV.1. Introduction	189
IV.2. Interpolation et lissage des données avec contrainte de monotonie	192
IV.3. Lissage de données avec contraintes de convexité locale	196
IV.4. Fonctions splines avec contraintes linéaires	202
IV.5. Essais numériques	221
IV.6. Conclusions	238

CHAPITRE V. Fonctions splines avec repoussoir **239**

V.1. Introduction	241
V.2. Etude d'un problème équivalent au problème (P_0)	242
V.3. Un problème dual associé à (P)	244
V.4. Algorithmes de descente pour la résolution de (P) et (Q)	247
V.5. Essais numériques	252
V.6. Conclusion	262

CONCLUSION GENERALE **263**

REFERENCES **267**

INTRODUCTION GENERALE



Dans de multiples problèmes d'interpolation et d'ajustement de données, il est important que la solution préserve certaines propriétés de forme de la fonction sous-jacente aux données.

En général, les méthodes classiques (les fonctions splines polynomiales étant parmi les plus utilisées) ne tiennent pas compte de ce type de conditions. Ces fonctions peuvent présenter des inflexions ou des oscillations indésirables et faire perdre à la courbe ses caractéristiques de positivité, de monotonie ou de convexité.

C'est ainsi que dans les derniers temps l'approximation et l'interpolation avec contraintes ont été l'objet de nombreuses recherches.

Notre travail est une contribution à l'étude de la théorie et des algorithmes liés au problème suivant :

Problème (P)

Soit $D = \{(t_i, z_i)\}_{i=1,2,\dots,n}$, un ensemble de données dans le plan réel. On cherche une fonction s qui satisfasse les conditions suivantes :

- (i) s interpole (ou lisse) les données
- (ii) s est, au moins, de classe C^1
- (iii) s préserve la forme de la fonction sous-jacente aux données.

Si les valeurs z_i sont exactes, le problème est d'interpolation ; sinon de lissage.

Les conditions de forme que nous considérons sont : positivité, monotonie, convexité.

Selon les différentes approches et types de fonctions utilisées pour représenter s nous avons classé en cinq groupes les méthodes pour résoudre (P):

1. Méthodes fondées sur les splines polynomiales
2. Méthodes fondées sur les splines rationnelles
3. Méthodes des splines sous tension
4. Méthodes dérivées de la programmation mathématique
5. Méthodes des splines avec repoussoir.

Les deux premiers types de méthodes s'appliquent exclusivement à l'interpolation des données, tandis que les autres assurent des propriétés variationnelles et permettent de traiter aussi le lissage.

Dans le dessin graphique assisté par ordinateur on dispose d'autres méthodes, par exemple celles de Bernstein-Bézier. L'utilisateur procède d'une manière interactive jusqu'à trouver la forme souhaitée de la courbe. Nous ne traiterons pas ici ces méthodes, sauf un cas particulier à la fin du premier chapitre.

Nous avons consacré un chapitre à chaque groupe de méthodes, en gardant l'ordre du classement ci-dessus.

L'idée de base des méthodes dont l'interpolante est formée par morceaux de fonctions polynomiales ou rationnelles est la suivante : on cherche des conditions nécessaires et suffisantes pour que la spline d'interpolation soit monotone ou convexe. A partir de ces résultats on dégage des algorithmes pour calculer les dérivées aux noeuds et, éventuellement, des points supplémentaires qu'on insère entre les données.

Dans le premier chapitre nous faisons un tour d'horizon des méthodes d'interpolation monotone et convexe par splines quadratiques et cubiques.

Dans le chapitre deux, nous construisons à partir de données monotones des splines homographiques monotones de classe C^1 , et pour conserver la convexité nous proposons d'utiliser des splines rationnelles (2/1) continûment différentiables.

La notion de tension provient de l'idée que si l'on tire aux deux bouts sur le graphe de la spline cubique on peut faire disparaître les oscillations indésirables. A la limite, on s'approche de la polygonale qui relie les données. Selon la manière d'appliquer la tension, on obtient des splines hyperboliques, rationnelles ou, encore, des cubiques. Nous suggérons d'introduire dans la

fonctionnelle à minimiser un terme de pénalisation des dérivées premières aux noeuds. On obtient encore des fonctions splines cubiques telles que , à la limite, leurs dérivées aux noeuds convergent vers une approximation locale d'ordre 2. Ceci est développé dans le troisième chapitre.

Des techniques de la programmation linéaire, quadratique et convexe avec contraintes ont été appliquées pour résoudre le problème (P). La mise en oeuvre numérique de l'algorithme de Laurent [57] pour minimiser une fonction fortement convexe avec contraintes linéaires, appliqué au lissage avec contrainte de positivité ou de monotonie est le sujet central du chapitre quatre.

On remarque que lorsque la contrainte de forme devient active, alors la positivité, monotonie ou convexité de la fonction n'est pas stricte dans certains intervalles et des oscillations parasites peuvent apparaître. Ainsi par exemple, sous la condition de positivité la spline deviendra égale à zéro en certains points. Pour éviter ces situations il faut dresser une "barrière", une pénalisation de type intérieur qui empêche la contrainte d'être active. Cette idée est nouvelle et nous l'avons développée pour le lissage de données avec la propriété de positivité. Ces splines nous les appelons : *fonctions splines avec repoussoir*. Ceci, fait l'objet du cinquième chapitre.

Chaque chapitre contient aussi les résultats numériques et graphiques des essais réalisés avec les différents algorithmes. Les méthodes ont été mises sous la forme de procédures Pascal et testées sur un micro-ordinateur Micral 9050.

CHAPITRE I

Splines Polynomiales pour l'Interpolation des Données Monotones et Convexes

I.1. INTRODUCTION

Le but de ce chapitre est de faire un tour d'horizon de plusieurs méthodes polynomiales pour l'interpolation des données monotones et convexes. Ces différentes approches ne sont pas de type variationnel.

Si $\Delta : a = t_1 < t_2 < \dots < t_n = b$ est une partition de l'intervalle $[a,b]$, et k et m deux entiers non-négatifs tels que $k \leq m$, nous noterons $S(m,k,\Delta)$ ou $S(m,k)$ **l'espace des fonctions splines polynomiales de degré m et de déficience $m-k$ relatives à Δ** . C'est-à-dire, $s \in S(m,k)$ si et seulement si :

- (i) $s \in C^k[a,b]$
- (ii) s est un polynôme de degré $\leq m$ sur chaque intervalle $[t_i, t_{i+1}]$
 $i = 1, 2, \dots, n-1$.

Un nombre considérable de travaux a été consacré à la détermination des conditions nécessaires et suffisantes pour l'existence des splines polynomiales qui préservent la forme des données. Différents auteurs ([65],[86]) ont montré que, dans certains cas, pour des données monotones ou convexes et pour m et k données, il peut être impossible de construire des splines monotones ou convexes de degré m et déficience $m-k$. D'autres chercheurs ([73],[78],[79]) ont trouvé des conditions supplémentaires sur des données monotones ou convexes pour que la solution qui reflète les caractéristiques de forme appartienne à certains espaces de splines polynomiales.

Plusieurs études ont abouti à l'élaboration d'algorithmes de construction numérique. En mettant à profit la simplicité de calcul et la souplesse de la forme, ces solutions ont été cherchées surtout parmi les splines cubiques et quadratiques, en exigeant qu'elles aient au moins la continuité C^1 .

Pour représenter ces splines dans l'intervalle compris entre deux noeuds consécutifs on a besoin de connaître les valeurs de la fonction et des dérivées aux noeuds et, éventuellement, l'information qui provient des noeuds supplémentaires qui sont insérés. Alors les algorithmes proposés calculent les dérivées aux abscisses et, si c'est le cas, aux points supplémentaires.

Dans le troisième paragraphe de ce chapitre nous discutons les algorithmes pour les splines quadratiques. L'utilisation des splines cubiques est étudiée dans le quatrième paragraphe. Dans les deux cas on suppose que l'interpolante est une fonction à une variable mais il arrive souvent que la fonction doit être considérée en forme paramétrée. La cinquième partie traite de ce dernier problème. Les résultats numériques et graphiques sur différents exemples sont donnés dans un dernier paragraphe.

1.2. NOTATIONS ET DEFINITIONS

Soit $D = \{(t_i, z_i)\} \quad i=1,2, \dots, n$ un ensemble de données, où $\Delta : a = t_1 < t_2 < \dots < t_n = b$ est une partition de l'intervalle $[a,b]$.

On pose

$$h_i = t_{i+1} - t_i$$

et $m_i = (z_{i+1} - z_i) / h_i \quad i = 1,2, \dots, n-1$

Les données seront dites **croissantes** si

$$(2.1) \quad m_i \geq 0 \quad i = 1,2, \dots, n-1$$

et **convexes** si

$$(2.2) \quad m_i - m_{i-1} \geq 0 \quad i = 2,3, \dots, n-1$$

Si les égalités ne sont jamais atteintes les données seront dites strictement croissantes et strictement convexes, respectivement. De manière analogue, on peut définir les données (strictement) décroissantes et (strictement) concaves.

Si les données sont croissantes ou décroissantes on dira qu'elles sont **monotones**

Nous allons étudier le problème d'interpolation suivant :

Problème 2.1

Soit la partition Δ et supposons que $z_i (i=1,2,\dots,n)$ sont des nombres réels donnés. On cherche une fonction spline polynomiale $s \in C^1[a,b]$ telle que

$$(2.3) \quad s(t_i) = z_i \quad i=1,2,\dots,n$$

en présence de conditions de forme. \square

On dira que l'interpolante s **préserve la monotonie** si :
 $s'(t)$ et m_i ont le même signe pour tout $t \in]t_i, t_{i+1}[$

et que s **préserve la convexité** si :
 s' est croissante (décroissante) lorsque les données sont convexes (concaves).

1. 3. INTERPOLATION MONOTONE ET CONVEXE PAR SPLINES QUADRATIQUES

Ce paragraphe concerne l'utilisation des splines quadratiques pour l'interpolation avec conditions de monotonie ou de convexité.

On remarque d'abord qu'il n'existe pas toujours une polynôme quadratique d'Hermite qui interpole deux points dont la position et la pente sont connues. Par contre, on peut toujours insérer un point pour avoir une interpolante continue formée par morceaux de polynômes de degré deux (spline quadratique).

Schumaker [103], à partir de l'étude des extrema locaux et des points d'inflexion des splines quadratiques, détermine les conditions nécessaires et suffisantes sur les dérivées aux noeuds et sur le choix des points supplémentaires afin d'avoir la monotonie ou la convexité de l'interpolante. Nous appelons cette méthode la méthode (S).

La méthode (McA/R) est due à Mc Allister et Roulier [66]. La manière de choisir les pentes (dérivées aux noeuds) et les points supplémentaires se base sur de considérations de caractère

géométrique. La spline est représentée dans la base de polynômes quadratiques de Bernstein.

3.1. La méthode (S)

Nous allons reprendre la démarche établie par Schumaker dans [103].

3.1.1. Etude locale de la monotonie et de la convexité

Considérons d'abord un problème d'interpolation d'Hermite pour deux points, qui va nous servir pour résoudre le problème 2.1.

Problème (PIH)

Soit $t_1 < t_2$. Supposons que $z_1, z_2, \lambda_1, \lambda_2$ sont des nombres réels donnés. Trouver une fonction $s \in C^1[t_1, t_2]$ telle que

$$(3.1) \quad s(t_i) = z_i \quad s'(t_i) = \lambda_i \quad i = 1, 2 \quad \square$$

La proposition suivante montre sous quelle condition un polynôme quadratique s est solution de (PIH).

Proposition 3.1.

Une condition nécessaire et suffisante pour qu'il existe un polynôme quadratique qui résolve le problème (PIH) est que

$$(3.2) \quad \lambda_1 + \lambda_2 = 2m$$

$$\text{où} \quad m = (z_2 - z_1) / h, \quad h = t_2 - t_1$$

Dans ce cas, la solution peut s'exprimer sous la forme

$$(3.3) \quad s(t) = z_1 + \lambda_1(t - t_1) + \frac{(\lambda_2 - \lambda_1)}{2h}(t - t_1)^2 \quad t \in I = [t_1, t_2] \quad \square$$

Si la condition (3.2) n'est pas vérifiée nous cherchons une solution qui soit une spline quadratique avec un noeud intermédiaire ξ entre t_1 et t_2 .

Proposition 3.2.

Pour tout $\xi \in]t_1, t_2[$ il existe une et une seule spline quadratique s avec un noeud en ξ qui résout (PIH).

Nous pouvons écrire

$$(3.4) \quad s(t) = \begin{cases} a_1 + b_1(t-t_1) + c_1(t-t_1)^2 & t_1 < t < \xi \\ a_2 + b_2(t-\xi) + c_2(t-\xi)^2 & \xi \leq t < t_2 \end{cases}$$

avec

$$(3.5) \quad a_1 = z_1, \quad b_1 = \lambda_1, \quad c_1 = (\bar{\lambda} - \lambda_1) / 2\alpha$$

$$a_2 = a_1 + b_1\alpha + c_1\alpha^2, \quad b_2 = \bar{\lambda}, \quad c_2 = (\lambda_2 - \bar{\lambda}) / 2\beta$$

où

$$(3.7) \quad \bar{\lambda} = s'(\xi) = 2m + (\alpha\lambda_1 + \beta\lambda_2) / h$$

$$\alpha = \xi - t_1, \quad \beta = t_2 - \xi$$

Les preuves de ces deux dernières propositions sont élémentaires. \square

En dérivant (3.3) on obtient

$$(3.7) \quad s'(t) = \lambda_1 + (\lambda_2 - \lambda_1)(t-t_1) / h \quad t \in I$$

$$s''(t) = (\lambda_2 - \lambda_1) / h$$

et de (3.4) on a

$$(3.8) \quad s'(t) = \begin{cases} \lambda_1 + (\bar{\lambda} - \lambda_1)(t-t_1) / \alpha & t_1 \leq t < \xi \\ \bar{\lambda} + (\lambda_2 - \bar{\lambda})(t-\xi) / \beta & \xi \leq t \leq t_2 \end{cases}$$

$$s''(t) = \begin{cases} (\bar{\lambda} - \lambda_1) / \alpha & t_1 \leq t \leq \xi \\ (\lambda_2 - \bar{\lambda}) / \beta & \xi \leq t \leq t_2 \end{cases}$$

Alors on peut déduire aisément quelques conséquences par rapport à la forme de s , car sa dérivée première est linéaire (par morceaux) et sa dérivée seconde est constante (par morceaux).

D'autre part, il est clair qu'une condition nécessaire pour la monotonie est :

$$(3.9) \quad \begin{cases} \operatorname{sgn}(\lambda_1) = \operatorname{sgn}(\lambda_2) = \operatorname{sgn}(m) & \text{si } m \neq 0 \\ \lambda_1 = \lambda_2 = 0 & \text{si } m = 0 \end{cases}$$

et qu'une condition nécessaire pour la convexité (concavité) est :

$$(3.10) \quad \lambda_1 < m < \lambda_2 \quad (\lambda_1 > m > \lambda_2)$$

Proposition 3.3.

Supposons que la condition $\lambda_1 + \lambda_2 = 2m$ soit vérifiée. Alors la fonction s définie par (3.3) a les propriétés suivantes :

- (i) s est monotone sur I si $\lambda_1 \cdot \lambda_2 \geq 0$
- (ii) s est convexe sur I si $\lambda_1 < \lambda_2$.

Dans le cas contraire, si $\lambda_1 + \lambda_2 \neq 2m$, la fonction s caractérisée par (3.4) possède les propriétés suivantes :

- (i) Supposons que $\lambda_1 \cdot \lambda_2 \geq 0$. Alors s est monotone sur I si et seulement si $\lambda_1 \cdot \bar{\lambda} \geq 0$.
- (ii) Supposons que $\lambda_1 < \lambda_2$. Alors s est convexe sur I si et seulement si $\lambda_1 < \bar{\lambda} < \lambda_2$.

Preuve

La preuve est immédiate d'après ce qui vient d'être observé plus haut. \square

Remarques

1) Pour la concavité il faut simplement changer le sens des inégalités dans (ii) et $\overline{\text{(ii)}}$.

2) La condition $\lambda_1 \bar{\lambda} \geq 0$ est équivalente à

$$(3.11) \quad 2|z_2 - z_1| \geq |\lambda_1(\xi - t_1) + \lambda_2(t_2 - \xi)|$$

On constate d'après les affirmations $\overline{\text{(i)}}$ et $\overline{\text{(ii)}}$ que si le noeud ξ est placé d'une manière arbitraire dans l'intervalle $]t_1, t_2[$ la spline ne sera pas nécessairement monotone ou convexe. Dans ce qui suit on va traiter plus en détail cette question pour la convexité.

Proposition 3.4.

(i) Si $(\lambda_1 - m)(\lambda_2 - m) \geq 0$ alors il existe un point d'inflexion dans l'intervalle I.

(ii) Si, par contre, $(\lambda_1 - m)(\lambda_2 - m) < 0$

a) la condition $|\lambda_2 - m| < |\lambda_1 - m|$ implique que pour tout ξ tel que

$$(3.12) \quad \xi \in]t_1, \bar{\xi}] \quad \text{où} \quad \bar{\xi} = t_1 + 2(\lambda_2 - m)(t_2 - t_1) / (\lambda_2 - \lambda_1)$$

la fonction s(3.4) soit

$$(3.13) \quad \begin{aligned} & - \text{convexe sur } I \text{ si } \lambda_1 < \lambda_2 \\ & - \text{concave sur } I \text{ si } \lambda_1 > \lambda_2 \end{aligned}$$

b) la condition $|\lambda_2 - m| > |\lambda_1 - m|$ implique que pour tout ξ tel que

$$(3.14) \quad \xi \in [\underline{\xi}, t_2[\quad \text{où} \quad \underline{\xi} = t_2 + 2(\lambda_1 - m)(t_2 - t_1) / (\lambda_2 - \lambda_1)$$

la fonction s (3.4) a les propriétés (3.13)

c) dans les deux cas précédents a) et b) si $\lambda_1, \lambda_2 \geq 0$ alors la spline s est aussi monotone.

Preuve

Elle est donnée dans [103].□

3.1.2. Un algorithme pour l'interpolation des données monotones ou convexes.

Les résultats du dernier paragraphe vont nous permettre de résoudre le problème (2.1).

Nous reprendrons les notations

$$I_i = [t_i, t_{i+1}], \quad m_i = (z_{i+1} - z_i) / (t_{i+1} - t_i) \quad i = 1, 2, \dots, n-1$$

si un noeud doit être inséré dans l'intervalle I_i nous l'appellerons ξ_i .

Deux questions essentielles sont à examiner : le calcul des dérivées aux noeuds et l'insertion des points supplémentaires.

• Calcul des dérivées $\{\lambda_i\} \quad i = 1, 2, \dots, n$

Si les données $D = \{(t_i, z_i)\} \quad i = 1, 2, \dots, n$ sont monotones, les dérivées aux noeuds $\{\lambda_i\} \quad i = 1, 2, \dots, n$ doivent vérifier la condition nécessaire de monotonie :

$$(3.15) \quad \left\{ \begin{array}{ll} \text{sgn}(\lambda_i) = \text{sgn}(\lambda_{i+1}) = \text{sgn}(m_i) & \text{si } m_i \neq 0 \\ \lambda_i = \lambda_{i+1} = 0 & \text{si } m_i = 0 \end{array} \right. \quad i = 1, 2, \dots, n-1$$

et si les données D sont convexes (concaves), les dérivées $\{\lambda_i\} \quad i = 1, \dots, n$ doivent vérifier la condition nécessaire de convexité (concavité) :

$$(3.16) \quad \lambda_i < m_i < \lambda_{i+1} \quad (\lambda_i > m_i > \lambda_{i+1}) \quad i = 1, 2, \dots, n-1$$

Dans le cadre de l'interpolation cubique par morceaux de classe C^1 plusieurs méthodes de type local (basées sur des considérations géométriques [1] ou autres [17],[35]) ont été proposées pour le calcul des dérivées aux noeuds. Nous procédons de la manière suivante :

- On trouve dans un noeud intérieur t_i ($i=2,3,\dots,n-1$) la valeur de la dérivée du polynôme quadratique qui interpole (t_{i-1}, z_{i-1}) , (t_i, z_i) et (t_{i+1}, z_{i+1}) . La formule correspondante est :

$$p_i = (h_{i-1}m_i + h_i m_{i-1}) / (h_{i-1} + h_i) \quad i = 2, 3, \dots, n-1$$

- Aux bouts on prend :

$$p_1 = m_1 + (m_1 - m_2)h_1 / (h_1 + h_2) \quad ,$$

$$p_n = m_{n-1} + (m_{n-1} - m_{n-2})h_{n-1} / (h_{n-2} + h_{n-1})$$

- On définit les schémas :

$$(D0) : \left\{ \begin{array}{l} \lambda_i = p_i \quad i = 1, 2, \dots, n \end{array} \right.$$

$$(D1) \left\{ \begin{array}{l} \lambda_1 = \begin{cases} 0 & \text{si } m_1 = 0 \text{ ou } \text{sgn}(p_1) \neq \text{sgn}(m_1) \\ p_1 & \text{sinon} \end{cases} \\ \lambda_i = \begin{cases} 0 & \text{si } m_{i-1} \text{ ou } m_i = 0 \\ p_i & \text{sinon} \end{cases} \quad i=2, 3, \dots, n-1 \\ \lambda_n = \begin{cases} 0 & \text{si } m_{n-1} = 0 \text{ ou } \text{sgn}(p_{n-1}) \neq \text{sgn}(m_{n-1}) \\ p_n & \text{sinon} \end{cases} \end{array} \right.$$

Remarques

1) Si les données sont convexes alors les $\{\lambda_i\}$ $i = 1, 2, \dots, n-1$ calculés par (D0) vérifient les conditions (3.16). Si, en plus, elles sont monotones on a aussi les relations (3.15).

2) Le schéma (D1) fournit des valeurs de $\{\lambda_i\}$ pour lesquelles la condition nécessaire de monotonie (3.15) est assurée lorsque les données sont monotones.

• **Choix des noeuds supplémentaires**

Supposons que les données sont monotones et que les dérivées aux noeuds $\{\lambda_i\}$ vérifient (3.15). Si $\lambda_i + \lambda_{i+1} \neq 2m_i$, le noeud ξ_i doit être choisi de telle sorte que :

$$(3.17) \quad 2|z_{i+1} - z_i| \geq |\lambda_i(\xi_i - t_i) + \lambda_{i+1}(t_{i+1} - \xi_i)|$$

pour garantir la monotonie locale de s sur I_i .

Supposons maintenant que les données sont convexes (concaves) et les dérivées aux noeuds vérifient (3.16). Si $\lambda_i + \lambda_{i+1} \neq 2m_i$ le noeud ξ_i doit être choisi de telle sorte que :

$$(3.18) \quad t_i < \xi_i \leq t_i + \frac{2(\lambda_{i+1} - m_i)(t_{i+1} - t_i)}{\lambda_{i+1} - \lambda_i} \quad \text{si } |\lambda_{i+1} - m_i| < |\lambda_i - m_i|$$

$$(3.19) \quad t_{i+1} + \frac{2(\lambda_i - m_i)(t_{i+1} - t_i)}{\lambda_{i+1} - \lambda_i} \leq \xi_i < t_{i+1} \quad \text{si } |\lambda_{i+1} - m_i| > |\lambda_i - m_i|$$

pour garantir la convexité (concavité) locale de s sur I_i .

Dans les exemples numériques nous avons pris les noeuds ξ_i au milieu des intervalles définis par (3.17), (3.18) ou (3.19), selon le cas.

Nous allons décrire un algorithme pour résoudre le problème 2.1.

Algorithme 3.1

1. (en entrée)

n : nombre de données

$\{t_i\}$, $\{z_i\}$ $i = 1, 2, \dots, n-1$ données

2. Calculer $m_i = (z_{i+1} - z_i) / (t_{i+1} - t_i)$ $i = 1, 2, \dots, n-1$

3. Calculer les dérivées $\{\lambda_i\}$ $i = 1, 2, \dots, n$ avec le schéma (D1) si les données sont monotones. Sinon utiliser (D0).

4. Pour $i = 1$ jusqu'à $n-1$

(i) Si $\lambda_i + \lambda_{i+1} = 2m_i$

calculer les coefficients du polynôme quadratique (3.3).

(ii) Sinon, $\lambda_i + \lambda_{i+1} \neq 2m_i$,

a) Si $(\lambda_i - m_i)(\lambda_{i+1} - m_i) \geq 0$

a₁) Si $\text{sgn}(\lambda_i) = \text{sgn}(\lambda_{i+1}) = \text{sgn}(m_i)$

prendre ξ_i au milieu de l'intervalle (3.17)

a₂) Sinon,

prendre ξ_i au milieu de l'intervalle I_i .

b) Dans le cas contraire, $(\lambda_i - m_i)(\lambda_{i+1} - m_i) < 0$,

b₁) Si $|\lambda_{i+1} - m_i| < |\lambda_i - m_i|$

prendre ξ_i au milieu de l'intervalle (3.18)

b₂) Sinon, $|\lambda_{i+1} - m_i| > |\lambda_i - m_i|$,

prendre ξ_i au milieu de l'intervalle (3.19).

c) Calculer les coefficients des morceaux des polynômes quadratiques (3.4) - (3.6).

5. (en sortie)

K : nombre de noeuds

$\{x_i\}$ $i = 1, 2, \dots, K$ noeuds

$\{a_i, b_i, c_i\}$ $i = 1, 2, \dots, K-1$ coefficients des morceaux des polynômes quadratiques. □

Commentaires

L'algorithme 3.1. produit les coefficients d'une fonction spline quadratique s telle que :

- $s(t) = a_j + b_j(t - x_j) + c_j(t - x_j)^2 \quad x_j \leq t \leq x_{j+1}, \quad j=1,2,\dots,K-1$
- $s(t_i) = z_i \quad i=1,2,\dots,n$
- $s \in C^1 [a,b]$
- s est monotone sur $[a,b]$ si les données sont globalement monotones et si les dérivées $\{\lambda_i\}$ sont calculées par (D1).
- s est convexe (concave) sur $[a,b]$ si les données sont globalement convexes (concaves) et si les dérivées $\{\lambda_i\}$ sont calculées par (D0).
- s préserve, en même temps, la monotonie et la convexité globales des données si les dérivées $\{\lambda_i\}$ sont calculées par (D0).

3.2. La méthode (Mc A/R)

Nous présentons dans ce paragraphe une version simplifiée de la méthode de McAllister et Roulier [66] pour la préservation de la forme par splines quadratiques.

3.2.1. Un problème d'Hermite pour deux points

Considérons à nouveau le problème (PIH).

Soient $t_1 < t_2$. Supposons $z_1, z_2, \lambda_1, \lambda_2$ sont des nombres réels donnés. Nous rappelons qu'il existe un polynôme quadratique p tel que

$$(3.20) \quad p(t_i) = z_i, \quad p'(t_i) = \lambda_i \quad i=1,2$$

si et seulement si

$$(3.21) \quad \lambda_1 + \lambda_2 = 2m \quad \text{où } m = (z_2 - z_1)/(t_2 - t_1).$$

Dans ce cas, la solution s'écrit :

$$(3.22) \quad p(t) = z_1 + \lambda_1(t - t_1) + (\lambda_2 - \lambda_1) (t - t_1)^2 / (2h) \quad t \in [t_1, t_2]$$

Remarque

Ce résultat exprime le fait que les tangentes à p en t_1 , et t_2 se coupent au milieu de l'intervalle $I = [t_1, t_2]$. \square

D'autre part, les conditions nécessaires de :

-monotonie : $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$ et de

- convexité : $\lambda_1 < m < \lambda_2$

sont aussi suffisantes pour que le polynôme p soit monotone et convexe sur I .

En général, la condition (3.21) n'est pas vérifiée, alors on va résoudre (PIH) en utilisant une spline quadratique.

Soit L_1 le polynôme de degré 1 de pente λ_1 qui interpole $P = (t_1, z_1)$ et L_2 le polynôme de degré 1 de pente λ_2 qui interpole $Q = (t_2, z_2)$. Supposons que les graphes de L_1 et L_2 se coupent en un point dont nous notons l'abscisse t_c . Nous avons le résultat suivant :

Proposition 3.5

L'abscisse t_c du point d'intersection des graphes de L_1 et L_2 se trouve à l'intérieur de l'intervalle I si et seulement si

$$(3.23) \quad \lambda_1 < m < \lambda_2 \quad \text{ou} \quad \lambda_1 > m > \lambda_2$$

Preuve

On a $t_c = t_1 + (m - \lambda_2)(t_2 - t_1) / (\lambda_1 - \lambda_2)$

Alors $t_c \in]t_1, t_2[$ si et seulement si $0 < \frac{m - \lambda_2}{\lambda_1 - \lambda_2} < 1$.

Cette dernière condition est équivalente à (3.23). \square

Voyons comment construire une spline quadratique s avec un noeud intermédiaire ξ entre t_1 et t_2 , de manière qu'elle satisfasse les

conditions d'interpolation d'Hermite (3.20). Pour cela considérons deux cas possibles

Cas 1. $t_c \in]t_1, t_2[$.

Prenons $\xi = t_c$.

Considérons les points $A = (t_a, z_a)$ et $B = (t_b, z_b)$

où $t_a = (t_1 + \xi)/2$, $t_b = (\xi + t_2)/2$, $z_a = L_1(t_a)$, $z_b = L_2(t_b)$.

Soit \bar{L} le polynôme de degré 1 qui interpole A et B et posons $C = (\xi, \bar{L}(\xi))$. Notons $\bar{\lambda}$ la pente de \bar{L} (voir Fig.1).

Cas 2. $t_c \in]t_1, t_2[$ ou les graphes de L_1 et L_2 ne se coupent pas. Prenons $\xi = (t_1 + t_2)/2$.

Pour déterminer A, B, C et \bar{L} nous suivons le même schéma que dans le cas 1. (voir Fig.2).

Il est clair que, par construction, il existe deux polynômes quadratiques p_1 et p_2 tels que :

$$(3.24) \quad \left\{ \begin{array}{l} p_1(t_1) = z_1, \quad p_1(\xi) = \bar{L}(\xi) \\ p_1'(t_1) = \lambda_1, \quad p_1'(\xi) = \bar{\lambda} \end{array} \right.$$

et

$$(3.25) \quad \left\{ \begin{array}{l} p_2(\xi) = \bar{L}(\xi), \quad p_2(t_2) = z_2 \\ p_2'(\xi) = \bar{\lambda}, \quad p_2'(t_2) = \lambda_2 \end{array} \right.$$

On peut résumer ces résultats sous la forme de la proposition suivante :

Proposition 3.6.

Supposons $\lambda_1 + \lambda_2 \neq 2m$

Pour $\xi = t_c$ (cas1) ou pour $\xi = (t_1 + t_2)/2$ (cas2) il existe une et une seule spline quadratique s des noeuds t_1, ξ, t_2 qui résolve (PIH).

Cette spline est définie par

$$(3.26) \quad s(t) = \begin{cases} p_1(t), & t_1 \leq t < \xi \\ p_2(t), & \xi \leq t \leq t_2 \end{cases}$$

où p_1 et p_2 sont caractérisés par (3.24) et (3.25). Les coefficients de p_1 et p_2 sont donnés par les formules (3.5) - (3.6). □

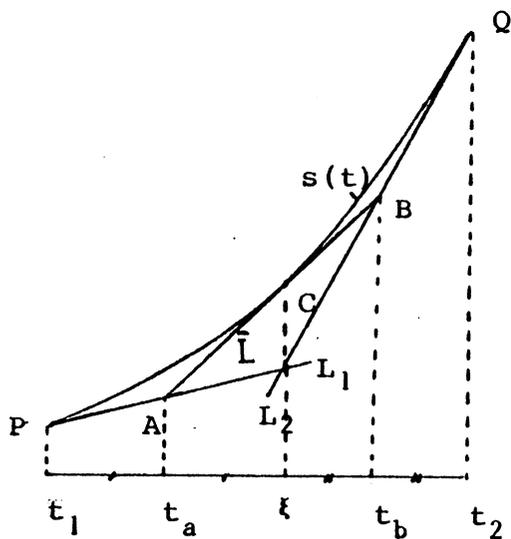


Fig. 1. Cas 1.

$$\lambda_1 > 0, \lambda_2 > 0, \lambda_1 < m < \lambda_2$$

Remarque

Mc Allister et Roulier [66] présentent d'une autre manière les différents cas possibles liés au comportement des graphes de L_1 et L_2 . Mais pour nos applications, l'étude ci-dessus sera suffisante.

Examinons maintenant la forme de cette spline. Commençons par le cas 1.

Proposition 3.7

Supposons $\lambda_1 + \lambda_2 \neq 2m$ et $\lambda_1 < m < \lambda_2$.

Alors la spline quadratique s définie par (3.26) est convexe (concave). Si, en plus, $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$, alors s est aussi monotone.

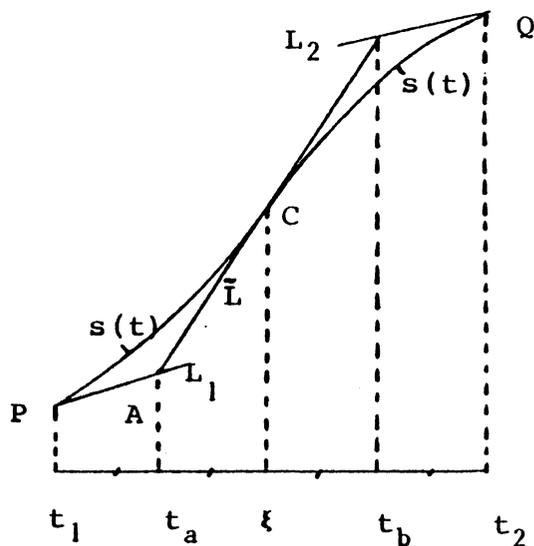


Fig. 2. Cas 2.

$$\lambda_1 > 0, \lambda_2 > 0, \lambda_1 + \lambda_2 < 4m$$

Preuve

On a $\xi = t_c \in]t_1, t_2[$. On peut montrer aisément que la pente $\bar{\lambda}$ qui correspond à \bar{L} est égale à m . La conclusion est immédiate d'après la Proposition 3.3. \square

Examinons le *cas 2*, c'est-à-dire $(\lambda_1 - m)(\lambda_2 - m) \geq 0$. Il est clair que la spline s possède un point d'inflexion à l'intérieur de l'intervalle I . Intéressons-nous à l'étude de la monotonie de la fonction.

Proposition 3.8

Supposons que $\lambda_1 + \lambda_2 \neq 2m$, $(\lambda_1 - m)(\lambda_2 - m) \geq 0$ et $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$.

Alors la spline quadratique s définie par (3.26) est monotone si et seulement si $|\lambda_1 + \lambda_2| \leq 4m$

Preuve

Après quelques calculs élémentaires on a $\bar{\lambda} = 2m - (\lambda_1 + \lambda_2) / m$. Comme on doit avoir $\bar{\lambda} \cdot \lambda_1 \geq 0$, le résultat est immédiat. \square

3.2.2. Un algorithme pour l'interpolation des données monotones ou convexes

Revenons au problème 2.1. Pour sa résolution on va se servir des résultats du paragraphe précédent.

Reprenons les notations :

$$I_i = [t_i, t_{i+1}] , m_i = (z_{i+1} - z_i) / (t_{i+1} - t_i) \quad i=1, 2, \dots, n-1$$

et ξ_i si un noeud doit être inséré dans l'intervalle I_i .

Voyons la manière de choisir les pentes aux noeuds. La procédure utilisée dans [66] conduit aux formules suivantes :

$$(D2) \left\{ \begin{array}{l} \lambda_i = \begin{cases} 2m_{i-1} m_i / (m_{i-1} + m_i) & \text{si } m_{i-1} m_i > 0 \\ 0 & \text{sinon} \end{cases} & i=2,3,\dots,n-1 \\ \lambda_1 = \begin{cases} 2m_1 - \lambda_2 & \text{si } m_1 m_2 > 0 \\ 0 & \text{sinon} \end{cases} \\ \lambda_n = \begin{cases} 2m_{n-1} - \lambda_{n-1} & \text{si } m_{n-2} m_{n-1} > 0 \\ 0 & \text{sinon} \end{cases} \end{array} \right.$$

On note au passage que λ_i est la moyenne harmonique de m_{i-1} et m_i si $m_{i-1} \cdot m_i > 0$.

On va établir les propriétés des pentes $\{\lambda_i\}$ $i = 1, 2, \dots, n$

Proposition 3.9

Soient D un ensemble de données et $\{\lambda_i\}$ $i = 1, 2, \dots, n$ les dérivées calculées par le schéma (D2).

- (i) Si $(m_i - m_{i-1}) \geq 0$ $(m_i - m_{i-1}) \leq 0$ $i = 2, 3, \dots, n-1$
 alors $\lambda_i \leq m_i \leq \lambda_{i+1}$ $(\lambda_i \geq m_i \geq \lambda_{i+1})$ $i = 1, 2, \dots, n-1$
- (ii) Si $m_i \geq 0$ $(m_i \leq 0)$ $i = 1, 2, \dots, n-1$
 alors $\lambda_i \geq 0$ $(\lambda_i \leq 0)$ $i = 1, 2, \dots, n$
 et $\lambda_i + \lambda_{i+1} \leq 4|m_i|$ $i = 1, 2, \dots, n-1$. \square

Autrement dit, si les données sont globalement convexes (concaves) ou monotones alors les pentes $\{\lambda_i\}$ $i = 1, 2, \dots, n$ vérifient la condition nécessaire de convexité (concavité) ou de monotonie.

Maintenant, nous sommes en mesure de définir un algorithme pour résoudre le problème 2.1.

Algorithme 3.2

1. (en entrée)

n : nombre de données

$\{t_i\}, \{z_i\} \quad i = 1, 2, \dots, n$ données

2. Calculer $m_i = (z_{i+1} - z_i) / (t_{i+1} - t_i) \quad i = 1, 2, \dots, n-1$

3. Calculer les dérivées $\{\lambda_i\}$ avec la procédure (D2).

4. Pour $i = 1$ jusqu'à $n-1$

(i) Si $\lambda_i + \lambda_{i+1} = 2m_i$

calculer les coefficients du polynôme quadratique (3.3).

(ii) Sinon, $\lambda_i + \lambda_{i+1} \neq 2m_i$,

a) Si $(\lambda_i - m_i)(\lambda_{i+1} - m_i) \geq 0$

prendre $\xi_i = (t_i + t_{i+1}) / 2$

b) Sinon, $(\lambda_i - m_i)(\lambda_{i+1} - m_i) < 0$,

prendre $\xi_i = t_i + (m_i - \lambda_{i+1})(t_{i+1} - t_i) / (\lambda_i - \lambda_{i+1})$

c) Calculer les coefficients des morceaux des polynômes quadratiques (3.4)-(3.6).

5. (en sortie)

K: nombre de noeuds

$\{x_i\} \quad i = 1, 2, \dots, K$ noeuds

$\{a_i, b_i, c_i\} \quad i = 1, 2, \dots, K-1$ coefficients des morceaux de polynômes quadratiques. \square

Commentaires

Cet algorithme produit les coefficients d'une fonction spline quadratique s telle que :

• $s(t) = a_j + b_j (t - x_j) + c_j (t - x_j)^2 \quad x_j \leq t \leq x_{j+1}, j = 1, 2, \dots, K-1$

• $s(t_i) = z_i \quad i = 1, 2, \dots, n$

• $s \in C^1 [a, b]$

- s est monotone sur $[a,b]$ si les données sont globalement monotones.
- s est convexe (concave) sur $[a,b]$ si les données sont globalement convexes (concaves).

3.4. INTERPOLATION CUBIQUE PAR MORCEAUX DES DONNEES MONOTONES ET CONVEXES.

Maintenant nous passons en revue les algorithmes pour construire des interpolantes monotones et/ou convexes de classe C^1 formées par morceaux des polynômes de degré 3.

Fritsch et Carlson [41] se sont concentrés sur le cas de la monotonie, tandis que Mettke [73] s'est intéressé à la convexité. L'interpolation monotone et convexe est discutée dans [21], [22] par Costantini et Morandi. D'autres développements se trouvent dans la littérature chinoise ([53], [116], [117]).

Soit $\Delta : a = t_1 < t_2 < \dots < t_n = b$ une partition de $[a,b]$ et un ensemble de valeurs $\{z_i\} \quad i = 1, \dots, n$, on va construire une fonction $s \in S(3,1)$ telle que

$$(4.1) \quad s(t_i) = z_i \quad i = 1, 2, \dots, n$$

Cette spline cubique avec noeuds (doubles) $t_i, \quad i=1, \dots, n$ est déterminée de manière unique par les valeurs $\{z_i\}, \{\lambda_i\} \quad i=1, \dots, n$, où

$$(4.2) \quad s'(t_i) = \lambda_i \quad i=1, \dots, n$$

Une représentation possible pour $s(t)$ est la suivante :

$$s(t) = z_i + \lambda_i(t-t_i) + \frac{m_i - \lambda_i}{h_i} (t-t_i)^2 + \frac{\lambda_{i+1} - 2m_i + \lambda_i}{h_i^2} (t-t_i)^2 (t-t_{i+1})$$

$$\text{pour } t \in [t_i, t_{i+1}] \quad (4.3)$$

où $m_i = (z_{i+1} - z_i) / h_i, \quad h_i = t_{i+1} - t_i \quad i=1, 2, \dots, n-1$

Comme les z_j sont connus, le problème se réduit à calculer les λ_j .

Pour garantir que $s(t)$ respecte la forme des données on a cherché des conditions nécessaires et suffisantes sur les dérivées aux noeuds et à partir de là on a déduit des algorithmes pour les calculer.

4.1. Préservation de la monotonie

On va étudier la monotonie locale sur l'intervalle $I_j = [t_j, t_{j+1}]$

Proposition 4.1

Soit p_j le polynôme cubique tel que

$$p_j(t_j) = z_j, p'_j(t_j) = \lambda_j \quad j = i, i+1$$

Alors p_j est monotone sur I_j si et seulement si

(i) $\lambda_j = \lambda_{j+1} = 0$ si $m_j = 0$

(ii) $(\alpha_j, \beta_j) \in \mathcal{M}$ si $m_j \neq 0$

où $\alpha_j = \lambda_j/m_j$, $\beta_j = \lambda_{j+1}/m_j$ et \mathcal{M} est la région montrée dans la Fig3. et caractérisée par

$$\mathcal{M} = \{(\alpha, \beta) / \alpha \geq 0, \beta \geq 0, (\alpha + \beta \leq 3 \text{ ou } \alpha^2 + \beta^2 + \alpha\beta - 6(\alpha + \beta) - 9) \leq 0\}$$

Preuve Voir [41]

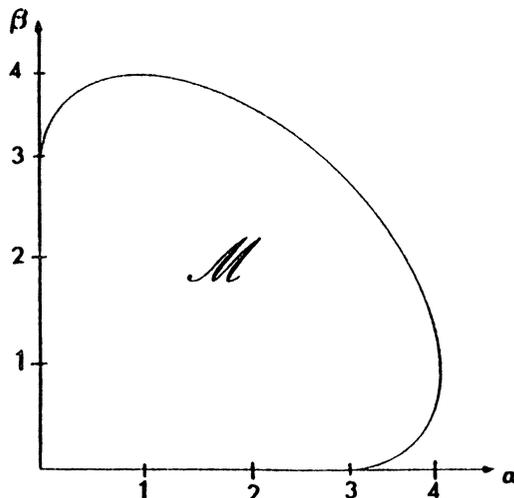


Fig 3. La région "monotone" \mathcal{M} .

Cette proposition exprime le fait que la fonction est monotone si le rapport entre les dérivées aux noeuds et la pente de la corde qui relie les deux points d'interpolation n'est pas "trop grande". Il est clair que $\lambda_i=0$ ($i=1,2,\dots,n$) est une solution triviale mais qui ne donne pas forcément de bons résultats graphiques et numériques.

Pour avoir la monotonie globale, dans [41] on propose de calculer les dérivées λ_i avec un algorithme à deux pas :

1. Donner ou calculer $\{\lambda_i\}$ $i=1,2,\dots,n$, telles qu'elles aient le même signe.
2. Pour $i=1,2,\dots,n-1$:
 si $(\alpha_i, \beta_i) \notin \mathcal{M}$ modifier α_i et β_i pour trouver $\bar{\alpha}_i$ et $\bar{\beta}_i$ tels que $(\bar{\alpha}_i, \bar{\beta}_i) \in \mathcal{M}$. Alors $\bar{\lambda}_i = \bar{\alpha}_i m_i$ et $\bar{\lambda}_{i+1} = \bar{\beta}_i m_i$. \square

Pour deux intervalles adjacents on a la relation $\beta_{i-1} m_{i-1} = \lambda_i = \alpha_i m_i$. Si l'on modifie α_i on change λ_i et par conséquent β_{i-1} , alors il faut faire attention pour continuer à préserver la monotonie sur l'intervalle I_{i-1} .

Pour cela on va choisir une région $\tilde{\mathcal{M}} \subset \mathcal{M}$ telle que :

- (i) Si $(\alpha, \beta) \in \tilde{\mathcal{M}}$ alors $(\bar{\alpha}, \bar{\beta}) \in \tilde{\mathcal{M}}$ pour $0 \leq \bar{\alpha} \leq \alpha$ et $0 \leq \bar{\beta} \leq \beta$
- (ii) Si $(\alpha, \beta) \in \tilde{\mathcal{M}}$ alors $(\beta, \alpha) \in \tilde{\mathcal{M}}$.

Il existe plusieurs régions qui possèdent telles propriétés. Nous ne retiendrons que deux :

$$\tilde{\mathcal{M}}_1 = \{(\alpha, \beta) / \alpha > 0, \beta > 0, \alpha^2 + \beta^2 \leq 9\}$$

et

$$\tilde{\mathcal{M}}_2 = [0, 3] \times [0, 3]$$

Alors l'algorithme peut s'écrire :

Algorithme 4.1.

1. Calculer $\{\lambda_i\}_{i=1,2,\dots,n}$ telles que
 $\text{sgn}(\lambda_i) = \text{sgn}(\lambda_{i+1}) = \text{sgn}(m_i) \quad i=1,2,\dots,n-1.$
2. Si $(\alpha, \beta) \notin \tilde{\mathcal{M}}$ alors trouver $(\bar{\alpha}_i, \bar{\beta}_i) = L \cap \text{Front}(\tilde{\mathcal{M}})$, où L est la droite qui relie (0,0) à (α_i, β_i) . Prendre $\alpha_i = \bar{\alpha}_i$ et $\beta_i = \bar{\beta}_i$. \square

Pour le pas 1. on peut utiliser les schémas (D1) ou (D2) du § 1.2.

D'autre part, Fritsch et Butland [409] ont donné des formules explicites pour les dérivées λ_i afin que les (α_i, β_i) restent dans $\tilde{\mathcal{M}}_2$. Plus précisément :

$$(4.4) \quad \lambda_i = G(m_{i-1}, m_i, h_{i-1}, h_i) \quad i=1,2,\dots,n-1$$

où

$$(4.5) \quad G(v_1, v_2, u_1, u_2) = \begin{cases} \frac{v_1 \cdot v_2}{\alpha v_2 + (1-\alpha) v_1} & \text{si } v_1 \cdot v_2 > 0 \\ 0 & \text{sinon} \end{cases}$$

$$\text{avec } \alpha = \left[1 + \frac{u_2}{u_1 + u_2} \right] / 3$$

Il faut, bien entendu, ajouter deux conditions de bout, pour avoir les valeurs λ_1 et λ_n .

4.2. Préservation de la convexité

Nous commençons par donner un résultat sur la convexité locale.

Proposition 4.2.

Soit p_i le polynôme cubique tel que

$$p_i(t_j) = z_j, \quad p'_i(t_j) = \lambda_j \quad j=i, i+1$$

Alors $p_i(t)$ est convexe sur I_i si et seulement si

$$(4.6) \quad 2\lambda_i + \lambda_{i+1} \leq 3m_i \leq \lambda_i + 2\lambda_{i+1}$$

Preuve

La dérivée seconde p''_i est un morceau de droit dans l'intervalle I_i .

La condition $p''_i(t) \geq 0, t \in I_i$, est équivalente à $p''_i(t_i^+) \cdot p''_i(t_{i+1}^-) \geq 0$

Alors on déduit (4.6). \square

Si $m_i > 0$, la Fig. 4 représente la région "convexe" \mathcal{E}_i définie par

$\mathcal{E}_i = \{(\alpha, \beta) / 2\alpha + \beta \leq 3m_i \leq \alpha + 2\beta\}$. Alors (4.6) est équivalente à :

$(\lambda_i, \lambda_{i+1}) \in \mathcal{E}_i$.

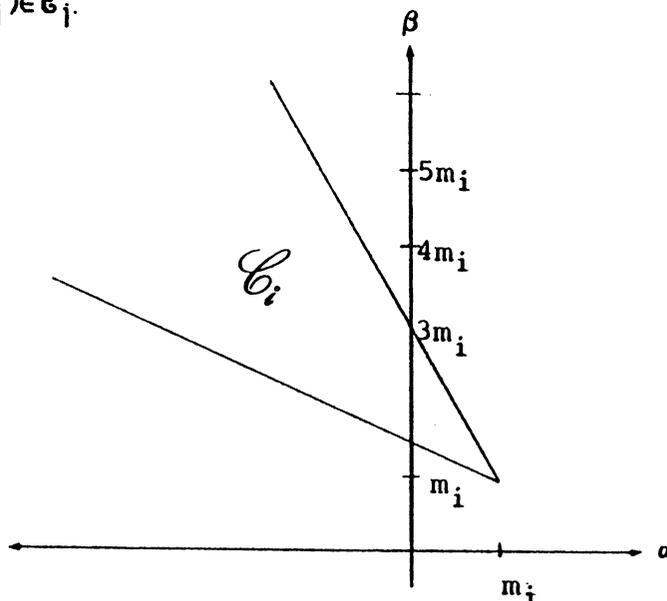


Fig. 4. La région "convexe" \mathcal{E}_i ($m_i > 0$)

Pour avoir la convexité globale de la spline cubique $s(4.3)$ il faut que le système des inégalités:

$$(4.7) \quad \begin{cases} 2\lambda_i + \lambda_{i+1} \leq 3m_i \\ -\lambda_i - 2\lambda_{i+1} \leq -3m_i \end{cases} \quad i = 1, 2, \dots, n-1$$

soit solvable.

On peut montrer qu'il y a des ensembles de données convexes pour lesquels (4.7) n'a pas de solution (voir [86]).

Pour la résolution de (4.7) Mettke [73] propose la méthode suivante

Algorithme 4.2.

1. Poser $a_1^{(0)} = 3m_1 - 2m_2$; $b_1^{(0)} = (3m_1 - 3m_2 + 2m_3)/2$

Résoudre :

$$a_k^{(0)} - a_{k-1}^{(0)} = (m_{2k-1} - m_{2k-2}) - 2(m_{2k} - m_{2k-1}) \quad k=2, 3, \dots, [(n-1)/2]$$

$$b_k^{(0)} - b_{k-1}^{(0)} = (m_{2k+1} - m_{2k}) - (m_{2k} - m_{2k-1})/2 \quad k=2, 3, \dots, [(n-2)/2]$$

2. $\bar{\lambda}_1 = \min_k \{m_1, \min_k \{b_k^{(0)}\}\}$, $\underline{\lambda}_1 = \max_k \{a_k^{(0)}\}$

Si $\underline{\lambda}_1 > \bar{\lambda}_1$ stopper l'algorithme, sinon choisir $\lambda_1 \in [\underline{\lambda}_1, \bar{\lambda}_1]$ et continuer.

3. Pour $i = 2, \dots, n-2$.

$$a_k^{(i)} = 3m_{i-1} - 2b_k^{(i-1)} \quad k=1, \dots, [(n-i)/2]$$

$$b_k^{(i)} = (3m_{i-1} - a_{k+1}^{(i-1)})/2 \quad k=1, \dots, [(n-i-1)/2]$$

$$\underline{\lambda}_i = \max \{(3m_{i-1} - \lambda_{i-1})/2, \max_k \{a_k^{(i)}\}\}$$

$$\bar{\lambda}_i = \min \{3m_{i-1} - 2\lambda_{i-1}, m_i, \min_k \{b_k^{(i)}\}\}$$

Si $\underline{\lambda}_i > \bar{\lambda}_i$ stopper l'algorithme, sinon choisir $\lambda_i \in [\underline{\lambda}_i, \bar{\lambda}_i]$ et continuer.

4. Poser $\underline{\lambda}_{n-1} = (3m_{n-2} - \lambda_{n-2}) / 2$

$$\bar{\lambda}_{n-1} = \min \{3m_{n-2} - 2\lambda_{n-2}, m_{n-1}\}$$

Si $\underline{\lambda}_{n-1} > \bar{\lambda}_{n-1}$ stopper l'algorithme, sinon choisir $\lambda_{n-1} \in [\underline{\lambda}_{n-1}, \bar{\lambda}_{n-1}]$ et continuer.

5. Poser $\underline{\lambda}_n = (3m_{n-1} - \lambda_{n-1}) / 2$, $\bar{\lambda}_n = 3m_{n-1} - 2\lambda_{n-1}$

Si $\underline{\lambda}_n \leq \bar{\lambda}_n$ choisir $\lambda_n \in [\underline{\lambda}_n, \bar{\lambda}_n]$

6. Stop. \square

Il se pose la question de la faisabilité de cet algorithme. Le résultat suivant est démontré dans [73].

Proposition 4.3

Supposons que l'ensemble des données $\{(t_i, z_i)\}_{i=1,2,\dots,n}$ soit convexe. Alors les conditions suivantes sont équivalentes :

- (i) le système (4.7) est solvable
- (ii) l'algorithme 4.2 est faisable.
- (iii) les conditions
 - a) $a_k^{(i)} \leq m_{j+1}$, $k=1,\dots,[(n-i-1)/2]$
 - b) $a_k^{(i)} \leq b_j^{(i)}$, $k=1,\dots,[(n-i-1)/2]$, $j=1,\dots,[(n-i-2)/2]$sont valables pour $i=0,1,\dots,n-2$.

Remarques

1) S'il n'existe pas de solution pour (4.7), il y a un indice r pour lequel $\underline{\lambda}_r > \bar{\lambda}_r$ et l'algorithme est arrêté. La spline peut être calculée dans $[t_1, t_{r-1}]$ avec les paramètres $\{\lambda_j\}_{j=1,\dots,r-1}$, mais pour $[t_{r-1}, t_n]$ la fonction ne sera plus convexe.

2) L'algorithme 4.2 est faisable sous des hypothèses plus fortes que la convexité des données. Plus précisément, si l'on a la condition

$$(4.8) \quad 2(m_{i+1} - m_i) > (m_i - m_{i-1}) \quad i = 2, \dots, n-2$$

Alors on peut considérer une version simplifiée de cette méthode.

4.3. Préservation de la convexité et de la monotonie

Pour fixer les idées supposons que l'ensemble des données $D = \{t_i, z_i\} \quad i=1, \dots, n$ soit croissante et convexe. Les autres cas peuvent se ramener à cette forme, via une transformation.

D'après les propositions 4.1 et 4.2 on découle le résultat suivant :

Proposition 4.4

Soit p_i le polynôme cubique tel que

$$p_i(t_j) = z_j, \quad p'_i(t_j) = \lambda_j \quad j=i, i+1.$$

Alors $p_i(t)$ est croissant et convexe sur $[t_i, t_{i+1}]$ si et seulement si

$$(4.9) \quad (\lambda_i, \lambda_{i+1}) \in \mathfrak{R}_i$$

$$\text{où } \mathfrak{R}_i = \{(\alpha, \beta) / \alpha \geq 0, \beta \geq 0, \frac{1}{2}(3m_i - \alpha) \leq \beta \leq 3m_i - 2\alpha\}$$

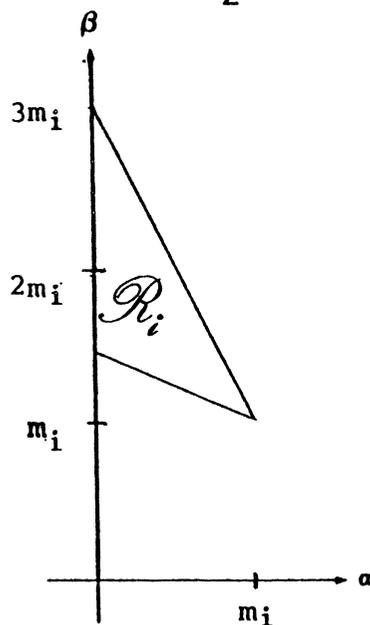


Fig.5. Region "croissante" et "convexe" \mathfrak{R}_i

Remarque

Remplaçons dans l'algorithme 4.2 la borne $\underline{\lambda}_1$ par

$$(4.10) \quad \underline{\lambda}_1 = \max \{0, \max \{a_k^{(0)}\}.$$

Si l'algorithme 4.2 modifié est faisable, c'est-à-dire si les pentes $\{\lambda_i\}_{i=1,2,\dots,n}$ résolvent le système (4.7), alors la spline cubique s (4.3) en plus de convexe est monotone sur $[a,b]$.

Les résultats suivants sont donnés dans [22].

Algorithme 4.3

1. $a_0 = +\infty$, $b_0 = 0$, $m_0 = 0$, $m_n = +\infty$, $k=1$.
2. Pour $i = 1, 2, \dots, n$
 - (i) $a_i = 3m_{i-1} - 2b_{i-1}$
 - (ii) $b_i = \max \{m_i, (3m_{i-1} - a_{i-1}) / 2\}$
 - (iii) Si $b_i > a_i$ alors $k=i+1$ et arrêter l'algorithme, sinon continuer. \square

Proposition 4.5

Supposons que $D = \{(t_i, z_i)\}_{i=1,2,\dots,n}$ soit un ensemble de données convexes et croissantes.

(i) Il existe une spline cubique interpolante croissante et convexe de classe C^1 si et seulement si l'algorithme 4.3 termine avec $k=1$.

(ii) Supposons que l'algorithme 4.3 donne $k=1$.

Alors pour tout $\lambda_i \in [a_i, b_i]$ il existe $\lambda_{i+1} \in [a_{i+1}, b_{i+1}]$ tel que

$(\lambda_i, \lambda_{i+1}) \in \mathfrak{R}_i$, $i = 1, 2, \dots, n-1$. \square

Les dérivées $\{\lambda_i\}$ seront calculées avec le schéma suivant :

Algorithme 4.4

1. Calculer $\{a_i\}$ et $\{b_i\}$ $i=1, \dots, n$ avec l'algorithme 4.3. On suppose que $k=1$.
2. Donner $\theta \in [0, 1]$
3. $\lambda_1 = \theta a_1 + (1-\theta)b_2$
4. Pour $i=1, 2, \dots, n$
 - (i) $a = \min \{ \gamma \in \mathbb{R} : (\lambda_{i-1}, \gamma) \in \mathcal{R}_{i-1} \}$
 - (ii) $b = \max \{ \gamma \in \mathbb{R} : (\lambda_{i-1}, \gamma) \in \mathcal{R}_{i-1} \}$
 - (iii) $\lambda_i = \theta a + (1-\theta)b$ \square

Commentaires

- 1) Cet algorithme est de type global.
- 2) Si D est un ensemble de données quelconque on peut le diviser en sous-ensembles de données qui aient les mêmes caractéristiques de monotonie et convexité et appliquer séparément la méthode décrite ci-dessus. Il rest alors à calculer les dérivées dans les points de jonction.

1.5. COURBES PARAMETREES PLANES QUI CONSERVENT LA FORME

Dans ce paragraphe nous allons considérer les courbes interpolantes sous forme paramétrée.

On notera P la courbe plane définie par

$$(5.1) \quad P = \{[x(u), y(u)], u \in I\}$$

où x et y sont des fonctions de la variable réelle u (le paramètre) et I est un intervalle de \mathbb{R} .

Si l'on considère l'ensemble des données $D = \{(x_i, y_i)\} \quad i=1, \dots, n$ les techniques décrites avant peuvent être utilisées si on résout séparément les problèmes pour $\{(u_i, x_i)\}$ et $\{(u_i, y_i)\} \quad i=1, \dots, n$ pour obtenir les splines $s_x(u)$ et $s_y(u)$ et, finalement, la courbe

$$s = \{(s_x(u), s_y(u)), u \in I\}$$

Lorsque la paramétrisation n'est pas connue, une des solutions consiste à approcher la longueur d'arc par

$$(5.2) \quad \begin{cases} u_1 = 0 \\ u_i = u_{i-1} + \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \quad i=2, \dots, n \end{cases}$$

Il faut signaler que le choix de la paramétrisation ainsi que le calcul des pentes aux points d'interpolation sont très importants pour éviter la formation de boucles (voir [36], [70], [71]).

Dans le cas où les données D correspondent aux valeurs d'une fonction $y_i = f(x_i) \quad i=1, 2, \dots, n$, Butland [13] propose l'utilisation de polynômes cubiques par morceaux paramétrés pour préserver la monotonie. Dimsdale [30], pour éliminer des points d'inflexion artificiels lorsque les pentes des données sont très grandes utilise aussi de cubiques paramétrées mais en ajoutant des points supplémentaires. Harada et Nakamae [48] appliquent les techniques Bernstein-Bézier pour conserver la convexité.

5.1. Interpolation monotone par cubiques paramétrées

La méthode que nous allons décrire constitue une extension des idées du paragraphe 3.1.

Considérons un ensemble de données monotones $D = \{(x_i, y_i)\} \quad i=1, 2, \dots, n$, où $a=x_1 < x_2 < \dots < x_n=b$. Notons :

v_i soit x_i , soit y_i

\dot{v}_i soit $\dot{x}(u_i)$, soit $\dot{y}(u_i) \quad (i=1, 2, \dots, n)$

$dv_i = v_{i+1} - v_i \quad (i=1, 2, \dots, n-1)$

Butland [13] prend la paramétrisation uniforme $u_j=i$ ($i=1,2,\dots,n$) et la dérivée première aux noeuds avec la formule (4.4), c'est-à-dire

$$(5.3) \quad \dot{v}_i = G(dv_{i-1}, dv_i) \quad i=2,\dots,n-1$$

avec

$$(5.4) \quad G(dv_{i-1}, dv_i) = \begin{cases} \frac{2 dv_{i-1} dv_i}{dv_{i-1} + dv_i} & \text{si } dv_{i-1} \cdot dv_i > 0 \\ 0 & \text{sinon} \end{cases}$$

La fonction $G(dv_{i-1}, dv_i)$ possède les propriétés suivantes :

(i) symétrie par rapport à dv_{i-1} et dv_i : $G(dv_{i-1}, dv_i) = G(dv_i, dv_{i-1})$

(ii) $|G(dv_{i-1}, dv_i)| \in [0, \min(2|dv_{i-1}|, 2|dv_i|)]$

Remarque

Si $dv_{i-1} \cdot dv_i > 0$ alors \dot{v}_i et \dot{v}_{i+1} appartiennent à la région "monotone" $\tilde{\mathcal{M}} = [0, 3dv_i] \times [0, 3dv_i]$. D'après la proposition 3.1 le polynôme cubique p_i tel que

$$p_i(j) = v_j, \quad p'_i(j) = \dot{v}_j \quad j=i, i+1$$

est monotone sur $[i, i+1]$. \square

On définit la courbe

$$(5.5) \quad s = \{ [x(u), y(u)], u \in [1, n] \}$$

où

$$(5.6) \quad \begin{cases} x(u) = p_i(u) \\ y(u) = q_i(u) \end{cases} \quad u \in [i, i+1] \quad , \quad i=1, 2, \dots, n-1$$

et p_i et q_i sont des polynômes cubiques tels que :

$$(5.7) \quad p_i(j) = x_j, \quad p'_i(j) = \dot{x}_j \quad j=i, i+1$$

$$(5.8) \quad q_i(j) = y_j, \quad q'_i(j) = \dot{y}_j \quad j=i, i+1$$

avec \dot{x}_j et \dot{y}_j ($j=1,2,\dots,n$) déterminés par (5.3) - (5.4).

On a le résultat suivant :

Proposition 5.1

La courbe s caractérisée par (5.5) - (5.8) est monotone sur $[a,b]$ si les données D sont monotones.

Preuve

La remarque ci-dessus entraîne que la fonction $x(u)$ est monotone croissante sur $[1,n]$, et que $y(u)$ est monotone sur ce même intervalle. Supposons que les données sont croissantes alors :

$$\frac{d_y}{d_x} = \frac{\dot{y}_u}{\dot{x}_u} \geq 0$$

car $\dot{x}(u) > 0$ et $\dot{y}(u) \geq 0$ pour $u \in [1,n]$.

On raisonne de la même manière si les données sont décroissantes. \square

5.2. Application des courbes de Bézier pour préserver la convexité.

Les courbes de Bézier sont très utilisées en dessin graphique assisté par ordinateur. Une courbe de Bézier est un arc paramétré polynomial qui peut être représenté dans la base de Bernstein : ses coefficients sont des points qui forment une ligne polygonale (polygone de Bézier), dont l'enveloppe convexe contient la courbe associée. (voir [12],[39],[45]).

Une courbe de Bézier $P(u)$ de degré m s'exprime sous la forme :

$$(5.9) \quad P(u) = \sum_{j=0}^m P_j B_{j,m}(u)$$

où les polynômes de Bernstein de degré m sont définis par

$$(5.10) \quad B_{j,m}(u) = \binom{m}{j} u^j (1-u)^{m-j} \quad j=0,1,\dots,m$$

u étant le paramètre (en général on prend $0 < u < 1$). Les points P_j $j=1,\dots,m$ sont les points de Bézier.

Les caractéristiques de ces courbes ont permis à Harada et Nakamae [48] de formuler une algorithmme de type local et qui conserve la convexité des données interpolées.

Le problème à étudier est le suivant :

Problème 5.1

Soit $D = \{(x_i, y_i)\}_{i=1,\dots,n}$ un ensemble de données convexes. On veut construire une courbe de classe C^1 formée par courbes de Bézier qui préserve la convexité. \square

D'abord nous allons construire la courbe entre $P_i = (x_i, y_i)$ et $P_{i+1} = (x_{i+1}, y_{i+1})$. Soit L_i le polynôme de degré 1 de pente λ_i qui interpole P_i . Supposons que $\lambda_i < m_i < \lambda_{i+1}$, donc les graphes de L_i et L_{i+1} se coupent en un point dont l'abscisse se trouve à l'intérieur de $[x_i, x_{i+1}]$.

Introduisons un paramètre $\alpha \in]0,1[$ et définissons pour $m=2k$ ($k=2,3,\dots$) les points de contrôle:

$$(5.11) \quad \left\{ \begin{array}{l} Q_0 = (x_i, y_i) \\ Q_1 = Q_2 = \dots = Q_{k-1} = Q_a \\ Q_k = (x_i, y_i) \\ Q_{k+1} = Q_{k+2} = \dots = Q_{2k-1} = Q_b \\ Q_{2k} = (x_{i+1}, y_{i+1}) \end{array} \right.$$

où Q_a et Q_b sont tels que

$$(5.12) \quad \alpha(Q_a - Q_0) = Q_k - Q_0, \quad \alpha(Q_b - Q_{2k}) = Q_k - Q_{2k}$$

Traçons alors la courbe de Bézier associée aux points Q_j ($j=1,2,\dots,2k$). Voir Fig. 6.

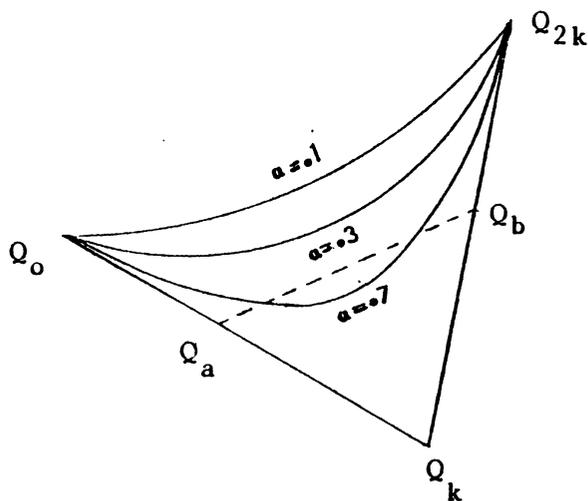


Fig. 6. Courbes de Bézier associées aux points Q_j ($j=1,2,\dots,6$).

Pour éviter de calculer des polynômes de degré élevé on va prendre $k=3$, c'est-à-dire $m=6$. Dans ce cas l'équation (5.9) se réduit à

$$(5.13) \quad P(u) = \sum_{j=0}^6 R_j u^j$$

où

$$(5.14) \quad \left\{ \begin{array}{l} R_0 = Q_0 \\ R_1 = 6\alpha(Q_3 - Q_0) \\ R_2 = 15\alpha(Q_0 - Q_3) \\ R_3 = 20\alpha(Q_3 - Q_0) \\ R_4 = 15[(3-2\alpha)Q_0 + (3\alpha-4)Q_3 + (1-\alpha)Q_6] \\ R_5 = 6[(5\alpha-6)Q_0 + (10-9\alpha)Q_3 + (4\alpha-4)Q_6] \\ R_6 = (10-9\alpha)Q_0 + (18\alpha-20)Q_3 + (10-9\alpha)Q_6 \end{array} \right.$$

Proposition 5.2 (voir [48])

Supposons $\lambda_i < m_i < \lambda_{i+1}$ où $m_i = (y_{i+1} - y_i) / (x_{i+1} - x_i)$.

La courbe (5.13)-(5.14) est convexe pour $0 < u < 1$ et $0 < \alpha < 1$. \square

Remarques

1) le paramètre α contrôle la "courbure" de P, mais il n'existe pas une formule analytique pour le déterminer. La courbe est assez sensible à la variation de α . Une valeur trop près de 1 conduit à l'approcher du polygone de Bézier $Q_0, Q_2, Q_k, Q_b, Q_{2k}$.

2) La courbe interpolante solution du problème 5.1. se forme en enchaînant les courbes de Bézier $P(u)$ définies par (5.13) - (5.14), et elle est de classe C^1 .

3) Pour calculer $\{\lambda_i\}_{i=1, \dots, n}$ on peut utiliser les procédures (D0) ou (D2) du §1.3.

1.6. ESSAIS NUMERIQUES

Nous avons résumé dans le tableau 1 les caractéristiques principales des différentes méthodes décrites dans les paragraphes précédents.

Méthode	Degré de la spline	Continuité	Globale/ Locale	Préservation de la Monotonie/Convexité
(McA/R): McAllister - Roulier	2	C^1	L	M et/ou C
(S): Schumaker	2	C^1	L	M et/ou C
(F/C): Fritsch-Carlson	3	C^1	L	M
(F/B): Fritsch-Butland	3	C^1	L	M
(B): Butland	3	C^1	L	M
(M): Mettke	3	C^1	\emptyset	C
(C/M): Costantini - Morandi	3	C^1	\emptyset	M et C
(H/N): Harada-Nakamae	6	C^1	L	C

Tableau 1. Méthodes fondées sur des splines polynomiales pour préserver la forme.

Ces algorithmes ont été testés sur plusieurs exemples. La comparaison entre eux est basée sur deux critères principaux : l'aspect "visuel" et le temps d'exécution.

• Préservation de la monotonie

Reprenons les données monotones considérées par Pruess dans [93].

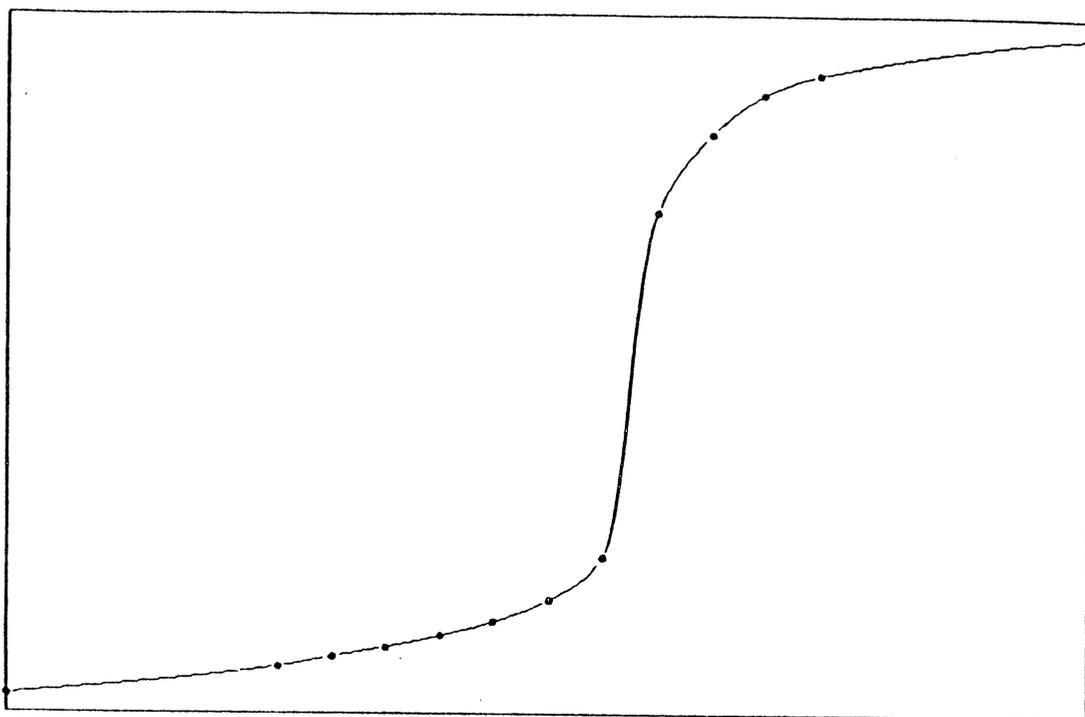
t_j	22	22.5	22.6	22.7	22.8	22.9	23	23.1	23.2	23.3	23.4	23.5	24
z_j	523	543	550	557	565	575	590	620	860	915	944	958	986

Tableau 2. Données monotones 1.

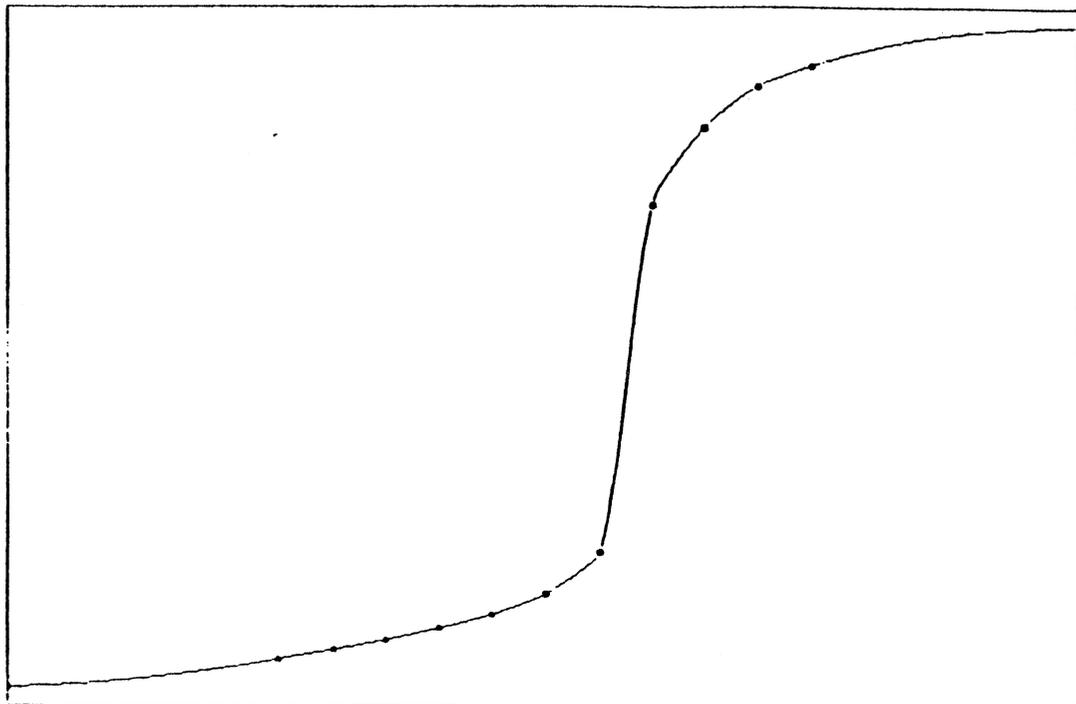
Un bref aperçu des données montre que la variation la plus importante se produit entre les points (23.1,620) et (23.3,860), alors il faut s'attendre à ce que les interpolantes reflètent une telle situation.

La Fig.7. représente les courbes monotones des splines quadratiques. Les dérivées aux noeuds ont été calculées par la procédure (D2) pour la méthode (McA/R) et par le schéma (D1) pour (S). Dans le deuxième cas, l'insertion des points est faite au milieu des intervalles (3.17), (3.18) ou (3.19),selon le cas.

Les graphes des splines cubiques, dont les dérivées sont estimées par les algorithmes de (F/C) et (F/B), apparaissent, respectivement, dans la Fig. 8(a) et 8(b). Le deuxième profil est plus "plaisant" que le premier. Un résultat semblable est donné par la Fig.9(a), lorsque la fonction est considérée sous forme paramétrée et on applique la méthode (B).



(a)

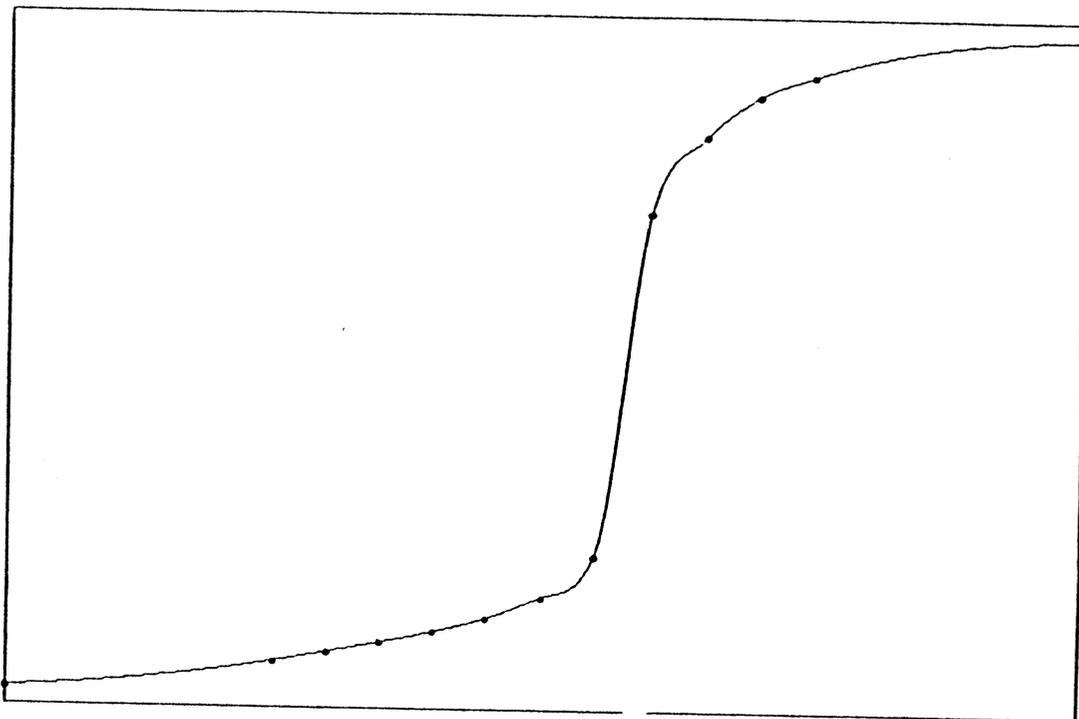


(b)

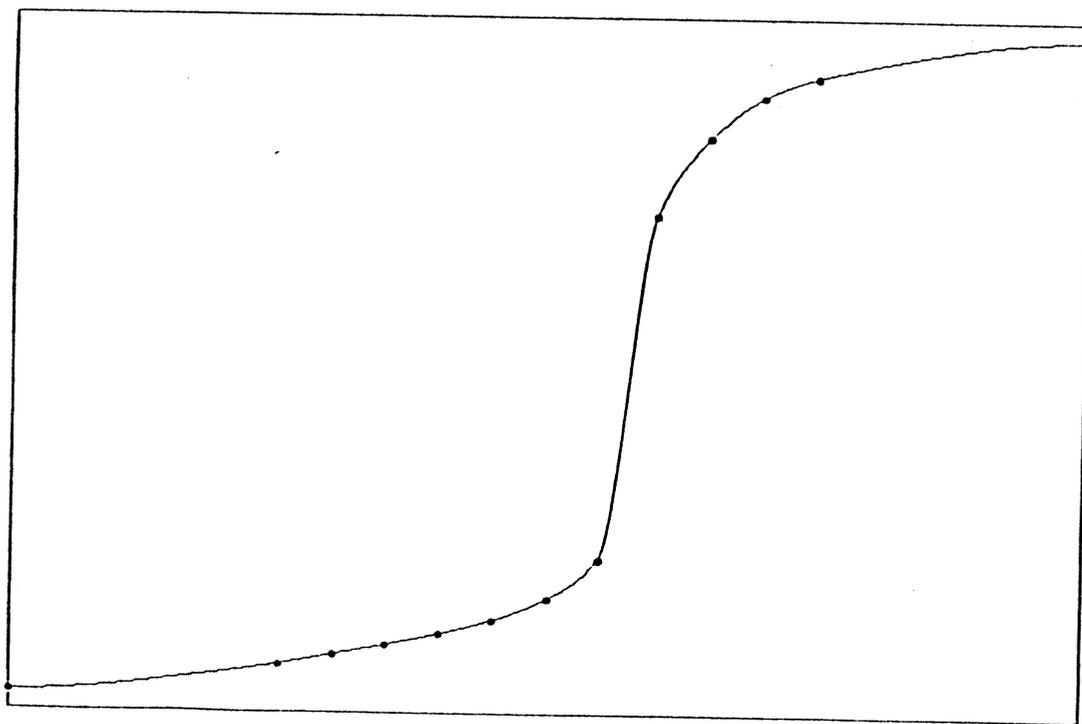
Fig.7. Splines quadratiques pour les données monotones 1.

(a) méthode (McA/R)

(b) méthode (S)

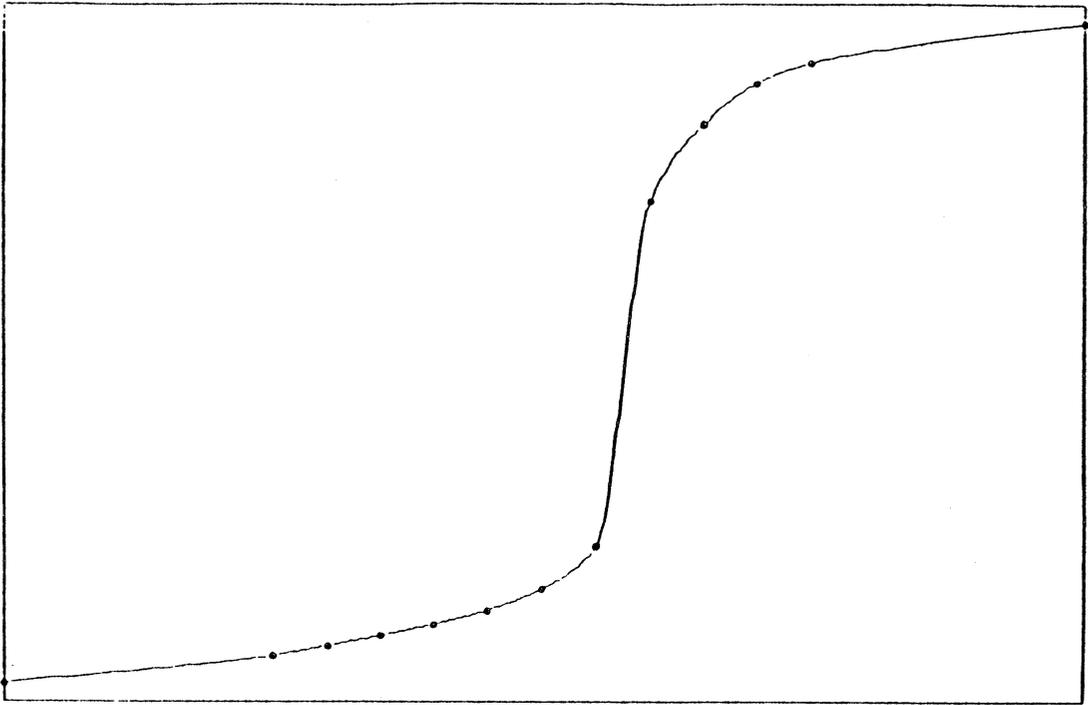


(a)

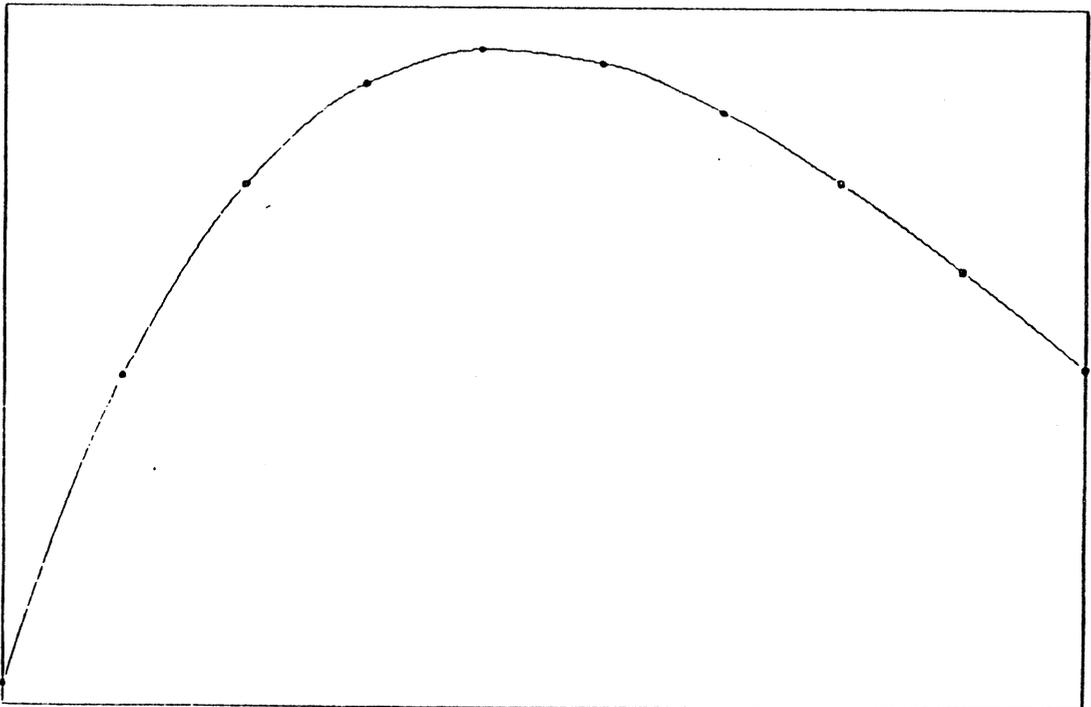


(b)

Fig. 8. Splines cubiques C^1 pour les données monotones 1.
(a) méthode (F/C)
(b) méthode (F/B)



(a)



(b)

Fig.9. (a) Spline cubique C^1 pour les données monotones 1. Méthode (B)
(b) Spline quadratique pour les données concaves 2. Méthode (McA/R)

• Préservation de la convexité

Considérons l'ensemble des données rangées dans le tableau 3.

t_j	1	5/3	7/3	3	11/3	13/3
z_j	8.56681	11.20710	12.83481	13.69863	13.99474	13.87708
t_j	5	17/3	19/3	7		
z_j	13.46573	12.85362	12.11209	11.29530		

Tableau 3. Données concaves 2

Les ordonnées correspondent aux valeurs $z_j = f_1(t_j)$ où

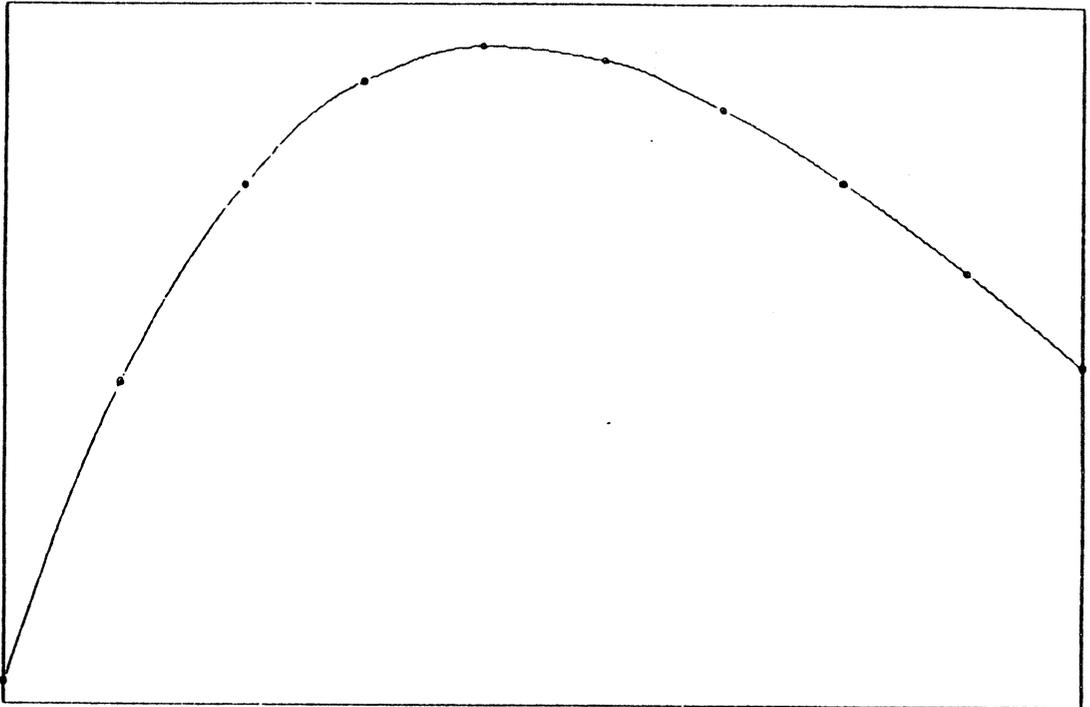
$$f_1(t) = (-9t+2)^{-0.25t}$$

Cette fonction est concave sur l'intervalle $]-\infty, 70/9[$.

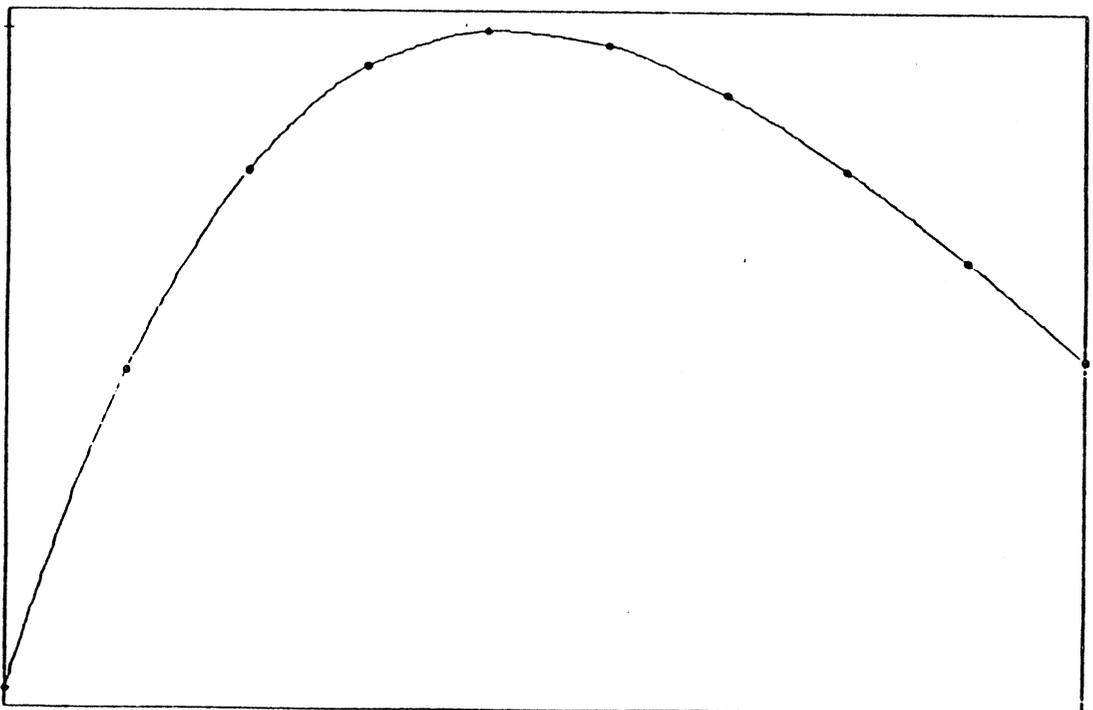
La courbe 9(b) montre bien que les splines quadratiques (McA/R) conservent la concavité des données. On peut dire la même chose pour la méthode (S), en regardant la Fig.10(a).

L'utilisation des courbes de Bézier par l'algorithme (H/N) est illustrée par la Fig.10(b). Il faut signaler que la courbe (tout en restant concave) est sensible à la variation du paramètre α . Ici on a obtenu une représentation plaisante visuellement lorsque $\alpha=0.3$.

Pour mieux apprécier les différences entre ces méthodes on a tracé aussi les dérivées premières de ces fonctions. (Voir Fig.11). On remarque immédiatement le caractère C^0 et la décroissance des dérivées, autrement dit, le caractère C^1 des splines et leur concavité.

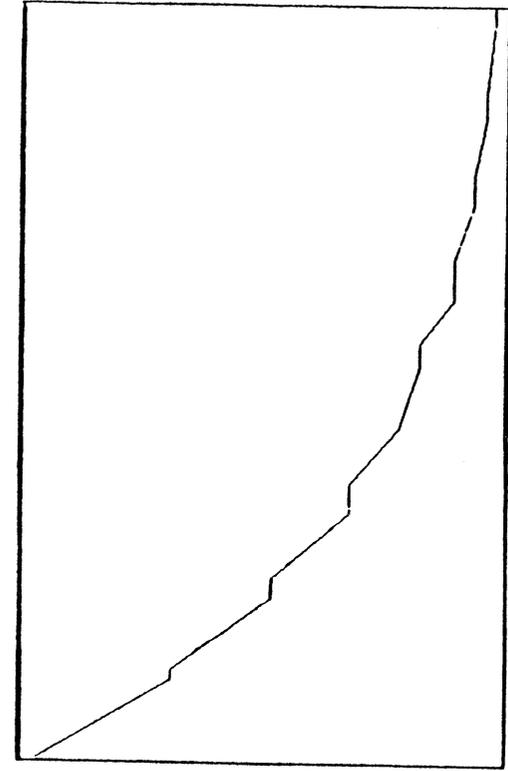


(a)

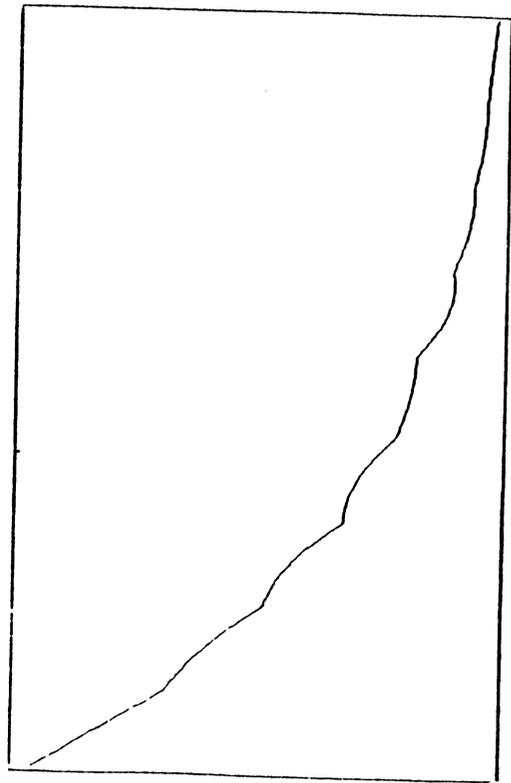


(b)

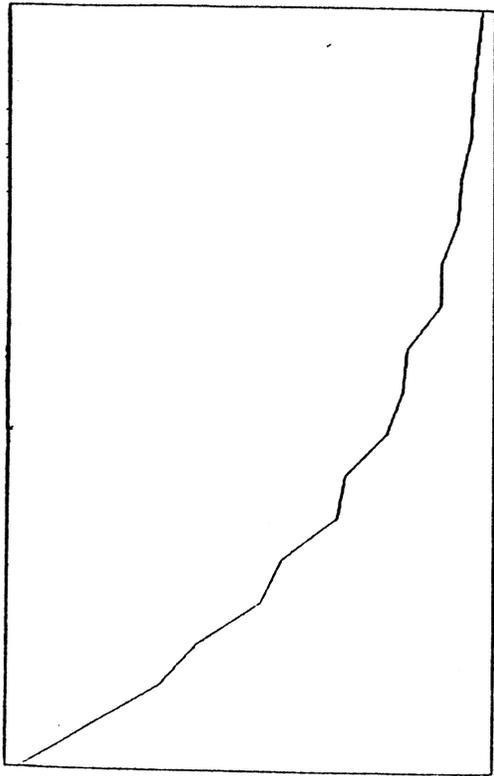
Fig.10. Interpolation des données concaves 2.
(a) Spline quadratique, méthode (S)
(b) Courbe de Bézier, méthode (H/N)



(a)



(b)



(c)

Fig.11. Dérivées premières des splines concaves
(a) Méthode (McA/R)
(b) Méthode (S)
(c) Méthode (H/N)

• Préservation de la monotonie et de la convexité

Pour pouvoir comparer toutes les méthodes il faut traiter des exemples où on considère, en même temps, les conditions de monotonie et de convexité. Prenons 10 abscisses équidistantes dans l'intervalle $[-1,8]$ et comme ordonnées les valeurs respectives de la fonction

$$f_2(t) = (9t+2)/(4t+5).$$

t_j	- 1	0	1	2	3	4	5
z_j	11.0	0.4	-0.77777	-1.23077	-1.47059	-1.6190	-1.72
t_j	6	7	8				
z_j	-1.79310	-1.84848	-1.89189				

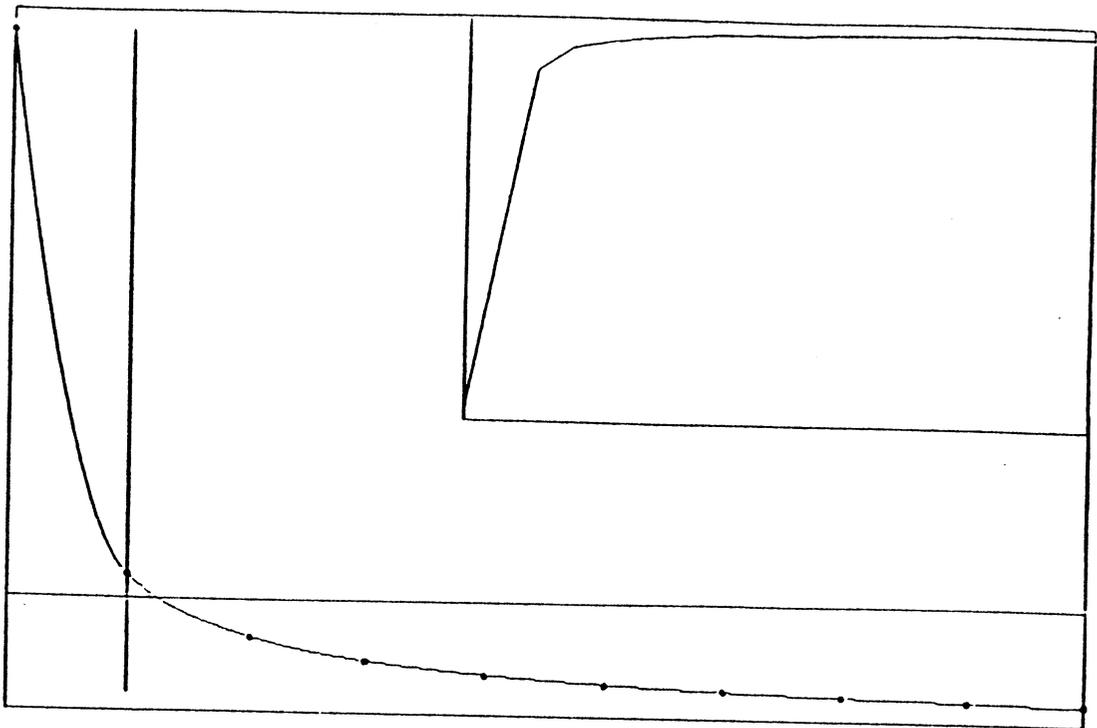
Tableau 4. Données décroissantes et convexes 3.

Nous cherchons une interpolante monotone en appliquant les algorithmes (F/C),(F/B) et (B). La convexité seule est assurée par la méthode (H/N). La conservation simultanée de la monotonie et la convexité est assurée par (McA/R) , (S) et (C/M).

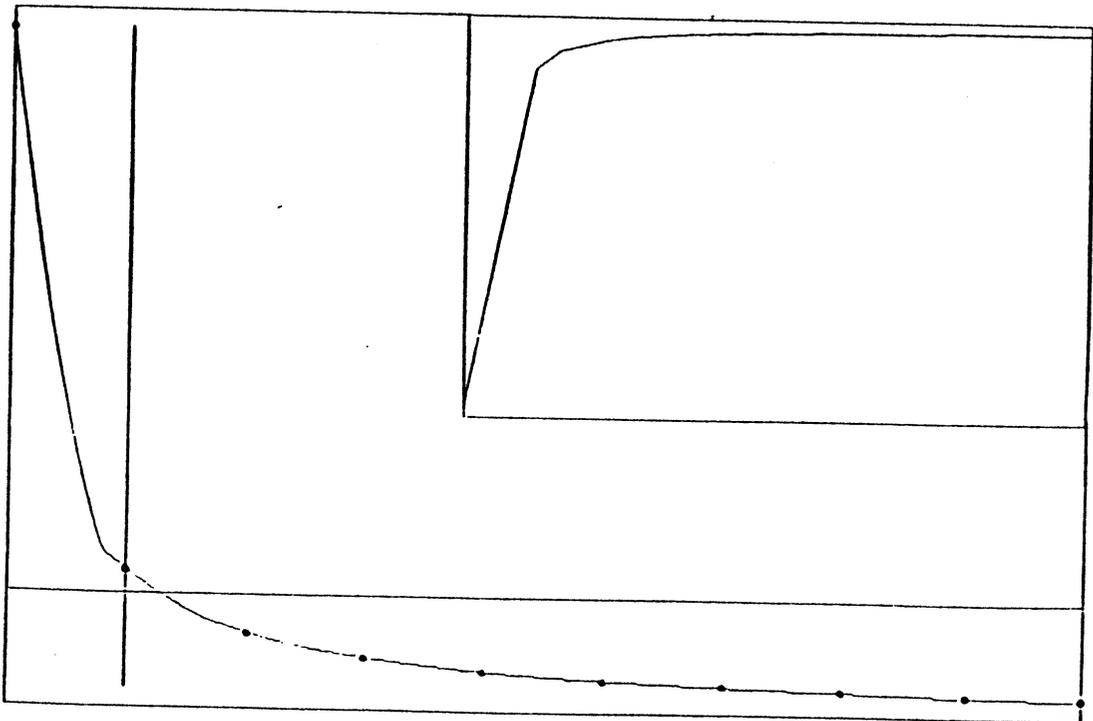
Les Fig. 12,13 et 14 contiennent les courbes obtenues par ces différentes techniques. Dans la fenêtre de l'angle supérieur droit on a placé le graphe correspondant à leur dérivée première, qui doit être croissante si la convexité est préservée.

Dans cet exemple, les changements les plus remarquables se produisent autour du point $(t_2, z_2) = (0,0.4)$, car c'est la région où la décroissance de la fonction est la plus rapide.

Une fois de plus on constate que pour les splines quadratiques la méthode (McA/R) donne un meilleur résultat que (S). (Fig.12). Mais nous avons remarqué que ces méthodes donnent des résultats similaires si toutes les deux utilisent le schéma (D2) pour calculer les dérivées aux noeuds.



(a)

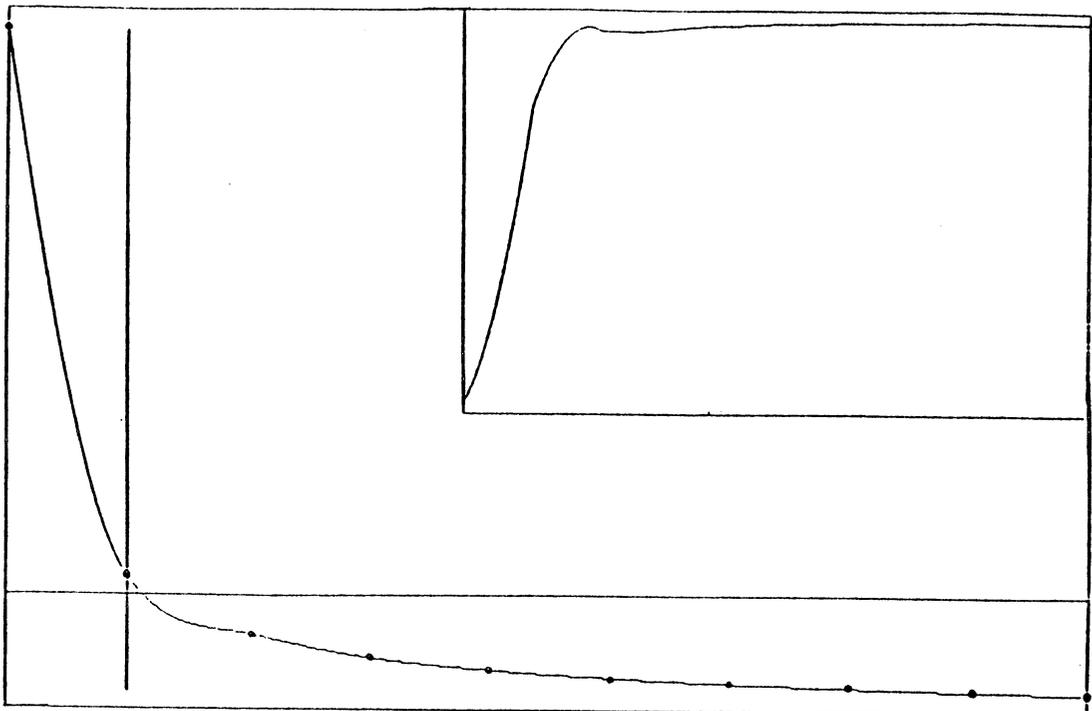


(b)

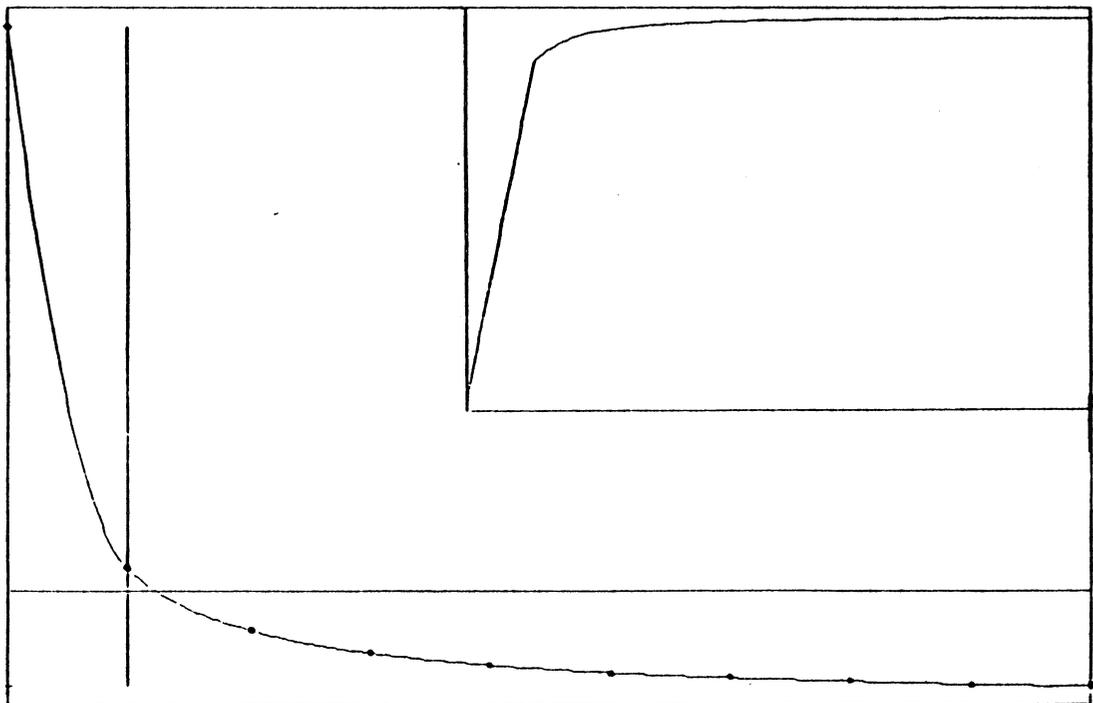
Fig.12. Splines quadratiques pour les données monotones et convexes 3.

(a) Méthode (McA/R)

(b) Méthode (S)

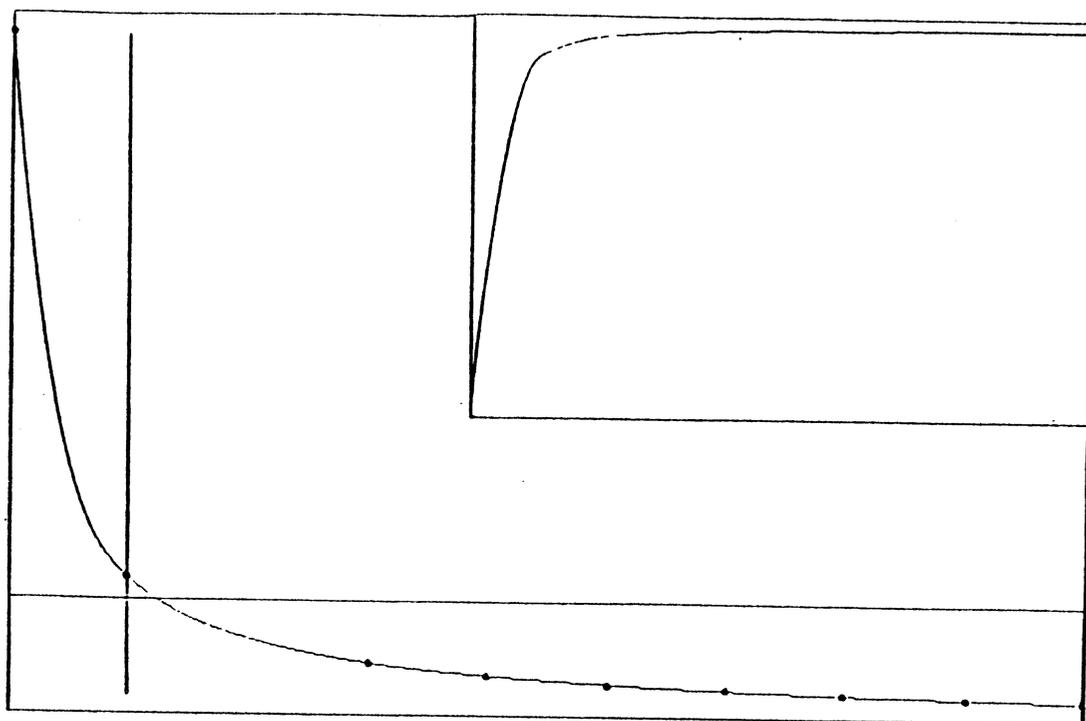


(a)

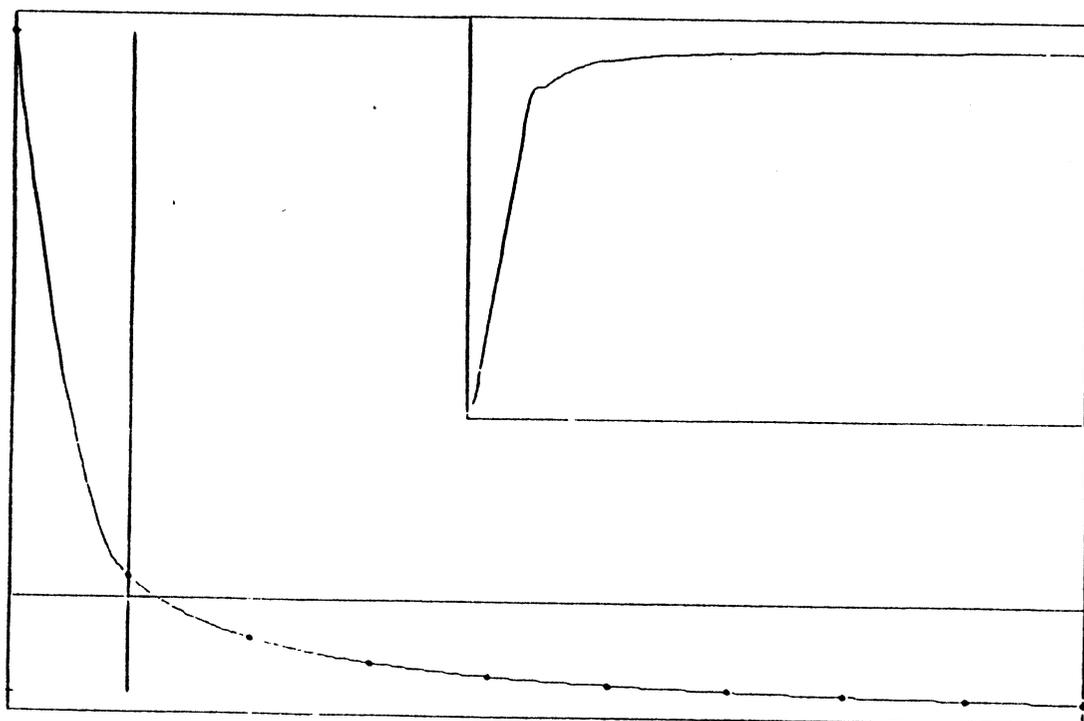


(b)

Fig.13. Splines cubiques C^1 pour les données monotones et convexes 3.
(a) Méthode (F/C)
(b) Méthode (B)



(a)



(b)

Fig.14. Splines pour les données monotones et convexes 3.
(a) Méthode (C/M)
(b) Méthode (H/N)

La condition de monotonie est bien respectée par la cubique de (F/C) qui apparaît dans la Fig.13(a). Mais en regardant la dérivée première on s'aperçoit que la convexité n'est pas conservée. Par contre, la méthode paramétrée de (B) produit une solution meilleure (Fig.13(b)).

La prise en compte des deux conditions : convexité et décroissance des données, par la méthode (C/M) permet d'arriver à une courbe satisfaisante du point de vue graphique. Voir la Fig. 14(a).

Enfin, la convexité est respectée, par la courbe de (H/N) lorsque $\alpha=0.3$, mais sa dérivée présente un "palier" dans l'intervalle $]t_2, t_3[$.

• Comparaison du temps de calcul

Pour donner une indication sur l'évolution du temps de calcul des algorithmes, continuons à considérer la fonction $f_2(t)$, $t \in [-1,8]$, dont le nombre des points à interpoler sera $n=50,100,150,200$. Le tableau 5 donne cette information.

n	Méthode						
	McA/R	S	F/C	F/B	B	C/M	H/N
50	0.44	0.56	0.22	0.16	0.16	0.30	0.32
100	0.86	1.12	0.44	0.34	0.34	0.60	0.66
150	1.30	1.68	0.64	0.5	0.52	0.88	0.98
200	1.76	2.22	0.86	0.6	0.68	1.18	1.30

Tableau 5. Temps (en secondes) de calcul pour $f_2(t)$, $t \in [-1,8]$ pour n abscisses équidistantes.

On constate que :

- Pour les splines quadratiques, l'algorithme de (McA/R) est plus rapide que celui de (S), ce dernier étant le moins rapide de tous.

- Parmi les splines cubiques, le calcul des dérivées par les formules proposées par (F/B) est le plus rapide. Un résultat semblable est trouvé lorsqu'on considère la forme paramétrée de la fonction; ceci peut être expliqué par l'uniformité de la paramétrisation utilisée par (B) ce qui réduit le nombre (et donc le temps) de calculs. L'algorithme de (F/C) est à deux pas; de même que celui de (C/M) donc ils prennent plus de temps.
- La durée de calcul pour (H/N) se place entre celle de (C/M), la plus lente pour les cubiques, et celle de (McA/R), la plus rapide pour les quadratiques.

1.7. CONCLUSIONS

Nous avons essayé différentes méthodes d'interpolation qui préservent la forme des données, en utilisant des fonctions splines polynômiales.

Les splines quadratiques semblent les mieux adaptées pour conserver la monotonie et/ou la convexité des données. De plus, elles peuvent être utilisées pour des données quelconques, sans les diviser en régions monotones et convexes. Cependant, il faut rappeler qu'entre deux points données on insère (au plus) un point supplémentaire, en augmentant ainsi l'encombrement de la mémoire.

Si l'on doit considérer exclusivement la condition de monotonie alors la méthode (F/B), ou (B) pour le cas paramétré, est conseillée. Ces procédés n'ajoutent pas de points supplémentaires et sont les plus rapides.

CHAPITRE II

Interpolation Rationnelle pour Données Monotones et Convexes



II.1. INTRODUCTION

Ce chapitre concerne l'utilisation des fonctions rationnelles pour l'interpolation avec conditions de monotonie et de convexité. Ces courbes présentent un avantage : la monotonie et la convexité sont déterminées par des conditions nécessaires et suffisantes "simples", à partir desquelles on peut définir des algorithmes pour leur construction.

Dans le cas de données convexes, Schaback [100] a défini une fonction spline rationnelle (2/1) ayant la même propriété.

Dupuy [34] et Delbourgo et Gregory [28], [47] ont appliqué, pour des données monotones, les fonctions rationnelles (2/2) par morceaux.

C'est Späth [107] qui a introduit la notion de tension dans le contexte des fonctions splines rationnelles. Ces fonctions, pour des paramètres de tension assez grands, s'approchent de la polygonale qui relie les données. Tandis que, pour des paramètres égaux à zéro, on retrouve les fonctions splines cubiques. Ces propriétés permettent de construire une courbe qui conserve la monotonie et/ou la convexité des données.

Pour résoudre ce problème, nous recourons aux fonctions rationnelles (1/1) et (2/1) par morceaux.

Nous remarquons qu'il n'existe pas toujours une interpolante homographique d'Hermite entre deux points dont la position et la pente sont connues. Si la donnée "hermitienne" d'interpolation sur l'intervalle compris entre deux noeuds a la propriété de monotonie, (c'est-à-dire la corde et les tangentes extrémales sont à pentes positives), nous montrons qu'on peut insérer un point, pour avoir une représentation homographique continue par morceaux qui reste finie et monotone dans l'intervalle. Dans le cas où l'on veut respecter aussi la convexité, nous donnons les conditions pour le choix du noeud intermédiaire.

A partir de ces résultats, nous dégageons un algorithme qui produit une interpolante monotone (et convexe) pour données monotones (et convexes)

D'autre part, nous construisons une fonction rationnelle (2/1) par morceaux pour laquelle la condition nécessaire de convexité est aussi suffisante. Cette condition est : que la pente de la corde qui relie deux points consécutifs appartienne à l'intervalle dont les bornes sont les dérivées aux noeuds. De plus, si les données possèdent le caractère monotone, cette fonction l'aura aussi. La méthode n'a pas besoin de l'insertion de points supplémentaires.

Les techniques que nous développons dans ces deux cas sont de type local, donnent des solutions continûment différentiables et peuvent s'avérer intéressantes lorsqu'on veut que la courbe satisfasse à la fois les conditions de monotonie et de convexité.

En fin de chapitre nous donnerons divers exemples numériques traités et leurs résultats.

II.2. FONCTIONS SPLINES HOMOGRAPHIQUES MONOTONES ET CONVEXES

Dans ce paragraphe nous nous sommes intéressés au problème d'interpolation suivant :

Problème 2.1

Soit $\Delta : a = t_1 < t_2 < \dots < t_n = b$ une partition de l'intervalle $[a, b]$ et $D = \{(t_i, z_i)\} \quad i=1, 2, \dots, n$ un ensemble de données strictement monotones (et convexes). On cherche une fonction s telle que :

- (i) s soit formée par morceaux de fonctions homographiques
- (ii) $s(t_i) = z_i \quad i=1, 2, \dots, n$
- (iii) $s \in C^1 [a, b]$
- (iv) s soit strictement monotone (et convexe) sur $[a, b]$. \square

Nous rappelons que les données sont *strictement monotones* si

$$m_i > 0 \quad \text{ou} \quad m_i < 0 \quad i=1, 2, \dots, n-1$$

et qu'elles sont *strictement convexes (concaves)* si

$$m_i - m_{i-1} > 0 \quad (m_i - m_{i-1} < 0) \quad i=2, 3, \dots, n-1$$

où $m_i = (z_{i+1} - z_i) / h_i$, $h_i = t_{i+1} - t_i$ $i=1,2,\dots,n-1$

2.1. Un problème d'interpolation d'Hermite pour deux points

On va traiter d'abord un problème d'interpolation d'Hermite pour deux points, qui va nous servir pour résoudre le problème 2.1.

Problème (PIH)

Soit $t_1 < t_2$. Supposons que z_1, z_2, λ_1 et λ_2 sont des nombres réels donnés. Trouver une fonction $s \in C^1[t_1, t_2]$ telle que

$$(2.1) \quad s(t_i) = z_i, \quad s'(t_i) = \lambda_i \quad i=1,2$$

La proposition suivante montre sous quelles conditions une fonction homographique peut être solution de (PIH).

Proposition 2.1

Supposons $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$ où $m = (z_2 - z_1) / (t_2 - t_1)$. Une condition nécessaire et suffisante pour qu'il existe une fonction rationnelle (1/1) qui résolve le problème (PIH) est que

$$\lambda_1 \lambda_2 = m^2.$$

Dans ce cas, la solution peut s'exprimer sous la forme :

$$(2.2) \quad s(t) = \frac{(z_2 - z_1)z_1 + (z_2 \lambda_1 - z_1 m)(t - t_1)}{(z_2 - z_1) + (\lambda_1 - m)(t - t_1)} \quad \text{pour } t \in [t_1, t_2]$$

Preuve

Elle est élémentaire. \square

On remarque que si $\lambda_1 = \lambda_2 = m$ alors la solution est une droite.

La fonction (2.2) peut s'écrire d'une manière plus appropriée pour le traitement numérique :

$$(2.3) \quad s(t) = z_1 + \frac{\lambda_1 (z_2 - z_1) (t - t_1)}{(z_2 - z_1) + (\lambda_1 - m)(t - t_1)}$$

Les dérivées première et seconde de s sont

$$(2.4) \quad s'(t) = \lambda_1 (z_2 - z_1)^2 / L^2(t)$$

$$(2.5) \quad s''(t) = -2\lambda_1 (\lambda_1 - m) (z_2 - z_1)^2 / L^3(t)$$

où $L(t) = (z_2 - z_1) + (\lambda_1 - m)(t - t_1)$.

Nous pouvons énoncer la

Proposition 2.2

Supposons que $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$ et $\lambda_1 \lambda_2 = m^2$.

La fonction s (2.3), solution de (PIH), est monotone sur $I = [t_1, t_2]$.

De plus, si $\lambda_1 < m$ (donc $\lambda_1 < m < \lambda_2$) alors s est convexe sur I, tandis que si $\lambda_1 > m$ (donc $\lambda_1 > m > \lambda_2$) alors s est concave sur I.

Preuve

Des hypothèses et de (2.4) on découle immédiatement que s' a le même signe que λ_1 et λ_2 dans I. La monotonie est donc établie.

Pour la convexité, la condition $s''(t) > 0$ est équivalente à $(\lambda_1 - m)/L(t) < 0$. L'étude du signe nous donne que l'unique condition consistante est $\lambda_1 < m$.

On raisonne d'une façon analogue pour la concavité. \square

En général, la condition $\lambda_1 \lambda_2 = m^2$ n'est pas remplie. Alors on va chercher une solution de (PIH) utilisant une spline homographique.

On définit une **fonction spline homographe** (FSH) aux noeuds $x_1 < x_2 < \dots < x_p$ comme une fonction formée par morceaux de fonctions homographiques se raccordant aux x_i ainsi que leurs dérivées premières.

Posons

$$\theta = \begin{cases} 1 & \text{si } \lambda_1 > 0 \text{ et } \lambda_2 > 0 \\ -1 & \text{si } \lambda_1 < 0 \text{ et } \lambda_2 < 0 \end{cases}$$

Proposition 2.3

Supposons $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$.

Pour tout $\xi \in]t_1, t_2[$ il existe une spline homographe s avec un noeud en ξ qui résout (PIH).

Nous pouvons écrire :

$$(2.6) \quad s(t) = \begin{cases} [a_1 u_1 + b_1(1-u_1)] / [(c_1 u_1 + d_1(1-u_1))] & t_1 < t < \xi \\ [a_2 u_2 + b_2(1-u_2)] / [(c_2 u_2 + d_2(1-u_2))] & \xi \leq t < t_2 \end{cases}$$

avec

$$(2.7) \quad \begin{aligned} a_1 &= z_1 \beta \theta \sqrt{\lambda_1 \lambda_2} + z_2 \alpha \lambda_1 & b_1 &= z_1(z_2 - z_1) \\ c_1 &= \beta \theta \sqrt{\lambda_1 \lambda_2} + \alpha \lambda_1 & d_1 &= z_2 - z_1 \\ a_2 &= z_2(z_2 - z_1) & b_2 &= z_1 \beta \lambda_2 + z_2 \alpha \theta \sqrt{\lambda_1 \lambda_2} \\ c_2 &= z_2 - z_1 & d_2 &= \beta \lambda_2 + \alpha \theta \sqrt{\lambda_1 \lambda_2} \end{aligned}$$

où

$$(2.8) \quad \begin{aligned} u_1 &= (t - t_1) / \alpha & , & \quad u_2 = (t - \xi) / \beta \\ \alpha &= \xi - t_1 & , & \quad \beta = t_2 - \xi \end{aligned}$$

Preuve

On vérifie aisément que la fonction s , définie par (2.6) -(2.8) remplit les conditions d'interpolation et satisfait $s(\xi^-) = s(\xi^+)$ et $s'(\xi^-) = s'(\xi^+)$. \square

On a

$$s(\xi) = \bar{z} = z_1 + \frac{\alpha\lambda_1(z_2 - z_1)}{\alpha\lambda_1 + \beta\theta\sqrt{\lambda_1\lambda_2}}$$

et

$$s'(\xi) = \bar{\lambda} = \lambda_1 \left(\frac{z_2 - z_1}{\alpha\lambda_1 + \beta\theta\sqrt{\lambda_1\lambda_2}} \right)^2$$

Par la suite on va étudier la forme de cette spline.

Proposition 2.4.

Supposons $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$. La spline (2.6) - (2.8) a les propriétés suivantes :

- (i) s est monotone sur I
- (ii) Si $\lambda_1 < \lambda_2$ alors s est convexe sur I si et seulement si

$$(2.9) \quad \lambda_1 < \bar{\lambda} < \lambda_2$$

De la même manière, si $\lambda_1 > \lambda_2$ alors s est concave sur I si et seulement si

$$(2.10) \quad \lambda_1 > \bar{\lambda} > \lambda_2$$

Preuve

En effet,

$$s'(t) = \begin{cases} \lambda_1(z_2 - z_1)^2 / L_1^2(t) & t_1 < t < \xi \\ \lambda_2(z_2 - z_1)^2 / L_2^2(t) & \xi \leq t < t_2 \end{cases}$$

où $L_1(t) = c_1 u_1 + d_1(1 - u_1)$, $L_2(t) = c_2 u_2 + d_2(1 - u_2)$

donc s' a le même signe que λ_1 , λ_2 et m dans I .

Limitons-nous à prouver le cas convexe de (ii). On a

$$s''(t) = \begin{cases} -2 \lambda_1 (c_1 - d_1) (z_2 - z_1)^2 / L_1^3(t) & t_1 < t < \xi \\ -2 \lambda_2 (c_2 - d_2) (z_2 - z_1)^2 / L_2^3(t) & \xi \leq t < t_2 \end{cases}$$

Comme le dénominateur représente un morceau de droite, il n'y a pas de pôles si c_1 et d_1 (c_2 et d_2) ont le même signe. En tenant compte que les signes de λ_1 et d_1 (λ_2 et d_2) sont les mêmes, on déduit que $s''(t) > 0$ si et seulement si $c_1 - d_1 < 0$ ($c_2 - d_2 < 0$).

Ceci entraîne

$$(2.11) \quad \beta \theta \sqrt{\lambda_1 \lambda_2} + \alpha \lambda_1 < z_2 - z_1 < \beta \lambda_2 + \alpha \theta \sqrt{\lambda_1 \lambda_2}.$$

Ces inégalités sont équivalentes à $\lambda_1 < \bar{\lambda} < \lambda_2$. \square

Attirons l'attention sur le fait que l'on peut placer arbitrairement le noeud ξ dans l'intervalle $]t_1, t_2[$ tout en préservant la monotonie. Ce n'est pas le cas pour la convexité. Il se pose la question du choix du noeud ξ pour la spline, qui vérifie déjà la monotonie, demeure convexe.

Nous rappelons d'abord qu'une condition nécessaire de convexité (concavité) est

$$\lambda_1 < m < \lambda_2 \quad (\lambda_1 > m > \lambda_2).$$

Proposition 2.5

Soit $\text{sgn}(\lambda_1) = \text{sgn}(\lambda_2) = \text{sgn}(m)$.

- (i) Si $(\lambda_1 - m)(\lambda_2 - m) \geq 0$ alors il existe un point d'inflexion dans l'intervalle I.
- (ii) Si, par contre, $(\lambda_1 - m)(\lambda_2 - m) < 0$
 - a) la condition $(\theta \sqrt{\lambda_1 \lambda_2} - m)(\lambda_2 - m) < 0$ implique que pour tout ξ tel que

$$(2.12) \quad \xi \in]t_1, \bar{\xi}] \quad \text{ou} \quad \bar{\xi} = t_1 + (\lambda_2 - m)(t_2 - t_1) / (\lambda_2 - \theta\sqrt{\lambda_1\lambda_2})$$

la fonction s(2.6) - (2.8) soit :

$$(2.13) \quad \begin{array}{l} - \text{convexe sur } I \text{ si } \lambda_1 < \lambda_2 \\ - \text{concave sur } I \text{ si } \lambda_1 > \lambda_2 \end{array}$$

b) la condition $(\theta\sqrt{\lambda_1\lambda_2} - m)(\lambda_2 - m) > 0$ implique que pour tout ξ tel que

$$(2.14) \quad \xi \in [\underline{\xi}, t_2[\quad \text{ou} \quad \underline{\xi} = t_2 + (\lambda_1 - m)(t_2 - t_1) / (\theta\sqrt{\lambda_1\lambda_2} - \lambda_1).$$

la fonction s(2.6) - (2.8) a les propriétés (2.13).

Preuve

Avec la condition $(\lambda_1 - m)(\lambda_2 - m) \geq 0$ ni (2.9) ni (2.10) ne sont vérifiées donc s n'est ni concave ni convexe sur tout l'intervalle I.

Bornons-nous à démontrer la partie a) de (ii) si $\lambda_1 < \lambda_2$, car les autres cas sont similaires. Des hypothèses, on déduit les inégalités suivantes :

$$\lambda_1 < \frac{\alpha\lambda_1 + \beta\theta\sqrt{\lambda_1\lambda_2}}{t_2 - t_1} < \theta\sqrt{\lambda_1\lambda_2} < m < \frac{\beta\lambda_2 + \alpha\theta\sqrt{\lambda_1\lambda_2}}{t_2 - t_1} < \lambda_2$$

$$\text{Il s'ensuit que } t_1 < \bar{\xi} < t_2 \text{ car } 0 < \frac{\lambda_2 - m}{\lambda_2 - \theta\sqrt{\lambda_1\lambda_2}} < 1$$

Après quelques calculs supplémentaires, on détermine que pour tout $\xi \in]t_1, \bar{\xi}]$ on a la condition (2.9) qui nous permet d'achever la preuve. \square

2.2. Un algorithme pour l'interpolation des données monotones (et convexes).

On se propose d'utiliser les résultats du paragraphe précédent sur la monotonie (et convexité) locale pour construire numériquement une spline homographique globalement monotone (et convexe).

Dans tout ce paragraphe nous reprendrons les notations :

$$I_i = [t_i, t_{i+1}] , m_i = (z_{i+1} - z_i) / (t_{i+1} - t_i) \quad i=1, 2, \dots, n-1$$

Si un noeud doit être inséré dans l'intervalle I_i nous l'appellerons ξ_i .

Deux questions importantes sont à régler : le calcul des dérivées aux noeuds et le choix des points à ajouter.

• Calcul des dérivées (λ_i) $i=1, 2, \dots, n$

La condition nécessaire de monotonie globale est que les pentes λ_i aient le même signe.

Considérons que les données ont globalement un comportement uniforme, c'est-à-dire, sont monotones et convexes ou monotones et concaves, alors les dérivées λ_i doivent être calculées pour assurer la monotonie et la convexité (concavité) locales dans chaque intervalle I_i .

Il découle de tout ceci, qu'on doit avoir pour la monotonie et la convexité strictes:

$$(2.15) \quad \left\{ \begin{array}{l} \operatorname{sgn}(\lambda_i) = \operatorname{sgn}(\lambda_{i+1}) = \operatorname{sgn}(m_i) \\ \lambda_1 < m_1 < \lambda_2 < \dots < m_{n-1} < \lambda_n \end{array} \right. \quad i=1, 2, \dots, n-1$$

et pour la monotonie et la concavité strictes:

$$(2.16) \quad \left\{ \begin{array}{l} \text{sgn}(\lambda_i) = \text{sgn}(\lambda_{i+1}) = \text{sgn}(m_i) \\ \lambda_1 > m_1 > \lambda_2 > \dots > m_{n-1} > \lambda_n \end{array} \right. \quad i=1,2,\dots,n-1$$

Pour calculer les dérivées aux noeuds on va employer la méthode du §1.3.12. qui consiste à faire passer un polynôme de degré deux à travers (t_{i-1}, z_{i-1}) , (t_i, z_i) , (t_{i+1}, z_{i+1}) pour $i=2,3,\dots,n-1$; et prendre ensuite λ_i égal à la valeur de la dérivée de ce polynôme au point central t_i . Deux conditions de bout sont ajoutées. On obtient ainsi les schémas (D0) et (D1) (voir pp. 17).

Une autre méthode, que nous utiliserons, pour trouver λ_i est suggérée par la précédente, mais au lieu d'un polynôme quadratique, nous prendrons une fonctions homographique. Donc

$$(D2) \quad \left\{ \begin{array}{l} \lambda_1 = \begin{cases} 0 & \text{si } z_3 - z_1 = 0 \\ m_1^2 (t_3 - t_1) / (z_3 - z_1) & \text{sinon} \end{cases} \\ \lambda_i = \begin{cases} 0 & \text{si } z_{i+1} - z_{i-1} = 0 \\ m_i m_{i-1} (t_{i+1} - t_{i-1}) / (z_{i+1} - z_{i-1}) & \text{sinon} \end{cases} \\ \lambda_n = \begin{cases} 0 & \text{si } z_n - z_{n-2} = 0 \\ m_{n-1}^2 (t_n - t_{n-2}) / (z_n - z_{n-2}) & \text{sinon.} \end{cases} \end{array} \right. \quad i=2,3,\dots,n-1$$

Les formules (D1) et (D2) produisent des approximations en h^2 des dérivées aux noeuds intérieurs, où $h = \max(h_i)$. L'intérêt de (D2) est que $\lim_{i \rightarrow n-1} \lambda_i = 0$ lorsque $m_{i-1} \rightarrow 0$ ou $m_i \rightarrow 0$, ce qui n'est pas assuré par (D1).

Remarque

Les conditions (2.15) et (2.16) sont vérifiées par les schémas (D1) et (D2) pour des données strictement monotones et convexes (concaves).

En effet, ce résultat découle immédiatement de la forme des données et du fait que λ_i est une combinaison convexe de m_{i-1} et m_i ($i=2,3,\dots,n-1$). □

• **Choix des noeuds supplémentaires**

La proposition 2.5 affirme que s'il faut insérer un noeud dans l'intervalle I_i , sa position arbitraire n'affecte pas la monotonie locale (donc globale).

Signalons que nous pouvons faire appel aux propositions 2.5 et 2.6 pour produire une spline localement convexe ou concave sur l'intervalle I_i , lorsque $(\lambda_i - m_i)(\lambda_{i+1} - m_i) < 0$, tout en restant monotone. En effet, il suffit de vérifier les conditions (2.9) ou (2.10) en prenant le noeud ξ_i dans I_i , en conformité avec la proposition 2.6, c'est-à-dire :

$$(2.17) \quad t_i < \xi_i \leq t_{i+1} + \frac{(\lambda_{i+1} - m_i)(t_{i+1} - t_i)}{\lambda_{i+1} - \theta_i \sqrt{\lambda_i \lambda_{i+1}}} \quad \text{si } (\theta_i \sqrt{\lambda_i \lambda_{i+1}} - m_i)(\lambda_{i+1} - m_i) < 0$$

$$(2.18) \quad t_{i+1} + \frac{(\lambda_i - m_i)(t_{i+1} - t_i)}{\theta_i \sqrt{\lambda_i \lambda_{i+1}} - \lambda_i} \leq \xi_i < t_{i+1}$$

si $(\theta_i \sqrt{\lambda_i \lambda_{i+1}} - m_i)(\lambda_{i+1} - m_i) > 0$

où $\theta_i = \begin{cases} 1 & \text{si } \lambda_i > 0 \text{ et } \lambda_{i+1} > 0 \\ -1 & \text{si } \lambda_i < 0 \text{ et } \lambda_{i+1} < 0 \end{cases}$

Maintenant , nous pouvons définir une algorithmme pour calculer une F.S.H. monotone (et convexe).

Algorithmme A.1

1. (calcul des pentes des cordes)

$$m_i = (z_{i+1} - z_i) / (t_{i+1} - t_i) \quad i=1,2,\dots,n-1$$

2. (calcul des dérivées aux noeuds)

Utiliser les formules (D1) ou (D2). Si $\lambda_i > 0$ alors $\theta := 1$ sinon $\theta := -1$

3. (calcul des noeuds et des coefficients)

$j := 0$;

Pour $i := 1$ jusqu'à $n-1$

Si $\lambda_i \lambda_{i+1} = m_i^2$ alors

$j := j+1$; $x_j := t_j$;

$a_j := z_j$; $b_j := \lambda_j (z_{i+1} - z_i)$;

$c_j := z_{i+1} - z_i$; $d_j := \lambda_i - m_i$;

Sinon , $\lambda_i \lambda_{i+1} \neq m_i^2$,

$p := \lambda_i - m_i$; $q := \lambda_{i+1} - m_i$;

si $pq \geq 0$ alors $\xi_i := (t_{i+1} - t_i) / 2$

sinon, $pq < 0$,

$r := \theta \sqrt{\lambda_i \lambda_{i+1}} - m_i$

si $pr < 0$ alors

$$\xi_i := t_{i+1} + \frac{\rho(t_{i+1} - t_i)}{(\theta \sqrt{\lambda_i \lambda_{i+1}} - \lambda_i)}$$

ou

$$\xi_i := t_{i+1} + \frac{\rho(t_{i+1} - t_i)}{2(\theta \sqrt{\lambda_i \lambda_{i+1}} - \lambda_i)}$$

sinon

$$\xi_j := t_j + \frac{q(t_{j+1} - t_j)}{(\lambda_{j+1} - \theta \sqrt{\lambda_j \lambda_{j+1}})}$$

ou

$$\xi_j := t_{j+1} - \frac{q(t_{j+1} - t_j)}{2(\lambda_{j+1} - \theta \sqrt{\lambda_j \lambda_{j+1}})}$$

$$\begin{aligned} \alpha &:= \xi_j - t_j & ; & \beta := t_{j+1} - \xi_j ; \\ j &:= j+1 & ; & x_j := t_j \\ a_j &:= z_j & ; & b_j := \alpha \lambda_j (z_{j+1} - z_j) \\ c_j &:= \alpha (z_{j+1} - z_j) ; & d_j &:= \beta \theta \sqrt{\lambda_j \lambda_{j+1}} + \alpha \lambda_j (z_{j+1} - z_j) \\ j &:= j+1 & ; & x_j := \xi_j ; \\ v &:= \beta \lambda_{j+1} + \alpha \theta \sqrt{\lambda_j \lambda_{j+1}} \\ a_j &:= (z_j \beta \lambda_{j+1} + z_{j+1} \alpha \theta \sqrt{\lambda_j \lambda_{j+1}}) / v \\ b_j &:= \beta \lambda_{j+1} (z_{j+1} - z_j)^2 / v \\ c_j &:= \beta v \\ d_j &:= (z_{j+1} - z_j) - v \end{aligned}$$

$$k := j$$

Commentaires

L'algorithme A1 produit pour des données monotones les coefficients d'une fonction spline homographique d'interpolation telle que :

- $s(t) = a_i + b_i(t-x_i) / (c_i + d_i(t-x_i)) \quad x_i \leq t < x_{i+1}, \quad i = 1, 2, \dots, k$
- s est monotone sur $[a, b]$
- s est co-convexe avec les données, dans le sens qu'elle a des points d'inflexion exclusivement dans les intervalles où la convexité des données change. Evidemment, si les données sont globalement convexes, alors s est convexe sur $[a, b]$.
- $s \in C^1[a, b]$ si la monotonie des données est stricte.

Il faut remarquer dans le cas $p \geq 0$ le noeud ξ_i peut être choisi librement dans l'intervalle I_i ; et nous le prenons au milieu de cet intervalle. Si $p < 0$ nous prenons ξ_i au milieu ou dans la borne de l'intervalle correspondant (voir (2.15) - (2.16))

2.3. Une autre approche pour les fonctions splines homographiques

Signalons qu'on peut envisager la construction d'une FSH monotone sans élargir l'ensemble des noeuds, de la manière suivante :

Proposition 2.6

Soit D un ensemble de données strictement monotones et λ_i un nombre réel tel que $\text{sgn}(\lambda_i) = \text{sgn}(m_i)$. Alors, il existe une seule FSH monotone s sans noeuds supplémentaires qui satisfait les conditions :

$$s(t_i) = z_i \quad (i=1,2,\dots,n) \quad \text{et} \quad s'(t_i) = \lambda_i$$

Preuve

La preuve immédiate compte tenu des propositions 2.1 et 2.2. \square

Pour déterminer explicitement les dérivées de λ_i aux noeuds nous disposons de la formule

$$\lambda_i \lambda_{i+1} = m_i^2 \quad i=1,2,\dots,n-1$$

en partant de λ_1 connue.

Il faut cependant noter que la stabilité numérique de la procédure n'est pas garantie. Pour cette raison nous ne l'avons pas utilisée dans la pratique.

II. 3. FONCTIONS SPLINES RATIONNELLES QUADRATIQUES MONOTONES

Ce paragraphe expose les techniques pour l'interpolation par des fonctions rationnelles (2/2) en présence de condition de monotonie. (voir [28], [34], [47]).

L'intérêt de ces courbes est le suivant : sur un intervalle, la condition nécessaire et suffisante pour que la représentation soit monotone (pourvu qu'elle ne possède aucun pôle) est que les pentes de la corde et des tangentes extrémales aient le même signe.

L'équation représentative de la fonction s dans $[t_i, t_{i+1}]$ est de la forme

$$(3.1) \quad s(t) = \frac{a_i u^2 + 2b_i u(1-u) + c_i (1-u)^2}{u^2 + 2d_i u(1-u) + e_i (1-u)^2}$$

où $u = (t - t_i) / h_i$

On peut introduire comme paramètres, les valeurs z_i, z_{i+1} , les pentes extrémales λ_i, λ_{i+1} et l'une des courbures β_i ou β_{i+1} ($\beta_i = s''(t_i)$) pour déterminer les coefficients de s .

Delbourgo et Gregory [47] prennent $e_i = 1$; avec cette simplification on n'a pas besoin de considérer la courbure comme paramètre.

3.1. Une interpolante rationnelle (2/2) monotone et C^1

Proposition 3.1

Supposons que les données $D = \{(t_i, z_i)\} \quad i=1,2,\dots,n$ sont monotones et que

$$(3.2) \quad \left\{ \begin{array}{ll} \operatorname{sgn}(\lambda_i) = \operatorname{sgn}(\lambda_{i+1}) = \operatorname{sgn}(m_i) & \text{pour } m_i \neq 0 \\ \lambda_i = \lambda_{i+1} = 0 & \text{pour } m_i = 0 \end{array} \right.$$

Alors la fonction s définie par

$$(3.3) \quad s(t) = \begin{cases} p_i(u)/q_i(u) & \text{si } m_i \neq 0 \\ z_i & \text{si } m_i = 0 \end{cases} \quad t_i \leq t \leq t_{i+1}$$

où

$$(3.4) \quad \begin{aligned} p_i(u) &= z_{i+1}u^2 + (z_i \lambda_{i+1} + z_{i+1} \lambda_i)m_i^{-1}u(1-u) + z_i(1-u)^2 \\ q_i(u) &= u^2 + (\lambda_{i+1} + \lambda_i)m_i^{-1}u(1-u) + (1-u)^2. \end{aligned}$$

possède les propriétés suivantes :

- (i) s n'a pas de pôles dans l'intervalle $[a,b]$
- (ii) $s(t_i) = z_i$, $s'(t_i) = \lambda_i$ $i=1,2,\dots,n$
- (iii) s est monotone sur $[a,b]$
- (iv) $s \in C^1[a,b]$.

Preuve

La preuve est donnée dans [47].□

Pour les applications pratiques, on doit fournir les dérivées aux noeuds vérifiant (3.2), dans le cas contraire, nous les calculons avec les procédés (D1) ou (D2) du paragraphe 2.2.

3.2. Une spline rationnelle quadratique monotone

Considérons maintenant que les données D sont strictement monotones.

Nous rappelons que pour la fonction spline cubique d'interpolation, les dérivées $\lambda_1, \lambda_2, \dots, \lambda_n$ sont déterminées à partir de la condition suivante : s doit être deux fois continûment différentiable. Le raccordement aux noeuds conduit à un système linéaire tridiagonal ([56],[87]).

Pour que la fonction définie dans la section 3.1. ait aussi la courbure continue, on impose aux noeuds intérieurs $s''(t_i^-) = s''(t_i^+)$, ce qui nous donne:

$$-\frac{2}{h_{i-1}} [m_{i-1} + \lambda_i - \lambda_i(\lambda_{i-1} + \lambda_i)/m_{i-1}] = \frac{2}{h_i} [m_i + \lambda_i - \lambda_i(\lambda_i + \lambda_{i+1})/m_i]$$

Cette expression nous la réécrivons:

$$(3.5) \quad (a_{i-1} + a_i)\lambda_i^2 + (a_i\lambda_{i+1} + a_{i-1}\lambda_{i-1} - b_i)\lambda_i - c_i = 0 \quad \text{où}$$

$$(3.6) \quad a_i = 1/(h_i m_i), \quad b_i = 1/h_{i-1} + 1/h_i, \quad c_i = m_{i-1}/h_{i-1} + m_i/h_i$$

$i=2,3,\dots,n-1$

Connaissant λ_1 et λ_n on arrive à un système de $n-2$ équations non-linéaires où les inconnues sont $\lambda_2, \lambda_3, \dots, \lambda_{n-1}$.

Notons que l'équation (3.5) est quadratique par rapport à la variable λ_i , dont la racine positive (négative) est :

$$(3.7) \quad \lambda_i = \frac{1}{2(a_{i-1} + a_i)} \left\{ b_i - a_{i-1} \lambda_{i-1} - a_i \lambda_{i+1} + \left[(b_i - a_{i-1} \lambda_{i-1} - a_i \lambda_{i+1})^2 + 4(a_{i-1} + a_i)c_i \right]^{1/2} \right\}$$

si $z_1 < z_2 < \dots < z_n$ ($z_1 > z_2 > \dots > z_n$).

Si λ_1 et λ_n sont données telles que $\lambda_1 > 0$ et $\lambda_n > 0$ alors la preuve de l'existence et de l'unicité de la solution $[\lambda_1, \lambda_2, \dots, \lambda_n]$ telle que $\lambda_i > 0$ ($\lambda_i < 0$) pour $i=1,2,\dots,n$ est établie en construisant une itération de type Jacobi :

$$(3.8) \quad \lambda_i^{(k)} = \frac{1}{2(a_{i-1} + a_i)} \left\{ b_i - a_{i-1} \lambda_{i-1}^{(k-1)} - a_i \lambda_{i+1}^{(k-1)} + \left[(b_i - a_{i-1} \lambda_{i-1}^{(k-1)} - a_i \lambda_{i+1}^{(k-1)})^2 + 4(a_{i-1} + a_i)c_i \right]^{1/2} \right\}$$

pour $i=2,\dots,n-2$

et $\lambda_1^{(k)} = \lambda_1^{(k-1)} = \lambda_1, \quad \lambda_n^{(k)} = \lambda_n^{(k-1)} = \lambda_n$

Dans la pratique, il suffit d'utiliser une itération type GaussSeidel. Alors on obtient l'algorithme suivant :

Algorithme A.2.

1. $k:=0$;

. Donner $\lambda_i^{(0)}$ t.q $\lambda_i > 0$ ($\lambda_i < 0$) pour $i=1,2,\dots,n$

. Donner la tolérance $\epsilon > 0$

2. (Itération k-ième)

$k:=k+1$

$$\lambda_1^{(k)} = \lambda_1^{(k-1)} = \lambda_1^{(0)}$$

$$\lambda_i^{(k)} = \frac{1}{2(a_{i-1}+a_i)} \left\{ b_i - a_{i-1} \lambda_{i-1}^{(k)} - a_i \lambda_{i+1}^{(k-1)} + \left[(b_i - a_{i-1} \lambda_{i-1}^{(k)} - a_i \lambda_{i+1}^{(k-1)})^2 + 4(a_{i-1}+a_i)c_i \right]^{1/2} \right\}$$

$$\lambda_n^{(k)} = \lambda_n^{(k-1)} = \lambda_n^{(0)}$$

3. Si $\max_{2 \leq i \leq n-1} |\lambda_i^{(k)} - \lambda_i^{(k-1)}| < \epsilon$ alors stop. \square

Pour démarrer l'algorithme ci-dessus, Delbourgo et Gregory [47] proposent de prendre

$$\lambda_i^{(0)} = [c_i / (a_{i-1}+a_i)]^{1/2} \quad i=2,3,\dots,n-1.$$

On peut aussi penser à utiliser le schéma (D1) ou (D2) du paragraphe 2.2.

3.3. Préservation de la co-monotonie

Un ensemble de données peut ne pas être globalement monotone, mais changer autour d'un point de monotone croissante à monotone décroissante (ou viceversa).

Alors on procède de la manière suivante : on repère les régions qui ont un même comportement et on applique à chaque région les techniques décrites précédemment, en posant $\lambda_i = 0$ lorsque $m_{i+1}m_i \leq 0$.

La fonction résultante suit la variation de la monotonie et on dira qu'elle est **co-monotone avec les données**.

II.4. FONCTIONS SPLINES RATIONNELLES (2/1) CONVEXES

Nous venons de voir l'interpolation monotone. Dans ce qui suit nous abordons les méthodes qui permettent de refléter la convexité des données.

L'idée de base est simple : on cherche une fonction qui réalise directement une représentation convexe sur un intervalle, ayant comme condition nécessaire et suffisante que la donnée "hermitienne" soit convexe (c'est-à-dire que la valeur de la pente de la corde soit comprise entre les valeurs des pentes des tangentes extrémales). Nous l'avons trouvée en utilisant les fonctions rationnelles (2/1).

Nous supposons que les données $D = \{(t_i, z_i) \mid i=1,2,\dots,n\}$ sont strictement convexes (concaves).

4.1. Une interpolante rationnelle 2/1 convexe et C^1

Considérons la fonction rationnelle (2/1) par morceaux suivante :

$$(4.1) \quad s(t) = p_i(t) / q_i(t) \quad \text{où}$$

$$(4.2) \quad p_i(t) = z_{i+1}(\lambda_{i+1} - m_i) h_i + [\lambda_i h_i (\lambda_{i+1} m_i) - z_i (\lambda_{i+1} - 2m_i + \lambda_i)] \\ (t-t_i) + (m_i^2 - \lambda_i \lambda_{i+1}) (t-t_i)^2$$

$$(4.3) \quad q_i(t) = (\lambda_{i+1} - m_i)h_i - (\lambda_{i+1} - 2m_i + \lambda_i)(t - t_i)$$

pour $t \in [t_i, t_{i+1}]$ et $i = 1, 2, \dots, n-1$.

La fonction s peut également s'écrire :

$$(4.4) \quad s(t) = z_i + \lambda_i(t - t_i) + \frac{(m_i - \lambda_i)^2 (t - t_i)^2}{(\lambda_{i+1} - m_i)h_i - (\lambda_{i+1} - 2m_i + \lambda_i)(t - t_i)}$$

Proposition 4.1.

Etant donné les hypothèses :

- | | |
|--|------------------------|
| a) $m_i - m_{i-1} > 0$ ($m_i - m_{i-1} < 0$) | $i = 2, 3, \dots, n-1$ |
| b) $\lambda_i < m_i < \lambda_{i+1}$ ($\lambda_i > m_i > \lambda_{i+1}$) | $i = 1, 2, \dots, n-1$ |

la fonction s définie par (4.4.) possède les propriétés suivantes :

- (i) s n'a pas de pôles dans l'intervalle $[a, b]$
- (ii) $s(t_i) = z_i$, $s'(t_i) = \lambda_i$ $i = 1, 2, \dots, n$
- (iii) s est convexe (concave) sur $[a, b]$
- (iv) $s \in C^1 [a, b]$

Preuve

Par suite de la condition b) les valeurs $q_i(t_i) = (\lambda_{i+1} - m_i)h_i$ et $q_i(t_{i+1}) = (m_i - \lambda_i)h_i$ ont le même signe (positif si $\lambda_i < m_i < \lambda_{i+1}$, négatif si $\lambda_i > m_i > \lambda_{i+1}$). On obtient ainsi (i), car $q_i(t)$ est l'équation d'une droite.

En dérivant (4.1) il résulte

$$(4.5) \quad s'(t) = r_i(t) / [q_i(t)]^2 \quad \text{pour } t \in [t_i, t_{i+1}], \quad i = 1, 2, \dots, n-1 \quad \text{où}$$

$$(4.6) \quad r_i(t) = \lambda_i (\lambda_{i+1} - m_i)^2 h_i^2 + 2(\lambda_{i+1} - m_i)(m_i^2 - \lambda_i \lambda_{i+1})h_i(t - t_i) - (\lambda_{i+1} - 2m_i + \lambda_i)(m_i^2 - \lambda_i \lambda_{i+1})(t - t_i)^2$$

et, à son tour,

$$(4.7) \quad s''(t) = 2(\lambda_{i+1} - m_i)^2 (m_i - \lambda_i)^2 / [q_i(t)]^3$$

Moyennant un calcul simple on vérifie les conditions d'interpolation (ii). Alors (iv) est immédiat.

Le signe de $s''(t)$ pour $t \in [t_i, t_{i+1}]$ dépend exclusivement du signe de $q_i(t)$. D'après le raisonnement pour démontrer (i) on obtient que

$\lambda_i < m_i < \lambda_{i+1}$ ($\lambda_i > m_i > \lambda_{i+1}$) entraîne $q_i(t) > 0$ ($q_i(t) < 0$) et donc $s''(t) > 0$ ($s''(t) < 0$). \square

Pour les applications numériques, on calculera les dérivées λ_i ($i=1,2,\dots,n$) à l'aide des formules des trois points (D0) ; celles-ci garantissent qu'on ait b).

Remarque sur la co-convexité

Si les données changent de convexité autour d'un point, on cherche à ce que l'interpolante reflète ces variations, autrement dit qu'elle soit **co-convexe avec les données**.

Pour fixer les idées, supposons qu'on a $m_1 < m_2 < \dots < m_{r-1} > m_r > \dots > m_{n-1}$ c'est-à-dire que $D_1 = \{(t_i, z_i)\}$, $i=1,\dots,r$ et $D_2 = \{(t_i, z_i)\}$, $i=r+1,\dots,n$ sont respectivement des données convexes et concaves. On calcule séparément les dérivées de $d_1^{(1)}, d_2^{(1)}, \dots, d_r^{(1)}$ et $d_r^{(2)}, d_{r+1}^{(2)}, \dots, d_n^{(2)}$, pour avoir la condition b) sur chaque ensemble, et on prend :

$$(4.8) \quad \left\{ \begin{array}{ll} \lambda_i = d_i^{(1)} & i=1,2,\dots,r-1 \\ \lambda_r = \begin{cases} d_r^* & \text{si } d_r^* > m_{r-1} \\ d_r^{(1)} & \text{sinon} \end{cases} & \text{où } d_r^* = (d_r^{(1)} + d_r^{(2)}) / 2 \\ \lambda_i = d_i^{(2)} & i=r+1,\dots,n \end{array} \right.$$

Dans ce cas le point d'inflexion sera placé en (t_r, z_r) .

4.2. Interpolation rationnelle (2/1) convexe et monotone

Continuons à considérer les hypothèses a) et b) de la proposition (4.1) et supposons de plus que

$$c) \operatorname{sgn}(\lambda_i) = \operatorname{sgn}(\lambda_{i+1}) = \operatorname{sgn}(m_i) \quad i=1,2,\dots,n-1$$

$$\text{Notons } u_i = \lambda_i/m_i, \quad v_i = \lambda_{i+1}/m_i \quad i=1,2,\dots,n-1$$

Bornons-nous au cas typique des données strictement convexes et strictement croissantes. Il découle que $0 < u_i < 1 < v_i$

Nous allons examiner en détail le comportement de s' dans l'intervalle I_i . Nous omettons les indices de u_i et v_i .

cas 1. $uv-1=0$

Dans ce cas s est homographique et pourtant $s'(t) > 0$.

cas 2. $u+v-2=0$

Dans ce cas s est quadratique et s' linéaire. Puisque $\lambda_i \leq s'(t) \leq \lambda_{i+1}$, alors s est monotone croissante sur I_i .

cas 3. $uv-1 \neq 0$ et $u+v-2 \neq 0$

Les racines de $r_i(t)$ sont $t_i + \gamma_1 h_i$ et $t_i + \gamma_2 h_i$

où

$$\gamma_1 = (v-1) [(1-uv) + (1-u)\sqrt{1-uv}] / [(1-uv)(u+v+2)]$$

$$\gamma_2 = (v-1) [(1-uv) - (1-u)\sqrt{1-uv}] / [(1-uv)(u+v+2)]$$

On a $r_i(t_i) = (\lambda_{i+1} - m_i) h_i > 0$ et $r_i(t_{i+1}) = (m_i - \lambda_i) h_i > 0$.

Il est clair que $r_i(t) > 0$ sur I_i (donc $s'(t) > 0$) si et seulement si γ_1 et γ_2 n'appartiennent pas à l'intervalle $[0,1]$. Pour $1-uv < 0$ c'est évident. Soit $1-uv > 0$; après quelques calculs on établit que :

$$\gamma_1 > 1 \quad \text{et} \quad \gamma_2 < 0 \quad \text{si} \quad u+v-2 > 0 \quad \text{et}$$

$$\gamma_1 < 0 \quad \text{et} \quad \gamma_2 < 0 \quad \text{si} \quad u+v-2 < 0.$$

Les résultats ci-dessus peuvent être résumés sous la forme de la

Proposition 4.2.

En faisant les hypothèses a), b) et c) la fonction s(4.4) en plus de vérifier les conclusions (i), (ii), (iii) et (iv) de la proposition 4.1, elle est monotone sur [a,b]. \square

De la même manière que nous avons procédé pour la construction de la FSH monotone et convexe, nous utiliserons les schémas (D1) et (D2) pour calculer les dérivées de λ_j . \square

Maintenant, nous pouvons faire le point sur les interpolantes convexes et monotones quadratiques, homographiques et rationnelles (2/1).

Examinons les cas $0 < m_1 < m_2 < \dots < m_{n-1}$. Nous venons de voir que pour s(4.4), $s'(t) > 0$ et $s''(t) > 0$ pour $t \in [t_j, t_{j+1}]$ si, et seulement si, le point $(u_j, v_j) = (\lambda_j/m_j, \lambda_{j+1}/m_j)$ appartient à la région "croissante et convexe" $\mathcal{R} =]0, 1[\times]1, \infty[$ (représentée dans les figures 1(a) et 1(b)).

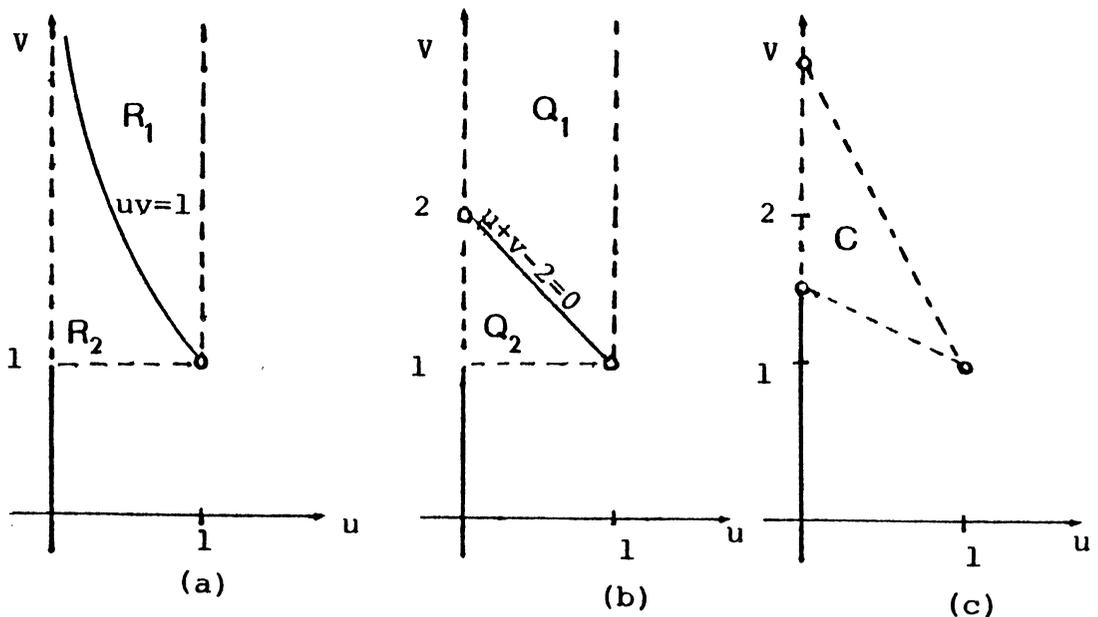


Fig.1.

D'autre part, si nous utilisons une représentation rationnelle linéaire pour s , on doit insérer un point ξ_j dans l'intervalle $]t_j, t_{j+1}[$, selon les modalités suivantes :

- si $(u_j, v_j) \in \mathfrak{R}_1 = \{ (u, v) \in \mathfrak{R} / uv > 1 \}$
prendre $\xi_j \in]\underline{\xi}_j, t_{j+1}[$ où $\underline{\xi}_j = t_{j+1} + (t_{j+1} - t_j)(u_j - 1) / \sqrt{u_j v_j} - u_j$
- si $(u_j, v_j) \in \mathfrak{R}_2 = \{ (u, v) \in \mathfrak{R} / uv < 1 \}$
prendre $\xi_j \in]t_j, \bar{\xi}_j]$ où $\bar{\xi}_j = t_j + (t_{j+1} - t_j)(v_j - 1) / (v_j - \sqrt{u_j v_j})$

Par contre, si $u_j v_j = 1$ il n'y a pas besoin d'ajouter un point entre les données.

De façon analogue, si s est une quadratique (voir [102]) et $u_j + v_j - 2 = 0$ on n'ajoute pas de points supplémentaires, sinon

- pour $(u_j, v_j) \in \mathfrak{Q}_1 = \{ (u, v) \in \mathfrak{R} / u + v - 2 > 0 \}$
prendre $\xi_j \in]\underline{\xi}_j, t_{j+1}]$ où $\underline{\xi}_j = t_{j+1} + 2(t_{j+1} - t_j)(u_j - 1) / (v_j - u_j)$
- pour $(u_j, v_j) \in \mathfrak{Q}_2 = \{ (u, v) \in \mathfrak{R} / u + v - 2 < 0 \}$
prendre $\xi_j \in]t_j, \bar{\xi}_j]$ où $\bar{\xi}_j = t_j + 2(t_{j+1} - t_j)(u_j - 1) / (v_j - u_j)$

En résumé, la région \mathfrak{R} nous sert pour trouver les (u_j, v_j) et donc les $(\lambda_j, \lambda_{j+1})$ pour construire des fonctions monotones et convexes quadratiques, homographiques et rationnelles (2/1). Ces dernières ont l'avantage de ne pas avoir besoin d'insertion de points.

A titre d'illustration, la Fig. 1(c) représente la région \mathfrak{E} monotone et convexe lorsque s est une cubique d'Hermite, où $\mathfrak{E} = \{ (u, v) \in \mathfrak{R} / 2u + v < 3 < u + 2v \}$. Il est plus simple de chercher (u_j, v_j) appartenant à la région \mathfrak{R} qu'à \mathfrak{E} . En fait, nous l'avons déjà dit, une combinaison convexe de m_j et m_{j-1} pour λ_j suffit dans le premier cas.

4.3. Une spline rationnelle (2/1) convexe

Comme la convexité peut s'exprimer en termes de la dérivée seconde, il paraît naturel de demander qu'une interpolante qui conserve cette propriété ait la continuité C^2 . Les fonctions rationnelles (2/1) sont capables de satisfaire cette demande.

Au lieu de considérer la représentation (4.4) pour s , nous introduisons $\beta_i = s''(t_i)$ ($i=1, \dots, n$) comme paramètres pour le calcul des coefficients de la fonction.

Il est toujours possible d'écrire $\beta_i = M_i^3$.

Définissons s de la manière suivante pour $t \in [t_i, t_{i+1}]$

$$(4.9) \quad s(t) = z_i + \frac{(m_i - 1) M_i^2 M_{i+1} h_i}{2} (t - t_i) + \frac{M_i^3 M_{i+1} h_i (t - t_i)^2}{2[M_{i+1} h_i + (M_i - M_{i+1})(t - t_i)]}$$

On remarque que le dénominateur $L_i(t) = M_{i+1} h_i + (M_i - M_{i+1})(t - t_i) \neq 0$ si et seulement si $L_i(t_i) \cdot L_i(t_{i+1}) = M_{i+1} M_i h_i^2 > 0$.

En dérivant s deux fois on obtient

$$(4.10) \quad s''(t) = m_i^3 M_{i+1}^3 h_i / [L_i(t)]^3$$

Une relation non-linéaire qui lie M_{i-1} , M_i et M_{i+1} existe dans chaque noeud intérieur t_i , si l'on impose que $s'(t_i^-) = s'(t_i^+)$. Plus précisément,

$$s'(t_i^-) = m_{i-1} + (M_{i-1} M_i^2 h_{i-1})/2$$

$$s'(t_i^+) = m_i - (M_i^2 M_{i+1} h_i)/2$$

donc

$$(4.11) \quad M_{i-1} M_i^2 h_{i-1} + M_i^2 M_{i+1} h_i = 2(m_i - m_{i-1}) \quad i=2,3,\dots,n-1$$

Proposition 4.3

$$\begin{array}{ll} \text{Soient } m_i - m_{i-1} > 0 & (m_i - m_{i-1} < 0) & i=1,2,\dots,n-1 \\ M_i > 0 & (M_i < 0) & i=1,2,\dots,n \end{array}$$

et tels que l'on a les relations (4.11). Alors la fonction s définie par (4.9) possède les propriétés suivantes :

- (i) s n'a pas de pôles dans l'intervalle $[a,b]$
- (ii) $s(t_i) = z_i$, $s''(t_i) = M_i^3$ $i=1,2,\dots,n$
- (iii) s est convexe (concave) sur $[a,b]$
- (iv) $s \in C^2[a,b]$

Preuve

La preuve est immédiate d'après ce qui vient d'être observé plus haut. \square

Les M_i seront calculés par un algorithme de type itératif.

Algorithme A3

1. $k:=0$

. Donner $M_i^{(0)}$ tel que $\text{sgn}(M_i^{(0)}) = \theta$ $i=1,2,\dots,n$

où $\theta=1$ ($\theta=-1$) si les données sont convexes (concaves)

. Donner la tolérance $\epsilon > 0$

2. (Itération k -ième)

. $k:=k+1$;

$$M_1^{(k)} = M_1^{(k-1)} = M_1^{(0)}$$

$$M_i^{(k)} = \theta \left(\frac{2(m_i - m_{i-1})}{h_{i-1} M_{i-1}^{(k)} + h_i M_{i+1}^{(k-1)}} \right)^{1/2} \quad i=2,\dots,n-1$$

$$M_n^{(k)} = M_n^{(k-1)} = M_n^{(0)}$$

3. Si $\max_{2 \leq i \leq n-1} |M_i^{(k)} - M_i^{(k-1)}| < \epsilon$ alors stop

sinon aller à 2. \square

Cette itération type Gauss-Seidel est convergente comme le démontre Schabak [100].

Nous proposons d'utiliser comme point de démarrage de l'algorithme l'approximation de β_j qui provient de la "formule de 3 points" (cf. § 1.2). On peut penser, pour trouver les conditions de bout, faire passer une rationnelle (2/1) par les 4 premières (dernières) données et prendre la dérivée seconde en $t_1(t_n)$ comme une valeur de $\beta_1 (\beta_n)$.

11.5. ESSAIS NUMERIQUES

Nous avons traité plusieurs exemples numériques pour tester et comparer les différentes méthodes décrites dans les sections précédentes.

Nous avons considéré deux ensembles de données provenant des papiers de Pruess [93], Fritsch et Carlson [41], et les fonctions

$$f_1(t) = (-9t+2) / (4t+5) \quad \text{pour } t \in [-1,8]$$

$$f_2(t) = (9t+2) e^{-0.25t} \quad \text{pour } t \in [0,15]$$

en n points équidistants.

t_j	22	22.5	22.6	22.7	22.8	22.9	23	23.1	23.2	23.3	23.4	23.5	24
z_j	523	543	550	557	565	575	590	620	860	915	944	958	986

Tableau 1. (Pruess)

t_j	7.99	8.09	8.19	8.7	9.2	10
z_j	0	2.76429 E-5	4.37498 E-2	0.169183	0.469428	0.943740
t_j	12	15	20			
z_j	0.998636	0.999919	0.999994			

Tableau 2. (Fritsch-Carlson)

Les critères retenus pour comparer les méthodes sont principalement deux : le temps d'exécution et l'aspect "visuel".

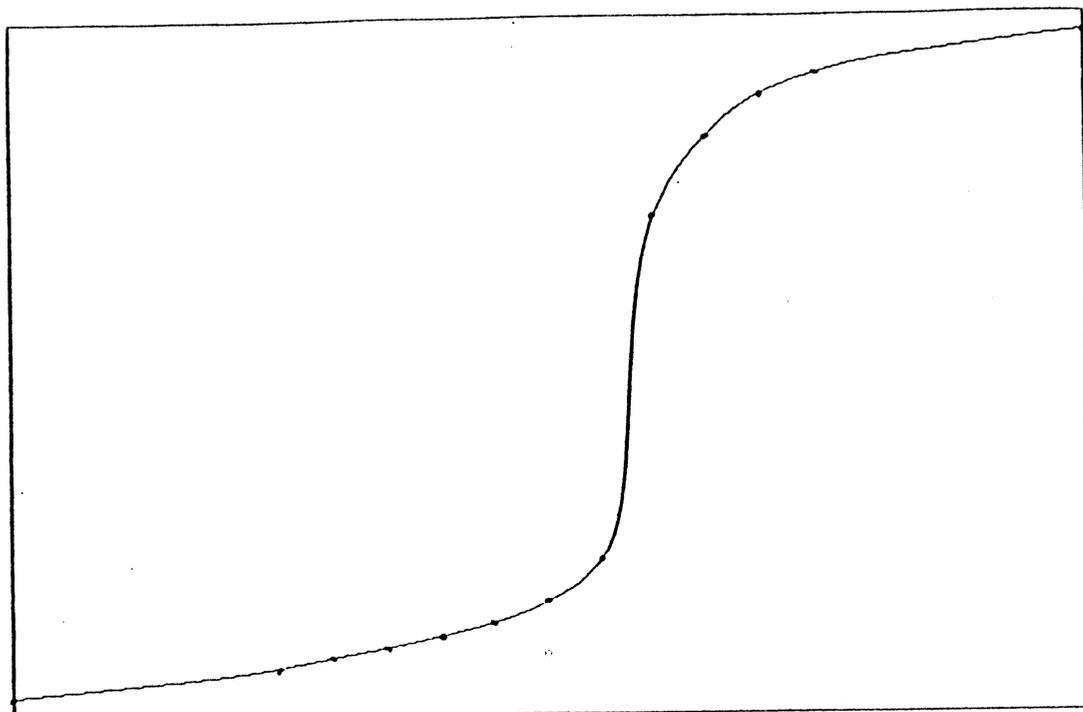
Sur l'aspect "visuel"

● La figure 2 montre les profils monotones pour les données du tableau 1 obtenus par l'algorithme A1 et les dérivées calculées par (D2), pour différents choix des noeuds supplémentaires.

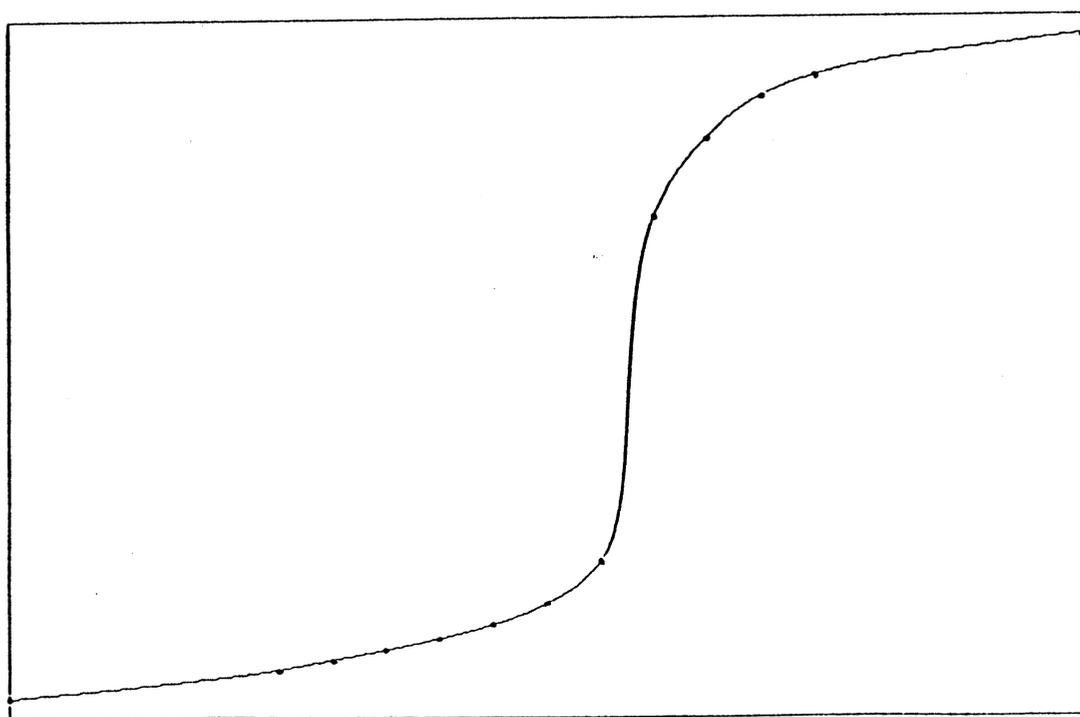
Pour ce même problème, nous avons sélectionné la représentation rationnelle quadratique "plus plaisante" visuellement, celle dont les dérivées sont données par le schéma (D2). (voir Fig. 3(a)).

Au lieu d'imposer la contrainte de monotonie, nous avons exigé la co-convexité de la courbe avec les données 1. La fig. 3(b) contient le graphe de la fonction rationnelle (2/1) par morceaux ; les paramètres λ_j sont fournis aussi par (D2).

Nous allons porter un regard attentif sur l'intervalle $[t_7, t_{10}] = [23, 23.3]$ et plus particulièrement sur $[t_8, t_9]$. On remarque que les courbes de la fig. 2 se ressemblent et sont plus "douces" que celles de la fig. 3. En effet, sur $[t_8, t_9]$ il y a un changement de convexité de l'interpolante, et d'après l'algorithme A1 le point d'inflexion est placé en $(t_8 + t_9)/2$. Le graphe 3(b) est le plus proche de la droite qui relie (t_8, z_8) et (t_9, z_9) ; dans ce cas-là le passage de la convexité à la concavité se réalise dans la borne t_9 et non à l'intérieur de l'intervalle, selon les formules (4.8).

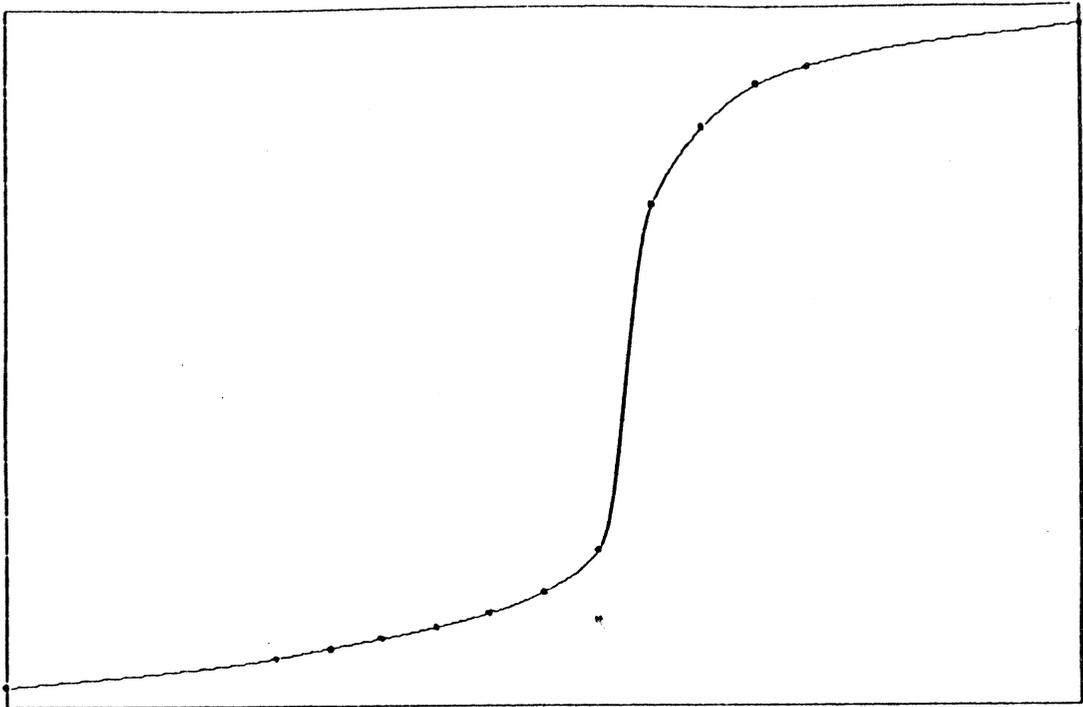


(a)

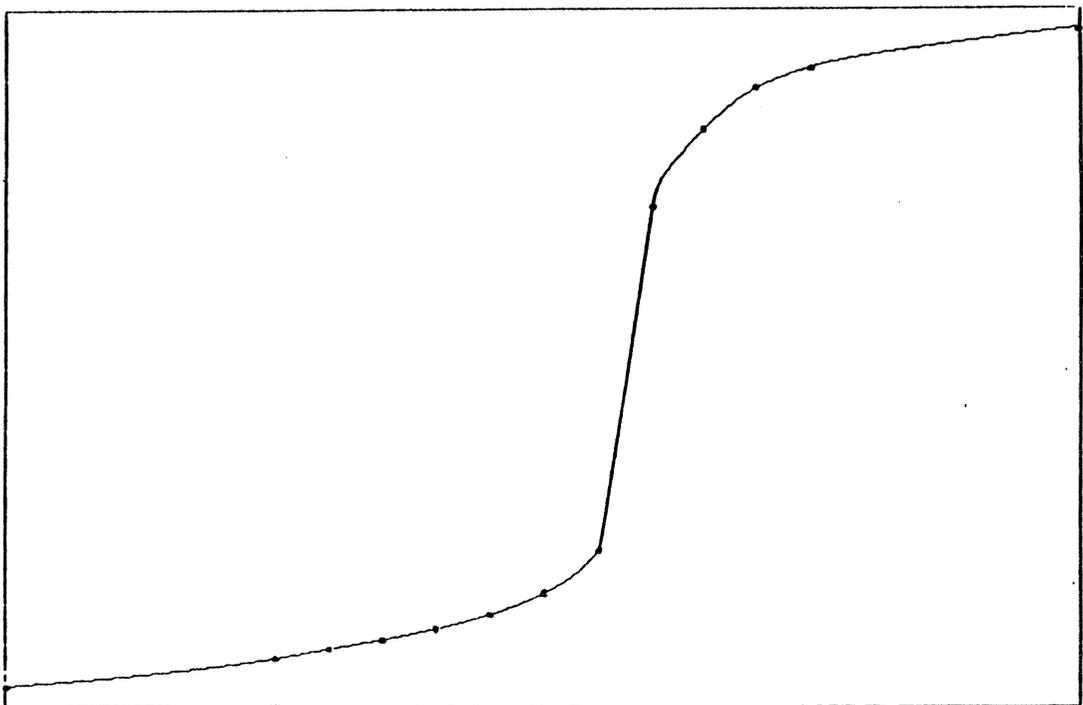


(b)

Fig. 2. F.R(1/1) pour les données monotones 1, avec noeuds insérés.
(a) au milieu des intervalles.
(b) aux bornes des intervalles.



(a)



(b)

Fig. 3. Interpolation des données monotones 1.

(a) F.R.(2/2) - C^1

(b) F.R.(2/1) - C^1 (co-convexe)

• Lorsqu'on doit insérer des points pour les FSH, leur localisation peut donner des solutions différentes. Nous illustrons cette situation pour le jeu de données 2. Les variations plus importantes se produisent entre $t_6=10$ et $t_7=12$.

Les petites fenêtres de la figure 4 sont des agrandissements pour $t \in [9.5,12]$. L'intervalle dans lequel on doit placer le noeud supplémentaire ξ_6 est $]10,10.3176[$. Le dessin en haut correspond à la courbe dont ξ_6 est pris au milieu de l'intervalle, tandis qu'en bas on a placé ξ_6 dans la borne supérieur 10.3176.

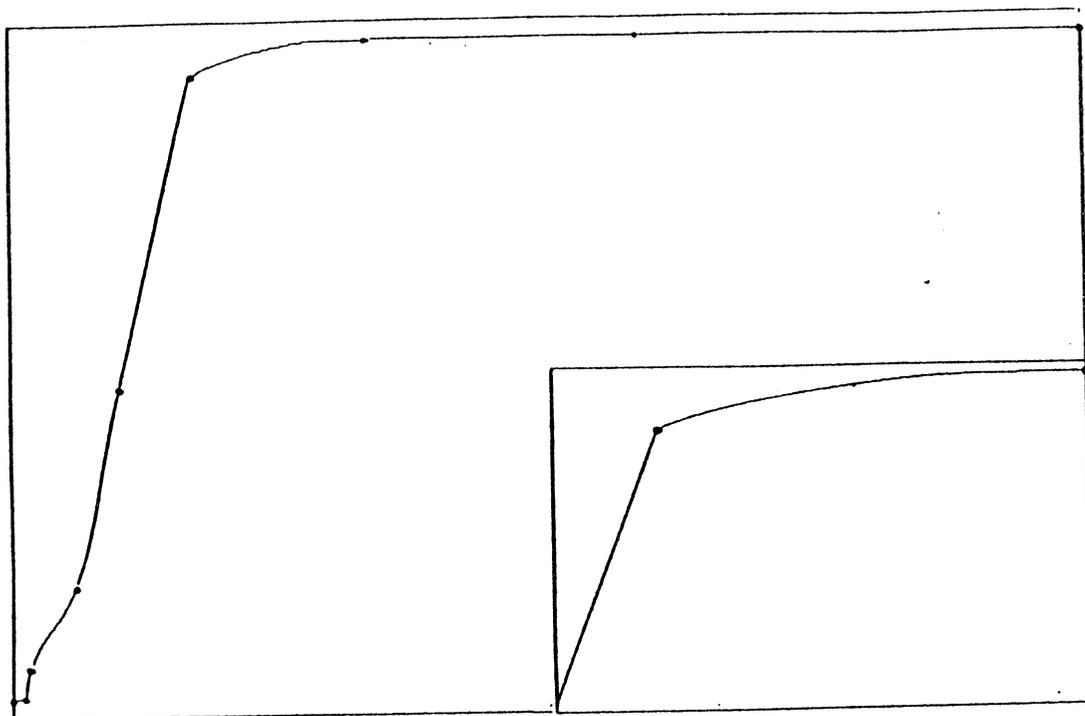
• Pour pouvoir faire une comparaison de toutes les techniques, il faut partir des exemples où les données sont à la fois monotones et convexes (concaves).

Considérons la fonction décroissante et convexe $f_1(t)$ pour 10 points équidistants dans l'intervalle $[-1,8]$. Nous rangeons ces valeurs dans le tableau 3 pour aider à la compréhension des résultats.

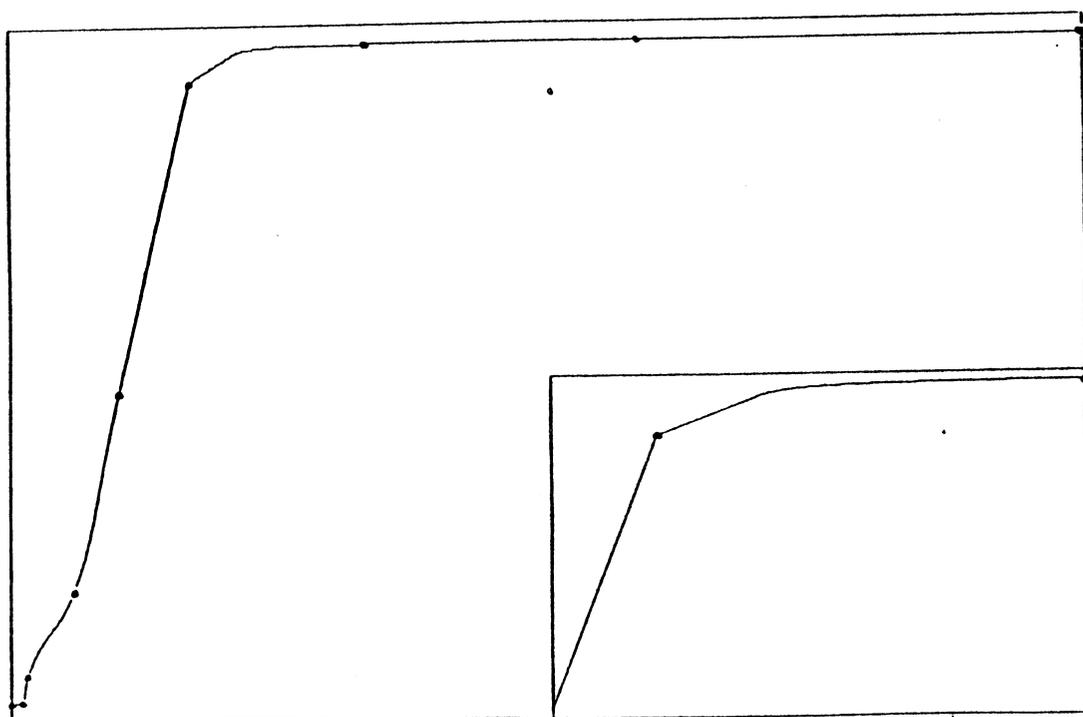
t_j	-1	0	1	2	3	4	5
z_j	11	0.4	-0.77777	-1.23077	1.47059	-1.6190	-1.72
t_j	6	7	8				
z_j	-1.79310	-1.84848	-1.89189				

Tableau 3.

Si l'on cherche une représentation ayant la continuité C^1 , pour laquelle les dérivées aux noeuds sont calculées par le schéma (D1), les courbes (a) des Figs. 5,6 et 7 sont respectivement homographique, rationnelle quadratique et rationnelle (2/1). Dans le deuxième cas, c'est la condition de monotonie qui est considérée, tandis que dans les cas restants on utilise aussi la convexité des données.

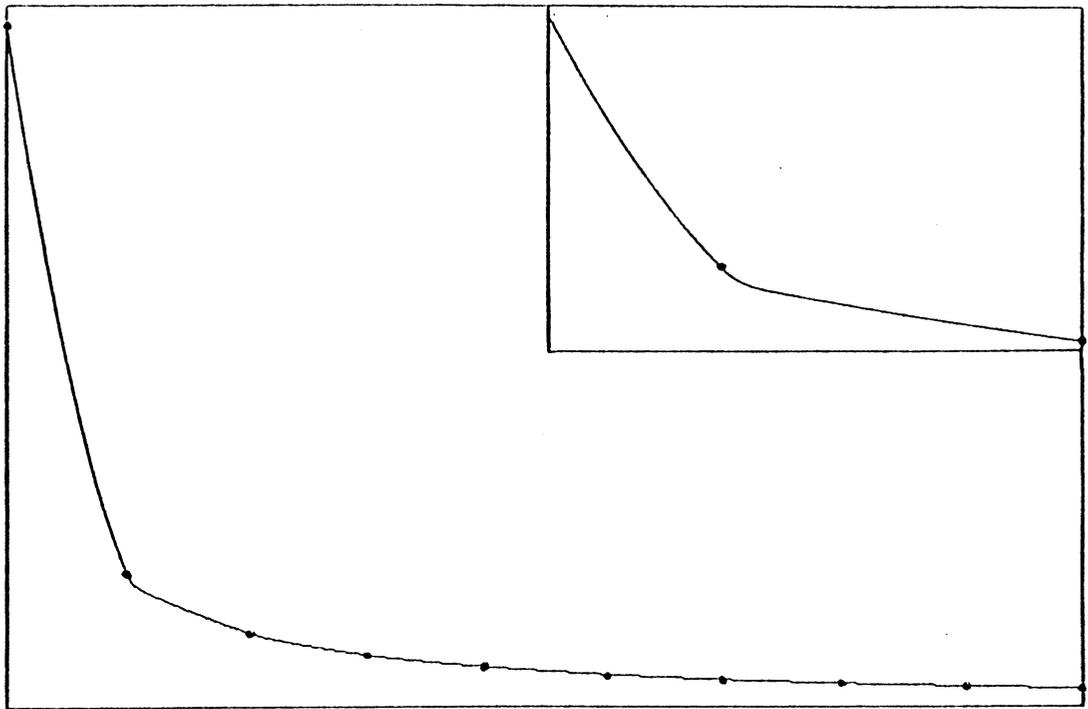


(a)

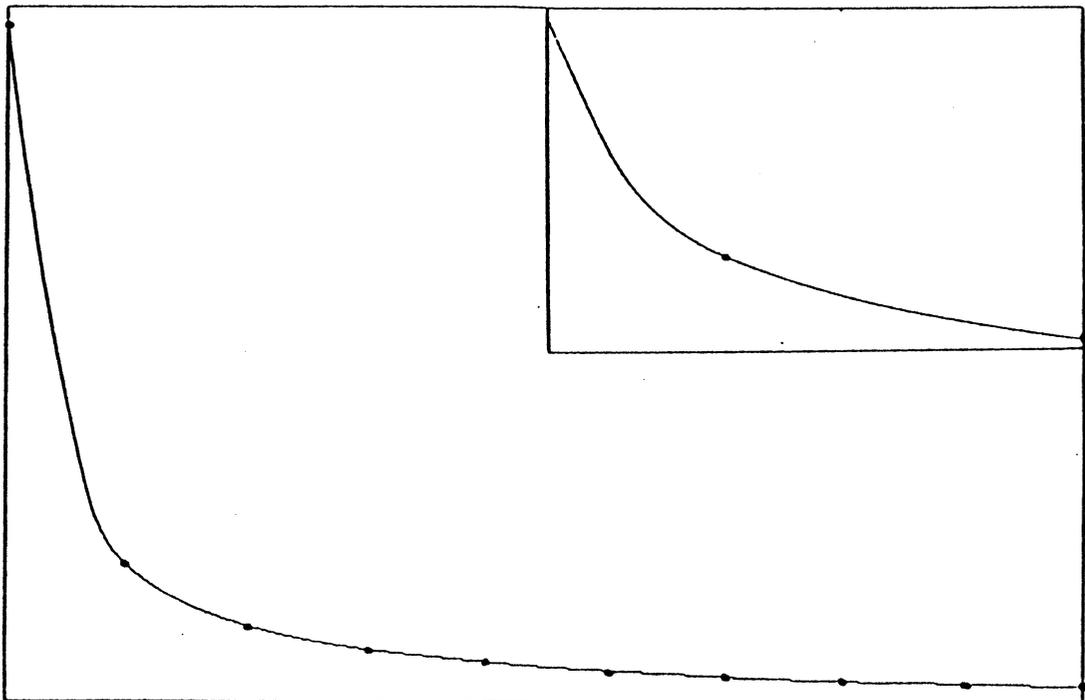


(b)

Fig.4. F.R.(1/1) pour les données monotones 2, avec noeuds insérés
a) au milieu des intervalles
b) aux bornes des intervalles



(a) .

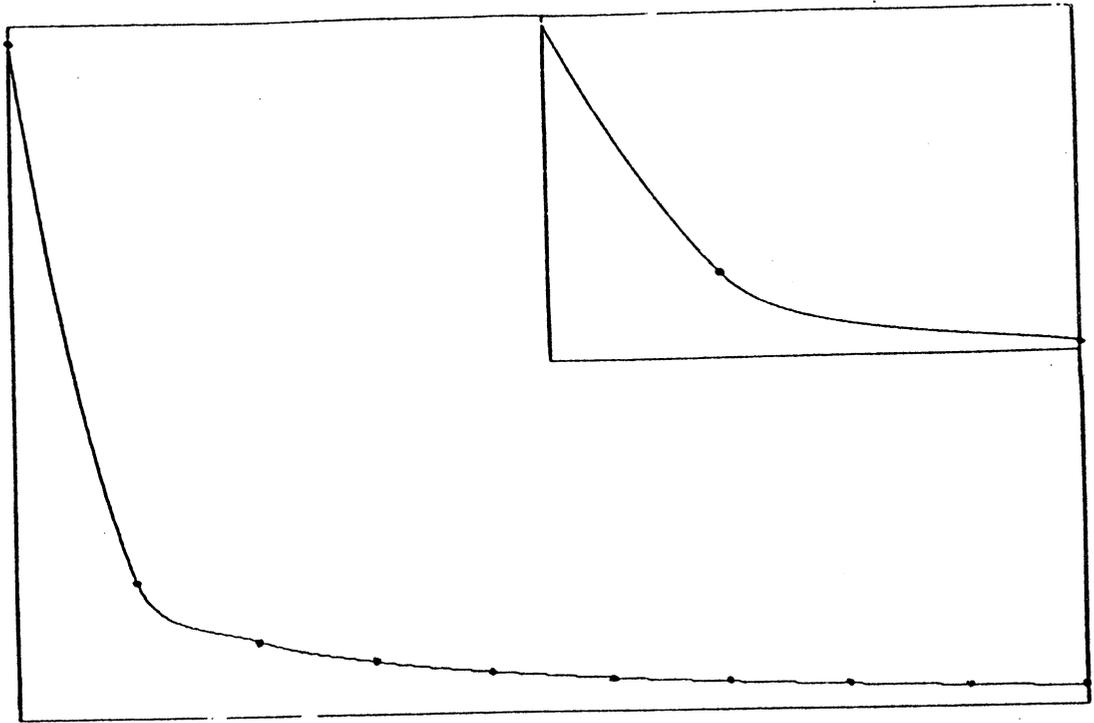


(b)

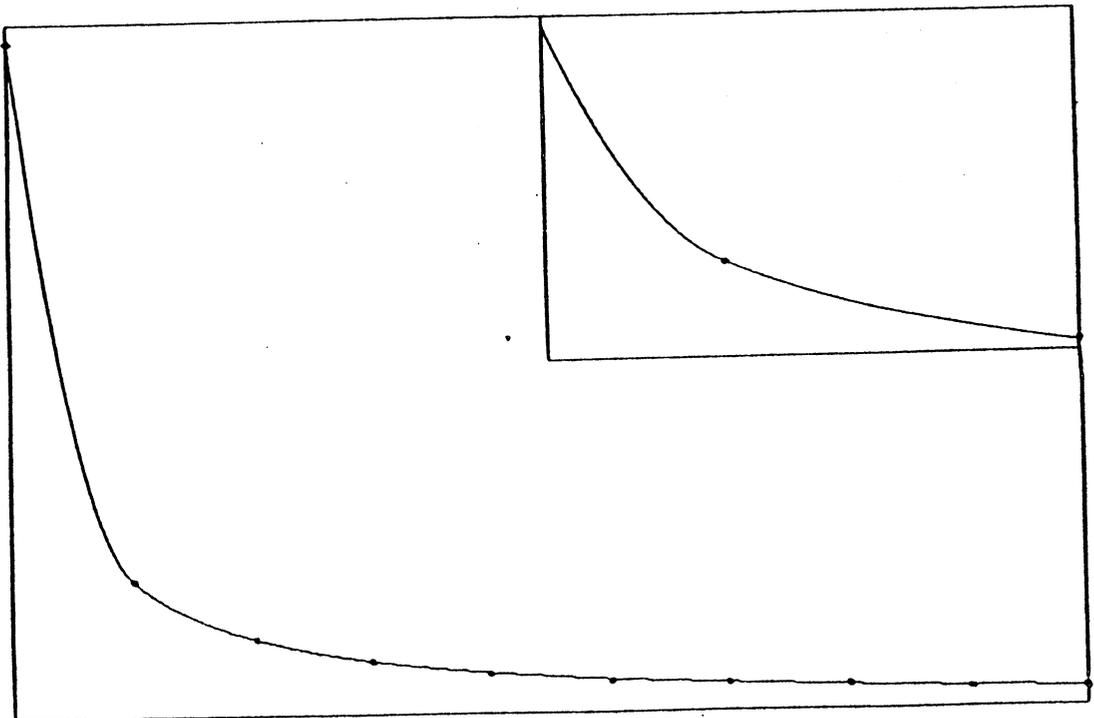
Fig.5. F.R. (1/1) pour les données monotones et convexes 3.

(a) méthode (D1)

(b) méthode (D2)

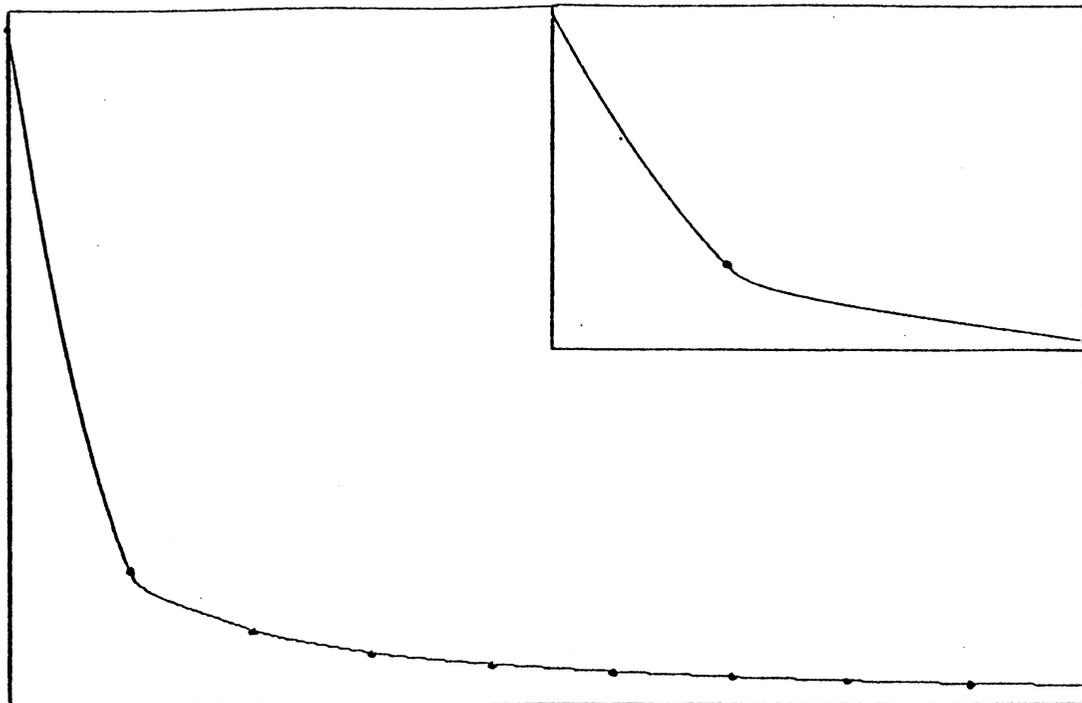


(a)

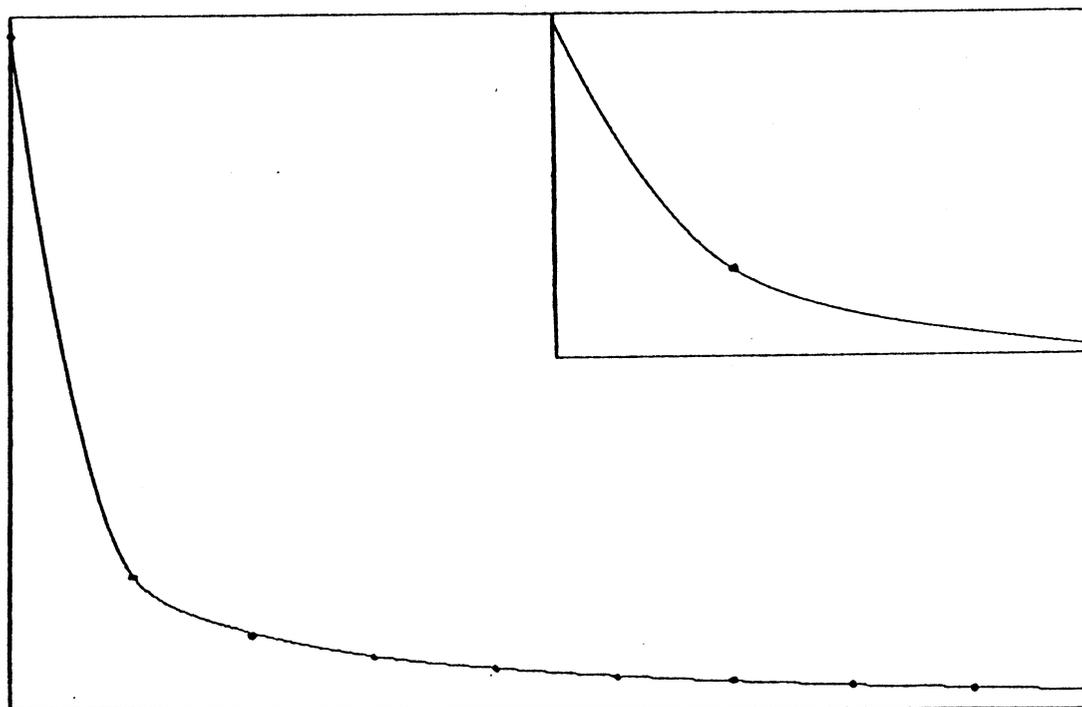


(b)

Fig. 6 F.R. (2/2) pour les données monotones 3.
(a) C^1 -méthode (D1)
(b) C^2 .



(a)



(b)

Fig. 7. F.R. (2/1) pour les données convexes et monotones 3.

(a) C^1 -méthode (D1)

(b) C^2 .

Evidemment , les changements plus importants se trouvent autour du point $(t_2, z_2) = (0, 0.4)$, dans la zone où la décroissance de la fonction est plus rapide.

Si on applique la méthode (D2) au lieu de (D1), pour la fonction rationnelle linéaire, une amélioration non négligeable se produit du point de vue graphique (voir fig. 5(b)).

On arrive à de bons résultats si la courbure est aussi continue, comme on peut le voir dans la figure 6(b) et 7(b).

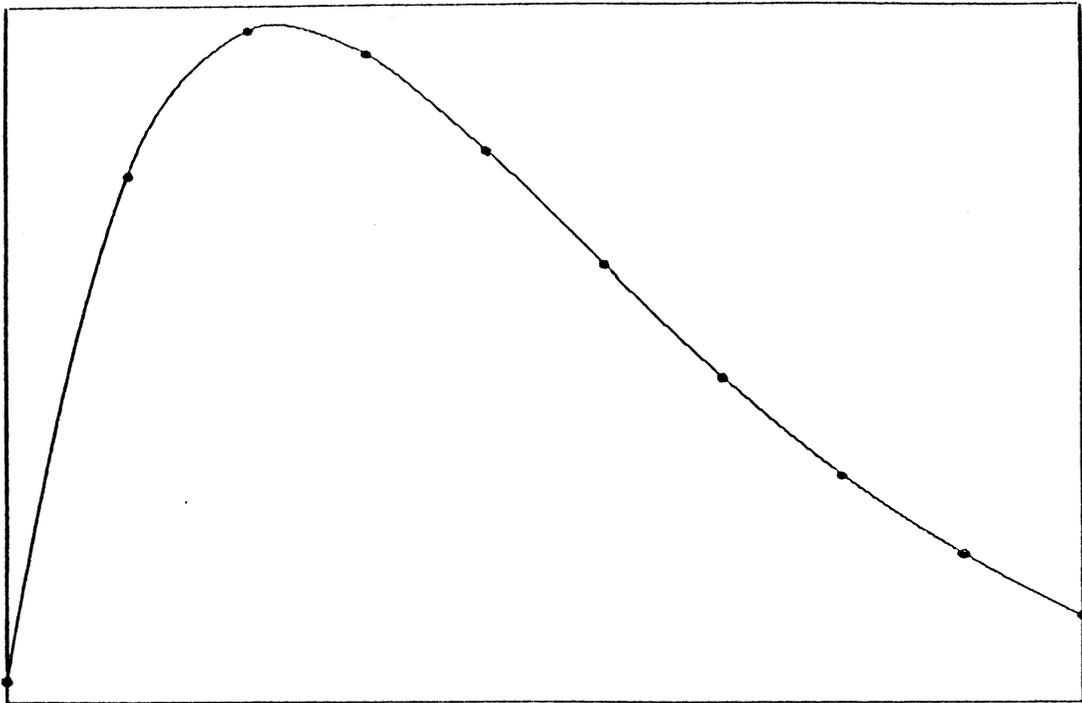
L'agrandissement des graphes porte sur les abscisses variant dans l'intervalle $[-0.5, 1]$.

D'une manière générale, on peut affirmer que pour les interpolantes rationnelles monotones de classe C^1 , les dérivées calculées par les formules (D2) donnent des résultats plus satisfaisants.

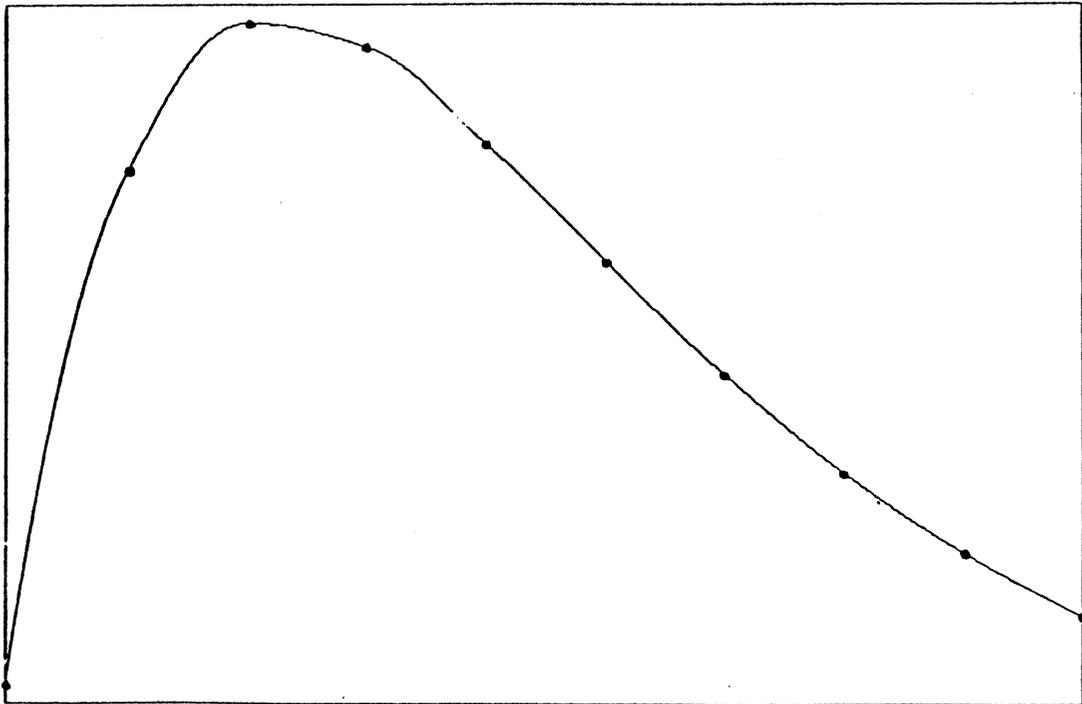
- La fonction f_2 est concave sur l'intervalle $[0, 70/9]$ et convexe sur $[70/9, 15]$. Elle est croissante pour $t \in [0, 34/9]$ et décroissante pour $t \in [34/9, 15]$.

Prenant $n = 10$, nous avons essayé de refléter la co-monotonie avec les fonctions rationnelles (2/2) et la co-convexité avec les fonctions rationnelles (2/1). Les dérivées λ_1 ont été données respectivement par les approximations (D1) et (D0).

La différence entre les deux courbes (fig.8) est plus marquée sur l'intervalle $[t_2, t_5] = [0, 8/3]$. En effet, le passage de la région croissante à la zone décroissante se fait en imposant l'existence d'un maximum relatif (ici global) égal à $f_2(t_3)$. La courbe co-convexe (en haut) donne un résultat plus raisonnable.



(a)



(b)

Fig. 8. Interpolation de $f_2(t)$, $t \in [0, 15]$, $n=10$.

(a) F.R. (2/1) - C^1 , co-convexe.

(b) F.R. (2/2) - C^1 , co-monotone

•Comparaison du temps de calcul

Nous avons choisi des abscisses équidistantes pour la fonction $f_1(t)$, $-1 \leq t \leq 8$, avec $n = 50, 100, 200$ afin de regarder l'évolution du temps de calcul des méthodes.

n	dérivées	méthode	temps(sec.)	points ajoutés	itérations
50	λ_j	D1	0.10	-	-
		D2	0.10	-	-
		A1			
	β_j	D1	0.70	49	-
		D2	0.66	43	-
		A2	1.44	-	10
		A3	1.38	-	7
100	λ_j	D1	0.20	-	-
		D2	0.22	-	-
		A1			
	β_j	D1	1.40	99	-
		D2	1.36	97	-
		A2	2.96	-	10
		A3	2.84	-	7
200	λ_j	D1	0.40	-	-
		D2	0.44	-	-
		A1			
	β_j	D1	2.78	199	-
		D2	2.72	197	-
		A2	5.88	-	10
		A3	5.70	-	7

Tableau 4.

Nous constatons que :

- les schémas (D1) et (D2) sont les plus rapides.
- l'itération Gauss-Seidel de l'algorithme A3 est plus rapide que celle de A2. Le nombre d'itérations pour atteindre le seuil $\epsilon=10^{-5}$ est respectivement 7 et 10 , et reste stable même si n croît.
- l'algorithme A1 a deux pas, le premier : l'application des formules (D1) ou (D2), le second : l'élargissement de l'ensemble des données et des dérivées. La deuxième partie est la plus coûteuse ; mais le temps total du calcul de cet algorithme représente à peu près la moitié du temps de A3.

Si l'on tient compte du nombre de points ajoutés, on remarque qu'il est légèrement plus petit lorsque A1 est utilisé avec (D2).

On peut dire qu'un comportement semblable a été observé pour les autres exemples abordés.

• Sur l'encombrement

Une question importante pour les méthodes numériques est la taille de la place mémoire utilisée.

On a toujours besoin de 3 tableaux unidimensionnels $t[i]$, $z[i]$, $dy[i]$ $i=1,2,\dots,n$ pour les abscisses, les ordonnées et les dérivées aux noeuds. Le nombre de tableaux additionnels pour les différentes méthodes est :

- . 0 pour (D0) , (D1) et (D2) .
- . 3 pour A1: $ts[i]$, $zs[i]$, $dys[i]$ $i=1,2,\dots,2n-1$
- . 3 pour A2: $a[i]$, $b[i]$, $c[i]$ $i=1,2,\dots,n$
- . 2 pour A3: $m[i]$, $h[i]$ $i=1,2,\dots,n-1$

Il est clair que l'algorithme A1 demande plus de mémoire, même sans tenir compte des variables simples.

II.6. CONCLUSIONS

Nous avons proposé des méthodes pour l'interpolation C^1 monotone et convexe en utilisant les fonctions homographiques et rationnelles (2/1) par morceaux.

- Pour préserver la monotonie des données on doit se servir des splines homographiques. Le schéma (D2) est le plus approprié pour le calcul des dérivées aux noeuds.

- Si l'on veut garder la convexité, les fonctions rationnelles (2/1) par morceaux donnent des résultats satisfaisants si les dérivées sont fournies par les formules (D0).

- Si les données sont en même temps convexes et monotones, les fonctions rationnelles (2/1) semblent mieux placées pour refléter ces propriétés de forme lorsque les dérivées sont calculées par (D2). Ces fonctions, à la différence des splines homographiques, ont l'avantage de ne pas insérer de points supplémentaires.

CHAPITRE III

Fonctions Splines Pénalisées et sous Tension

III.1. INTRODUCTION

Plusieurs algorithmes pour l'interpolation monotone et convexe ont été expérimentés récemment (voir chap. I, II), mais leur extension au lissage de données entachées d'erreurs s'avère difficile du fait qu'ils n'assurent aucune propriété de minimisation.

Ceci nous a amené à étudier la classe de fonctions splines appelées "sous-tension", qui en quelque sorte sont des splines intermédiaires entre celles d'ordre 3 (cubiques) et celles d'ordre 1 (linéaires). Une approche variationnelle permet de traiter l'interpolation et aussi l'ajustement. Un choix convenable des paramètres de tension peut contraindre la courbe à satisfaire les conditions auxiliaires sur la monotonie ou la convexité.

C'est Schweikert [104] qui a introduit le concept de tension pour éliminer des points d'inflexion artificiels produits par la spline cubique d'interpolation. Le résultat est une fonction qui dans chaque intervalle s'exprime comme une combinaison linéaire de $\{1, t, \text{sh}(rt), \text{ch}(rt)\}$ où r est la tension, et qui se raccorde aux noeuds ainsi que ses dérivées jusqu'à l'ordre deux.

Plus tard, Späth [106], Cline [20], Dube [32], Baum [7], Pruess [91], Rentrop [97] ont étudié leurs propriétés ou écrit des procédures pour l'interpolation. Pruess [92] a considéré le cas du lissage.

L'idée de base est liée à la notion physique suivante : les fonctions spline cubiques d'interpolation minimisent l'énergie de flexion d'une tige flexible qui passe par des points donnés; en introduisant alors une force longitudinale on peut tirer la tige jusqu'à la disparition des tortillements.

Nous consacrons le deuxième paragraphe de ce chapitre aux fonctions spline exponentielles ou hyperboliques sous tension.

L'inconvénient majeur de ces fonctions est l'évaluation coûteuse de l'exponentielle. Des alternatives ont été cherchées; ainsi Späth [107] a défini une fonction avec des propriétés analogues à celles de la spline hyperbolique, dont la représentation est donnée par morceaux de fonctions rationnelles.

Nielson [81] a proposé une spline cubique de classe C^1 , qui, par contre, a la tangente et la courbure continues lorsque la courbe s'exprime sous la forme paramétrée. Pour sa part, de Boor [24] a construit aussi une spline cubique pour préserver la convexité mais en insérant des points supplémentaires. Une approche générale, englobant ces différentes alternatives a été présentée par Pruess dans [93]. Ces fonctions sont applicables pour l'interpolation et, sauf la cubique de Nielson, ne proviennent pas de la minimisation d'une fonctionnelle.

Cinquin [19] et Salkaukas [99] ont obtenu des fonctions dont l'intégrale du carré de la dérivée seconde affecté par des poids est la plus petite possible. On retrouve encore de cubiques de classe C^1 qui ont été utilisées pour des données monotones ou qui varient rapidement.

En dessin graphique assisté par ordinateur on considère la représentation paramétrée de la courbe. Dube [33] a donné des bases locales pour construire une spline cubique paramétrée d'interpolation, dont la forme peut être modifiée à l'aide de paramètres de tension. Beatty et Barsky [6] ajoutent dans la définition de leurs courbes deux paramètres supplémentaires, appelés biais et tension, pour contrôler la continuité des vecteurs tangent et courbure et donc contrôler la forme des fonctions.

Un tour d'horizon de ces fonctions est fait dans les troisième, quatrième et cinquième parties du chapitre. En particulier, nous avons étendu la spline de Nielson au cas du lissage.

Pour essayer d'avoir des splines cubiques avec un comportement semblable à celui des splines sous tension, nous proposons une pénalisation sur la dérivée première aux noeuds. Bien entendu, on perd la continuité C^2 . Nous développons ce sujet dans le sixième paragraphe de ce chapitre.

Différents moyens de choisir les paramètres de tension et de lissage sont discutés dans les paragraphes suivants. On termine avec la présentation des expériences numériques pour illustrer les méthodes étudiées.

III.2. FONCTIONS SPLINES HYPERBOLIQUES SOUS TENSION

2.1. Définitions et notations

Considérons $\Delta: a=t_0 < t_1 < \dots < t_n < t_{n+1}=b$ une partition de l'intervalle réel $[a,b]$ et soit r une fonction en escalier $r: [a,b] \rightarrow \mathbb{R}$ telle que $r(t) = r_i \quad \forall t \in]t_i, t_{i+1}[\quad i=0,1,\dots,n$ avec $r_0 = r_n = 0$.

On notera

$H^2 = H^2[a,b] = \{f: [a,b] \rightarrow \mathbb{R} / f, f' \text{ absolument continues, } f'' \in L^2[a,b]\}$,
et L l'opérateur différentiel défini par $L = D^2 - r(t)D$.

L'ensemble S_H des fonctions s telles que

- (i) $L^*L s = 0$ sur chaque intervalle $]t_i, t_{i+1}[\quad i=1,2,\dots,n-1$
- (ii) $Ls = 0$ sur $[a, t_1[$ et $]t_n, b]$
- (iii) $s \in C^2[a,b]$

sera appelé **l'ensemble des fonctions splines hyperboliques ou exponentielles sous tension (F.S.H.T.)**.

Remarques

1) Comme $L^*L[s(t)] = (D^4 - r_i^2 D^2)[s(t)]$ alors (i) est équivalente à

- (\bar{i}) s s'exprime comme combinaison linéaire de $\{1, t, e^{r_i t}, e^{-r_i t}\}$ ou de $\{1, t, \operatorname{sh} r_i t, \operatorname{ch} r_i t\}$ dans chacun des intervalles $]t_i, t_{i+1}[$ $i=1,2,\dots,n-1$;

et la condition (ii) est équivalente à

- (\bar{ii}) s est un polynôme de degré 1 dans chacun des intervalles $[a, t_1[,]t_n, b]$

2) Les paramètres de tension r_i sont pris positifs, car si $r_i < 0$ on a encore $L^*L = D^4 - r^2(t)D^2$.

Dans le travail original de Scheweikert [104] la tension est considérée constante (uniforme) sur tout l'intervalle $[t_1, t_n]$, mais pour les applications pratiques il est convenable de faire varier la tension sur chaque segment de la fonction.

3) Rappelons que le choix des conditions de bout des fonctions splines "naturelles" correspond à une approche variationnelle.

On peut vérifier facilement que pour $n > 2$, S_H est un sous-espace vectoriel de dimension n de H^2 .

2.2. Interpolation

Nous allons étudier d'abord les fonctions splines hyperboliques qui interpolent les données y_i , $i=1,2,\dots,n$.

2.2.1. Propriétés extrémales

Définissons pour $f, g \in H^2[a,b]$

$$P_0(f,g) = \int_a^b Lf(t).Lg(t)dt$$

$$P_1(f,g) = \int_a^b [D^2f(t).D^2g(t) + r^2(t)Df(t).Dg(t)] dt$$

et $E_0(f) = P_0(f,f)$, $E_1(f) = P_1(f,f)$.

Nous considérons le problème de minimisation suivant

$$(2.1) \left\{ \begin{array}{l} \text{minimiser } E_k(x) \text{ (} k=0,1 \text{)} \\ \text{avec } x \in H^2 \text{ vérifiant } x(t_j) = y_j \text{ (} i=1,2,\dots,n \text{)} \end{array} \right.$$

Proposition 2.1

Il existe une solution unique σ du problème (2.1) pour $k=0$. Cette solution est caractérisée par

(i) σ s'exprime comme combinaison linéaire de $\{1, t, \operatorname{sh}_r t, \operatorname{ch}_r t\}$ dans chacun des intervalles $]t_i, t_{i+1}[$, $i=1, 2, \dots, n-1$

(ii) σ est une polynôme de degré 1 sur $[a, t_1[$ et $]t_n, b]$

(iii) $\sigma \in C^1[a, b]$ et $(\sigma'' - r\sigma') \Big|_{t_i^-}^{t_i^+} = 0$, $i=1, 2, \dots, n$

(iv) $\sigma(t_i) = y_i$ $i=1, 2, \dots, n$ \square

Proposition 2.2

(i) Il existe un élément unique $\sigma \in S_H$ tel que $\sigma(t_i) = y_i$ ($i=1, 2, \dots, n$)

(ii) Si l'on note $I_y = \{ x \in H^2 / x(t_i) = y_i \quad i=1, 2, \dots, n \}$

$$\text{alors} \quad E_1(\sigma) = \min_{x \in I_y} E_1(x)$$

et σ est le seul élément de I_y qui a cette propriété. \square

Remarque

Si la tension est uniforme on obtient le même résultat lorsqu'on minimise E_0 ou E_1 , une fonction qui appartient à l'espace S_H . La preuve est donnée par Nielson [81].

Dans le cas où la tension est variable nous préférons traiter le problème associé à E_1 , car la solution est deux fois continûment différentiable.

2.2.2. Représentation des F.S.H.T.

Soit f la fonction à interpoler définie sur $[a,b]$.

$$\begin{aligned} \text{Notons } y_i &= f(t_i) \\ \lambda_i &= \sigma'(t_i), \quad \beta_i = \sigma''(t_i) & i=1,2,\dots,n \\ h_i &= t_{i+1}-t_i, \quad u_i = (t-t_i)/h_i, \quad m_i = (y_{i+1}-y_i)/h_i \\ & & i=1,2,\dots,n-1 \end{aligned}$$

La représentation de $\sigma(t)$ sur $[t_i, t_{i+1}]$ peut être exprimée en termes des données d'interpolation y_i, y_{i+1} et des dérivées premières inconnues λ_i, λ_{i+1} . Alors

$$(2.2) \quad \sigma(t) = y_{i+1}u_i + y_i(1-u_i) + [(\lambda_i + \eta_i \lambda_{i+1} - (\eta_i + 1)m_i) / (\eta_i^2 - 1)] \varphi(u_i) - [(\eta_i \lambda_i + \lambda_{i+1} - (\eta_i + 1)m_i) / (\eta_i^2 - 1)] \varphi(1-u_i)$$

où

$$(2.3) \quad \left\{ \begin{array}{l} \eta_i = (p_i \operatorname{ch} p_i - \operatorname{sh} p_i) / (\operatorname{sh} p_i - p_i), \\ \varphi(v) = (\operatorname{sh}(p_i v) - v \operatorname{sh} p_i) / (\operatorname{sh} p_i - p_i) \\ p_i = r_i h_i \end{array} \right. \quad i=1,2,\dots,n-1$$

Mais pour des petites valeurs de p_i ces formules peuvent être instables pour le calcul sur l'ordinateur. Alors on va prendre un développement de Taylor tronqué au terme p_i^6 pour $p_i < 0.5$, ce qui donne

$$(2.4) \quad \eta_i = 2 \left[1 + \frac{p_i^2}{20 + p_i^2} \right], \quad \varphi(v) = (v^3 - v) \left[1 + \frac{(p_i v)^2}{20 + p_i^2} \right]$$

Bien entendu pour $p_i = 0.5$, il y a une discontinuité par rapport à r_i ou à h_i . Dans différents exemples numériques nous avons constaté que les valeurs de cette discontinuité étaient "faibles" (de l'ordre de 10^{-5}).

Le raccordement de la dérivée seconde aux noeuds intérieurs entraîne pour $i = 2, 3, \dots, n-1$:

$$(2.5) \quad \alpha_{i-1} \lambda_{i-1} + (\alpha_{i-1} \eta_{i-1} + \alpha_i \eta_i) \lambda_i + \alpha_i \lambda_{i+1} = \alpha_{i-1} (\eta_{i-1} + 1) m_{i-1} + \alpha_i (\eta_i + 1) m_i$$

$$\text{où } \alpha_i = \begin{cases} \frac{r_i \rho_i \operatorname{sh} \rho_i}{(\eta_i^2 - 1) (\operatorname{sh} \rho_i - \rho_i)} & \text{si } \rho_i \geq 0.5 \\ \frac{6 + \rho_i^2}{(4(1 + \rho_i^2 / (20 + \rho_i^2))^2 - 1) h_i} & \text{sinon} \end{cases}$$

Les conditions de bout donnent

$$(2.6) \quad \begin{cases} \eta_1 \lambda_1 + \lambda_2 = (\eta_1 + 1) m_1 \\ \lambda_{n-1} + \eta_{n-1} \lambda_n = (\eta_{n-1} + 1) m_{n-1} \end{cases}$$

Pour calculer les dérivées premières λ_i nous devons résoudre un système tridiagonal symétrique, à diagonale strictement dominante. \square

Nous pouvons utiliser les valeurs des dérivées secondes aux noeuds β_j au lieu des dérivées premières λ_j dans la définition de la fonction. On a

$$(2.7) \quad \sigma(t) = \gamma_{i+1} u_i + \gamma_i (1 - u_i) + h_i^2 [\beta_{i+1} \psi(u_i) + \beta_i \psi(1 - u_i)]$$

$$\text{où } \psi(v) = \begin{cases} ((\operatorname{sh}(\rho_i v) / \operatorname{sh} \rho_i) - v) / \rho_i^2 & \text{si } \rho_i \geq 0.5 \\ (v^3 - v) ((20 + \rho_i^2 (v^2 - 1)) / (120 + 20 \rho_i^2)) & \text{sinon} \end{cases}$$

La continuité de la dérivée première aux points intérieurs plus les conditions de bout nous amènent au système tridiagonal suivant :

$$(2.8) \begin{cases} \beta_1 = 0 \\ e_{i-1} \beta_{i-1} + (d_{i-1} + d_i) \beta_i + e_i \beta_{i+1} = m_i - m_{i-1} \quad i=2,3,\dots,n-1 \\ \beta_n = 0 \end{cases}$$

où

$$e_i = (\text{sh}p_i - p_i) h_i / (p_i^2 \text{sh}p_i), \quad d_i = (p_i \text{ch}p_i - \text{sh}p_i) h_i / (p_i^2 \text{sh}p_i)$$

si $p_i \geq 0.5$;

$$e_i = h_i \left(1 - \frac{7p_i^2}{60 + 10p_i^2} \right) / 6, \quad d_i = h_i \left(1 - \frac{2p_i^2}{30 + 5p_i^2} \right)$$

si $p_i < 0.5$

Ce système est aussi symétrique et à diagonale strictement dominante, donc l'élimination Gauss sans pivot est stable.

2.2.3. Un cas particulier : tension uniforme et pas équadistants

Avec le double propos, d'optimiser le calcul et améliorer la convergence de la fonction spline cubique d'interpolation aux noeuds équadistants, Laurent [58] étudie la manière de choisir les conditions de bout. Nous allons suivre une démarche semblable pour simplifier la résolution du système linéaire (2.5) lorsque la tension est uniforme et les abscisses équadistantes.

On notera

$$h = h_i, \quad r = r_i, \quad p = p_i, \quad \eta = \eta_i \quad i = 1, 2, \dots, n-1$$

Nous allons considérer le cas $p \geq 0.5$ donc $\eta = (p \text{ch}p - \text{sh}p) / (\text{sh}p - p)$

Les équations (2.5) deviennent

$$(2.9) \quad \lambda_{i-1} + 2\eta\lambda_i + \lambda_{i+1} = \theta_i = (\eta+1)(y_{i+1}-y_i)/h \quad i=2,\dots,n-1$$

auxquelles on ajoute les conditions de bout

$$(2.10) \quad \begin{cases} a\lambda_1 + b\lambda_2 = \theta_1 \\ b\lambda_{n-1} + a\lambda_n = \theta_n \end{cases}$$

Les valeurs θ_1 et θ_n sont choisies en fonction des y_i et le rapport b/a de façon à optimiser les calculs.

Les relations (2.9) correspondent à l'identité

$$q'(t_{i-1}) + 2\eta q'(t_i) + q'(t_{i+1}) = (\eta+1)(q(t_{i+1}) - q(t_{i-1}))/h$$

pour tout $q \in \mathcal{P}$, où

$$\mathcal{P} = \{ q(t) = \alpha_1 + \alpha_2 t + \alpha_3 t^2 + \alpha_4 \text{sh}(rh) + \alpha_5 \text{ch}(rh) \}$$

$$\text{On va choisir } \theta_1 = \left(\sum_{i=1}^5 \delta_i y_i \right) / h \quad \text{et} \quad \theta_n = \left(\sum_{i=1}^5 \delta_i y_{n+1-i} \right) / h$$

où les coefficients δ_i sont tels que

$$a q'(t_1) + b q'(t_2) = \left(\sum_{i=1}^5 \delta_i q(t_i) \right) / h \quad \forall q \in \mathcal{P}.$$

Résolution du système linéaire

Nous allons décrire la manière de résoudre le système

$$(2.11) \quad \begin{cases} a\lambda_1 + b\lambda_2 = \theta_1 \\ \lambda_{i-1} + 2\eta\lambda_i + \lambda_{i+1} = \theta_i \\ b\lambda_{n-1} + a\lambda_n = \theta_n \end{cases} \quad i=2,\dots,n-1$$

Dans l'élimination de Gauss, la triangularisation jusqu'à l'indice i peut s'écrire comme une relation

$$a_i \lambda_i + b_i \lambda_{i+1} = c_i$$

On commence avec $a_1 = a$, $b_1 = b$ et $c_1 = \theta_1$.

En utilisant (2.9) on obtient

$$\begin{cases} a_{i+1} = -2\eta + b_i \\ b_{i+1} = -a_i \\ c_{i+1} = c_i - a_i b_{i+1} \end{cases}$$

On passe de a_i, b_i à a_{i+1}, b_{i+1} par la transformation linéaire de matrice

$$M = \begin{bmatrix} -2\eta & 1 \\ -1 & 0 \end{bmatrix}$$

dont les valeurs propres sont $\mu = -\eta + (\eta^2 - 1)^{1/2}$ et $\mu' = -\eta - (\eta^2 - 1)^{1/2}$. On remarque que $-2 + \sqrt{3} \leq \mu < 0$ et $\mu' \leq -2 - \sqrt{3}$.

On a la convergence de b_i/a_i vers $x_2/x_1 = -\mu$ où $[x_1, x_2]$ est le vecteur propre associé au rayon spectral μ . Si l'on prend le rapport $b/a = -\mu$ alors on aura toujours $b_i/a_i = -\mu$.

Pour la triangularisation on part de la relation

$$(-1/\mu)\lambda_1 + \lambda_2 = c_1 = \theta_1$$

et on arrive à

$$(-1/\mu)\lambda_i + \lambda_{i+1} = c_i$$

avec $c_{i+1} = \mu c_i + \theta_{i+1}$ pour $i=1, \dots, n-2$

Si on combine l'avant dernière équation

$$(-1/\mu)\lambda_{n-1} + \lambda_n = c_{n-1}$$

avec $\lambda_{n-1} + (-1/\mu)\lambda_n = \theta_n$

on obtient

$$\lambda_n = \frac{1}{2(\eta^2-1)^{1/2}} [\theta_n + \mu.c_{n-1}]$$

Le retour en arrière devient

$$\lambda_i = \mu(\lambda_{i+1} - c_i) \quad i = n-1, n-2, \dots, 1$$

Après quelques calculs élémentaires on trouve les coefficients δ_i :

$$\left\{ \begin{array}{l} \delta_1 = -\alpha + \beta \\ \delta_2 = 2(\alpha - 2\beta) - (3 - 5/\mu)/2 \\ \delta_3 = 6\beta + 2(1 - 2/\mu) \\ \delta_4 = -2(\alpha + 2\beta) - (1 - 3/\mu)/2 \\ \delta_5 = \alpha + \beta \end{array} \right.$$

où $\alpha = [(pch - shp) - (pch(2p) - shp)/\mu] / [2(sh(2p) - 2shp)]$

$$\beta = [(2chp - pshp) - (4chp - psh(2p))/\mu] / [2(ch(2p) - 4chp)]$$

2.2.4. Comportement dans les cas limites

On remarque que si $r_i = 0$ ($i=1, 2, \dots, n-1$) σ est la fonction spline cubique. L'autre cas limite intéressant se présente quand les paramètres de tension tendent vers l'infini.

De l'équation

$$\frac{1}{r_i^2} D^4[\sigma(t)] - D^2[\sigma(t)] = 0 \quad t_i \leq t \leq t_{i+1}$$

on obtient que la fonction σ a la dérivée seconde égale à zéro lorsque $r_i \rightarrow \infty$, donc σ est un segment de droite sur cet intervalle.

Ces propriétés ont été l'objet des travaux de Späth [106], Baum [7] et de manière plus particulière de Pruess [91]. A partir de ces résultats et l'étude des dérivées on peut envisager de trouver des paramètres de tension suffisamment grands pour construire des approximations monotones et convexes.

Si la dérivée seconde d'une fonction connue ne change pas de signe dans un certain intervalle alors que la dérivée seconde de la fonction interpolante σ change de signe on peut dire que le point d'inflexion de l'interpolante est "*artificiel*". Mais nous ne connaissons que les valeurs discrètes de la fonction dans les abscisses t_j , ce qui a motivé la définition suivante :

un **point d'inflexion** de σ qui apparaît dans l'intervalle $[t_j, t_{j+1}]$ est **artificiel** si Δ_j^2 et Δ_{j+1}^2 ont le même signe.

Ici Δ_j^2 désigne la différence divisée d'ordre 2 qui correspond aux abscisses t_{j-1}, t_j, t_{j+1} .

Späth [106] démontre que pour des valeurs suffisamment grandes du triplet (r_{j-1}, r_j, r_{j+1}) on peut faire disparaître les points d'inflexion aberrants et donc préserver la convexité des données. Une étude de la dérivée première donne des conclusions semblables pour la conservation de la monotonie.

Ces résultats sont de nature asymptotique et ne donnent pas de formules analytiques des paramètres de tension à partir desquels la monotonie ou la convexité est respectée. Alors leur choix est fait avec des algorithmes de caractère heuristique. Nous traiterons plus loin cette question.

2.3. Lissage

Lorsque les données $z_i (i=1, 2, \dots, n)$ sont entachées d'erreurs, il est convenable d'utiliser un critère qui réalise un compromis entre le lissage et l'approximation des données.

Si $J(x) = \sum_{i=1}^n (x(t_i) - z_i)^2$ pour $x \in H^2$, nous étudions le problème :

$$(2.12) \quad \begin{cases} \text{minimiser } E_k(x) + \rho J(x) & (k=0, 1) \\ \text{avec } x \in H^2 \end{cases}$$

Le coefficient ρ détermine l'importance relative que l'on accorde au lissage de la solution et à l'approximation des données. Les algorithmes de choix de ce paramètre seront discutés ultérieurement.

Étudions d'abord la question de l'existence, l'unicité et la caractérisation de la solution du problème (2.12).

Proposition 2.3.

Il existe une solution unique σ du problème (2.12) pour $k=0$. Cette solution est caractérisée par

- (i) σ s'exprime comme combinaison linéaire de $(1, t, \operatorname{sh}_r t, \operatorname{ch}_r t)$ dans chacun des intervalles $]t_i, t_{i+1}[$, $i=1, 2, \dots, n-1$
- (ii) σ est un polynôme de degré 1 sur $[a, t_1[$, $]t_n, b]$

$$(iii) \quad \sigma \in C^1[a, b] \text{ et } (\sigma'' - r \sigma') \Big|_{t_i^-}^{t_i^+} = 0, \quad i=1, 2, \dots, n$$

$$(iv) \quad (\sigma''' - r^2 \sigma') \Big|_{t_i^-}^{t_i^+} + \rho(\sigma(t_i) - z_i) = 0 \quad i=1, 2, \dots, n \quad \square$$

Proposition 2.4

(i) Il existe un élément unique $\sigma \in S_H$ tel que

$$(\sigma''' - r^2 \sigma') \int_{t_i^-}^{t_i^+} + \rho(\sigma(t_i) - z_i) = 0, \quad i=1,2,\dots,n$$

(ii) De plus, on a

$$E_1(\sigma) + \rho J(\sigma) = \min_{x \in H^2} [E_1(x) + \rho J(x)]$$

et σ est l'unique élément de H^2 qui a cette propriété. \square

Remarques

1) L'équation de lissage est la même pour $k=0$ et $k=1$.

2) Si la tension est uniforme on obtient le même résultat pour le problème (2.12) pour $k=0$ et $k=1$, une fonction qui appartient à S_H .

3) Dans le cas où la tension est variable nous préférons traiter le problème associé à E_1 , car la solution est de classe C^2 . La preuve est donnée par Pruess dans [92].

2.3.1. Algorithme de calcul

Pour déterminer complètement la spline, il faut calculer les valeurs de $\gamma_i = \sigma(t_i)$ et $\beta_i = \sigma''(t_i)$, $i=1,2,\dots,n$. Pour cela on utilise la continuité de $\sigma'(t)$ et la relation (i) de la proposition 2.4.

Pruess [92] suggère une procédure qui fait appel aux bases locales. Par contre, Cinquin [19] résout un système d'ordre $2n$ dont les inconnues sont rangées dans le vecteur $[\gamma_1, \gamma_2, \dots, \gamma_n, \beta_1, \beta_2, \dots, \beta_n]^T$.

Nous préférons écrire le système dans une forme tridiagonale par blocs 2×2 :

$$(2.14) \quad \begin{cases} B_1 \Lambda_1 + C_1 \Lambda_2 = D_1 \\ A_i \Lambda_{i-1} + B_i \Lambda_i + C_i \Lambda_{i+1} = D_i & i=2,3,\dots,n-1 \\ A_n \Lambda_{n-1} + B_n \Lambda_n = D_n \end{cases}$$

où

$$B_1 = \begin{bmatrix} -(\rho h_1 + r_1^2) & 0 \\ 0 & 1 \end{bmatrix} \quad B_n = \begin{bmatrix} -(\rho h_{n-1} + r_{n-1}^2) & 0 \\ 0 & 1 \end{bmatrix}$$

$$B_i = \begin{bmatrix} -(\rho h_i + r_i^2 + \gamma_i r_{i-1}^2) & 1 + \gamma_i \\ 1 + \gamma_i & b_i + \gamma_i b_{i-1} \end{bmatrix} \quad i=2,\dots,n-1$$

$$C_1 = \begin{bmatrix} r_1^2 & -1 \\ 0 & 0 \end{bmatrix} \quad C_{n-1} = \begin{bmatrix} r_{n-1}^2 & 0 \\ -1 & 0 \end{bmatrix}$$

$$C_i = \begin{bmatrix} r_i^2 & -1 \\ -1 & a_i \end{bmatrix} \quad i=2,\dots,n-2$$

$$A_i = \gamma_i C_{i-1} \quad i=2,\dots,n \quad \text{avec } h_n = h_{n-1}, \quad \gamma_i = h_i / h_{i-1}$$

$$\Lambda_i = [y_i \ \beta_i]^T \quad \text{et} \quad D_i = [-\rho h_i z_i \ 0]^T \quad i=1,2,\dots,n$$

$$a_i = (shp_i - p_i) / (r_i^2 shp_i), \quad b_i = (p_i chp_i - shp_i) / (r_i^2 shp_i) \\ \text{si } p_i \geq 0.5$$

$$a_i = \left[1 - \frac{7p_i^2}{60 + 10p_i^2} \right] h_i^2 / 6, \quad b_i = \left[1 - \frac{2p_i^2}{30 + 5p_i^2} \right] h_i^2 / 3 \\ \text{si } p_i < 0.5$$

Nous rappelons qu'il existe une discontinuité de a_i et b_i par rapport à r_i et h_i lorsque $p_i = 0.5$.

La résolution de ce système nous la faisons par l'élimination simple de Gauss par bloc.

$$(2.15) \quad \left\{ \begin{array}{l} U_i = B_i \qquad \qquad \qquad Y_i = D_i \\ U_{i+1} = B_{i+1} - \gamma_{i+1} C_i U_i^{-1} C_i \\ Y_{i+1} = D_{i+1} - \gamma_{i+1} C_i U_i^{-1} Y_i \end{array} \right. \quad i=1, \dots, n-1$$

Le retour en arrière est donné par

$$(2.16) \quad \left\{ \begin{array}{l} \Lambda_n = U_n^{-1} Y_n \\ \Lambda_i = U_i^{-1} (Y_i - C_i \Lambda_{i+1}) \end{array} \right. \quad i=n-1, \dots, 1$$

La stabilité de la méthode peut être démontrée en utilisant les résultats de Varah [112].

Il est clair que si la tension est uniforme et les noeuds équidistants, on trouve d'importantes simplifications dans le calcul. En particulier, si le pas est uniforme le système devient symétrique par blocs.

III.3. FONCTIONS SPLINES RATIONNELLES SOUS TENSION

Nous avons déjà signalé que l'inconvénient principal des splines hyperboliques sous tension est l'évaluation coûteuse de la fonction exponentielle. Späth [107] a cherché, dans l'espace des splines rationnelles, une alternative qui ait un comportement semblable aux splines exponentielles. Le calcul, alors, devient plus économique.

On procède de la manière suivante : dans chaque intervalle, on incorpore à la fonction spline cubique un dénominateur contenant la tension. L'introduction de ce degré de liberté supplémentaire sert à l'élimination des pôles dans l'intervalle qui nous intéresse.

Plus précisément, la spline cubique peut s'exprimer dans l'intervalle $[t_j, t_{j+1}]$ comme une combinaison linéaire de la base

$$\{ u, 1-u, b_j(u), b_j(1-u) \} \quad \text{où } u = (t_j - t) / h_j \text{ et } b_j(v) = v^3 - v$$

Nous cherchons une spline telle que dans chaque intervalle $[t_j, t_{j+1}]$ elle puisse s'écrire dans la base

$$\{ u, 1-u, \varphi_j(u), \varphi_j(1-u) \} \quad \text{où } \varphi_j(v) = \frac{v^3}{r_j(1-v) + 1} - v$$

$$\text{avec } r_j \geq 0 \quad (i=1,2,\dots,n-1)$$

et qui soit deux fois continûment différentiable sur $[a,b]$.

Cette approche n'étant pas variationnelle, elle nous permet de traiter exclusivement l'interpolation des données.

Représentation et calcul de la fonction

La fonction $\sigma(t)$ sur $[t_j, t_{j+1}]$ ($i=1,2,\dots,n-1$) en termes de points d'interpolation y_j, y_{j+1} et des dérivées λ_j, λ_{j+1} s'écrit:

$$(3.1) \quad \sigma(t) = y_{i+1}u + y_i(1-u) + \{ [\lambda_{i+1} + (2+r_i)\lambda_i - (3+r_i)m_i] \varphi_i(u) + \\ [(2+r_i)\lambda_{i+1} + \lambda_i - (3+r_i)m_i] \varphi_i(1-u) \} h_i / (4r_i^2 + 4r_i + 3)$$

Comme $r_i > 0$ on peut vérifier aisément que $\sigma(t)$ n'a pas de pôles dans cet intervalle.

La condition requise pour que $\sigma''(t)$ soit continue en t_i ($i=2, \dots, n-1$) se traduit par :

$$(3.2) \quad \alpha_{i-1} \lambda_{i-1} + [(2+r_{i-1})\alpha_{i-1} + (2+r_i)\alpha_i] \lambda_i + \alpha_i \lambda_{i+1} = \\ (3+r_{i-1})\alpha_{i-1}m_{i-1} + (3+r_i)\alpha_i m_i$$

avec $\alpha_i = \frac{1}{h_i} \left(\frac{2r_i^2 + 3r_i + 3}{4r_i^2 + 4r_i + 3} \right)$

Les conditions naturelles de bout donnent

$$(3.3) \quad \begin{cases} (2+r_1)\lambda_1 + \lambda_2 = (3+r_1)m_1 \\ \lambda_{n-1} + (2+r_{n-1})\lambda_n = (3+r_{n-1})m_{n-1} \end{cases}$$

Le système (3.2), (3.3) est symétrique, à diagonale strictement dominante et donc solvable par la méthode de Gauss sans pivot.

Ces splines rationnelles sous tension se trouvent entre les splines cubiques et les splines linéaires qui correspondent aux cas limites lorsque, respectivement, $r_i \rightarrow 0$ et $r_i \rightarrow \infty$.

Les fluctuations de la spline peuvent être quantifiées par la déviation de la corde qui relie deux points donnés adjacents. A partir de cette remarque Frost et Kinzel [42] calculent les paramètres de tension en imposant le respect d'une distance maximale entre la fonction interpolante et la spline linéaire par morceaux. Ces auteurs préfèrent utiliser les dérivées secondes comme inconnues.

III.4. FONCTIONS ν -SPLINES

Une première réponse au problème de trouver des méthodes polynomiales en appliquant la tension aux splines cubiques a été donnée par Nielson [81].

Au lieu du critère de lissage

$$E_1(x) = \int_a^b [x''(t)]^2 dt + \sum_{i=1}^{n-1} r_i^2 \left(\int_{t_i}^{t_{i+1}} [x'(t)]^2 dt \right)$$

Nielson propose d'utiliser

$$E_2(x) = \int_a^b [x''(t)]^2 dt + \sum_{i=1}^n \nu_i [x'(t_i)]^2 \quad (\nu_i > 0)$$

La solution, appelée ν -*spline*, est une cubique mais qui perd la continuité de la dérivée seconde aux noeuds. On remarque que la tension est appliquée aux points et non aux intervalles comme c'est le cas pour les fonctions splines hyperboliques sous tension.

Cette spline présente l'intérêt d'avoir la continuité de la courbure lorsqu'on l'utilise sous la forme paramétrée. Barsky [5] a étudié de plus près ce problème.

La méthode a été développée originalement pour l'interpolation des données, nous l'avons étendue au lissage.

4.1. Interpolation

Traitons le problème suivant

$$(4.1) \left\{ \begin{array}{l} \text{minimiser } E_2(x) \\ \text{avec } x \in H^2[a,b] \text{ vérifiant } x(t_i) = y_i \quad (i=1, \dots, n). \end{array} \right.$$

L'ensemble S_{ν} des **fonctions ν -splines** sera formé par les éléments s tels que

- (i) s est un polynôme de degré 3 sur chacun des intervalles $[t_i, t_{i+1}]$ pour $i=1,2,\dots,n-1$.
- (ii) s est un polynôme de degré 1 sur chacun des intervalles $[a, t_1], [t_n, b]$
- (iii) $s \in C^1[a, b]$
- (iv) $\Delta s''(t_i) = \nu_i s'(t_i) \quad i=1,2,\dots,n$

Ici, on note $\Delta s^{(j)}(t_i) = s^{(j)}(t_i^+) - s^{(j)}(t_i^-)$.

Remarques

1) L'ensemble S_{ν} forme un sous-espace de dimension n de H^2 . Une base de S est constituée par les éléments $\{\sigma_i\} \quad i=1,2,\dots,n$ tels que

$$\sigma_i(t_j) = \begin{cases} 1 & \text{si } j=i \\ 0 & \text{si } j \neq i \end{cases}$$

2) Si $\nu_i = 0 \quad (i=1,2,\dots,n)$ on retrouve les fonctions splines cubiques "naturelles".

Proposition 4.1

Le problème (4.1) a une solution unique σ dans S_{ν} . \square

On donne ensuite deux propriétés importantes sur ces splines:

Propriétés

(i) Soit $k \in \{2,3,\dots,n\}$. Si les ν_j tels que $j \neq k$ restent fixes

$$\text{alors } \lim_{\nu_k \rightarrow \infty} (\lambda_k)^2 = 0$$

(ii) Soient $s \in S_\nu$ et $x \in H^2[a,b]$ alors

$$\int_a^b s''(t) x''(t) dt + \sum_{i=1}^n \nu_i s'(t_i) x'(t_i) = \sum_{i=1}^n \Delta s'''(t_i) x(t_i)$$

La première affirmation permet de voir le comportement des dérivées aux noeuds lorsque le paramètre de tension associé devient grand, ce qui explique la formation de paliers dans le graphe autour des points à interpoler.

L'égalité (ii) résulte de l'intégration par parties. \square

Nous exprimons la fonction σ sous la forme de Newton avec noeuds confondus, ce qui nous donne une représentation

(4.2)

$$\sigma(t) = y_j + \lambda_j(t-t_j) + \frac{m_j - \lambda_j}{h_j} (t-t_j)^2 + \frac{\lambda_{j+1} - 2m_j + \lambda_j}{h_j^2} (t-t_j)^2(t-t_{j+1})$$

pour $t \in [t_j, t_{j+1}]$, $j=1,2,\dots,n-1$

Les inconnues $\lambda_j (j=1,2,\dots,n)$ sont solutions du système tridiagonal suivant qui provient du raccordement de $s''(t)$ en t_j :

$$(4.3) \left\{ \begin{array}{l} 2 \left[\frac{1}{h_1} + \frac{v_1}{4} \right] \lambda_1 + \frac{\lambda_2}{h_1} = \frac{3}{h_1} m_1 \\ \frac{\lambda_{i-1}}{h_{i-1}} + 2 \left[\frac{1}{h_{i-1}} + \frac{1}{h_i} + \frac{v_i}{4} \right] \lambda_i + \frac{\lambda_{i+1}}{h_i} = 3 \left[\frac{m_{i-1}}{h_{i-1}} + \frac{m_i}{h_i} \right] \\ \hspace{15em} i=2,3,\dots,n-1 \\ \frac{\lambda_{n-1}}{h_{n-1}} + 2 \left[\frac{1}{h_{n-1}} + \frac{v_n}{4} \right] \lambda_n = 3 \frac{m_{n-1}}{h_{n-1}} \end{array} \right.$$

On retrouve le même système que pour les cubiques naturelles avec en plus une perturbation dans la diagonale qui croît au fur et à mesure que les paramètres de tension augmentent. Alors la méthode de Gauss sans pivot est stable.

4.2. Cas paramétré

La représentation paramétrée d'une courbe consiste à exprimer chaque composante comme une fonction d'une seule variable (le paramètre). Les coordonnées d'un point dans le plan réel nous les écrirons en forme vectorielle :

$$P(t) = [x(t), y(t)]$$

et ses dérivées :

$$P'(t) = [\dot{x}(t), \dot{y}(t)]$$

$$P''(t) = [\ddot{x}(t), \ddot{y}(t)]$$

Si l'ensemble des données à interpoler est $((x_i, y_i)) \quad i=1,2,\dots,n$, le problème à considérer est

Nous pouvons énoncer la

Proposition 4.2

- (i) Si $\dot{x}(t) \neq 0$ alors d^2y/dx^2 est continue en $x(t)$ et
si $\dot{y}(t) \neq 0$ alors d^2x/dy^2 est continue en $y(t)$
- (ii) Si $\dot{x}(t) \neq 0$ ou $\dot{y}(t) \neq 0$ alors la courbure de $P = [x, y]$ est continue en t . \square

Nous rappelons la formule de la courbure :

$$\kappa(t) = \frac{|\dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t)|}{(\dot{x}(t)^2 + \dot{y}(t)^2)^{3/2}}$$

Pour terminer ce paragraphe nous écrivons un résultat dû à Nielson [81] sur le comportement asymptotique de ces fonctions.

Proposition 4.3

La courbe (σ_x, σ_y) solution du problème (4.4) tend vers le segment de droite qui relie (x_1, y_1) et (x_{i+1}, y_{i+1}) lorsque v_i et v_{i+1} tendent vers l'infini (les autres paramètres v_j $j \neq i$ restant fixes).

4.3. Lissage

Examinons maintenant le problème

$$(4.5) \left\{ \begin{array}{l} \text{minimiser } E_2(x) + \rho J(x) \quad \text{avec } x \in H^2 \end{array} \right.$$

On rappelle que $J(x) = \sum_{i=1}^n (x(t_i) - z_i)^2$ où les z_i sont des données

entachées d'erreurs, et que $E_2(x) = \int_a^b [\ddot{x}(t)]^2 dt + \sum_{i=1}^n v_i [x'(t_i)]^2$

La détermination de la fonction v -spline d'ajustement peut être ramenée à un problème de minimisation dans \mathbb{R}^n .

Posons $I_y = \{ x \in H^2 / x(t_i) = y_i, i = 1, 2, \dots, n \}$

On définit la fonction Φ de \mathbb{R}^n dans \mathbb{R} de la manière suivante :

$$(4.6) \quad \Phi(y) = \min_{x \in I_y} E_2(x)$$

Alors (4.5) est équivalent à

$$(4.7) \quad \min_{y \in \mathbb{R}^n} [\Phi(y) + \rho \sum_{i=1}^n (y_i - z_i)^2]$$

Dans la proposition qui suit, nous allons établir les propriétés de la fonction Φ .

Proposition 4.4

La fonction $\Phi(y)$ est une fonction convexe, quadratique, dérivable. Elle s'écrit matriciellement

$$\Phi(y) = \langle y, \Omega y \rangle$$

avec $\Omega = (w_{ij}) \quad i, j = 1, 2, \dots, n$

$$w_{ij} = \int_a^b \sigma''_i(t) \sigma''_j(t) dt + \sum_{k=1}^n v_k \sigma'_i(t_k) \sigma'_j(t_k)$$

Son gradient est

$$\Phi'(y) = 2\Omega y = 2g$$

et l'on a

$$g_i = \sigma'''(t_i^+) - \sigma'''(t_i^-)$$

où $\sigma \in S_v$ vérifie $\sigma(t_i) = y_i \quad (i=1, 2, \dots, n)$

Preuve

La fonction $\sigma_y \in S_v$ solution du problème d'interpolation (4.6) s'écrit

$$\sigma_y = \sum_{i=1}^n y_i \sigma_i$$

On a donc

$$\int_a^b [\sigma_y''(t)]^2 dt + \sum_{k=1}^n v_k [\sigma_y'(t_k)]^2 = \sum_{i,j=1}^n (\alpha_{ij} + \beta_{ij}) y_i y_j$$

où $\alpha_{ij} = \int_a^b \sigma_i''(t) \sigma_j''(t) dt, \quad \beta_{ij} = \sum_{k=1}^n v_k \sigma_i'(t_k) \sigma_j'(t_k)$

La propriété (ii) du paragraphe 4.1. nous permet l'évaluation de

$$w_{ij} = \alpha_{ij} + \beta_{ij},$$

$$w_{ij} = \sum_{k=1}^n [\sigma_i'''(t_k^+) - \sigma_i'''(t_k^-)] \sigma_j(t_k) = \sigma_i'''(t_j^+) - \sigma_i'''(t_j^-)$$

$$i, j = 1, 2, \dots, n$$

La matrice $\Omega = (w_{ij})_{i,j=1,\dots,n}$ est symétrique. On déduit que

$$\Phi(y) = E_2(\sigma_y) = \langle y, \Omega y \rangle$$

La fonction Φ est quadratique par rapport à y . En plus, $\langle y, \Omega y \rangle \geq 0$ pour tout $y \in \mathbb{R}^n$, ce qui revient à dire que Ω est semi-définie positive. On conclut que Φ est convexe.

La fonction Φ est aussi continue et dérivable. Son gradient en y est

$$\Phi'(y) = 2\Omega y$$

Posons $\Phi'(y) = 2g$. On a donc $g = \Omega y$ et par linéarité

$$g_i = \sum_{j=1}^n w_{ij} y_j$$

ou encore

$$g_i = \sigma'''(t_i^+) - \sigma'''(t_i^-)$$

σ étant l'élément de S_y qui vérifie $\sigma(t_i) = y_i$ ($i=1,2,\dots,n$). \square

Il en découle alors le résultat suivant:

Proposition 4.5.

Le problème (4.7) admet une solution unique $\sigma \in S_y$ caractérisée par

$$\sigma'''(t_i^+) - \sigma'''(t_i^-) + \rho(\sigma(t_i) - z_i) = 0 \quad i=1,2,\dots,n$$

Preuve

En effet, le gradient correspondant à la fonction convexe

$$\langle y, \Omega y \rangle + \rho \sum_{i=1}^n (y_i - z_i)^2$$

est

$$\Omega y + \rho(y - z)$$

et il est égal à zéro si et seulement si

$$g_i + \rho(\sigma(t_i) - z_i) = 0 \quad i=1,2,\dots,n$$

où $\sigma \in S_y$ tel que $\sigma(t_i) = y_i$ $i=1,2,\dots,n$.

D'après le théorème 7.4.4. ([18], p.156) sur les minimums des fonctions convexes, nous pouvons conclure que σ est la solution unique de (4.7.) donc de (4.5.)□

4.3.1. Algorithme de calcul

Pour calculer les valeurs $y_i = \sigma(t_i)$ et $\lambda_i = \sigma'(t_i)$, $i=1,2,\dots,n$, nous utilisons les conditions

$$\Delta\sigma''(t_i) = v_i\sigma'(t_i), \quad \Delta\sigma'''(t_i) = \rho(z_i - \sigma(t_i)) \quad i=1,2,\dots,n$$

On obtient un système linéaire qui se présente sous la forme:

$$(4.8.) \left\{ \begin{array}{l} (\alpha_1 + 6)y_1 + 3h_1\lambda_1 - 6y_2 + 3h_1\lambda_2 = \alpha_1 z_1 \\ 3y_1 + 2\left[1 + \frac{v_1}{4}h_1\right]h_1\lambda_1 - 3y_2 + h_1\lambda_2 = 0 \\ -6\gamma_i^3 y_{i-1} - 3\gamma_i^2 h_i \lambda_{i-1} + [\alpha_i + 6(\gamma_i^3 + 1)]y_i + 3[1 - \gamma_i^2]h_i \lambda_i - \\ \quad 6y_{i+1} + 3h_i \lambda_{i+1} = \alpha_i z_i \\ 3\gamma_i^2 y_{i-1} + \gamma_i h_i \lambda_i + 3[1 - \gamma_i^2]y_i + 2\left[1 + \gamma_i + \frac{v_i}{4}h_i\right]h_i \lambda_i - \\ \quad 3y_{i+1} + h_i \lambda_{i+1} = 0 \quad i=2,3,\dots,n-1 \\ -6y_{n-1} - 3h_{n-1}\lambda_{n-1} + (\alpha_{n-1} + 6)y_n - 3h_{n-1}\lambda_n = \alpha_{n-1} z_n \\ 3y_{n-1} + h_{n-1}\lambda_{n-1} - 3y_n - 2\left[1 + \frac{v_n}{4}h_{n-1}\right]\lambda_n = 0 \end{array} \right.$$

$$\text{où } \gamma_i = h_i/h_{i-1} \quad i=2,\dots,n-1$$

$$\alpha_i = \rho h_i^3/2 \quad i=1,\dots,n-1$$

Cas noeuds équidistants

Posons

$$h = h_i, \alpha = \alpha_i \quad i=1,2,\dots,n-1$$

$$\delta_i = h\lambda_i \quad i=1,2,\dots,n$$

Le système à résoudre d'après (4.8) est

$$(4.9) \quad \begin{cases} B_1 \Lambda_1 + C \Lambda_2 = D_1 \\ A \Lambda_{i-1} + B_i \Lambda_i + C \Lambda_{i+1} = D_i \\ A \Lambda_{n-1} + B_n \Lambda_n = D_n \end{cases} \quad i=2,\dots,n-1$$

où

$$B_1 = \begin{bmatrix} 6+\alpha & 3 \\ 3 & 2+\frac{\nu_1}{2}h \end{bmatrix}, \quad B_i = \begin{bmatrix} 12+\alpha & 0 \\ 0 & 4+\frac{\nu_i}{2}h \end{bmatrix} \quad i=2,\dots,n-1$$

$$B_n = \begin{bmatrix} 6+\alpha & -3 \\ -3 & 2+\frac{\nu_n}{2}h \end{bmatrix} \quad C = \begin{bmatrix} -6 & 3 \\ -3 & 1 \end{bmatrix} \quad A = \begin{bmatrix} -6 & -3 \\ 3 & 1 \end{bmatrix}$$

$$\Lambda_i = [y_i \quad h\lambda_i]^T, \quad D_i = [\alpha z_i \quad 0]^T \quad i=1,2,\dots,n,$$

La matrice associée à (4.9) est symétrique par blocs, car $A=C^T$. La résolution de ce système sera faite par l'élimination simple de Gauss par bloc.

$$(4.10) \begin{cases} U_1 = B_1 & b_1 = D_1 \\ U_i = B_i - AU_{i-1}^{-1}c & b_i = D_i - AU_{i-1}^{-1} b_{i-1} \quad i=2, \dots, n-1 \\ U_n = B_n - AU_{i-1}^{-1}c & b_n = D_n - AU_{n-1}^{-1} b_{n-1} \end{cases}$$

Alors

$$(4.11) \begin{cases} \Lambda_n = U_n^{-1} b_n \\ \Lambda_i = U_i^{-1}(b_i - C\Lambda_{i+1}) \quad i=n-1, n-2, \dots, 1 \end{cases}$$

Remarque

B_1 est symétrique et donc U_1^{-1} aussi. Par récurrence on peut montrer que U_i est symétrique.

III.5. FONCTIONS SPLINES CUBIQUES AVEC POIDS

L'alternative que développe Cinquin [19] pour trouver des fonctions splines cubiques avec des propriétés voisines de celles de splines sous tension provient de la minimisation de la norme L_2 de la dérivée seconde avec poids.

Si $a = t_0 < t_1 < \dots < t_n < t_{n+1} = b$, posons

$$w(t) = w_i \quad \forall t \in]t_i, t_{i+1}[\quad i=0, 1, \dots, n \quad \text{et} \quad w_0 = w_n = 0 .$$

$$E_3(x) = \int_a^b w(t) [x''(t)]^2 dt = \sum_{i=1}^{n-1} w_i \int_{t_i}^{t_{i+1}} [x''(t)]^2 dt$$

On voit intuitivement que si l'on minimise cette fonctionnelle, une valeur grande du poids w_k (les autres restant fixes) contraint la dérivée seconde à être petite et la courbe à tendre vers le morceau de droite dans l'intervalle $[t_k, t_{k+1}]$.

Cinquin a utilisé ce modèle pour lisser des données monotones. Plus tard, Salkaukas [99] l'a appliqué pour l'interpolation.

5.1. Interpolation

Nous revenons au problème

$$(5.1) \left\{ \begin{array}{l} \text{minimiser } E_3(x) \\ \text{avec } x \in H^2[a,b] \text{ vérifiant } x(t_i) = y_i \quad i=1,2,\dots,n \end{array} \right.$$

Cinquin reprend le cadre fonctionnel général proposé par Laurent [56] pour déterminer l'existence, l'unicité et la caractérisation de la solution.

Proposition 5.1

Il existe une solution unique σ du problème (5.1). Cette solution est caractérisée par

- (i) σ est un polynôme de degré 3 sur $[t_i, t_{i+1}] \quad i=1,2,\dots,n-1$
- (ii) σ est un polynôme de degré 1 sur $[a, t_1]$ et $[t_n, b]$
- (iii) $\sigma \in C^1[a,b]$ et $w_{i-1} \sigma''(t_i^-) = w_i \sigma''(t_i^+) \quad i=1,2,\dots,n$
- (iv) $\sigma(t_i) = y_i \quad i=1,2,\dots,n \quad \square$

D'après la représentation (4.2) de la section 4.1 pour les splines cubiques, il faut déterminer les valeurs $\lambda_i = \sigma'(t_i) \quad i=1,2,\dots,n$. Pour cela on utilise la continuité de $w(t) \sigma''(t)$ aux noeuds, ce qui nous amène au système tridiagonal :

$$(3.2) \left\{ \begin{array}{l} 2\lambda_1 + \lambda_2 = 3m_1 \\ \frac{w_{i-1}}{h_{i-1}} \lambda_{i-1} + 2 \left[\frac{w_{i-1}}{h_{i-1}} + \frac{w_i}{h_i} \right] \lambda_i + \frac{w_i}{h_i} \lambda_{i+1} = 3 \left[\frac{w_{i-1}}{h_{i-1}} m_{i-1} + \frac{w_i}{h_i} m_i \right] \\ \lambda_{n-1} + 2\lambda_n = 3m_{n-1} \end{array} \right. \quad i=2,3,\dots,n-1$$

Là encore, l'élimination simple de Gauss est stable, car la matrice est à diagonale dominante.

Sur le comportement asymptotique de ces splines nous avons le résultat suivant .

Proposition 5.2

La fonction σ solution du problème (5.1) tend vers le segment de droite qui relie (t_i, y_i) et (t_{i+1}, y_{i+1}) lorsque w_i tend vers l'infini et les autres poids restent fixes. \square

L'étude du signe des dérivées permet de conclure à Cinquin que pour w_i suffisamment grand on peut préserver la monotonie. Par contre, il n'existe pas de conclusion analogue pour la convexité.

Le modèle le plus général qui correspond à la configuration stable d'une tige flexible qui passe à travers certains points est donné par la spline appelée **non linéaire** ou "**elastica**" (voir [50],[61],[64]). Ceci s'exprime comme le minimum de la fonctionnelle:

$$E(x) = \int_a^b \frac{[x''(t)]^2}{[1 + (x'(t))^2]^{5/2}} dt \quad \text{avec } x(t_i) = y_i \quad i=1,2,\dots,n$$

On sait que pour des valeurs modérées de la dérivée première on obtient la fonction spline cubique naturelle ; et que certaines linéarisations conduisent à la spline hyperbolique sous tension.

La raison pour laquelle on évoque cette intégrale, dans ce paragraphe, est qu'on peut choisir les poids w_i de la forme suivante :

$$w_i = (1 + \varphi_i^2)^{-5/2}$$

où φ_i est une approximation discrète de $x'(t_i)$.

On peut prendre par exemple

$$\varphi_i = (y_{i+1} - y_i) / h_i$$

Ainsi l'intégrale $\int_a^b w(t) [x''(t)]^2 dt$ "ressemble" à $E(x)$.

5.2. Lissage

Nous allons étudier le problème

$$(5.3) \left\{ \begin{array}{l} \text{minimiser } E_{\mathcal{J}}(x) + \rho J(x) \\ x \in H^2 \end{array} \right.$$

En utilisant les résultats de Laurent sur les fonctions splines de lissage on aboutit à la

Proposition 5.3

Il existe une solution unique σ pour (5.3). Cette solution est caractérisée par

- (i) σ est un polynôme de degré 3 sur $[t_i, t_{i+1}]$ $i=1,2,\dots,n-1$
- (ii) σ est un polynôme de degré 1 sur $[a, t_1], [t_n, b]$
- (iii) $\sigma \in C^1[a, b]$ et $w_{i-1} \sigma''(t_i^-) = w_i \sigma''(t_i^+)$ $i=1,2,\dots,n$
- (iv) $w_i \sigma'''(t_i^+) - w_{i-1} \sigma'''(t_i^-) = \rho(z_i - \sigma(t_i))$ $i=1,2,\dots,n$ \square

L'algorithme du transport de relations, présenté par Laurent [56] avec les modifications faites par Paihua [87], est appliqué par Cinquin pour le calcul des y_i et λ_i .

Par contre, nous allons utiliser les conditions de continuité (iii) et de lissage (iv) pour écrire un système tridiagonal par blocs 2×2 et le résoudre par la méthode de Gauss.

$$\begin{cases}
 (\alpha_1 + 6w_1)y_1 + 3w_1h_1\lambda_1 - 6w_1y_2 + 3w_1h_1\lambda_2 = \alpha_1z_1 \\
 3y_1 + 2h_1\lambda_1 - 3y_2 + h_1\lambda_2 = 0 \\
 -6w_{i-1}\gamma_i^3y_{i-1} - 3w_{i-1}\gamma_i^2h_i\lambda_{i-1} + [\alpha_i + 6(w_{i-1}\gamma_i^3 + w_i)]y_i + \\
 3[w_{i-1} - w_i\gamma_i^2]h_i\lambda_i - 6w_iy_{i+1} + 3w_ih_i\lambda_{i+1} = \alpha_iz_i \\
 (5.4) \begin{cases}
 3w_{i-1}\gamma_i^2y_{i-1} + w_{i-1}\gamma_ih_i\lambda_i + 3[-w_{i-1}\gamma_i^2 + w_i]y_i + \\
 2[w_{i-1}\gamma_i + w_i]h_i\lambda_i - 3w_iy_{i+1} + w_ih_i\lambda_{i+1} = 0 \\
 i=2, \dots, n-1
 \end{cases} \\
 -6w_{n-1}y_{n-1} - 3w_{n-1}h_{n-1}\lambda_{n-1} + (\alpha_{n-1} + 6w_{n-1})y_n - \\
 3w_{n-1}h_{n-1}\lambda_n = \alpha_{n-1}z_n \\
 3y_{n-1} + h_{n-1}\lambda_{n-1} - 3y_n - 2h_{n-1}\lambda_n = 0
 \end{cases}$$

où $\gamma_i = h_i / h_{i-1}$ $i=2, \dots, n-1$, $\alpha_i = \rho h_i^3 / 2$ $i=1, \dots, n-1$

Cas noeuds équidistants

Posons

$$h = h_i, \alpha = \alpha_i \quad i=1, 2, \dots, n-1$$

$$\delta_i = h\lambda_i \quad i=1, 2, \dots, n$$

Alors nous devons résoudre le système

$$(5.5) \begin{cases}
 B_1\Lambda_1 + w_2C\Lambda_2 = D_1 \\
 w_{i-1}A\Lambda_{i-1} + B_i\Lambda_i + w_iC\Lambda_{i+1} = D_i \quad i=2, \dots, n-1 \\
 w_{n-1}A\Lambda_{n-1} + B_n\Lambda_n = D_n
 \end{cases}$$

où

$$B_i = \begin{bmatrix} \alpha + 6(w_i + w_{i-1}) & 3(w_i - w_{i-1}) \\ 3(w_i - w_{i-1}) & 2(w_i + w_{i-1}) \end{bmatrix} \quad i=1,2,\dots,n$$

$$C = \begin{bmatrix} -6 & 3 \\ -3 & 1 \end{bmatrix}, \quad A = C^T$$

$$\Lambda_i = [y_i \ \delta_i]^T, \quad D_i = [\alpha z_i \ 0]^T \quad i=1,2,\dots,n$$

Notons

$$\hat{i} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad P = \begin{bmatrix} 6 & 3 \\ 3 & 2 \end{bmatrix} \quad Q = \begin{bmatrix} 6 & -3 \\ -3 & 2 \end{bmatrix}$$

donc

$$\begin{cases} B_1 = \alpha \hat{i} + w_1 P \\ B_i = \alpha \hat{i} + w_i P + w_{i-1} Q \\ B_n = \alpha \hat{i} + w_{n-1} Q \end{cases} \quad i=2,\dots,n-1$$

Pour l'élimination par bloc supposons que nous ayons une relation de la forme

$$U_i \Lambda_i + w_i C \Lambda_{i+1} = b_i \quad (\text{on part avec } U_1 = B_1, \ b_1 = D_1)$$

après quelques calculs simples on peut montrer qu'on passe à

$$U_{i+1} \Lambda_{i+1} + w_i C \Lambda_{i+1} = b_{i+1}$$

par les formules

$$(5.6) \quad \left\{ \begin{array}{l} U_{i+1} = (\alpha i + w_{i+1} P + w_i Q) - w_i^2 A U_i^{-1} C \\ b_{i+1} = D_{i+1} - w_i A U_i^{-1} b_i \end{array} \right. \quad i=1, \dots, n-1$$

On obtient enfin

$$(5.7) \quad \Lambda_n = U_n^{-1} b_n$$

et on remonte avec

$$(5.8) \quad \Lambda_i = U_i^{-1} [b_i - w_i C \Lambda_{i+1}] \quad i=n-1, n-2, \dots, 1$$

Remarque

U_1 est symétrique donc U_1^{-1} aussi. On peut démontrer par récurrence que U_{i+1} est symétrique à partir de la symétrie de U_i^{-1} et du fait que $A = C^T$. Le calcul peut être donc optimisé.

III.6. FONCTIONS SPLINES CUBIQUES PENALISEES

Dans les paragraphes précédents nous avons vu deux manières différentes d'appliquer la tension aux fonctions splines :

- la première, dans les intervalles (splines hyperboliques, rationnelles et cubiques avec poids),
- la deuxième, dans les noeuds (ψ -splines).

Nous allons continuer dans cette dernière voie, en faisant intervenir une pénalisation sur la dérivée première aux abscisses. Nous proposons de minimiser la fonctionnelle :

$$E_4(x) = \int_a^b [x''(t)]^2 dt + \sum_{i=1}^n r_i [x'(t_i) - \varphi x_i]^2$$

avec

$$\varphi x_i = \begin{cases} [x(t_2) - x(t_1)] / h_1 & i=1 \\ \{ [x(t_i) - x(t_{i-1})] (h_i/h_{i-1}) + [x(t_{i+1}) - x(t_i)] (h_{i-1}/h_i) \} / (h_{i-1} + h_i) & i=2,3,\dots,n-1 \\ [x(t_n) - x(t_{n-1})] / h_{n-1} & i=n \end{cases}$$

et $r_i > 0$ ($i=1,2,\dots,n$).

Ce critère de minimisation conduit encore à des fonctions splines polynomiales cubiques. L'expression de pénalité apparaît plus "naturelle" que celle proposée par Nielson ; cette dernière présente l'inconvénient d'aboutir à des fonctions ayant des paliers aux noeuds pour des valeurs élevées du paramètre de tension. Dans notre cas, à la limite, les dérivées aux noeuds convergent vers une approximation locale d'ordre 2.

6.1. Interpolation

On se place dans le cadre fonctionnel défini par Laurent dans [56].

Soient

$$X = H^2[a,b], \quad Y = H^0[a,b] \times \mathbb{R}^n \quad \text{où} \quad H^0[a,b] = L^2[a,b].$$

On munit X du produit scalaire

$$\langle g_1, g_2 \rangle = \sum_{i=0}^2 \int_a^b g_1^{(i)} g_2^{(i)}(t) dt$$

et Y du produit scalaire

$$\langle [g_1, z_1], [g_2, z_2] \rangle = \int_a^b g_1(t) g_2(t) dt + \sum_{i=1}^n r_i z_1^i z_2^i$$

Les normes associées sont notées respectivement $\| \cdot \|$ et $\| \cdot \|_1$.

On considère les applications

$$T : X \rightarrow Y \text{ définie par } T(x) = [x''; x'(t_1) - \varphi x_1, \dots, x'(t_n) - \varphi x_n]$$

$$K_i : X \rightarrow \mathbb{R} \text{ telle que } K_i(x) = x(t_i) \quad (i=1,2,\dots,n)$$

$$A : X \rightarrow \mathbb{R}^n \text{ définie par } A(x) = [k_1(x), \dots, k_n(x)]$$

On étudie le problème

$$(6.1) \quad \begin{cases} \text{minimiser } \|Tx\|^2 \\ \text{avec les contraintes } x(t_i) = y_i \quad i=1,2,\dots,n. \end{cases}$$

Proposition 6.1

Le problème (6.1) a une solution unique dans X .

Preuve

L'espace nul de T , $\mathcal{N}(T) = \{x \in X / Tx=0\}$, est P_1 , l'espace des polynômes de degré 1. Il est donc de dimension finie. Alors $\mathcal{N}(T) + \mathcal{N}(A)$ est fermé dans X . De plus, $\mathcal{N}(T) \cap \mathcal{N}(A) = \{0\}$ si $n \geq 2$.

La preuve est établie d'après le théorème 4.4.2 de [56,p.196]. \square

On va donner tout de suite la forme explicite de cette solution. On définit l'espace S_p des **fonctions splines cubiques pénalisées** par

$$(6.2) \quad S_p = \{ \sigma \in X / \langle T\sigma, Tx \rangle = 0 \quad \forall x \in \mathcal{N}(A) \}$$

Cette caractérisation pour $\sigma \in S$ est équivalente à

(6.3) Il existe $[\delta_1, \delta_2, \dots, \delta_n] \in \mathbb{R}^n$ tel que

$$\langle T\sigma, Tx \rangle = \sum_{i=1}^n \delta_i x(t_i) \quad \forall x \in X$$

ou encore, à

(6.4) il existe $[\delta_1, \delta_2, \dots, \delta_n] \in \mathbb{R}^n$ tel que $T^*T\sigma \in \text{Im}(A^*)$

où T^* est l'application transposée de T .

Comme $T^*T\sigma = \sum_{i=1}^n \delta_i k_i$ et $\text{Im}(A^*) = N(T)^\perp$ on déduit que

$$(6.5) \quad \sum_{i=1}^n \delta_i = 0 \quad \text{et} \quad \sum_{i=1}^n \delta_i t_i = 0$$

Soit $x \in X$; son développement de Taylor avec reste intégral s'écrit

$$(6.6) \quad x(t) = x(a) + (t-a)x'(a) + \int_a^b (t-\xi)_+ x''(\xi) d\xi$$

$$\text{où} \quad (t-\xi)_+ = \begin{cases} t-\xi & \text{si } t-\xi \geq 0 \\ 0 & \text{sinon} \end{cases}$$

En utilisant les conditions (6.4), (6.5) et le développement (6.6) on obtient :

$$\langle Ts, Tx \rangle = \langle T^*T\sigma, x \rangle = \left\langle \sum_{i=1}^n \delta_i k_i, x \right\rangle$$

$$\begin{aligned}
 &= \sum_{i=1}^n \delta_i x(t_i) \\
 &= \int_a^b \left[\sum_{i=1}^n \delta_i (t_i - \xi)_+ \right] x''(\xi) d\xi \\
 &= \int_a^b \sigma''(t) x''(t) dt + \sum_{i=1}^n r_i [\sigma'(t_i) - \varphi_{\sigma i}] [x'(t_i) - \varphi x_i]
 \end{aligned}$$

On a donc

$$(6.7) \quad \sigma''(\xi) = \sum_{i=1}^n (\delta_i - \Psi_{\sigma i}) (t_i - \xi)_+ - \sum_{i=1}^n r_i (\sigma'(t_i) - \varphi_{\sigma i}) (t_i - \xi)_+^0$$

Ici on note

$$(t_i - \xi)_+^0 = \begin{cases} 1 & \text{si } t_i - \xi \geq 0 \\ 0 & \text{sinon} \end{cases}$$

Les valeurs $\Psi_{\sigma i}$ sont données par (6.8) :

$$\Psi_{\sigma 1} = \frac{r_1}{h_1} [\sigma'(t_1) - \varphi_{\sigma 1}] + \frac{r_2 h_2}{(h_1 + h_2) h_1} [\sigma'(t_2) - \varphi_{\sigma 2}]$$

$$\Psi_{\sigma 2} = -\frac{r_1}{h_1} [\sigma'(t_1) - \varphi_{\sigma 1}] + \frac{r_2}{h_1 + h_2} \left[\frac{h_1}{h_2} - \frac{h_2}{h_1} \right] [\sigma'(t_2) - \varphi_{\sigma 2}] + \frac{r_3 h_3}{(h_2 + h_3) h_2} [\sigma'(t_3) - \varphi_{\sigma 3}]$$

$$\begin{aligned}
 \Psi_{\sigma i} = & -\frac{r_{i-1}}{(h_{i-2} + h_{i-1}) h_{i-1}} [\sigma'(t_{i-1}) - \varphi_{\sigma i-1}] + \frac{r_i}{(h_{i-1} + h_i) \left[\frac{h_{i-1}}{h_i} - \frac{h_i}{h_{i-1}} \right]} [\sigma'(t_i) - \varphi_{\sigma i}] + \\
 & \frac{r_{i+1}}{(h_i + h_{i+1}) h_i} [\sigma'(t_{i+1}) - \varphi_{\sigma i+1}] \quad i=3,4,\dots,n-2
 \end{aligned}$$

$$\Psi_{\sigma_{n-1}} = -\frac{r_{n-2} h_{n-2}}{(h_{n-3}+h_{n-2})h_{n-3}} [\sigma'(t_{n-2}) - \varphi_{\sigma_{n-2}}] + \frac{r_{n-1}}{h_{n-1}+h_{n-2}} \left[\frac{h_{n-2}}{h_{n-1}} - \frac{h_{n-1}}{h_{n-2}} \right] \\ [\sigma'(t_{n-1}) - \varphi_{\sigma_{n-1}}] + \frac{r_n}{h_{n-1}} [\sigma'(t_n) - \varphi_{\sigma_n}]$$

$$\Psi_{\sigma_n} = -\frac{r_{n-1} h_{n-2}}{(h_{n-2}+h_{n-1})h_{n-1}} [\sigma'(t_{n-1}) - \varphi_{\sigma_{n-1}}] - \frac{r_n}{h_{n-1}} [\sigma'(t_n) - \varphi_{\sigma_n}]$$

Si $\xi > t_n$ alors $\sigma''(\xi) = 0$, et si $\xi < t_1$ alors

$$\sigma''(\xi) = \sum_{i=1}^n (\delta_i - \Psi_{\sigma_i}) (t_i - \xi)_+ - \sum_{i=1}^n r_i (\sigma'(t_i) - \varphi_{\sigma_i})$$

La condition (6.5) et d'autres calculs permettent de conclure que le terme droit de cette expression s'annule.

$$\text{On a aussi } \sigma''(t_i^+) - \sigma''(t_i^-) = r_i (\sigma'(t_i) - \varphi_{\sigma_i}) \quad i=1,2,\dots,n$$

Finalement on a la caractérisation suivante :

Proposition 6.2

Une condition nécessaire et suffisante pour que $\sigma \in X$ soit la fonction spline pénalisée solution du problème (6.1) est que les cinq conditions suivantes soient vérifiées :

- (i) σ est formée de morceaux de polynômes de degré 3 dans chacun des intervalles $[a_1, t_1[$, $]t_i, t_{i+1}[$ ($i=1, \dots, n-1$) et $]t_n, b[$
- (ii) $\sigma'(t_i^-) = \sigma'(t_i^+)$ $i=1, 2, \dots, n$
- (iii) $\sigma''(t_i^+) - \sigma''(t_i^-) = r_i (\sigma'(t_i) - \varphi_{\sigma_i})$ $i=1, 2, \dots, n$
- (iv) $\sigma''(t_1^-) = \sigma''(t_n^+) = 0$
- (v) $\sigma(t_i) = y_i$ $i=1, 2, \dots, n$

Remarque

La condition (iv) exprime le fait que σ est un polynôme de degré 1 dans $[a, t_1[$ et $]t_n, b]$.

6.1.1. Calcul des dérivées aux noeuds

Dans chaque intervalle $[t_i, t_{i+1}]$ ($i=1,2,\dots,n-1$) la spline cubique σ peut être représentée par (6.9) :

$$\sigma(t) = y_i + \lambda_i(t-t_i) + \frac{m_i - \lambda_i}{h_i}(t-t_i)^2 + \frac{\lambda_{i+1} - 2m_i + \lambda_i}{h_i^2}(t-t_i)^2(t-t_{i+1})$$

Les valeurs des inconnues $\lambda_i = \sigma'(t_i)$ sont calculées en utilisant les conditions (iii) et (iv) de la proposition 6.2. On obtient ainsi le système tridiagonal:

$$(6.10) \left\{ \begin{array}{l} 2\left(\frac{1}{h_1} + \frac{r_1}{4}\right)\lambda_1 + \frac{\lambda_2}{h_1} = \left(\frac{3}{h_1} + \frac{r_1}{2}\right)m_1 \\ \frac{\lambda_{i-1}}{h_{i-1}} + 2\left(\frac{1}{h_{i-1}} + \frac{1}{h_i} + \frac{r_i}{4}\right)\lambda_i + \frac{\lambda_{i+1}}{h_i} = \left(\frac{3}{h_{i-1}} + \frac{h_i r_i}{2(h_{i-1} + h_i)}\right)m_{i-1} + \\ \quad \left(\frac{3}{h_i} + \frac{h_{i-1} r_i}{2(h_{i-1} + h_i)}\right)m_i \\ \lambda_{n-1} + 2\left(\frac{1}{h_{n-1}} + \frac{r_n}{4}\right)\lambda_n = \left(\frac{3}{h_{n-1}} + \frac{r_n}{4}\right)m_{n-1} \end{array} \right. \quad i=2,3,\dots,n-1$$

On remarque que la matrice du système linéaire (6.10) est la même que celle associée aux ν -splines ; ce qui change, c'est le terme indépendant.

Les équations du système (6.10) deviennent pour $i=2, \dots, n-1$

$$(6.12) \quad \lambda_{i-1} + 2 \eta(r) \lambda_i + \lambda_{i+1} = \theta_i = [\eta(r)+1] (y_{i+1} - y_i) / h$$

$$\text{où } \eta = \eta(r) = 2 + \frac{rh}{4}$$

On parvient alors au même type d'équations que pour les splines hyperboliques et rationnelles mais ce qui change c'est le terme $\eta(r)$. On voit que $\eta(0)=2$ et que $\eta(r) \rightarrow \infty$ lorsque $r \rightarrow \infty$. Cette remarque suggère le terme de pénalisation de E_4 .

Signalons qu'on peut chercher des conditions de bout de la forme

$$(6.13) \quad \begin{cases} a\lambda_1 + b\lambda_2 = \theta_1 \\ b\lambda_{n-1} + a\lambda_n = \theta_n \end{cases}$$

pour simplifier la résolution du système linéaire, de manière analogue à celle que nous avons utilisée au paragraphe 2.2.3.

On a (6.12) exacte pour $\tilde{\mathcal{P}}_4 = (p(t) = \sum_{i=0}^4 \alpha_i t^i / \alpha_3 = 0)$ si $r > 0$.

Rappelons rapidement le schéma suivi :

On a $\mu = -\eta + (\eta^2 - 1)^{1/2}$ et le rapport $b/a = \mu$

Dans la triangularisation, en partant de la relation

$$(-1/\mu) \lambda_1 + \lambda_2 = c_1 = \theta_1$$

on aura à chaque fois

$$(-1/\mu) \lambda_i + \lambda_{i+1} = c_i$$

avec $c_i = \mu \cdot c_{i-1} + \theta_i$ pour $i=2, \dots, n-1$

Pour la remontée on obtient

$$\lambda_n = \frac{1}{2(\eta^2 - 1)^{1/2}} [\theta_n + \mu \cdot c_{n-1}]$$

et $\lambda_i = \mu(\lambda_{i+1} - c_i) \quad i = n-1, n-2, \dots, 1$

Le choix de θ_1 et θ_n sera encore

$$\theta_1 = \left(\sum_{i=1}^5 \delta_i y_i \right) / h \quad \text{et} \quad \theta_n = \left(\sum_{i=1}^5 \delta_i y_{n+1-i} \right) / h$$

où

$$\delta_1 = (-3 + 25/\mu) / 12$$

$$\delta_2 = -(10 + 48/\mu) / 12$$

$$\delta_3 = (18 + 36/\mu) / 12$$

$$\delta_4 = -(6 + 16/\mu) / 12$$

$$\delta_5 = (1 + 3/\mu) / 12$$

Remarque

Si $r=0$, donc $\eta = -2 + \sqrt{3}$, on retrouve les valeurs données par Laurent dans [58].

6.1.2. Propriétés des splines pénalisées

On note au passage que si l'on divise les équations (6.10) par les r_i respectifs, on obtient

$$\lim_{r_i \rightarrow \infty} \lambda_i = \theta_i \quad i = 1, 2, \dots, n$$

où

$$\theta_i = \begin{cases} m_1 & i = 1 \\ u_i m_{i-1} + v_i m_i & i = 2, \dots, n-1 \\ m_{n-1} & i = n \end{cases}$$

Lemme 6.3

La matrice A du système (6.11) est définie positive.

Preuve

On pose $u_1 = v_n = 0$ et $u_i = v_i = 1$.

Soit $x = [x_1, \dots, x_n]^T$ un vecteur quelconque de \mathbb{R}^n . Alors

$$x^T A x = \frac{1}{2} \sum_{i=1}^n [2p_i + 3 - v_{i-1} + u_{i+1}] x_i^2 + \frac{1}{2} \sum_{i=1}^{n-1} [v_i + u_{i+1}] [x_i + x_{i+1}]^2$$

Comme $u_i \geq 0$ et $v_i \leq 1$ on conclut que $x^T A x > 0$. \square

Notons

$$I = \{1, 2, \dots, n\}$$

$$D = \text{diag}(\lambda_1 - \theta_1, \dots, \lambda_n - \theta_n)$$

$$P = \left(\frac{\partial \lambda_j}{\partial p_i} \right)_{i,j}$$

$$\frac{\partial p_j}{\partial r_j} \quad (i,j) \in I \times I$$

$$M = \left(\frac{\partial(\lambda_i - \theta_i)}{\partial r_j} \right)_{i,j}$$

Lemme 6.4

La matrice M est semi-définie négative.

Preuve

Considérons à nouveau la relation

$$u_i \lambda_{i-1} + (2+p_i) \lambda_i + v_i \lambda_{i+1} = (3+p_i) [u_i m_{i-1} + v_i m_i]$$

Et prenons la dérivée partielle par rapport à p_j , alors

$$u_i \frac{\partial \lambda_{i-1}}{\partial p_j} + (2+p_i) \frac{\partial \lambda_i}{\partial p_j} + v_i \frac{\partial \lambda_{i+1}}{\partial p_j} = (\theta_i - \lambda_i) \frac{\partial p_i}{\partial p_j}$$

Le terme à droite est nul si $i \neq j$.

Alors on a la relation matricielle $AP = -D$

Maintenant,

$$\begin{aligned} \bar{M} &= \left(\frac{\partial(\lambda_i - \theta_i)^2}{\partial p_j} \right) = 2(\lambda_i - \theta_i) \frac{\partial(\lambda_i - \theta_i)}{\partial p_j} \\ &= 2 DP \end{aligned}$$

mais $P = -A^{-1} D$, c'est-à-dire $\bar{M} = -2 D A^{-1} D$.

D'après le lemme précédent A est définie positive et donc A^{-1} aussi. Alors \bar{M} est semi-définie négative. On a pris ici les dérivées partielles par rapport à p_j , lequel est un multiple de r_j , d'où finalement on a le résultat. \square

Proposition 6.4

Si σ est la solution du problème (6.1) alors

$$\sum_{i=1}^n r_i (\sigma(t_i) - \varphi_{\sigma i})^2 \leq K$$

où K est une constante indépendante des r_i .

Preuve

Soit $I_y = \{x \in X / x(t_i) = y_i \quad i=1, \dots, n\}$, alors

$$\|T\sigma\|^2 = \min_{x \in I_y} \|Tx\|^2$$

On a évidemment les inégalités

$$\sum_{i=1}^n r_i (\sigma(t_i) - \varphi_{\sigma i})^2 \leq \langle\langle T\sigma, T\sigma \rangle\rangle \leq \langle\langle Tx, Tx \rangle\rangle$$

pour tout $x \in I_y$.

Si l'on choisit $x \in Y$ tel que $x'(t_i) - \varphi_{xi} = 0$, $i \in I$, alors

$$\langle Tx, Tx \rangle = \int_a^b [x''(t)]^2 dt.$$

Il est clair que le terme à droite de l'expression (6.14) ne dépend pas des r_i . \square

Il reste à donner le résultat principal de cette section.

Proposition 6.5.

Supposons $k \in I$. Si les paramètres r_i restent fixes pour $i \in I \setminus \{k\}$, alors

$(\lambda_k - \varphi_{\sigma k})^2$ est une fonction décroissante de r_k telle que

$$\lim_{r_k \rightarrow \infty} (\lambda_k - \varphi_{\sigma k})^2 = 0$$

Preuve

D'après la proposition précédente

$$r_k (\lambda_k - \varphi_{\sigma k})^2 \leq \text{constante}$$

alors pour $r_k \rightarrow \infty$ on a la convergence vers zéro.

La matrice $M = \left(\frac{\partial^2 [\lambda_i - \theta_i]}{\partial r_j} \right)$ est semi-définie négative.

Si l'on prend $x = [0, \dots, 0, 1, 0, \dots, 0]$, la valeur 1 correspond à la k -ième

coordonnée, alors $x^T M x = \frac{\partial^2 [\lambda_k - \theta_k]}{\partial r_k} \leq 0$.

Ceci entraîne que $(\lambda_k - \theta_k)^2$ est décroissante. \square

6.1.3. Sur la préservation de la convexité et de la monotonie

Considérons le polynôme cubique $q_i(t)$ tel que

$$q_i(t_j) = y_j \quad \text{et} \quad q'_i(t_j) = \lambda_j \quad j=i, i+1$$

alors pour $t \in [t_i, t_{i+1}]$ on a :

$$q_i(t) = y_i + \lambda_i(t-t_i) + \frac{m_i - \lambda_i}{h_i} (t-t_i)^2 + \frac{\lambda_{i+1} - 2m_i + \lambda_i}{h_i^2} (t-t_i)^2 (t-t_{i+1})$$

Les splines cubiques de Nielson , avec poids ou pénalisées σ peuvent être représentées par $\sigma(t) = q_i(t)$ pour $t_i \leq t \leq t_{i+1}$.

Alors la dérivée seconde de $q_i(t)$ s'écrit pour $t \in]t_i, t_{i+1}[$:

$$(6.15) \quad q''_i(t) = \frac{2}{h_i} [-2\lambda_i - \lambda_{i+1} + 3m_i] + \frac{6}{h_i^2} [\lambda_i + \lambda_{i+1} - 2m_i] (t-t_i)$$

Comme $q''_i(t)$ est un morceau de droite $]t_i, t_{i+1}[$, la condition pour qu'il n'existe pas de points d'inflexion dans cet intervalle est que $q''_i(t_i^+) \cdot q''_i(t_{i+1}^-) > 0$. Alors on découle immédiatement la :

Proposition 6.6.

Le polynôme $q_i(t)$ n'a pas de points d'inflexion dans l'intervalle $]t_i, t_{i+1}[$ si et seulement si :

$$(6.16) \quad 2\lambda_i + \lambda_{i+1} < 3m_i < \lambda_i + 2\lambda_{i+1} \quad (\text{convexité})$$

ou

$$(6.17) \quad 2\lambda_i + \lambda_{i+1} > 3m_i > \lambda_i + 2\lambda_{i+1} \quad (\text{concavité}) \quad \square$$

Voyons le comportement à la limite des différentes splines cubiques sous tension.

- Pour les ν -splines, si ν_i et ν_{i+1} tendent vers l'infini, alors λ_i et λ_{i+1} tendent respectivement vers zéro. on a donc

$$\lim_{\nu_i, \nu_{i+1} \rightarrow \infty} \sigma''(t_i^+) = 6 \frac{m_i}{h_i} \quad \text{et} \quad \lim_{\nu_i, \nu_{i+1} \rightarrow \infty} \sigma''(t_{i+1}^-) = -6 \frac{m_i}{h_i}$$

Ce qui signifie que la convexité (concavité) ne sera pas respectée dans l'intervalle $[t_i, t_{i+1}]$.

- Pour les splines hyperboliques ou rationnelles sous tension on démontre que

$$\lim_{r_i, r_{i+1} \rightarrow \infty} \lambda_i = \frac{1}{2} [m_i + m_{i-1}],$$

par contre pour les cubiques avec poids on a:

$$\lim_{r_i \rightarrow \infty} \lambda_i = \lim_{r_{i+1} \rightarrow \infty} \lambda_{i+1} = m_i$$

En effet, l'équation de raccordement au noeud t_i s'écrit

$$\frac{w_{i-1} h_i}{w_i h_{i-1}} \lambda_{i-1} + 2 \left[1 + \frac{w_{i-1} h_i}{w_i h_{i-1}} \right] \lambda_i + \lambda_{i+1} = 3 \left[\frac{w_{i-1} h_i}{w_i h_{i-1}} m_{i-1} + m_i \right]$$

Si l'on prend la limite $w_i \rightarrow \infty$, w_{i-1} restant fixe, alors

$$(2\lambda_i + \lambda_{i+1}) \mapsto 3 m_i$$

Maintenant, si l'on considère l'équation de continuité au noeud t_{i+1} ,

$$\lambda_{i+2} + 2 \left[1 + \frac{w_{i+1} h_i}{w_i h_{i+1}} \right] \lambda_{i+1} + \frac{w_{i+1} h_i}{w_i h_{i+1}} \lambda_{i+2} = 3 \left[m_i + \frac{w_{i+1} h_i}{w_i h_{i+1}} m_{i+1} \right]$$

et on prend la limite lorsque $w_i \rightarrow \infty$, w_{i+1} restant fixe, alors

$$\lambda_i + 2 \lambda_{i+1} \rightarrow 3 m_i$$

On déduit que $\lim_{w_i \rightarrow \infty} \lambda_i = \lim_{w_i \rightarrow \infty} \lambda_{i+1} = m_i$.

Autrement dit la fonction σ se réduit à la droite qui relie (t_i, y_i) et (t_{i+1}, y_{i+1}) .

La conséquence que l'on peut tirer est que $\sigma''(t_i^+) \rightarrow 0$ et $\sigma''(t_{i+1}^-) \rightarrow 0$, donc l'étude du signe de σ'' n'est pas possible. Cependant dans les exemples numériques on a pu constater la préservation de la convexité par ce type de splines.

• Examinons ce qui se passe pour les splines pénalisées. D'après la proposition (6.5)

$$\lim_{r_i \rightarrow \infty} \lambda_i = \frac{h_i m_{i-1} + h_{i-1} m_i}{h_i + h_{i-1}}, \quad \lim_{r_{i+1} \rightarrow \infty} \lambda_{i+1} = \frac{h_{i+1} m_i + h_i m_{i-1}}{h_{i+1} + h_i}$$

Dans ces conditions les inégalités (6.16) sont équivalentes à

$$(6.18) \quad \frac{h_i + h_{i+1}}{2(h_i + h_{i-1})} (m_i - m_{i-1}) < m_{i+1} - m_i < 2 \left(\frac{h_i + h_{i+1}}{h_i + h_{i-1}} \right) (m_i - m_{i-1})$$

Considérons une suite de paramètres de tension $(r_i^k)_{i \in I}$ pour laquelle, si k est suffisamment grand, il existe une constante c telle que

$$\frac{r_{\max}^k}{r_{\min}^k} \leq c \quad \text{et} \quad \lim_{k \rightarrow \infty} r_{\max}^k = \infty$$

$$\text{Ici } r_{\max}^k = \max_{i \in I} (r_i^k), \quad r_{\min}^k = \min_{i \in I} (r_i^k)$$

La suite (σ^k) est formée par les splines pénalisées qui correspondent aux paramètres r_i^k , $i \in I$.

Proposition 6.7

Prenons l'ensemble des données $\{(t_i, y_i)\} \quad i=1,2,\dots,n$ tel que

$$(6.19) \quad 2 \left(\frac{h_{i-1} + h_i}{h_{i+1} + h_i} \right) (m_{i+1} - m_i) > (m_i - m_{i-1}) \quad i=2,\dots,n-2$$

et une suite de splines pénalisées d'interpolation $\{\sigma^k\}$ définie comme ci-dessus.

Alors pour r_{\max}^k suffisamment grand, σ^k sera convexe. \square

On remarque que la condition (6.19), sur les données est plus forte que la convexité. Il existe un résultat analogue pour la concavité.

• Conséquences sur la monotonie

Dans [41] on démontre que :

le polynôme cubique p_i tel que $p_i(t_j) = y_j$ et $p'_i(t_j) = \lambda_j \quad (j=i, i+1)$ est monotone sur l'intervalle $[t_i, t_{i+1}]$ si et seulement si

$$(\alpha_i, \beta_i) = \left(\frac{\lambda_j}{m_i}, \frac{\lambda_{j+1}}{m_i} \right) \text{ appartient à la région } \mathcal{M} \text{ du plan } \alpha\text{-}\beta, \text{ définie}$$

par :

$$\mathcal{M} = \{(\alpha, \beta) / \alpha \geq 0, \beta \geq 0, (\alpha + \beta \leq 3 \text{ ou } \alpha^2 + \beta^2 + \alpha\beta - 6(\alpha + \beta) + 9 \leq 0)\}$$

Si $m_i = 0$ on prend $\lambda_i = \lambda_{i+1} = 0$. \square (Voir Fig3. chap.1)

• Pour les ν -splines, au cas limite, lorsque $\nu_i \rightarrow \infty, \nu_{i+1} \rightarrow \infty$ on a $(\alpha_i, \beta_i) = (0, 0)$. Comme on l'a déjà signalé, ceci conduit à la formation de paliers autour des points d'interpolation dans le graphe de la fonction.

• Les dérivées des splines avec poids, s'approchent du point (1,1) lorsque $w_i \rightarrow \infty$. Cinquin [19] montre, par d'autres moyens, la préservation de la monotonie lorsque w_i devient suffisamment grand.

• On sait que, si m_{i-1}, m_i, m_{i+1} ont le même signe, la condition nécessaire de monotonie, $\alpha_i > 0$ et $\beta_i > 0$, est remplie par les splines pénalisées lorsque $r_i \rightarrow \infty$ et $r_{i+1} \rightarrow \infty$. En effet

$$\alpha_i = \frac{h_{i-1} + h_i m_{i-1} / m_i}{h_{i-1} + h_i}, \quad \beta_i = \frac{h_{i+1} + h_i m_{i+1} / m_i}{h_{i+1} + h_i}$$

Mais cette condition sera aussi suffisante si (α_i, β_i) se trouve dans la région \mathcal{M} .

On peut étendre la proposition 6.7. de la manière suivante :

Proposition 6.8

Si en plus des hypothèses de la proposition 6.7., les données sont monotones alors pour r_{\max}^k suffisamment grand σ^k sera convexe et monotone. \square

6.2. Lissage

Passons maintenant au problème du lissage des données z_i ($i=1,2,\dots,n$) affectées d'erreurs. Le problème consiste à

$$(6.20) \quad \left\{ \begin{array}{l} \text{minimiser } E_4(x) + \rho J(x) \\ x \in H^2[a,b] \end{array} \right.$$

On rappelle que $J(x) = \sum_{i=1}^n (x(t_i) - z_i)^2$

On sait que l'espace des fonctions splines pénalisées est le même, qu'il s'agisse d'interpolation ou d'ajustement. La seule différence dans la proposition de caractérisation est qu'une nouvelle condition appelée condition du lissage remplace la condition d'interpolation (relation (v) de la proposition (6.2)). Dans chaque cas, les conditions d'existence et d'unicité sont les mêmes (cf. Laurent [56]).

Un élément $\sigma \in H^2[a,b]$ est solution de (6.20) si et seulement s'il existe des coefficients $\delta_i (i=1, \dots, n)$ vérifiant

$$(6.21) \quad T * T \sigma = \sum_{i=1}^n \delta_i k_i \quad \text{et}$$

$$(6.22) \quad \delta_i = \rho(z_i - \sigma(t_i)) \quad i=1, 2, \dots, n$$

Dans le paragraphe 6.1. on a trouvé

$$\sigma''(t) = \sum_{i=1}^n [\delta_i - \Psi \sigma_i] (t_i - t)_+ + \sum_{i=1}^n r_i (\sigma'(t_i) - \varphi \sigma_i) (t_i - t)_+^0 \quad t \in [a,b]$$

On en déduit que

$$\sigma'''(t_i^+) = \sum_{j=1}^i (\delta_j - \Psi \sigma_j) , \quad \sigma'''(t_i^-) = \sum_{j=1}^{i-1} (\delta_j - \Psi \sigma_j) \quad i=1, 2, \dots, n$$

alors

$$(6.23) \quad \delta_i = \sigma'''(t_i^+) - \sigma'''(t_i^-) + \Psi \sigma_i \quad i=1, 2, \dots, n$$

La condition de lissage s'écrit :

$$(6.24) \quad \begin{cases} \sigma'''(t_1^+) + \Psi \sigma_1 = \rho(z_1 - \sigma(t_1)) \\ \sigma'''(t_i^+) - \sigma'''(t_i^-) + \Psi \sigma_i = \rho(z_i - \sigma(t_i)) \quad i=2, 3, \dots, n-1 \\ -\sigma'''(t_n^-) + \Psi \sigma_n = \rho(z_n - \sigma(t_n)) \end{cases}$$

6.2.1 Algorithmes de calcul

On remarque que les formules de lissage (6.24) sont plus compliquées que celles qui correspondent aux ν -splines ou aux splines avec poids. La présence de $\Psi\sigma_i$ détermine que le système linéaire ne soit plus tridiagonal par blocs mais pentadiagonal par blocs.

Plus précisément, pour le cas $h = h_i$ ($i=1,2,\dots,n-1$), c'est-à-dire, si les abscisses t_i sont équidistantes on a :

$$(6.26) \left\{ \begin{array}{l} D_1 \Lambda_1 + E_1 \Lambda_2 + F_1 \Lambda_3 = b_1 \\ C_2 \Lambda_1 + D_2 \Lambda_2 + E_2 \Lambda_3 + F_2 \Lambda_4 = b_2 \\ B_i \Lambda_{i-2} + C_i \Lambda_{i-1} + D_i \Lambda_i + E_i \Lambda_{i+1} + F_i \Lambda_{i+2} = b_i \quad i=3,4,\dots,n-2 \\ B_{n-1} \Lambda_{n-3} + C_{n-1} \Lambda_{n-2} + D_{n-1} \Lambda_{n-1} + E_{n-1} \Lambda_n = b_{n-1} \\ B_n \Lambda_{n-2} + C_n \Lambda_{n-1} + D_n \Lambda_n = b_n \end{array} \right.$$

où

$$D_1 = \begin{bmatrix} 6 + \alpha + 2p_1 + \frac{p_2}{2} & 3 + 2p_1 \\ 3 + 2p_1 & 2(1 + p_1) \end{bmatrix} \quad D_2 = \begin{bmatrix} 12 + \alpha + 2p_1 + \frac{p_3}{2} & 0 \\ 0 & 2(2 + p_2) \end{bmatrix}$$

$$D_i = \begin{bmatrix} 12 + \alpha + \frac{1}{2}(p_{i-1} + p_{i+1}) & 0 \\ 0 & 2(2 + p_i) \end{bmatrix} \quad i=3,\dots,n-2$$

$$D_{n-1} = \begin{bmatrix} 12 + \alpha + \frac{p_{n-2} + 2p_n}{2} & 0 \\ 0 & 2(2 + p_{n-1}) \end{bmatrix} \quad D_n = \begin{bmatrix} 6 + \alpha + \frac{p_{n-1} + 2p_n}{2} & -(3 + 2p_n) \\ -(3 + 2p_n) & 2(1 + p_n) \end{bmatrix}$$

$$E_i = \begin{bmatrix} -(6+2p_i) & 3+p_{i+1} \\ -(3+2p_i) & 1 \end{bmatrix} \quad E_i = \begin{bmatrix} -6 & 3+p_{i+1} \\ -(3+p_i) & 0 \end{bmatrix} \quad i=2,3,\dots,n-2$$

$$E_{n-1} = \begin{bmatrix} -(6+2p_{n-2}) & 3+2p_n \\ -(3+2p_{n-1}) & 1 \end{bmatrix} \quad F_i = \begin{bmatrix} \frac{-p_{i+1}}{2} & 0 \\ 0 & 0 \end{bmatrix} \quad i=1,2,\dots,n-2$$

$$B_i = F_{i-2} \quad i=3,\dots,n ; \quad C_i = E_{i-1}^T \quad i=2,\dots,n ;$$

$$p_i = r_i h/4 , \quad \Lambda_i = [y_i \quad h\lambda_i]^T , \quad b_i = [\alpha z_i \quad 0]^T \quad i=1,2,\dots,n$$

avec $\alpha = \rho h^3 / 2$.

Remarque

La matrice du système linéaire (6.25), que nous noterons M , est symétrique par blocs 2x2. Si l'on ne considère pas la structure bloc alors M est une matrice bande symétrique.

Pour résoudre (6.25) nous allons utiliser deux méthodes: la première, directe, celle de Cholesky adaptée aux matrices bandes; la seconde, itérative, celle de la relaxation successive par blocs (BSOR).

III.7. LE CHOIX DES PARAMETRES DE TENSION

Le choix des paramètres de tension ou de pénalisation est très important si l'on veut construire des splines monotones ou convexes. Malheureusement, il n'existe pas de formules analytiques pour ces paramètres qui permettent d'établir un seuil à partir duquel la convexité ou la monotonie est respectée. Ainsi, si la tension est uniforme ou trop grande, la courbe peut être monotone ou sans oscillations indésirables mais des points anguleux apparaissent aux noeuds.

En général, pour l'interpolation on a la procédure suivante :

Procédure 7.1

- 1) Pour commencer, on donne un ensemble de paramètres et on calcule la fonction spline correspondante.
- 2) On vérifie si la convexité ou la monotonie est préservée. Si c'est le cas on termine, sinon on va à 3).
- 3) On met à jour les paramètres, en augmentant "suffisamment" ces valeurs. On recalcule la spline pour ce nouveau jeu des paramètres. On revient à 2). □

Commentaires

- 1) S'il s'agit du lissage, alors il faudra choisir aussi le paramètre de lissage "optimal" aux pas 1) et 3).
- 2) Habituellement on démarre l'algorithme 7.1 en calculant la fonction spline cubique ("naturelle").
- 3) Le critère pour vérifier la convexité de σ dans l'intervalle $[t_j, t_{j+1}]$ est que $\beta_j \beta_{j+1} > 0$ si $\Delta_j^2 \cdot \Delta_{j+1}^2 > 0$, où $\beta_j = \sigma''(t_j)$ et Δ_j^2 est la différence divisée d'ordre 2 au noeud t_j .
- 4) Pour la monotonie de s dans l'intervalle $[t_j, t_{j+1}]$ la condition est que $\text{sgn}(\lambda_j) = \text{sgn}(\lambda_{j+1}) = \text{sgn}(m_j)$ mais aussi que

l'extremum de $\sigma'(t)$ (s'il existe dans $[t_i, t_{i+1}]$) ait le même signe que λ_i , λ_{i+1} et m_i . On rappelle les notations

$$m_i = (y_{i+1} - y_i) / (t_{i+1} - t_i) \text{ et } \lambda_i = \sigma'(t_i).$$

• Pour les splines hyperboliques sous tension, Späth [106] suggère de prendre $r_i h_i < 5$, tandis que Rentrop [97] choisit $r_i h_i$ dans l'intervalle $[4, 15]$. Lorsqu'on doit augmenter la tension dans l'intervalle $[t_i, t_{i+1}]$ Pruess [92] propose la mise à jour suivante

$$(7.2) \begin{cases} r_i &= 2 r_i \\ r_{i-1} &= \max (r_{i-1}, (r_{i-1} h_{i-1} + r_i h_i) / 2 h_{i-1}) \\ r_{i+1} &= \max (r_{i+1}, (r_{i+1} h_{i+1} + r_i h_i) / 2 h_{i+1}) \end{cases}$$

• Le théorème de Rolle permet de conclure qu'il existe $t^* \in [t_i, t_{i+1}]$ tel que $\sigma'(t^*) = m_i$. Frost et Kinze [42], au lieu d'utiliser le critère de convexité défini plus haut, imposent à la spline rationnelle sous tension d'interpolation que la distance de $(t^*, \sigma(t^*))$ à la droite qui relie (t_i, y_i) et (t_{i+1}, y_{i+1}) soit plus petite qu'une constante donnée. Si tel n'est pas le cas, alors on augmente le paramètre de tension r_i d'une unité.

• Salkaukas [99] prend les poids $w_i = (1 + m_i^2)^{-3}$ pour la spline cubique avec poids d'interpolation.

Dans le paragraphe 9 on présentera un ensemble d'expériences numériques pour ces différentes propositions.

Il existe une autre manière (non automatique) de choisir les paramètres de tension. L'utilisateur essaie d'une façon *interactive*, à l'aide d'un terminal graphique, différentes valeurs de ces paramètres et il doit choisir ensuite le dessin de la fonction (et éventuellement de ses dérivées) qui semble être le plus approprié.

III.8. LE CHOIX DU PARAMETRE DE LISSAGE

Soient $a < t_1 < t_2 < \dots < t_n < b$, n points de l'intervalle réel $[a, b]$. Soient $f(t_i)$, $i=1,2,\dots,n$, les valeurs d'une fonction f aux noeuds t_i : ces valeurs étant connues avec des erreurs $\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n$. Notons

$$(8.1) \quad z_i = f(t_i) + \varepsilon_i \quad i=1,2,\dots,n$$

Les ε_i sont considérées comme des variables aléatoires indépendantes telles que

$$(8.2) \quad \begin{cases} E(\varepsilon_i) = 0 \\ E(\varepsilon_i \varepsilon_j) = \begin{cases} v_i^2 & \text{si } i=j \\ 0 & \text{si } i \neq j \end{cases} \end{cases} \quad i=1,2,\dots,n$$

Pour approcher f , on se propose de lisser les données $\{z_i\}$, ce qui nous conduit à trouver la solution $\sigma_{p,k}$ du problème de minimisation

$$(8.3) \quad \begin{cases} \text{minimiser } E_k(x) + \rho J(x) & k = 1, 2, 3, 4 \\ x \in H^2[a, b] \end{cases}$$

$$\text{où } J(x) = \sum_{i=1}^n \delta_i (x(t_i) - z_i)^2$$

$$E_1(x) = \int_a^b [x''(t)]^2 dt + \int_a^b r(t) [x'(t)]^2 dt \quad (\text{splines hyperboliques})$$

$$E_2(x) = \int_a^b [x''(t)]^2 dt + \sum_{i=1}^n v_i [x'(t_i)]^2 dt \quad (v\text{-splines})$$

$$E_3(x) = \int_a^b w(t) [x''(t)]^2 dt \quad (\text{splines avec poids})$$

$$E_4(x) = \int_a^b [x''(t)]^2 dt + \sum_{i=1}^n r_i [x'(t_i) - \varphi_{x_i}]^2 dt \text{ (splines pénalisées)}$$

Dans les paragraphes précédents nous avons étudié ces différents problèmes. Il nous reste à faire un "bon choix" de ρ :

si ρ est trop faible, le lissage est trop marqué et on risque d'effacer les variations de la fonction ; par contre si ρ est trop grand la spline est contrainte de passer très près des données, ce qui produit, à cause des erreurs, une mauvaise approximation de f .

Nous considérons deux méthodes pour la détermination de ρ : la première, dûe à Reinsch, la seconde connue sous le nom de méthode de validation croisée.

8.1. La méthode de Reinsch

On suppose que les v_i sont connues, et on prend $\delta_i = v_i^{-2}$.

L'idée de base de cette méthode est la suivante :

$\varepsilon_i = z_i - f(t_i)$, $i=1,2,\dots,n$, et les hypothèses (8.2) donnent

$$(8.4) \quad E \left[\sum_{i=1}^n v_i^{-2} (f(t_i) - z_i)^2 \right] = n$$

Alors pour n suffisamment grand on doit avoir

$$(8.5) \quad \sum_{i=1}^n v_i^{-2} (f(t_i) - z_i)^2 \approx n$$

ce qui suggère de choisir ρ tel que

$$(8.6) \quad J(\sigma_{\rho,k}) = n$$

Reinsch [95], [96] étudie le problème suivant :

$$(8.7) \quad \left\{ \begin{array}{l} \text{minimiser } E_k(x) \text{ avec la contrainte } J(x) \leq S, \\ x \in H^2[a,b] \\ \text{où } S \text{ est une constante (} k=1,2,3,4 \text{)} \end{array} \right.$$

Il démontre que les problèmes (8.3) et (8.7) sont équivalents à condition de choisir ρ vérifiant

$$(8.8.) \quad J(\sigma_{\rho,k}) = S$$

Plusieurs auteurs [23] , [113], [114] conseillent de prendre $S < n$. En particulier Reinsch propose que S appartienne à l'intervalle $[n-\sqrt{2n}, n+\sqrt{2n}]$.

Résolution de l'équation (8.8)

Notons

$$(8.9) \quad F(\rho) = J(\sigma_{\rho,k})$$

et réécrivons l'équation (8.8) sous la forme

$$(8.10) \quad H(\rho) = 0$$

$$\text{avec} \quad H(\rho) = [F(\rho)]^{1/2} - [S]^{-1/2}$$

La résolution de (8.10) sera faite par la méthode de Newton. Le schéma itératif est le suivant :

$$(8.12) \quad \left\{ \begin{array}{l} \rho_0 \text{ donné} \\ \rho_{j+1} = \rho_j + \Delta\rho_j \quad j=0,1,2,\dots \text{ jusqu'à la convergence} \\ \text{où } \Delta\rho_j = 2 F(\rho_j) [1 + [F(\rho_j)]^{1/2} [S]^{-1/2}] / F'(\rho_j) \end{array} \right.$$

Alors il faut savoir calculer $F'(\rho_j)$. De (8.9) on obtient

$$(8.13) \quad F'(\rho) = 2 \sum_{i=1}^n \delta_i (\sigma_{\rho,k}(t_i) - z_i) \frac{d\sigma_{\rho,k}(t_i)}{d\rho}$$

Pour calculer les splines de lissage nous étions à chaque fois amenés à résoudre un système linéaire de la forme.

$$(8.14) \quad M_k \Lambda = b$$

$$\text{où } \Lambda = [\Lambda_1 \dots \Lambda_n]^T \text{ et } b = [b_1 \dots b_n]^T$$

$$\text{avec } \Lambda_i = [y_i \beta_i] \quad i=1,2,\dots,n \quad \text{si } k=1$$

$$\Lambda_i = [y_i \lambda_i] \quad i=1,2,\dots,n \quad \text{si } k=2,3,4$$

$$\text{et } b_i = [\rho \mu_i z_i \ 0] \quad i=1,2,\dots,n$$

$$\text{où } \mu_i = -\delta_i h_i \quad \text{si } k=1$$

$$\mu_i = \delta_i h_i^3 / 2 \quad \text{si } k=2,3,4$$

Si les pas sont équidistants ($h=h_i$, $i=1,2,\dots,n-1$) alors pour $k=2,3,4$ on pose $\Lambda_i = [y_i \ h \lambda_i] \quad i=1,2,\dots,n$.

Si l'on dérive les équations (8.14) par rapport à ρ on a

$$(8.15) \quad M_k \frac{d\Lambda}{d\rho} + \frac{dM_k}{d\rho} \Lambda = \frac{db}{d\rho}$$

C'est-à-dire

$$(8.16) \quad M_k \frac{d\Lambda}{d\rho} = [\mu_1(z_1 - y_1), 0, \dots, \mu_1(z_1 - y_1), 0, \dots, \mu_n(z_n - y_n), 0]^T = c$$

$$\text{où } y_i = \sigma_{\rho,k}(t_i) \quad i=1,2,\dots,n$$

Alors

$$(8.17) \quad \frac{d\Lambda}{d\rho} = M_k^{-1} c$$

En conclusion, pour trouver $\frac{d\Lambda}{d\rho}$ et donc $\frac{d\sigma_{p,k}(t_i)}{d\rho}$ ($i=1,2,\dots,n$) il

faut calculer d'abord la spline de lissage par rapport aux données z et après la spline de lissage par rapport au vecteur c .

Si l'on veut éviter le calcul de $F'(\rho)$, on peut résoudre (8.10) par des algorithmes du type dichotomie ou sécante.

8.2. La méthode de validation croisée

Avant d'étudier cette méthode, nous voulons signaler que le problème (8.3) peut être ramené à un problème de minimisation dans \mathbb{R}^n . Pour cela on emprunte le même chemin que nous avons utilisé pour les ν -splines (§4.3).

$$\text{On note } (\sigma_i^k) \quad i=1,2,\dots,n, \quad \text{où } \sigma_i^k(t_j) = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases}$$

la base canonique de l'espace des fonctions splines:

- . hyperboliques sous tension si $k=1$
- . ν -splines si $k=2$
- . cubiques avec poids si $k=3$
- . cubiques pénalisées si $k=4$

Alors (8.3) est équivalente à

$$(8.18) \quad \underset{x \in \mathbb{R}^n}{\text{minimiser}} \quad \langle x, D_k x \rangle + \rho J(x)$$

où Ω_k est une matrice symétrique, semi-définie positive, caractérisée par $\Omega_k = (\omega_{ij}^k)$ $i, j = 1, 2, \dots, n$, $k = 1, 2, 3, 4$

$$\omega_{ij}^1 = \sigma_i'''(t_j^+) - \sigma_i'''(t_j^-) - r_i^2 \sigma_i'(t_j^+) + r_{i-1}^2 \sigma_i'(t_j^-)$$

$$\omega_{ij}^2 = \sigma_i'''(t_j^+) - \sigma_i'''(t_j^-)$$

$$\omega_{ij}^3 = w_i \sigma_i'''(t_j^+) - w_{i-1} \sigma_i'''(t_j^-)$$

$$\omega_{ij}^4 = \sigma_i'''(t_j^+) - \sigma_i'''(t_j^-) - \Psi(\sigma_i)_j \quad i, j = 1, 2, \dots, n$$

Pour alléger la notation nous n'avons pas écrit l'indice k sur les dérivées des σ_i^k .

Le minimum de (8.18) est atteint pour la valeur de $x = \tilde{x}$ qui est donnée par l'expression

$$(8.19) \quad \Omega_k \tilde{x} + \rho D \tilde{x} = \rho D z \quad \text{avec } D = \text{diag}(\delta_i).$$

ce qui conduit finalement à

$$(8.20) \quad \tilde{x} = (D + \frac{1}{\rho} \Omega_k)^{-1} D z$$

On notera

$$(8.21) \quad A_k(\rho) = (D + \frac{1}{\rho} \Omega_k)^{-1} D$$

En pratique nous prendrons $\delta_i = 1$ ($i=1, 2, \dots, n$), c'est-à-dire

$$A_k(\rho) = (I + \frac{1}{\rho} \Omega_k)^{-1}$$

La méthode de validation croisée dans sa version "généralisée" ([23],[114]) consiste à minimiser la quantité

$$(8.22) \quad V_k(\rho) = \frac{\frac{1}{n} \| z - A_k(\rho) z \|^2}{\left(1 - \frac{1}{n} \text{Tr}(A_k(\rho))\right)}$$

Le vecteur $y = A_k(\rho) z$ est évalué en résolvant le système (8.14). Pour donner la forme explicite de $A_k(\rho)$ on calcule $A_k(\rho) e_i$ $i=1, \dots, n$ où $(e_i)_{i=1, \dots, n}$ est la base canonique de \mathbb{R}^n .

Dans le cas des splines cubiques naturelles à pas et variances égales, Utreras ([108]), donne des approximations des valeurs propres de $A(\rho)$ pour calculer la trace de cette matrice. Le nombre d'opérations du dénominateur baisse dans ce cas de $30n^2$ à $2n$.

Pour la mise en oeuvre numérique nous allons expliciter à chaque fois la matrice $A_k(\rho)$ et minimiser $V_k(\rho)$ par la méthode de la section dorée.

Nous rappelons que l'intérêt de la validation croisée est qu'à la différence de la méthode de Reinsch, elle ne suppose pas connu l'écart type.

III.9. ESSAIS NUMERIQUES

Les expériences numériques se divisent en deux groupes : le premier, pour l'interpolation, et, le second, pour le lissage de données.

9.1. Interpolation

Signalons d'abord la manière de choisir les paramètres de tension.

- * Pour les splines hyperboliques et rationnelles sous tension nous utilisons la procédure 7.1. de la façon suivante :

- On part de $r_i = 0 \quad i=1,2,\dots,n-1$ (spline cubique)
- Si l'on doit modifier des r_i on emploie la formule suggérée par Rentrop [97]:

$$(9.1) \quad r_i = r_i + \theta_i$$

$$\text{où } \theta_i h_i = \begin{cases} 15 & \text{si } y_i = y_{i+1} = 0 \\ 4 + 1 / \left(0.1 + \frac{|y_{i+1} - y_i|}{\max(|y_i|, |y_{i+1}|)} \right) & \text{sinon} \end{cases}$$

jusqu'à atteindre la préservation de la forme des données. \square

Dans les exemples que nous avons traités le nombre de mise à jour des r_i a été, au plus, de 4.

- * Pour la spline avec poids w_i ($i=1,2,\dots,n-1$) on a pris :

$$(9.2) \quad w_i = (1 + ((y_{i+1} - y_i)/h)^2)^{-5/2}$$

$$\text{pour que } E_3(x) = \int_a^b w(t)[x''(t)]^2 dt \text{ "ressemble" } E(x) = \int_a^b \frac{[x''(t)]^2}{[1+(x'(t))^2]^{5/2}} dt$$

Nous rappelons qu'en minimisant $E(x)$ on obtient la spline "elastica" (voir S. 5.1).

- * Pour les splines cubiques pénalisées et de Nielson, les paramètres sont mis à jour en ajoutant à chaque fois la quantité $15/h_i$ si $h_i < 1$ ou $15h_i$ si $h_i \geq 1$.

• Préservation de la monotonie

Considérons les données monotones regroupées dans le tableau 1.

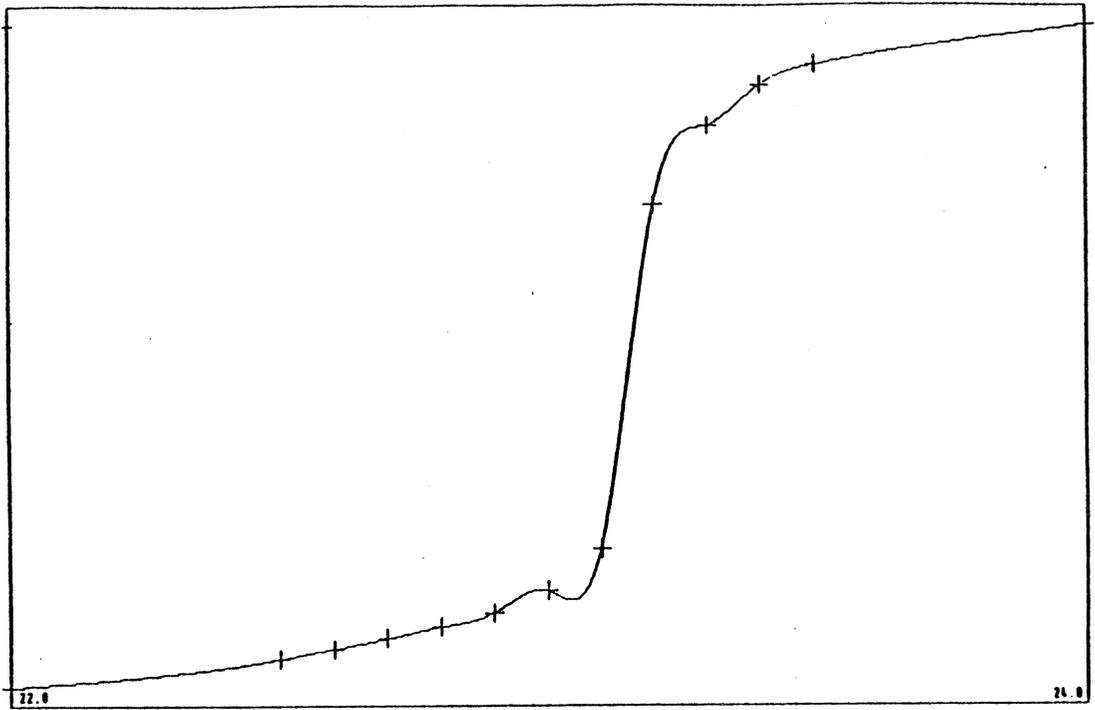
t_i	22	22.5	22.6	22.7	22.8	22.9	23	23.1	23.2	23.3	23.4	23.5	24
z_i	523	543	550	557	565	575	590	620	860	915	944	958	986

Tableau 1. Données monotones (Pruess [93])

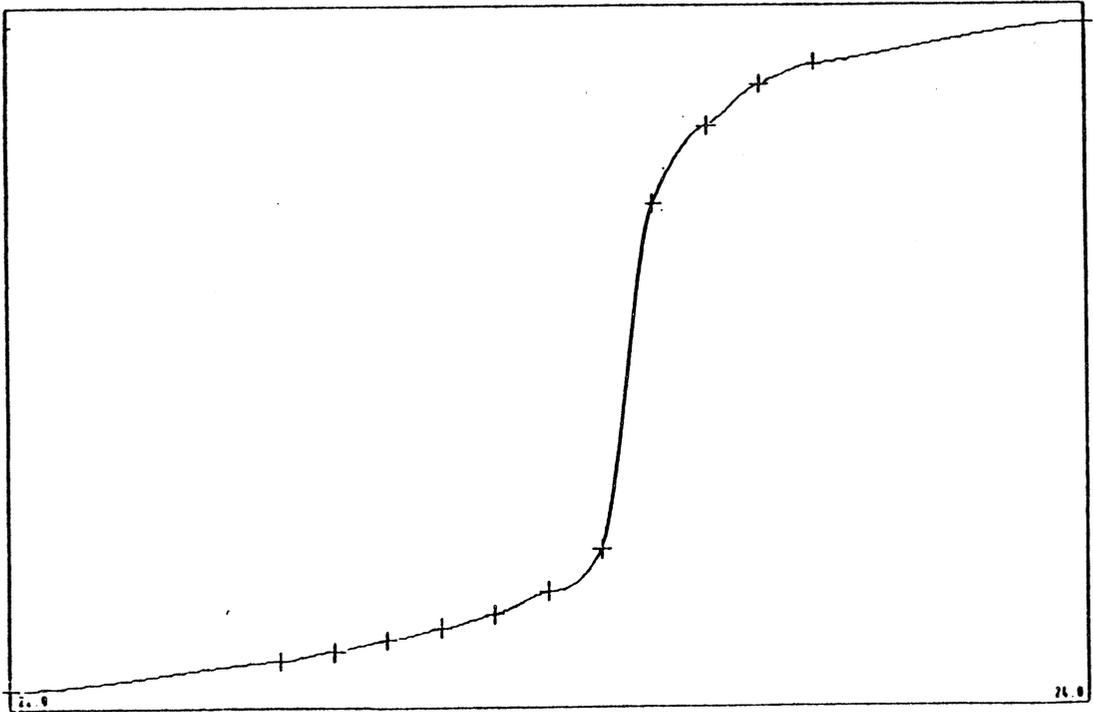
La Fig.1(a) montre le graphe de la fonction spline cubique. La monotonie n'est pas conservée dans l'intervalle $[t_5, t_6] = [22.8, 22.9]$. Pour $v_i = 200$ ($i=1,2,\dots,13$), le profil de la courbe v -spline est monotone voir Fig.1(b). Mais si l'on augmente les valeurs des v_i alors on va trouver des paliers autour des points d'interpolation car la dérivée première aux noeuds tend vers zéro.

Si la tension est appliquée localement en t_5 et t_6 on rend la courbe monotone pour la spline pénalisée (Fig.2(a)).

On obtient une courbe monotone et libre des points d'inflexion "artificiels" lorsqu'on calcule la spline cubique avec les poids w_i fournis par la formule (9.2). Voir Fig.2(b). Les splines hyperboliques et rationnelles (Fig.3) donnent des résultats semblables si la tension est mise à jour par (9.1).

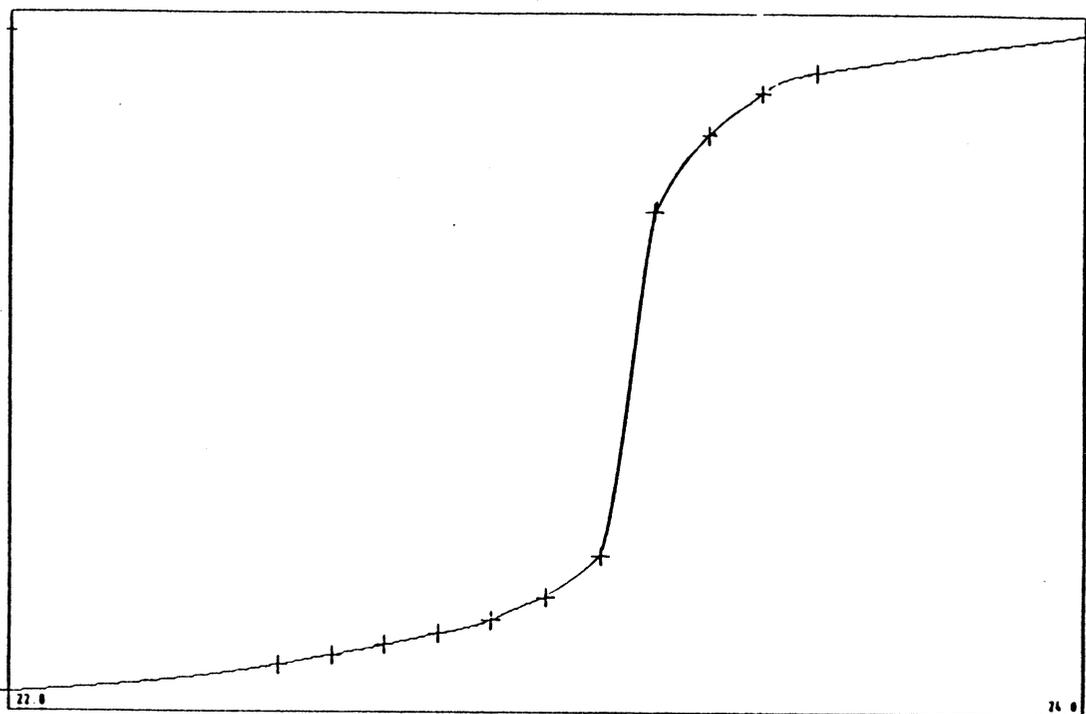


(a)

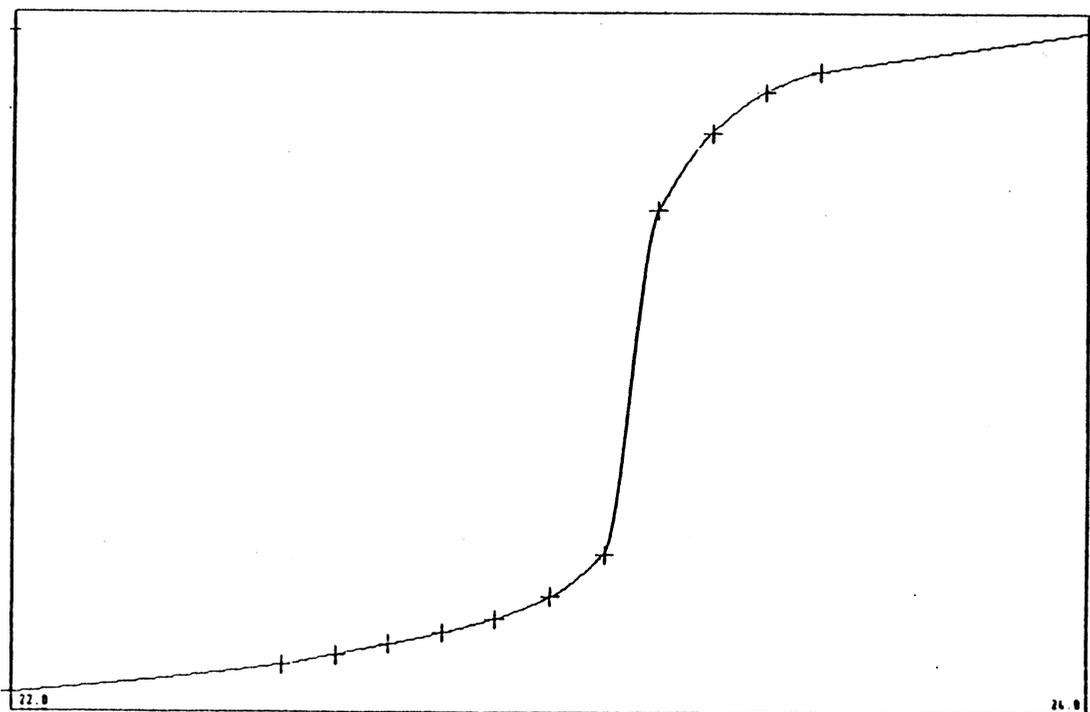


(b)

Fig. 1. Interpolation des données monotones 1.
(a) F.S. Cubique naturelle
(b) F.S. Cubique de Nielson ($\nu_j=200$)



(a)

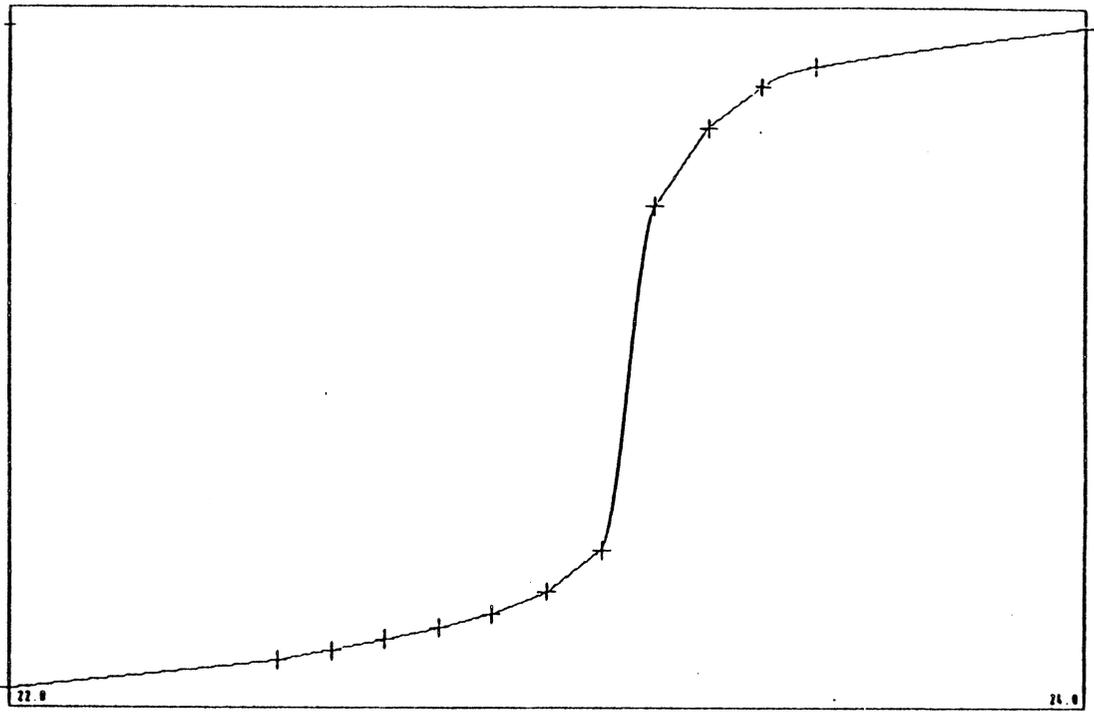


(b)

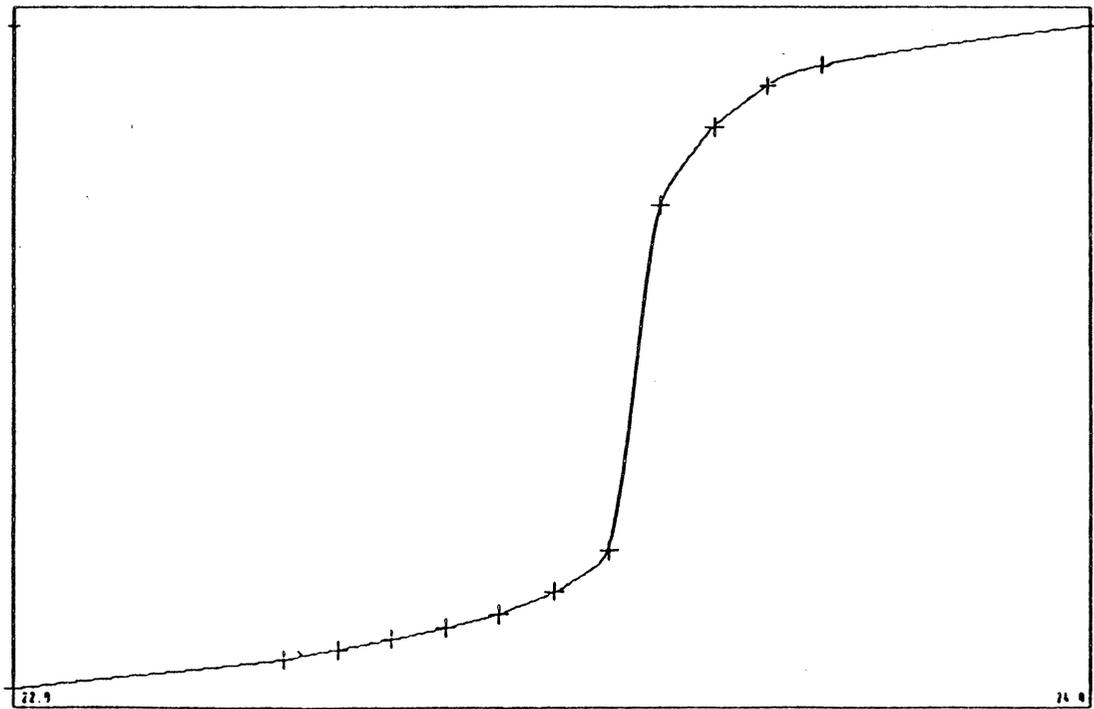
Fig. 2. Interpolation des données monotones 1.

(a) F.S. Cubique pénalisée

(b) F.S. Cubique avec poids.



(a)



(b)

Fig. 3. Interpolation des données monotones 1.
(a) F.S. Rationnelle sous tension
(b) F.S. Hyperbolique sous tension

• Elimination des points d'inflexion "artificiels"

Présentons maintenant l'exemple dont les données apparaissent dans le tableau 2.

t_j	-3.0	-2.0	-1.0	0.9	3.1	6.0	10.5	13.0
y_j	-4.0	1.2	3.7	1.3	-1.5	-1.5	-0.8	4.0

Tableau 2.

Le dessin de la fonction spline cubique montre deux "bosses" entre $t_5 = 3.1$ et $t_7 = 10.5$ (Fig.4(a)).

Si les poids w_j sont fournis par la relation (9.2) les points d'inflexion "artificiels" ont disparu (Fig.4(b)).

En mettant à jour les paramètres de tension r_j par (9.1) on obtient des courbes sans oscillations en 4 itérations pour les splines rationnelles (Fig.5(a)) et en 2 itérations pour les splines hyperboliques (Fig.5(b)). La fonction devient linéaire sur l'intervalle $[t_5, t_6]$, ce qui correspond à la forme des données.

• Sur l'encombrement et le temps de calcul

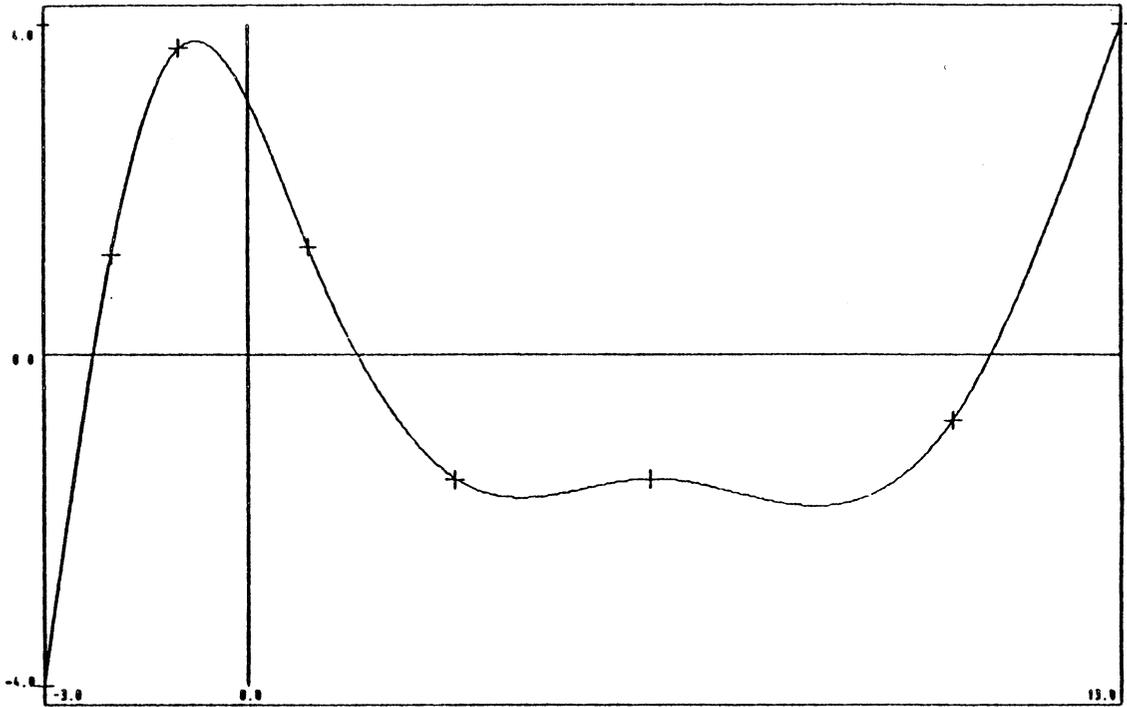
Nous rappelons que la méthode programmée pour calculer les dérivées premières (ou secondes) aux noeuds est celle de Gauss sans pivot pour systèmes linéaires tridiagonaux.

Du point de vue mémoire on a toujours besoin de 4 tableaux unidimensionnels :

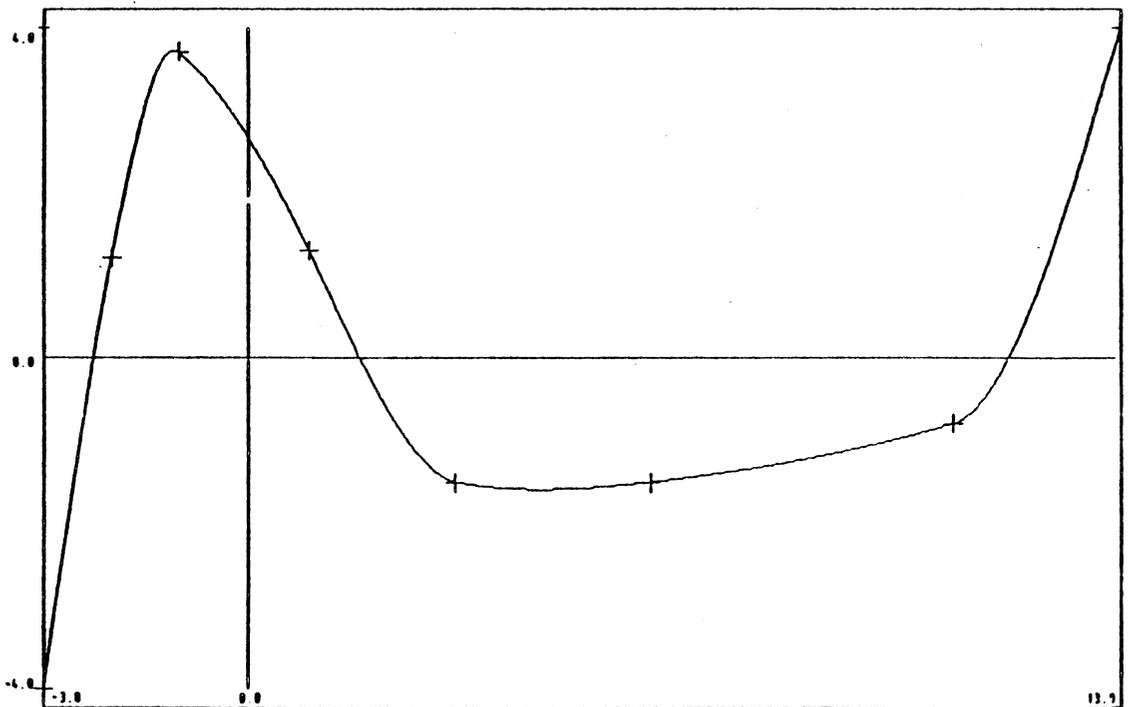
$$t[i], y[i], dy[i] \text{ (ou } d^2y[i]), r[i] \quad i=1,2,\dots,n$$

pour les abscisses, les ordonnées, les dérivées aux noeuds et la tension. En plus, pour les calculs intermédiaires on utilise un tableau supplémentaire de la même taille que les antérieurs.

En suivant la démarche de Laurent [58] si le pas et la tension sont uniformes, on utilise exclusivement les tableaux $y[i]$ et $dy[i]$ si l'on introduit des conditions de bout convenables.

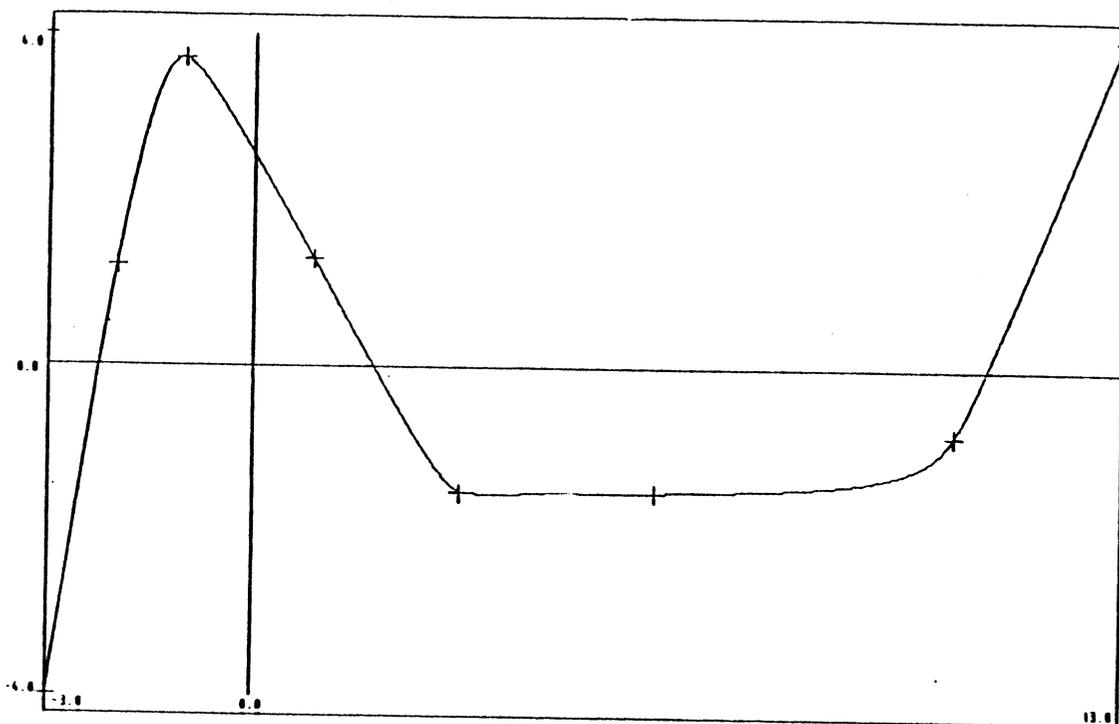


(a)

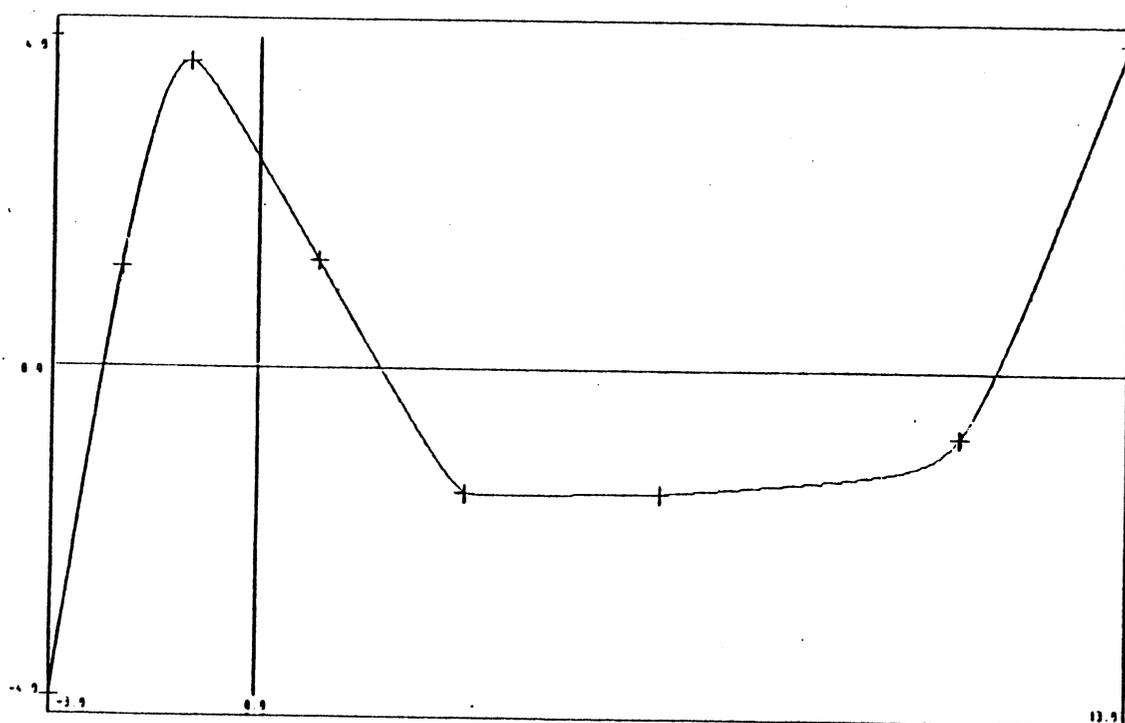


(b)

Fig. 4. Interpolation des données 2.
(a) F.S. Cubique naturelle
(b) F.S. Cubique avec poids.



(a)



(b)

Fig. 5. Interpolation des données 2.
(a) F.S. Rationnelle sous tension
(b) F.S. Hyperbolique sous tension

A titre comparatif pour n abscisses équidistantes et tension uniforme, les résultats sur le temps de calcul des dérivées premières (secondes) de

$$g(t) = (-9t + 2) / (4t + 5) \quad t \in [-1, 8]$$

sont résumés dans le tableau 3.

n	$r_i h_i$	Cubique	C.Nielson	C.pénalisée	Ration.	Hyperbolique	
		0	5 et 15	5 et 15	5 et 15	5	15
50		0.20	0.22	0.26	0.30	0.66	0.86
100		0.40	0.46	0.54	0.60	1.34	1.74
200		0.82	0.92	1.12	1.24	2.68	3.50

Tableau 3. Temps de calcul (en secondes) pour $g(t)$ $t \in [-1, 8]$ avec n abscisses équidistantes.

Nous constatons que :

- Si la tension croît alors le temps de calcul pour la spline hyperbolique augmente. Ceci dû à l'évaluation de la fonction exponentielle pour les coefficients de la matrice. Les autres méthodes ne bougent pas.

- Pour les splines rationnelles le temps de calcul est à peu près 1.5 fois celui des splines cubiques. Si $r_i h_i = 5$, les splines hyperboliques demandent 3 fois le temps de calcul de la spline cubique. Les cubiques de Nielson et pénalisées se placent entre les splines cubiques et les splines rationnelles.

Ces tendances ont été remarquées sur d'autres cas testés.

- Lorsque les poids sont égaux, la spline avec poids se réduit à la spline cubique. Sur d'autres exemples nous avons pu vérifier que le temps de calcul des splines avec poids reste un peu plus élevé que pour les cubiques de Nielson mais sans dépasser celui des cubiques pénalisées.

- Pour les applications pratiques, pour dessiner les courbes par exemple, il faut tenir compte aussi du temps que demande l'évaluation de la fonction spline, ce qui pénalise les splines hyperboliques.

9.2. Lissage

La première question importante à régler est la manière de choisir les paramètres de tension.

Pour les splines hyperboliques nous nous sommes servis des formules de mise à jour :

$$(9.3) \quad \begin{cases} r_i &= 2r_i \\ r_{i-1} &= \max (r_{i-1}, (r_{i-1} h_{i-1} + r_i h_i) / 2h_{i-1}) \\ r_{i+1} &= \max (r_{i+1}, (r_{i+1} h_{i+1} + r_i h_i) / 2h_{i+1}) \end{cases}$$

proposées par Pruess [92]. De même pour les splines cubiques avec poids, lorsqu'on considère la condition de convexité.

Pour les différentes splines cubiques sous tension en présence de la contrainte de monotonie, ces paramètres sont mis à jour en ajoutant à chaque fois la quantité $15/h_i$.

Le paramètre de lissage ρ a été obtenu par la méthode de validation croisée, qui consiste à minimiser la fonction $V(\rho)$:

$$V(\rho) = \frac{\frac{1}{n} \| z - A(\rho)z \|^2}{\left[1 - \frac{1}{n} \text{Tr}(A(\rho)) \right]}$$

Pour calculer $A(\rho)z$ pour ρ donné, on utilise la méthode de Gauss appliquée aux systèmes tridiagonaux par blocs, sauf pour les splines pénalisées qui emploient la décomposition Cholesky. La trace est obtenue en explicitant les éléments de la diagonale de $A(\rho)$, après le calcul des $A(\rho)e_i$ où $\{e_i\}_{i=1,\dots,n}$ est la base canonique de \mathbb{R}^n . Enfin, la minimisation de $V(\rho)$ est faite par la méthode de la section dorée.

Par commodité pratique d'exécution nous avons choisi des exemples avec des abscisses équidistantes.

• Exemple 1

Les données qui se trouvent dans le tableau 4 proviennent du papier de Lafata et Rosen [55]. La fonction sous-jacente est concave.

i	t_i	z_i	i	t_i	z_i
1	0.00	0.0	12	0.55	6.2
2	0.05	2.4	13	0.60	6.3
3	0.10	2.7	14	0.65	6.4
4	0.15	3.5	15	0.70	5.8
5	0.20	4.5	16	0.75	6.3
6	0.25	5.2	17	0.80	5.9
7	0.30	5.5	18	0.85	6.0
8	0.35	6.0	19	0.90	5.5
9	0.40	5.6	20	0.95	3.6
10	0.45	6.2	21	1.00	0.5
11	0.50	5.9			

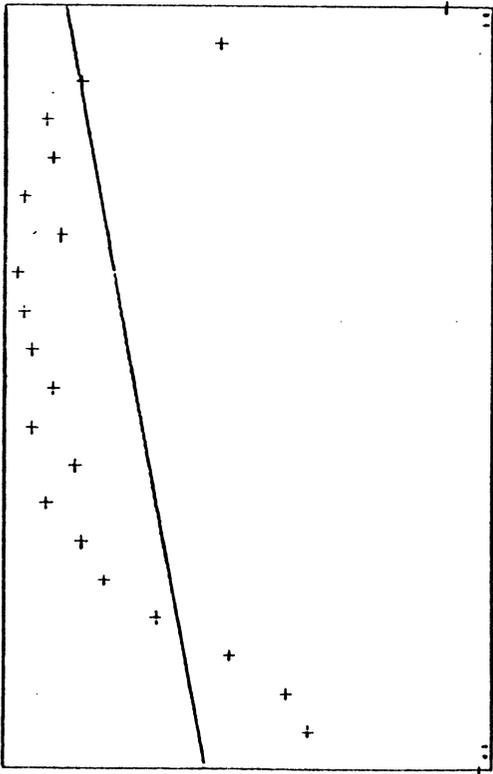
Tableau 4. Données pour l'exemple 1.

Les dessins de la Fig. 6 illustrent le comportement de la spline pour des valeurs faibles et grandes des paramètres de lissage et de tension.

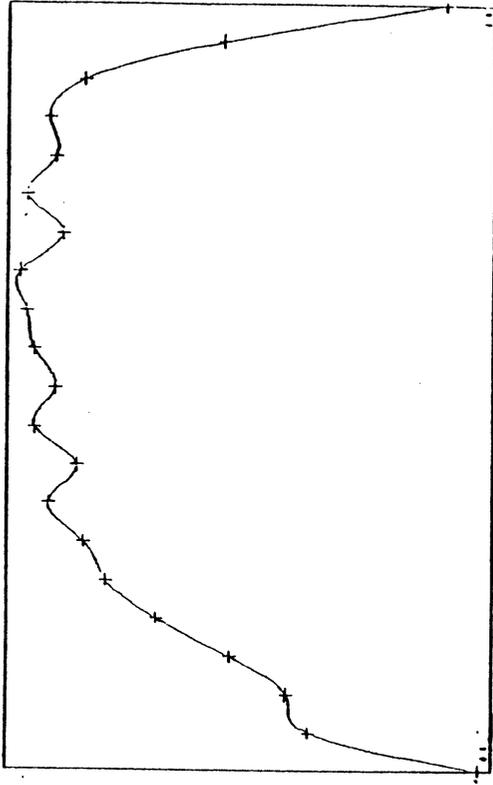
Considérons d'abord le cas où la tension est nulle, c'est-à-dire la fonction spline cubique. Si ρ est petit on tend vers la droite de moindres carrés (Fig.6(a)), par contre lorsque ρ est grand on s'approche de l'interpolation (Fig.6(b)). La spline cubique calculée pour la valeur de ρ optimal est présentée dans la Fig.6(c), et on voit que la concavité n'est pas respectée.

Dans le cas de l'interpolation, on sait que si la tension est grande, la spline hyperbolique tend vers la polygonale qui relie les données. Une situation similaire est mise en évidence pour la Fig.6(d) lorsqu'on a pris $\rho=10^7$ et $r_i=10^2$.

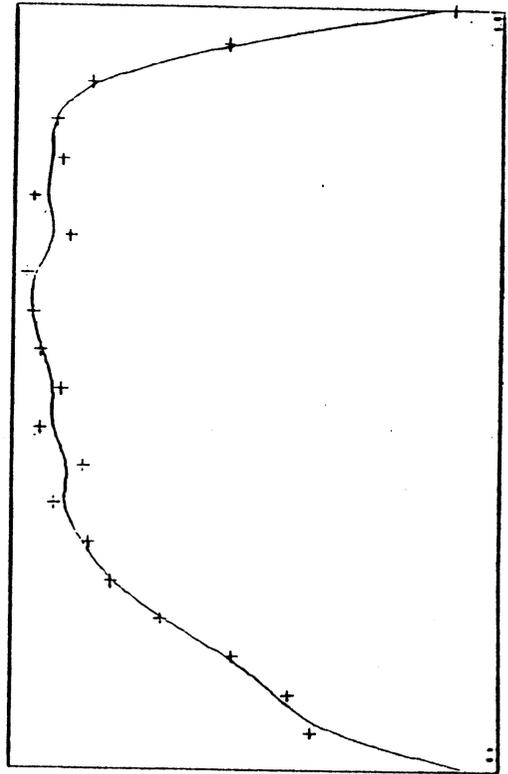
Il est clair qu'on peut obtenir une fonction spline cubique concave en prenant ρ en dessous de sa valeur optimale, mais au risque de perdre les propriétés d'approximation de la solution.



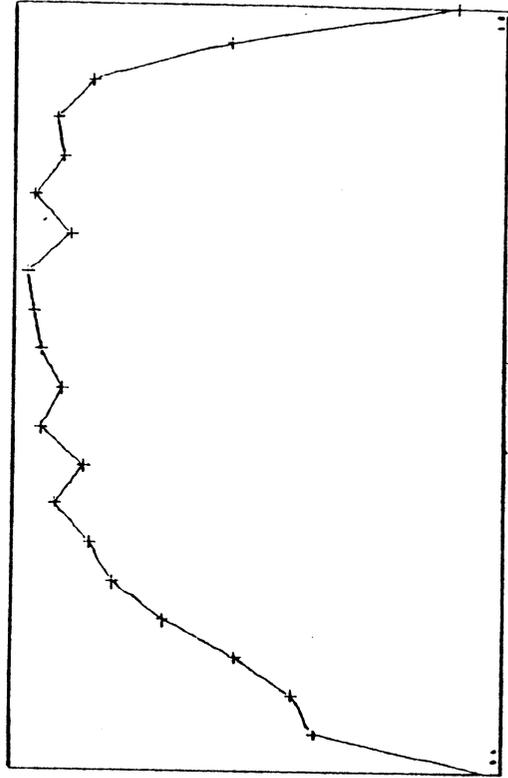
(a) $\rho=10^3$, $r=0$



(b) $\rho=10^7$, $r=0$

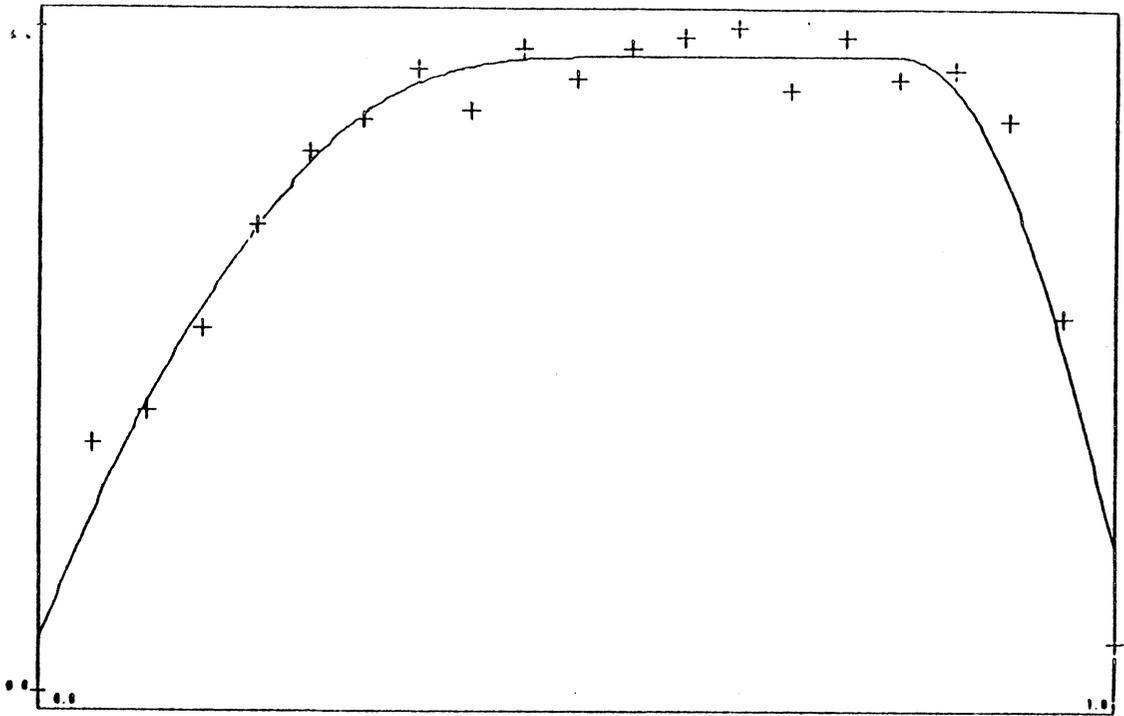


(b) ρ "optimal", $r=0$

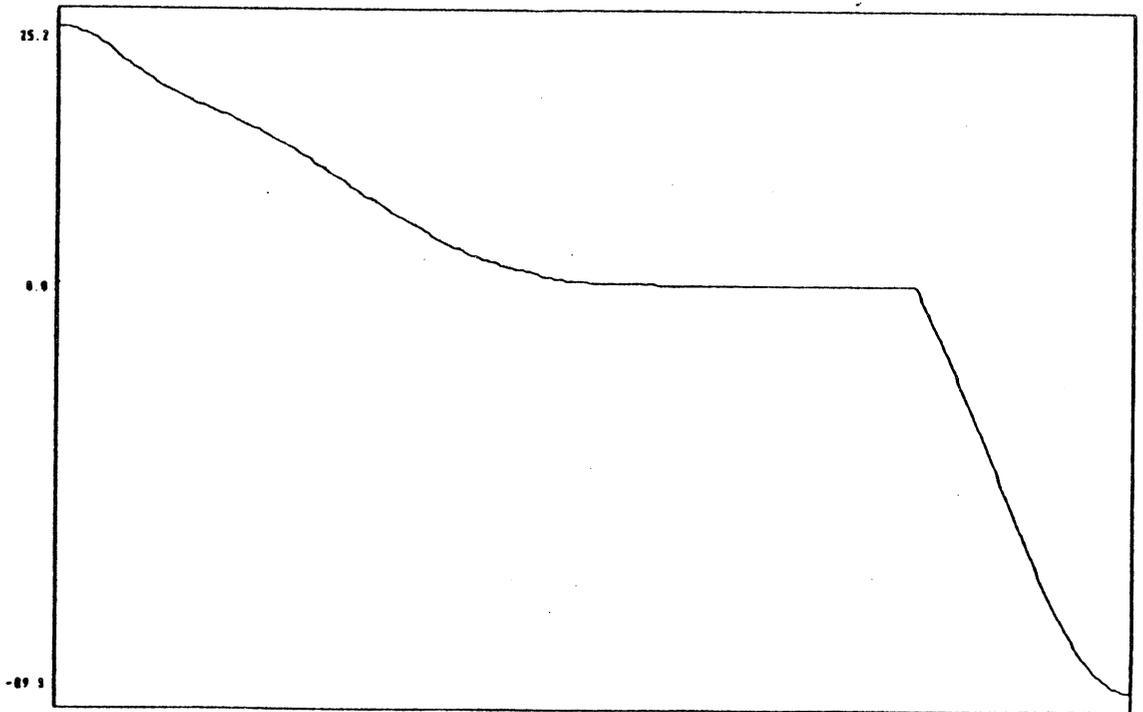


(d) $\rho=10^7$, $r=10^2$

Fig.6.

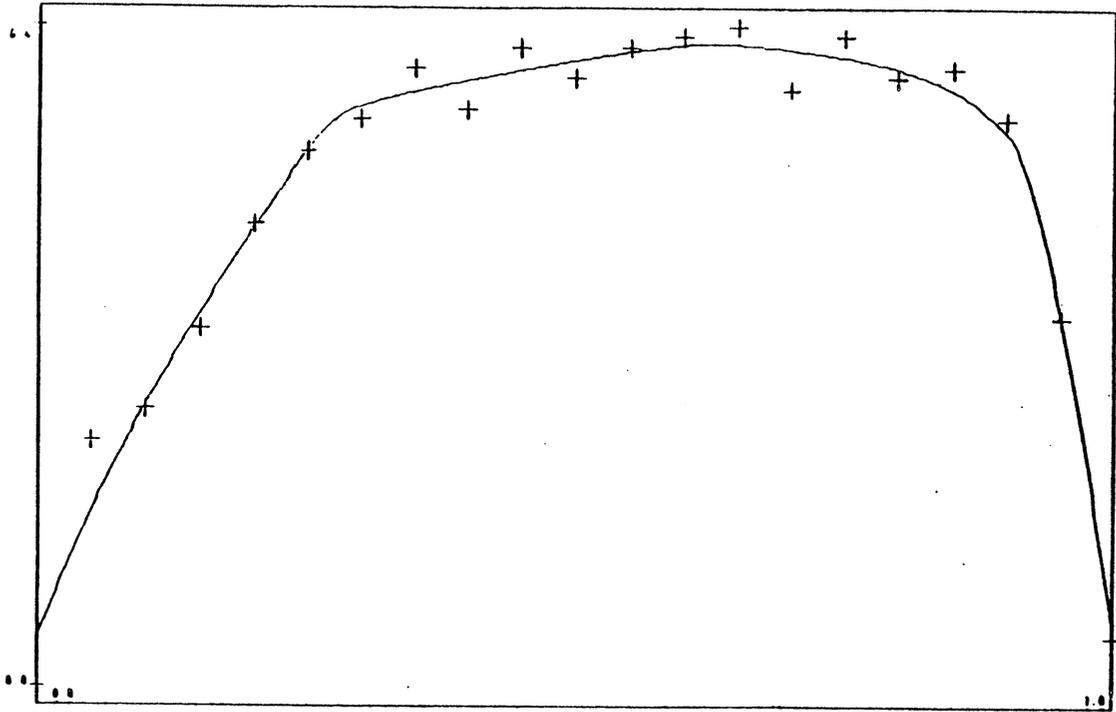


(a)

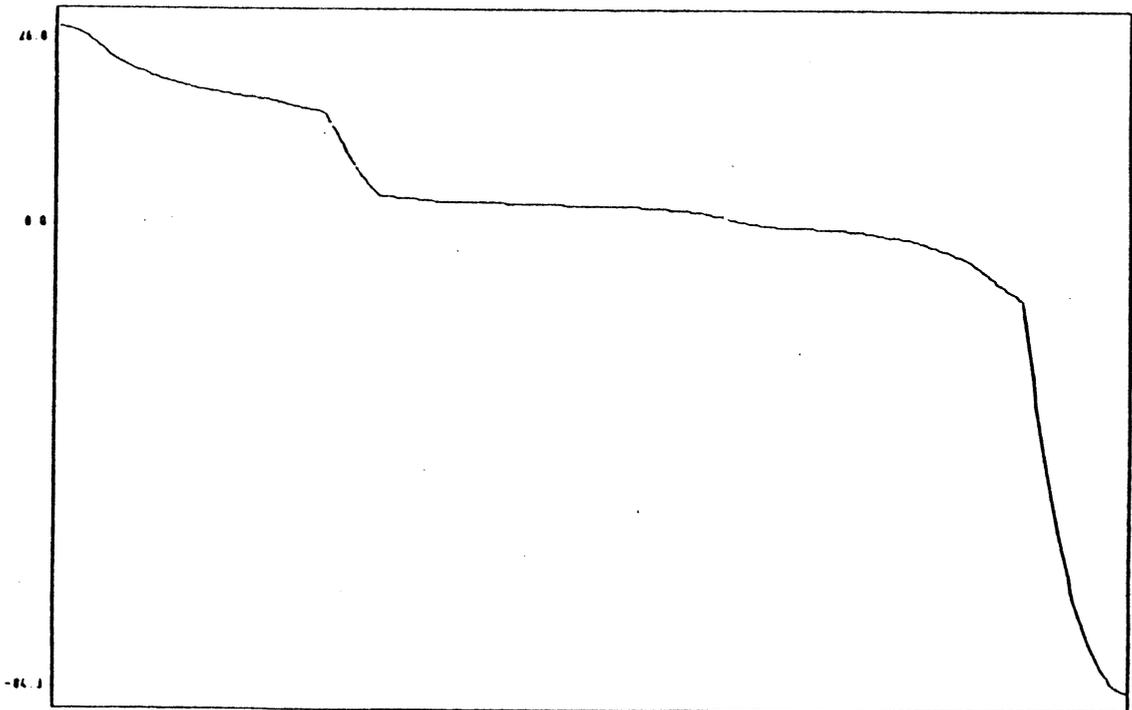


(b)

Fig.7. Lissage des données de l'exemple 1.
(a) F.S.Hyperbolique sous tension
(b) Derivée première de la spline



(a)



(b)

Fig.8. Lissage des données de l'exemple 1.
(a) F.S.Cubique avec poids
(b) Derivée première de la spline

La Fig.7(a) montre la spline hyperbolique sous tension après 3 itérations (9.3) et dont la valeur de ρ est donnée par la méthode de validation croisée. En bas, la dérivée première décroissante confirme le caractère concave de la fonction.

Nous avons procédé de la même manière pour les splines cubiques avec poids (Fig. 8) , mais dans ce cas-là il a fallu 2 mises à jour des poids (9.3) pour atteindre le résultat.

• ***Exemple 2***

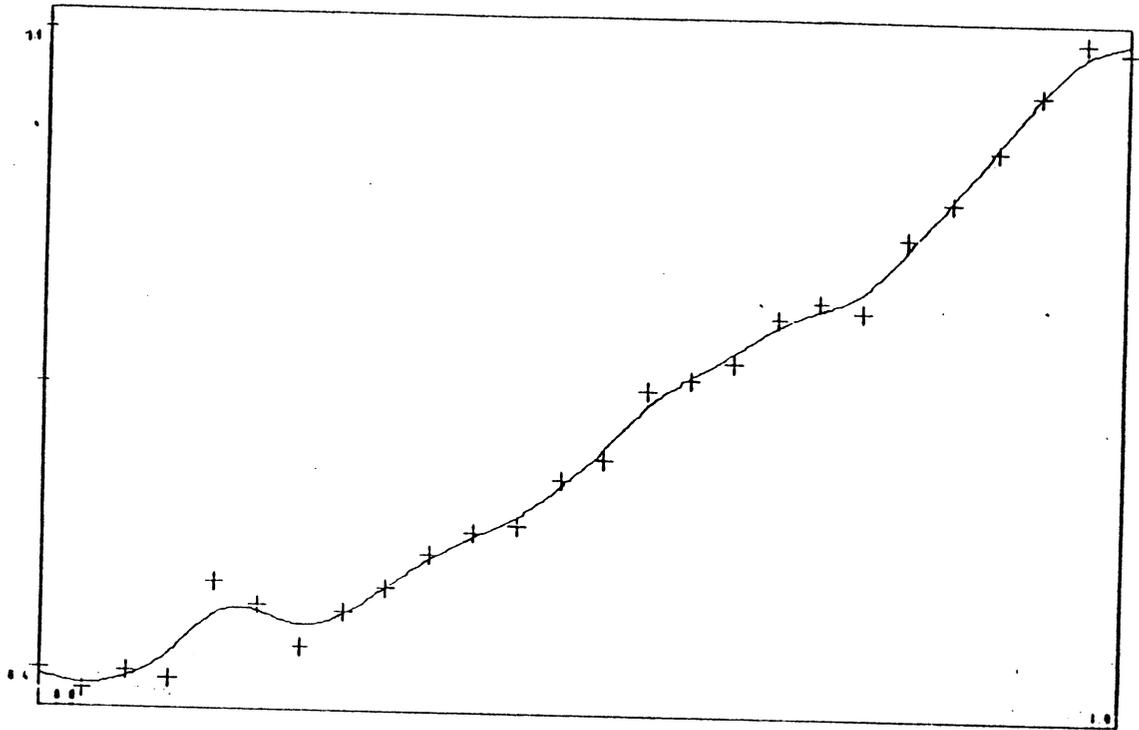
Reprenons l'ensemble des données de Lafata & Rosen [55]. Le but est de trouver des splines de lissage monotones.

i	t_i	z_i	i	t_i	z_i
1	0.00	0.431	14	0.52	0.669
2	0.04	0.409	15	0.56	0.746
3	0.08	0.429	16	0.60	0.760
4	0.12	0.422	17	0.64	0.778
5	0.16	0.530	18	0.68	0.828
6	0.20	0.505	19	0.72	0.846
7	0.24	0.459	20	0.76	0.836
8	0.28	0.499	21	0.80	0.916
9	0.32	0.526	22	0.84	0.956
10	0.36	0.563	23	0.88	1.014
11	0.40	0.587	24	0.92	1.076
12	0.44	0.595	25	0.96	1.134
13	0.48	0.647	26	1.00	1.124

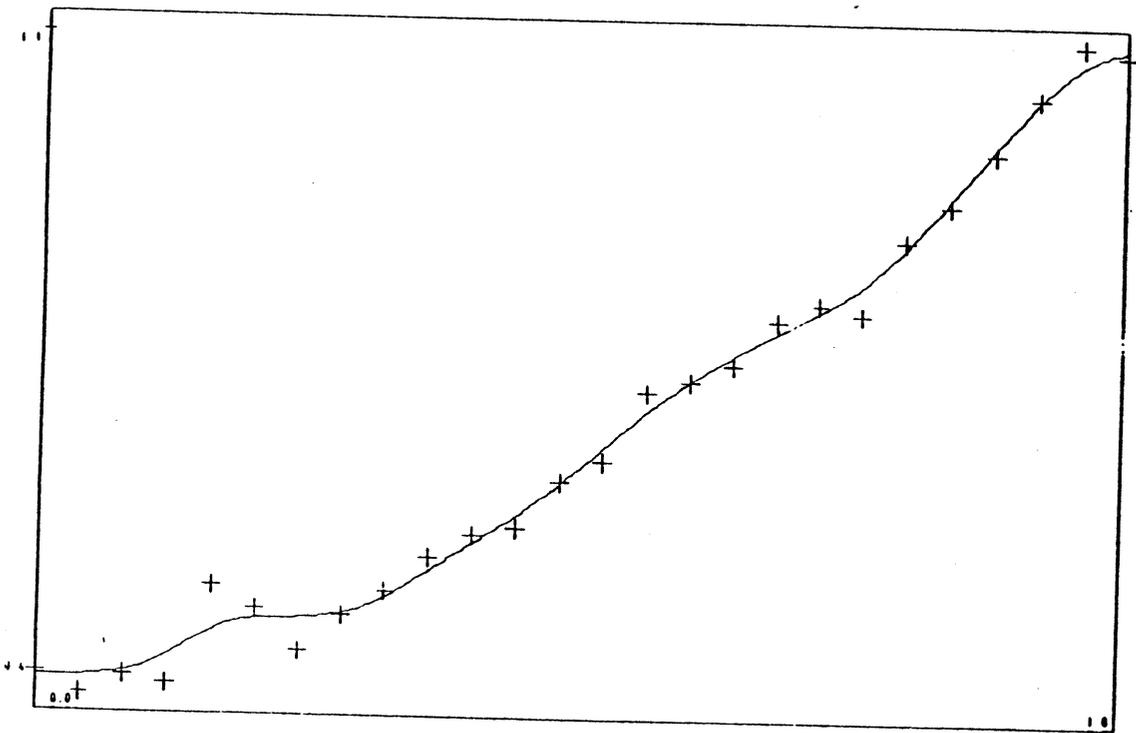
Tableau 5. Données pour l'exemple 2.

On s'aperçoit dans la Fig. 9(a) que la spline cubique ne respecte pas la condition de monotonie. En itérant deux fois sur les paramètres de tension , les cubiques pénalisées et avec poids sont monotones. Voir Figs. 9(b) et 10(a). Le lissage étant plus marqué pour la spline avec poids.

Au bout de quatre itérations la monotonie a été acquise pour les splines hyperboliques (Fig.10(b)).

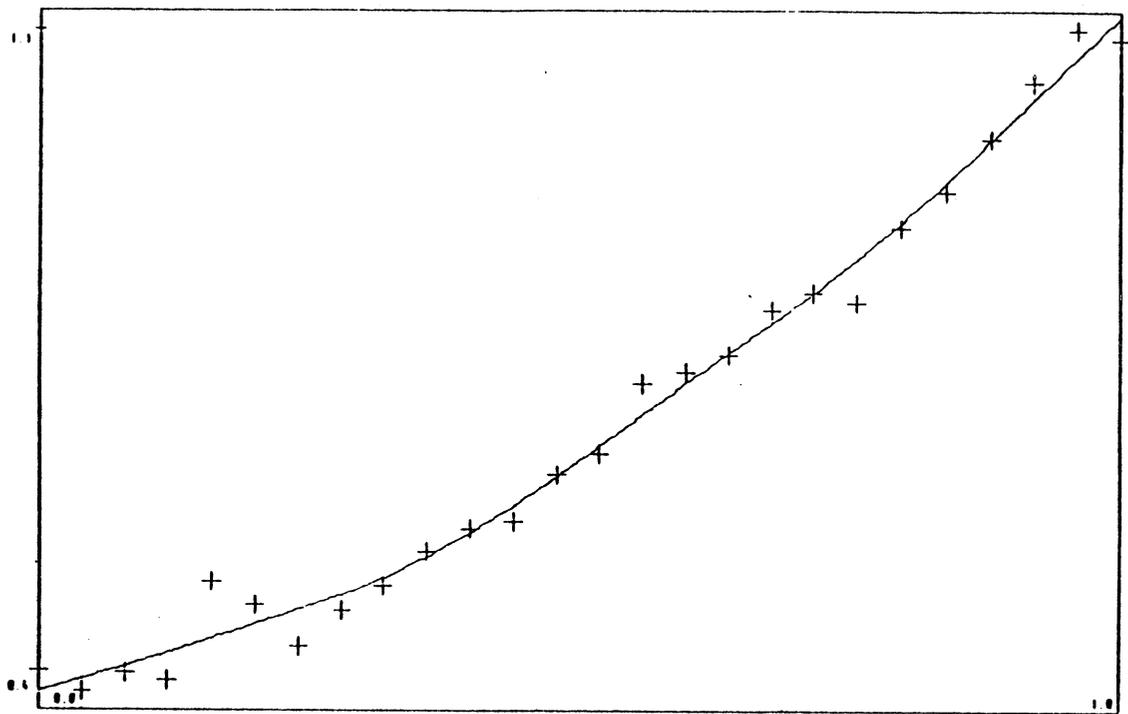


(a)

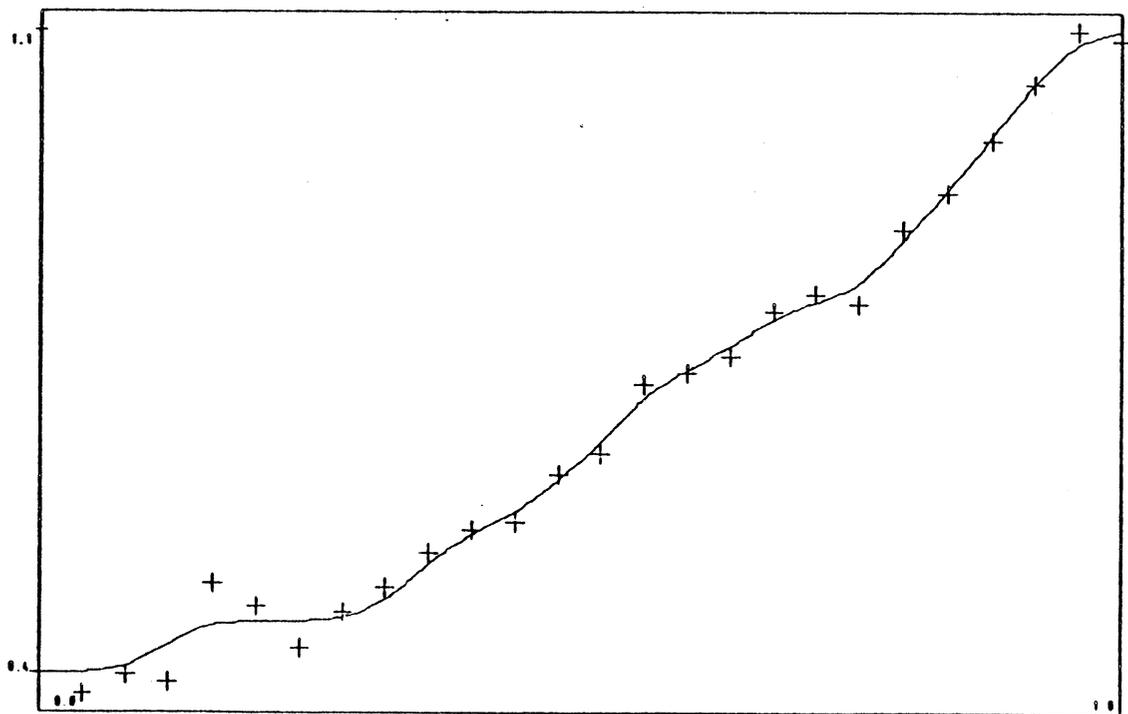


(b)

Fig. 9. Lissage de données de l'exemple 2.
(a) F.S.Cubique naturelle
(b) F.S.Cubique pénalisée



(a)



(b)

Fig.10. Lissage de données de l'exemple 2.

(a) F.S.Cubique avec poids

(b) F.S.Hyperbolique sous tension

Regardons maintenant le tableau 6, où on donne les valeurs de ρ optimal et le temps total de calcul pour chacune des méthodes appliquées à l'exemple 2.

	Cubique	C.Nielson	C.poids	C. pénalisée	Hyperb.
ρ optim.	56795.68	14056.85	935.60	19875.3	56781.22
temps (min.)	4.538	7.996	8.596	15.734	22.254

Tableau 6.

Le nombre d'itérations sur la tension pour les cubiques de Nielson, avec poids et pénalisées a été de deux, et pour la spline hyperbolique de quatre.

Le temps consommé varie entre 8 et 22.25 minutes, et il est assez grand, car à chaque fois que la tension change la méthode de validation croisée est appliquée pour déterminer ρ optimal.

Or le calcul de la trace de la matrice $A(\rho)$ demande l'évaluation de $n=26$ splines, et le nombre d'itérations de la méthode de la section dorée se situe entre 14 et 20, c'est-à-dire que pour calculer ρ optimal pour un jeu de paramètres de tension donné on a besoin de 364 à 520 évaluations de la spline.

Du fait que la tension n'est pas uniforme on n'a pas pu trouver des approximations des valeurs propres comme Utreras [108] a fait pour les splines cubiques naturelles. D'autre part, la méthode de Reinsch n'était pas applicable à cet exemple car on ne connaît pas l'écart type.

• Exemple 3

Dans de multiples problèmes de mécanique les profils de courbes doivent être convexes ou concaves.

Les mesures, dont un ensemble typique de données, apparaît dans l'angle supérieur droit de la Fig. 11(a), dépendent d'un paramètre

uniforme (le temps) et sont entachées d'erreurs. Nous sommes amenés à considérer la courbe plane de lissage sous forme paramétrée.

Les mécaniciens s'intéressent au comportement de la dérivée première, qui pour les splines cubiques présente des oscillations. Voir Fig. 11(a). En appliquant les splines cubiques de Nielson on a réussi à la rendre décroissante (Fig 11(b)), c'est-à-dire que la fonction sera concave.

Pour évaluer le paramètre de lissage on s'est servi de la méthode de Reinsch puisqu'on connaît une estimation de l'écart type.

• Comparaison du temps de calcul

Nous voulons donner une idée de l'évolution du temps de calcul des valeurs de la fonction spline et des dérivées aux noeuds, pour des paramètres de tension et de lissage qui restent fixes.

Pour cela on a bruité la fonction

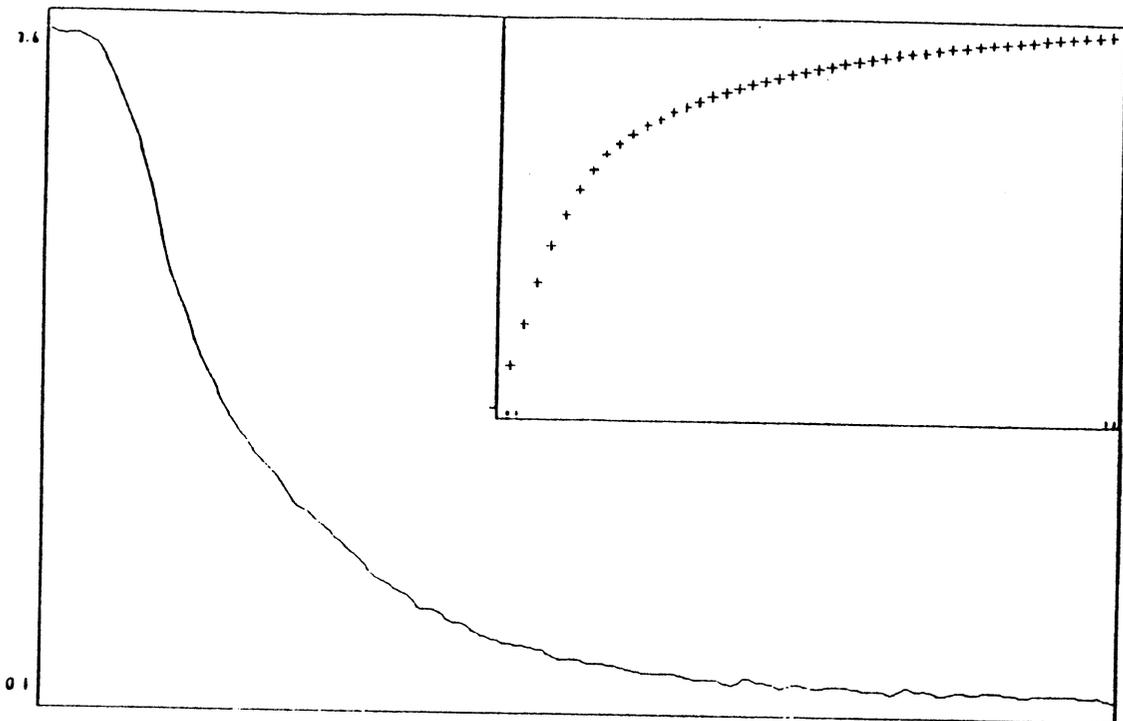
$$g(t) = (-9t+2) / (4t+5) \text{ sur } [-1,8]$$

avec $n=25,50$ et 100 points équidistants.

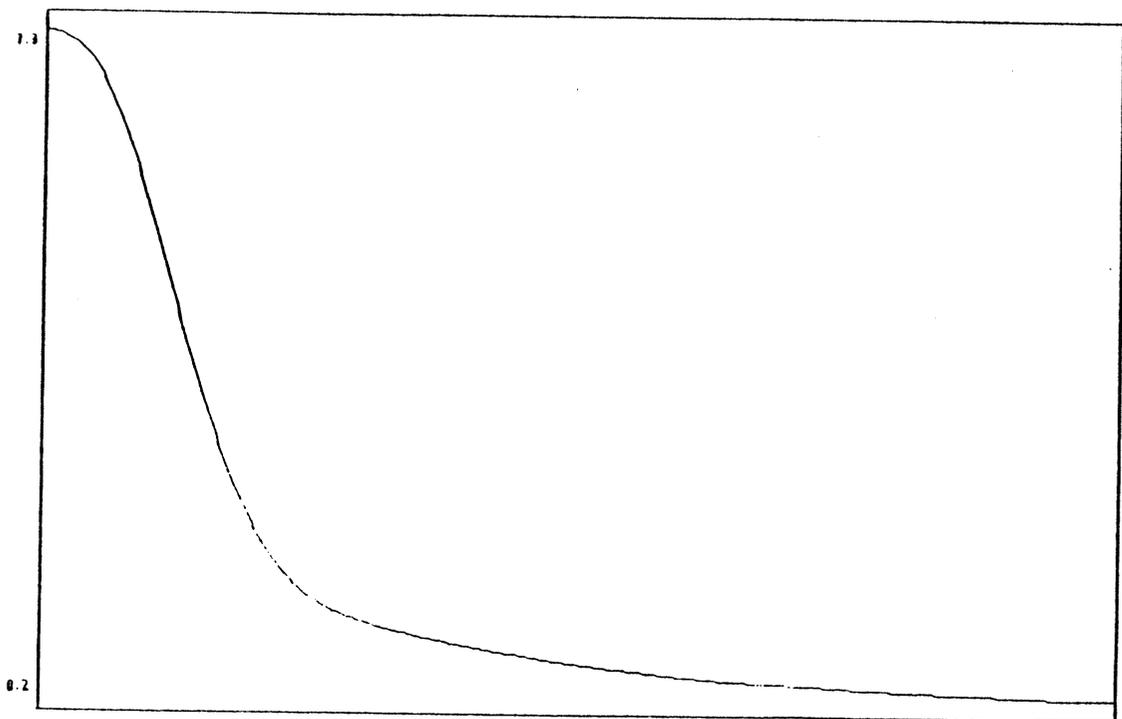
L'erreur sur la valeur de g aux noeuds a pour écart type 0.2. On a pris $\rho=10^4$ et la tension uniforme de telle sorte que $rh=5$ et $rh=15$.

		Cubique	C.Nielson	C.pénalisée	Hyperbolique	
n	rh	0	5 et 15	5 et 15	5	15
25	0.26	0.26	0.28	0.84	0.50	0.60
50	0.54	0.54	0.58	1.71	1.02	1.22
100	1.10	1.10	1.16	3.42	2.04	2.46

Tableau 7. Temps (en secondes).



(a)



(b)

Fig. 11. Lissage de données avec condition de concavité
(a) Dérivée première de la F.S.Cubique paramétrée
(b) Dérivée première de la F.S.Cubique de Nielson paramétrée

A l'exception des cubiques pénalisées, pour lesquelles on utilise la décomposition Cholesky, les autres splines ont été calculées par la méthode de Gauss appliquée aux systèmes tridiagonaux par blocs 2×2 .

D'après le tableau 7, on constate que :

- Si la tension augmente, le calcul pour la spline hyperbolique s'accroît. Ainsi, pour $rh=5$ on consomme presque 2 fois le temps que demande la spline cubique. Nous avons remarqué, par contre, que pour les différentes cubiques sous tension le changement de ces paramètres n'a pas d'influence sur la rapidité des calculs.
- Les cubiques pénalisées prennent plus de temps que les autres méthodes car la décomposition Cholesky demande plus d'opérations.
- Dans les situations où la tension n'est pas uniforme, nous nous sommes aperçus que le calcul de la spline avec poids est légèrement moins rapide que celui de la cubique de Nielson.

III.10. CONCLUSIONS

Nous avons étudié différents types de fonctions splines sous tension pour construire des courbes monotones ou convexes lorsqu'on interpole ou lisse des données.

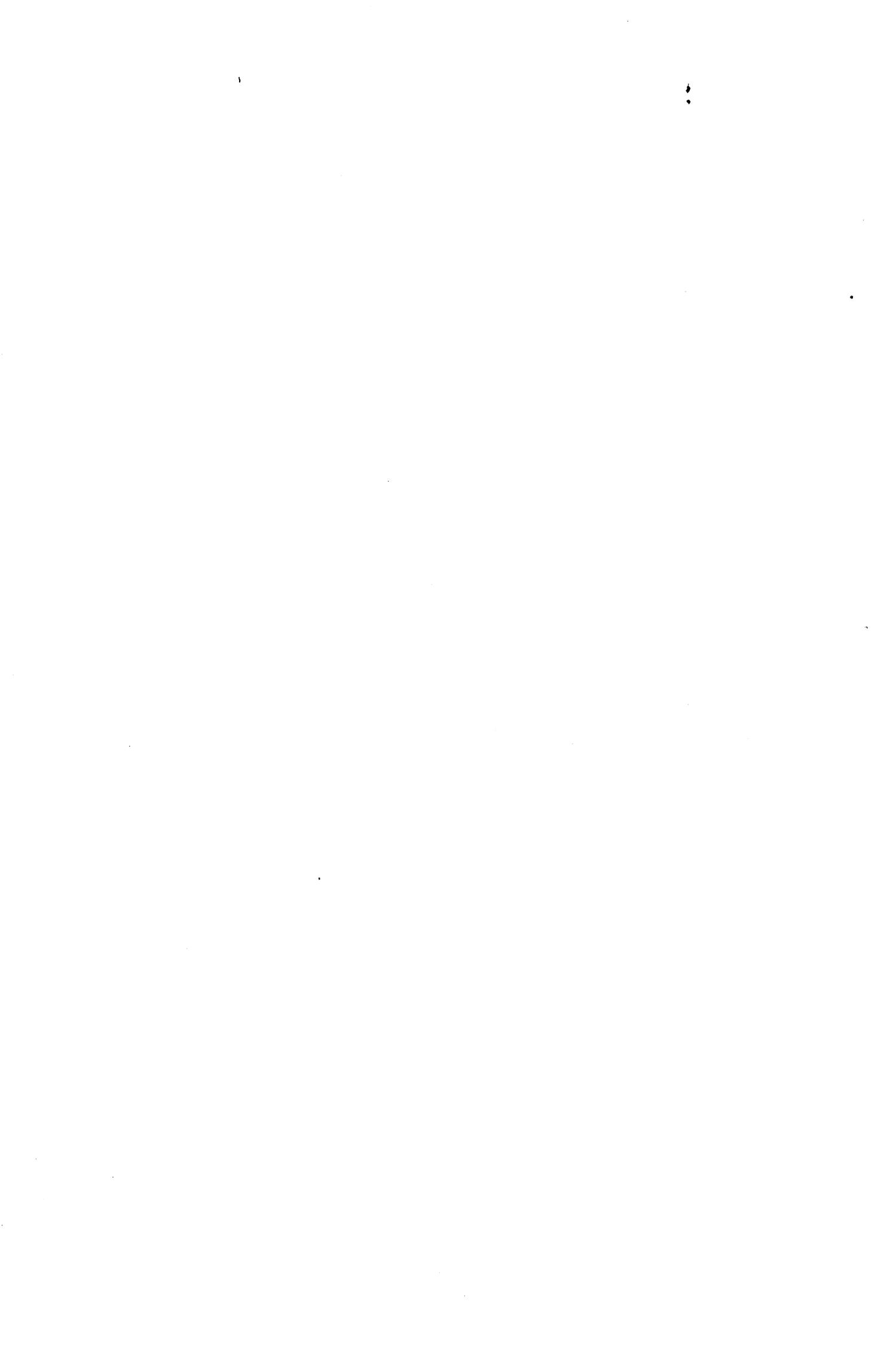
On peut dire que les splines cubiques avec poids donnent des bons résultats pour l'interpolation et aussi pour le lissage. Bien qu'elles ne gardent pas la continuité C^2 comme les splines hyperboliques sous tension, leur coût de calcul est très inférieur.

Si la courbe est considérée sous la forme paramétrée alors on peut se servir des ν - splines de Nielson.

Les splines cubiques pénalisées d'interpolation exigent des hypothèses plus fortes que la monotonie ou la convexité des données pour préserver de telles propriétés. Il semble qu'elles produisent des résultats satisfaisants, pour le lissage avec la condition de monotonie, mais le calcul est plus cher que pour les autres cubiques sous tension car les systèmes linéaires à résoudre sont plus compliqués.

CHAPITRE IV

Fonctions Splines avec Contraintes de Forme via la Programmation Mathématique



IV. 1. INTRODUCTION

Dans ce chapitre nous allons étudier d'un point de vue variationnel les fonctions splines polynomiales sous contraintes de forme (positivité, monotonie, convexité).

Parmi les spline polynomiales, les cubiques sont les plus utilisées pour l'interpolation et le lissage des données. De plus, l'allure et l'expression de leurs dérivées sont simples. Ces splines sont donc bien placées pour construire des interpolantes ou des approximantes qui conservent les caractéristiques de forme de la fonction sous-jacente aux données.

Dans la littérature, on trouve plusieurs applications de la programmation mathématique pour résoudre ce type de problème. Ainsi, pour le lissage de données sous la contrainte de convexité, Rabinowitz [94], Lafata, et Rosen [55] utilisent la norme L_1 ou L_∞ , c'est-à-dire que le problème est traité dans le cadre de la programmation linéaire. Par contre, Amos et Slater [2], Fu-qui-xiang [43] et Dierckx [29], minimisent par rapport à la norme L_2 (problème de programmation quadratique).

Considérons n abscisses distinctes ordonnées t_j d'un intervalle réel $[a,b]$,

$$a < t_1 < t_2 < \dots < t_n < b \quad (n \geq 2)$$

et $H^2[a,b]$ l'espace défini par

$$H^2[a,b] = \{ f : [a,b] \rightarrow \mathbb{R} \mid f, f' \text{ absolument continues, } f'' \in L_2[a,b] \}$$

Soit $z = [z_1, z_2, \dots, z_n] \in \mathbb{R}^n$ et notons

$$E(x) = \int_a^b [x''(t)]^2 dt, \quad J(x) = \sum_{i=1}^n [x(t_i) - z_i]^2$$

Il est bien connu que le problème d'interpolation

$$(1.1) \quad \left\{ \begin{array}{l} \text{minimiser } E(x) \\ \text{avec } x \in H^2[a,b] \text{ vérifiant } x(t_i) = z_i \quad (i=1,2,\dots,n) \end{array} \right.$$

et le problème de lissage

$$(1.2) \quad \text{minimiser } E(x) + \rho J(x) \quad \text{avec } x \in H^2[a,b]$$

ont respectivement, une solution unique qui appartient à l'espace des fonctions splines cubiques naturelles ([56].) Le coefficient ρ est un paramètre qui réalise un compromis entre le lissage de la fonction (caractérisé par $E(x)$) et l'approximation des données (caractérisée par $J(x)$)

En tenant compte des contraintes de forme, on ne minimise plus sur tout l'espace $H^2[a,b]$ mais sur un sous-ensemble convexe C de celui-ci, qui est défini par :

$$(1.3) \quad C = \{ f \in H^2[a,b] / f(t) \geq 0, \forall t \in [a,b] \} \quad (\text{positivité})$$

$$(1.4) \quad C = \{ f \in H^2[a,b] / f'(t) \geq 0, \forall t \in [a,b] \} \quad (\text{monotonie})$$

$$(1.5) \quad C = \{ f \in H^2[a,b] / f''(t) \geq 0, \forall t \in [a,b] \} \quad (\text{convexité})$$

Nous sommes donc intéressés par les problèmes suivants :

$$(1.6) \quad \left\{ \begin{array}{l} \text{minimiser } E(x) \\ \text{avec } x \in C \text{ vérifiant } x(t_i) = z_i \quad (i=1,2,\dots,n) \end{array} \right.$$

et

$$(1.7) \quad \text{minimiser } E(x) + \rho J(x) \quad \text{avec } x \in C$$

Hornung ([51], [52]) a traité le problème (1.6) d'interpolation monotone ou convexe mais sans développer des algorithmes de calcul. Wright et Wegman [120] sont arrivés à des conclusions

théoriques semblables. Pour l'interpolation avec la contrainte de monotonie, Utreras et Celis [111] ont cherché la solution dans l'espace des polynômes cubiques par morceaux de classe C^1 et non pas dans l'ensemble C (1.4).

Dans le cas du lissage des données avec la condition de monotonie, Nagelkerke et Oosting [76] préfèrent discrétiser la contrainte $x'(t) \geq 0$ de la manière suivante :

$$\left\{ \begin{array}{l} x(t_i) \leq x(t_{i+1}) \quad i=1,2,\dots,n-1 \\ x'(t_i) \geq 0 \end{array} \right.$$

Utreras [110] donne des résultats théoriques sur l'existence, la caractérisation et la convergence des splines cubiques, solutions du problème (1.7).

En utilisant l'analyse convexe, Laurent [57] propose un algorithme de type dual pour le calcul des fonctions splines avec contraintes type inégalité. Cette méthode peut s'appliquer à une ou deux dimensions et les contraintes sont considérées de façon continue (c'est-à-dire sans être discrétisées).

La partie principale de ce chapitre est consacrée à la mise en oeuvre numérique de l'algorithme de Laurent pour le lissage avec la contrainte de positivité ou de monotonie.

IV.2. INTERPOLATION ET LISSAGE DES DONNEES AVEC CONTRAINTE DE MONOTONIE (DANS UN ESPACE DE FONCTIONS SPLINES).

Considérons un ensemble de données $D = ((t_i, z_i))_{i=1, \dots, n}$ où $t_1 < t_2 < \dots < t_n$. Les données seront dites *croissantes* si $z_1 \leq z_2 \leq \dots \leq z_n$; de manière similaire on peut définir les données *décroissantes*. Dans les deux cas on dira que les données sont *monotones*.

On définit l'ensemble $\Pi_3^1[t_1, t_n]$ de la manière suivante :

$s(t) \in \Pi_3^1[t_1, t_n]$ si et seulement si :

(i) $s(t)$ est un polynôme de degré ≤ 3 dans chaque intervalle $]t_i, t_{i+1}[$, $i=1, 2, \dots, n-1$

(ii) $s(t) \in C^1[t_1, t_n]$.

La fonction $s(t)$ peut s'exprimer sur l'intervalle $]t_i, t_{i+1}[$ en termes des données d'interpolation z_i, z_{i+1} et des dérivées premières aux noeuds $\lambda_i = s'(t_i)$, $\lambda_{i+1} = s'(t_{i+1})$,

$$s(t) = z_i + \lambda_i(t-t_i) + \frac{m_i - \lambda_i}{h_i}(t-t_i)^2 + \frac{\lambda_{i+1} - 2m_i + \lambda_i}{h_i^2}(t-t_i)^2(t-t_{i+1}) \quad (2.1)$$

où $h_i = t_{i+1} - t_i$
 $i=1, 2, \dots, n-1$

$$m_i = (z_{i+1} - z_i) / h_i$$

Nous allons étudier l'interpolation et le lissage avec contrainte de monotonie dans l'espace $\Pi_3^1[t_1, t_n]$.

2. 1. Interpolation monotone

Rappelons d'abord un résultat de Fritsch et Carlson [41] qui concerne les conditions nécessaires et suffisantes sur les dérivées aux noeuds $\lambda_1, \lambda_2, \dots, \lambda_n$ pour que la fonction $s(t) \in \Pi_3^1$ soit une interpolante monotone.

Proposition 2.1

Considérons l'intervalle $I_j = [t_j, t_{j+1}]$ et le polynôme cubique $q(t)$ tel que

$$q(t_j) = z_j, \quad q'(t_j) = \lambda_j \quad j = i, i+1$$

Alors une condition nécessaire et suffisante pour que $q(t)$ soit monotone sur I_j est que

- (i) $\lambda_j = \lambda_{j+1} = 0$ si $z_j = z_{j+1}$
 (ii) $(\lambda_j/m_j, \lambda_{j+1}/m_j) \in \mathcal{M}$ si $z_j \neq z_{j+1}$

où $\mathcal{M} = \{(u,v) / u \geq 0, v \geq 0, (u+v \leq 0 \text{ ou } u^2 + v^2 + uv - 6(u+v) + 9 \leq 0)\}$

La région \mathcal{M} est représentée par la Fig.3 du Chap.I. §4.1.

Le problème que nous allons considérer est le suivant :

$$(2.2) \left\{ \begin{array}{l} \text{minimiser } \frac{1}{2} \int_{t_1}^{t_n} [x''(t)]^2 dt \text{ avec } x \in \Pi_3^1[t_1, t_n] \\ \text{vérifiant } x(t_i) = z_i \text{ (} i=1, 2, \dots, n \text{) et} \\ x'(t) \geq 0 \quad \forall t \in [t_1, t_n] \end{array} \right.$$

A partir de la représentation (2.1) nous pouvons écrire d'une manière explicite la forme quadratique en termes des n inconnues $\lambda_j (j=1, 2, \dots, n)$.

$$(2.3) \quad \frac{1}{2} \int_{t_1}^{t_n} [x''(t)]^2 dt = \Lambda^T Q \Lambda - 6R^T \Lambda + 6k^T k$$

où C est le convexe

$$(2.5) \quad C = \{ \Lambda \in \mathbb{R}^n / \lambda_j = \lambda_{j+1} = 0 \text{ si } m_j = 0 \text{ ou}$$

$$\left(\frac{\lambda_j}{m_j} ; \frac{\lambda_{j+1}}{m_{j+1}} \right) \in \mathcal{M} \}$$

Dans [111] on résout (2.4) par deux méthodes : la première : des directions admissibles ; la seconde : de relaxation.

2.2. Lissage monotone

Maintenant on va étudier le problème

$$(2.6) \quad \left\{ \begin{array}{l} \min_a \int^b [x''(t)]^2 dt + \rho \sum_{i=1}^n (x(t_i) - z_i)^2 \\ \text{avec } x \in \Pi_3^1[t_1, t_n] \text{ vérifiant} \\ \quad \quad \quad x(t_i) \leq x(t_{i+1}) \quad (i=1, 2, \dots, n-1) \text{ et} \\ \quad \quad \quad x'(t_i) \geq 0 \quad \quad \quad (i=1, 2, \dots, n) \end{array} \right.$$

En employant la représentation (2.1) pour $x(t)$ on peut expliciter la forme quadratique en termes des inconnues $y_i = x(t_i)$, $\lambda_i = x'(t_i)$ ($i=1, 2, \dots, n$). Alors on est ramené à résoudre un problème avec contraintes dans \mathbb{R}^{2n} .

Les auteurs de l'article [76] considèrent $\beta_i = x''(t_i)$ $i=1, 2, \dots, n$ comme inconnues, au lieu des λ_i , ce qui revient à supposer que la solution de (2.6) est de classe C^2 . Ils font une erreur car en tout t_i où la contrainte $x'(t_i) \geq 0$ sera active, on aura une discontinuité de la dérivée seconde ; donc la solution n'appartiendra pas à l'espace de fonctions splines naturelles.

IV.3. LISSAGE DE DONNÉES AVEC CONTRAINTES DE CONVEXITÉ LOCALE

L'imposition de la contrainte de convexité à une fonction spline peut permettre d'éliminer des oscillations indésirables qui apparaissent lorsqu'on interpole ou lisse des données.

Dans ce paragraphe nous allons décrire les solutions proposées dans le contexte de la programmation linéaire et de la programmation quadratique. L'utilisation des splines cubiques permet une étude simple du signe de la dérivée seconde, car entre deux noeuds celle-ci se réduit à un morceau de droite. Mais le choix de la base de représentation de ces fonctions est important pour le calcul numérique ; par exemple la forme suivante :

$$(3.1) \quad s(t) = \sum_{i=0}^3 a_i t^i + \sum_{i=1}^3 A_i (t-t_i)_+^3$$

$$\text{où } (x)_+^k = \begin{cases} x^k & \text{si } x > 0 \\ 0 & \text{si } x \leq 0 \end{cases}$$

conduit à des erreurs trop grandes ([56], pp279). Différents auteurs conseillent de prendre plutôt la base de B-splines.

3.1. Rappel sur les B-splines cubiques ([24])

Soit $a = u_0 < u_1 < \dots < u_m < u_{m+1} = b$, une partition de $[a;b]$, alors une fonction $s(t)$ définie sur $[a,b]$ est une **spline cubique avec noeuds u_j** ($j=1, \dots, m$) si et seulement si

(i) dans chaque intervalle $[u_j, u_{j+1}]$, $j=1, 2, \dots, m-1$, $s(t)$ est un polynôme de degré ≤ 3 .

(ii) $s(t) \in C^2[a,b]$

La fonction **B-spline cubique normalisée** $N_j(t)$ est une spline définie par

$$(3.2) \quad N_j(t) = (u_{j+4} - u_j) \Delta_{j, j+1, \dots, j+4}^4 (t-u)_+^3$$

où $\Delta_{j,j+1,\dots,j+4}^4 g(u)$ est la différence divisée d'ordre 4 de la fonction $g(u)$ calculées aux noeuds u_j, \dots, u_{j+4} .

Ces splines normalisées possèdent entre autres les propriétés suivantes :

1. $N_j(t) \geq 0$ (positivité)
2. $\sum_{j=1}^m N_j(t) = 1$ (partition de l'unité)
3. $N_j(t) = 0$ si $t \notin [u_j, u_{j+4}]$ (support local)
4. Formules de récurrence très stables pour l'évaluation de ces fonctions et de leurs dérivées.

Pour représenter d'une manière unique une fonction spline cubique $s(t)$ avec noeuds u_1, \dots, u_m dans la base de B-spline, une solution possible consiste à introduire 8 noeuds tels que

$$u_{-3} \leq u_{-2} \leq u_{-1} \leq u_0 = a$$

$$b = u_{m+1} \leq u_{m+2} \leq u_{m+3} \leq u_{m+4}$$

alors

$$(3.3) \quad s(t) = \sum_{j=-3}^m \alpha_j N_j(t) ; t \in [a, b]$$

3.3. Une application de la programmation linéaire

Le problème considéré par Lafata et Rosen [55], est le suivant :

Soit $((t_i, z_i))_{i=1,2,\dots,n}$ un ensemble de données et

$a = u_0 < u_1 < \dots < u_m < u_{m+1} = b$ une partition de $[a, b]$. On cherche une spline cubique $s(t)$ avec noeuds u_j ($j=1,2,\dots,m$) telle que

$$(3.4) \quad \max_{1 \leq i \leq n} |z_i - s(t_i)| \quad (\text{norme } L_\infty)$$

ou

$$(3.5) \quad \sum_{i=1}^n |z_i - s(t_i)| \quad (\text{norme } L_1)$$

soit minimal et vérifie

$$(3.6) \quad s''(t_i) \geq 0 \quad i=1,2,\dots,n. \quad \square$$

Si l'on utilise la représentation (3.3) pour $s(t)$ alors (3.4) et (3.5) deviennent respectivement :

$$(3.7) \quad \max_{1 \leq i \leq n} |z_i - \sum_{j=-3}^m \alpha_j N_j(t_i)|$$

et

$$(3.8) \quad \sum_{i=1}^n |z_i - \sum_{j=-3}^m \alpha_j N_j(t_i)|$$

Bornons-nous à étudier le cas de la norme L_∞ , car pour la norme L_1 On procède de manière analogue.

On considère d'abord le problème sans contraintes (3.4) en introduisant une nouvelle variable réelle γ :

$$(3.9) \quad \min_{\gamma, \alpha} \left\{ \gamma \mid -\gamma \leq z_i - \sum_{j=-3}^m \alpha_j N_j(t_i) \leq \gamma, \quad i=1,2,\dots,n \right\}$$

où

$$\alpha^T = [\alpha_{-3} \quad \alpha_{-2} \quad \dots \quad \alpha_m].$$

On remarque que si $[\gamma^*, \alpha^*]$ est la solution optimale de (3.9) alors γ^* est le montant de (3.4) et α^* est sa solution optimale.

Soient

$$F = \begin{bmatrix} N_{-3}(t_1) & \dots & N_{-3}(t_n) \\ N_{-2}(t_1) & \dots & N_{-2}(t_n) \\ \vdots & & \vdots \\ N_m(t_1) & \dots & N_m(t_n) \end{bmatrix} \quad (m+4) \times n$$

$$(3.10) \quad c^T = [-z^T \ z] \in \mathbb{R}^{2n} \quad , \quad w^T = [\alpha^T \ \gamma] \in \mathbb{R}^{m+5}$$

$$b^T = [0 \dots 0 \ 1] \in \mathbb{R}^{m+5} \quad , \quad e^T = [1 \dots 1] \in \mathbb{R}^n$$

et

$$A = \begin{bmatrix} F & -F \\ e^T & e^T \end{bmatrix} \quad (m+5) \times 2n$$

alors l'expression matricielle de (3.9) est

$$(3.11) \quad \min_{w \in \mathbb{R}^{m+5}} \quad \{ b^T w / A^T w \geq c \}$$

Le problème dual de (3.11) est le suivant :

$$(3.12) \quad \max_{y \in \mathbb{R}^n} \quad \{ c^T y / Ay = b, y \geq 0 \}$$

L'introduction de la condition (3.6) modifie la matrice A et le vecteur c de la manière suivante :

$$(3.13) \quad A = \begin{bmatrix} F & -F & F'' \\ e^T & e & 0 \end{bmatrix}$$

$$c^T = [z^T \ -z^T \ \underbrace{0 \ 0 \dots 0}_n \ 0 \ 0]$$

n

avec

$$F'' = \begin{bmatrix} N''_{-3}(t_1) & \dots & N''_{-3}(t_n) \\ N''_{-2}(t_1) & \dots & N''_{-2}(t_n) \\ \vdots & & \vdots \\ N''_m(t_1) & \dots & N''_m(t_n) \end{bmatrix}$$

Lafata et Rosen [55] résolvent le problème (3.12) en profitant du fait que la matrice A n'a que m+1 lignes.

Remarques

- 1) Rabinowitz [94] considère la base (3.1) au lieu de celle des B-splines.
- 2) La même démarche peut être utilisée pour une contrainte de monotonie. (Il suffit de changer la matrice F'').

3.3. La spline des moindres carrés

Dierckx [29] étudie le problème de programmation quadratique suivant :

Soit $((t_i, z_i)) \ i=1,2,\dots,n$ un ensemble de données et considérons une partition de l'intervalle $[a,b]$, $a = u_0 < u_1 < \dots < u_m < u_{m+1} = b$.
On cherche une fonction spline cubique $s(t)$ avec noeuds u_j ($j=1,\dots,m$) telle que

$$(3.14) \quad \sum_{i=1}^n [z_i - s(t_i)]^2 \text{ soit minimal}$$

et vérifie

$$(3.15) \quad s''(u_j) \geq 0 \quad j=0,1,\dots,m+1 \quad . \quad \square$$

En dérivant deux fois la fonction $s(t)$ représentée par (3.3) on obtient :

$$(3.16) \quad s''(u_k) = \sum_{j=-3}^m \delta_{j,k} \alpha_j \quad k=0,1,\dots,m+1$$

où

$$\begin{aligned} \delta_{j,k} &= 0 && \text{si } j \in [k-4, k] \\ \delta_{k-3,k} &= 6 / [(u_{k+1} - u_{k-2})(u_{k+1} - u_{k-1})] \\ \delta_{k-1,k} &= 6 / [(u_{k+2} - u_{k-1})(u_{k+1} - u_{k-1})] \\ \delta_{k-2,k} &= -(\delta_{k-3,k} + \delta_{k-1,k}) \end{aligned}$$

En réécrivant (3.14), (3.15), on arrive à

$$(3.17) \quad \left\{ \begin{array}{l} \min \sum_{i=1}^n [z_i - \sum_{j=-3}^m \alpha_j N_j(t_i)]^2 \\ \text{avec } \sum_{j=-3}^m \delta_{j,k} \alpha_k \geq 0 \quad k=0,1,\dots,m+1 \end{array} \right.$$

Dierckx [29] propose une version améliorée de la procédure de Theil et Van de Panne pour la résolution numérique de (3.17).

Remarques

1) L'intérêt de cette approche (et celle de la programmation linéaire) est que "normalement" m (le nombre de noeuds u_j) est petit par rapport à n (le nombre de données). Dierckx propose une forme automatique de choisir les noeuds.

2) En suivant le même schéma on peut traiter le cas du lissage avec la contrainte de monotonie. Si l'on utilise les splines paraboliques on a l'avantage que la dérivée première est formée par morceaux de droites.

3) Le problème (3.14), (3.15) a été l'objet des papiers de Amos et Slater [2] et de Fu-Qiang-Xiang [43], mais ils représentent la fonction $s(t)$ sous la forme (31).

IV.4. FONCTIONS SPLINES AVEC CONTRAINTES LINEAIRES

Laurent [57] a étudié les fonctions splines avec contraintes linéaires et proposé un algorithme de calcul. Nous avons travaillé sur la mise au point de cette méthode pour l'interpolation monotone par fonctions splines cubiques [72], et dans ce paragraphe nous présentons l'application au lissage de données avec contraintes de positivité, de monotonie ou de convexité.

Nous rappelons le problème auquel nous nous intéressons :

$$(4.1) \left\{ \begin{array}{l} \text{minimiser } E(x) + \rho J(x) \\ x \in C \end{array} \right.$$

$$\text{avec } E(x) = \int_a^b [x''(t)]^2 dt, \quad J(x) = \sum_{i=1}^n (x(t_i) - z_i)^2, \quad \text{et}$$

$$C = \{ x \in H^2[a,b] / x(t) \geq 0, \forall t \in [a,b] \}$$

$$\text{ou } C = \{ x \in H^2[a,b] / x'(t) \geq 0, \forall t \in [a,b] \}$$

$$\text{ou } C = \{ x \in H^2[a,b] / x''(t) \geq 0, \forall t \in [a,b] \}$$

4.1. Etude de l'existence et de l'unicité de la solution

Considérons les espaces

$$X = H^2[a,b], \quad P = H^0[a,b] \times \mathbb{R}^n \quad (n \geq 2)$$

On munit X du produit scalaire

$$\langle g_1, g_2 \rangle = \sum_{i=0}^2 \int_a^b g_1^{(i)}(t) g_2^{(i)}(t) dt \quad \text{avec } g_1, g_2 \in X$$

et P du produit scalaire

$$\langle [g_1, z_1], [g_2, z_2] \rangle = \int_a^b g_1(t) g_2(t) + e \sum_{i=1}^n z_1^i z_2^i$$

avec $g_1, g_2 \in H^0[a, b]$, $z_1, z_2 \in \mathbb{R}^n$.

X et P sont donc des espaces de Hilbert, et leurs normes associées seront notées respectivement $\| \cdot \|$ et $\| \cdot \|$.

On désigne T l'application de X dans $H^0[a, b]$ définie par

$$T(x) = x''$$

A partir des fonctionnelles linéaires suivantes :

$$\begin{aligned} a_i : X &\rightarrow \mathbb{R} \\ x &\mapsto a_i(x) = x(t_i) \quad i=1, 2, \dots, n \end{aligned}$$

on définit l'application linéaire

$$\begin{aligned} A : X &\rightarrow \mathbb{R}^n \\ x &\mapsto A(x) = [a_1(x), \dots, a_n(x)] \end{aligned}$$

Soit L l'application de X dans P caractérisée par :

$$L(x) = [T(x), A(x)]$$

Soit $z = [z_1, z_2, \dots, z_n]$ un élément de \mathbb{R}^n et 0 le zéro de $H^0[a, b]$.
Notons $p = [0, z]$.

Le problème (4.1) peut donc s'écrire :

$$(4.2) \quad \begin{aligned} &\text{minimiser} \quad \| L(x) - p \| \\ &x \in C \end{aligned}$$

Proposition 4.1

Le problème (4.1) a une solution unique σ dans C.

Preuve

Posons

$$\bar{C} = L(C) = \{ [y,z] \in P / y = T(x), z = A(x) \text{ avec } x \in C \}$$

Le problème (4.2) est équivalent à trouver $\bar{c} \in \bar{C}$ tel que

$$(4.3) \quad \| \bar{c} - p \| = \min_{c \in \bar{C}} \| c - p \|^2$$

Si \bar{C} est fermé alors on sait que (4.3) admet une solution unique \bar{c} qui est la projection orthogonale de p sur \bar{C}

Considérons

$$V = L(X) = \{ [y,z] \in P / y = T(x), z = A(x) \text{ avec } x \in X \}$$

Comme $\mathcal{N}(T)$ est de dimension finie, alors V est fermé (Laurent [56], pp. 200)

Si l'on restreint le domaine d'arrivée de L , on a

$$L_1 : X \rightarrow V \\ x \mapsto L_1(x) = L(x)$$

une application linéaire continue injective de l'espace de Banach X dans l'espace de Banach V . Alors $L_1^{-1} : V \rightarrow X$ est continue.

Donc

$$\bar{C} = (L_1^{-1})^{-1} C$$

est fermé car C est fermé.

Soit \bar{c} la solution unique de (4.3). Comme $\bar{C} = L_1^{-1}(C)$ et L_1 est injective alors il existe un unique élément $\sigma \in C$

$$\sigma = L_1^{-1}(\bar{c})$$

tel que

$$(4.4) \quad \| L(\sigma) - p \| = \min_{x \in C} \| L(x) - p \|. \quad \square$$

4.2. Description de l'algorithme

L'algorithme que nous allons décrire peut s'appliquer plus généralement à la minimisation d'une fonction différentiable fortement convexe avec contraintes du type inégalité.

Notons $I = [a, b]$ et pour $\forall t \in I$ considérons la fonctionnelle linéaire

$$\gamma_t : X \rightarrow \mathbb{R}$$
$$x \mapsto \gamma_t(x) = \begin{cases} x(t) & \text{[contrainte de positivité] ou} \\ x'(t) & \text{[contrainte de monotonie] ou} \\ x''(t) & \text{[contrainte de convexité]} \end{cases}$$

alors

$$C = \{ x \in X / \gamma_t(x) \geq 0 \quad \forall t \in I \}$$

De plus, si

$$\forall t \in I, \quad C_t = \{ x \in X / \gamma_t(x) \geq 0 \}$$

et

$$g(x) = \inf_{t \in I} \{ \gamma_t(x) \}$$

on a les égalités

$$(4.5) \quad C = \bigcap_{t \in I} C_t = \{ x \in X / g(x) \geq 0 \}$$

Notons f la fonction de X dans \mathbb{R} définie par

$$(4.6) \quad f(x) = \| \| L(x) - p \| \|^2$$

Le problème (4.1) est équivalent à

$$(4.7) \quad \min_{x \in C} f(x)$$

Remarques

- 1) La fonction f est convexe et différentiable sur X et fortement convexe sur C (voir [57]).
- 2) $\forall x \in X$ l'ensemble $\{\gamma_t(x), t \in I\}$ est borné.
- 3) $C \neq \emptyset$

Ces remarques montrent que les hypothèses utilisées par Laurent dans sa méthode sont vérifiées. L'algorithme consiste à minimiser f sur une suite de convexes plus simples que C et qui le contiennent.

On aura besoin aussi d'une suite $\{\varepsilon^k\}$ de réels positifs qui converge vers zéro lorsque k tend vers l'infini.

Algorithme 4.1

Itération 0

* Résoudre :

$$(P_0) \quad \text{minimiser } f(x) \quad \text{avec } x \in X$$

Soit σ_0 la solution unique de (P_0) , c'est-à-dire la fonction spline de lissage sans contraintes.

* Si $\sigma_0 \in C$ alors c'est la solution du problème (4.7).

Itération 1

* Si $\sigma_0 \in C$, alors trouver t^0 tel que

$$\begin{cases} \gamma_{t^0}(\sigma_0) < 0 \\ \gamma_{t^0}(\sigma_0) \leq g(\sigma_0) + \varepsilon \end{cases}$$

* Résoudre :

$$(P_1) \quad \text{minimiser } f(x) \quad \text{avec } x \in C_{t^0}$$

Soit σ_1 la solution unique de (P_1) .

* Si $\sigma_1 \in C$ alors c'est la solution du problème (4.7).

Remarque

Au lieu de résoudre (P_1) on peut obtenir la solution si on ne considère que la contrainte égalité :

$$\text{minimiser } f(x) \quad \text{avec } x \in \Gamma(C_t^*)$$

où $\Gamma(C_t^*) = \{x \in X / \gamma_t^*(x) = 0\}$ est la frontière de C_t^* .

Itération 2

* Si $\sigma_1 \notin C$ alors trouver $t^1 \in I$ tel que

$$\begin{cases} \gamma_{t^1}(\sigma_1) < 0 \\ \gamma_{t^1}(\sigma_1) \leq g(\sigma_1) + \varepsilon^1. \end{cases}$$

* Résoudre:

$$(P_2) \quad \text{minimiser } f(x) \quad \text{avec } x \in C_t^* \cap C_{t^1}$$

Soit σ_2 la solution unique de (P_2) .

* Si $\sigma_2 \in C$ alors c'est la solution du problème (4.7).

Remarques

1) En deux étapes (P_2) peut être résolu avec contraintes du type égalité. On considère d'abord le problème :

$$(i) \quad \text{minimiser } f(x) \quad \text{avec } x \in \Gamma(C_{t^1})$$

Si la solution appartient à

C_t^* , c'est σ_2 . Sinon, on obtient σ_2 en résolvant le problème :

(ii) minimiser $f(x)$ avec $x \in \Gamma(C_t) \cap \Gamma(C_t^*)$

2) Les conditions de Kuhn-Tucker qui caractérisent σ_2 sont :
qu'il existe de coefficients $v_j^2 \geq 0$ ($j=0,1$) tels que

$$f'(\sigma_2) = \sum_{j=0}^1 v_j^2 \gamma_{tj} \quad , \quad \sum_{j=0}^1 v_j^2 \gamma_{tj}(\sigma_2) = 0$$

Si l'on considère le demi-espace

$$D^2 = \{ x \in X / \sum_{j=0}^1 v_j^2 \gamma_{tj}(x) \geq 0 \} \quad \text{on a } C \subset D^2 \text{ et } \sigma_2 \in \Gamma(D^2).$$

Si l'on applique de nouveau le théorème de Kuhn-Tucker on trouve que $\sigma_2 \in \Gamma(D^2)$ réalise le minimum de f sur D^2 .

Passage de l'itération k à l'itération $k+1$:

* A l'itération k on a :

k points ajoutés $t^j \in I$ ($j=0, \dots, k-1$)

k coefficients $v_j^k \geq 0$ ($j=0, \dots, k-1$)

le demi-espace

$$D^k = \{ x \in X / \sum_{j=0}^{k-1} v_j^k \gamma_{t^j}(x) \geq 0 \}$$

et $\sigma_k \in \Gamma(D^k)$ qui réalise le minimum de f sur D^k .

Les conditions caractérisant σ_k sont fournies par le théorème de Kuhn-Tucker :

$$f'(\sigma_k) = \sum_{j=0}^{k-1} v_j^k \gamma_{tj} \quad , \quad \sum_{j=0}^{k-1} v_j^k \gamma_{tj}(\sigma_k) = 0$$

Si $\sigma_k \in C$ c'est la solution du problème (4.7)

* Si $\sigma_k \notin C$ alors on détermine $t^k \in I$ tel que

$$\begin{cases} \gamma_{t^k}(\sigma_k) < 0 \\ \gamma_{t^k}(\sigma_k) \leq g(\sigma_k) + \epsilon^k \end{cases}$$

et on résout :

$$(P_{k+1}) \text{ minimiser } f(x) \text{ avec } x \in C_{t^k} \cap D^k$$

Si la solution unique σ_{k+1} de (P_{k+1}) appartient à C , elle réalise le minimum de f sur C .

Remarques

1) Le problème (P_{k+1}) peut être réduit à un ou deux problèmes de minimisation avec contraintes de type égalité :

On résout d'abord :

$$(i) \text{ minimiser } f(x) \text{ avec } x \in \Gamma(C_{t^k})$$

Si la solution est un élément de D^k , c'est σ_{k+1} . Sinon, on prend pour σ_{k+1} la solution de :

$$(ii) \text{ minimiser } f(x) \text{ avec } x \in \Gamma(C_{t^k}) \cap \Gamma(D^k).$$

2) Utilisons les conditions de Kuhn-Tucker pour la caractérisation de σ_{k+1} :

il existe des nombres $\eta^{k+1} \geq 0$ et $\theta^{k+1} \geq 0$ tels que

$$\left\{ \begin{array}{l} f'(\sigma_{k+1}) = \eta^{k+1} \gamma_{t^k} + \theta^{k+1} \left(\sum_{j=0}^{k-1} v_j^k \gamma_{t^k} \right) \\ \eta^{k+1} \gamma_{t^k}(\sigma_{k+1}) = 0 \\ \theta^{k+1} \sum_{j=0}^{k-1} v_j^k \gamma_{t^k}(\sigma_{k+1}) = 0 \end{array} \right.$$

Si l'on pose :

$$\left\{ \begin{array}{l} v_j^{k+1} = \theta^{k+1} v_j^k \quad (j=0, \dots, k-1) \\ v_k^{k+1} = \eta^{k+1} \end{array} \right.$$

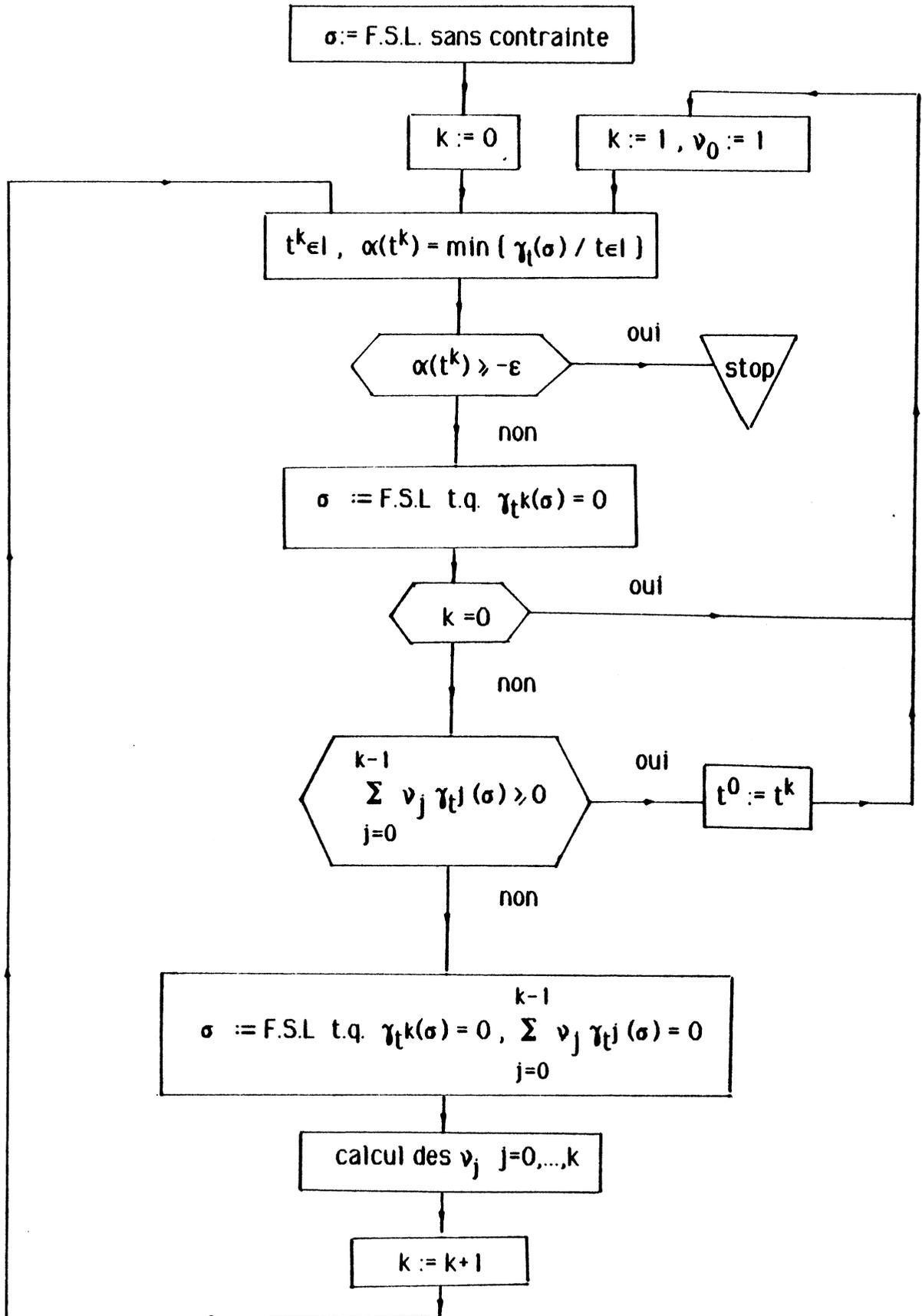
on peut réécrire les relations qui se trouvent ci-dessus :

$$\left\{ \begin{array}{l} f'(\sigma_{k+1}) = \sum_{j=0}^k v_j^{k+1} \gamma_{t^j} \\ \sum_{j=0}^k v_j^{k+1} \gamma_{t^j}(\sigma_{k+1}) = 0 \end{array} \right.$$

En vertu du théorème de Kuhn-Tucker, $\sigma_{k+1} \in \Gamma(D^{k+1})$ réalise le minimum de f sur D^{k+1} , où D^{k+1} est le demi-espace

$$D^{k+1} = \left\{ x \in X / \sum_{j=0}^k v_j^{k+1} \gamma_{t^j}(x) \geq 0 \right\} \quad \square$$

Dans l'organigramme 1 on a résumé les principales étapes de l'algorithme.



Organigramme 1.

(F.S.L. : fonction spline de lissage)

Commentaires

1) A chaque itération le nombre de contraintes à considérer est 1

ou 2. Ainsi à l'itération $k+1$, on a $\gamma_{t^k}(\sigma)$ et $\sum_{j=0}^{k-1} v_j^k \gamma_{t^j}(\sigma) = 0$

la deuxième contrainte tient compte des points $t^j (j=0, 1, \dots, k-1)$ introduits précédemment.

2) Si le coefficient θ^{k+1} est égal à zéro, c'est-à-dire, en fait si σ_{k+1} est déjà obtenu en résolvant le problème (i) ci-dessus, alors tous les coefficients v_k^{k+1} sont nuls, sauf v_k^{k+1} et l'on a :

$$D^{k+1} = \{ x \in X / \gamma_{t^k}(x) \geq 0 \}.$$

Ainsi il n'y a plus qu'une seule contrainte à considérer (toutes les contraintes $\gamma_{t^i}(x) \geq 0$, $i=0, \dots, k-1$, introduites précédemment peuvent être "oubliées"). On se retrouve exactement dans la situation de la première itération, mais avec une solution améliorée bien sûr. On dit que l'on a "éclatement des contraintes" (ceci apparaît dans l'organigramme 1 lorsque la réponse au test

$$\sum_{j=0}^{k-1} v_j \gamma_{t^j}(\sigma) \geq 0 \text{ est "oui"}).$$

Ce phénomène fait que le nombre de points que l'on a à considérer, à un moment donné au fil des itérations, reste en général peu élevé.

4.3. Convergence de l'algorithme

Voici la propriété de convergence de l'algorithme que nous venons de décrire.

Proposition 4.2.

La suite (σ_k) construite par l'algorithme 4.1 vérifie :

$$\lim_{k \rightarrow \infty} f(\sigma_k) = f(\sigma) , \quad \lim_{k \rightarrow \infty} \inf g(\sigma_k) \geq 0$$

et σ_k converge fortement vers σ dans X ;

σ étant la solution unique de (4.7).

Preuve

La preuve de cette proposition se trouve dans [57]. Ses grandes lignes sont les suivantes :

- (i) Par construction on a $D^k \supset C_{t^k} \cap D^k \supset C$ et les inégalités $f(\sigma) \geq f(\sigma_{k+1}) \geq f(\sigma_k)$.
- (ii) La convexité forte de f entraîne que $\lim_{k \rightarrow \infty} \|\sigma_{k+1} - \sigma_k\| \geq 0$
- (iii) Le choix de t^k détermine que $\lim_{k \rightarrow \infty} \inf g(\sigma_k) = 0$
- (iv) La compacité faible permet de construire une sous-suite de (σ_k) qui converge vers σ .
- (v) La convexité forte de f implique que (σ_k) converge fortement vers σ .

4.4. Application au lissage de données avec contrainte de positivité

Dans ce paragraphe on va considérer

$$r_t(x) = x(t) \quad x \in X, \quad \forall t \in I \quad \text{et}$$

$$g(x) = \min_{t \in I} \{x(t)\} \quad x \in X$$

Les parties principales de l'algorithme sont :

- (i) Le calcul de la fonction spline de lissage sans contraintes
- (ii) Le calcul de la fonction spline de lissage avec la contrainte $\sigma(t^k) = 0$ (problème auxiliaire Q_1)
- (iii) La résolution du problème auxiliaire Q_2 :
trouver la fonction spline de lissage qui satisfasse les contraintes

$$\sigma(t^k) = 0 \quad \text{et} \quad \sum_{j=0}^{k-1} v_j \sigma(t^j) = 0$$

On doit aussi calculer les coefficients de Kuhn-Tucker.

- (iv) La recherche du minimum de la fonction spline.

4.4.1. Calcul de la spline de lissage sans contraintes

Nous rappelons que la solution unique σ du problème

$$(4.8) \quad \text{minimiser } E(x) + \rho J(x) \quad \text{avec } x \in X$$

est caractérisée par les conditions suivantes :

- (i) σ est un polynôme de degré 3 dans chaque intervalle $[t_i, t_{i+1}]$ ($i=1,2,\dots,n-1$)
- (ii) σ est un polynôme de degré 1 dans $[a, t_1]$ et $[t_n, b]$

(iii) $\sigma \in C^2[a,b]$

(iv) $\Delta \sigma'''(t_i) = \rho(z_i - \sigma(t_i)) \quad i=1,2,\dots,n$

où $\Delta \sigma^{(j)}(t_i) = \sigma^{(j)}(t_i^+) - \sigma^{(j)}(t_i^-)$. \square

Si l'on note $y_i = \sigma(t_i)$, $\lambda_i = \sigma'(t_i)$ pour $i=1,2,\dots,n$, la fonction σ peut s'écrire pour $t_i \leq t \leq t_{i+1}$:

$$(4.9) \quad s(t) = y_i + \lambda_i(t-t_i) + \frac{m_i - \lambda_i}{h_i}(t-t_i)^2 + \frac{\lambda_{i+1} - 2m_i + \lambda_i}{h_i^2}(t-t_i)^2(t-t_{i+1})$$

où $h_i = t_{i+1} - t_i$, $m_i = (y_{i+1} - y_i) / h_i$

Pour évaluer les inconnues y_i, λ_i ($i=1,2,\dots,n$) on utilise les conditions (ii), (iii) et (iv), ce qui nous donne un système tridiagonal par blocs 2×2 :

$$(4.10) \quad \begin{cases} B_1 \Lambda_1 + C_1 \Lambda_2 = D_1 \\ A_i \Lambda_{i-1} + B_i \Lambda_i + C_i \Lambda_{i+1} = D_i \quad i=2,3,\dots,n-1 \\ A_n \Lambda_{n-1} + B_n \Lambda_n = D_n \end{cases}$$

où

$$B_1 = \begin{bmatrix} 2+w_1 & h_1 \\ 3 & 2h_1 \end{bmatrix}, \quad B_n = \begin{bmatrix} 2+w_n & -h_n \\ -3 & 2h_n \end{bmatrix}$$

$$B_i = \begin{bmatrix} 2(1+\gamma_i^3) + w & (1-\gamma_i^2)h_i \\ 3(1-\gamma_i^2) & 2(1+\gamma_i)h_i \end{bmatrix} \quad i=2,3,\dots,n-1$$

$$A_i = \begin{bmatrix} -2\gamma_i^3 & -\gamma_i^2 h_i \\ 3\gamma_i^2 & \gamma_i h_i \end{bmatrix} \quad i=2,3,\dots,n$$

$$C_i = \begin{bmatrix} -2 & 3h_i \\ 3 & -h_i \end{bmatrix} \quad i=1,2,\dots,n-1$$

$$D_i^T = [w_i z_i \ 0] \quad , \quad \Lambda_i^T = [\gamma_i \ \lambda_i] \quad i=1,2,\dots,n$$

avec $h_n = h_{n-1}$

$$\gamma_i = h_i / h_{i-1} \quad i=2,3,\dots,n$$

$$w_i = \rho h_i^3 / 6 \quad i=1,2,\dots,n$$

On résout (4.10) par l'élimination Gauss par bloc.

4.4.2. Résolution des problèmes auxiliaires (Q₁) et (Q₂)

A l'itération k il faut trouver la solution de

$$(Q_1) \begin{cases} \text{minimiser } E(x) + \rho J(x) & \text{avec } x \in X \\ \text{avec la contrainte } x(t^k) = 0 & \text{pour } t^k \text{ donné.} \end{cases}$$

On peut déduire aisément que la solution unique $\bar{\sigma}$ de (Q₁) est caractérisée par les conditions (i)-(iv) du paragraphe précédent (en remplaçant σ par $\bar{\sigma}$) et , en plus, par :

(v) Il y a une discontinuité de la dérivée troisième en t^k et $\bar{\sigma}(t^k) = 0$.
($\bar{\sigma}$, $\bar{\sigma}'$ et $\bar{\sigma}''$ étant continues en t^k).

Il faut aussi résoudre

$$(Q_2) \left\{ \begin{array}{l} \text{minimiser } E(x) + \rho J(x) \quad \text{avec } x \in X \\ \text{et les contraintes } x(t^k) = 0, \quad \sum_{j=0}^{k-1} \nu_j x(t^j) = 0 \end{array} \right.$$

où les t^j et les ν^j sont donnés

Par commodité on va supposer que les t^j sont distincts des t_i ($j=0,1,\dots,k$, $i=1,2,\dots,n$). Alors la solution unique $\hat{\sigma}$ de (Q_2) est caractérisée par les conditions (i)-(v) (en remplaçant $\tilde{\sigma}$ par $\hat{\sigma}$) et, en plus, par :

(vi) En t^j ($j=0,1,\dots,k-1$) on a une discontinuité de la dérivée troisième $\Delta \hat{\sigma}'''(t^j)$ qui doit vérifier $\Delta \hat{\sigma}'''(t^j) = \theta \nu_j$, où θ est un coefficient choisi de telle sorte que l'on ait

$$\sum_{j=0}^{k-1} \nu_j \hat{\sigma}(t^j) = 0$$

Le coefficient θ n'est pas connu, mais à cause de la linéarité on peut l'obtenir en faisant deux essais :

Premier essai : $\theta=0$

On obtient alors $\tilde{\sigma}$ la solution de (Q_1) déjà calculée.

Posons

$$\nu_0 = \sum_{j=0}^{k-1} \nu_j \tilde{\sigma}(t^j)$$

Supposons $\nu_0 \neq 0$ (sinon $\hat{\sigma} = \tilde{\sigma}$)

Deuxième essai : $\theta=1$

Soit $\tilde{\tilde{\sigma}}$ la solution correspondante.

Posons

$$\nu_1 = \sum_{j=0}^{k-1} \nu_j \tilde{\tilde{\sigma}}(t^j)$$

Si σ_θ désigne la solution relative à un θ , on a

$$(4.11) \quad \sigma_\theta = \tilde{\sigma} + \theta(\tilde{\sigma} - \tilde{\sigma})$$

et pour le v_θ correspondant

$$(4.12) \quad v_\theta = v_0 + \theta(v_1 - v_0)$$

Comme on veut avoir $V_\theta = 0$, il faut choisir

$$(4.13) \quad \theta = v_0 / (v_0 - v_1) \quad \square$$

L'algorithme de résolution du problème (4.1) avec la contrainte de positivité est explicité par l'organigramme 2.

Remarque

Si $v_0 / (v_0 - v_1)$ est grand (éloigné de 1 et 0) on risque d'assigner une valeur erronée à θ , ce qui entraînerait des erreurs dans les coefficients v_j^k de Kuhn-Tucker. Alors il faut refaire le calcul pour deux valeurs plus proches du résultat.

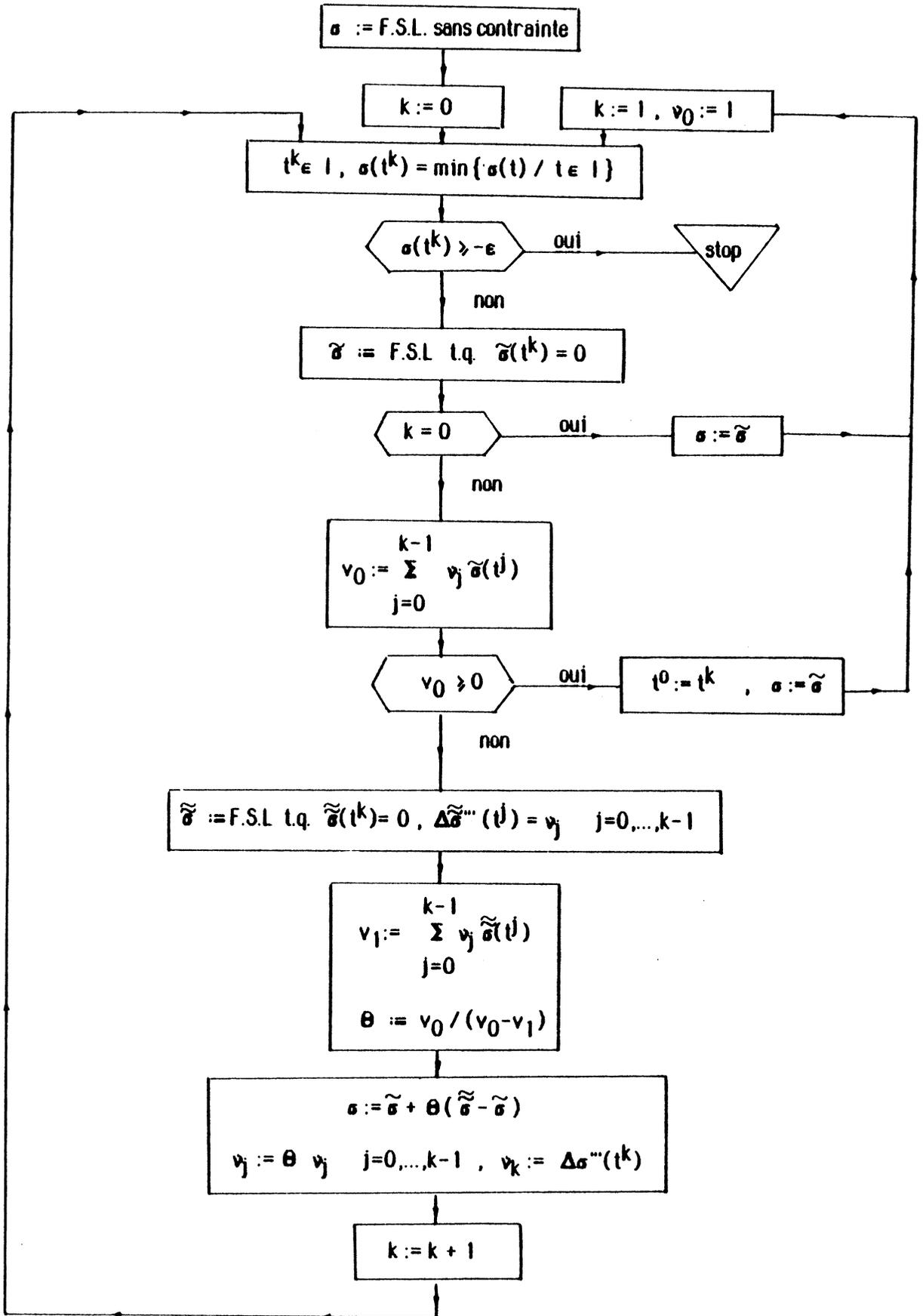
Pour évaluer les valeurs de la fonction spline et de sa dérivée première aux noeuds dans les problèmes (Q_1) et (Q_2) on est amené aussi à résoudre des systèmes tridiagonaux par blocs 2×2 . Ces systèmes sont obtenus aisément en modifiant (4.10) par l'introduction des conditions (v) et (vi).

4.5. Application au lissage des données avec contrainte de monotonie

Nous supposons ici que

$$\gamma_t(x) = x'(t) \quad \text{pour } x \in X, \quad \forall t \in I$$

et
$$g(x) = \min_{t \in I} \{x'(t)\} \quad \text{pour } x \in X$$



Organigramme 2.

Les parties principales de l'algorithme sont :

- (i) Le calcul de la fonction spline de lissage sans contraintes.
- (ii) La résolution du problème auxiliaire

$$(Q_1) \left\{ \begin{array}{l} \text{minimiser } E(x) + \rho J(x) \quad \text{avec } x \in X \\ \text{et la contrainte } x'(t^k) = 0 \quad \text{où } t^k \text{ est donné} \end{array} \right.$$

- (iii) La résolution du problème auxiliaire

$$(Q_2) \left\{ \begin{array}{l} \text{minimiser } E(x) + \rho J(x) \quad \text{avec } x \in X \\ \text{et les contraintes } x'(t^k) = 0, \quad \sum_{i=0}^{k-1} \nu_i x'(t^i) = 0 \end{array} \right.$$

où les t^k et les ν_i sont donnés.

- (iv) La recherche du minimum de la dérivée première de la fonctionnelle spline.

Nous procédons de la même manière que pour les splines de lissage avec contrainte de positivité. Dans ce qui suit nous allons étudier brièvement les problèmes (Q_1) et (Q_2) .

La solution unique $\tilde{\sigma}$ du problème auxiliaire (Q_1) est caractérisée par les cinq conditions suivantes :

- (i) $\tilde{\sigma}$ est un polynôme de degré 3 dans chaque intervalle $[t_i, t_{i+1}]$ ($i=1, 2, \dots, n-1$)
- (ii) $\tilde{\sigma}$ est un polynôme de degré 1 dans $[a, t_1]$ et $[t_n, b]$
- (iii) $\tilde{\sigma} \in C^1[a, b]$
- (iv) $\tilde{\sigma}''$ est continue en t_i et $\Delta \tilde{\sigma}'''(t_i) = \rho(z_i - \tilde{\sigma}(t_i)) \quad i=1, 2, \dots, n$

- (v) $\tilde{\sigma}'(t^k)=0$ et $\tilde{\sigma}''$ est discontinue en t^k ($\tilde{\sigma}$ et $\tilde{\sigma}'''$ étant continues en t^k).

La solution unique $\hat{\sigma}$ du problème auxiliaire (Q_2) est caractérisée par les conditions (i) - (v), en remplaçant $\tilde{\sigma}$ par $\hat{\sigma}$, et, en plus, par

- (vi) $\hat{\sigma}'''$ est continue en t^j ($j=0,1,\dots,k-1$). En t^j il y a une discontinuité de la dérivée seconde $\Delta\hat{\sigma}''(t^j)$ qui doit vérifier $\Delta\hat{\sigma}''(t^j)=\theta v_j$, où θ est un coefficient choisi de telle sorte que

$$\text{l'on ait } \sum_{j=0}^{k-1} v_j \hat{\sigma}'(t^j) = 0.$$

Le coefficient θ se calcule d'une manière analogue à celle du paragraphe précédent.

IV. 5. ESSAIS NUMERIQUE

L'algorithme 4.1. a été testé pour le lissage de données avec la contrainte de positivité et de monotonie.

5.1. Préservation de la positivité

- Considérons d'abord l'ensemble de données qui apparaissent dans le tableau 1.

i	t_j	z_j	i	t_j	z_j
1	59.5	0.039	8	87.5	0.731
2	63.5	0.046	9	91.5	0.993
3	67.5	0.046	10	95.5	0.067
4	71.5	0.058	11	99.5	0.001
5	75.5	0.081	12	103.5	-0.002
6	79.5	0.089	13	107.5	0.002
7	83.5	0.158			

Tableau 1. Données 1.

Nous avons fait des essais pour différentes valeurs de ρ , qui correspondent aux puissances de 10 comprises entre 10^{-1} et 10^2 . Pour cet exemple, si ρ est plus petit que 10^{-1} le lissage est trop

marqué et, par contre, si ρ est plus grand que 10^2 on s'approche de l'interpolation. La tolérance de calcul a été fixée $\epsilon=10^{-3}$ et $\epsilon=10^{-5}$. Quelques résultats sont résumés dans le tableau 2.

ρ	ϵ	points ajoutés	y min.	temps(sec.)
10^{-1}	10^{-3}	1	-3.7782 E-4	1.18
	10^{-5}	8	-6.4747 E-6	16.38
1	10^{-3}	3	-9.8695 E-4	5.40
	10^{-5}	6	-5.9958 E-6	20.44
10	10^{-3}	2	-6.0260 E-4	3.34
	10^{-5}	8	-1.8137 E-6	25.80
10^2	10^{-3}	3	-7.5351 E-4	16.51
	10^{-5}	7	-9.3620 E-6	37.28

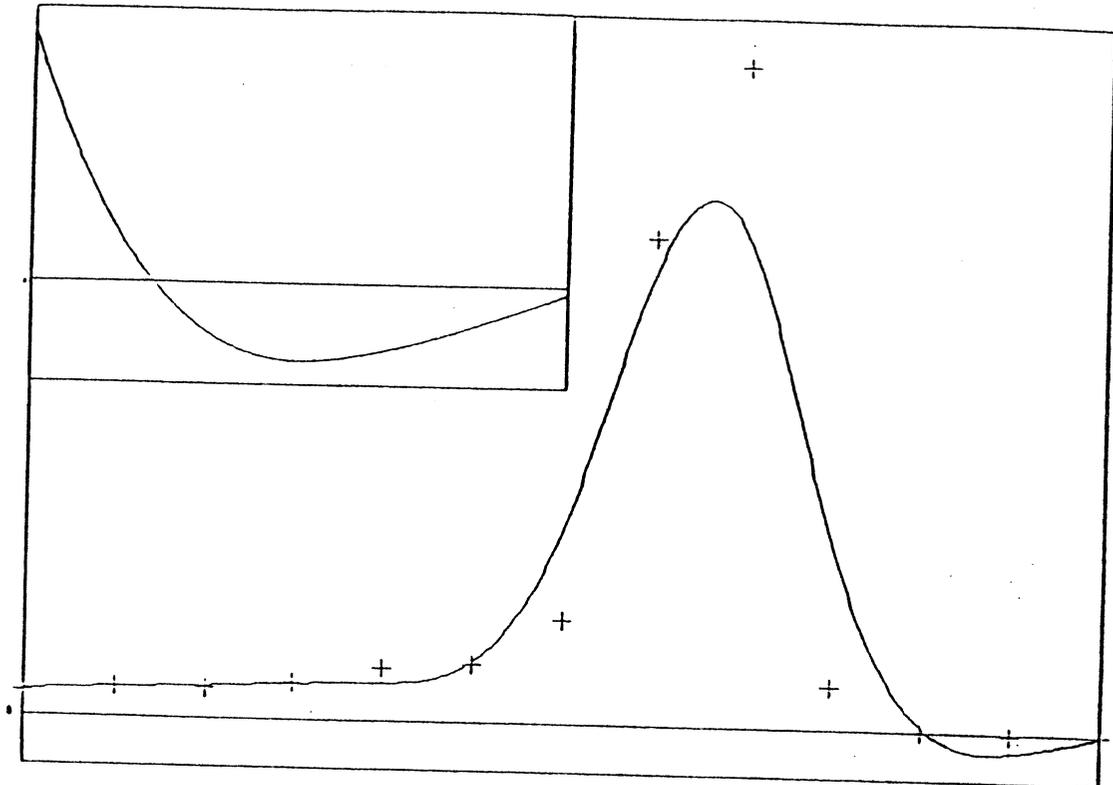
Tableau 2.

La quatrième colonne du tableau 2 donne la valeur des minimum de la spline pour lequel le seuil ϵ a été atteint.

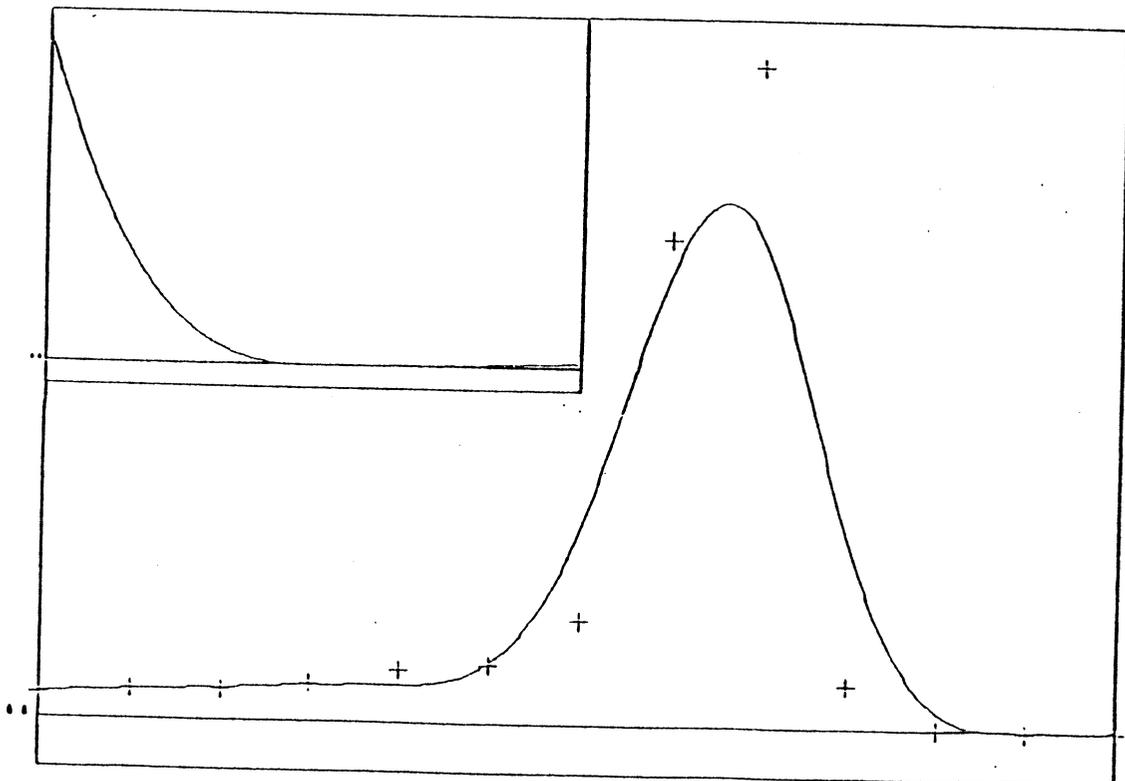
On remarque que si l'on exige une meilleure précision, le nombre de points ajoutés est plus grand et de même le temps de calcul. Cependant, le nombre de points insérés ne correspond pas exactement au nombre d'itérations, puisqu'il peut se produire des éclatements de contraintes pendant l'exécution du programme.

Maintenant voyons ce qui se passe du point de vue graphique. Nous avons sélectionné les graphes des fonctions qui correspondent à $\rho=10^{-1}$, 1 et 100 (Figs. 1,2,3). En haut apparaissent les splines sans contrainte et en bas les splines avec contrainte après que la précision $\epsilon=10^{-5}$ ait été obtenue. Dans les fenêtres on a placé des agrandissements des courbes sur l'intervalle [95.5,107.5], là où les différences sont les plus remarquables.

On observe dans tous les cas que les oscillations autour de la droite zéro ont disparu "à vue". Le reste de la courbe reste pratiquement inchangé.



(a)



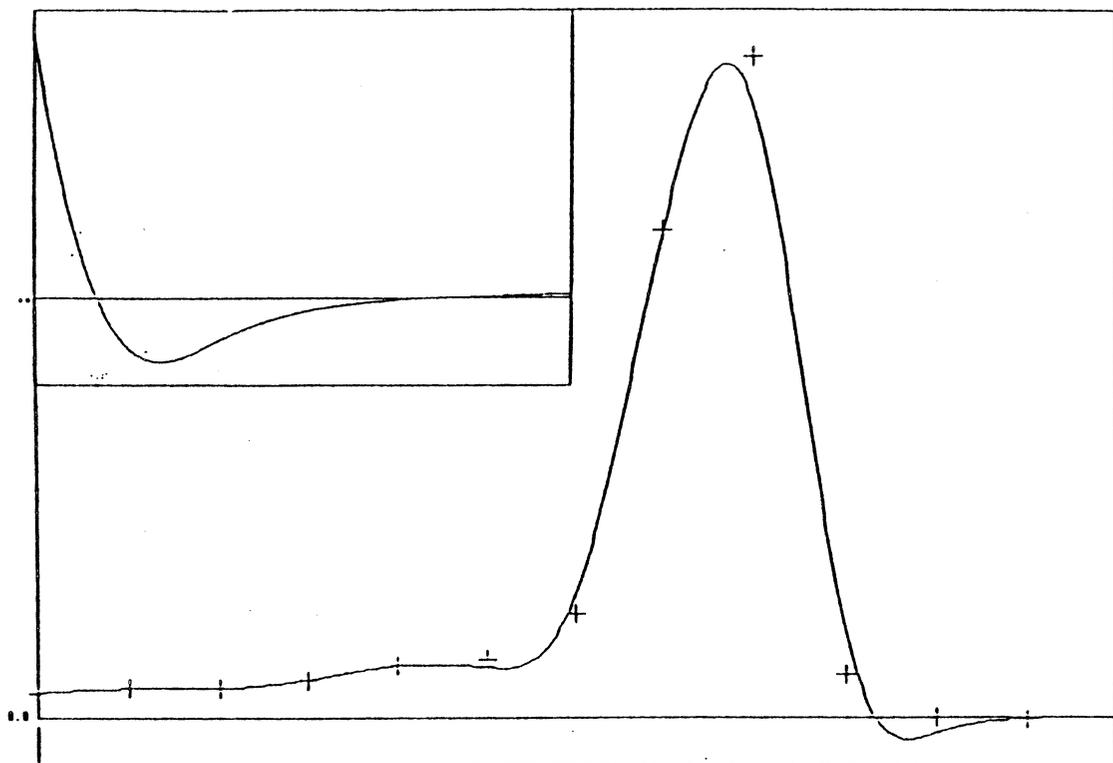
(b)

Fig.1. F.S. de lissage pour les données 1. ($\rho=1/10$)

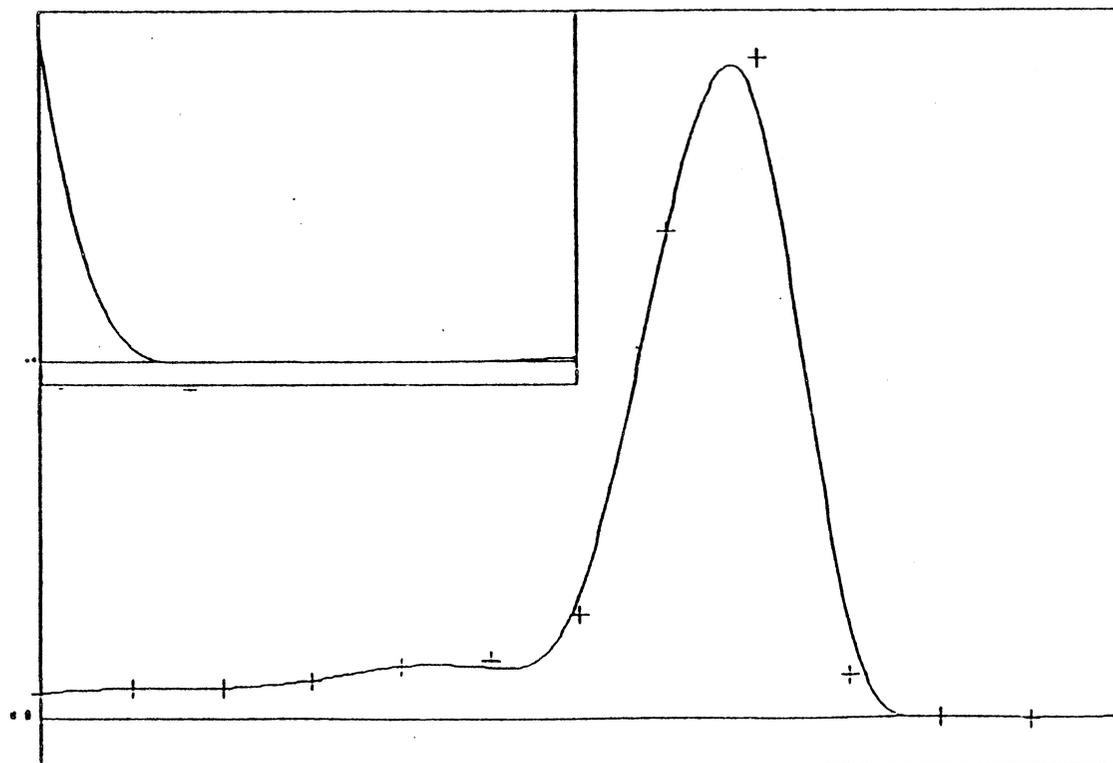
(a) sans contrainte

(b) avec contrainte de positivité

Dans les fenêtres, les fonctions sur l'intervalle [95.5, 107.5]



(a)



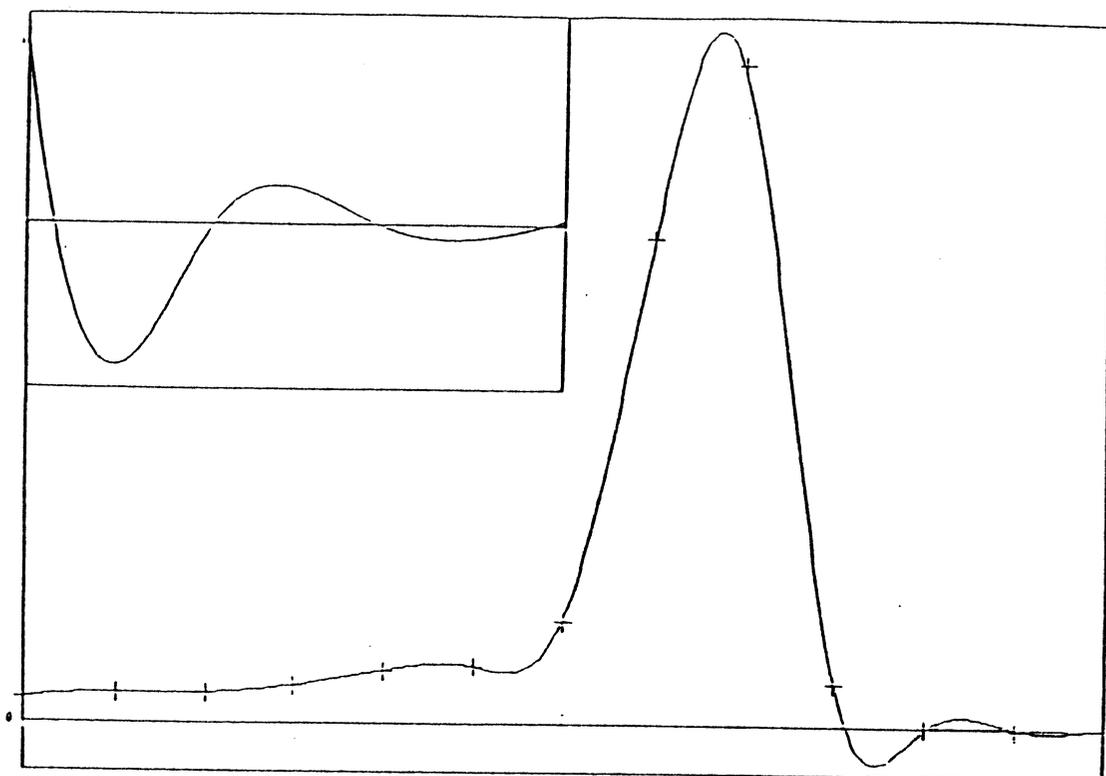
(b)

Fig.2. F.S. de lissage pour les données 1. ($\rho=1$)

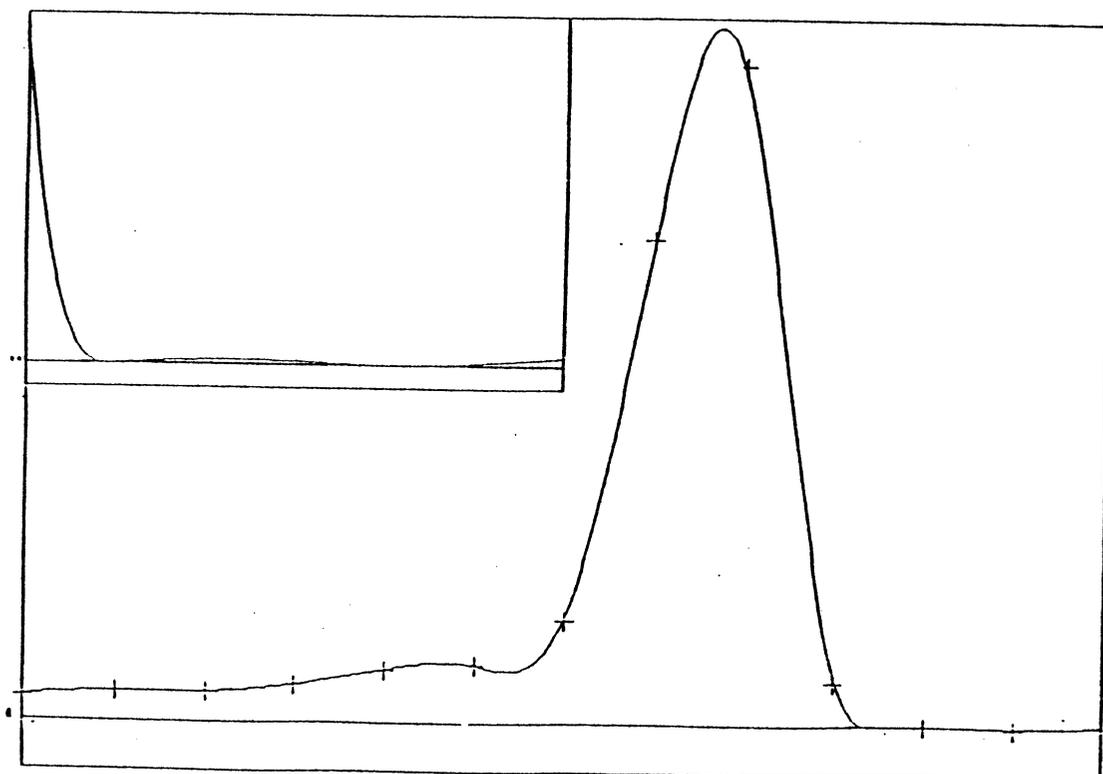
(a) sans contrainte

(b) avec contrainte de positivité

Dans les fenêtres, les fonctions sur l'intervalle [95.5, 107.5]



(a)



(b)

Fig.3. F.S. de lissage pour les données 1. ($\rho=100$)

(a) sans contrainte

(b) avec contrainte de positivité

Dans les fenêtres, les fonctions sur l'intervalle [95.5, 107.5]

- Un deuxième exemple que nous avons traité est la fonction

$$f_1(t) = t^2 e^t \quad \text{sur l'intervalle } [-5, 1]$$

Nous avons considéré n abscisses équidistantes ($n=10,20,30$) et bruitée les valeurs de f_1 , l'écart type étant de 0.5.

En fait, le choix des noeuds équidistants n'a pas d'importance pratique : les méthodes doivent être programmées pour des pas non-équidistants car on insère des points.

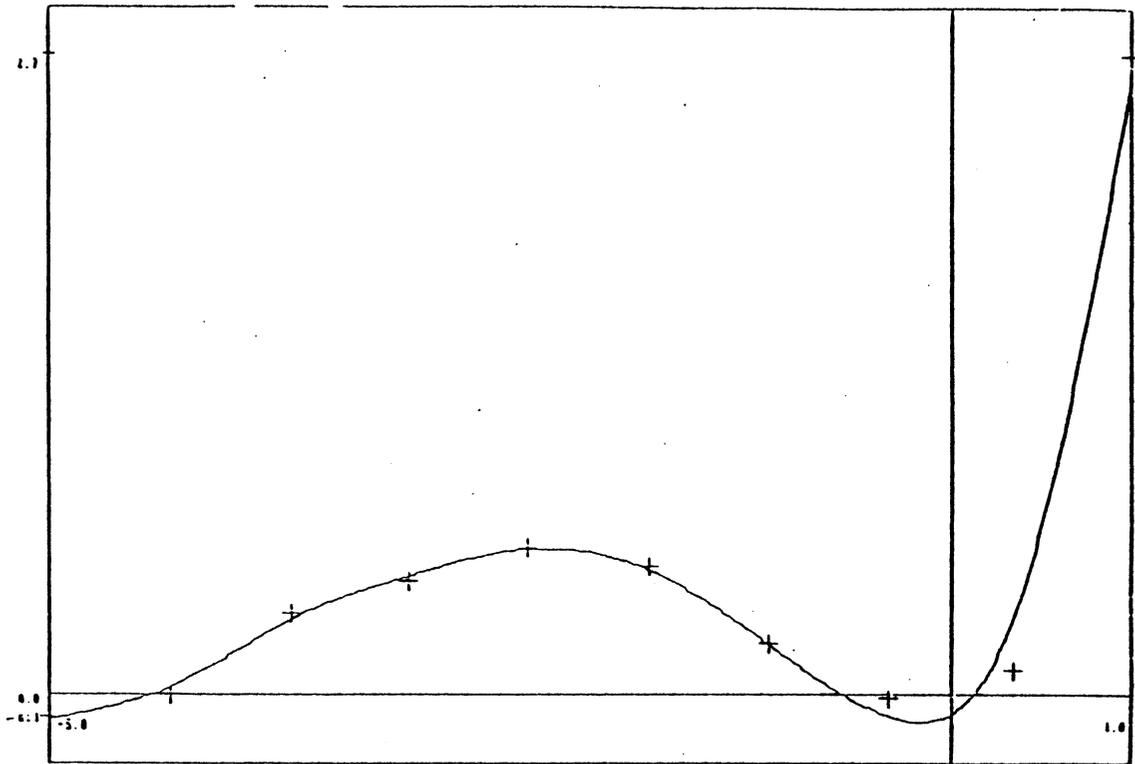
Les valeurs du paramètre de lissage sont données par la méthode de Reinsch. On a arrêté les calculs à une précision de 10^{-4} .

n	ρ	points ajoutés	y min.	temps(sec.)
10	78.5	5	-2.9032 E-5	17.50
20	153.2	6	-8.8703 E-5	31.98
30	201.6	2	-3.2646 E-5	5.10

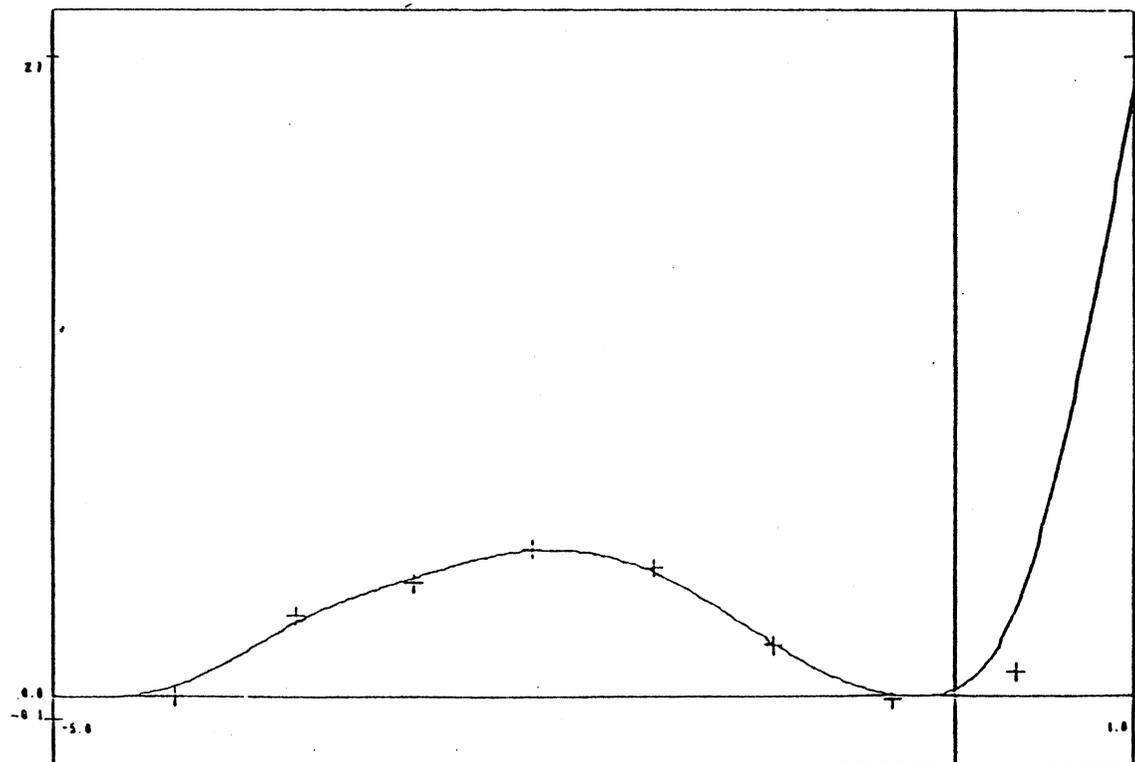
Tableau 3.

En regardant le tableau 3 on remarque que la valeur de ρ pour $n=30$ correspond à un "bon choix" dans le sens que le nombre de points à ajouter n'est que de 2, ce qui a été obtenu rapidement.

On s'aperçoit sur les dessins (Figs. 4,5,6) que les variations plus importantes se produisent à la borne gauche de l'intervalle et autour du point 0. Les splines avec contraintes donnent encore des résultats satisfaisants, en particulier pour $n=30$ (Fig. 6(b)).



(a)

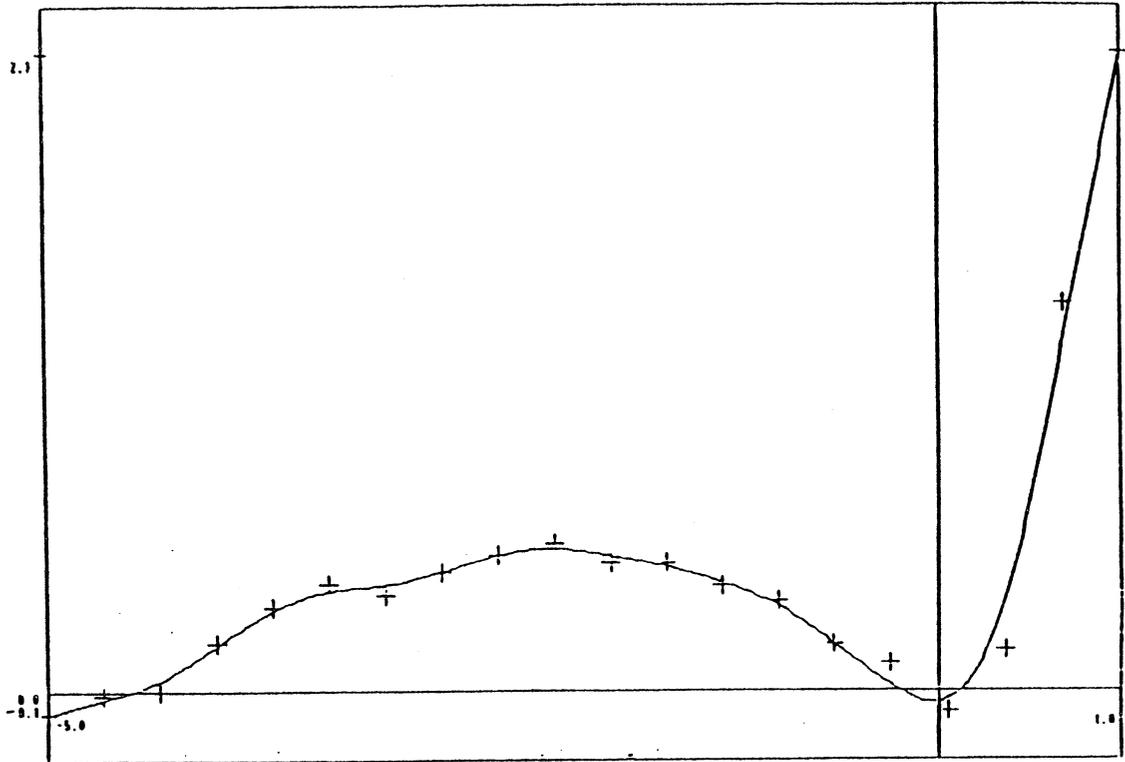


(b)

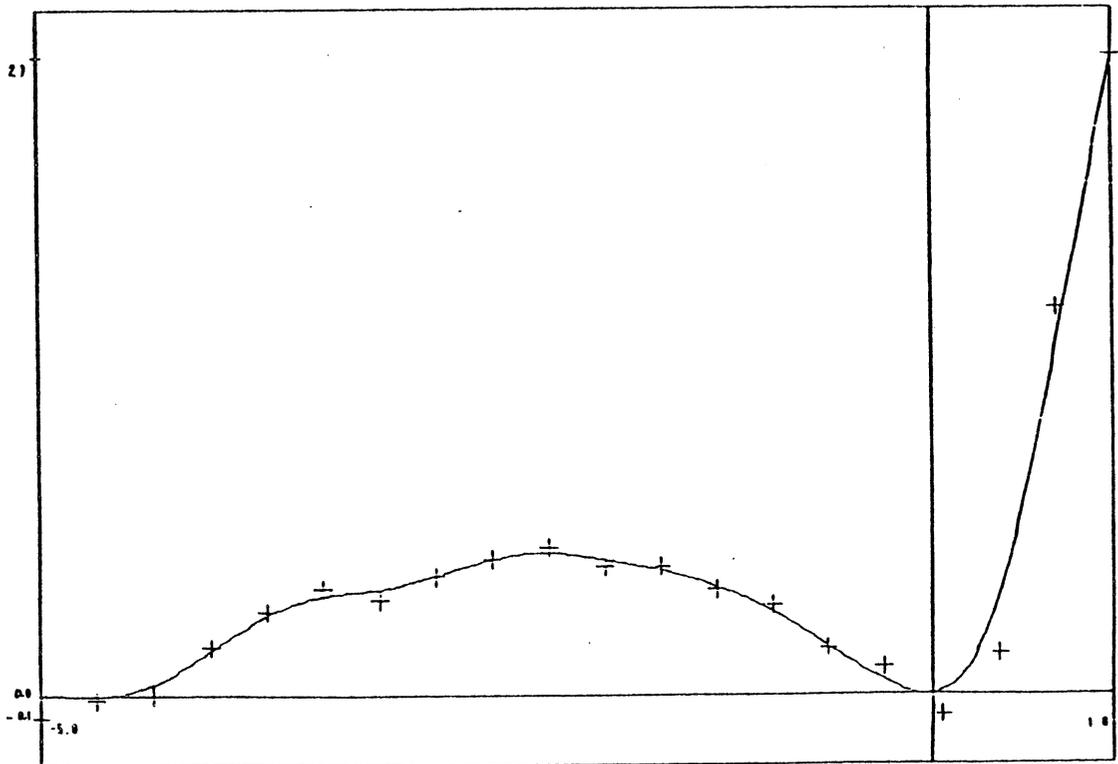
Fig. 4. F.S. de lissage pour $f_1(t)$, $t \in [-5, 1]$ ($n=10$)

(a) sans contrainte

(b) avec contrainte de positivité



(a)

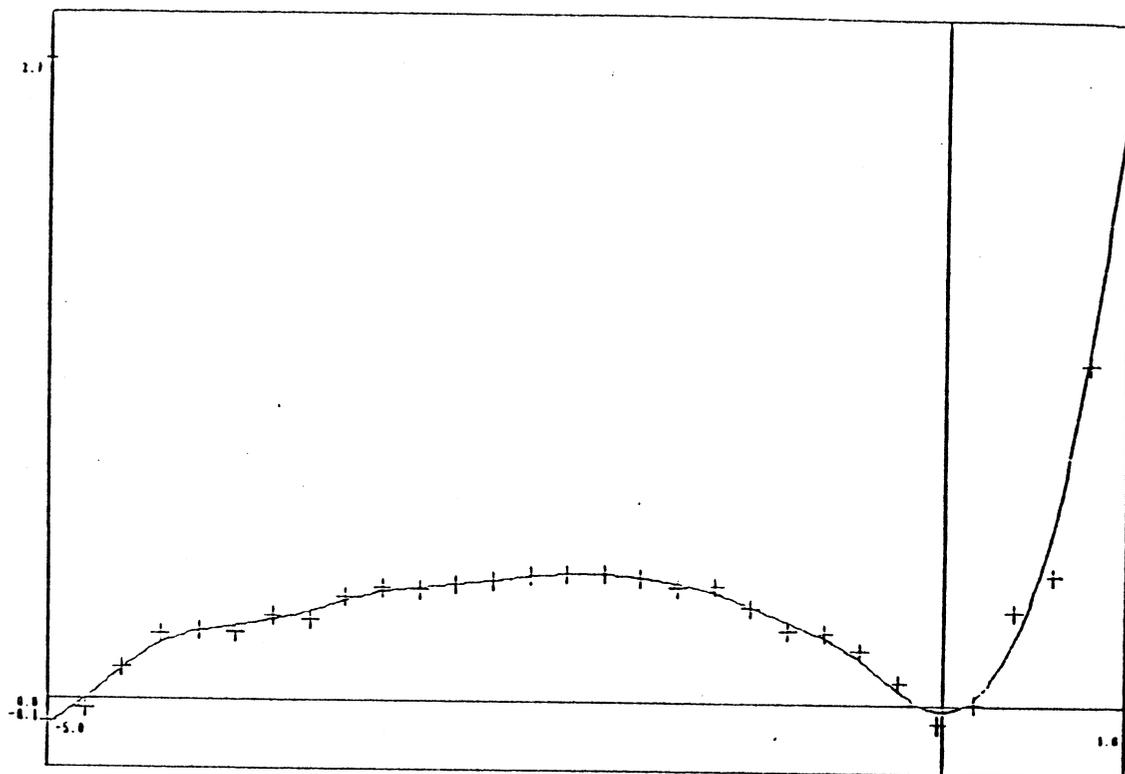


(b)

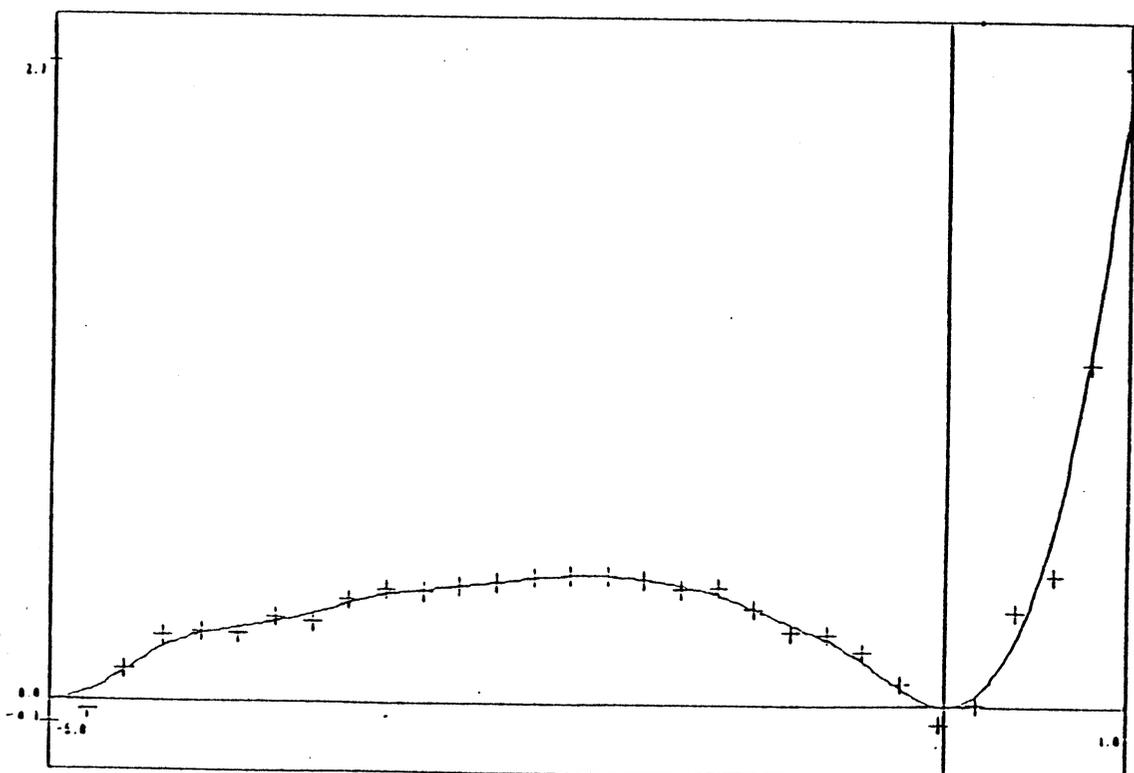
Fig. 5. F.S. de lissage pour $f_1(t)$, $t \in [-5, 1]$ ($n=20$)

(a) sans contrainte

(b) avec contrainte de positivité



(a)



(b)

Fig. 6. F.S. de lissage pour $f_1(t)$, $t \in [-5, 1]$ (n=30)

(a) sans contrainte

(b) avec contrainte de positivité

5.2. Préservation de la monotonie

- Reprenons l'ensemble des données du papier de Dierckx [29]

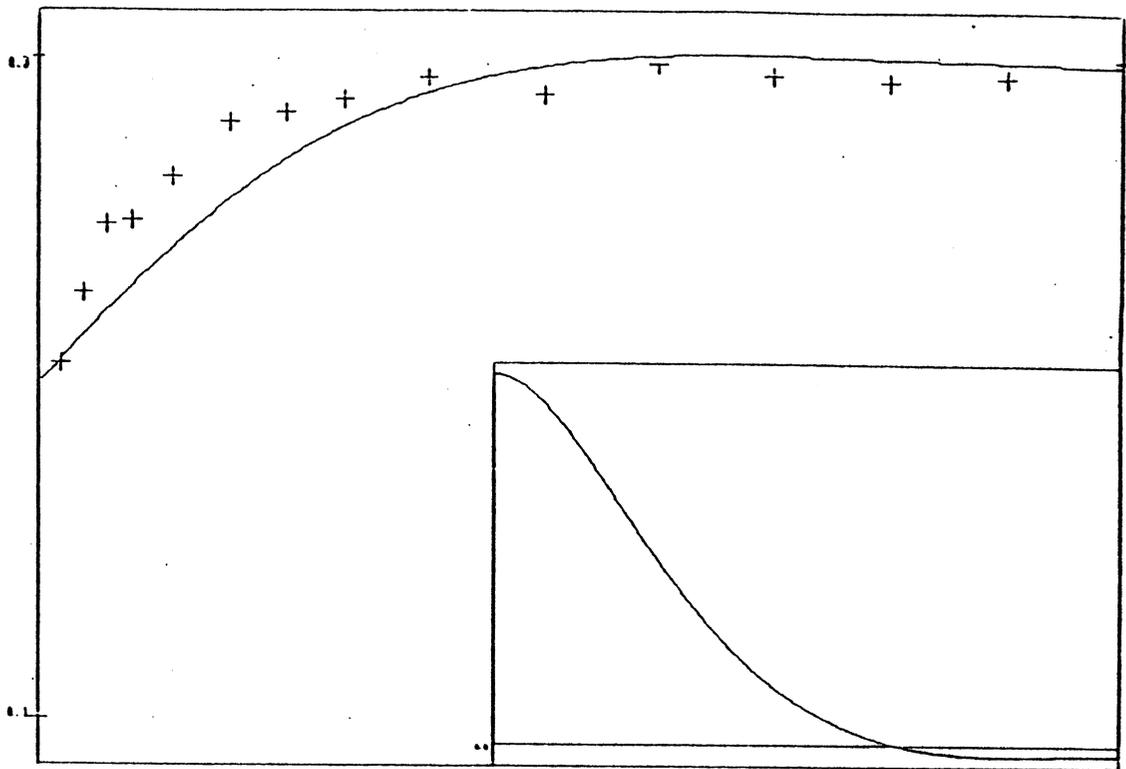
i	t_i	z_i	i	t_i	z_i
1	0.100	0.124	9	2.750	0.315
2	0.300	0.234	10	3.500	0.322
3	0.500	0.256	11	4.500	0.317
4	0.700	0.277	12	5.500	0.326
5	0.900	0.278	13	6.500	0.323
6	1.250	0.291	14	7.500	0.321
7	1.750	0.308	15	8.500	0.322
8	2.250	0.311	16	9.500	0.328

Tableau 4. Données 2.

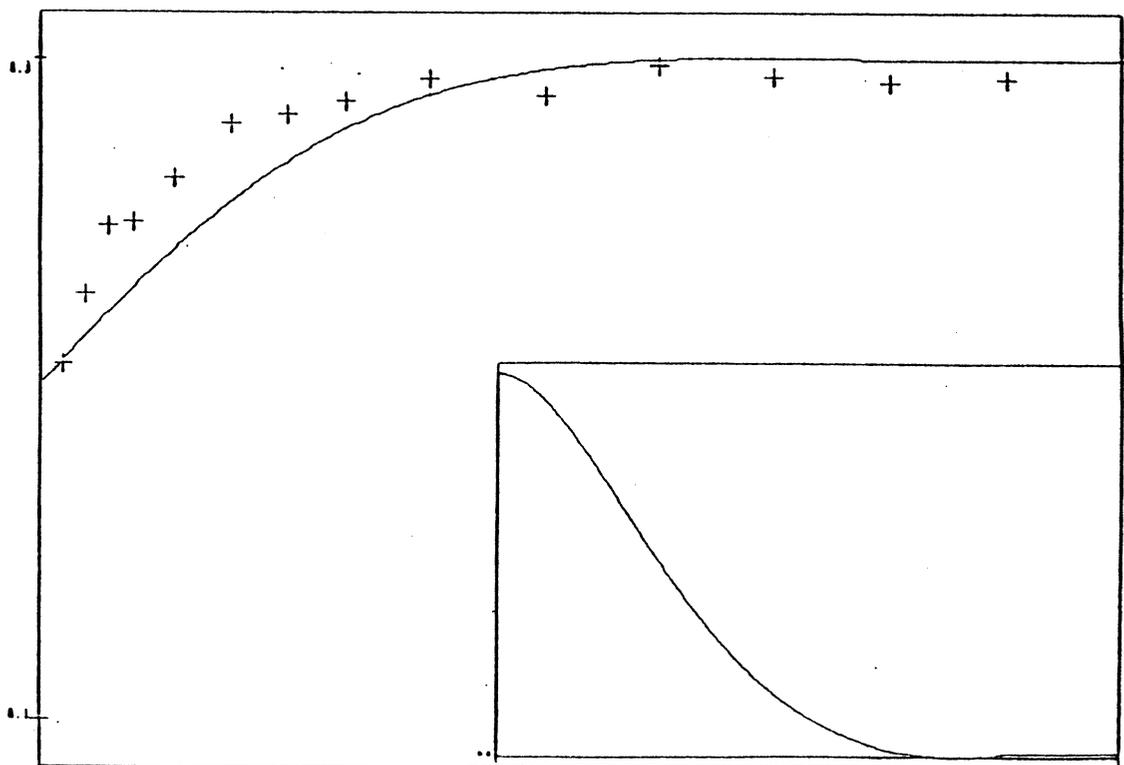
Nous avons lissé ces données pour différentes valeurs de ρ comprises entre 10^{-1} et 10^2 . La tolérance de calcul a été choisie $\epsilon = 10^{-3}$ et $\epsilon = 10^{-5}$. Le tableau 5 regroupe quelques renseignements sur le temps de calcul.

ρ	ϵ	points ajoutés	γ' min.	temps(sec.)
10^{-1}	10^{-3}	1	-4.0144 E-4	1.32
	10^{-5}	2	-8.7492 E-6	5.83
1	10^{-3}	2	-7.9125 E-4	6.62
	10^{-5}	2	-9.4524 E-6	10.21
10	10^{-3}	3	-9.9799 E-4	13.52
	10^{-5}	2	-9.7938 E-6	54.27
10^2	10^{-3}	4	-1.3572 E-4	21.80
	10^{-5}	5	-8.4524 E-6	69.35

Tableau 5.



(a)



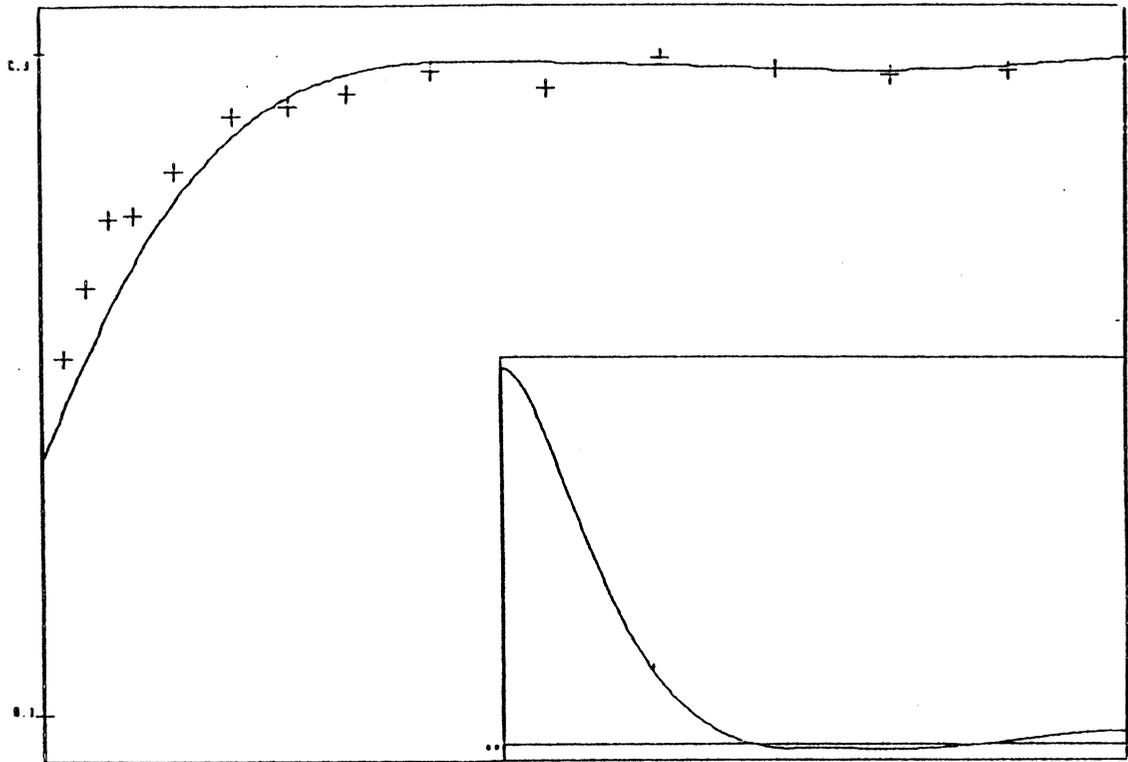
(b)

Fig.7. F.S. lissage pour les donnés 2. ($p=1/10$)

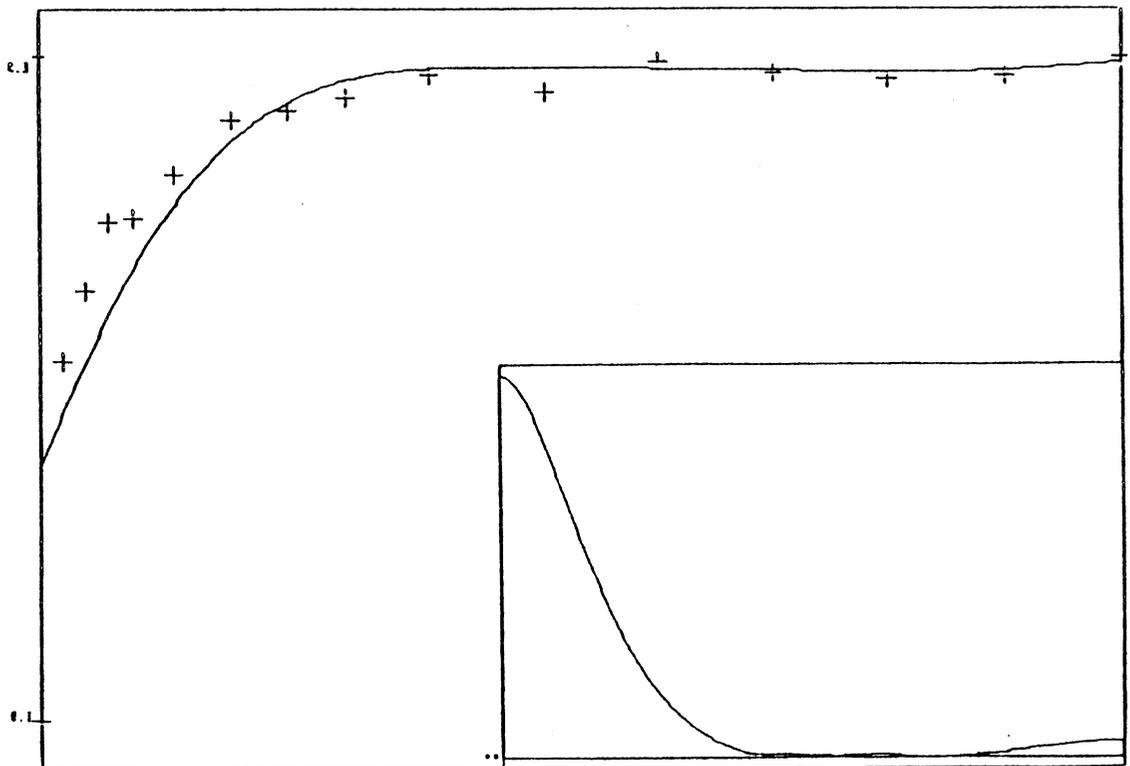
(a) sans contrainte

(b) avec contrainte de monotonie

Dans les fenêtres leur dérivée première



(a)



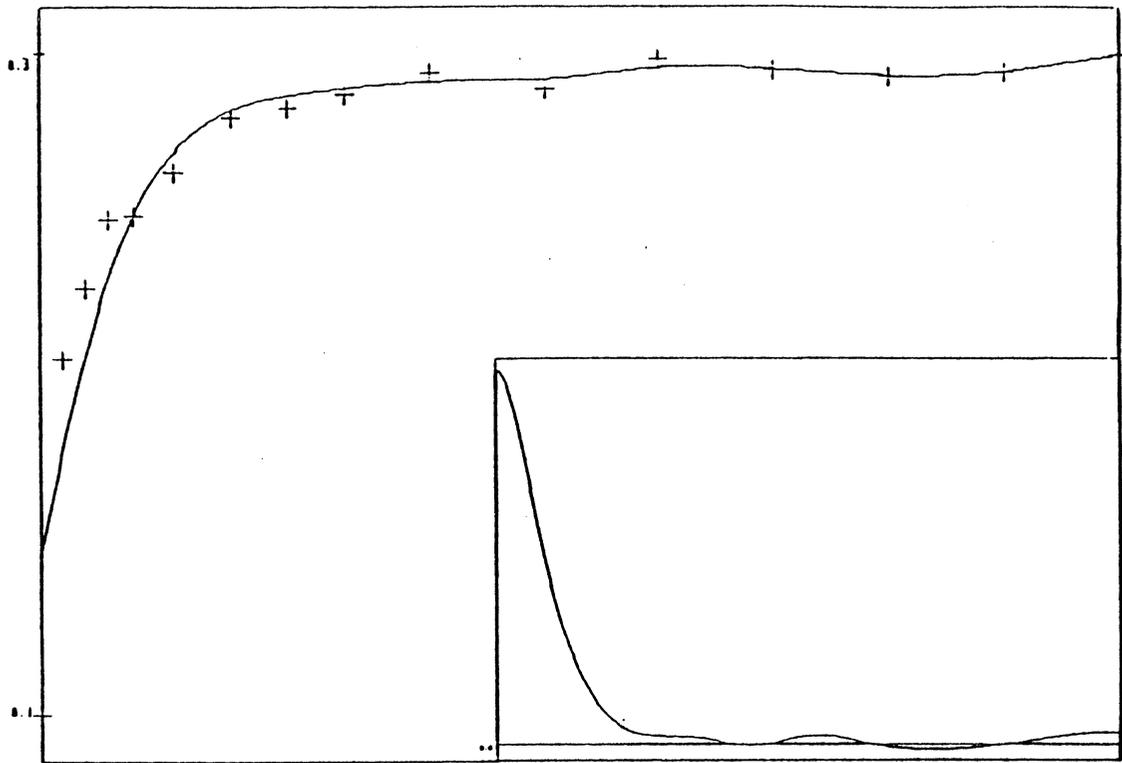
(b)

Fig.8. F.S. lissage pour les donnés 2. ($\rho=1$)

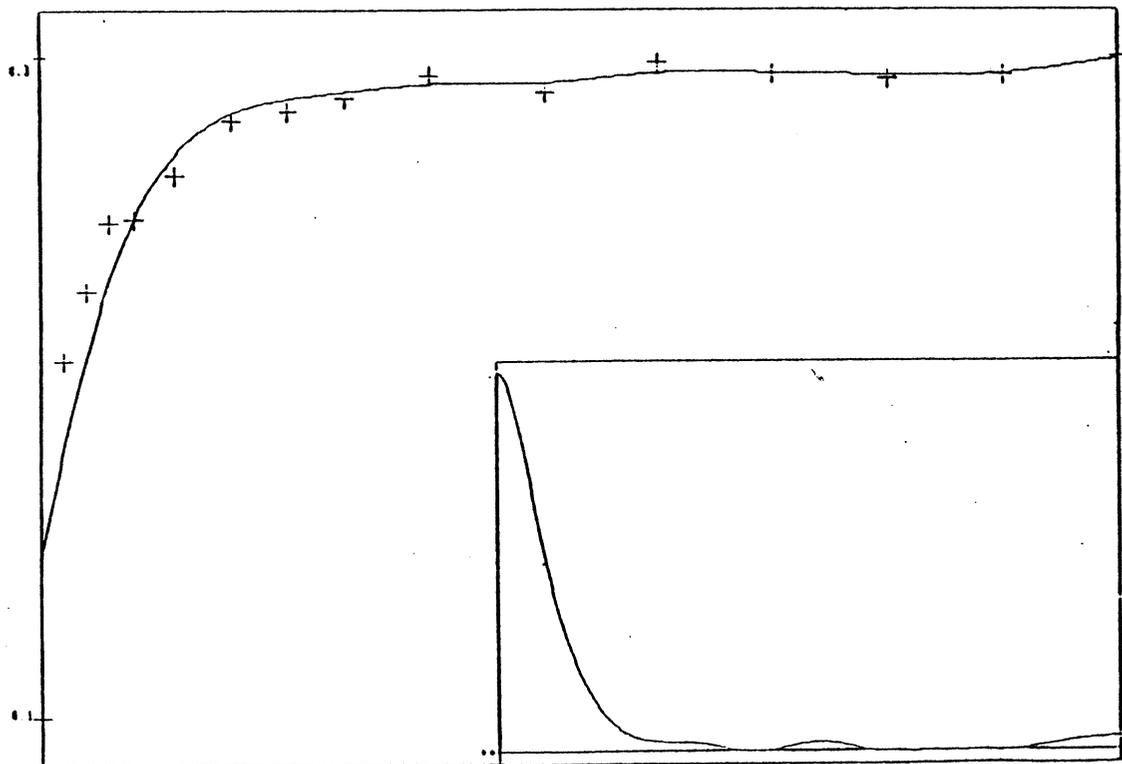
(a) sans contrainte

(b) avec contrainte de monotonie

Dans les fenêtres leur dérivée première ..



(a)



(b)

Fig.9. F.S. lissage pour les donnés 2. ($\rho=100$)

(a) sans contrainte

(b) avec contrainte de monotonie

Dans les fenêtres leur dérivée première

Là encore, des éclatements de contraintes se sont produits, le nombre de points supplémentaires n'est pas égale au nombre d'itérations.

Les courbes qui correspondent aux valeurs de $\rho=10^{-1}$, 1 et 10^2 apparaissent dans les Figs. 7, 8 et 9. En haut nous avons placé les splines sans contraintes et en bas les splines une fois que la contrainte de monotonie est respectée à 10^{-5} près. Les fenêtres contiennent les graphes des dérivées premières sur tout l'intervalle [0.1, 9.5], qui doivent être positives ou nulles si les fonctions sont croissantes.

Les différences plus grandes se situent vers le dernier quart de l'intervalle, ce que l'on remarque mieux en regardant la dérivée. Les zones de décroissance de la fonction ont été éliminées.

- Etudions maintenant la méthode à partir de la fonction

$$f_2(t) = (t-3) / (t^2 - t+1) \quad \text{pour } t \in [0,2].$$

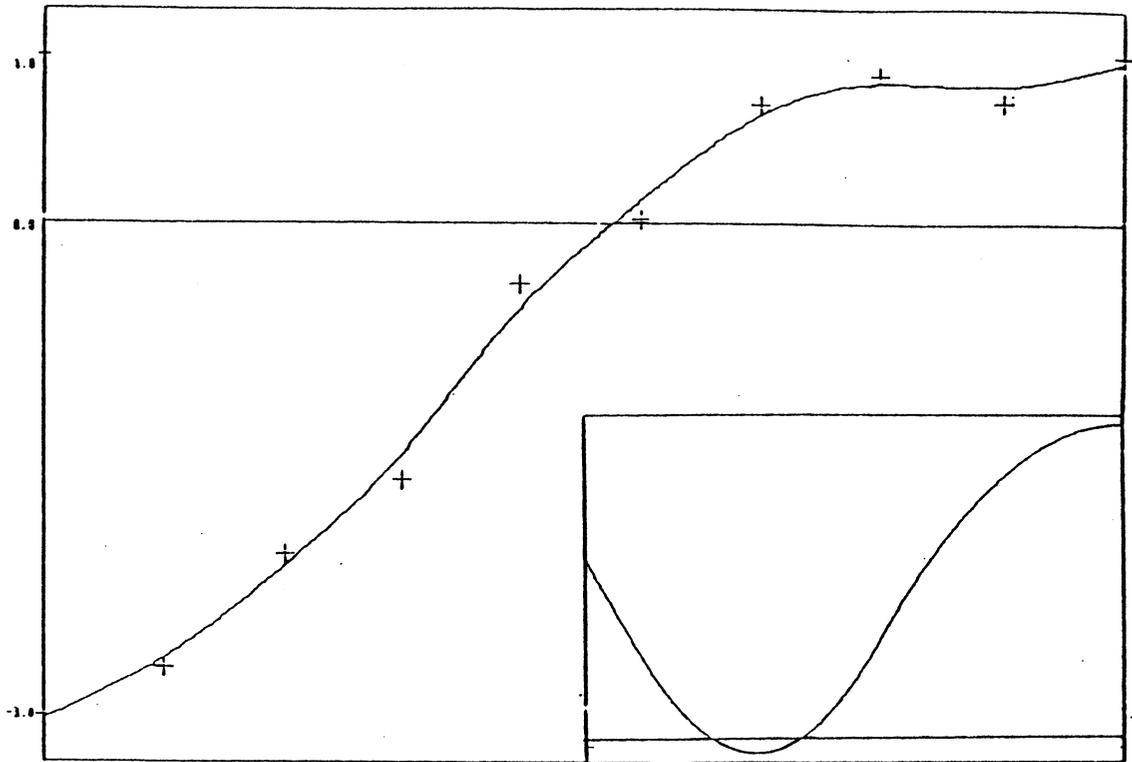
On a choisi n abscisses équidistantes n=10,20,30. L'erreur sur la valeur de f aux noeuds a pour écart type 0.5.

On a calculé les valeurs du paramètre du lissage ρ par la méthode de Reinsch. Quelques résultats se trouvent dans le tableau 6 (la précision est $\epsilon=10^{-4}$).

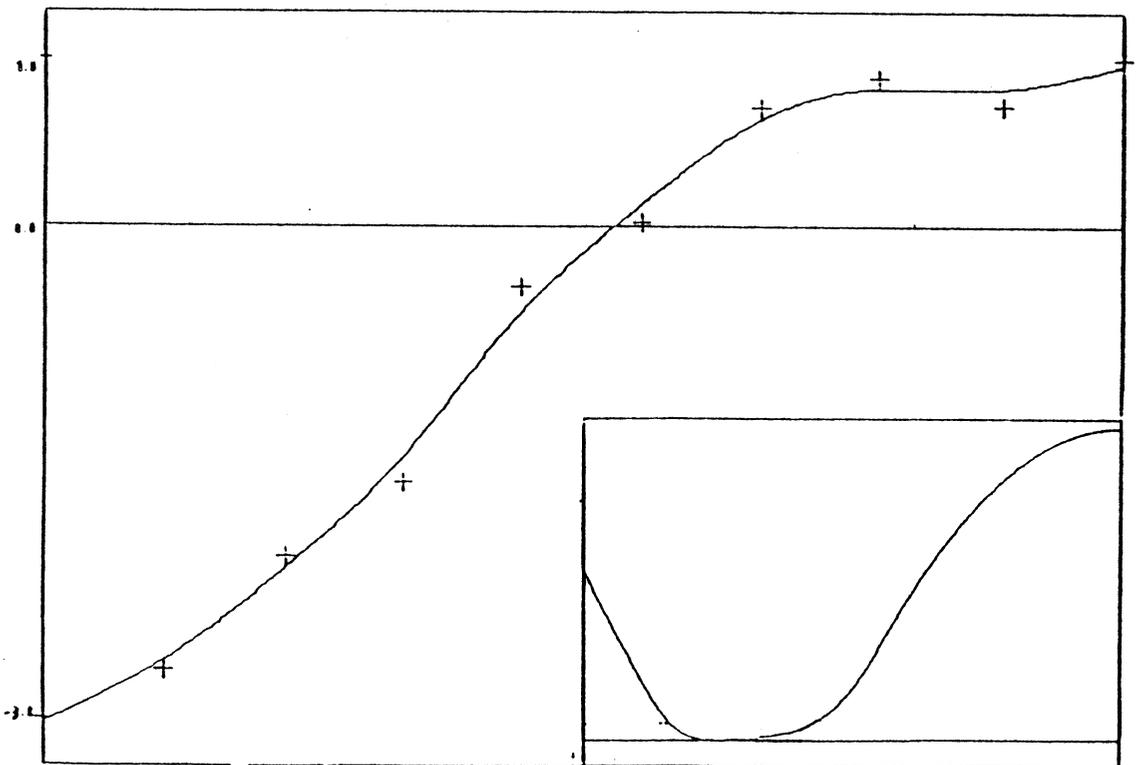
n	ρ	points ajoutés	y' min.	temps(sec.)
10	853.5	2	-9.0187 E-5	74.26
20	995.3	3	-9.4015 E-5	97.80
30	1210.6	5	-9.9408 E-5	135.64

Tableau 6.

Il est clair, en regardant le nombre de points ajoutés et le temps de calcul, que des éclatements de contraintes se sont produits pendant l'exécution du programme.



(a)



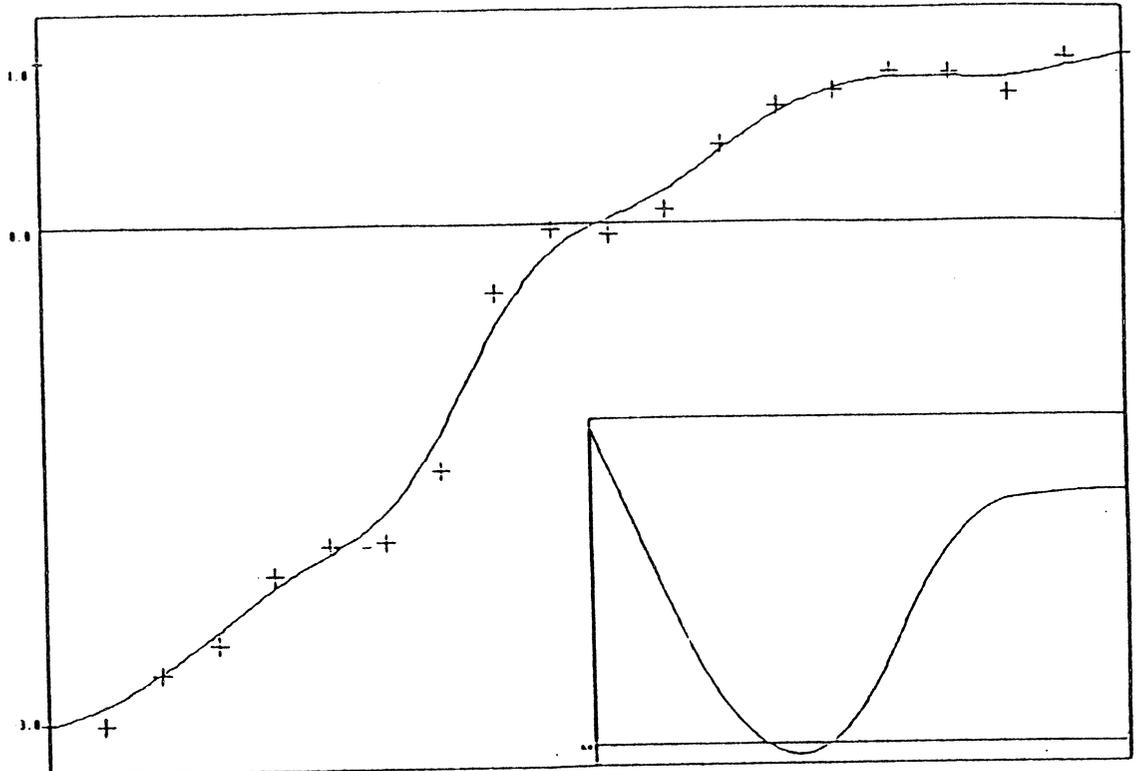
(b)

Fig. 10. F.S. lissage pour $f_2(t)$, $t \in [0, 2]$ ($n=10$)

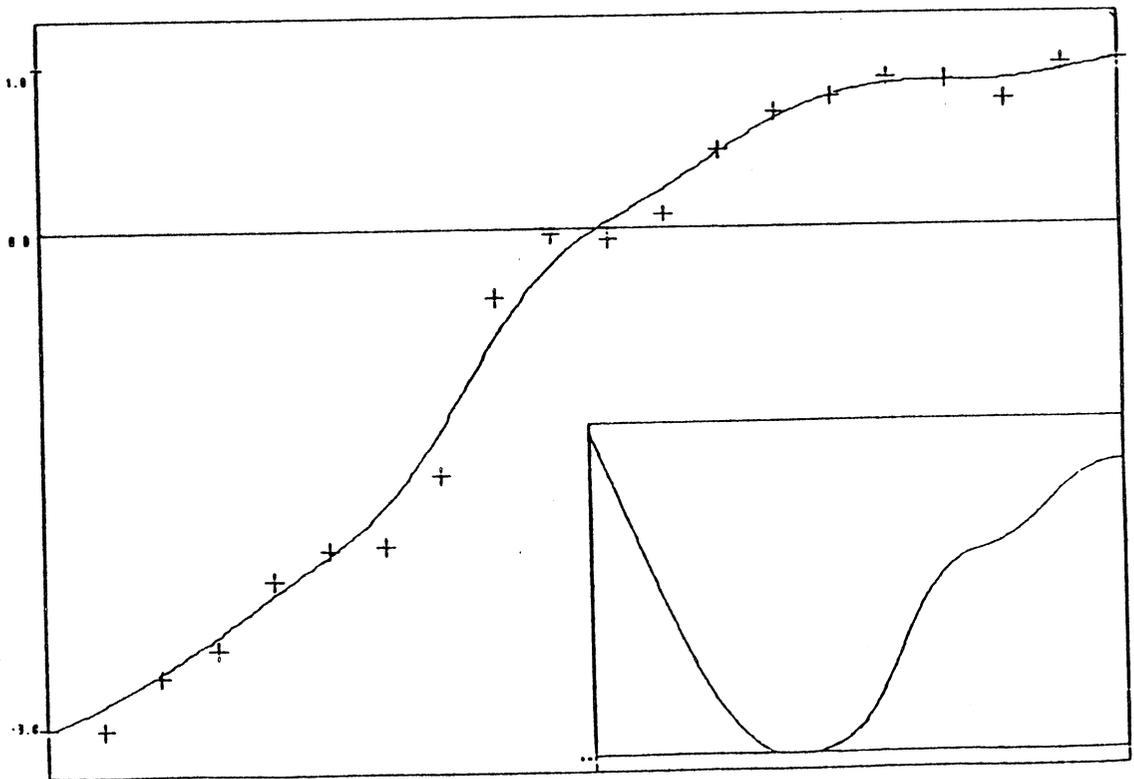
(a) sans contraintes

(b) avec contrainte de monotonie

Dans les fenêtres, les dérivées premières sur l'intervalle $[1.5, 2]$



(a)



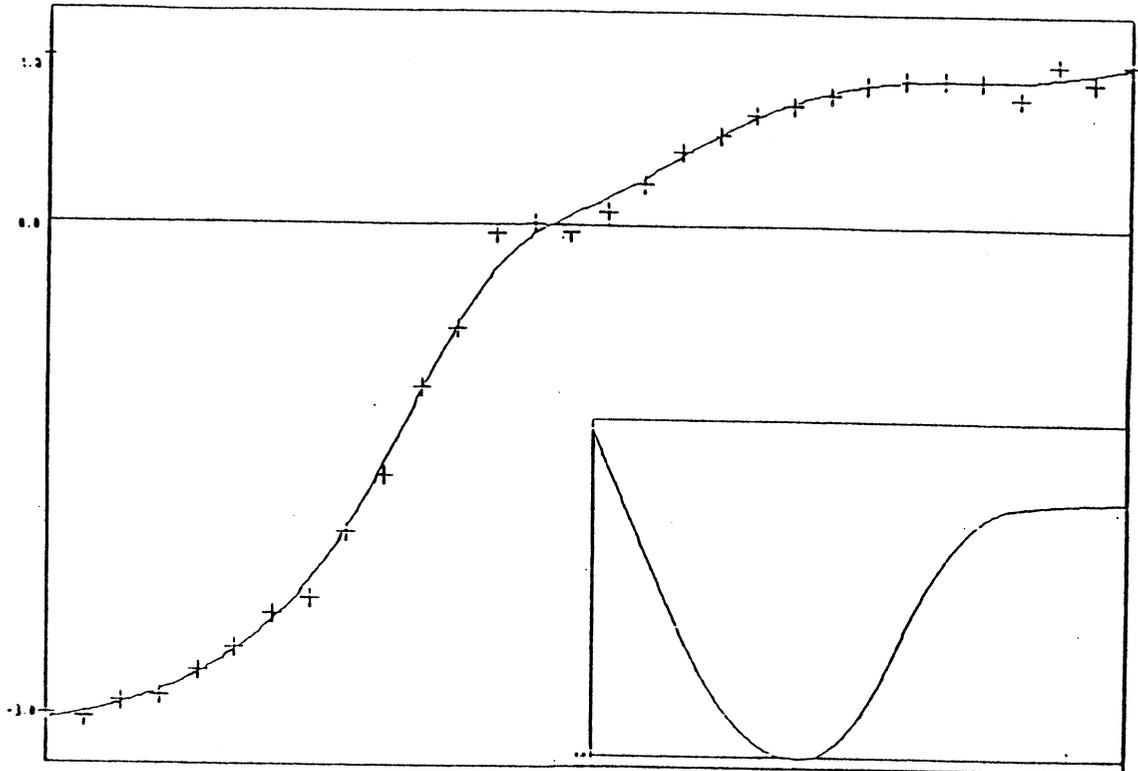
(b)

Fig. 11. F.S. lissage pour $f_2(t)$, $t \in [0, 2]$ ($n=20$)

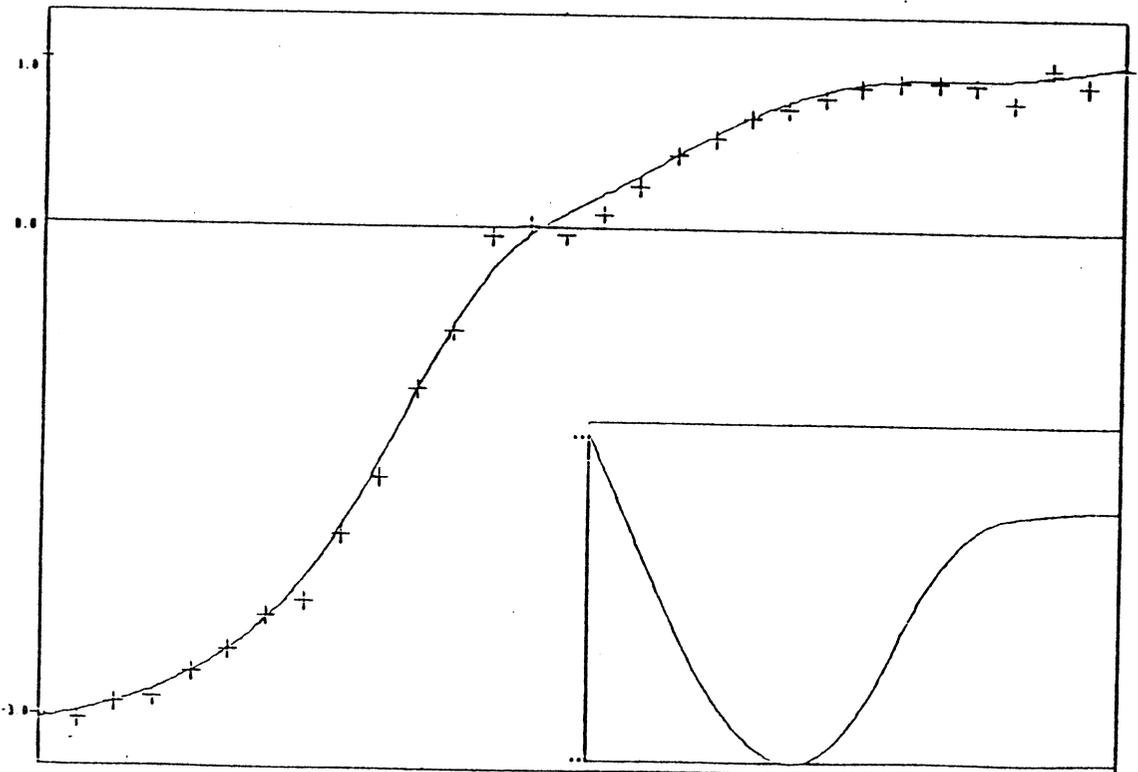
(a) sans contraintes

(b) avec contrainte de monotonie

Dans les fenêtres, les dérivées premières sur l'intervalle $[1.5, 2]$



(a)



(b)

Fig. 12. F.S. lissage pour $f_2(t)$, $t \in [0, 2]$ ($n=30$)

(a) sans contraintes

(b) avec contrainte de monotonie

Dans les fenêtres, les dérivées premières sur l'intervalle $[1.5, 2]$

Les courbes des splines sans contraintes sont contenues dans la partie supérieure des figures 10,11 et 12, tandis qu'en bas on trouve les graphes des splines avec la contrainte de monotonie croissante. Les fenêtres représentent les dérivées premières dans l'intervalle [1.5,2.0], car c'est la zone où les variations des fonctions (donc des dérivées) sont les plus remarquables.

Dans ce cas on peut dire aussi que l'algorithme donne de bons résultats.

IV.6. CONCLUSIONS

Nous avons mis en oeuvre numériquement l'algorithme de Laurent pour le lissage de données avec contrainte de monotonie ou de positivité. Cette méthode considère les contraintes continues et, à la différence d'autres qui discrétisent les contraintes, n'est pas lourde à programmer.

L'algorithme travaille d'une manière satisfaisante lorsqu'on a fait un "bon choix" du paramètre de lissage ρ , car l'approximation de la fonction et le temps de calcul sont sensibles à la variation de ρ . Comme ce paramètre de lissage reste le même au long des calculs, nous conseillons d'utiliser la valeur donnée par la méthode de validation croisée ou de Reinsch pour la spline sans contraintes.

Dans certains cas, par exemple ρ "trop grand", obtenir une précision au delà de 10^{-5} peut demander un temps de calcul assez élevé. Si les applications pratiques le permettent on peut se contenter d'une tolérance de l'ordre de 10^{-4} ou 10^{-5} .

CHAPITRE V

Fonctions Splines avec Repoussoir



V. I. INTRODUCTION

Pour construire des fonctions spline de lissage avec certaines caractéristiques de forme (positivité, monotonie ou convexité), deux approches variationnelles ont été étudiées dans les chapitres III et IV : la première emploie les splines sous tension, la deuxième résout le problème dans le contexte de la programmation mathématique avec contraintes.

Nous avons pu observer que si les paramètres de tension sont "suffisamment" grands ou la contrainte de forme est active, alors la positivité, la monotonie ou la convexité de la spline n'est pas stricte dans certains intervalles, ce qui peut entraîner des oscillations indésirables. Pour éviter de telles situations nous proposons d'introduire un terme de pénalisation intérieure qui "repousse" la fonction de façon infinie lorsque la contrainte est en train de devenir active. Cette idée est appliquée dans le cas où la spline de lissage doit être positive.

Plaçons nous dans l'espace $H^2 = H^2[a,b]$. Considérons n abscisses distinctes

$$a < t_1 < t_2 < \dots < t_n < b$$

et n nombres réels z_1, z_2, \dots, z_n . Supposons $n \geq 2$.

Soit p une fonction définie sur $]0, +\infty[$ à valeurs réelles (on supposera que $p(t) = +\infty$ si $t \in]0, +\infty[$) ayant les propriétés suivantes :

a) p est convexe et a une dérivée continue sur $]0, +\infty[$

b) $p(t) \geq 0, \quad t \in]0, +\infty[$

c) $\lim_{t \rightarrow 0^+} p(t) = +\infty$

On s'intéresse au problème suivant :

$$(P_0) \left\{ \begin{array}{l} \inf \frac{1}{2} \int_a^b [u''(t)]^2 dt + \frac{\rho}{2} \sum_{i=1}^n [u(t_i) - z_i]^2 + \eta \sum_{i=1}^n p(u(t_i)) \\ \text{avec } u \in H^2, \text{ vérifiant } u(t_i) \geq 0 \quad i=1,2,\dots,n. \end{array} \right.$$

Une solution du problème (P_0) sera appelée **fonctions spline relative aux paramètres de lissage ρ et de repoussoir η** . Les coefficients ρ et η sont strictement positifs.

Nous allons d'abord ramener (P_0) à un problème de minimisation dans \mathbb{R}^n et former un problème dual. Ensuite, on examine les méthodes numériques de résolution et on termine avec la présentation des essais numériques.

V.2. ETUDE D'UN PROBLEME EQUIVALENT AU PROBLEME (P_0)

Considérons $X = \mathbb{R}^n$ muni du produit scalaire euclidien, noté $\langle \cdot, \cdot \rangle$ et $C = \mathbb{R}_+^n = \{x = [x_1, \dots, x_n] \in \mathbb{R}^n / x_i \geq 0\}$

Soit $x \in \mathbb{R}^n$ et posons

$$I_x = \{u \in H^2 / u(t_i) = x_i \quad i=1,2,\dots,n\}$$

On définit la fonction

$$\Phi(x) = \inf_{u \in I_x} \int_a^b [u''(t)]^2 dt$$

On montre de la même manière que dans le chapitre III §4.3 que Φ est une fonction convexe quadratique de \mathbb{R}^n dans \mathbb{R} :

$$\Phi(x) = \langle x, \Omega x \rangle$$

la matrice Ω est carrée $(n \times n)$, symétrique et semi-définie positive et ses termes ω_{ij} sont donnés par

$$\omega_{ij} = \int_a^b \sigma_i''(t) \sigma_j''(t) dt = \sigma_i'''(t_j^+) - \sigma_i'''(t_j^-) \quad i, j=1, 2, \dots, n$$

où les σ_i sont les fonctions splines d'interpolation de base.

En plus, Φ est continue et dérivable. Son gradient $\Phi'(x)$ en x est égal à $2 \Omega x$.

Notons $z = [z_1, z_2, \dots, z_n]$ et désignons

$$(5.1) \quad f_1(x) = \frac{1}{2} \langle x, \Omega x \rangle + \frac{1}{2} e \langle x-z, x-z \rangle$$

$$(5.2) \quad f_2(x) = \eta \sum_{i=1}^n p(x_i)$$

$$(5.3) \quad f(x) = f_1(x) + f_2(x)$$

Alors le problème (P_0) est équivalent à :

$$(P) \quad \alpha = \inf_{x \in X} f(x)$$

Remarque

La fonction f est convexe car est la somme de deux fonctions convexes. Alors nous sommes en présence d'un problème de minimisation d'une fonction convexe sur l'hyperoctant strictement positif :

$$\overset{\circ}{C} = \overset{\circ}{\mathbb{R}}_+^n = \{ x = [x_1, x_2, \dots, x_n] / x_i > 0 \quad i=1, 2, \dots, n \}$$

Proposition 5.1

Le problème (P) a une solution unique dans $\overset{\circ}{C}$.

Preuve

Soit $\tilde{x} \in \mathring{C}$. Considérons l'ensemble $G = \{ x \in \mathring{C} / f(x) \leq f(\tilde{x}) \}$. C'est un ensemble borné fermé non vide de \mathbb{R}^n . Donc (P) admet un minimum dans \mathring{C} . En fait f_1 est fortement convexe, alors la solution est unique. \square

Remarques

1) Soit \bar{x} la solution unique de (P) et notons $g(\bar{x})$ le gradient de $f(\bar{x})$. Alors le théorème de caractérisation du minimum d'une fonction dérivable entraîne :

$$(5.4) \quad g(\bar{x}) = \Omega \bar{x} + \rho(\bar{x}-z) + \eta g_p = 0$$

$$\text{où } g_p = [p'(\bar{x}_1), \dots, p'(\bar{x}_n)]$$

Nous pouvons écrire aussi pour $i=1,2,\dots,n$

$$(5.5) \quad g_i(\bar{x}) = \sigma_{\bar{x}}''(t_i^+) - \sigma_{\bar{x}}''(t_i^-) + \rho(\bar{x}_i - z_i) + \eta p'(\bar{x}_i) = 0$$

où $\sigma_{\bar{x}}$ est la fonction spline d'interpolation relative à \bar{x}

(c'est-à-dire vérifiant $\sigma(t_i) = \bar{x}_i \quad i=1,2,\dots,n$).

2) Si dans le problème (P_0) on suppose que p est semi-continue inférieurement et on utilise la notion de sous-différentiel on pourrait arriver à un résultat comparable à (5.5).

V.3. UN PROBLEME DUAL ASSOCIE A (P)

Dans ce paragraphe on va utiliser les méthodes introduites par Laurent au chapitre VII de [56].

Prenons $U = \mathbb{R}^n$ comme espace de perturbations et considérons le problème perturbé :

$$(P_U) \quad \inf_{x \in X} (f_1(x) - f_2(x-U)) \quad U \in U$$

On a

$$\varphi(x,u) = f_1(x) + f_2(x-u)$$

La fonctionnelle Ψ (polaire de φ) et le lagrangien L sont :

$$(5.6) \quad \Psi(y,v) = f_1^*(v+y) + f_2^*(-v)$$

$$(5.7) \quad L(x,v) = f_1(x) - f_2^*(-v) - \langle x, v \rangle \quad \text{où } y, v \in \mathbb{R}^n.$$

Donc le problème dual associé à ces perturbations est le suivant :

$$(Q) \quad \beta = \inf_{y \in \mathbb{R}^n} h(y)$$

où

$$h(y) = h_1(y) + h_2(-y)$$

(5.8)

$$h_1(y) = f_1^*(y), \quad h_2 = f_2^*(y)$$

Proposition 5.2

Le problème (Q) a au moins une solution.

Preuve

On remarque d'abord que le problème (P) est inf-dif-stable. En effet, α est fini et f_1 et f_2 sont finies et continues en $x \in \overset{\circ}{C}$. D'après la proposition 7.8.3 ([56], pp 412) le problème (P) est stable, donc inf-dif-stable. Alors $-\beta = \alpha$ et le problème (Q) a au moins une solution. \square

On va calculer f_1^* et f_2^* . On a :

$$(5.9) \quad f_2^*(y) = \begin{cases} \eta \sum_{i=1}^n p^*(y_i/\eta) & \text{si } y_i < 0 \quad \forall i=1,2,\dots,n \\ +\infty & \text{sinon} \end{cases}$$

et

$$(5.10) \quad f_1^*(y) = \sup_{x \in \mathbb{R}^n} \left(\langle x, y \rangle - \frac{1}{2} \langle x, \Omega x \rangle - \frac{\rho}{2} \langle x-z, x-z \rangle \right)$$

Le sup est atteint pour

$$(5.11) \quad \tilde{x} = \rho(\Omega + \rho I)^{-1} (z + y/\rho)$$

c'est-à-dire pour le vecteur des ordonnées de la fonction spline de lissage associée à $z + y/\rho$.

Alors on peut écrire :

$$(5.12) \quad f_1^*(y) = \inf_{x \in \mathbb{R}^n} \left(\frac{1}{2} \langle x, \Omega x \rangle - \frac{\rho}{2} \langle x - (z + y/\rho), x - (z + y/\rho) \rangle \right) + \frac{1}{2\rho} \langle y, y \rangle + \langle z, y \rangle$$

La borne inférieure n'est autre que l'énergie de la fonction spline de lissage relative à $z + y/\rho$.

En fait on n'a pas besoin de cette expression si l'on utilise le lagrangien, qui est caractérisé par:

$$L(x, y) = f_1(x) - f_2^*(-y) - \langle x, y \rangle$$

$$\text{donc } f(x) = \sup_{y \in \mathbb{R}^n} L(x, y) \quad \text{et} \quad h(y) = \inf_{x \in \mathbb{R}^n} L(x, y)$$

Pour un y donné, la minimisation en x de $L(x, y)$ correspond à x_y tel que :

$$\Omega x_y + \rho (x_y - z) - y = 0$$

$$\text{ou } \Omega x_y + \rho (x_y - (z + y/\rho)) = 0$$

c'est-à-dire, à nouveau le vecteur des ordonnées de la fonction spline de lissage aux données modifiées $z + y/\rho$.

Remarque

On a $\tilde{x} \in \partial f_1^*(y)$ et comme $f_1^*(y)$ est dérivable, \tilde{x} est son gradient. Le gradient de $f_2^*(y)$ est $[p^*(y_1/\eta), \dots, p^*(y_n/\eta)]$. Alors

$$(5.13) \quad h'(y) = \tilde{x} + [p^*(-y_1/\eta), \dots, p^*(-y_n/\eta)]$$

V.4. ALGORITHMES DE DESCENTE POUR LA RESOLUTION DE (P) ET (Q)

Nous allons décrire les méthodes dites directes ou de descente pour la résolution numérique de (P). On peut procéder de manière analogue pour (Q).

Ces méthodes consistent à construire une suite de points qui appartiennent tous au convexe $\overset{\circ}{C}$ et pour lesquels les valeurs de la fonction f forment une suite décroissante.

Etant donné $x^k \in \overset{\circ}{C}$, x^k non solution de (P), le point successeur x^{k+1} est déterminé par le choix d'une direction $d^k = d(x^k)$, appelé *direction de descente* ou *de déplacement*, et par le choix d'un pas λ^k dans cette direction :

$$(5.14) \quad x^{k+1} = x^k + \lambda^k d^k$$

4.1. Choix de la direction de descente.

Supposons que l'on ait obtenu $x^k \in \overset{\circ}{C}$ (non solution), la direction de descente doit être telle que :

- (i) l'on puisse trouver des points de $\overset{\circ}{C}$ de la forme $x^k + \alpha d^k$ avec $\alpha > 0$.
- (ii) si α est "assez petit" on ait $f(x^k + \alpha d^k) < f(x^k)$. Ici, $\overset{\circ}{C}$ est un ouvert donc c'est toujours possible.

Une condition suffisante pour avoir (i) et (ii) est :

$$\langle f'(x^k), d^k \rangle < 0$$

Ici nous considérons deux choix possibles pour d^k :

d1) La direction opposée à celle du gradient, c'est-à-dire

$$(5.15) \quad d_i^k = -g_i(x^k) \quad i=1,2,\dots,n$$

(on note ici d_i^k , les composantes de $d(x^k) = d^k$ et $g_i(x^k)$ les composantes de $f'(x^k)$).

d2) Le gradient projeté

On construit la projection (au sens de la norme euclidienne) de l'élément $x^k - \alpha f'(x^k)$ sur $C = \mathbb{R}_n^+$ ($\alpha > 0$). Elle est notée

$\bar{x}^k = P_C(x^k - \alpha f'(x^k))$. On prend comme direction de déplacement $d^k = \bar{x}^k - x^k$. Alors on a :

$$(5.16) \quad d_i^k = \begin{cases} -x_i^k & \text{si } x_i^k \leq \alpha g_i(x^k) \\ -\alpha g_i(x^k) & \text{sinon.} \end{cases}$$

4.2. Recherche linéaire

On détermine d'abord

$$(5.17) \quad \theta^k = \theta(x^k) = \sup \{ \lambda \geq 0 / x^k + \lambda d^k \in \overset{\circ}{C} \}.$$

Pour obtenir λ^k on peut utiliser deux méthodes :

$\lambda 1$) rechercher λ^k tel que

$$(5.18) \quad \lambda^k = \min_{\lambda \in [0, \theta^k]} f(x^k + \lambda d^k)$$

ce qui revient à minimiser une fonction à une seule variable.

λ^k) faire un déplacement ni "trop grand" (pour assurer la décroissance de la suite), ni "trop petit" (pour assurer la convergence). Pour cela on choisit $\lambda^k \in [0, \theta^k]$ de telle sorte qu'il satisfasse :

$$(5.19) \quad f(x^k + \lambda^k d^k) - f(x^k) \leq \mu_1 \lambda^k \langle g^k, d^k \rangle$$

et une des conditions suivantes :

$$(5.20) \quad f(x^k + m\lambda^k d^k) - f(x^k) \geq m\mu_1 \lambda^k \langle g^k, d^k \rangle \quad (m > 0) \quad \text{ou}$$

$$(5.21) \quad f(x^k + \lambda^k d^k) - f(x^k) \geq \mu_2 \lambda^k \langle g^k, d^k \rangle \quad \text{ou}$$

$$(5.22) \quad \langle g(x^k + \lambda^k d^k), d^k \rangle \geq \mu_2 \langle g^k, d^k \rangle$$

où les constantes μ_1 et μ_2 sont telles que $0 < \mu_1 < \mu_2 < 1$.

Les règles (5.20), (5.21) et (5.22) ont été proposées, respectivement, par Armijo [3], Wolfe [118] et Goldstein et Price [44].

4.3. Description de l'algorithme

0). Soit $x^0 \in \overset{\circ}{C}$ un point initial et $\epsilon > 0$ un seuil. On pose $k = 0$.

1). Définir une direction de descente d^k .

2). On calcule θ^k défini par (5.17).

3). On détermine les pas $\lambda^k > 0$, tel qu'il

- réalise le minimum de $f(x^k + \lambda d^k)$ par rapport à $\lambda \in]0, \theta^k[$

- ou satisfasse la condition (5.19) et une des règles (5.20), (5.21) ou (5.22).

On pose $x^{k+1} = x^k + \lambda^k d^k$

4). Si le test d'arrêt est vérifié : fin, sinon on va à 1) avec $k = k+1$.

4.4. Convergence de l'algorithme

Soit $T = \{ x \in \overset{\circ}{C} / x \text{ n'est pas solution de } (P) \}$. On note d la fonction qui à \bar{x} associe une direction de descente définie par (5.15) ou (5.16), et θ la fonction définie en x par (5.17).

Alors on constate que :

- 1) $\forall x \in T, \langle f'(x), d(x) \rangle < 0$
- 2) d et θ sont continues par rapport à x
- 3) $\forall x \in T, x + \theta(x)d^k(x) \in C$.
- 4) $G = \{ x \in \overset{\circ}{C} / f(x) \leq f(x^*) \}$ est borné et fermé (donc compact).

Proposition 5.3

Toute sous-suite convergente de (x_n^k) admet pour limite une solution, et $\lim_{k \rightarrow \infty} f(x^k) = \alpha$.

Preuve

Par construction la suite $f(x^k)$ est décroissante : $f(x^{k+1}) < f(x^k)$.

La suite x^k parcourt le compact G . Soit $(x^k)_{k \in N_1}, N_1 \subseteq \mathbb{N}$, une sous-suite convergente dont la limite est notée \tilde{x} . On a $d(x^k) \rightarrow \tilde{d} = d(\tilde{x})$ pour $n \in N_1$.

Supposons que \tilde{x} ne soit pas solution. On aurait donc $\langle f'(\tilde{x}), \tilde{d} \rangle < 0$. Comme $\theta(\tilde{x}) > 0$ et θ est continue $\exists k_0$ et $\nu > 0$ tels que $\forall k \in N_1, k > k_0$ on ait $\theta(x^k) > \nu$ et par conséquent $x^k + \nu d^k \in G$. Donc

$x^k + \lambda d^k \in \overset{\circ}{C} \quad \forall \lambda \in [0, \nu]$, et

$f(x^{k+1}) = f(x^k + \lambda^k d(x^k)) \leq f(x^k + \lambda d(x^k)) \quad \forall \lambda \in [0, \nu], \forall \lambda \in N_1, k \geq k_0$.

Passons à la limite pour $n \in N_1$, on a $\lim_{n \in N_1} f(x^k + \lambda d(x^k)) = f(\tilde{x} + \lambda \tilde{d})$

car d et f sont continues.

La suite $f(x^k)$ est monotone décroissante et bornée inférieurement, donc elle est convergente et sa limite ne peut être que $f(\tilde{x})$.

Par la compacité de G , il existe $N_2 \subseteq N_1$ tel que $\lim_{n \in N_2} x^{k+1} = \tilde{x}$

La continuité et la monotonie de $f(x^k)$ entraînent $f(\tilde{x}) = f(\tilde{x})$.

Si l'on passe à la limite pour $k \in N_1$ on a :

$$f(\tilde{x}) \leq f(\tilde{x} + \lambda \tilde{d}) \quad \forall \lambda \in]0, \nu]$$

Ce qui est contradictoire avec le fait que $\langle f'(\tilde{x}), \tilde{d} \rangle < 0$. \square

4.5. Remarques sur la mise en oeuvre numérique

- Pour les termes de pénalisation nous employons

$$p_1(t) = 1/t^n \quad n=1,2$$

$$p_2(t) = e^{1/t} - 1 \quad t \in]0, +\infty[$$

- Pour évaluer $f(x) = f_1(x) + f_2(x)$ il faut savoir calculer le terme

$\Phi(x) = \langle x, \Omega x \rangle / 2$. Deux stratégies peuvent être envisagées:

- Dans la première, on explicite la matrice Ω et on la garde en mémoire pendant l'exécution de l'algorithme ; ceci suppose n activations de la procédure de calcul de la fonction spline d'interpolation correspondant aux n splines de base.

- Dans la deuxième stratégie on ne calcule pas la matrice Ω et on procède de la manière suivante : pour un vecteur x donné on calcule la fonction spline d'interpolation σ_x correspondante et le

vecteur gradient $\Phi'(x) = \text{grad } \Phi$ caractérisé par :

$$(\text{grad } \Phi)_i = \sigma'''_x(t_i^+) - \sigma'''_x(t_i^-) \quad i=1,2,\dots,n$$

et on a $\Phi(x) = \langle x, \Phi'(x) \rangle / 2$

• Le montant $h(y) = f_1^*(y) + f_2^*(-y)$ du problème dual (Q) est aussi facile à trouver. Le terme $\langle y, \Omega y \rangle / 2$ est égal à $\langle \tilde{x}, \tilde{\delta} \rangle / 2$ où \tilde{x} est le vecteur des ordonnées de la fonction spline de lissage relatif à $z+y/\rho$ et $\tilde{\delta}$ est le vecteur des discontinuités de la dérivée troisième de cette même spline. Le gradient de $f_1^*(y)$ est encore \tilde{x} .

V.5. ESSAIS NUMERIQUES

Dans ce paragraphe on va étudier le comportement des méthodes décrites au §5.3. à partir des fonctions suivantes :

$$\text{fonc}_1(t) = t^2 e^t \quad t \in [-5, 1.0]$$

$$\text{fonc}_2(t) = e^{-t^2} \quad t \in [-2.5, 2.5]$$

Nous prenons par commodité n abscisses équidistantes ($n=10,20,30$). L'erreur sur la valeur de fonc_1 et fonc_2 aux noeuds a pour écart type 0.5.

On prend les fonctions p_1 et p_2 comme fonctions de pénalité intérieure.

• Dans les problèmes que nous considérons il y a deux paramètres à choisir ρ et η . Comme le terme de pénalisation est non-linéaire, pour déterminer ρ on peut appliquer la méthode de validation croisée non-linéaire, mais elle est difficile à mettre en oeuvre. On peut utiliser la méthode de Reinsch de la manière suivante : on prend pour ρ la valeur pour laquelle les ordonnées x_i de la fonction spline vérifient :

$$\sum_{i=1}^{n_1} (x_i - z_i)^2 \approx kn_1 \quad \text{et} \quad \sum_{j=1}^{n_2} (x_j - z_j)^2 \approx kn_2$$

avec $n_1 + n_2 = n$ et $k \in [0.7, 0.95]$. Les indices $j=1, 2, \dots, n_2$ correspondent à $i = n_{j+1}, n_{j+2}, \dots, n$.

Toutefois, nous n'avons pas obtenu de résultats satisfaisants, dû peut-être au faible nombre de points.

Finalement, nous avons procédé par tâtonnement.

- Pour démarrer les algorithmes les composantes x_i^0 $i=1, 2, \dots, n$ du point de départ ont été prises égales aux valeurs de la fonction $\text{fonc}_j(t_j)$ $j=1, 2, i=1, 2, \dots, n$ arrondies à la troisième décimale.

- Le test d'arrêt pour les itérations est :

$$\|x^{k+1} - x^k\| < \epsilon, \quad \epsilon = 10^{-5}.$$

*** Exemple 1**

Ici on considère la fonction fonc_1 .

Une première série de calculs a pour but de donner une indication sur le comportement de la méthode du gradient projeté appliquée au problème (P), avec la pénalisation correspondante à p_1 . Le tableau 1 contient les renseignements relatifs au nombre d'itérations et le temps de calcul pour les valeurs $\rho=100$ et $\eta=100$.

n	Itérat.	temps(sec.)
10	19	13.42
20	24	28.15
30	37	47.78

Tableau 1. Problème (P). Méthode du gradient projeté

Maintenant considérons le problème dual (Q) avec les fonctions de pénalisation p_1 et p_2 et $\rho = \eta = 100$. Les résultats numériques qui correspondent à la méthode du gradient projeté sont résumés dans le tableau 2.

n	pénalisation p_1		pénalisation p_2	
	itér.	temps(sec.)	itér.	temps(sec.)
10	11	9.75	9	6.38
20	16	18.34	13	15.42
30	30	36.80	25	26.14

Tableau 2. Problème (Q). Méthode du gradient projeté

En regardant les tableaux 1 et 2 on constate que :

- le problème dual (Q) est résolu plus rapidement que le problème primal (P). Nous pensons que cela est dû au fait que la résolution de (Q) demande le calcul de la fonction spline de lissage associée à $z + y^k / \rho$ tandis que pour (P) on doit calculer la fonction spline d'interpolation correspondante à y^k .

- Le nombre d'itérations (et donc le temps de calcul) est plus petit lorsque la fonction de pénalité intérieure est p_2 . Dans ce cas on repousse plus fortement la fonction car on est en présence d'une fonction exponentielle.

Nous avons remarqué ces deux tendances sur d'autres exemples.

Continuons à traiter le problème dual avec la fonction de pénalisation $p_1(t)$. Pour ces mêmes valeurs $\rho = \eta = 100$, nous avons appliqué la méthode de la plus profonde descente. Les résultats sont regroupés dans le tableau 3.

n	itérat.	temps(sec.)
10	12	10.87
20	19	22.16
30	33	37.54

Tableau 3. Problème (Q). Méthode du gradient.

Les figures 1 et 2 illustrent les situations décrites pour l'exemple 1 lorsque $n=20$ et $\rho=\eta=100$. Le dessin 1(a) montre la spline solution du problème (P) par la méthode du gradient projeté. En bas, (Fig.1(b)) apparaît la courbe qui correspond à la solution de (Q) par la même méthode. La fonction de pénalité intérieure, dans les deux cas, est associée à p_1 .

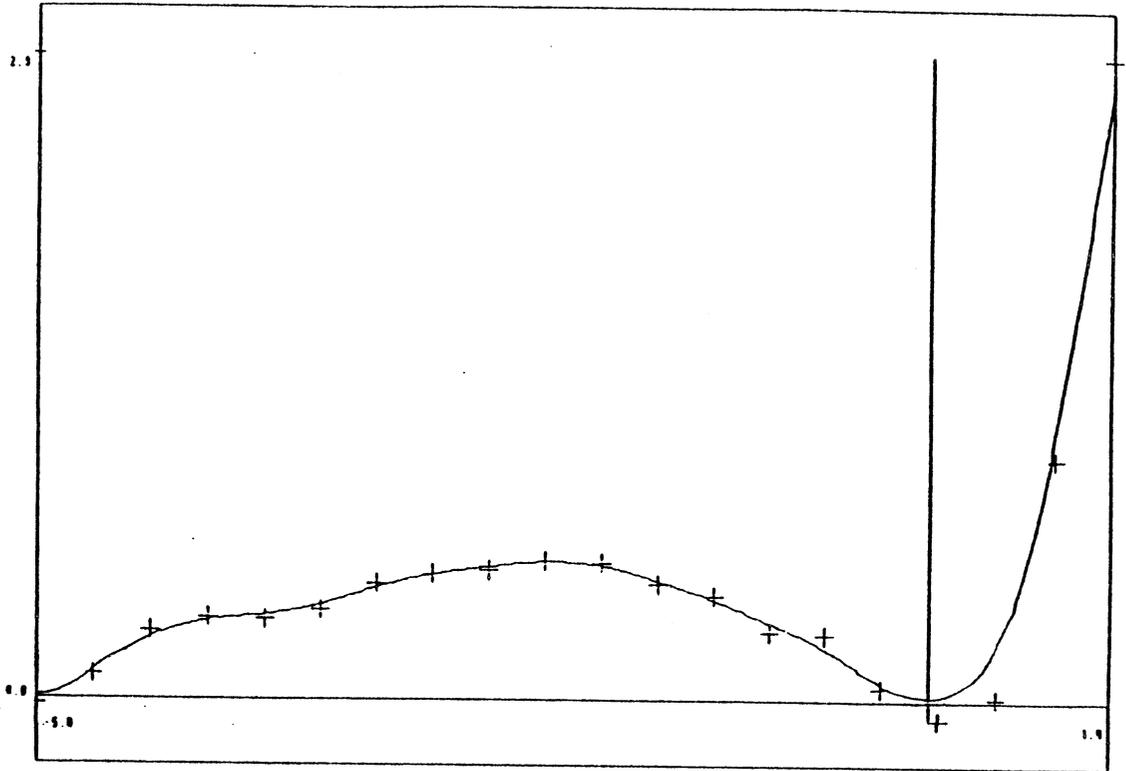
La Fig. 2 contient les courbes qui proviennent de la résolution de (Q). Pour obtenir le dessin 2(a) on a employé la méthode de la plus profonde descente avec le repoussoir correspondant à p_1 . Si on utilise la méthode du gradient projeté avec la pénalisation associée à p_2 , on arrive au résultat graphique 2(b).

Nous voyons que la courbe 1(b) est "plus plaisante" et dans le graphe 2(b) on peut constater qu'il est plus repoussé car la pénalisation contient des termes exponentiels.

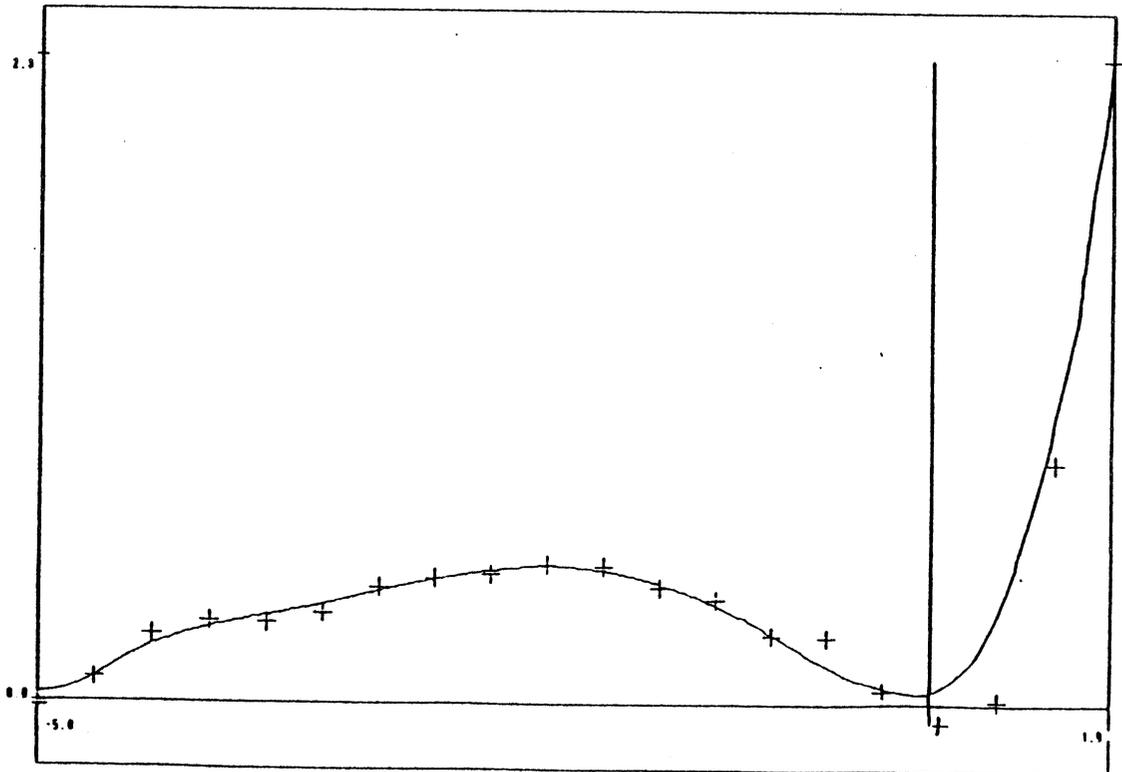
**** Exemple 2**

Pour les essais avec fonc_2 nous avons traité le problème dual (Q) avec le repoussoir correspondant à p_1 . Les deux méthodes, de la plus profonde descente et du gradient projeté ont été appliquées.

Les résultats qui correspondent aux différentes valeurs de ρ et η se trouvent dans le tableau 4.

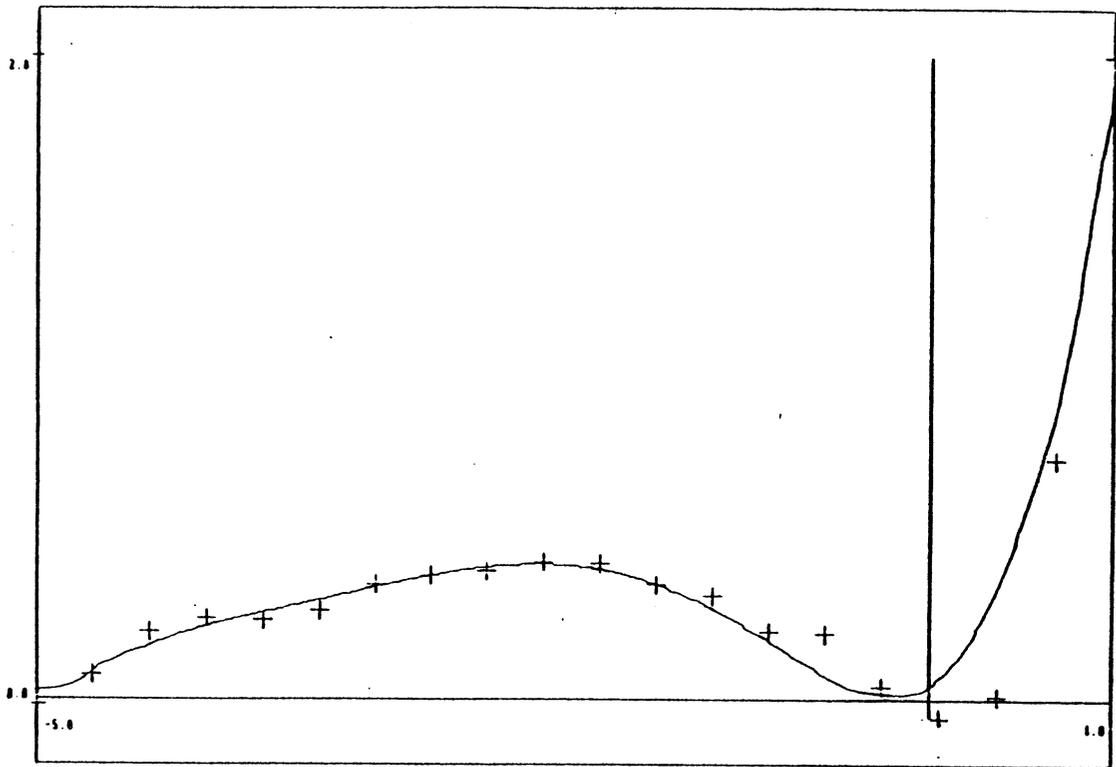


(a)

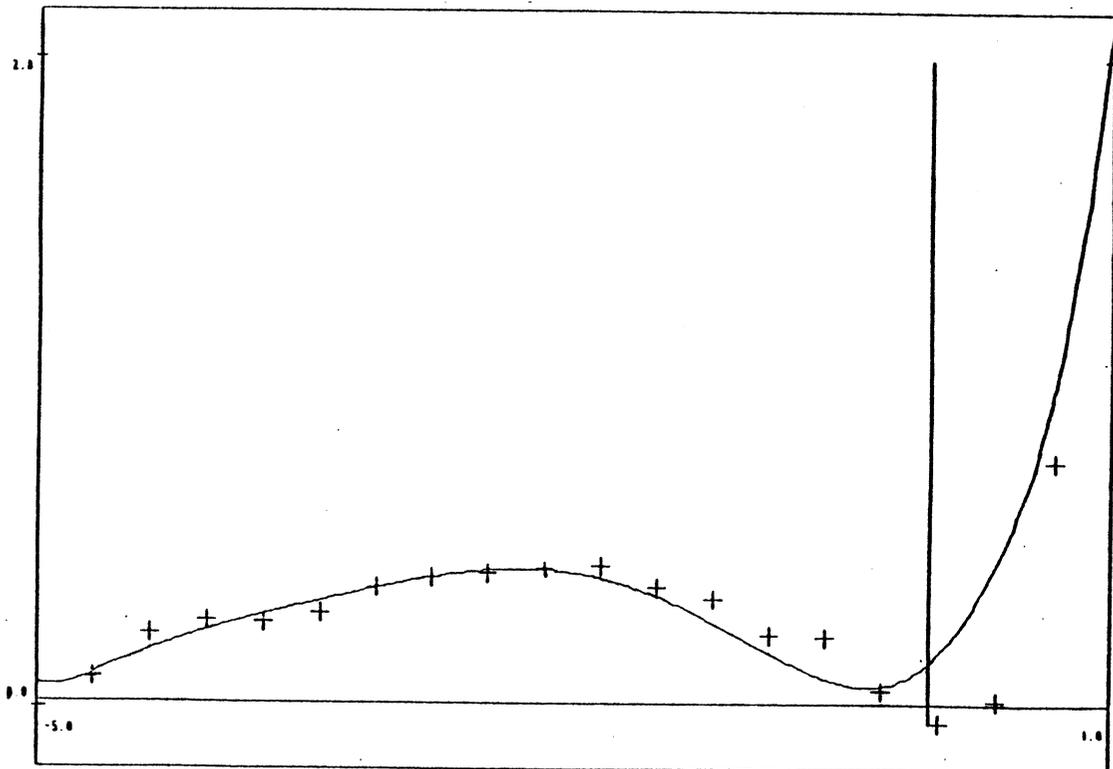


(b)

Fig. 1. Exemple 1. Méthode du gradient projeté ($\rho = \eta = 100$)
(a) Problème (P) avec repoussoir p_1
(b) Problème (Q) avec repoussoir p_1



(a)



(b)

Fig.2. Exemple 1. Problème (Q) ($\rho = \eta = 100$)

(a) Méthode du gradient. Pénalisation p_1

(b) Méthode du gradient projeté. Pénalisation p_2

n	p	η	Méth. du gradient		Méth. du gradient projeté	
			itér.	temps(sec.)	itér.	temps(sec.)
10	150	40	25	24.62	22	18.75
		100	32	34.83	32	34.12
20	200	40	33	42.10	29	35.59
		150	41	53.55	37	50.18
30	100	40	30	59.68	29	55.27
		200	45	80.52	44	77.12

Tableau 4 . Problème (Q).

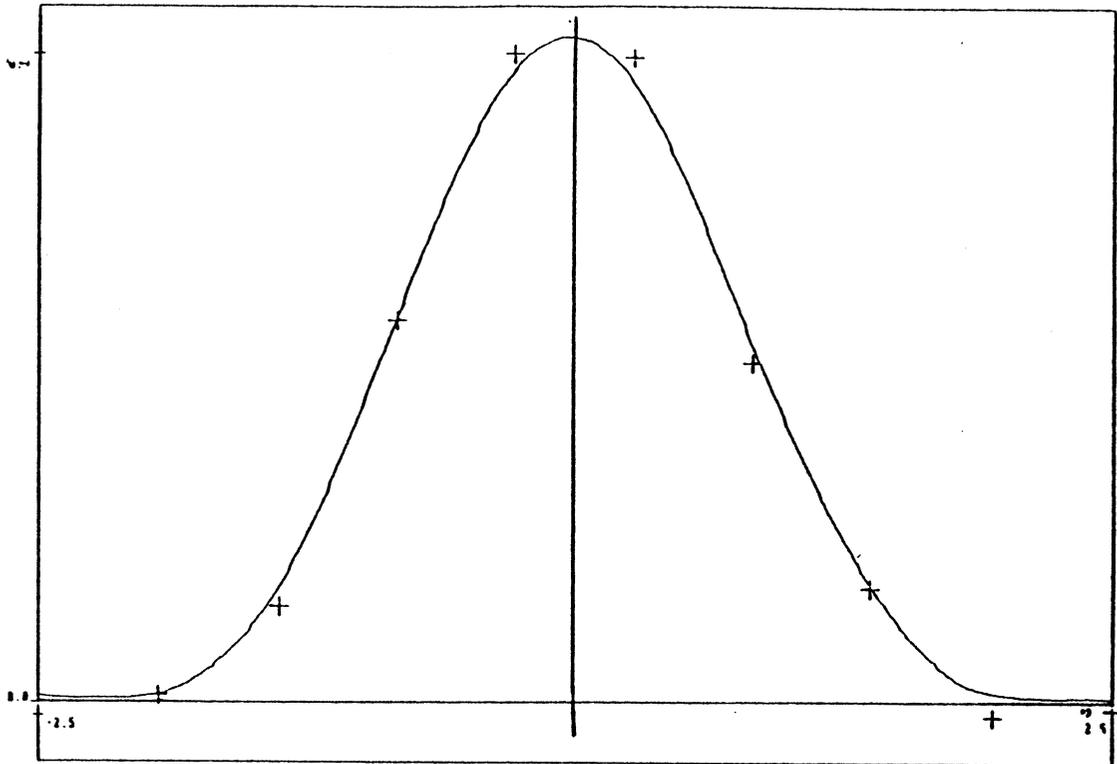
On constate encore que la méthode du gradient projeté est plus rapide et donne aussi de meilleurs résultats graphiques (voir Figs.3,4,5).

Nota

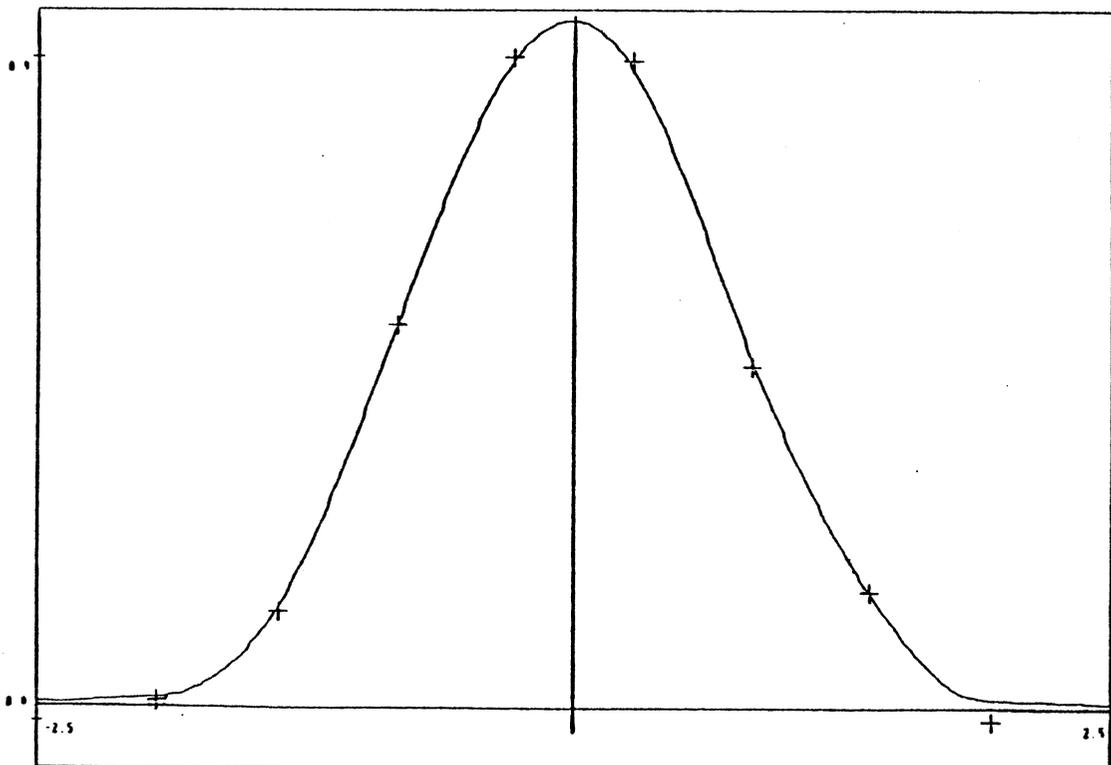
Dans tous les exemples présentés ci-dessous la recherche linéaire a été faite en utilisant à la règle d'Armijo avec $m=2$. Pour cela nous cherchons un intervalle fermé $[\lambda_g, \lambda_d]$ à l'intérieur duquel on est sûr qu'il y a un "intervalle" des points qui vérifient les deux conditions (5.19) et (5.20). Après on procède par dichotomie.

Pour les premières trois ou quatre itérations (dans les exemples 1 et 2) on active entre 15 et 20 fois la procédure de calcul de la fonction spline d'interpolation (problème (P)) ou de lissage (problème (Q)), et après on est tombé à moins de 6 évaluations.

Les règles de Wolfe et de Goldstein et Price, ont besoin à peu près du double de calculs que celle d'Armijo.



(a)

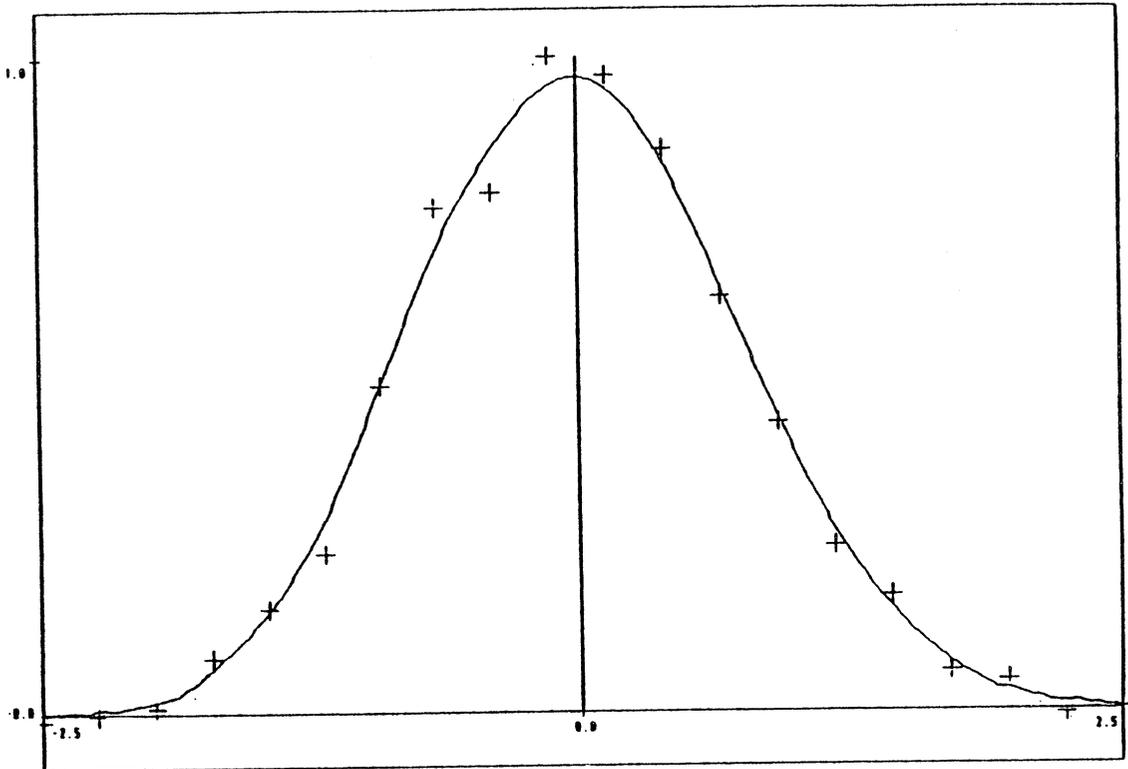


(b)

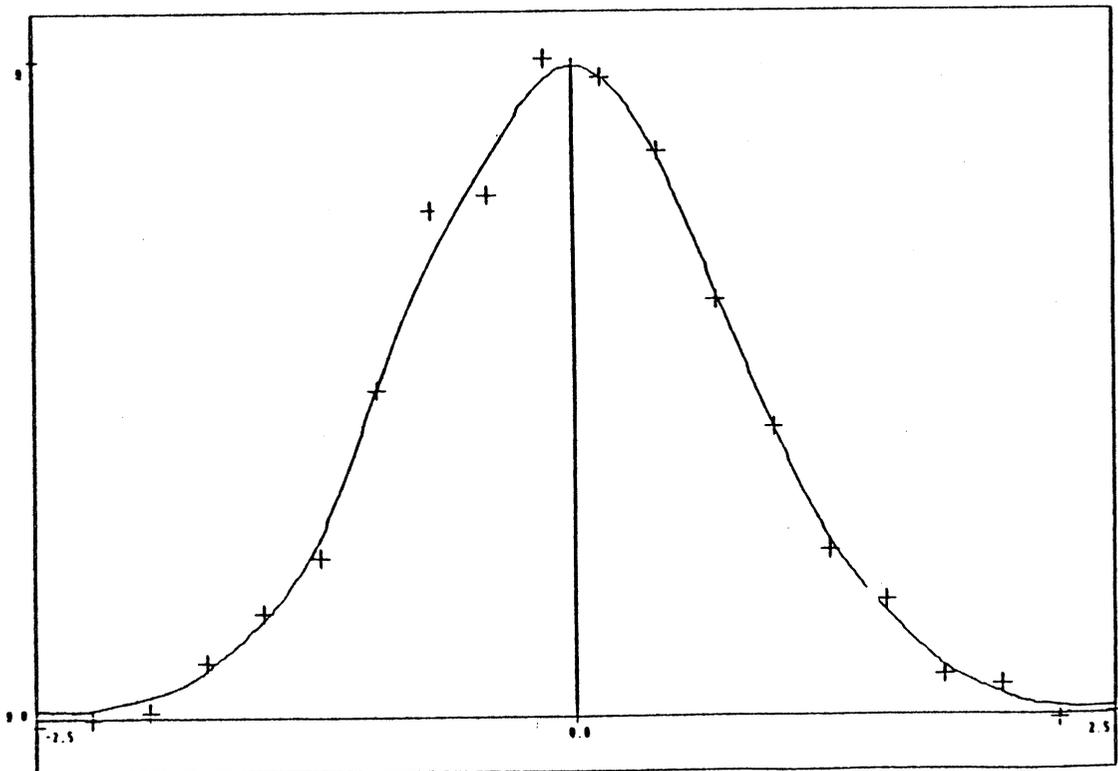
Fig. 3. Exemple 2 ($n=10$). Problème (Q) avec la pénalisation p_1 .

(a) Méthode du gradient ($\rho=150, \eta=40$)

(b) Méthode du gradient projeté ($\rho=150, \eta=100$)



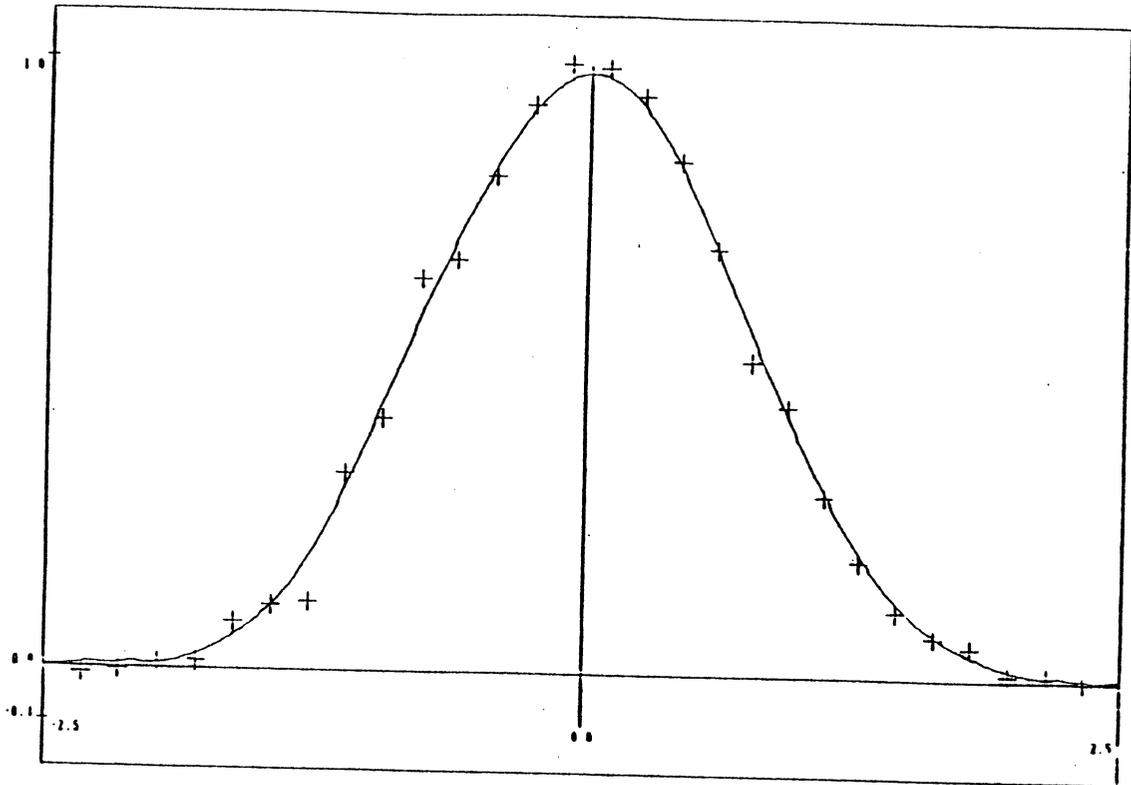
(a)



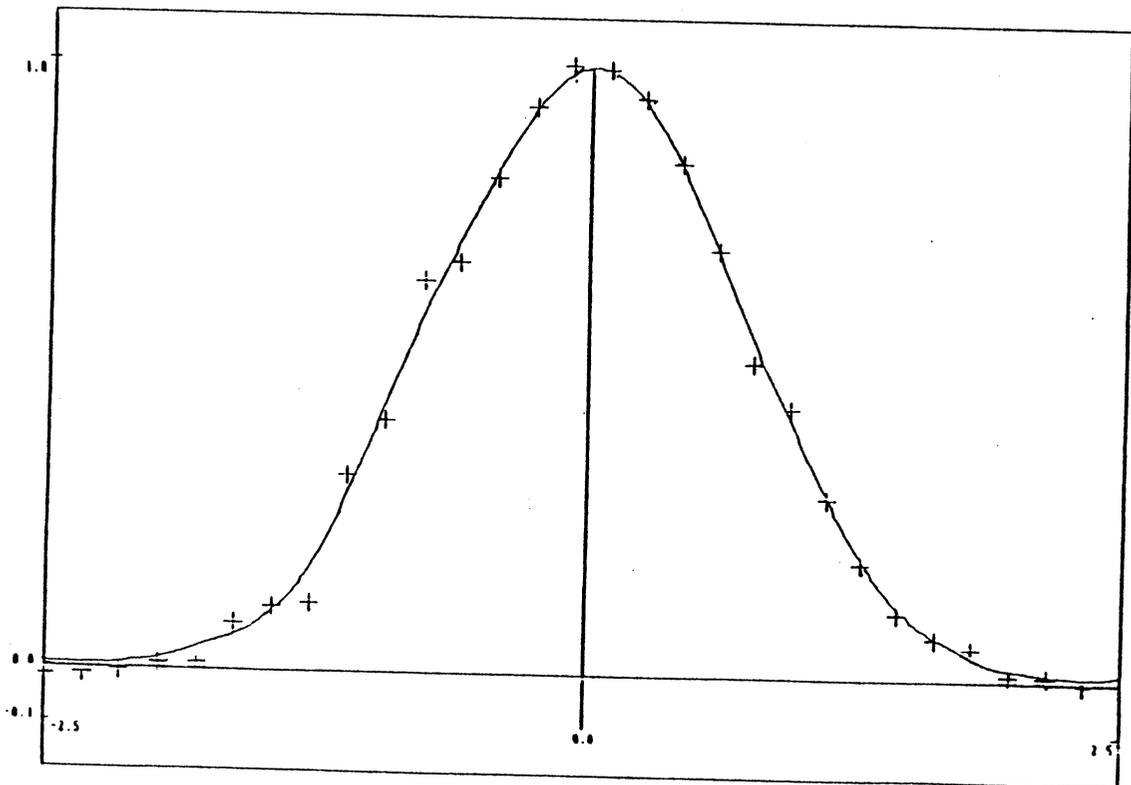
(b)

Fig. 4. Exemple 2 ($n=20$). Problème (Q) avec la pénalisation p_1 .

- (a) Méthode du gradient ($\rho=200, \eta=40$)
(b) Méthode du gradient projeté ($\rho=200, \eta=150$)



(a)



(b)

Fig. 5. Exemple 2 ($n=30$). Problème (Q) avec la pénalisation p_1 .

(a) Méthode du gradient ($\rho=100, \eta=40$)

(b) Méthode du gradient projeté ($\rho=100, \eta=200$)

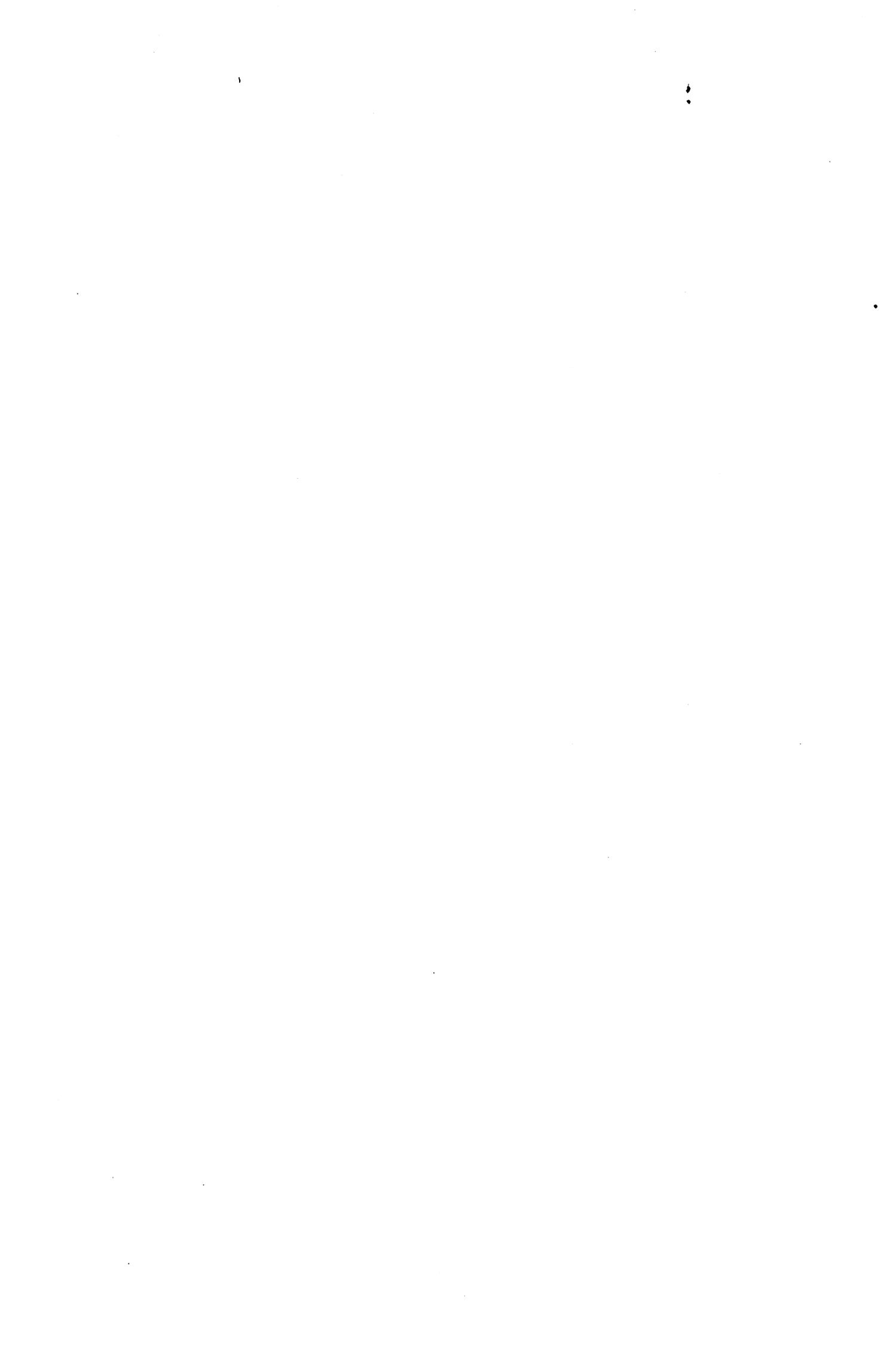
V.6. CONCLUSIONS

Pour construire des fonctions spline de lissage qui respectent la condition de positivité d'une manière stricte, nous avons proposé d'introduire un repoussoir qui empêche la contrainte de devenir active.

Le problème auquel on se ramène, c'est un problème de programmation convexe dans \mathbb{R}^n , plus particulièrement, sur le "pavé" $]0, +\infty[{}^n$. Il semble préférable de résoudre le problème dual. La méthode du gradient projeté donne des résultats satisfaisants. Il faut faire un choix judicieux du terme de pénalisation intérieure; car si l'on emploie la fonction exponentielle on risque de repousser loin la fonction.

Le choix des paramètres ρ (de lissage) et η (de repoussoir) reste un problème ouvert, dont l'inconvénient provient de la nature non-linéaire des opérateurs qu'on minimise.

CONCLUSION GENERALE



L'objectif de ce travail a été d'étudier d'un point de vue algorithmique l'interpolation et le lissage de données par fonctions splines à une variable avec conditions de forme. Nous avons expérimenté un nombre important de méthodes existantes dans la littérature et formulé et proposé de nouveaux algorithmes.

Pour l'interpolation monotone et convexe il y a un bon arsenal de méthodes. Parmi celles à caractère local, les splines rationnelles sont à conseiller.

Le problème du lissage a été traité avec une approche variationnelle. Nous avons pu constater que les splines sous tension et les splines avec repoussoir donnent des résultats satisfaisants. Nous pensons que ces deux idées peuvent se compléter, car la présence de la tension permet d'éliminer des oscillations indésirables (donc de conserver la monotonie, la convexité ou la positivité) et le terme de repoussoir oblige à ce que la condition de forme soit respectée d'une manière stricte. Il nous semble qu'on doit travailler dans cette direction. Un problème délicat qui va se présenter c'est le choix simultané des 3 paramètres : de lissage, de tension et de repoussoir pour que la solution, tout en préservant la forme, ait de bonnes propriétés d'approximation.

Nous considérons aussi qu'une autre direction de recherche importante à développer concerne les fonctions splines à deux dimensions avec contraintes de forme.



REFERENCES

- [1] AKIMA H. A new method of interpolation and smooth curve fitting based on local procedures, *J. Assoc. Comput. Mach.*, 17(1970), pp.589-602.
- [2] AMOS D.E. & SLATER M.L. Polynomial and spline approximation by quadratic programming, *Comm. A.C.M.*, 12 (1969),pp.379-381.
- [3] ARMIJO L. Minimisation of functions having continuous partial derivatives, *Pacific J. Math.*, 16 (1966), pp.1-3.
- [4] BARNHILL R.E. & RIESENFELD R.F. (eds) *Computer Aided Geometric Design*, Academic Press, New York, 1974.
- [5] BARSKY B.A. Exponential and polynomial methods for applying tension to an interpolating spline curve. *Comp. Vis. ,Graph. & Imag. Proc.* 27 (1984), pp.1-18.
- [6] BARSKY B.A. & BEATTY J.C. Local control of bias and tension in beta-splines. *ACM. Trans. on Graph.*, 2 (1983), pp.109-134.
- [7] BAUM A.M. An algebraic approach to simply hyperbolic splines of the real line. *J. Approx. Th.*, 17 (1976),pp.189-199.
- [8] BEATSON R.K. Convex approximation by splines. *SIAM J. Math. Anal.*, 12 (1981), pp.549-559.
- [9] BEATSON R.K. Restricted range approximation by splines and variational inequalities. *SIAM J.Num.Anal.* 19(1982),pp.372-380.
- [10] BEATSON R.K. Monotone and convex approximation by splines : errors estimates and a curve fitting algorithm. *SIAM J. Num. Anal.*, 19 (1982),pp.1278-1285.
- [11] BEATSON R.K. & ZIEGLER Z. Monotonicity preserving surface interpolating, *SIAM J., Num. Anal.*, 22 (1985),pp.401-411.
- [12] BOHM W., FARIN G., KAHMANN J. A survey of curve and surface methodes in CAGD; *Comp. Aid. Geom.Des.*, 1 (1984), pp.1-60.

- [13] BUTLAND J. A method of interpolation reasonable-shaped curves through any data, in *Proc. Computer Graphics 80*. Online Publications, Middlesex, UK, 1980, pp.409-422.
- [14] CARLSON R.E. & FRITSCH F.N. Monotone piecewise bicubic interpolation. *SIAM J. Num. Anal.* , 22 (1985), pp.386-400.
- [15] CHALMERS B.L. & TAYLOR G.D; Uniform approximation with constraints. *Jber.d.Math.Verein.*, 81 (1979) pp.49-86.
- [16] CHUI C.K., SMITH P.W. & WARD J.D. Approximation monotone for splines. *SIAM J., Math. Anal.* , 11 (1980), pp.436-447.
- [17] CHUNG W.L. Automatic curve fitting using an adaptive local algorithm. *ACM Trans. Math. Software*, 6(1980), pp.45-57.
- [18] CIARLET P.G. *Introduction à l'analyse numérique matricielle et à l'optimisation* , Masson, Paris , 1982.
- [19] CINQUIN P. Splines unidimensionnelles sous tension et bidimensionnelles paramétrées : deux applications médicales. Thèse, U. Saint-Etienne, Oct. 1981.
- [20] CLINE A.K. Curve fitting in one and two dimensions using splines under tension. *Comm. ACM*, 17 (1974), pp.213-218.
- [21] COSTANTINI P. & MORANDI R. Monotone and convex cubic spline interpolation (1983) (to appear in *Calcolo*).
- [22] COSTANTINI P. & MORANDI R. An algorithm for computing shape-preseving cubic spline interpolation to data (1983) (to appear in *Calcolo*).
- [23] CRAVEN P. & WAHBA G; Smoothing noisy data with spline functions. Estimating the correct degree of smoothing by the method of generalized cross validation. *Numer. Math.* , 31 (1979), pp.377-403.
- [24] DE BOOR C. *A practical guide to splines*, Springer Verlag, New York, 1978.

- [25] DE BOOR C. & SWARTZ B. Piecewise monotone interpolation , J. Approx. Th., 21 (1977), pp. 411-416.
- [26] DE VORE R. Monotone approximation by splines, SIAM J. Math. Anal. 8 (1977), pp.891-905.
- [27] DE VORE R. Monotone approximation by polynomials, SIAM J. Math. Anal 8 (1977), pp.906-921.
- [28] DELBOURGO R. & GREGORY J. C^2 rational quadratic spline interpolation to monotonic data. IMA J. Numer. Anal. , 3(1983),pp.141-152.
- [29] DIERCKX P. Algorithm 42. An algorithm for cubic spline fitting with convexity constraints, Computing, 24 (1980), pp.349-371.
- [30] DIMSDALE B. Convex cubic splines . IBM J. Res. Develop.,22 (1978), pp.168-178.
- [31] DODD S.L., Mc ALLISTER D.F. & ROULIER J.A. , Shape-preserving spline interpolation for specifying bivariate functions on grids. IEEE CG & A. (1983),pp.70-79.
- [32] DUBE P. Univariate blending functions and alternatives. Comp. Graph. Imag. Proc. ,6 (1977),pp.394-408.
- [33] DUBE P. Preliminary specification of spline curves. IEEE Trans.Comp. C-28 (1979), pp.286-290.
- [34] DUPUY M. Interpolation rationnelle continue par morceaux ayant un caractère de spline. Colloque d'Analyse Numérique , Giens , mai 1978.
- [35] ELLIS T.M.R. & McLAIN D.H. Algorithm 514. A new method of cubic curve fitting using local data. ACM Trans. Math. Software, 3 (1977), pp.175-178.
- [36] EPSTEIN M.P. On the influence of parametrization in parametric interpolation. SIAM J. Num. Anal., 13 (1976), pp.261-268.

- [37] FARIN G. E. Visually C^2 cubic splines , CAD, 14 (1982), pp.137-139.
- [38] FORD W.T. & ROULIER J.A. On interpolation and approximation by polynomials with monotone derivatives, J. Approx. Th., 10 (1974), pp.123-130.
- [39] FORRESTER A.R. Interactive interpolation and approximation by Bezier polynomials. Comput. J. , 15 (1972), pp.71-79.
- [40] FRITSCH F.N. & BUTLAND J. A method for constructing local monotone piecewise cubic interpolants. SIAM J., Sci. Stat. Comp., 5 (1984), pp.300-304.
- [41] FRITSCH F.N. & CARLSON R.E. Monotone piecewise cubic interpolation. SIAM J. Num. Anal., 17 (1980), pp.235-246.
- [42] FROST C.E. & KINZEL G.L. An automatic adjustment procedure for rational splines, Comput. & Graphics, 6 (1982) pp.171-176.
- [43] FU-QUING-XIANG. Optimum cubic spline fitting with desired convexity and its algorithms (chinese). Acta Math. Appl. Sinica , 2(1979), pp.331-339.
- [44] GOLDSTEIN A.A. & PRICE J.F. An effective algorithm for minimization, Numer. Math., 10 (1967), pp.184-189.
- [45] GORDON W.J. & RIENSEFELD R.F. Bernstein-Bezier methods for the computer aided design of free form curves and surfaces, J. Assoc.Comp.Mach., 21 (1974), pp.293-310.
- [46] GORENFLO R. & HILPERT M. On the continuity of convexly constrained interpolation, in *Approximation Theory III* (E. Cheney, Edit.) Academic Press, New York, 1980, pp.449-453.
- [47] GREFORY J. & DELBOURGO R. Piecewise rational quadratic interpolation to monotonic data, IMA J. Num. Anal. , 2 (1982), pp.123-130.
- [48] HARADA K. & NAKAMAE E. Application of the Bezier curve to data interpolation. CAD, 14 (1982), pp.55-60;

- [49] HARADA K. KANEDA K. & NAKAMAE E. A further investigation of segmented Bezier interpolants. CAD , 16(1984)), pp.186-190.
- [50] HORN B.K The curve of least energy ,ACM Trans. Math. Software, 9 (1983),pp.. 441-460.
- [51] HORNING U. Monotone spline interpolation, in *Numerische Methoden der Approximationstheorie 4*(L. Collatz, G. Meinardus and H. Werner , Eds) Birkhäuser, Basel , 1978, pp.. 172-191.
- [52] HORNING U. Interpolation by smooth functions under restrictions on the derivatives, J. Approx. Th., 28 (1980) pp.227-237.
- [53] HUANG DA-REN. A sufficient condition of monotone cubic splines (chinese), Math. Num. Sinica , 4 (1982), pp.214-217.
- [54] INSELBERG A. Cubic splines with infinite derivatives at some knots, IBM J. Res. Develop. , 20 (1976), pp.430-436.
- [55] LAFATA P. & ROSEN J.B;. An interactive display for approximation by linear programming. Comm. ACM, 13 (1970) pp.651-659.
- [56] LAURENT P.J. *Approximation et optimisation* , Hermann, Paris, 1972.
- [57] LAURENT P.J. An algorithm for the computation of spline fonctions under inequallity constraints. Seminaire n°335, Fev. 1980, Université de Grenoble.
- [58] LAURENT P.J. Fonctions spline cardinales tronquées. Rapport de Recherche n° , Oct. 1983 , Université de Grenoble.
- [59] LAURENT P.J. & MARTINET B. Méthodes duales pour le calcul du minimum d'une fonction convexe sur une intersection de convexes. Symposium on Optimization, Nice, June 29-July 5, 1969.
- [60] LAWSON C.L. & HANSON R.J. *Solving Least Squares Problems*. Prentice Hall, New York, 1974.

- [61] LEE E.H. Variational study of non linear spline curves , SIAM Rev. 15 (1973), pp.120-133.
- [62] LEVIATAN D. & MHASKAR H.N. Comonotone approximation by splines of piecewise monotone function, J.Approx. Th., 35 (1982),pp.364-369.
- [63] LEVIATAN D. & MHASKAR H.N. The rate of monotone approximation in L_p -norm, SIAM J.Math. Anal., 13 (1982), pp.866-874.
- [64] MALCOM M.A. On the computation of non-linear spline functions . SIAM J. Num. Anal., 14 (1977), pp.254-282.
- [65] Mc ALLISTER & ROULIER J.A. Interpolation by convex quadratic splines, Math. Comput., 32 (1978) ,pp.1154-1164.
- [66] Mc ALLISTER & ROULIER J.A. An algorithm for computing a shape-preserving osculatory quadratic spline, ACM Trans. Math. Software, 7 (1981),pp. 331-347.
- [67] Mc ALLISTER & ROULIER J.A. Algorithm 574, shape-reserving osculatory quadratic splines, ACM Trans. Maths. Software, 7 (1981),pp.384-286.
- [68] Mc ALLISTER, PASSOW E. & ROULIER J.A. Algorithms for computing shape preserving spline interpolations to data, Math. Comp. 31 (1977) pp.717-725.
- [69] Mc LAUGHLIN H.W. Shape-preserving planar interpolation : an algorithm. IEEE C.G. & A, 21, (1983), pp.58-67.
- [70] MANNING J.R. Continuity conditions for spline curves. Comp.J. 17(1974),pp.181-186.
- [71] MARIN S.P. An approach to data parametrization in parametric cubic spline interpolation problems, J. Approx. Th., 41 (1984),pp.64-86.
- [72] MEDINA J. Fonctions splines d'interpolation monotones, Rapport de DEA, Université de Grenoble, 1982.

- [73] METTKE H. Convex cubic Hermite spline interpolation. *J. Comput. Appl. Math.*, 9 (1983), pp.205-211; Voir aussi une correction dans le même journal, 11 (1984), pp.377-378.
- [74] MINOUX M. *Programmation Mathématique* (tome 1), Dunod, Paris 1983.
- [75] MORIN M. Méthodes de calcul des fonctions splines dans un convexe, thèse, Université de Grenoble, Sept. 1969.
- [76] NAGELKERKE N.J.D. & OOSTING J. The fitting of growth curves using constrained splines, *Comp. Biom. Res.*, 16 (1983), pp.144-148.
- [77] NEUMAN E. Convex interpolation splines of odd degree, *Utilitas Math.*, 14 (1978), pp. 129-140.
- [78] NEUMAN E. Convex interpolating splines of arbitrary degree, in *Numerical Methods of Approximation Theory 5* (L. Collatz, G. Meinardus, H. Werner, Eds), Birkhäuser, Basel, 1980, pp.211-222.
- [79] NEUMAN E. Convex interpolating splines of arbitrary degree II, *BIT*, 22 (1982), pp.331-338.
- [80] NEWMAN D.J. Efficient co-monotone approximation, *J. Approx. Th.*, 25(1979), pp.189-192.
- [81] NIELSON G. Some piecewise polynomial alternatives to splines under tension, in *CAGD* (voir [4]), pp.209-235.
- [82] NIELSON G. & FRANKE R. A method for construction of surfaces under tension, *Rocky Mountain J. Math.*, 14 (1984), pp.203-221.
- [83] PAGALLO G. & PEREIRA V. Smooth monotone spline interpolation, in *Lectures Notes in Mathematics n°909*, pp. 142-145.
- [84] PASSOY E. Piecewise monotone spline interpolation, *J. Approx. Th.*, 12(1974), pp.240-241.

- [85] PASSOW E. Monotone quadratic spline interpolation, J. Approx. Th., 19(1977),pp.143-147.
- [86] PASSOW E. & ROULIER J.A. Monotone and convex spline interpolation,SIAM J. Numer. Anal., 14 (1977) pp.904-909.
- [87] PAIHUA L. Quelques méthodes numériques pour le calcul de fonctions spline à une et plusieurs variables, Thèse, Université de Grenoble, mai 1978.
- [88] PENCE D. Hermite-Birkhoff interpolation and monotone approximations by splines, J. Approx. Th. ,25 (1979), pp.248-257.
- [89] PENCE D. Best mean approximation by splines satisfying generalized convexity constraints, J. Approx. Th., 28 (1980), pp.333-348.
- [90] PRENTER P. Piecewise L-splines Numer. Math. 18 (1971),pp.243-253.
- [91] PRUESS S. Proprieties of splines in tension, J. Approx. Th.,17, (1976), p86-96.
- [92] PRUESS S. An algorithm for computing smoothing splines in tension, Computing,19 (1978) pp.365-373.
- [93] PRUESS S. Alternatives to the exponential spline in tension, Math. Comp., 33 (1979),pp.1273-1281.
- [94] RABINOWITZ P. Applications of linear programming to numerical analysis, SIAM Rev., 10 (1968),pp.121-159.
- [95] REINSCH C. Smoothing by spline functions I, Numer. Math.,10 (1967) pp.177-184.
- [96] REINSCH C. Smoothing by spline functions II, Numer. Math., 16(1971),pp.451-454.
- [97] RENTROP P. An algorithm for the computation of the exponential spline, Numer.Math., 35(1980),pp.81-93.

- [98] ROULIER J.A. Constrained interpolation, SIAM J. Sci. Stat. Comput., 1(1980),pp.333-334.
- [99] SALKAUKAS K. C^1 splines for interpolation of rapidly varying data, Rocky Mountain J.Math., 14(1984),pp.239-250.
- [100] SCHABAK R. Spezielle rationale spline funktionent, J. Approx. Th., 7(1973) , pp.281-292.
- [101] SCHULTZ M. & VARGA R. L-splines, Numer.Math., 10(1967), pp.319-345.
- [102] SCHUMAKER L.L. *Splines Functions : Basic Theory* , Wiley-Interscience, New-York, 1981.
- [103] SCHUMAKER L.L. On shape preserving quadratic spline interpolation, SIAM J. Numer. Anal., 20(1983),pp.854-864.
- [104] SCHWEIKERT D. An interpolation curve using splines in tension, J. Math. Phys. , 45 (1966),pp.312-317.
- [105] SCOTT D.S. The complexity of interpolating given data in three spaces with a convex function on two variables, J. Approx. Th., 42 (1984),pp.42-63.
- [106] SPATH H. Exponential spline interpolation, Computing , 4 (1969), pp.225-233.
- [107] SPATH H. *Spline Algorithm for Curves and Surfaces*, Utilitas Mathematica Publ., Winnipeg, 1974.
- [108] UTRERAS F. Utilisation de la méthode de validation croisée pour le lissage par fonctions splines à une ou deux variables, Thèse, Université de Grenoble, Mai 1979.
- [109] UTRERAS F. Convergence rates for monotone cubic spline interpolation, J. Approx. Th., 36 (1982),pp.86-90.
- [110] UTRERAS F. Smoothing noisy data under monotonicity constraints : existence, characterization and convergence rates, Informe Técnico N°MA.83-B283, 1983, U. de Chile.

- [111] UTRERAS F. & CELIS V. Piecewise cubic monotone interpolation : a variational approach. Informe Técnico N° MA-83-B.281, 1983, U. de Chile.
- [112] VARAH. J. On the solution of block-tridiagonal systems arising from certain finite difference equations, Math. Comp. 26 (1972),pp.859-868.
- [113] WAHBA G. Smoothing noisy data with spline functions, Numer. Math. , 24 (1975), pp.383-393.
- [114] WAHBA G. Practical approximate solution to linear operator equation when the data are noisy, SIAM J. Numer. Anal., 14 (1977), pp.. 651-667.
- [115] WANG J.Y. A spline interpolation with convexity (chinese) , Math. Num. Sinica, 1 (1979),pp.233-243.
- [116] WEN TAO. A keep convex interpolation for monotonic smooth function (chinese), Math. Num. Sinica, 2(1980), pp.299-306.
- [117] WEN TAO. On spline interpolation for monotone smooth fuction (chinese), Math. Num. Sinica, 3(1981),pp.143-151.
- [118] WOLFE P. Convergence conditions for ascent methodes, SIAM Review, 11 (1969), pp.226-235.
- [119] WONG H.W. On constrained multivariate splines and their approximations, Numer. Math. , 43 (1984),pp.141-152.
- [120] WRIGHT I.W. & WEGMAN E.J., Isotonic, convex and related splines, Ann. Stat., 8(1980),pp.1023-1035.

AUTORISATION DE SOUTENANCE

DOCTORAT 3ème CYCLE, DOCTORAT-INGENIEUR, DOCTORAT USMG

Vu les dispositions de l'arrêté du 16 avril 1974,

Vu les dispositions de l'arrêté du 5 juillet 1984,

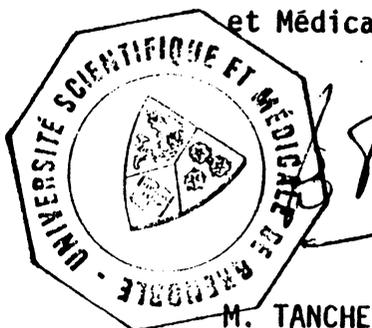
Vu les rapports de M.^r **C. CARASSO**.....

M.^r **F. ROBERT**.....

M.^r **Julio MÉDINA**..... est autorisé
à présenter une thèse en vue de l'obtention du **DOCTORAT USMG** en.....
Mathématiques Appliquées.....

Grenoble, le **20 SEP. 1985**.....

Le Président de l'Université Scientifique
et Médicale



M. TANCHE

Tanche



RESUME

Nous étudions le problème de l'interpolation et du lissage de données avec conditions de forme (positivité, monotonie, convexité) dans le plan réel.

Pour sa résolution cinq classes de méthodes sont utilisées:

- i) splines polynomiales
- ii) splines rationnelles
- iii) splines sous tension
- iv) programmation mathématique
- v) splines avec repoussoir

Mots-clés : interpolation, lissage, approximation, contraintes de forme, fonctions splines (polynomiales, rationnelles, sous tension, avec repoussoir)

