MAX-PLANCK-INSTITUT FÜR PLASMAPHYSIK

GARCHING BEI MÜNCHEN

DIE VERWENDUNG VON KUBISCHEN INTERPOLATIONS- UND AUSGLEICHSSPLINES ZUR MESSDATENVERARBEITUNG

J. Hellauer

IPP R-41 Mai 1983

Die nachstehende Arbeit wurde im Rahmen des Vertrages zwischen dem Max-Planck-Institut für Plasmaphysik und der Europäischen Atomgemeinschaft über die Zusammenarbeit auf dem Gebiete der Plasmaphysik durchgeführt.

Abstract

An extensive domain of numerical analysis is engaged on problems of approximating a real function f on an interval $I \subset \mathbb{R}$ by another function g, which approximates f as exactly as possible and is simpler to handle than f. Familiar methods such as numerical integration and differentiation or the solution analysis of operator equations, e. g. integral and differential equations, based on the approximation by suitable functions mostly polynomials.

This paper deals with a branch of approximation theory, viz. the problem of Data Fitting and Smoothing. In many cases only the ordinate values $y_i := f(x_i)$ in the abscissae x_i, $i := 1,\ldots,n$, of an actual or fictitious function f are available. Assuming that f exists we make accessible some of the algorithms of approximation theory. Nevertheless, we are very restricted with respect to its theoretical instruments. Vice versa almost every method of Fitting and Smoothing is applicable for approximating a given continous function $f: I \dashrightarrow \mathbb{R}$.

Section 1 intended as an introduction to the essential concepts of approximation calculus is followed by a presentation of the well-known Cubic Spline Interpolation. Concrete examples of section 4 make evident the uselessness of interpolation methods for Data Smoothing problems on the whole. Section 3 deals with the applicability of Cubic Smoothing Splines in experiment data processing. Its success or failure depends essentially on the collection of input smoothing parameters. Selection principles and algorithms for suitable determination of these parameters are also treated in this section. These are based on statistical calculus.

Zusammenfassung

Zahlreiche Probleme der Numerischen Mathematik führen auf die Approximation einer auf einem Intervall $I \subset \mathbb{R}$ definierten Funktion $f: I \rightarrow \mathbb{R}$ durch eine leichter zu handhabende Funktion g, die f möglichst "gut" approximiert. Man denke hierbei an so bekannte Verfahren wie die Numerische Integration, Differentation, das Lösen von Operatorgleichungen, wie Integral- und Differentialgleichungen, die fast ausnahmslos auf der Approximation von Funktionen, meistens auf der Interpolation durch Polynome, basieren.

Im Rahmen dieser Arbeit werden wir uns mit einem Teilgebiet der Approximationstheorie beschäftigen, mit dem Fitten und Glätten von Messdaten. In diesen Fällen hat man von der zu approximierenden Funktion f nur die Funktionswerte $y_i := f(x_i)$, $i := 1,..n$, zur Verfügung. Wir sprechen in diesem Zusammenhang von einem "Datenfeld(X , Y)", das durch die Auswertung einer, eventuell nur fiktiv existierenden, Funktion f entstanden ist bzw. entstanden sein könnte. Mit dieser Annahme über f erschliessen wir zumindest einen Teil der Lösungsverfahren der Approximationstheorie. Deren theoretisches Werkzeug zur Bestimmung von Fehlerschranken und Konvergenzaussagen steht uns jedoch nur eingeschränkt zur Verfügung. Umgekehrt lassen sich fast alle Verfahren zum Fitten und Glätten von Messdaten zur gewöhnlichen Interpolation oder Approximation von stetigen Funktionen $f: I \rightarrow \mathbb{R}$ anwenden.

Nach Kapitel 1, gedacht als kleine Einführung in die wichtigsten Begriffe der Approximationstheorie, werden im 2. Kapitel die bekannten kubischen Spline-Interpolationsmethoden vorgestellt. Anhand von Beispielen des Kapitels 4 wird aufgezeigt, dass im allgemeinen Interpolationsmethoden zum Glätten von Messdaten ungeeignet sind. Kapitel 3 befasst sich mit der Anwendung von kubischen Ausgleichssplines bei der Messdatenauswertung. Ob mit Erfolg oder Misserfolg hängt hier wesentlich von der Auswahl der einzugebenden Glättungsparameter ab. Auf statistischen Methoden aufbauende Auswahlkriterien und Algorithmen zur Bestimmung geeigneter Eingabeparameter sind ebenfalls Inhalt dieses Kapitels.

INHALT

1. Einführung

1.1 Beispiele einfacher Approximationsverfahren

Eine bewährte Vorgehensweise zur Bestimmung einer möglichst guten Approximierenden g ist eine formale Darstellung als Summe von Ansatzfunktionen g_i, die auf dem Intervall I definiert sind,

$$g(x) := \sum_{i=1}^{k} c_i * g_i(x)$$

Die g_i sind frei gewählte Elemente eines geeigneten, linearen und reellen Funktionenraumes G. Werden die Elemente $g_1,\ldots,g_k$ als linear unabhängig gewählt, so spannen diese einen linearen Teilraum G_k der Dimension k auf, dessen sämtlichen Elemente die obige Darstellung mit eindeutig bestimmten Koeffizienten c_i besitzen. Im Folgendem benutzen wir die Schreibweise

$$G_k := \text{span}(g_1,\ldots, g_k) \subseteq G \quad .$$

Zur Bestimmung einer konkreten approximierenden Funktion g an f müssen die unbekannten Koeffizienten (c_i), i = 1,..., k, berechnet werden, die man als Lösungen linearer oder nichtlinearer Gleichungssysteme erhält.

Obige Vorgehensweise lässt sich gut an den 3, wohl bekanntesten, Approximationsverfahren verfolgen.

1.1.1 Polynominterpolation

Gegeben:

Datenfeld (X,Y) mit
Stützstellen X, $a =: x_1 < x_2 < \ldots x_n := b$ und
Ordinaten Y, $y_i := f(x_i)$, mit $i := 1,\ldots,n$.

Approximationsvorgabe:

Bestimmung eines Polynoms g auf I mit der Interpolationsvorgabe:

$$g(x_i) = y_i \text{ , fur alle } i = 1,\ldots,n \qquad (1.1.1.1)$$

Ansatz für g:

$$g(x) := \sum_{i=1}^{n} c_i * l_i(x) \qquad (1.1.1.2)$$

mit

$$l_j(x) := \prod_{\substack{k=1 \\ k \neq j}}^{n} \frac{x - x_k}{x_j - x_k} \qquad (1.1.1.3)$$

Die "Lagrange-Faktoren" l_i sind Polynome vom Grad n-1 auf [a,b] und erfüllen die Bedingung

$$l_j(x_i) = \delta_{ij} := \begin{cases} 1, & \text{für } i = j \\ 0, & \text{für } i \neq j. \end{cases}$$

Fur die Koeffizienten c_i, gilt:

$$c_i = y_i \text{ , für } i := 1,\ldots,n.$$

Die c_i werden somit ohne jegliche Berechnung bestimmt. Die explizite Darstellung der Ansatzfunktionen (d.h. die Bestimmung der Polynomkoeffizienten der Ansatzfunktionen l_i) ist dagegen umso komplizierter und umfangreicher. Für

praktische Anwendungen empfiehlt sich daher die gewöhnliche Polynominterpolation nur bis zum Höchstgrad 5. An einem konkreten Beispiel wird später gezeigt, dass die gleichmässige Konvergenz von g an f, d.h. Konvergenz bezüglich der Norm (1.2.4.1), bei beliebiger Erhöhung von n nicht gesichert ist.

#

1.1.2 Lineare Regression

Gegeben: $a =: x_1 < x_2 < \ldots x_n = b$ und
$y = f(x_i),\ i = 1,\ldots,n.$

Approximationsvorgabe:

$$\sum_{i=1}^{n} (g(x_i) - y_i)^{**}2 \quad \leqq \quad \sum_{i=1}^{n} (h(x_i) - y_i)^{**}2$$

fur alle $g, h \in \text{span}(g_1, g_2) := \text{span}(1, x)$.

Dieses ursprünglich in der Statistik angewandte Näherungsverfahren entspricht der Minimierung der Summe der Fehlerquadrate $(g(x_i) - y_i)^{**}2$ mit den Ansatzfunktionen $g_1(x) := 1$ und $g_2(x) := x$. Die Lineare Regression ist somit ein Vertreter der sogenannten "Least Squares"-Verfahren (1.2.8).

Die zur Aufstellung von g benötigten Koeffizienten c_1 und c_2 erhält man als Lösung der sogenannten Normalgleichungen (1.2.8.3). Die explizite Darstellung lautet hier:

$$\bar{x} := 1/n * \sum_{i=1}^{n} x_i , \qquad \bar{y} := 1/n * \sum_{i=1}^{n} y_i$$

$$c_2 = \frac{\sum_{i=1}^{n} y_i * x_i \; - \; n * \bar{x} * \bar{y}}{\sum_{i=1}^{n} x_i * x_i \; - \; n * \bar{x} * \bar{x}}$$

$$c_1 = \bar{y} - c_2 * \bar{x}$$

$$g(x) = c_1 + c_2 * x, \qquad x \in I := [a,b]$$

1.1.3 Fourierapproximation von periodischen Funktionen

Gegeben:

f als stetige, periodische Funktion auf dem Intervall $[-\pi, \pi]$ mit $f(x) = f(x+\pi)$

Ansatz :

$$g(x) := \sum_{i=0}^{n} c_i * g_i(x)$$

mit

$$g_0(x) := 1/\pi$$

$$g_{2i}(x) := 1/\pi * \cos(i*x)$$

$$g_{2i+1}(x) := 1/\pi * \sin(i*x)$$

Die Bestimmung der Koeffizienten c_i ist bei der "Least Squares"-Approximation explizit möglich, da die Ansatzfunktionen g ein Orthonormalsystem bezüglich des Skalarproduktes

$$\langle f , g \rangle := \int_{-\pi}^{\pi} f(x)*g(x)dx \qquad (1.1.3.1)$$

bilden.

Eine allgemeinere Darstellung dieser Approximationsmethode wird im nächsten Paragraphen behandelt.

1.2 Einige Begriffe und Definitionen

In diesem Paragraphen werden wir einige, im allgemeinen wohlbekannte, Begriffe und Bezeichnungen kennenlernen, die für eine mathematische Untersuchung unseres Themas unerlässlich sind, zunächst der Begriff Norm.

1.2.1. Definition

Ein reeller Vektorraum V heisst normiert, wenn es eine Abbildung

$$\|\cdot\| : V \longrightarrow \mathbb{R}$$

mit den Eigenschaften

$$\|f\| = 0 \iff f = 0, \tag{1.2.1.1}$$

$$\|r * f\| = |r| * \|f\|, \ \forall r \in \mathbb{R}, \forall f \in V, \tag{1.2.1.2}$$

$$\|f + g\| \leq \|f\| + \|g\|, \ \forall f, g \in V \tag{1.2.1.3}$$

gibt. Die so definierte Abbildung $\|.\|$ heisst "Norm" auf dem Vektorraum V.

#

Mit Hilfe einer geeignet erklärten "Metrik" lässt sich auf einem Vektorrraum V die Güte einer Näherungsfunktion g an $f \in V$ bewerten und vergleichen. Eine Metrik "dist" ist durch folgende Eigenschaften charakterisiert.

1.2.2 Definition

Eine Abbildung

$$\text{dist}: V \times V \longrightarrow \mathbb{R}$$

heisst "Metrik auf V", wenn sie folgende, von der Geometrie abgeleiteten, Eigenschaften erfüllt:

$$\text{dist}(f,g) = 0 \iff f = g \tag{1.2.2.1}$$

$$\text{dist}(f,g) = \text{dist}(g,f) \tag{1.2.2.2}$$

$$\text{dist}(f,g) \leq \text{dist}(f,h) + \text{dist}(h,g) \tag{1.2.2.3}$$

#

Auf einem Vektorraum können im allgemeinen die unterschiedlichsten Metriken festlegt werden. In normierten Vektorräumen lässt sich immer eine von der Norm abgeleitete Metrik einführen, die spezielle Eigenschaften erfüllt.

1.2.3 Satz

Sei V ein normierter Vektorraum mit der Norm $\|.\|$. Dann definiert die Abbildung

$$\text{dist}: V \times V \longrightarrow \mathbb{R}$$

$$\text{dist}(f,g) := \|f - g\|$$

eine Metrik auf V.

#

Die obige Aussage folgt sofort aus den Eigenschaften (1.2.2) der Norm.

Im folgenden werden Beispiele von normierten Vektorräumen angeführt.

1.2.4 Beispiele von Normierten Räumen:

Tschebyschev-Norm im $\mathbb{R}^n$: (1.2.4.1)

$x := (x_1, \ldots, x_n)$, $x \in \mathbb{R}^n$

$$\|x\|_\infty := \max_{1 \le k \le n} |x_k|$$

Euklidische Norm im $\mathbb{R}^n$: (1.2.4.2)

$$\|x\|_2 := \left(\sum_{i=1}^{n} x_i^2 \right)^{1/2}$$

---

l_p-Norm im $\mathbb{R}^n$, $1 \le p < \infty$: (1.2.4.3)

$$\|x\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

Für unsere Zwecke benötigen wir hauptsächlich Normen auf Funktionenräumen, deren Bezeichnungen wir folgendermassen festlegen.

$C^m[a,b]$:= Vektorraum der im Intervall $[a,b]$ stetigen und in (a,b) m-mal stetig differenzierbaren Funktionen, $m = 0,\ldots$.

$C[a,b]$:= Vektorraum der auf $[a,b]$ stetigen Funktionen

$C^{-1}[a,b]$:= Vektorraum der auf $[a,b]$ stückweise stetigen Funktionen

$$L_p[a,b] := \left\{ f: a,b \dashrightarrow R \;/\; \left(\int_a^b |f(x)|^p dx \right) < \infty \;,\; 1 \leq p < \infty \right\}$$

:= Vektorraum der (im Sinne von Lebesque) p-integrierbaren Funktionen auf $[a,b]$.

Tschebyscheff-Norm im $C[a,b]$.

$$\| f \|_\infty := \sup_{x \in [a,b]} |f(x)| \qquad (1.2.4.4)$$

L_p-Räume werden für ein p, $1 \leq p < \infty$, im Hinblick auf die folgenden L_p-Normen definiert:

L_p-Norm in L_p-Räumen.

$$\| f \|_p := \left(\int_a^b |f(x)|^p \right)^{1/p} \qquad (1.2.4.5)$$

#

Die oben definierten L_p-Räume haben fuer $p := \infty$ und $p := 2$ zusätzliche spezielle Eigenschaften. Fur $p := \infty$ erhält man eine Verallgemeinerung der Tschebyscheff-Norm auf nicht-notwendig stetige Funktionen aus $L_\infty[a,b]$, die allerdings im Rahmen dieser Arbeit nicht benötigt wird. Der Leser findet in [WLO] eine ziemlich umfassende Zusammenstellung der Topologischen und Funktionalanalytischen Eigenschaften der L_p-Räume.

L_2-Räume V haben eine für die Approximationstheorie wichtige zusätzliche Eigenschaft, sie bilden einen sogenannten Prähilbertraum, deren Norm über ein Skalarprodukt definiert werden kann. Hier ist die Existenz und Eindeutigkeit einer optimalen Approximation von f aus V durch ein $g \in \text{span}(g_1,\ldots,g_n) \subset V$ gesichert. Die Berechnung von g erfolgt ebenfalls mit Hilfe des zugrundeliegenden Skalarproduktes $\langle \cdot,\cdot \rangle$, das folgende Bedingungen erfüllen muss.

1.2.5 Definition:

Eine Abbildung $\langle \cdot,\cdot \rangle : V \times V \longrightarrow \mathbb{R}$ auf dem reellen Vektorraum V mit den Eigenschaften

$$\langle f,f \rangle \geq 0 \qquad (1.2.5.1)$$

$$\langle f,f \rangle = 0 \iff f = 0 \qquad (1.2.5.2)$$

$$\langle f,g \rangle = \langle g,f \rangle \qquad (1.2.5.3)$$

$$\langle r*f,g \rangle = r*\langle f,g \rangle, \quad \bigwedge r \in \mathbb{R} \qquad (1.2.5.4)$$

$$\langle f+h,g \rangle = \langle f,g \rangle + \langle h,g \rangle \qquad (1.2.5.5)$$

heisst Skalarprodukt auf V.

#

1.2.6 Definition:

Ein linearer Raum V mit der Norm

$$\| f \|_2 := \langle f,f \rangle^{1/2}$$

heisst Prähilbertraum.

#

1.2.7 Beispiele für Prähilberträume:

$\mathbb{R}^n$ mit Vektorskalarprodukt

$X := (x_1,\ldots,x_n), \quad Y := (y_1,\ldots,y_n)$

$$\langle X,Y \rangle := X * Y := \sum_{i=1}^{n} x_i * y_i \qquad (1.2.7.1)$$

$V := C[a,b]$ als Teilraum des $L_\infty[a,b]$

$$< f,g > := \int_a^b f(x)*g(x)dx, \quad \wedge f, g \in V \qquad (1.2.7.2)$$

#

Mit den obigen Begriffen lassen sich in Hilberträumen Approximationsalgorithmen, welche unter den Namen "Least-Squares" oder "Methode der kleinsten Quadrate" bekannt wurde, allgemein und elegant formulieren.

1.2.8 Existenz- und Eindeutigkeitssatz

Sei V ein nach (1.2.6) definierter reeller Prähilbertraum und $g_1,\ldots,g_n$ eine linear unabhängige Menge von n Ansatzfunktionen aus V.

Dann gelten bezüglich der "besten" Approximation von Elementen f aus V folgende Existenz- und Eindeutigkeitsaussagen:

a) Fur jedes $f \in V$ gibt es genau ein g, $g \in G_n := \mathrm{span}(g_1,\ldots,g_n)$, mit der Eigenschaft

$$\| f - g \|_2 \leq \| f - h \|_2, \quad \wedge h \in G_n \qquad (1.2.8.1)$$

b) Die Koeffizienten c_i, i:= 1,..,n der besten Lösung g,

$$g(x) = \sum_{i=1}^{n} c_i * g_i(x), \qquad (1.2.8.2)$$

erfüllen für k:= 1,..,n die Normalgleichungen

$$\sum_{i=1}^{n} c_i * < g_i, g_K > = < f, g_K >. \qquad (1.2.8.3)$$

c) Die Lösung g ist orthogonal zum Residuum, d.h.

$$< g - f , g > = 0 \qquad (1.2.8.4)$$

#

Der Beweis zu (1.2.8) kann z.B. in [LIN] nachgelesen werden. Entscheidend für die Optimalität von g ist die Eigenschaft (1.2.8.1). Hieraus erklärt sich auch die Bezeichung "Normalgleichung", die nach (1.2.8.4) aufgestellt werden. Unter den gegebenen Voraussetzungen ist das Gleichungssystem (1.2.8.3) eindeutig lösbar, woraus wiederum die Aussage (1.2.8.1) folgt.

1.2.9 Orthogonalsysteme

Bilden die Ansatzfunktionen $g_1, \ldots, g_n$ ein sogenanntes Orthogonalsystem, d.h.

$$< g_i , g_K > = \delta_{iK} \qquad (1.2.9.1)$$

dann ergibt sich sich für die Gramsche Matrix A,

$$A := (a_{iK}), \quad a_{iK} := < g_i , g_K > = \delta_{iK} , \qquad (1.2.9.2)$$

also die nxn-Einheitsmatrix E_n.

Die Lösung der Normalgleichungen erhält man hier, wie aus (1.2.8.3) ersichtlich, sofort zu

$$c_K = < f, g_K >, \qquad k := 1,..,n. \qquad (1.2.9.3)$$

Die Fourierentwicklung und die Approximation mit Legendre-Polynome, auf die auch die Gausschen Quadraturformeln basieren, sind bekannte Anwendungen der Orthogonalreihendarstellung.

2. Polynomsplines

2.1 Polynominterpolation

Im vorherigem Kapitel wurden die Grundlagen der Approximation von Funktionen angesprochen. In diesem Abschnitt werden einige konkrete Approximationsmethoden vorgestellt. Ausgangspunkt bleibt die Darstellung von g als Summe stückweiser definierter Polynome g_i als Ansatzfunktionen. Die spezielle Wahl der g_i geschieht aus mehreren Gründen.

- Die Berechnung von Funktionswerten $g_i(x_0)$ an beliebigen Stellen $x_0 \in [a,b]$ ist relativ einfach und schnell auf Rechenanlagen durchführbar. Iterationen oder Reihenentwicklungen, wie sie etwa für trigonometrische oder Exponentialfunktionen nötig sind, entfallen, da jeder beliebige Wert $g_i(x_0)$ mit Additionen und Multiplikationen, vom Rundungsfehler abgesehen, exakt darstellbar ist.

- Mathematische Operationen wie Differentation oder Integration lassen sich bei Polynome analytisch durch Koeffizientenmanipulation einfach und schnell ausführen. Dieser Vorteil spielt bei der numerischen Lösung von Operatorgleichungen, wie Differential- und Integralgleichungen, mit approximationstheoretischen Methoden eine grosse Rolle.

- Die Existenz und Eindeutigkeit eines Interpolationspolynoms p_K vom Grad k-1 mit genau k Stützwerten ist gesichert. Die Algorithmen zur Bestimmung von p sind einfach und wohlbekannt.

- Bei der Approximation von genügend oft differenzierbaren Funktionen $f \in C^m[a,b]$, lässt sich mit Hilfe des Taylorschen Satzes ([BÖH], S. 46),

  $$f(x) = f(z)+f'(z)*(x-z)+..+f^{m-1}(z)*(x-z)^{m-1}/(m-1)! + R(f^m)$$

  eine Abschätzung des Approximationsfehlers oder dessen Konvergenzordnung gewinnen.

Wie bereits erwähnt, richtet sich der Grad des Interpolationspolynoms p_n nach der Anzahl n der Elemente des Datenfeldes $X := (x_1,..,x_n)$ und $Y := (y_1,..,y_n)$. Schon bei relativ

wenig Stützstellen, wie etwa bei n:= 11, treten Polynome $g_k(x) := x^{k-1}$ mit dem relativ hohen Grad k:= (n-1), hier k = 10, auf, die sich numerisch extrem ungünstig verhalten. Schon sehr kleine Rundungsfehler erzeugen hier relativ grosse globale Oszillationen des Interpolationspolynoms. Zahlreiche Beispiele dieses Problemkreises findet der Leser in [VIE].

Aber selbst bei theoretisch rundungsfreier Berechnung des Interpolationspolynoms p an eine sogar beliebig oft differenzierbare Funktion f konvergiert p_n bei beliebiger Erhöhung der Stützstellenzahl n nicht gleichmässig gegen f.

Ein bekanntes, aber bei weitem nicht einziges, Beispiel ist die äquidistante Polynom-Interpolation der ansonsten relativ einfach zu approximierenden, beliebig oft differenzierbaren Funktion f,

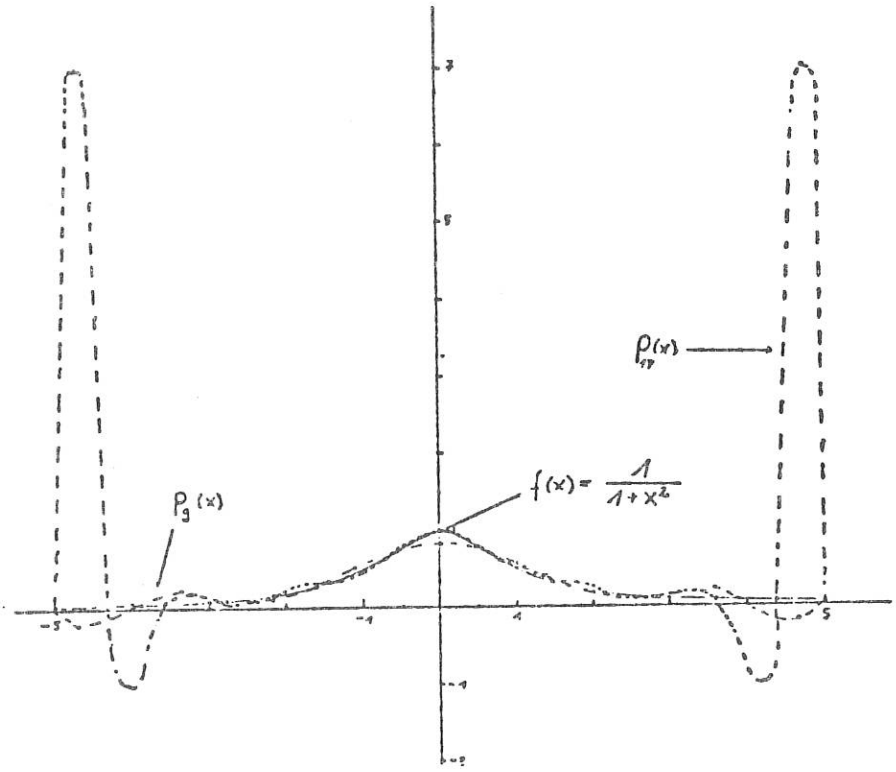
$$f(x) := \frac{1}{1 + x*x} \quad , \quad x \in [-5, 5] , \qquad (2.1.1.1)$$

an den äquidistanten Stützstellen $x_K := -5 + 10*k/n$, k:= 0,..,n, an.

Erhöht man die Stützstellenzahl (n+1) so konvergiert das interpolierende Polynom p_n vom Grad n nicht gleichmässig gegen f, d.h. es gilt,

$$\| f - p \|_\infty := \max \{ |f(x) - p_n(x)| \} \longrightarrow \infty \text{ fur } n \longrightarrow \infty$$

wie in Fig. (2.1.1.2) fur n:=9 und n:= 14 dargestellt wird.



Die Nicht-Konvergenz von g an f ist bei der Polynominterpolation nicht, wie im obigen Beispiel (2.1.1.1), nur auf den Rand des Definitionsbereiches beschränkt, wie man anhand weiterer Beispiele in [BÖH], S. 11, sehen kann. Ebenfalls ist die Wahl von äquidistanten Stützstellen x_K nicht immer ausschlaggebend.

Verzichtet man an den Stützstellen x_K, k:= 1,...,n, auf die Interpolationsvorgabe, $g(x_K) = y_K$, kann man für ein vorgegebenes Datenfeld (X,Y) ein Näherungspolynom g_m an f mit weitaus niedrigerem Polynomgrad (m-1) bestimmen. An die Stelle der Interpolationsbedingung treten dann Optimalitätsbedingungen der Näherung g_m an f bezüglich einer Metrik (1.2.2). Im Falle der optimalen Approximation in der L_2-Norm (1.2.4.2) mit den Polynomen $g_K(x) := x^{k-1}$, k = 1,..,m, $x \in [0,1]$, als Ansatzfunktionen erhält man als Gramsche Matrix $A := (a_{iK})$, i,k=1,..,m,

$$a_{iK} := \langle g_i, g_K \rangle = \int_0^1 g_i(x) * g_K(x)\,dx =$$

$$= \int_0^1 x^{i+k-2}\,dx = 1/(i+k-1), \qquad (2.1.1.2)$$

also die für grosse m überaus schlecht konditionierte m-reihige Hilbertmatrix H, die bei der numerischen Berechnung der Normalgleichungen (1.2.8.3) enorme Genauigkeitsverluste verursacht.

Zu den numerischen Schwierigkeiten bei den obigen globalen Approximationsmethoden, bei der jeweils das gesamte Datenfeld den Näherungswert $g(x_0)$ an einer bestimmten Zwischenstelle $x_0 \in [a,b]$ gleichmässig beeinflusst, treten interpretatorische Probleme bei der Messdatenverarbeitung. Eine kleine Änderung des Messwertes y kann eine wesentliche Veränderung der Näherungslösung g im Punkte x bewirken, die, weil experimentell nicht gegeben, die Plausibilitat der Näherung g abschwächt und damit generell die Brauchbarkeit der Approximationsmethode für dieses spezielle Problem in Frage stellt.

Diesen unter Umständen fatalen Nachteil der globalen Approximation kann man umgehen, indem man das gesamte Datenfeld (X,Y) mit den n Wertepaaren in k kleine Teildatenfelder (X_j, Y_j), j = 1,..,k, mit jeweils m+1 Werten (m<<n) unterteilt und zu diesen nach obigen Verfahren jeweils ein Näherungspolynom p mit Grad m berechnet. Bei der Polynominterpolation mit m:= 2 würde man z.B. n/2 aufeinanderfolgende Parabelbögen erhalten, die zusammen eine zwar stetige aber nicht differenzierbare Gesamtnäherung g ergeben.

Bei der Anwendung der "Laest Squares"-Methode auf diese Teilbereiche erhält man Teilnäherungen p_j, deren Übergänge an den jeweiligen Teilknoten im allgemeinen unstetig bleiben. Offensichtlich liegt in diesen Bereichen eine sehr schlechte Approximation vor. Abgesehen davon dass für viele Anwendungen mehrmals differenzierbare oder zumindest stetige Näherungslösungen verlangt werden, liefern die obigen beiden Methoden im Gegensatz zur globalen Approximation eine im wesentlichen lokal festgelegte Näherung. In den Randpunkten der Teilintervalle wird die Näherungslösung sogar unabhängig von den Nachbarpunkten bestimmt. Die Funktionswerte $g(x_o)$ hängen dagegen für x_o in der Umgebung der Teilknoten sehr von m und der vorgenommen Einteilung der Teilbereiche ab, was, von bestimmten statistischen Verfahren (z.B. Clusteranalyse in [CUF]) abgesehen, ein unerwünschter Nebeneffekt sein dürfte.

Eine Approximationsmethode, welche die Vorteile der lokalen Approximation mit einer geschlossenen Darstellung der Näherung g verbindet und zugleich mehrmals differenzierbare Lösungen liefert, wird im nächsten Paragraphen vorgestellt. Es handelt sich dabei um die Approximation mit Splinefunktionen.

2.2 Kubische Splines

Unter einem Spline s einer reellen Veränderlichen versteht man eine Funktion, die stückweise auf einem Intervall definiert ist und deren Teilstücke an den Nahtstellen hinreichend glatt verheftet sind. Wir befassen uns in dieser Arbeit nur mit Polynomsplines, also solchen Splines, die in jedem Teilstück als Polynome definiert sind.

2.2.1 Definition

Sei $[a,b] \subset \mathbb{R}$ ein reelles Intervall und Z

Z: $a = x_1 < x_2 < .. < x_n = b$ eine Zerlegung von $[a,b]$.

Eine Funktion $s: [a,b] \dashrightarrow \mathbb{R}$ heisst "Spline vom Grad k", wenn gilt:

$s \in C^{k-1}[a,b]$ $\quad (k = 0,1,....)$ $\quad (2.2.1.1)$

$s \in P_k$ fur $x \in [x_i , x_{i+1}]$, $(i = 1,..,n-1)$ $\quad (2.2.1.2)$

Die Teilpunkte x_i $(i = 1,...,n)$ heissen "Knoten der Zerlegung Z".

$S_k(Z)$:= Menge aller Splines vom Grad k zur Zerlegung Z.

//

Man kann nun zeigen, dass $S_k(Z)$ bezüglich der punktweisen Addition und Multiplikation einen reellen Vektorraum über dem Körper $\mathbb{R}$ bildet. Jeder Vektorraum V hat bekanntlich eine Basis, also ein linear unabhängiges Darstellungssystem. Die eindeutig definierte Anzahl dim(V) dieser Basiselemente, die auch unendlich sein kann, wird als Dimension des Vektorraumes V bezeichnet.

Für den Vektorraum $S_K(Z)$ gelten diesbezüglich folgende Aussagen.

2.2.2 Satz

a) $\dim(S_K(Z)) = n + k - 1$ (2.2.2.1)

b) Die Funktionen $p_0, p_1, \ldots, p_K, q_2, \ldots, q_{n-1}$ mit,

$$p_l(x) := x^l \qquad (l = 0,1,\ldots,k)$$

$$q_l(x) := (x - x_l)_+^K := \begin{cases} 0, & \text{für } x < x_l \\ (x - x_l)^K, & \text{für } x \geq x_l \end{cases}$$

bilden eine Basis von $S_K(Z)$. (2.2.2.2)

#

Im Beweis von (2.2.2) [BÖH] wird gezeigt, dass jeder Polynomspline $s \in S_K(Z)$ eine eindeutige, sogenannte "Kardinalspline"-Darstellung

$$s(x) = \sum_{i=0}^{k} a_i * x^i + \sum_{i=2}^{n-1} b_i (x - x_i)_+^i \;, \quad x \in [a,b] \;, \; a, b \in \mathbb{R}, \qquad (2.2.3.1)$$

hat.

Fur numerische Anwendungen spielt die Darstellung mit Kardinalsplines keine grosse Rolle, die Berechnung der Koeffizienten a_i und b_i wäre numerisch zu schlecht konditioniert. Der Einfluss von Rundungsfehlern bei der Bearbeitung auf Rechenanlagen könnte die Genauigkeit empfindlich stören. In der Praxis verwendet man zur Berechnung von s andere Splinebasen, wie B-Splines usw, die sich diesbezüglich günstiger verhalten oder deren Koeffizienten einfacher zu berechnen sind.

Ausgehend von der Definition (2.2.1) und deren Existenzsatz (2.2.2) kann man für eine vorgebene Zerlegung und nichtnegatives k eine Splinefunktion $s \in S_K(Z)$ berechen, die entweder eine gegebene Funktion $f \in C[a,b]$ möglichst gut approximiert oder interpoliert. Neben dem Grad der Differenzierbarkeit beeinflusst die Wahl des Splinegradparameters k davon abhängige Extremaleigenschaften. Auf die allgemeine Erläuterung dieser Zusammenhange möchten wir im Rahmen dieser Arbeit verzichten, sie kann in [BÖH] nachgelesen werden.

Sowohl aus historischen wie auch praktischen Gründen geniesst ein Klasse von Splines eine herausgehobene Stellung. Es handelt sich dabei um den "Kubischen" Spline $s \in S_3(Z)$.

Fur k:= 3 gilt nach (2.2.2), $\dim(S_3(Z)) := n + 2$. Geht man von den üblichen n Interpolationsbedingungen, $s(x_i) = y_i$, i:= 1,...,n, aus, so müssen zur eindeutigen Darstellung des kubischen Splines s noch 2 freie Parameter bestimmt werden.

In diesen 2 zusätzlichen Vorgaben, die zur eindeutigen Bestimmung der Koeffizienten von s notwendig sind, unterscheiden sich die verschiedenen Spline-Interpolationen. Im Prinzip gibt es unendlich viele Möglichkeiten. Von theoretischer wie auch von praktischer Bedeutung sind jedoch nur die 3 folgenden Interpolationsaufgaben, die sich jeweils in ihren zusätzlichen Randvorgaben unterscheiden

2.2.3 Definition

Spline-Interpolation mit Hermite-Endbedingungen (2.2.3.1)

Sei $f \in C^1[a,b]$ und $s \in S_3(Z)$ mit

a) $s(x_i) = f(x_i), \qquad i = 1,...,n$

b) $s'(a) = f'(a)$
$s'(b) = f'(b)$

Natürliche Splines (2.2.3.2)

Sei $f \in C[a,b]$ und $s \in S_3(Z)$ mit

a) $s(x_i) = f(x_i), \qquad i = 1,...,n$

b) $s''(a) = 0$
$s''(b) = 0$

Periodische Splines (2.2.3.3)

Sei $f \in C^2[a,b]$ mit $f(a) = f(b)$ und $f'(a) = f'(b)$ und $s \in S_3(Z)$ mit

a) $s(x_i) = f(x_i), \qquad i = 1,...,n$

b) $s'(a) = s'(b)$
$s''(a) = s''(b)$

#

Fur die interpolierenden Splines, die nach den Vorgaben (2.2.3) bestimmt wurden, gelten nun folgende Existenz-, Eindeutigkeits- und Extremalaussagen, die gute Approximationseigenschaften der kubischen Interpolationssplines vermuten lassen.

2.2.4 Satz

Sei $f \in C^2[a,b]$ und erfüllt $s \in S_3(Z)$ eine der 3 Interpolationsvorgaben (2.2.3.1)-(2.2.3.3) so gelten fur s folgende Aussagen:

a) Die obigen Interpolationsaufgaben sind stets eindeutig lösbar. (2.2.4.1)

b) Die Lösung s erfüllt die Integralrelation

$$\|f''\|_2 = \|f'' - s''\|_2 + \|s''\|_2 \qquad (2.2.4.2)$$

#

Im Beweis zu (2.2.4.1), der in [BÖH] nachzulesen ist, wird gezeigt, dass die Koeffizienten a_i und b_i, die s in der Kardinalspline-Darstellung (2.2.2.1) bestimmen, durch die vorgebenen Interpolations- und Endbedingungen eindeutig bestimmt sind. Der Beweis zu (2.2.4.2) wird durch Anwendung der vollständigen Induktion und partieller Integration auf den Term $\|f'' - s''\|_2$ geführt. Das Auftreten des Gleichheitszeichens "=" in der ansonsten immer gültigen Dreiecksungleichung (1.2.1.3) ist in der L_2-Norm des Prähilbertraumes $C^2[a,b]$ (1.2.7.2) äquivalent zur Aussage $< f - s , s > = 0$, die folgende Extremaleigenschaft von s begründet (man vergleiche dazu (1.2.8)).

2.2.5 Satz

Sei $f \in C^2[a,b]$ und $s \in S_3(Z)$ interpolierender Spline, der eine der Vorgaben (2.2.3.1) - (2.2.3.3) erfüllt.
Dann gilt für jede beliebige, 2-mal stetig differenzierbare, Funktion $h \in C^2[a,b]$, die wie s eine der jeweiligen Interpolationsbedingungen (2.2.3.1) bzw. (2.2.3.2) bzw. (2.2.3.3) erfüllt,

$$\|s''\|_2 < \|h''\|_2$$

#

Die Extremaleigenschaft (2.2.5), sie folgt unmittelbar aus Satz (2.2.4), erklärt die Tatsache, wieso der kubische Spline einen gebogenen Stab, der freilagernd an Stützen festgehalten wird, sehr gut approximiert. Seine Spannungsenergie E im gebogenen Stab ergibt sich für kleine $g' \ll 1$ zu

$$E(g) = \int_a^b \frac{c * (g''(x))^{**2}}{(1 + (g'(x))^{**2})^{**3/2}} dx \approx c * \int_a^b (g''(x))^{**2} dx,$$

Da der gebogene Stab nach dem Lagrange-Prinzip in der Biegekurve g mit der minimalen Spannungsenergie E(g) verharrt und der interpolierende Spline nach (2.2.5) den Näherungsterm $\|s''\|_2$ minimiert, bietet s eine sehr gute Approximation der Biegelinie g. Diese Eigenschaft wird u. a. in der Technik zur näherungsweisen Darstellung von Biegelinien, glatten Oberflächen, Messdatenfitting usw ausgenutzt.

Bevor wir uns den Algorithmen zu Berechnung von kubischen Interpolationssplines zuwenden, wollen wir noch kurz auf die wichtige Konvergenzeigenschaft des interpolierenden Splines eingehen.

Wie wir anhand des Beispiels (2.1.1.1) gesehen haben, ist die Konvergenz in der $\| \cdot \|_\infty$ -Norm bei der gewöhnlichen Interpolationsnäherung p_{n-1} an die wahre Funktion bei beliebiger Erhöhung der Stützstellenzahl n nicht gesichert. Für den kubischen Spline kann eine Reihe von Konvergenzsätzen hergeleitet werden, die bei genügend glatten Funktionen f sogar eine Abschätzung des absoluten Messfehlers und dessen Konvergenzordnung erlauben. Für unsere Zwecke reicht folgende Aussage.

2.2.6 Konvergenzsatz für kubische Interpolationssplines

Sei $f \in C^3[a,b]$ und Z_n eine Folge von Gittern auf $[a,b]$ mit den Zerlegungen Z_n,

$$Z_n : a = x_1 < x_2 < \ldots < x_n = b$$

und

$$Y_n := (f(x_1), \ldots, f(x_n))$$

der zu f und Z_n gehörige Funktionswertevektor.

Sei weiter

d_n := minimale Schrittweite von Z_n,
D_n := maximale Schrittweite von Z_n,

und

$$v_n := D_n / d_n$$

der maximale Quotient der Schrittweiten von Z_n.

Erfüllt die obige Folge von Gittern Z_n die Voraussetzung

$$D_n/d_n < v_n < V < \infty$$

dann gilt für den nach (2.2.3) bestimmten interpolierenden Spline s die Konvergenzaussage:

a) $$\|s - f\|_\infty := \sup_{x \in [a,b]} \{ |s(x) - f(x)| \} \longrightarrow 0$$

wenn $n \longrightarrow \infty$ und $D_n \longrightarrow 0$ konvergiert. (2.2.6.1)

---

b) Ist $f \in C^4[a,b]$, so gilt sogar:

$$\|s' - f'\|_\infty \longrightarrow 0, \quad \text{für } n \longrightarrow \infty$$

(2.2.6.2)

#

Der Beweis zu einem allgemeineren Satz, der die Aussage (2.2.5) einschliesst, kann in [BÖH], S. 26-28, nachgelesen werden.

Die Konvergenz eines interpolierenden Näherungssplines an eine stetige Funktion f ist somit bei einer beliebig kleinen Verfeinerung des Gitters gesichert, wenn die maximale Schrittweite D_n von Z_n gegen null konvergiert und nicht allzu verschieden von der minimalen Schrittweite d_n ist.

Mit diesem, für die Approximationstheorie sehr wichtigen Satz, wollen wir uns den Algorithmen zur Berechnung von Interpolationssplines zuwenden.

2.3 Algorithmen

In Satz (2.2.2) vom letzten Paragraphen wurde gezeigt, dass der Vektorraum S ($\mathbb{Z}_n$) der kubischen Splines auf Z_n eine Basis mit (n+k-1) Elementen besitzt, die aber zur Darstellung von konkreten Splineapproximationen ungeeignet ist.

Wir wollen in diesem Abschnitt ein anderes Darstellungssystem fur den Spline s benutzen. Als vorgegeben betrachten wir wieder ein Datenfeld (X,Y) mit

$$Z_n : a = x_1 < x < \ldots < x_n := b$$

und den dazugehörigen Funktionswerten $y_i := f(x_i)$, $i := 1,\ldots,n$.

Für den interpolierenden Spline s führen wir den Ansatz,

$$s(x) = y_i + c_{i1}*(x-x_i) + c_{i2}*(x-x_i)^2 + c_{i3}*(x-x_i)^3,$$

fur $x_i \leq x < x_{i+1}$ und $i := 1,\ldots,n-1$, ein. (2.3.1.1)

Zur eindeutigen Darstellung von s benötigt man somit die $3*(n-1)$ Koeffizienten c_{ij}, $i := 1,\ldots,n-1$, $j := 1,..,3$.

Ansatz (2.3.1.1) bietet für unsere Zwecke wesentliche Vorteile.

- Das Gleichungssystem zur Berechnung der Splinekoeffizienten c_{ij} hat eine numerisch günstige Tridiagonalstruktur
- Es besteht Konsistenz mit der Darstellung des Ausgleichssplines im folgenden Kapitel 3.
- Funktionswerte und Ableitungen von s lassen sich in beliebigen Zwischenwerten z leicht und genau aus C_{ij} mit dem Hornerschema entwickeln. Fur z , $x_i \leq z < x_{i+1}$ gilt mit $d := z - x_i$:

$$s(z) = ((c_{i3}*d + c_{i2})*d + c_{i1})*d + y_i ,$$

$$s'(z) = (3*c_{i3}*d + 2*c_{i2})*d + c_{i1} ,$$

$$s''(z) = 6*c_{i3}*d + 2*c_{i2} .$$

- Die Funktionswerte von s' und s" in den Stützstellen x

sind direkt aus der Koeffizientenmatrix C ablesbar. Es gilt:

$$s'(x_i) = c_{i,1}, \qquad s''(x_i) = 2 * c_{i,2}.$$

- Die Darstellung (2.3.1.1) von s eignet sich gut zur numerischen Behandlung von Operatorgleichungen, speziell von Integralgleichungen.

Die Koeffizienten $c_{i,j}$ von (2.3.1.1) erhält man als Lösung eines linearen Gleichungssystems, indem man die Stetigkeits- und Differenzierbarkeitsbedingungen (2.2.1) an s berücksichtigt und die jeweiligen Endbedingungen der verschiedenen Interpolationsaufgaben (2.2.3) hinzufügt.

Mit $h_i := x_{i+1} - x_i$ ergeben sich aus den Stetigkeits- und Differenzierbarkeitsbedingungen folgende $3*(n-1) - 2$ linearen Gleichungen an die $c_{i,j}$:

$$s_-(x_{i+1}) = s_+(x_{i+1}) \Longrightarrow h_i^3 * c_{i,3} + h_i^2 * c_{i,2} + h_i * c_{i,1} = y_{i+1} - y_i$$

$$\text{für } i := 1,\ldots,n-1 \qquad (2.3.2.1)$$

$$s'_-(x_{i+1}) = s'_+(x_{i+1}) \Longrightarrow 3*h_i^2 * c_{i,3} + 2*h_i * c_{i,2} + c_{i,1} - c_{i+1,1} = 0$$

$$\text{für } i := 1,\ldots,n-2 \qquad (2.3.2.2)$$

$$s''_-(x_{i+1}) = s''_+(x_{i+1}) \Longrightarrow 3*h_i * c_{i,3} + c_{i,2} - c_{i+1,2} = 0$$

$$\text{für } i := 1,..,n-2 \qquad (2.3.2.3)$$

Aus den Randvorgaben (2.2.3) erhält man die beiden jeweiligen Zusatzgleichungen (2.3.3):

Hermite-Endbedingungen: (2.3.3.1)

$$s'(a) = f'(a) \Longrightarrow c_{1,1} = f'(x_1)$$

$$s'(b) = f'(b) \Longrightarrow c_{n-1,1} + 2*h_{n-1} * c_{n-1,2} + 3*h_{n-1} * c_{n-1,3} = f'(x_n)$$

Natürliche Splines: (2.3.3.2)

$$s''(a) = 0 \Longrightarrow c_{1,2} = 0$$

$$s''(b) = 0 \Longrightarrow 2*c_{n-1,2} + 6*h_{n-1} * c_{n-1,3} = 0$$

Periodische Splines: (2.3.3.3)

$s'(a) = s'(b) \Longrightarrow \quad c_{11} = c_{n-1,1} - 2*h_{n-1} * c_{n-1,2} - 3*h_{n-1}^2 * c_{n-1,3}$

$s''(a) = s''(b) \Longrightarrow \quad c_{12} = c_{n-1,2} - 3*h_{n-1}^2 * c_{n-1,3}$

#

Wie oben erwähnt führt der spezielle Ansatz (2.3.1.1) zu einem besonders einfach zu lösenden Gleichungssystem (2.3.2) mit einer günstigen Bandstruktur. Fügt man diesem System die 2 jeweils noch fehlenden Gleichungen der Randvorgaben in der obigen Form (2.3.3.1) bzw (2.3.3.2) bzw (2.3.3.3) dem Gesamtsystem hinzu, so wird dessen numerische Stabilität sehr negativ beeinflusst. Dieser unliebsame Effekt kann vermieden werden, indem man die beiden jeweiligen Zusatzgleichungen geeignet modifiziert.

Eine Lösung bietet der folgende Ansatz (2.3.4). Er ermöglicht zugleich eine Einbettung verschiedenster Randvorgaben in ein gemeinsames Lösungsschema. Unter anderen lassen sich die Interpolationsaufgaben (2.2.3.1) und (2.2.3.2) mit ein und demselben Verfahren zur Berechnung der Splinekoeffizienten lösen. Zur Lösung der beiden letzteren Interpolationsaufgaben wird im Rahmen dieser Arbeit ein Algorithmus vorgeschlagen, der auf dem Ansatz (2.3.1.1) basiert und folgende modifizierte Gleichungen zur Vorgabe der Randbedingungen benutzt.

$$2 * s''(x_1) + b_1 * s''(x_2) = b_2 \qquad (2.3.4.1)$$

$$b_3 * s''(x_{n-1}) + 2 * s''(x_n) = b_4 \qquad (2.3.4.2)$$

Mit diesen 4 Eingabeparameterm b_1, b_2, b_3 und b_4 lassen sich die Randbedingungen in die jeweilige Splineinterpolationsaufgabe einflechten. Für die betrachteten, speziellen Aufgaben (2.2.3) erhält man im einzelnen folgende Gleichungen für die Parameter b_1, b_2, b_3 und b_4:

Hermite-Endbedingungen (2.2.3.1):

Randvorgabe: $s'(a) = f'(a)$ und $s'(b) = f'(b)$

$b_1 := 1 \quad \Longrightarrow$

$$\begin{aligned} b_2 &= 2 * s''(x_1) + s''(x_2) \\ &= 2 * 2 * c_{12} + (2 * c_{12} + 6 * c_{13} * h_1) \\ &= 6 * c_{12} + 6 * c_{13} * h_1 \end{aligned}$$

Aus (2.3.2.1) bzw (2.3.2.1) folgt mit (2.3.1.1):

$$y_2 = y_1 + c_{11} * h_1 + c_{12} * h_1^2 + c_{13} * h_1^2$$
$$f_1' = c_{11} \quad (= s_1')$$
$$===> \quad y_2 - y_1 - f_1' * h_1 = c_{12} * h_1^2 + c_{13} * h_1^3$$

$$===> \quad b_2 = 6 * (y_2 - y_1 - f_1' * h_1) / h_1^2$$

Analog folgt für b_3 und b_4: (2.3.5.1)

$$b_3 := 1$$
$$b_4 = 6 * (f_n' * h_{n-1}) - (y_n - y_{n-1})) / h_{n-1}^2$$

Natürliche Splines (2.2.3.2):

Randvorgabe: $s''(a) := 0$ und $s''(b) := 0$

$$b_1 := 0 \quad ===> \quad b_2 = 0$$
$$b_3 := 0 \quad ===> \quad b_4 = 0 \qquad (2.3.5.2)$$

#

Mit obigem Ansatz kann man im Prinzip auch die Interpolationsaufgabe mit periodischen Randbedingungen (2.2.3.3), $s'(a) = s'(b)$ und $s''(a) = s''(b)$, formulieren und lösen. Die explizite Berechnung der Parameter b_K führt jedoch auf umständlich lange und komplizierte Terme, die zusätzliche Rundungsfehler bedingen. Aus diesem Grund wurde für diese spezielle Interpolationsaufgabe ein eigener, den periodischen Randbedingungen angepasster, Algorithmus entwickelt.

Mit Hilfe des folgenden, aufgelisteten Fortran-Programmpakets 'CUBSPLIN' kann für ein gegebenes Datenfeld (X,Y) ein interpolierender kubischer Spline s mit oben besprochenen Randvorgaben bestimmt werden. Als Ausgabeparameter erhält man jeweils die Koeffizientenmatrix C und falls erwünscht die Funktionswerte ys(i):= s(xs(i)) des Splines s an beliebig vorgebbaren Stützstellen xs(i).

Durch den Eingabeparameter "MODE" in der Prozedur "CUBSPIN" kann eine der 3 verschiedenen Interpolationsaufgaben (2.2.3) gewählt werden. Die Auswahl der Randvorgaben kann nach folgenden Gesichtspunkten geschehen.

Hermite-Endbedingungen eignen sich gut zur Interpolation von analytisch vorliegenden, differenzierbaren Funktionen, wenn deren Ableitungen an den Rändern des Intervalles leicht beschafft werden können. Ebenso eignet sich diese Methode für Anwendungen (Randwertaufgaben, Operatorgleichungen), bei der der interpolierende Spline an den Rändern festgelegte

Steigungen haben soll.

Natürliche Splines werden hauptsächlich zur Interpolation von grossen Datenfeldern benutzt. Eine eventuelle Abweichung der 2. Ableitung an den Rändern wirkt sich im Inneren des Definitionsgebietes nicht mehr wesentlich aus.

Periodische Randvorgaben braucht man hauptsächlich zur Parameterdarstellung von geschlossenen Kurven in der Ebene oder im Raum. Soll durch das Datenfeld (X,Y) eine geschlossene, stetige und differenzierbare Kurve mit den Koordinaten (x(t) , y(t)) gelegt werden, so definiert man sich eine streng geordnete "Zeit"-Skala T:= (t_i), i:= 1,...,n, und berechnet zu den Datenfeldern (T,X) bzw (T,Y) die jeweilige Splinedarstellung s_x bzw s_y und daraus die Koordinatenwerte ($s_x(t)$, $s_y(t)$) für gewünschte Parameter t. Die Gestalt der erhaltenen geschlossenen Kurve ist von der Wahl des Parameters t unabhängig; man kann z.B. das Intervall $[0,1]$ und $t_i := 1/(n-1)$ wählen. Zur Darstellung einer geschlossenen und überall 2-mal-stetig differenzierbaren Kurve müssen sowohl s_x wie auch s_y die Randbedingungen $s(t_1) = s(t_n)$, $s'(t_1) = s'(t_n)$ und $s''(t_1) = s''(t_n)$ der periodischen Spline-Interpolationsaufgabe erfüllen.

Das Programmpaket "CUBSPLIN" besteht aus den 6 FORTRAN-Prozeduren "CUSPIN", "CUINPO", "CUSPCO", "CUSPCP", "CUSPEV" und "CUSPED' sowie aus den beiden Testprogrammen "CUSPTEST" und "VE80UT". .

- "CUSPIN" dient als eigentliche Benutzerschnittstelle. Es besorgt die Parametervorgabe fur "CUSPCO" bzw "CUSPCP" und "CUSPEV" bzw. "CUSPED".

- "CUINPO" wird, falls notwendig, zur kubischen Polynominterpolation des Datenfeldes (X,Y) an den Rändern gebraucht.

- "CUSPCO" berechnet für die Interpolationsaufgaben (2.2.3.1) und (2.2.3.2) (Hermite-Endbedingungen bzw. Natürliche Splines) die Koeffizientenmatrix C des interpolierenden Splines s.

- "CUSPCP" berechnet für die Interpolationsaufgabe (2.2.3.3) (Periodische Endbedingungen) die Koeffizientenmatrix C von s.

- "CUSPEV" entwickelt, falls erwünscht, mit Hilfe des Hornerschemas für einen vorgebenen Stutzstellenvektor xs die dazugehörigen Funktionswerte des Splines s.

- "CUSPED" entwickelt, falls erwünscht, zu xs Funktions- und Ableitungswerte (1. und 2. Ableitung).

- "CUSPTEST" kann als Testprogramm der obigen Prozeduren verwendet werden.

- "VE80JT" dient zur Ausgabe von REAL*8-Vektoren zu Testzwecken.

Eine ausführliche Beschreibung zur Parameterversorgung ist den jeweiligen Prozeduren als Kommentar beigefügt.

```
C***********************************************************************
C
      SUBROUTINE CUSPIN( X,Y,N,MODE,RVOR,COEFF,IC,NS,XS,YS,WK,NWK,
     #                   IER, IDEV)
C
C***********************************************************************
C
C     PROZEDUR: CUSPIN   VERSION: CMS-FORTRAN IV/77          16.3.83
C
C***********************************************************************
C
C     AUTOR: J. HELLAUER, IPP                                TELEFON: 49
C
C***********************************************************************
C
C     ZWECK:
C
C        CUSPIN BESTIMMT FUER DAS DATENFELD (X,Y) EINE KUBISCHE
C        SPLINEINTERPOLATION S UND BERECHNET, FALLS ERWUNSCHT, DEREN
C        FUNKTIONSWERTE YS(I):= S( XS(I) ) VON S AN DEN STUTZSTELLEN
C        XS.
C
C        DURCH VORGABE DES PARAMETERS "MODE" KONNEN SPLINEINTERPOLA-
C        TIONEN MIT VERSCHIEDENEN RANDVORGABEN GEWAHLT WERDEN:
C
C        CUSPIN BENUTZT ZUR BERECHNUNG DER KOEFFIZIENTENMATRIX COEFF
C        DIE PROZEDUR CUSPCO BZW CUSPCP. MIT HILFE DIESER MATRIX
C        LASST SICH FUR EIN Z MIT  X(I) < Z < X(I+1), I:= 1,...,N-1
C        UND  H:= Z - X(I)  DESSEN FUNKTIONSWERT S(Z) ZU
C
C           S(Z) = Y(I)  +  COEFF(I,1) +
C                + COEFF(I,2)*H**2  +  COEFF(I,3)*H**3
C
C        BERECHNEN.
C
C***********************************************************************
C
C     AUFRUF:
C
C        CALL CUSPIN( X, Y, N, MODE, RVOR, COEFF, IC, NS, XS, YS,
C                     WK, NWK, IER, IDEV )
C
C     ******************************************************************
C
C     INPUT:
C
C        X.....: REAL*8 - VEKTOR DER LAENGE N,
C                ENTHAELT DIE ABSZISSENWERTE DES DATENFELDES (X,Y).
C                X MUSS STRENG GEORDNET SEIN, D.H.
C                X(1) < ... < X(I) < X(I+1) < ... < X(N).
C
C
C        Y.....: REAL*8 - VEKTOR DER LAENGE N,
C                ENTHAELT DIE N ORDINATENWERTE DES DATENFELDES (X,Y)
```

```
C               ODER DIE FUNKTIONSWERTE Y(I):= F( X(I) ) EINER ZU
C               INTERPOLIERENDEN FUNKTION F, DIE AUF DEM INTERVALL
C               (. X(1) , X(N) ) DEFINIERT IST.
C
C
C         N.....: INTEGER,
C                 DIMENSION VON X UND Y.
C
C
C         MODE..: INTEGER,
C                 STEUERPARAMETER DER RANDVORGABEN.
C
C                 MODE:=-1,   HERMITE-SPLINEINTERPOLATION:
C                               S'( X(1) ) := F'( X(1),
C                               S'( X(N) ) := F'( X(N).
C
C                             DIE WERTE  F'( X(1) )  BZW  F'( X(N) )
C                             DER ABLEITUNG VON F AN DEN RANDPUNKTEN
C                             X(1) UND X(N) MUSSEN ALS EINGABEPARA-
C                             METER RVOR(1) UND RVOR(2) BEIM AUFRUF
C                             VON CUSPIN UBERGEBEN WERDEN.
C
C                             EINGABE FUER RVOR:
C                             RVOR(1) = F'(X(1)), RVOR(2) = F'(X(N))
C
C
C                 MODE:=-2,   WIE MODE:=-1, DIE WERTE F'( X(1) ) BZW
C                             F'( X(N) ) WERDEN JEDOCH NAHERUNGSWEISE
C                             IN DER PROZEDUR "CUINPO" AUS DEN WERTEN
C                             DES DATENFELDES (X,Y) BERECHNET.
C
C
C                 MODE:=-3,   NATUERLICHE SPLINES:
C                               S"( X(1) ) := 0,
C                               S"( X(N) ) := 0.
C
C
C                 MODE:=-4,   PERIODISCHE SPLINES,
C                             (NUR FUR EINGABEDATEN (X,Y)
C                             Y(1) = Y(N)  ZULASSLICH).
C
C                               S'( X(1) ) := S'( X(N) ),
C                               S"( X(1) ) := S"( X(N) ).
C
C
C                 MODE:=-5,   SPLINES MIT BELIEBIGEN, EINZUGEBENDEN
C                             RANDVORGABEN  RVOR(I), I:= 1,..,4.
C
C                             DER INTERPOLIERENDE SPLINE S WIRD
C                             SO BESTIMMT, DASS GILT:
C
C                             2*S"( X(1) )+RVOR(1)*S"(X(2)) = RVOR(2)
C                             RVOR(3)*S"(X(N-1))+2*S"(X(N)) = RVOR(4)
C
C                             DIE WERTE  RVOR(1),..., RVOR(4)  MUSSEN
```

```
C                    DEMENTSPRECHEND VORGEGEBEN WERDEN.
C
C
C           MODE:= NN,  MIT  NN < 1   ODER   NN > 5,
C                       SPLINEINTERPOLATION ERFOLGT
C                       WIE MODE:=-3.
C
C
C       RVOR..: REAL*8-VEKTOR DER LAENGE 4,
C               RANDVORGABEN AN DEN INTERPOLIERENDEN SPLINE S.
C               ( SIEHE PARAMETER 'MODE' )
C
C
C       IC....: SPALTENZAHL DER KOEFFIZIENTENMATRIX COEFF(IC,3),
C               MUSS BEIM AUFRUF MIT  IC:= N -1 VORBESETZT
C               WERDEN.
C
C
C       NS....: INTEGER, NS > 0,
C               LAENGE DES ABSZISSENVEKTORS XS UND
C               DES DAZUGEHOERIGEN ORDINATENVEKTORS YS.
C
C               NS:= 1  BEDEUTET: KEINE AUSWERTUNG DER
C                                 SPLINEFUNKTIONSWERTE YS.
C
C
C       XS....: REAL*8-VEKTOR DER LAENGE NS,
C               ENTHAELT DIE ABSZISSENWERTE XS(I) AN DENEN
C               DIE FUNKTIONSWERTE YS(I):= S( XS(I) ) DES
C               ZU BERECHNENDEN SPLINES AUSGEWERTET WERDEN
C               SOLLEN.
C
C
C       NWK...: INTEGER, DIMENSION VOM HILSFELD WK.
C
C               NWK:= N, FUER  MODE:=-4
C                  := 1, SONST.
C
C
C       WK....: REAL*8-MATRIX MIT DER DIMENSION (6,NWK),
C               WIRD NUR BEI MODE:=-4 (PERIODISCHE SPLINES)
C               GEBRAUCHT.
C
C
C       IDEV..: INTEGER,
C               KANAL-NUMMER DER TESTAUSGABE.
C
C               IDEV < 6,          KEINE TESTAUSGABE.
C               IDEV = 6,          TESTAUSGABE AM TERMINAL
C               IDEV = NN > 6,     TESTAUSGABE IN CMS-DISK-FILE
C                                  FTNNF001
C
C
C     ------------------------------------------------------------
C
```

```
C      OUTPUT:
C
C         COEFF.: REAL*8-MATRIX DER DIMENSION (IC,3),
C                 ENTHAELT DIE SPLINEKOEFFIZIENTEN COEFF(I,K)
C                 MIT DER OBEN ERKLAERTEN BEDEUTUNG.
C
C
C         YS....: REAL*8-VEKTOR DER LAENGE NS,
C                 ENTHAELT FUR   NS > 1  DIE
C                 FUNKTIONSWERTE DES BERECHNENDEN SPLINES S
C                 AN DEN STUTZSTELLEN XS(I), I:= 1,..., NS.
C
C
C         IER...: INTEGER, ERROR PARAMETER,
C                 RETURNCODE FOR TERMINAL ERROR
C
C                 IER = 129, IC IST KLEINER ALS N - 1,
C                 IER = 130, N IST KLEINER ALS 2,
C                 IER = 131, ABSZISSENSKALA X IST NICHT SO GEORDNET
C                            DASS  X(1) < X(2) < ... < X(N).
C
C
C***********************************************************************
C
C      EXTERNE PROZEDUREN:       CUINPO (FORTRAN)
C                                CUSPCO (FORTRAN)
C                                CUSPCP (FORTRAN)
C                                CUSPEV (FORTRAN)
C
C***********************************************************************
C
C
       IMPLICIT REAL*8 ( A- H, O - Z )
C
       DIMENSION  X(N), Y(N), COEFF(IC,3), RVOR(4), WK(NWK)
       DIMENSION  RBED(4), AC(4), DY(4), D2Y(4), XS(NS), YS(NS)
C
       DATA  EPS /1.0D-5/
C
C
     1 NMODE = IABS(MODE)
C
       IF ( NMODE .LT. 1  .OR.  NMODE .GT. 5 )   NMODE = 3
C
       GOTO ( 1000, 2000, 3000, 4000, 5000 ), NMODE
C
  1000 CONTINUE
C
C      MODE =-1,-2  :     INTERPOLATION MIT HERMITE-BEDINGUNGEN
C
       H1 = X(2) - X(1)
       HN = X(N) - X(N-1)
C
       RBED(2) = 6.0/(H1*H1) * ( Y(2) - Y(1) - RVOR(1)*H1 )
       RBED(1) = 1.0
```

```
      RBED(4) = 6.0/(HN*HN) * ( RVOR(2)*HN - Y(N) + Y(N-1) )
      RBED(3) = 1.0
C
      GOTO 9000
C
 2000 CONTINUE
C
      CALL CUINPO( X, Y, N, 1, AC, DY, D2Y, IER, IDEV )
C
      RVOR(1) = DY(1)
C
      CALL CUINPO( X, Y, N, N-3, AC, DY, D2Y, IER, IDEV )
C
      RVOR(2) = DY(4)
C
      GOTO 1000
C
 3000 CONTINUE
C
C     NATURLICHE SPLINES
C
      DO 3100 K = 1, 4
 3100 RBED(K) = 0.0D0
C
      GOTO 9000
C
C
 4000 CONTINUE
C
C     PERIODISCHE SPLINES SIND NUR DANN SINNVOLL,
C     WENN DIE DATEN DIE BEDINGUNG  Y(1) = Y(N)
C     ERFUELLEN.
C     FALLS DIESE BEDINGUNG NICHT GEGEBEN IST,
C     ERFOLGT EINE WARNUNG UND DER SPLINE WIRD
C     NACH MODE:=2 BERECHNET.
C
      TRUEPS = DABS( Y(1) - Y(N) )
C
      IF ( TRUEPS .LE. EPS  )   GOTO 4100
C                                                    ELSE,
      WRITE(6,4050) Y(1),N, Y(N), TRUEPS, EPS
 4050 FORMAT(/'*** WARNING FROM SUBROUTINE CUSPIN ***'/
     # /' GIVEN DATA FIELD NOT SUITABLE FOR INTERPOLATION'/
     # /' BY PERIODIC SPLINES, BECAUSE:'/
     # /' Y(1) = ',E15.8,' DIFFERS TOO MUCH FROM Y(',I4,
     #  ') = ',E15.8/
     # /' ABS( Y(1) - Y(N) ) = ',E15.8,' >  EPS:= ',E15.8//
     # /' SPLINE WILL BE CALCULATED LIKE   MODE:= 2 '/ )
C
      MODE = -2
      GOTO 1
C
C
 4100 CALL CUSPCP( X, Y, N, COEFF, IC, WK, IER )
C
```

```
      IF ( IER .EQ. 0 )   GOTO 10000
C
      WRITE(6,4150) IER
 4150 FORMAT(/' ERRORMESSAGE FROM SBR CUSPCP!'/ ' RETURNCODE: ',
     # ' IER = ',I4/)
C
      GOTO 9100
C
C
 5000 CONTINUE
C
      DO 5100 K = 1, 4
 5100 RBED(K) = RVOR(K)
C
C
 9000 CALL CUSPCO ( X, Y, N, RBED, COEFF, IC, IER )
C
      IF ( IER .EQ. 0 )  GOTO 10000
C
      WRITE(6,128) IER
  128 FORMAT(/' ERRORMESSAGE FROM SBR CUSPCO!'/ ' RETURNCODE: ',
     # ' IER = ',I4/)
C
 9100 NM1 = N - 1
C
      IF ( IER .EQ. 129 )    WRITE(6,129) IC , NM1
  129 FORMAT(' IC = ',I4,' IS NOT EQUAL TO  (N-1) = ',I4/)
C
      IF ( IER .EQ. 130 )    WRITE(6,130)  N
  130 FORMAT(' N = ',I4,' IS LESS THAN 2'/)
C
      IF ( IER .EQ. 131 )    WRITE(6,131)
  131 FORMAT(' INPUT ABSCISSA X(I) ARE NOT ORDERED SO THAT'/
     # ' X(1) < ... < X(I) < X(I+1) < ... < X(N)'/)
C
      WRITE(6,133)
  133 FORMAT(' CALCULATION OF COEFFICIENT MATRIX COEFF '/
     # ' IS NOT POSSIBLE'//' PLEASE CORRECT YOUR INPUT PARAMETER'/
     # ' AND TRY AGAIN!'/)
C
C
10000 IF ( IDEV .LE. 5 ) GOTO 11000
C
      WRITE(IDEV,90)  MODE, ( RBED(K), K = 1, 4 )
   90 FORMAT(/'******** TEST OUTPUT FROM SBR CUSPIN ***** '//' ',
     #         ' MODE = ',I3,
     #        /' RBED(1) = ',E12.6,',      RBED(2) = ',E12.6/
     #         ' RBED(3) = ',E12.6,',      RBED(4) = ',E12.6//)
C
      WRITE(IDEV,100)
  100 FORMAT(' K',T6,'   X',T19,'  Y',T32,'COEFF(K,1)',T45,
     # 'COEFF(K,2)',T58,'COEFF(K,3)'/)
C
      DO 2010 K = 1, N
 2010 WRITE(IDEV,110) K, X(K), Y(K),COEFF(K,1),COEFF(K,2),COEFF(K,3)
```

```
  110 FORMAT(' ',I3,T6,F10.4,T19,E9.3,T32,E9.3,T45,E10.4,T56,E10.4)
C
11000 IF ( NS .LE. 1 )  RETURN
C
C     AUFRUF VON CUSPEV ZUR SPLINEENTWICKLUNG IN DEN PUNKTEN XS(I)
C
      CALL CUSPEV( X, Y, N, COEFF, IC, XS, YS, NS, IER, IDEV)
C
      IF ( IER .EQ.  999 )    WRITE(6,999)  X(1), N, X(N)
  999 FORMAT(/' ONE OR MORE ABSCISSA VALUES XS(I) IS OUT OF THE'/
     # ' DATA RANGE: X(1):= ',F12.4,' < XS(I) < X(',I4,'):= ',F12.6/
     # '.'//' EVALUATION OF THE WANTED VALUES  YS(I):= S( XS(I) )'/
     # ' IS NOT POSSIBLE.'//' PLEASE CHOOSE A NEW SKALA XS OR A NEW'
     # /' DATA FIELD (X,Y) AND TRIE AGAIN!'//)
C
      RETURN
C
C     ENDE VON CUSPIN
C
      END
```

```
C*********************************************************************
C
      SUBROUTINE CUSPCO( X, Y, N, BCON, C, IC, IER )
C
C*********************************************************************
C
C       FUNCTION:
C
C          CALCULATES THE COEFFICIENTS OF AN INTERPOLARTORY CUBIC
C          SPLINE APPROXIMATION FOR A GIVEN DATA FIELD (X,Y) WITH
C          ARBITRARY BOUNDARY CONDITIONS.
C
C          CUSPCO IS USED FROM SUBROUTINE "CUSPIN" FOR CALCULATING C.
C
C*********************************************************************
C
C       LINKAGE:
C
C          CALL CUSPCO( X, Y, N, BCON, C, IC, IER)
C
C*********************************************************************
C
C       INPUT:
C
C          X.....: REAL*8-VECTOR OF LENGTH N CONTAINING THE ABSCISSAE
C                  OF THE N DATA POINTS (X(I),Y(I)) I=1,...,N.
C                  X MUST BE ORDERED SO THAT <X(I) < X(I+1).
C
C
C          Y.....: REAL*8-VECTOR OF LENGTH N CONTAINING THE ORDINATES
C                  (OR FUNCTION VALUES) OF THE N DATA POINTS.
C
C
C          N.....:  INTEGER,  N > 1,
C                   NUMBER OF ELEMENTS IN X AND Y.
C
C
C          BCON..: REAL*8-VECTOR OF LENGTH 4 CONTAINING THE END
C                  CONDITION PARAMETERS. LOOK REMARKS TO SBR 'CUSPIN'
C
C                     2.0*S"(1)+BCON(1)*S"(2) = BCON(2),
C                     BCON(3)*S"(N-1)+2.0*S"(N) = BCON(4),
C
C                  WHERE S"(I) = SECOND DERIVATIVE OF THE
C                  CUBIC SPLINE FUNCTION S EVALUATED AT X(I).
C
C          IC....: INTEGER,  IC:= N - 1,
C                  ROW DIMENSION OF THE MATRIX C
C
C
C*********************************************************************
C
C      OUTPUT
C
C          C.....:  REAL*8- MATRIX WITH DIMENSION (IC,3).
```

```
C               C CONTAINS THE SPLINE COEFFICIENTS OF S.
C               THE VALUE OF THE SPLINE APPROXIMATION AT Z IS
C
C                  S(Z) = ((C(I,3)*H+C(I,2))*H+C(I,1))*H+Y(I),
C
C               WHERE  X(I) <= T < X(I+1) AND H:= Z - X(I).
C
C
C       IER....:  INTEGER, ERROR PARAMETER,
C                 RETURNCODE FOR TERMINAL ERROR
C
C                   IER = 129, IC IS LESS THAN N-1.
C                   IER = 130, N IS LESS THAN 2.
C                   IER = 131, INPUT ABSCISSA  X ARE NOT ORDERED
C                              SO THAT  X(1) < X(2) < ... < X(N).
C
C
C*************************************************************************
C
```

```
C*********************************************************************
C                                                                     *
      SUBROUTINE CUSPCP ( X, Y, NX, C, IC, WK, IER )
C                                                                     *
C*********************************************************************
C
C       FUNCTION:
C
C          CALCULATES THE COEFFICIENTS OF AN INTERPOLARTORY CUBIC
C          SPLINE APPROXIMATION FOR A GIVEN DATA FIELD (X,Y) WITH
C          PERIODIC END CONDITIONS.
C
C          CUSPCP IS USED FROM SUBROUTINE "CUSPIN" FOR CALCULATING
C          THE SPLINE COEFFICIENT MATRIX "C".
C
C*********************************************************************
C
C       LINKAGE:
C
C          CALL CUSPCP( X, Y, NX, C, IC, WK, IER )
C
C*********************************************************************
C
C       INPUT:
C
C          X.....: REAL*8-VECTOR OF LENGTH NX,
C                  CONTAINING THE ABSCISSAE OF THE NX
C                  DATA POINTS (X(I),Y(I)) I=1,...,NX.
C                  X MUST BE ORDERED SO THAT <X(I) < X(I+1).
C
C
C          Y.....: REAL*8-VECTOR OF LENGTH NX,
C                  CONTAINING THE ORDINATES (OR FUNCTION VALUES)
C                  OF THE NX DATA POINTS.
C
C
C          NX....:  INTEGER,  NX > 1,
C                   NUMBER OF ELEMENTS IN X AND Y.
C
C
C          IC....: INTEGER,  IC:= NX - 1,
C                  ROW DIMENSION OF THE MATRIX C
C
C
C          WK....:  REAL*8-MATRIX, DIMENSION (6,NX),
C                   WORKING ARREA.
C
C
C*********************************************************************
C
C       OUTPUT
C
C          C.....:  REAL*8- MATRIX WITH DIMENSION (IC,3).
C                   C CONTAINS THE SPLINE COEFFICIENTS OF S.
C                   THE VALUE OF THE SPLINE APPROXIMATION AT Z IS
```

```
C
C                 S(Z) = ((C(I,3)*H+C(I,2))*H+C(I,1))*H+Y(I)
C                 WHERE X(I) .LE. T .LT. X(I+1) AND H:= Z - X(I)
C
C
C       IER....:  INTEGER, ERROR PARAMETER,
C                 RETURNCODE FOR TERMINAL ERROR
C
C                   IER = 129, IC IS LESS THAN NX - 1.
C                   IER = 130, N IS LESS THAN 2.
C                   IER = 131, INPUT ABSCISSAE IS NOT ORDERED
C                              SO THAT  X(1) < X(2) < ... < X(N).
C
C
```

```
C*********************************************************************
c                                                                     *
      SUBROUTINE CUINPO ( X, Y, N, NI, AK, DY, D2Y, IER, IDEV )
C                                                                     *
C*********************************************************************
C
C     PROCEDURE:  CUINPO    VERSION: FORTRAN IV/77     DATE: 16.3.83
C
C*********************************************************************
C
C     AUTHOR:  J. HELLAUER, IPP                          PHONE:  494
C
C*********************************************************************
C
C     FUNCTION:
C
C        INTERPOLATION OF THE 4 DATA POINTS
C             ( X(NI),Y(NI) ),...,( X(NI+3),Y(NI+3) )
C        BY A CUBIC INTERPOLATION POLYNOM P AND CALCULATION OF ITS
C        1. AND 2. DERIVATION IN THESE ABSCISSAE.
C
C
C*********************************************************************
C
C     LINKAGE:
C
C        CALL CUINPO( X, Y, NI, AK, DY, D2Y, IER, IDEV )
C
C*********************************************************************
C
C     INPUT:
C
C        X.....:  REAL*8, WITH LENGHT N,
C                 CONTAINING THE ABSCISSAE OF THE DATAFIELD ( X,Y ).
C
C
C        Y.....:  REAL*8, WITH LENGHT N,
C                 CONTAINING THE ORDINATES OF THE DATAFIELD ( X, Y).
C
C
C        N.....:  INTEGER, LENGHT OF X AND Y.
C
C
C        NI....:  INTEGER,   1 <= NI <= N-3,
C                 INDEX OF THE FIRST 4 DATAPOINT X(I), IN WHICH
C                 CUBIC POLYNOM INTERPOLATION SHOULD BE DONE.
C
C
C        IDEV..: INTEGER,
C                UNIT  NUMBER FOR TEST OUTPUT,
C
C                   IDEV < 6,         NO TEST OUTPUT,
C                   IDEV: = 6,        TEST OUTPUT ON TERMINAL,
C                   IDEV: = NN > 6,   TEST OUTPUT IN CMS DISK
C                                     FILE 'FILE FTNNF001 A'.
```

```
C
C-----------------------------------------------------------------
C
C      OUTPUT:
C
C         AK....:  REAL*8, WITH LENGHT 4,
C                  CONTAINING THE 4 COEFFICIENTS OF THE CUBIC INTER-
C                  POLATION POLYNOM P.  FOR  T:= X - X(NI) IS,
C                  P(T):= AK(1) + AK(2)*T + AK(3)*T**2 + AK(4)*T**3.
C
C
C         DY....:  REAL*8, WITH LENGHT 4,
C                  CONTAINS THE VALUES OF 1. DERIVATION OF P IN THE
C                  ABSCISSAE  X(NI), X(NI+1), X(NI+2) AND X(NI+3),
C                  I. E.
C                          DY(I):=  D( P( X(NI-1+I) ) )/ DT.
C
C
C         D2Y...:  REAL*8, WITH LENGHT 4,
C                  CONTAINS THE VALUES OF 2. DERIVATION OF P IN THE
C                  ABSCISSAE  X(NI), X(NI+1), X(NI+2) AND X(NI+3),
C                  I. E.
C                        D2Y(I):=  D2( P( X(NI-1+I) ) )/ DT**2.
C
C
C         IER...:  INTEGER,  RETURNCODE FOR ERROR MESSAGES,
C
C                  IER:= 0,  NO ERROR DETECTED,
C                  IER:= -998,  IF  NI < 1  OR  NI+3 > N
C                  IER:= -999,  IF NOT :  X(NI) < ... < X(NI+3).
C
C
C*****************************************************************
C
C      EXTERNAL FUNCTIONS:        NONE
C
C*****************************************************************
C
C
       IMPLICIT REAL*8 ( A-H, O - Z )
C
       DIMENSION X(N), Y(N), AK(4), DY(4), D2Y(4), T(3)
C
       IER = 0
C
       IF ( NI .LE. 0  .OR.  NI+3 .GT. N )   GOTO 998
       IF ( X(NI+1) .LE. X(NI)  .OR. X(NI+2) .LE. X(NI+1)  .OR.
      1     X(NI+3) .LE. X(NI+2)   )    GOTO 999
C
       DO 1000 I = 1,3
          T(I) = X(NI+I) - X(NI)
          DY(I) = Y(NI+I) - Y(NI)
 1000  D2Y(I) = DY(I)/T(I)
C
C      CALCULATION OF AK
```

```
C
      DTI = 1.0D0/( T(2)-T(3) )
      AK(4) = (D2Y(1)-D2Y(2))*DTI/(T(1)-T(2))
      AK(4) = AK(4) - ( D2Y(1)-D2Y(3) )*DTI/( T(1)-T(3) )
C
      AK(3) = (D2Y(1)-D2Y(2))/(T(1)-T(2)) - AK(4)*(T(1)+T(2))
C
      AK(2) = D2Y(1) - AK(3)*T(1) - AK(4)*T(1)*T(1)
C
      AK(1) = Y(NI)
C
C     DETERMINATION OF DY AND D2Y
C
      DY(1) = AK(2)
      D2Y(1) = 2.0*AK(3)
C
      DO 3000 K = 1,3
        DY(K+1) = ( 3.0*AK(4)*T(K) + 2.0*AK(3) )*T(K) + AK(2)
 3000 D2Y(K+1) = 6.0*AK(4)*T(K) + 2.0*AK(3)
C
      IF ( IDEV .LT. 5 )  RETURN
C
C     TEST OUTPUT
C
      WRITE(IDEV,10)
   10 FORMAT(///' ***** TEST OUTPUT FROM SBR CUINPO **** '/)
C
      WRITE(IDEV,20)
   20 FORMAT(/' K',T12,'X',T25,'Y',T38,'A(K)',T53,'DY(K)',T64,
     # 'D2Y(K)'/)
   30 FORMAT(' ',I3,X ,E12.5,X  ,E12.5,X  ,E12.5, X ,E12.5,X  ,
     # E12.5 )
C
      NIM1 = NI - 1
C
      DO 4000 K = 1, 4
 4000 WRITE(IDEV,30) K, X(NIM1+K), Y(NIM1+K), AK(K), DY(K), D2Y(K)
C
      RETURN
C
  998 IER = -998
C
      RETURN
C
  999 IER = -999
C
      RETURN
C
C     END OF CUINPO
C
      END
```

```
C*********************************************************************
C                                                                     *
      SUBROUTINE CUSPEV (X,Y1,N,A,IA,XS,YS,NS,IER,IDEVAT)
C                                                                     *
C*********************************************************************
C
C     PROCEDURE: CUSPEV        VERSION: CMS         DATE: 16. DEC 82
C
C*********************************************************************
C
C     AUTHOR:    J. HELLAUER                              PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        EVALUATES WITH THE COEFFICIENT MATRIX A, WHICH WAS BUILT IN
C        A SUITABLE PROZEDURE BEFORE, THE FUNCTION VALUS YS(I) OF
C        THE SMOOTHING SPLINE SP AT POINTS VALUES XS(I).
C
C        EVALUATION HAPPENS WITH THE METHODE OF HORNER. FOR GIVEN Z,
C        X(I) <= Z <= X(I+1), THE VALUE SP(Z) CAN BE CALCULATED AS
C            SP(Z) = (( A(I,3)*H + A(I,2) )*H + A(I,1) )*H + Y1(I),
C        THEREFOR  H:= Z - X(I).
C
C*********************************************************************
C
C     LINKAGE:
C
C        CALL CUSPEV(X, Y1, N, A, IA, XS, YS, NS, IER, IDEVAT )
C
C    ---------------------------------------------------------------
C
C     INPUT:
C
C        X ....: REAL*8, DIMENSION X(N),
C                CONTAINING THE ABSCISSAE OF THE DATAFIELD (X,Y1).
C
C
C        Y1....: REAL*8, DIMENSION Y1(N),
C                CONTAINING THE ORDINATES OF THE DATAFIELD (X,Y1).
C
C
C        N.....: INTEGER,
C                LENGHT OF X AND Y1.
C
C
C        A ....: REAL*8,  DIMENSION A(IA,3)
C                COEFFICIENT MATRIX OF THE PLOTTING SPLINE SP. FOR
C                ITS USE SEE SOME LINES AGO. THIS MATRIX IS TO BE-
C                CALCULATED IN A PROCEDURE BEFORE.
C
C
C        IA....: INTEGER, DIMENSION OF COEFFICIENT MATRIX A,
C                IA MUST BE SET AS
```

```
C                 IA:= N - 1 .
C
C
C        XS ...: REAL*8, DIMENSION XS(NS),
C                COORDINATE-VECTOR, CONSISTS OF THE POINTS, IN
C                WHICH SPLINE FUNKTION SP SHOULD BE EVALUATED.
C
C
C        NS ...: INTEGER,
C                DIMENSION OF XS.
C
C
C        IDEVAT: INTEGER, UNIT CHANNEL FOR TEST OUTPUT.
C                ( SEE PROCEDURE "CUSPIN")
C
C
C     ------------------------------------------------------------
C
C     OUTPUT:
C
C        YS ...: REAL*8, DIMENSION YS(NYS)
C                YS CONSISTS OF FUNCTION VALUES OF THE
C                SMOOTHING SPLINE-FUNCTION SP IN THE POINTS XS(I),
C                YS(I):= SP( XS(I) ), FOR  1 <= I <= NS.
C
C
C        IER...: INTEGER,
C                IER = 0,    NO ERROR DETECTED.
C                IER = 999, ONE OF THE EVALUATION POINTS XS(I) IS
C                           OUT OF THE RANGE  (. X(1) , X(N) .).
C                           YS(I) COULD NOT BE CALCULATED.
C
C
C*********************************************************************
C
C     OPERATING SYSTEM:     CMS
C
C*********************************************************************
C
C     EXTERNAL PROCEDURES:      NONE
C
C*********************************************************************
C
C
      IMPLICIT  REAL*8 ( A - H, O - Z )
C
      DIMENSION   X(N), A(IA,3), Y1(N)
      DIMENSION XS(NS), YS(NS)
C
      IER = 0
C
      IF ( IDEVAT .GE. 6 )  WRITE(IDEVAT,10) N, IA, NS
   10 FORMAT(//' TEST OUTPUT FROM SBR CUSPEV '//
     #' N = ',I4,',   IA = ',I4,',   NS = ',I4//)
C
```

```
C
      NM1 = N - 1
C
C     SEARCH FOR THE SUITABLE INTERVAL WITH  X(I) <= XS(1) <= X(I+1)
C
      I = 1
      J = 0
C
 2000 J = J + 1
      Z = XS(J)
C
 3000 IF ( Z .GE. X(I) )      GOTO 3100
      IF ( I .LE. 1 )         GOTO 999
      I = I - 1
      GOTO  3000
C
 3100 IF ( I .GE. NM1 )       GOTO 4000
      IF ( Z .LT. X(I+1) )    GOTO 4500
      I = I + 1
      GOTO 3000
C
 4000 I = NM1
C
C     SUITABLE INTERVAL FOUND
C
 4500 H = Z - X(I)
C
C     METHODE OF HORNER
C
      YS(J) = ( ( A(I,3)*H + A(I,2) )*H + A(I,1) )*H + Y1(I)
C
C     IF ( IDEVAT .GE. 6 )
C    #WRITE(IDEVAT,20) J, XS(J), I, X(I), H, YS(J)
C  20 FORMAT(' XS(',I4,') = ',E12.4,',   X(',I5,') = ',E12.4/' ',
C    # '  H = ',E12.4,',  YS = ',E12.4)
C
      IF ( J .LT. NS )   GOTO 2000
C
C     EVALUATION IS READY
C
      RETURN
C
 999  IER = 999
C
      IF ( IDEVAT .GE. 6 ) WRITE(IDEVAT,99) J,XS(J),X(1),N,X(N)
   99 FORMAT(//' ERRORMESSAGE FROM SUBROUTINE CUSPEV '//
     #' EVALUATIONPOINT  XS(',I4,') = ',E12.6,'  IS OUT OF RANGE,',
     #/' X(1) = ',E12.4,' <= XS <=  X(',I4,') = ',E12.4//)
C
      RETURN
C
C     END OF CUSPEV
C
      END
```

```
C*********************************************************************
C                                                                     *
      SUBROUTINE CUSPED ( X, Y1, N, A, IA, XS, NS, NYS, NDY, ND2,
     #                    YS, DYSDX, D2YSD2, IER, IDEVAT )
C                                                                     *
C*********************************************************************
C
C     PROCEDURE: CUSPED        VERSION: CMS          DATE: 25. NOV 82
C
C*********************************************************************
C
C     AUTHOR:    J. HELLAUER                          PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        EVALUATES WITH THE COEFFICIENT MATRIX A, WHICH WAS BUILT IN
C        A PROCEDURE AGO, THE FUNCTION AND ITS DERIVATION VALUES
C        OF THE WANTED SMOOTHING SPLINE FUNKTION SPLINE SP AT POINTS
C        XS(I), 1 <= I <= NS.
C
C        EVALUATION HAPPENS WITH THE METHODE OF HORNER. FOR A GIVEN
C        Z, WITH  X(I) <= Z <= X(I+1), THE VALUE SP(Z) CAN BE
C        CALCULATED AS
C           SP(Z) = (( A(I,3)*H + A(I,2) )*H + A(I,1) )*H + Y1(I),
C           D( SP(Z) ) = ( 3*A(I,3)*H + 2*A(I,2) )*H + A(I,1)    AND
C           D2( SP(Z) ) = 6*A(I,3)*H + 2*A(I,2),
C
C        THEREFOR  H:= Z - X(I).
C
C     *****************************************************************
C
C     LINKAGE:
C
C        CALL CUSPED(X, Y1, N, A, IA, XS, NS, NYS, NDY, ND2,
C                    YS, DYSDX, D2YSD2, IER, IDEVAT)
C
C*********************************************************************
C
C     INPUT:
C
C        X ....: REAL*8, DIMENSION X(N),
C                CONTAINING THE ABSCISSAE OF THE DATAFIELD (X,Y1).
C
C
C        Y1....: REAL*8, DIMENSION Y1(N),
C                CONTAINING THE ORDINATES OF THE DATAFIELD (X,Y1).
C
C
C        N.....: INTEGER,
C                LENGHT OF X AND Y1.
C
C
C        A ....: REAL*8,  DIMENSION C(IA,3),
```

```
C               COEFFICIENT MATRIX OF THE PLOTTING SPLINE SP. FOR
C               ITS USE SEE SOME LINES AGO. THIS MATRIX HAD TO BE
C               CALCULATED IN PROCEDURES AGO.
C
C       IA....: INTEGER, DIMENSION OF COEFFICIENT MATRIX A,
C               IA MUST BE SET AS
C               IA:= N - 1
C
C
C       XS ...: REAL*8, DIMENSION XS(NS),
C               CONTAINING THE ABSCISSAE, IN WHICH
C               SPLINE FUNKTION SP SHOULD BE EVALUATED.
C
C
C       NS ...: INTEGER,
C               DIMENSION OF XS.
C
C
C       NYS,
C       NDY,
C       ND2 ..: INTEGER,
C               DIMENSION FOR  YS, DYSDX AND D2YSD2
C               MUST BE SET AS
C
C               NYS:= NS :  YS      IS TO EVALUATE,
C               NDY:= NS :  DYSDX   IS TO EVALUATE,
C               ND2:= NS :  D2YSD2  IS TO EVALUATE.
C
C               ELSE,  NYS:= 1  OR   NDY:= 1   OR   ND2:= 1.
C
C
C       ---------------------------------------------------------
C
C       OUTPUT:
C
C          YS ...: REAL*8, DIMENSION YS(NYS)
C                  IF  LYS:= .TRUE., YS CONSISTS OF FUNCTION VALUES OF
C                  THE SMOOTHING SPLINE-FUNCTION SP IN THE ABSCISSAE
C                  XS(I),  YS(I):= SP( XS(I) ), FOR  1 <= I <= NS.
C
C
C          DYSDX : REAL*8, DIMENSION DYSDX(NDY),
C                  IF LDY:= .TRUE., DYSDX CONSISTS OF FUNCTION VALUES
C                  OF THE FIRST DERIVATIVE OF SP IN THE ABSCISSAE
C                  XS(I), DYSDX(I):= D(SP)( XS(I) ), FOR 1 <= I <= NS.
C
C
C          D2YSD2: REAL*8, DIMENSION  D2YSDX(ND2),
C                  IF LD2:= .TRUE., D2YSD2 CONSISTS OF FUNCTION VALUES
C                  OF THE SECOND DERIVATIVE OF SP IN THE ABSCISSAE
C                  XS(I), D2YSD2(I):= D2 (SP)( XS(I) ),  1 <= I <= NS.
C
C
C          IER...: INTEGER,
C                  IER = 0,   NO ERROR DETECTED.
```

```
C                IER = 999, ONE OF THE EVALUATION POINTS XS(I) IS
C                           OUT OF THE RANGE  (. X(1) , X(N) .).
C                           YS(I) COULD NOT BE CALCULATED.
C
C
C***********************************************************************
C
C     OPERATING SYSTEM:     CMS
C
C***********************************************************************
C
C     EXTERNAL PROCEDURES:  NONE
C
C***********************************************************************
C
C
      IMPLICIT  REAL*8 ( A - H, O - Z )
C
      DIMENSION   X(N), A(IA,3), Y1(N)
      DIMENSION XS(NS), YS(NYS), DYSDX(NDY), D2YSD2(ND2)
      LOGICAL LYS,LDY,LD2
      LOGICAL*1 ANSWER
C
      DATA  LYS, LDY, LD2  /3*.F./
C
      IF ( NYS .EQ. NS ) LYS = .TRUE.
      IF ( NDY .EQ. NS ) LDY = .TRUE.
      IF ( ND2 .EQ. NS ) LD2 = .TRUE.
C
      IF (IDEVAT .GE. 6) WRITE(IDEVAT,10) N, IA, NS, NYS, NDY, ND2
   10 FORMAT(//' TEST OUTPUT FROM SBR CUSPED '//
     #' N = ',I4,',   IA = ',I4,',   NS = ',I4/
     #' NYS = ',I4,',   NDY = ',I4,';   ND2 = ',I4//
     #'  J  ',T8,' XS(J)',T25,' I ',T32,' X(I)',T48,'    H      ',
     # T60,' YS(J)'/ )
C
      IER = 0
C
      NM1 = N - 1
C
C     SEARCH FOR THE SUITABLE INTERVAL WITH X(I)<= XS(1)<= X(I+1)
C
      I = 1
      J = 0
C
 2000 J = J + 1
      Z = XS(J)
C
 3000 IF ( Z .GE. X(I) )      GOTO 3100
      IF ( I .LE. 1 )         GOTO 999
      I = I - 1
      GOTO  3000
C
 3100 IF ( I .GE. NM1 )       GOTO 4000
      IF ( Z .LT. X(I+1) )    GOTO 4500
```

```
      I = I + 1
      GOTO 3000
C
 4000 I = NM1
C
C     SUITABLE INTERVAL FOUND
C
 4500 H = Z - X(I)
C
C     EVALUATION WITH THE METHODE OF HORNER
C
      A13MH = A(I,3)*H
C
      IF ( LYS )
     # YS(J) = ( ( A13MH + A(I,2) )*H + A(I,1) )*H + Y1(I)
C
      IF ( LDY )
     # DYSDX(J) = ( 3.0D0*A13MH + 2.0D0*A(I,2) )*H + A(I,1)
C
      IF ( LD2 )
     # D2YSD2(J) = 6.0D0*A13MH + 2.0D0 * A(I,2)
C
C
      IF ( IDEVAT .GE. 6 )
     #WRITE(IDEVAT,20) J, XS(J), I, X(I), H, YS(J)
   20 FORMAT(' 'I4,T7,E12.4,T22,I5,T30,E12.4,T46,E12.4,T58,E12.4)
C
      IF ( J .LT. NS )   GOTO 2000
C
C     EVALUATION IS READY
C
C
      RETURN
C
 999  IER = 999
C
      IF ( IDEVAT .GE. 6 ) WRITE(IDEVAT,99) J,XS(J),X(1),N,X(N)
   99 FORMAT(//' ERRORMESSAGE FROM SUBROUTINE CUSPED '//
     #' EVALUATIONPOINT  XS(',I4,') = ',E12.6,'  IS OUT OF RANGE,',
     #/' X(1) = ',E12.4,' <= XS <=  X(',I4,') = ',E12.4//)
C
C
      RETURN
C
C     END OF CUSPED
C
      END
```

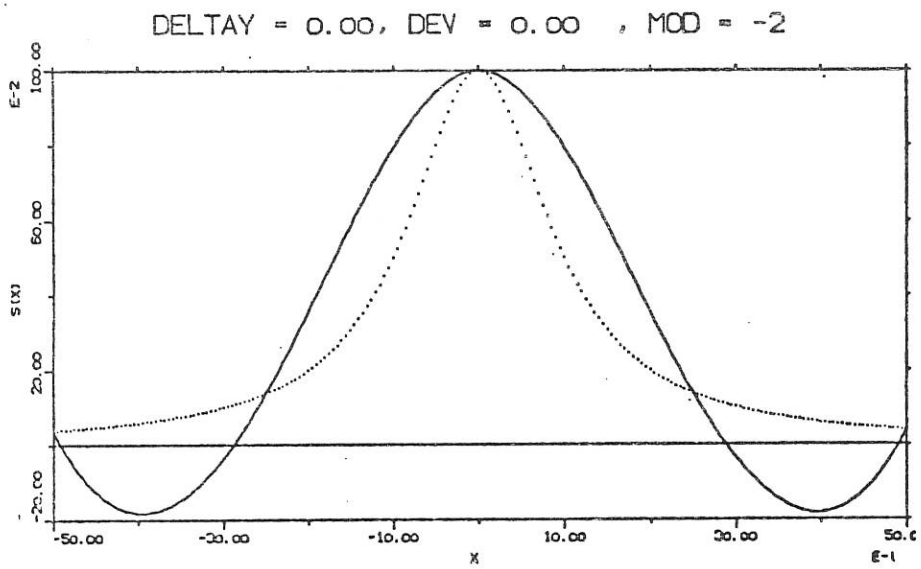
```
C*********************************************************************
C
      SUBROUTINE VE8OUT( X, Y, N, NI, NF, NSTEP, SX, NSX, SY, NSY,
     #                   TITLE, NTITLE, HEADER, NHEAD, IDEV )
C
C     PRINT THE ELEMENTS  ( X(I),Y(I) ), I = NI,NF NSTEP
C     WITH A TITLE AND HEADER.
C
      REAL*8  X(N), Y(N)
C
      LOGICAL*1  TITLE(NTITLE), HEADER(NHEAD), SX(NSX), SY(NSY)
C
      WRITE(IDEV,10) ( TITLE(K), K=1,NTITLE)
   10 FORMAT(//' ',80A1)
C
      WRITE(IDEV,20) ( HEADER(K), K = 1, NHEAD )
   20 FORMAT(//' ',80A1 )
C
      WRITE(IDEV,30)  (SX(K),K=1,NSX), (SY(K),K=1,NSY),
     1                (SX(K),K=1,NSX), (SY(K),K=1,NSY)
C
   30 FORMAT(//'  K ',T7 ,6A1,'(K)',T22,6A1,'(K)',T38,
     1       '  K ',T45,6A1,'(K)',T61,6A1,'(K)'/)
C
      NE = NF
      N2STEP = 2*NSTEP
C
      DO 100 I = NI, NE , N2STEP
         IN = MINO( I + NSTEP, NF )
  100 WRITE(IDEV, 40) I, X(I), Y(I), IN, X(IN), Y(IN)
C
   40 FORMAT(' ',I3,T7,E11.5,T23,E11.5,T39,I3,T45,E11.5,T61,E11.5 )
C
      WRITE(IDEV,50)
   50 FORMAT(//' ')
C
C
      RETURN
C
      END
```

```
C*********************************************************************
C
C      PROGRAMM CUSPTEST
C
C      PROGRAM FOR TESTING THE SOFTWARE PACKAGE 'CUSPLINE'
C
       IMPLICIT REAL*8 ( A - H, O - Z )
C
       LOGICAL  DECIDE
       INTEGER  EXTIME(2), TIMEAR(8)
C
       DIMENSION  X(401), Y(401), AK(4), DY(4), D2Y(4), RVOR(4)
       DIMENSION  WK(6,401),DF(401), D2F(401), XS(501), YS(501)
       DIMENSION  COEFF(400,3), DYS(501), D2YS(501)
C
       DATA  A3, A2, A1, A0  /1.0D0, -3.0D0, -1.0D0, 1.0D0/
C
C      F(Z) = (( A3*Z + A2)*Z + A1)*Z + A0
C      F1(Z) = ( 3.0*A3*Z + 2.0*A2 )*Z + A1
C      F2(Z) = 6.0*A3*Z + 2.0*A2
C
       F(Z) = DSIN( Z*PI)
       F1(Z) = PI * DCOS( Z*PI )
       F2(Z) = -PI*PI*DSIN( PI*Z )
C
       IDEV = 35
       PI = DATAN( 1.0D0 )* 4.0D0
C
       CALL CMSCMD(' FILEDEF 35 DISK CUSPTEST RESULT A $')
C
     1 WRITE(6,1010)
  1010 FORMAT(//' **** PROGRAM CUSPTEST ****'//
      # ' PLEASE ENTER   MODE, N, NS, AI. BI, ASI AND  BSI'//)
C
       READ*, MODE, N, NS, AI, BI, ASI, BSI
C
       C0 = AI
       C1 = (BI - AI )/ DFLOAT( (N-1) )
C
       D0 = ASI
       D1 = ( BSI - ASI ) / DFLOAT(NS-1)
C
       DO 100 I = 1, N
       X(I) = C1 * DFLOAT( (I-1) ) + C0
       Y(I) = F( X(I) )
       DF(I) = F1( X(I) )
   100 D2F(I) = F2( X(I) )
C
       DO 110 I = 1, NS
   110 XS(I) = D1 * DFLOAT( I-1)  + D0
C
       WRITE(IDEV,10) A3, A2, A1 ,A0
C
    10 FORMAT(/' **** SPLINETEST ****'/' VORGEBENENE FUNKTIONSWERTE',
      # ' DER TESTFUNKTION F,'/
```

```
     # ' F(X):= ',F12.5,'*X**3 + ',F12.5,'*X**2 + ',F12.5,'*X + ',
     # F12.5,
     # //'         X            Y                  DY                ',
     #          'D2F '//)
C
      DO 50 I = 1, N
   50 WRITE(IDEV,20) I, X(I), Y(I), DF(I), D2F(I)
   20 FORMAT(' ',I4,X,F8.4, 4X,E12.5,3X,E12.5,3X,E12.5)
C
    3 WRITE( IDEV, 22) MODE, N, NS
   22 FORMAT(////' **** ERGEBNISSE DER SPLINEINTERPOLATION ****'//
     # ' MODE:= ',I3,',    N:= ',I4,',    NS:= ',I4/)
C
      CALL MTIME( TIMEAR )
C
      RVOR(1) = DF(1)
      RVOR(2) = DF(N)
C
      IF ( MODE .NE. 2 )  GOTO 2
C
      CALL CUINPO( X,Y,N, 1, AK, DY, D2Y, IER, IDEV )
C
      RVOR(1) = DY(1)
C
      CALL CUINPO( X, Y, N, N-3, AK, DY, D2Y, IER, IDEV )
C
      RVOR(2) = DY(4)
C
    2 IC = N - 1
      NWK = N
C
      CALL CUSPIN( X, Y, N, MODE, RVOR, COEFF, IC, NS, XS, YS, WK,
     # NWK, IER, IDEV )
C
      CALL MTIME( TIMEAR(5) )
C
      CALL ELACPU( TIMEAR, EXTIME )
C
      WRITE(IDEV,210) IER
  210 FORMAT( //' RETURNCODE FROM CUSPIN:'/ ' IER:= ',I5//)
C
      CPUT = DFLOAT(EXTIME(1))/1000.0
      ELAT = DFLOAT(EXTIME(2))/1000.0
C
      WRITE(IDEV, 910) CPUT, ELAT
  910 FORMAT(' USED EXECUTION TIMES BY "CUSPIN" IN MILLISECONDS:'/
     # ' CPU TIME :',F10.2,' ,  ELAPSED TIME: ',F10.2//)
C
      CALL CUSPED( X, Y, N, COEFF, IC, XS, NS, 1, NS, NS, YS, DYS,
     # D2YS, -30  )
C
      WRITE(IDEV,500)
  500 FORMAT(//' TEST DER INTERPOLATIONSWERTE UND ABLEITUNGEN AN',
     # ' DEN STUTZSTELLEN XS(I)'//)
C
```

```
C
      WRITE(IDEV,510)
  510 FORMAT(//'   J          XS(J)              YS            ( ',
     #       'F( XS(J) - YS )'/)
C
      DO 410 J = 1, NS
      DIFF = F( XS(J) ) - YS(J)
  410 WRITE(IDEV, 610)  J, XS(J), YS(J), DIFF
C
  610 FORMAT(' ',I3,4X,F14.7,5X,E15.8,5X,E15.8 )
C
      WRITE(IDEV,520)
  520 FORMAT(//' J        XS(J)          DS( XS(J) )          ',
     # '( DF - DS )'/)
C
      DO 420 J = 1, NS
      DIFF = F1( XS(J) ) - DYS(J)
  420 WRITE(IDEV, 610)  J, XS(J), DYS(J), DIFF
C
C
      WRITE(IDEV,530)
  530 FORMAT(//'   J         XS(J)          D2S( XS(J) )       ',
     #  '     ( D2F - D2S )'/)
C
      DO 430 J = 1, NS
      DIFF = F2( XS(J) ) - D2YS(J)
  430 WRITE(IDEV, 610)  J, XS(J), D2YS(J), DIFF
C
      WRITE(6,215)
  215 FORMAT(' NEUER PARAMETER MODE?'/' Y/N')
C
      IF ( .NOT. DECIDE(1)  )   GOTO 98
C
      READ*, MODE
C
      GOTO 3
C
   98 WRITE( 6, 220)
  220 FORMAT(' NEUER TEST?'/' Y/N')
C
      IF ( DECIDE(1) )    GOTO 1
C
      WRITE(6,99)
   99 FORMAT(' WUNSCHEN SIE EINEN PRINT VON DEN '/' ERGEBNISSEN',
     # ' DES SPLINETESTS?'/' Y/N'/)
C
      IF ( DECIDE(1) )  CALL CMSCMD(' XP CUSPTEST RESULT A $')
C
      STOP
C
      END
```

F(X) := 1 / (1 + X*X)



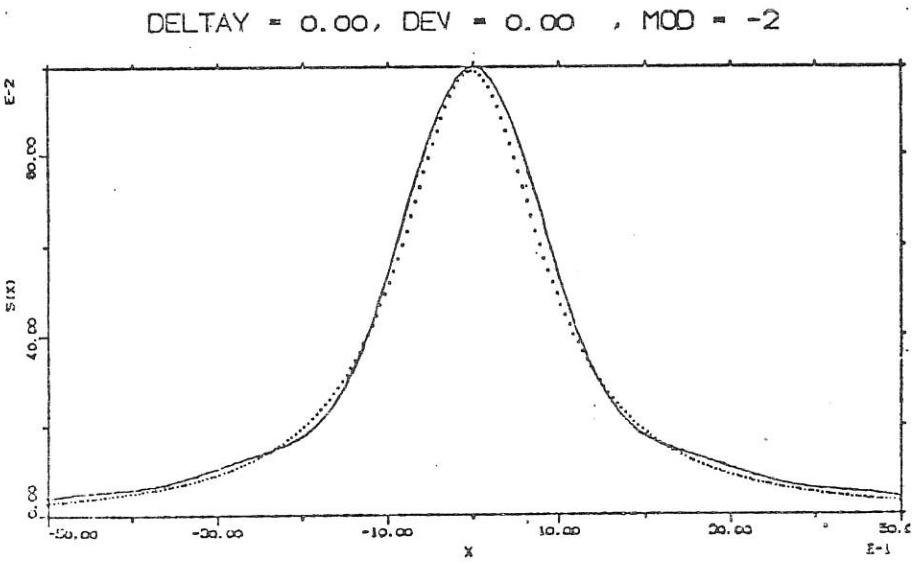


Fig.(2.3.6)

Beispiel einer Kubischen Spline-Interpolation mit Hermite- Randbedingungen. Die Randableitungen wurden durch Kubische Interpolation der Daten in 'CUINPO' berechnet.

Gezeigt wird das Konvergenzverhalten der Spline-Interpolation bei Erhöhung der Stützstellenzahl n.

Interpoliert wurde die Funktion f, $f(x) := 1/(1+x^*x)$ im Intervall $[-5, 5]$ an den n äquidistanten Stützstellen x_K, $x_K := -5 + 10*(k-1)/(n-1)$, $k := 1,\ldots,n$ mit dem in (2.3) beschriebenen Algorithmus "CUSPIN".

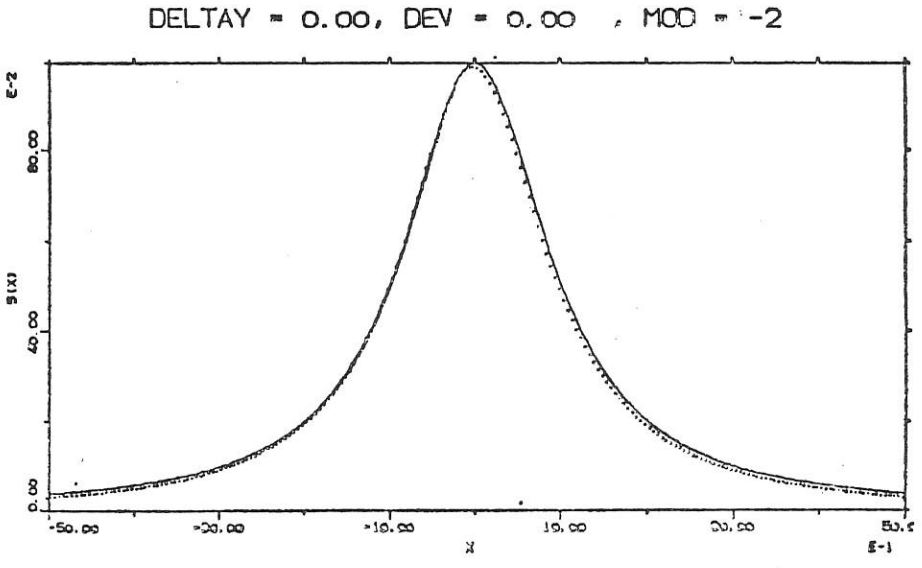
Der Wert "CPUT" bezeichnet die bei der jeweiligen Diskretisierung zur Koeffizientenberechnung und Splineentwicklung benötigte CPU-Zeit in ms.

Mit $\| s - f \|_\infty$ ist der Approximationsfehler in der Tschebyscheff-Norm (1.2.4.4) gemeint.

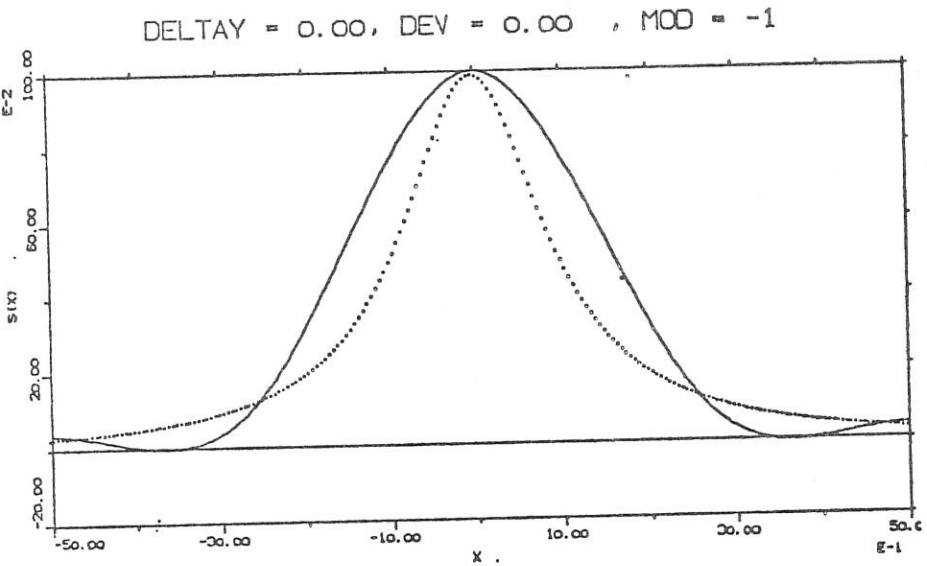
I n := 5, $\| f - s \|_\infty$ = 0.305 E 0, CPUT = 4.0 ms

II n := 21, $\| f - s \|_\infty$ = 0.317 E-2, CPUT = 5.4 ms

III n := 51, $\| f - s \|_\infty$ = 0.111 e-3, CPUT = 7.3 ms



F(X) := 1 / (1 + X*X)



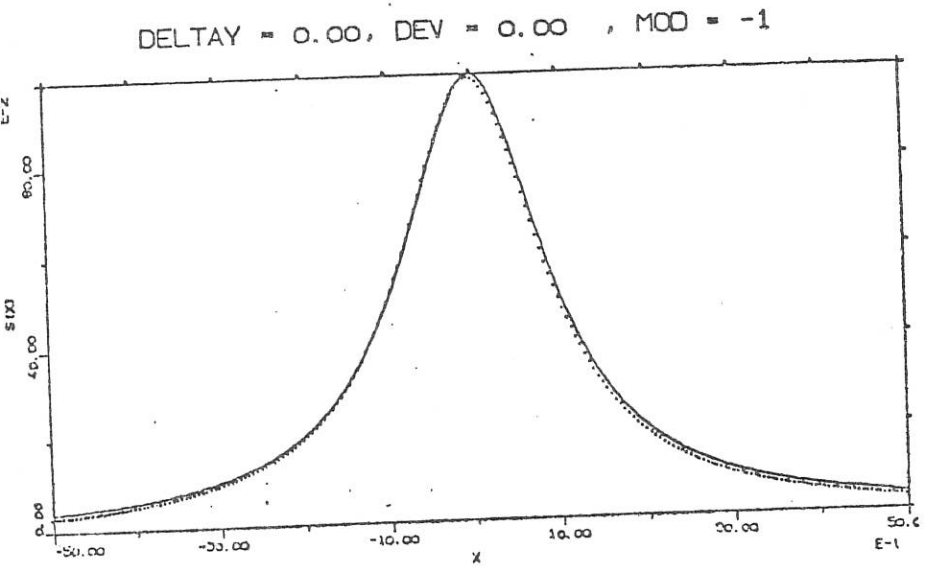


Fig.(2.3.7)

Kubische Spline-Interpolation mit Hermite-Randbedingungen. Im Gegensatz zu (2.3.7) wurden hier die exakten Randableitungen von f als Endbedingungen an den Spline s vorgegeben (MODE:= -1 in der Prozedur "CUSPIN").

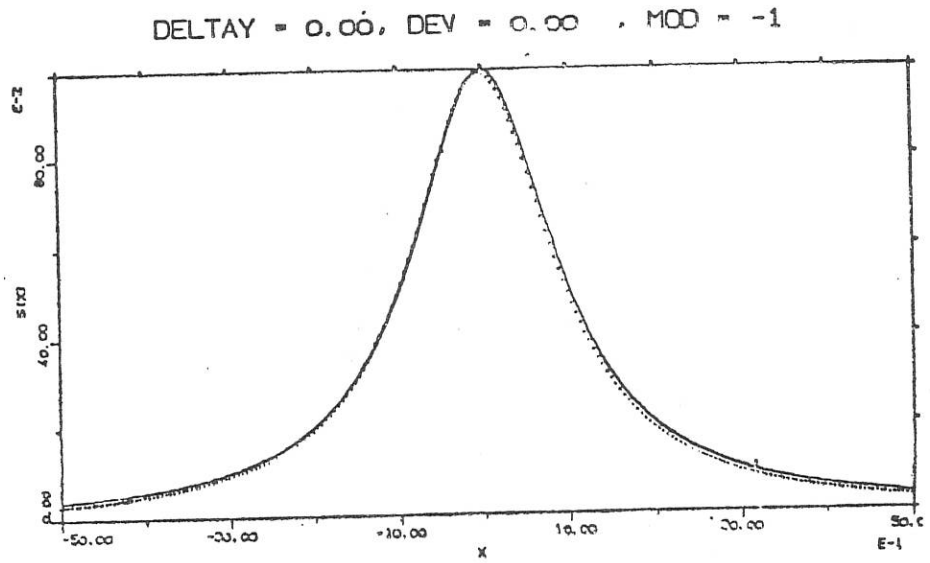
Die Anzahl und Verteilung der Stützstellen x_k, k:= 1,..,n ist identisch zu (2.3.6), ebenso die Bezeichnungen.

I- n:= 5, $\| f - s \|_\infty$ = 0.271 E 0, CPUT = 4.0 ms

Dieses Beispiel verdeutlicht den Einfluss der Randvorgaben auf die Approximationsgüte bei kleiner Stützstellenzahl n.

II- n:= 21, $\| f - s \|_\infty$ = 0.317 E-2, CPUT = 5.4 ms

III- n:= 51, $\| f - s \|_\infty$ = 0.111 E-3, CPUT = 7.3 ms



3. Ausgleichssplines

3.1 Kubische Ausgleichssplines

Im letztem Kapitel wurde eine häufig verwendete Methode zur Approximation von reellen, stetigen Funktionen f und Datenfeldern (X,Y) vorgestellt. Es handelte sich hierbei um die Interpolation mit Polynomsplines. Speziell wurde hierbei der kubische Spline $s \in S_3(Z)$ untersucht.

In diesem Kapitel möchte ich auf eine Approximationsmethode eingehen, bei der die "strenge" Interpolationsbedingung $s(x_i) := y_i$ abgeschwächt wird. Ein typisches Anwendungsgebiet dieses Verfahrens ist die Verarbeitung von Messdaten, wie das Fitten und Glätten von Datenfeldern. Experimentdaten sind im allgemeinen mit mehr oder weniger grossen zufälligen oder systematischen Fehlern behaftet. Aus diesem Grund erscheint die Bestimmung einer Näherungsfunktion g durch Interpolation der "unsicheren" Datenpunkte nicht immer sinnvoll. Relativ grosse, statistisch unregelmässige, Fehlerabweichungen der Messwerte bewirken ein starkes Oszillieren der Interpolierenden. Dieser unangenehme Effekt verschlechtert insbesonders Näherungen an die Ableitungen von f.

In seiner Arbeit [RE1] schlägt Reinsch vor, die Interpolationsvorschrift $s(x_i) := y_i$ an den kubischen Spline s zur Approximationsvorgabe

$$\sum_{i=1}^{n} \left(\frac{s(x_i) - y_i}{w_i} \right)^2 \leq S, \qquad (3.1.1.1)$$

abzuschwächen.

Mit Hilfe der Parameter w_i, $w_i > 0$, $i := 1,\ldots,n$ und $S \geq 0$ kann die gewichtete mittlere, quadratische Abweichung des Ausgleichssplines s von den einzelnen Datenpunkten gesteuert werden. Die Grösse der Gewichte w_i bestimmt somit wesentlich das Aussehen der Spline-Approximierenden s.

Für $S := 0$ erhält man die gewöhnliche Interpolationsvorgabe der Natürlichen Splines. Wählt man $S > 0$, so gibt es im allgemeinen für ein Datenfeld (X,Y) beliebig viele Funktionen $g \in C[a,b]$, welche die Approximationsbedingung (3.1.1.1) erfüllen. Zu einer eindeutigen Bestimmung müssen

daher zusätzliche Festlegungen von s erfolgen. Da der kubische Ausgleichsspline durch die Wegnahme der Interpolationsvorgabe seine Extremaleigenschaft (2.2.5) verliert, liegt es nahe, diese, für sehr viele Anwendungen, vorteilhafte Eigenheit, zu seiner eindeutigen Festlegung heranzuziehen

Wir wollen somit folgende Anforderungen an den kubischen Ausgleichsspline s stellen.

3.1.2 Definition

Sei $[a,b]$ ein reelles Intervall und Z_n, Z_n: $a =: x_1 < \ldots < x_n := b$, eine Zerlegung von $[a,b]$.

Weiter sei $W := (w_i)$, $i := 1,\ldots,n$ mit $w_i > 0$ ein reeller Gewichtsvektor und $S > 0$ eine reelle Zahl.

Eine Funktion $s:[a,b] \dashrightarrow R$ heisst "kubischer Ausgleichsspline" zu Z_n, W und S, wenn gilt:

a) $$s \in P_3|_{(x_i, x_{i+1})} \qquad (3.1.2.1)$$

b) $$s \in C^2[a,b] \qquad (3.1.2.2)$$

c) $$\sum_{i=1}^{n} \left(\frac{s(x_i) - y_i}{w_i} \right)^2 \leq S, \qquad (3.1.2.3)$$

d) $$s''(a) = 0 \quad \text{und} \quad s''(b) = 0 \qquad (3.1.2.4)$$

e) Für alle $g \in C^2[a,b]$, welche die Approximationsbedingung c) erfüllen, gilt:

$$\| s'' \|_2 \leq \| g'' \|_2 , \qquad (3.1.2.5)$$

#

Vom theoretischen Standpunkt aus ist die Frage nach Existenz und Eindeutigkeit der Lösung des Approximationsproblems (3.1.2) von besonderer Bedeutung. Diesbezüglich können wir Theorem 1 aus [RE2] heranziehen, welches unsere Frage positiv beantwortet.

3.1.3 Satz

Der kubische Ausgleichsspline s ist unter den obigen Voraussetzungen eindeutige Losung der Approximationsaufgabe (3.1.2).

#

Der Beweis zu (3.1.3) stützt sich, wie die gesamte Theorie der Allgemeinen Ausgleichssplines, auf Methoden der Variationsrechnung. Interessierte Leser dieses Problemkreises möchte ich auf die Arbeit [BÖH], Kapitel 6, hinweisen.

Aussagen über die Konvergenz des Ausgleichssplines (3.1.2) sind nur unter, in der Praxis schwer nachzuprüfenden, Zusatzvoraussetzungen möglich. Mit diesem Problem befassen sich unter anderen die Arbeiten [RIC] und [ROS]. Eine genauere Untersuchung dieses Themas würde den Rahmen dieser Arbeit bei weitem sprengen, so dass wir darauf verzichten wollen und den Leser wiederum auf die angeführte Literatur hinweisen.

3.2 Algorithmen

In diesem Kapitel werden wir uns einem Verfahren zur Berechnung von kubischen Ausgleichssplines s für ein vorgegebenes Datenfeld (X,Y) zu. Ein konkreter Algorithmus nebst aufgelisteten Algol60- Programm wird zur Bestimmung von s in [RE1] angegeben. Der Autor verwendet hierzu, wie schon im Beweis zur Existenz von s, Variationsmethoden.

Für eine vorgegebene Fehlerschranke S und Gewichte w_i, i:= 1,..,n, bildet man mit dem sogenannten "Lagrange-Multiplikator p " das Funktional $E_{p,S}$,

$$E_{p,S}(g) := \int_a^b (g''(x))^2\,dx + p * \left(\sum_{i=1}^{n} \left(\frac{g(x)-y}{w}\right)^2 + z - S\right).$$

Nach dem bekannten Satz der Variationstheorie ist das Minimum go des Funktionals $E_{p,S}$ zugleich Lösung unseres restringierten Minimalproblems (3.1.2). Damit erfüllt go die sogenannten "Eulerschen Gleichungen", die mit den, (2.3.1.) sehr ähnlichen Ansatz,

$$go(x) := co_i + c_{i1} * (x-x_i) + c_{i2} * (x-x_i)^2 + c_{i2} * (x-x_i)^2,$$

$$\text{für } x_i \leq x < x_{i+1}, \qquad i := 1,\ldots,n-1, \qquad (3.2.1.1)$$

und den Stetigkeits- und Differenzierbarkeitsbedingungen (3.1.2) ein lineares Gleichungssystem für die Unbekannten Koeffizienten c_{iK} liefern. Vor der Auflösung dieses Systems muss jedoch der unbekannte Lagrange-Multiplikator p durch Lösen eines nichtlinearen Gleichungssystems bestimmt werden. Dies geschieht im Algorithmus von Reinsch ([RE1]) durch eine Quasi-Newton-Iteration. Das nun festgelegte lineare System wird mit einer Cholesky-Zerlegung ([STO], S. 154) der auftretenden Tridiagonalmatrix gelöst. Der Arbeit [RE1] ist ein Algol60-Programm dieses Algorithmus beigefügt.

Zur Untersuchung der numerischen Eigenschaften des kubischen Ausgleichssplines wurde der Algorithmus in FORTRAN IV übertragen und anhand von konkreten Anwendungen ausgetestet.

Die Gestalt des Ausgleichsspline s hängt bei einem gegebenen Datenfeld (X,Y) wesentlich von den Steuerparametern S und w_i, i:= 1,...,n, ab. In einer geeigneten Wahl dieser Eingabewerte liegt das Problem bei der Anwendung des Approximationsalgorithmus. Eine denkbare Vorgehensweise basiert auf folgenden Überlegungen.

Man normiert zunächst die Schranke S auf n (= Anzahl der Stützstellen). Diese Festsetzung bedeutet für S keine Einschränkung, denn die transformierten Steuerparameter (k * w_i, S / k**2) führen für alle k > 0 zu denselben Interpolationsbedingungen wie die Parameter (w_i, S_i).

Bei S:= n erhalten wir nun als mittlere gewichtete quadratische Abweichung

$$\overline{\left(\frac{s(x) - y}{w}\right)^2} := 1/n * \sum_{i=1}^{n} \left(\frac{s(x_i) - y_i}{w_i}\right)^2 = 1$$

Bei der Approximation von fehlerbehafteten Daten durch Funktionswerte eines Ausgleichssplines sollte sinnvollerweise die mittlere Abweichung des Splines von den Messdaten in der Grössenordnung der angenommenen Messfehler liegen. Bei obigen Vorgehen empfiehlt sich daher als Wahl der Gewichte eine Schätzung des Messfehlers $d_i := y_i - yt_i$ von den wahren Werten yt_i in den Stützstellen x_i.

Einer dem Messvorgang entsprechenden Abschätzung des Messfehlers können die verschiedensten statistischen Modelle zugrundeliegen. In der beigefügten FORTRAN-Prozedur "CUSMCW", die von der Prozedur "CUSMSP" zur Berechnung der Gewichte w(i) aufgerufen wird, werden einige Messfehlerdarstellungen zur Bestimmung der w(i) herangezogen.

Die Auswahl des Verfahrens zur Festlegung der Gewichte in "CUSMCW" geschieht mit den Eingabeparameter 'MODE'. Zusaetzlich werden die Steuerparameter 'DELTAY' und 'DEVIAT', deren jeweilige Bedeutung von der Grösse 'MODE' abhängig ist, gebraucht.

Die Eingabe MODE:= 1 führt zur gewöhnlichen kubischen Spline-Interpolation mit Natürlichen Randvorgaben (2.2.3.2), wie sie in Kapitel 2 besprochen wurde. Die Lösung dieses Interpolationsproblems kann daher auch mit der Prozedur "CUSPIN" (MODE:= -3) bestimmt werden. Da sich die Rechenzeiten von "CUSPCO" gegenüber "CUSMCO" bei gleicher Datenmenge etwa wie 1:5 verhalten, wird in "CUSMSP" bei MODE:= 1 die Prozedur "CUSPIN" mit MODE:= -3 aufgerufen.

Die Eingabe MODE:= 2, 3, 4, 5, 6 in "CUSMSP" setzt einen Normal-bzw Gleichverteilten Messfehler d(i) der Art

d(i):= Max(DELTAY * abs($y(i)^E$) , DEVIAT), E:= 1, 2,

voraus. Entsprechend werden die Gewichte w(i) bestimmt.

Bei MODE:= 7, 8 berechnen sich die w(i) zu

w(i):= DEVY * DEVIAT + abs(y(i)^{E}) * DELTAY.

Die Grosse 'DEVY' ist hierbei die berechnete Standardabweichung der betrachteten Daten y(NISIT),..., y(NFSIT). 'DEVY' wird in "CUSMCW" durch Aufruf der Prozeduren "AVEDEV" und "SLIDEV" bestimmt. 'DELTAY' und 'DEVIAT' müssen als zusätzliche Steuerungsfaktoren eingegeben werden.

MODE:= 9, 10 sind zur eigenen Bestimmung der Gewichte w(i) frei gelassen worden. Die w(i) müssen hier als streng positive, w(i) > 0.0, Eingabeparameter vorliegen. Denkbar wäre z. B. die iterative Anwendung von "CUSMCO" mit einzelnen, nachkorrigierten Gewichten w(i) zur Einhaltung bestimmter Interpolations- oder Splinevorgaben, wie Tschebyscheff-Bedingungen oder die Vorgabe von Schranken für Ableitungswerte des Splines usw.

Bei MODE:= 10 + k, 0 < k < 21, werden die Gewichte w(i) nach der Formel

w(i):= DEVIAT * SLIDEV(i) + DELTAY * ABS(y(i)),

berechnet. Mit SLIDEV(i) wird hierbei die Gleitende Standardabweichung der 2*k + 1 Datenpunkte y(i-k),.., y(i),.., y(i+k) bezeichnet. Diese Methode eignet sich besonders für Datenfelder (X, Y), die über einzelnen Teilbereichen der Abszisse X relativ unterschiedliches Oszillationsverhalten zeigen. 'DEVIAT' und 'DELTAY' müssen wiederum als zusätzliche Steuerungsparameter beim Aufruf von "CUSMSP" oder "CUSMCW" vorliegen.

Das Programmpaket "CUSMOOTH" besteht aus den 3 FORTRAN Prozeduren "CUSMSP", "CUSMCW" und "CUSMCO" sowie aus den 3 Hilfsprozeduren "INDEXD", "AVEDEV" und "SLIDEV". Zur Auswertung des Splines werden zusätzlich die Prozeduren "CUSPEV" und "CUSPED", aus dem Programmpaket "CUBSPLIN" von Kapitel 2 benötigt.

- "CUSMSP" dient als Benutzerschnittstelle und besorgt die Parametervorgabe für die aufzurufenden Unterprogramme "CUSMCW" und "CUSMCO".

- "CUSMCW" bestimmt nach gewissen Vorgaben die in der Prozedur "CUSMCO" benötigten Gewichte w(i) zur Festlegung der Abstände des Splines von den Daten y(i) in den Stützstellen x(i).

- "CUSMCO" berechnet aus dem Datenfeld (X , Y) und den Gewichten w(i) die Koeffizientenmatrix des Ausgleichssplines s.

- "INDEXD" kann zur Bestimmung eines geeigneten Teilbereiches auf der Abszissenachse benutzt werden, falls nicht das gesamte Datenfeld (X , Y) ausgewertet werden soll.

- "AVEDEV" wird von "CUSMCW" zur Bestimmung von Durchschnittswerten der Daten y(i) aufgerufen.

- "SLIDEV" berechnet zum Datenfeld (X , Y) die Gleitenden Durchschnitte und Standardabweichungen, die in "CUSMCW" zur Bestimmung der Gewichte w(i) benötigt werden.

```
C*********************************************************************
C
      SUBROUTINE CUSMSP(X, Y, W, N, NISIT, NFSIT,ILNSIT,MODE,DELTAY,
     #                  DEVIAT, CO, COEFF, IC, NS, XS, NYS, YS, NDY,
     #                  DYSDX, ND2, D2YSD2, WK, NWK, IERROR, ILTEST)
C
C*********************************************************************
C
C     PROCEDURE: CUSMSP     VERSION: FORTRAN IV/77     DATE: 14.6.82
C
C*********************************************************************
C
C     AUTHOR: J. HELLAUER                                   PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        PROCEDURE 'CUSMSP' SMOOTHS AN ASSUMING REAL FUNCTION YY
C        OVER AN INTERVAL (. X(1) , X(N) ), WHOSE FUNCTION VALUES
C        YY(X) ARE ONLY AVAILABELY IN THE DISCRETE POINTS X(I).
C        I. E.  Y(I):= YY( X(I) ), FOR I = 1..N.
C
C        THE SMOOTHING FUNCTION SP TO BE CONSTRUCTED SHOULD MINIMIZE
C
C                          INTEGRAL( (G''(X))**2 DX )
C
C        AMONG ALL TWICE DIFFERENTABLE FUNCTIONS G ON THE GIVEN
C        RANGE  (. X(NISIT) , X(NFSIT) ), 1 <= NISIT < NFSIT <= N,
C        SUCH THAT
C
C              SUM( ( ( SP(X(I)) - Y(I) ) / W(I) )**2 ) <= S.
C
C        BY INPUT PARAMETERS "MODE", "DELTAY" AND "DEVIAT", YOU CAN
C        CONTROL THE 'SMOOTHNESS' OF SP AND ITS DISTANCE COMPARED
C        WITH GIVEN DATA VALUES Y(I) AT THE GIVEN POINTS X(I).
C
C        THE SOLUTION OF THIS MINIMUM PROBLEM IS OBTAINED BY USING
C        CALCULUS OF VARIATION.
C
C        FOR AN OPTIMAL INTEGRATION OF SMOOTHING TO PHYSICAL OC-
C        CURENCES STATISTICAL METHODS MUST BE INCLUDED. ACCORDING TO
C        THE PARTICULAR MEASURING METHODOLOGY VARIED PRINCIPLES FOR
C        DETERMINATION ARE SELECTABLE. THE INPUT PARAMETER 'MODE'
C        FOR THE NEEDED PROCEDURE 'CUSMCW' CONTROLS THIS SELECTION.
C
C        MORE INFORMATION ABOUT THIS PROBLEM YOU CAN GET FROM
C        COMMENTS OF ADDED PROCEDURES 'CUSMCW' AND 'CUSMCO' AND
C        OF COURSE FROM SUPPLIED BIBLIOGRAPHY.
C
C        ALMOST ALL INPUT PARAMETERS ARE USED UNALTERED IN THE
C        CALLED PROCEDURES 'CUSMCW', 'CUSMCO' AND 'CUSPED'. SO YOU
C        CAN SEE THE OPERATING INSTRUCTION OF IT IN RESPECTIVE
C        COMMENTS.
C
```

```
C***********************************************************************
C
C      BIBLIOGRAPHY:
C
C         CHRISTIAN H. REINSCH..: SMOOTHING BY SPLINE FUNCTIONS I.
C                                 NUMER. MATHEMATIK 10,
C                                 177 - 183 (1967)
C
C         --------------------..: SMOOTHING BY SPLINE FUNCTIONS II.
C                                 NUMER. MATHEMATIK 16,
C                                 451 - 454 (1971)
C
C
C***********************************************************************
C
C      LINKAGE:
C
C         CALL CUSMSP( X, Y, W, N, NISIT, NFSIT,ILNSIT, MODE, DELTAY,
C                      DEVIAT, CO, COEFF, IC, NS, XS, NYS, YS, NDY,
C                      DYSDX, ND2, D2YSD2, WK, NWK, IERROR, ILTEST )
C
C-----------------------------------------------------------------------
C
C      INPUT:
C
C         X.....: REAL*8,  DIMENSION X(N),
C                 ABSCISSAE VECTOR OF DATA FIELD ( X, Y).
C                 MUST BE ORDERED SO THAT  X(1) < X(2) < ... < X(N).
C
C
C         Y.....: REAL*8,  DIMENSION Y(N),
C                 ORDINATE VECTOR, GIVEN DATA FOR SMOOTHING
C
C
C         W.....: REAL*8, DIMENSION W(N),
C                 WEIGHTS, WHICH CONTROL THE EXTENT OF SMOOTHING
C                 SPLINE FROM THE ORIGINAL DATA Y.
C                 FOR MODE # 9  OR  MODE # 10  THE WEIGHTS ARE
C                 CALCULATED IN THE PROCEDURE "CUSMCW", WHICH
C                 IS CALLED IN THIS PROCEDURE.
C                 FOR  MODE:= 9  OR  MODE:= 10  YOU HAVE TO DETERMINE
C                 W(I) > 0.0  AUTONOMOUSLY.
C
C
C         N.....: INTEGER,
C                 DIMENSION OF  X, Y AND W.
C
C
C         NISIT,
C         NFSIT.: INTEGER,
C                 FIRST AND LAST INDEX OF X, WHOSE DATA
C                 Y(NISIT),...,Y(NFSIT) SHOULD BE CONSIDERED FOR
C                 SMOOTHING.
C                 IF NYS:= NS  OR  NDY:= NS  OR ND2:= NS  AND
C                 ILNSIT NOT EQUAL 1, NISIT AND NFSIT WILL BE
```

```
C               CALCULATED SUITABLE BY PROCEDURE 'CUSMSP'.
C
C
C       ILNSIT: INTEGER,
C               ILNSIT:= 1,  EFFECTS:
C               THE SUITABLE DATA RANGE  (. X(NISIT) , X(NFSIT) .)
C               FOR EVALUATION THE SPLINE IN XS(I)), I = 1,..,NS IS
C               DETERMINED BY THE GIVEN INDICES PARAMETER 'NISIT'
C               AND 'NFSIT'. ONLY IF THESE TWO VALUES ARE WRONG
C               PLACED, PROPER INDICES ARE CHOOSEN BY 'CUSMSP'
C               ITSELF.
C
C               ILNSIT:= -1,  SHOULD BE USED,
C               IF  NYS:= NS  OR  NDY:= NS  OR  NDY:= NS  AND IF
C               XS(I):= X(NISIT-1+I), FOR I:= 1,..,NS.
C               SO A SHORTER WAY TO EVALUATE THE SPLINE VALUES
C               SP( XS(I) ), I:= NISIT, NFSIT  IS DONE WITHIN THE
C               PROCEDURES.
C
C               ILNSIT:= K, K ANY OTHER INTEGER:
C               'NISIT' AND 'NFSIT' ARE SELECTED SUITABLE BY
C               'CUSMSP' FOR EVALUATION OF YS, DYSDX OR D2YSD2.
C
C
C       MODE..: INTEGER, 1 <= MODE <= 30,
C               SWITCH FOR DESTINATE THE WEIGHTS IN "CUSMCW".
C               SEE PROCEDURE 'CUSMCW'.
C
C
C       DELTAY: REAL*8,  0 <= DELTAY,
C               RELATIVE ERROR OF MEASUREMENT.
C               SEE: PROCEDURE 'CUSMCW'.
C
C
C       DEVIAT: REAL*8, 0 <= DEVIAT,
C               DEVIATION OF ERROR OF MEASUREMENT.
C               SEE: PROCEDURE 'CUSMCW'.
C
C
C       NS....: INTEGER, LENGHT OF XS.
C
C
C       XS....: REAL*8, DIMENSION XS(NS),
C               ABSCISSAE VECTOR FOR EVALUATING SPLINE VALUES YS,
C               VALUES YS, DYSDX AND D2YSD2.
C               SEE: PROCEDURE 'CUSPED'.
C
C
C       NYS...: INTEGER, DIMENSION OF YS,
C               NYS:= NS, MEANS: YS SHOULD BE CALCULATED.
C               OTHERWISE  NYS:= 1  MUST BE SET.
C
C
C       NDY...: INTEGER, DIMENSION OF DYSDX,
C               NDY:= NS, MEANS: DYSDX ( I. E. VALUES OF THE FIRST
```

```
C                DERIVATION OF SPLINE SP AT XS SHOULD BE EVALUATED.
C                OTHERWISE  NDY:= 1  MUST BE SET.
C
C
C         ND2...: INTEGER, DIMENSION OF D2YSDX,
C                 DERIVATION OF SPLINE SP AT XS SHOULD BE EVALUATED.
C                 OTHERWISE  ND2:= 1  MUST BE SET.
C
C
C         WK....: REAL*8, DIMENSION (7,NWK),
C                 WORKING AREA,  NWK:= N + 2.
C
C
C         NWK...: INTEGER, LENGHT OF WK,
C                 NWK MUST BE SET AS
C                 NWK:= N + 2.
C
C
C         ILTEST: INTEGER,
C                 UNIT NUMBER OF OUTPUT DEVICE FOR TESTING WEIGHTS
C                 W  AND  DATA Y  IN  PROCEDURE 'CUSMCW'
C
C                 ILTEST < 6,  NO TEST IS DONE,
C
C                 ILTEST:= 6,  Y AND W WILL BE PRINTED
C                              ON YOUR TERMINAL.
C
C                 7 <= ILTEST <= 99,  Y AND W WILL BE PRINTED IN THE
C                                     CMS FILE 'WEIGHTS TEST A'.
C
C
C*****************************************************************
C
C     OUTPUT:
C
C         CO....: REAL*8,  WITH LENGHT N,
C                 FUNCTION VALUES OF THE SMOOTHING SPLINE
C                 IN THE ABSCISSAE X(I), I. E.
C                 CO(I):= SP( X(I) ).
C
C
C         COEFF.: REAL*8,  DIMENSION  COEFF(IC,3)
C                 COEFFICIENT MATRIX FOR SPLINE EVALUATION.
C                 CALCULATION OF IT IS DONE IN PROCEDURE 'CUSMCO'.
C
C                 WITH CO AND COEFF, YOU CAN EVALUATE EVERY FUNCTION
C                 VALUE SP(Z) OF SPLINE SP FOR ABSCISSAE Z, WITH
C                 X(I) <= Z < X(I+1)  AND  NISIT <= I <= NFSIT.
C                 FOR  H:= Z- X(I)  IS
C                                          SP( Z ) =  CO +
C                 + (( COEFF(I,3)*H + COEFF(I,2) )*H + COEFF(I,1) )*H
C
C                 THIS EVALUATION IS DONE IN 'CUSPED'.
C
C
```

```
C       YS....: REAL*8, DIMENSION  YS(NYS),
C               IF  NYS IS SET TO NS, YS CONSISTS AFTER RETURN OF
C               the VALUES OF SMOOTHING SPLINE SP IN THE POINTS XS
C               I. E.  YS(I):= SP( XS(I) ), FOR  1 <= I <= NYS.
C               SEE PROCEDURE 'CUSPED'.
C
C
C       DYSDX.: REAL*8, DIMENSION  DYSDX(NDY),
C               IF NDY IS SET TO NS, DYSDX CONSISTS AFTER RETURN OF
C               the VALUES OF THE FIRST DEVIATION OF SP IN XS(I).
C               I. E. DYSDX(I):= D(SP(X(I))/DX, FOR 1 <= I <= NDY.
C
C
C       D2YSD2: REAL*8, DIMENSION  DYSDX(ND2),
C               IF ND2 IS SET TO NS, D2YSD2 CONSISTS AFTER RETURN
C               of the VALUES OF THE FIRST DEVIATION OF SP IN XS.
C               I. E. D2YSD2(I):= D2(SP(X(I))/DX2, I = 1,...,ND2.
C
C
C       IERROR: INTEGER,
C
C               := 0,      NO ERROR DETECTED,
C
C
C                < -900,   ERROR MESSAGES ( NO CALCULATION IS DONE )
C
C               := -996,   INPUT ABSCISSAE ARE NOT ORDERED SO THAT
C                          X(1) < X(2) < ... X(N).
C
C               := -997,   IC IS NOT N-1.
C
C               := -998,   NISIT < 1       OR
C                          NFSIT > N       OR
C                          NISIT => NFSIT
C
C               := -999,   EVALUATION OF YS, DYSDX OR D2YSD2
C                          IS IMPOSSIBLE, BECAUSE THE DATA RANGE
C                          (. X(1) , X(N) ) IS NOT ADAPTED TO XS.
C                          I. E. THE RANGE (. XS(1) , XS(NS) .)
C                          DOESN'T LIE COMPLETELY WITHIN THE
C                          RANGE (. X(1) , X(N) .).
C
C
C                > -100,   WARNINGS (SPLINE CALCULATION IS DONE)
C
C               :=  -1,    EVALUATION IS CORRECTLY EXECUTED, BUT THE
C                          GIVEN PARAMETER 'NISIT' MAY BE CHANGED.
C
C               :=  -2,    EVALUATION IS CORRECTLY EXECUTED, BUT THE
C                          GIVEN PARAMETER 'NFSIT' MAY BE CHANGED.
C
C               :=  -3,    INCLUDES THE CASES  IERROR:= - 1 AND
C                          IERROR:= -2.
C
```

```
C
C*********************************************************************
C
C      OPERATING SYSTEM:  VM/370 CMS
C
C*********************************************************************
C
C      EXTERNAL PROCEDURES:
C
C         CUSMCO  ( FOR CALCULATING "CO" AND "COEFF" )
C         CUSMCW  ( FOR CALCULATING THE WEIGHTS "W" )
C         INDEXD  ( FOR SEARCHING A SUITABLE RANGE  )
C         CUSPED  ( FOR SPLINE EVALUATION )
C
C*********************************************************************
C
C      EXTERNAL FILES:          NONE
C
C*********************************************************************
C
C
      IMPLICIT  REAL*8 ( A - H, O - Z )
C
      INTEGER  ILNSIT, NYS, NDY, ND2, ILTEST, IERROR
C
      DIMENSION  X(N), Y(N), W(N), CO(N), COEFF(IC,3), WK(7,NWK)
      DIMENSION  XS(NS), YS(NYS), DYSDX(NDY), D2YSD2(ND2)
C
      DATA  EPS / 1.0E-8 /
C
C     CHECK, IF EVALUATING IS POSSIBLE, I. E. THE NEEDED RANGE
C     IS SUITABLE.
C
      IF ( NS  .NE. NFSIT-NISIT+1 )   ILNSIT = IABS(ILNSIT)
C
      IERROR = -997
C
      IF ( IC .NE. N - 1 )  RETURN
C
      IERROR = -998
C
      IF (NISIT .LE. 0  .OR.  NFSIT  .GT. N  .OR.  NISIT .GE. NFSIT)
     # RETURN
C
      IERROR = 0
C
      IF ( NYS .LE. 1  .AND.  NDY .LE. 1  .AND.  ND2 .LE. 1 )
     # ILNSIT = -1
C
      IF ( ILNSIT .EQ. -1  )   GOTO 1000

  100 IERROR = -999
C
      IF ( XS(1) .LT. ( X(1)-EPS )  .OR.  XS(NS) .GT. ( X(N)+EPS ) )
     #RETURN
```

```
C
      IERROR = 0
C
C     SEARCH FOR THE NEEDED RANGE (. X(NISIT) , X(NFSIT) ).
C
      CALL INDEXD( X, N, XS(1), XS(NS), N1, N2 )
C
      IF ( ILNSIT .EQ. 1  .AND.  NISIT .LE.  N1  )  GOTO 500
C
C     THE GIVEN LEFT CUTOFF POINT X(NISIT) MUST BE CORRECTED.
C
      NISIT = MAX( 1, N1 )
      IERROR = -1
C
  500 CONTINUE
C
      IF  ( ILNSIT .EQ. 1  .AND.  NFSIT .GE. N2 )  GOTO 1000
C
C     THE GIVEN RIGHT CUTOFF POINT X(NFSIT) MUST BE CORRECTED.
C
      NFSIT = MIN ( N, N2 )
      IERROR = IERROR - 2
C
C
 1000 CONTINUE
C
C     THE PROPER RANGE IS DETERMINED.
C
      LTEST = ( ILTEST .GE. 6 )
C
C     FOR  MODE:= 1, USUAL SPLINE INTERPOLATION WITH NATURAL
C     BOUNDARY CONDITIONS, IT IS BETTER TO CALL "CUSPIN".
C
      IF ( MODE .EQ. 1  )     GOTO 1300
C
C
C     PROCEDURE 'CUSMCW' CALCULATES THE WEIGHT VECTOR 'W'
C
      CALL CUSMCW( Y, N, NISIT, NFSIT, MODE, DEVIAT, DELTAY,
     #             S, W, ILTEST )
C
C     PROCEDURE 'CUSMCO' CALCULATES THE COEFFICIENT MATRIX 'COEFF'.
C
      CALL CUSMCO( X, Y, W, N, NISIT, NFSIT, S, CO, COEFF, IC, IER)
C
      GOTO 1400
C
C
 1300 MODE = -3
C
      CALL CUSPIN( X, Y, N, MODE, W, COEFF, IC, 1, XS, YS, WK, 1,
     #             IER, ILTEST )
C
C
C     SPLINE EVALUATION
```

```
C
 1400 IF ( ILNSIT .EQ. -1 )          GOTO 2000
C
      IF ( NYS .LT. NS  .OR.  NDY .LT. NS  .OR.  ND2 .LT. NS   )
     # GOTO 1800
C
      CALL CUSPED( X, CO, N, COEFF, IC, XS, NS, NYS, NDY, ND2, YS,
     #             DYSDX, D2YSD2, IER, ILTEST)
C
      RETURN
C
C
 1800 IF ( NYS .EQ. NS )
     # CALL CUSPEV( X, CO, N, COEFF, IC, XS, YS, NS, IER, ILTEST )
C
      RETURN
C
C
C     A SHORT WAY TO EVALUATE YS(I)
C
 2000  IND = NISIT - 1
C
C
      IF ( NYS .LT. NS  )    GOTO  2200
C
      DO 2110 I = NISIT, NFSIT
 2110 YS(I-IND) = CO(I)
C
 2200 IF ( NDY .LT. NS )        GOTO 2300
C
      DO 2210 I = NISIT, NFSIT
 2210 DYSDX(I-IND) = COEFF(I,1)
C
 2300 IF ( ND2 .LT. NS )          RETURN
C
      DO 2310 I = NISIT, NFSIT
 2310 D2YSD2(I-IND) = 2.0*COEFF(I,2)
C
C
      RETURN
C
C     END OF CUSMSP
C
      END
C*****************************************************************************
C
```

```
C*********************************************************************
C
      SUBROUTINE CUSMCW( Y, N, NISIT, NFSIT, MODE, DEVIAT, DELTAY,
     #                   S, W, ILTEST )
C
C*********************************************************************
C
C     PROCEDURE: CUSMCW    VERSION: FORTRAN IV/77       DATE: 1.6.82
C
C*********************************************************************
C
C     AUTHOR: J. HELLAUER                                  PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        CALCULATES THE WEIGHTS W(I), WHICH CONTROL THE EXTENT OF
C        SMOOTHING SPLINE FROM ORIGINAL DATA FIELD Y. THE WEIGHTS
C        COEFFICIENTS  W(I)  ARE USED IN PROCEDURE 'CUSMSP' FOR
C        DETERMINATION OF SMOOTHING SPLINE SP.
C
C        LIKE ILLUSTRATED IN COMMENT OF PROCEDURE 'CUSMSP', THE
C        SMOOTHING SPLINE SP IS PLOTTED BY RESTRICTION
C            SUM ( ( SP( X(I) ) - Y(I) ) / W(I) )**2 )  <=  S
C        AND CERTAIN REQUIREMENT FOR ITS SMOOTHING AND CURVATURE.
C
C        FOR AN OPTIMAL INTEGRATION OF SMOOTHING TO THE PHYSICAL
C        OCCURENCES STATISTICAL METHODS MUST BE INCLUDED.
C        ACCORDING TO THE PARTICULAR MEASURING METHODOLOGY VARIED
C        PRINCIPLES OF CALCULATION W AND S ARE SELECTABLE.
C        THE INPUT PARAMETER 'MODE' FOR PROCEDURE 'CUSMCW' CONTROLS
C        THIS SELECTION.
C
C*********************************************************************
C
C     BIBLIOGRAPHY:
C
C        CHRISTIAN H. REINSCH: SMOOTHING BY SPLINE FUNCTION I,
C                              NUMERISCHE MATHEMATIK 10,
C                              177 - 183 (1967)
C
C*********************************************************************
C
C     LINKAGE:
C
C        CALL CUSMCW ( Y, N, NISIT, NFSIT, MODE, DEVIAT, DELTAY,
C                                   S, W, ILTEST )
C
C---------------------------------------------------------------------
C
C     INPUT:
C
C        Y     : REAL*8, DIMENSION Y(N),
C                DATA FIELD, FOR WHICH W SHOULD BE SPECIFIED
```

```
C
C       N      : INTEGER,
C                DIMENSION OF Y AND W
C
C       NISIT,
C       NFSIT  : INTEGER, WHERE  1 <= NISIT < NFSIT <= N,^
C                FIRST RESPECTIVELY LAST INDEX OF THE SUBDIVISION ON
C                Y, WHOSE DATA SHOULD CONTROL THE CALCULATION OF W.
C                ONLY THE DATA  Y(NISIT),..,Y(NFSIT)  ARE INCLUDED
C                FOR DETERMINATION OF  W(NISIT),...,W(NFSIT).
C
C
C
C       MODE :   INTEGER, 1 <= MODE <= 30,
C                SWITCH FOR SELECTION OF PROCEDURE FOR CALCULATION
C                OF THE WEIGHTS W.
C
C
C                MODE:= 1, USUAL SPLINE-INTERPOLATION OF DATA, I. E.
C                          THE SMOOTHING SPLINE PASSES THE POINTS
C                          Y(I) SUITABLE METHODE FOR DATA Y, WHICH
C                          ARE ASUMED WITHOUT ANY ERROR OF MEASURE-
C                          MENT.
C
C
C                MODE:= 2, SMOOTHING SPLINE,
C                          THE ERROR OF MEASUREMENT FOR ALL ORDI-
C                          NATES IS ASSUMED AS CONSTANT EQUAL DEVIAT
C                          IN BOTH DIRECTIONS. IN THIS CASE WEIGHTS
C                          W ARE CHOOSEN AS
C                                  W(I):=  DEVIAT / SQRT(3.0)
C                          ( STANDARD DEVIATION OF UNIFORM
C                          DISTRIBUTED DATA ).
C
C
C                MODE:= 3, SMOOTHING SPLINE,
C                          THE ERROR OF MEASUREMENT IS ASSUMED
C                          LINEAR RESPECTIVLY THE VALUES OF Y(I).
C                          DEVIAT IS THE LOWER BOUND OF DEVIATION,
C                          I. E.  W(I):=
C                          := MAX(ABS(Y(I))*DELTAY,DEVIAT)/SQRT(3),
C
C                          DELTAY MEANS HERE THE RELATIVE ERROR OF
C                          MEASUREMENT OF THE ORDINATE Y(I).
C
C
C                MODE:= 4, SMOOTHING SPLINE,
C                          THE ERROR OF MEASURE IS ASSUMED AS NORMAL
C                          N(0,DEVIAT)-DISTRIBUTED, I. E. MEAN:= 0,
C                          STANDARD DEVIATION = DEVIAT.
C                          LIKE OF MODE:= 2  WEIGHTS W(I) ARE
C                          CHOOSEN AS  W(I):= DEVIAT.
C
C
C                MODE:= 5; SMOOTHING SPLINE,                    ^
```

```
C                    LIKE  MODE:= 3, BUT ERROR OF MEASUREMENT
C                    IS ASSUMED AS  N(0, Y(I)*DELTAY) - NORMAL
C                    DISTIBUTED WITH THE LOWER ERROR BOUND
C                    DEVIAT I. E.
C                    W(I):= MAX( ABS(Y(I)) * DELTAY , DEVIAT )
C
C
C               MODE =:6, SMOOTHING SPLINE,
C                    THE ERROR OF MEASUREMENT IS HERE ASSUMED
C                    AS NORMAL DISTRIBUTED WITH DEVIATION
C                    DELTAY * SQRT( Y(I) ).
C                    THIS TYPE OF DISTRIBUTION IS
C                    ACCORDING TO STASTICAL TEST DATA, WHICH
C                    ARE RECEIVED AS A SAMPLE RATE LIKE BY
C                    COUNTER OF PHOTONS.
C
C                    THE SMOOTHING OF THE SPLINE CAN BE
C                    CORRECTED SIMPLE BY CHOOSING A SUITABLE
C                    DELTA IN THE RANGE  0.0 <= DELTAY <= 1.0.
C                    THE SMALLER DELTAY THE CLOSELIER TO THE
C                    DATA SMOOTHING OCCURES.
C
C
C               MODE =7:  SMOOTHING SPLINE,
C                    SPEZIAL FEATURE FOR EXPERIMENTAL DATA,
C                    WHOSE ERROR OF MEASUREMENT W(I) ARE
C                    ASSUMED AS CONSISTING OF TWO INDEPENDENT
C                    COMPONENTS, A LINEAR AND A CONSTANT ONE,
C                    I. E.
C                    W(I):= DEVY*DEVIAT + ABS(Y(I)) * DELTAY,
C                    WHERE  DEVY  IS THE AUTOMATICALLY
C                    CALCULATED STANDARD DEVIATION OF THE
C                    DATA Y(I), I:= NISIT,...,NFSIT.
C                    DEVIAT AND DELTAY HAVE TO BE GIVEN AS
C                    POSITIVE CONTROL PARAMETERS, THEY
C                    REGULATE SMOOTHING.
C
C                    DEVIAT  ACCENTUATES THE CONSTANT
C                    COMPONENT AND SHOULD BE CHOOSEN AS
C                    0.0 <= DEVIAT<= 1.0.
C                    FOR  DEVIAT:= 0.0  ONLY A LINEAR
C                    COMPONENT IS ASSUMED.
C
C                    DELTAY CONTROLS THE LINEAR COMPONENT
C                    OF W AND SHOULD BECOME A VALUE
C                    BETWEEN  0.0  AND  0.2.
C                    DELTAY:= 0.0  EFFECTS
C                    CONSTANT WEIGHTS W(I),
C                    W(I):= DEVY * DEVIAT, FOR ALL I.
C
C
C               MODE = 8, SMOOTHING SPLINE,
C                    ALMOST THE SAME AS MODE:= 7, BUT THE
C                    WEIGHTS W(I) CONSIST OF A SUBLINEAR
C                    COMPONENT INSTEAD OF A LINEAR ONE.
```

```
C                          I.E.  W(I):=
C                          := DEVY*DEVIAT + ABS(SQRT(Y(I)))*DELTAY.
C
C                          THIS MODE IS MORE SUITABLE FOR
C                          STATISTICAL DATA, SEE ALSO MODE:= 6.
C
C
C                MODE:= 9, 10,
C                          THESE MODES ARE LEAVED BLANK FOR SPECIAL
C                          REQUIREMENTS OF DATA SMOOTHING. IT CAN
C                          BE USED TO DETERMINE SINGULAR W(I) OUT OF
C                          RULE, FOR EXAMPLE IN A SPECIAL PROCEDURE.
C
C                          THE WEIGHTS W(I) MUST BE DESTINATED
C                          BEFORE CALL OF "CUSMCW".
C
C
C                MODE:= NN,  11 <= NN <= 30,
C                          SIMILAR TO  MODE:= 7  BUT THE STANDARD
C                          DEVIATION  DEVY  IS HERE CALCULATED AS A
C                          GLIDING AVERAGE DEVIATION OF THE DATA IN
C                          NEIGHBORHOOD OF X(I) ON THE TIMESCALE X.
C
C                          BE  NP:= NN - 10, THEN FOR A FIXED I, I:=
C                          NISIT,...,NFSIT, THE LOCAL DEVIATION D(I)
C                          IS CALCULATED AS THE STANDARD DEVIATION
C                          OF THE 2*NP+1 DATA Y(K), K:=I-NP,..,I+NP.
C
C                          THIS METHODE OF DETERMINATION OF THE
C                          WEIGHTS W(I) IS ESPECIALLY SUITABLE
C                          FOR DATA Y, WHOSE ERROR OF MEASUREMENT IS
C                          MORE OR LESS DIVERSIFIED IN CONTIGIOUS
C                          REGIONS OF THE TIME SCALE.
C
C
C        DEVIAT,
C        DELTAY: REAL*8,
C                SEE THE MEANINGS OF THESE TWO PARAMETERS IN THE
C                COMMENT OF CORRESPONDING MODE.
C                DEVIAT AND DELTAY MUST NOT BE SET LESS OR EQUAL
C                ZERO IN SOME DISTINCT MODES.
C
C        ILTEST: INTEGER, UNIT-NUMBER OF OUTPUT DEVICE,
C
C                :< 6,      NO TEST OUTPUT,
C                := 6,      OUTPUT AT TERMINAL,
C                :> 6,      OUTPUT IN CMS-DISK-FILE 'FILE FT18F001 A'
C
C
C-----------------------------------------------------------------------
C
C     OUTPUT:
C
C        S      : REAL*8,
C                 RESTRICTION PARAMETER OF SMOOTHING. IT IS NEEDED
```

```
C                IN THE PROCEDURE 'CUSMSP' FOR CALCULATING SPLINE
C                COEFFICIENTS. FOR ITS MODE OF PERFORMANCE SEE
C                THE REMARKS BY 'FUNCTION'.
C                EXCEPTING AT MODE:= 1  S IS ALWAYS DETERMINED AS
C                          S:= NFSIT - NISIT + 1
C                THIS IS THE NUMBER OF DATA POINTS, WHICH SHOULD
C                CONTROL THE SMOOTHING
C
C
C        W     : REAL*8, DIMENSION  W(N),
C                CALCULATED WEIGHTS NEEDED IN PROCEDURE 'CUSMSP'.
C                FOR IT MODE OF PERFORMANCE SEE ALSO THE REMARKS
C                BY 'FUNCTION' AND 'MODE' OR THE COMMENT OF
C                OF PROCEDURE 'CUSMSP'.
C
C
C*****************************************************************
C
C     OPERATING SYSTEM:          VM/370 - CMS
C
C*****************************************************************
C
C     EXTERNAL PROCEDURES:       AVEDEV  (FORTRAN)
C                                SLIDEV  (FORTRAN)
C
C*****************************************************************
C
C     EXTERNAL FILES:  CMS DISK FILE 'FILE FT18F001 A'
C
C*****************************************************************
C
C
      IMPLICIT  REAL*8 ( A - H, O - Z )
C
      DIMENSION Y(N), W(N)
C
      LOGICAL LTEST
C
      DATA  DEVFAC /1.003/, NSTEP /1/
C
      LTEST = ( ILTEST .GE. 6 )
C
      DO 1 I = 1, N
    1 W(I) = 1.0
C
      NDIFF = NFSIT - NISIT + 1
      S = DFLOAT( NDIFF )
C
C     CALL OF AVEDEV FOR CALCULATION OF
C
C     DEVY:=  STANDARD DEVIATION OF DATA  Y(NISIT),..,Y(NFSIT)  AND
C     AVERY:= AVERAGE OF DATA  Y(NISIT),...,Y(NFSIT).
C
C
      CALL AVEDEV( Y, N, NISIT, NFSIT, NTRUE, DEVY, AVERY, IERROR)
```

```
C
      DEVMIN = DMAX1( DEVY/DEVFAC , 1.0D-11 )
C
      IF ( MODE .GE. 11 )     GOTO  11000
C
      GOTO (1000,2000,3000,4000,5000,6000,7000,8000,9000,10000),
     #   MODE
C
C
 1000 CONTINUE
C
C     MODE:= 1,  USUAL INTERPOLATION OF DATA WITH CUBIC SPLINES
C
      S = 0.0
C
      GOTO 11111
C
C
 2000 CONTINUE
C
C     MODE:= 2, UNIFORM DISTRIBUTED CONSTANT ERROR OF MEASUREMENT
C               DEVIAT IS ASSUMED. DEVIAT IS TO BE GIVEN BY INPUT.
C
      SQ = DEVIAT / DSQRT( 3.0D0 )
C
      DO 2100 I = NISIT, NFSIT
 2100 W(I) = SQ
C
      GOTO 11111
C
C
 3000 CONTINUE
C
C     MODE:= 3, LINEAR ERROR OF MEASUREMENT RELATIVE  ABS( Y(I) )
C               WITH LOWER BOUND DEVIAT IS ASSUMED.
C               DELTAY AND DEVIAT HAVE TO BE AVAILABLE BY INPUT.
C
      SQ3 = 1.0 / DSQRT( 3.0D0 )
      DEVIAT = DMAX1( DEVIAT , DEVMIN )
C
      DO 3100 I = NISIT, NFSIT
      ERR = DABS( Y(I) ) * DELTAY
 3100 W(I) = DMAX1( ERR , DEVIAT ) * SQ3
C
      GOTO 11111
C
C
 4000 CONTINUE
C
C     MODE:= 4, NORMAL DISIRIBUTED ERRORS WITH ASSUMED CONSTANT
C               DEVIATION DEVIAT, WHICH MUST BE GIVEN BY INPUT.
C
      DEVIAT = DMAX1( DEVIAT , DEVMIN )
C
      DO 4100  I = NISIT, NFSIT
```

```
 4100 W(I) = DEVIAT
C
      GOTO 11111
C
C
 5000 CONTINUE
C
C     MODE:= 5, NORMAL DISTRIBUTED ERROR WITH DEVIATION
C               DELTAY * Y(I).
C               THE ABSOLUTE LOWER BOUND OF DEVIATION, HERE DEVIAT,
C             . MUST BE GIVEN BY INPUT.
C
      DEVIAT = DMAX1( DEVIAT , DEVMIN )

      DO 5100 I = NISIT, NFSIT
      ERR = DABS( Y(I) ) * DELTAY
 5100 W(I) = DMAX1( ERR, DEVIAT )
C
      GOTO 11111
C
C
 6000 CONTINUE
C
C     MODE:= 6,  LIKE MODE:= 5, BUT THE ERROR IS ASSUMED RELATIVE
C                TO THE SQUAREROOT OF Y(I).
C
      DEVIAT = DMAX1( DEVIAT , DEVMIN )
C
      DO 6010  I = NISIT, NFSIT
      Z = DMAX1( DABS( Y(I) ) , DEVIAT )
      W(I) = DSQRT( Z ) * DELTAY
 6010 CONTINUE
C
C
      GOTO 11111
C
 7000 CONTINUE
C
C     MODE:= 7,  THE ERROR OF MEASUREMENT IS ASSUMED AS CONSISTING
C                OF TWO COMPONENTS, A LINEAR ONE AND A CONSTANT ONE.
C
C           W(I):= DEVY * DEVIAT  +  ABS( Y(I) ) * DELTAY,
C
C     WHEREBY:   DEVY:= CALCULATED STANDARD DEVIATION OF DATA  Y(I),
C                       I:= NISIT,...,NFSIT.
C
      DEVY = DMAX1( DEVMIN , DEVY )
C
      DO 7100  I = NISIT, NFSIT
 7100 W(I) = DEVY * DEVIAT  +  DABS( Y(I) ) * DELTAY
C
      GOTO 11111
C
C
 8000 CONTINUE
```

```
C
C     MODE:= 8,
C                W(I):= DEVY * DEVIAT  +  SQRT( ABS(Y(I)) ) * DELTAY
C
C     WHEREBY:   DEVY:= CALCULATED STANDARD DEVIATION OF DATA  Y(I),
C                       I:=NISIT,...,NFSIT.
C
      DEVY = DMAX1( DEVMIN , DEVY )
C
      DO 8100  I = NISIT, NFSIT
 8100 W(I) = DEVY * DEVIAT  +  DSQRT( DABS(Y(I)) ) * DELTAY
C
      GOTO 11111
C
C
 9000 CONTINUE
C
C     MODE:= 9,  FREE MODE.
C
      GOTO 11111
C
C
10000 CONTINUE
C
C     MODE:= 10,  FREE MODE.
C
      GOTO 11111
C
11000 CONTINUE
C
C     DETERMINATION OF W WITH SLIDING AVERAGE STANDARD DEVIATION.
C     NP IS THE NUMBER OF INCLUDED NEIGHBOR POINTS OF Y(I) IN ONE
C     DIRECTION.
C
      NEELEM = MODE - 10
      NP = MIN( NEELEM , NFSIT-NISIT+1 )
      NP = MAX( 1, NP )
C
      CALL SLIDEV( Y,N,NISIT,NFSIT,NP,NTRUE,W,LERROR)
C
      DO 11100 I = NISIT, NFSIT
11100 W(I) = DEVIAT * DMAX1( DEVMIN , W(I) ) + DELTAY * DABS(Y(I))
C
C
11111 IF ( .NOT. LTEST )  RETURN
C
      ICTEST = ILTEST
      IF ( ILTEST .EQ. 6 )    GOTO 100
C
      ICTEST = 18
C
  100 WRITE(ICTEST,40)
   40 FORMAT(//'          WEIGHTS W OF DATA Y  '
     #        /'          ********************  '// )
C
```

```
      WRITE(ICTEST,50) NISIT, NFSIT, NDIFF, S, DEVY, AVERY
   50 FORMAT(//' NISIT = ',I5,'     NFSIT = ',I5,
     #      /' NDIFF = ',I5,'     S =     ',F12.4,
     #      /' DEVY = ',F14.5,',    AVERY = ',F14.5//)
C
      N5STEP = 5 * NSTEP
C
      DO 500 I = NISIT, NFSIT, N5STEP
      IFI = MIN ( I + 4*NSTEP, NFSIT )
      WRITE(ICTEST,55)  I, ( Y(K), K = I, IFI , NSTEP )
  500 WRITE(ICTEST,60)  I, (W(K), K = I, IFI , NSTEP )
C
   55 FORMAT('  Y(',I4,') = ',5F12.5,' ')
   60 FORMAT(' W(',I4,') = ',5F12.5/)
C
C
      RETURN
C
C     END OF CUSMCW
C
      END
C
C*******************************************************************************
```

```
C
C*********************************************************************
C
      SUBROUTINE CUSMCO ( X, Y, W, N, NISIT, NFSIT, SM, CO, C, IC,
     #                    WK, NWK, IER )
C
C*********************************************************************
C
C     PROCEDURE: CUSMSP    VERSION: FORTRAN IV/77      DATE: 22.3.82
C
C*********************************************************************
C
C     AUTHOR: J. HELLAUER                                   PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        PROCEDURE FOR CALCULATING THE COEFFICIENTS OF A SMOOTHING
C        CUBIC SPLINE FUNCTION APPROXIMATING DATA ( X, Y).
C
C        "CUSMCO" IS CALLED BY PROCEDURE "CUSMSP", WHICH
C        SMOOTHS DATA ( X,Y ) OR AN ASSUMING REAL FUNCTION YY
C        OVER AN INTERVAL (. X(1) , X(N) ), WHOSE FUNCTION VALUES
C        YY(X) ARE ONLY AVAILABELY IN THE DISCRETE POINTS X(I).
C        I. E.  Y(I):= YY( X(I) ), FOR I = 1..N.
C
C        THE SMOOTHING FUNCTION SP TO BE CONSTRUCTED SHOULD MINIMIZE
C
C                          INTEGRAL( (G''(X))**2 DX )
C
C        AMONG ALL TWICE DIFFERENTABLE FUNCTIONS G ON THE GIVEN
C        RANGE  (. X(NISIT) , X(NFSIT) ), 1 <= NISIT < NFSIT <= N,
C        SUCH THAT
C
C              SUM( ( ( SP(X(I)) - Y(I) ) / W(I) )**2 ) <= SM
C
C        THE SOLUTION OF THIS MINIMUM PROBLEM IS OBTAINED BY USING
C        CALCULUS OF VARIATION.
C
C        THE WEIGHTS W(I), I:= 1,..., N, CONTROLLS THE EXTENT OF
C        SMOOTHING AND CAN BE DESTINATED IN PROCEDURE "CUSMCW".
C        MORE INFORMATION ABOUT THIS PROBLEM YOU CAN GET FROM
C        COMMENTS OF ADDED PROCEDURES 'CUSMSP' AND 'CUSMCW' AND
C        OF COURSE FROM SUPPLIED BIBLIOGRAPHY.
C
C        ALMOST ALL INPUT PARAMETERS ARE USED UNALTERED IN THE
C        CALLED PROCEDURES 'CUSMSP' AND 'CUSMCW'. SO YOU CAN SEE
C        THE OPERATING INSTRUCTION OF IT IN RESPECTIVE COMMENTS.
C
C        THIS PROCEDURE IS A FORTRAN IV TRANSLATION OF THE ALGOL60-
C        PROCEDURE "SMOOTH" WRITTEN BY  C. H. REINSCH IN HIS ESSAY.
C
C
C*********************************************************************
```

```
C
C     BIBLIOGRAPHY:
C
C        CHRISTIAN H. REINSCH..: SMOOTHING BY SPLINE FUNCTIONS I.
C                                NUMER. MATHEMATIK 10,
C                                177 - 183 (1967)
C
C        --------------------..: SMOOTHING BY SPLINE FUNCTIONS II.
C                                NUMER. MATHEMATIK 16,
C                                451 - 454 (1971)
C
C
C***********************************************************************
C
C     LINKAGE:
C
C        CALL CUSMCO( X, Y, W, N, NISIT, NFSIT, SM, CO, IC,
C                     WK, NWK, IER )
C
C-----------------------------------------------------------------------
C
C     INPUT:
C
C        X.....: REAL*8,  DIMENSION X(N),
C                ABSCISSAE VECTOR OF DATA FIELD ( X, Y).
C
C
C        Y.....: REAL*8,  DIMENSION Y(N),
C                ORDINATE VECTOR, GIVEN DATA FOR SMOOTHING
C
C
C        W.....: REAL*8, DIMENSION W(N),  W(I) > 0.0,
C                WEIGHTS, WHICH CONTROL THE EXTENT OF SMOOTHING
C                SPLINE FROM THE ORIGINAL DATA  BY THE ABOVE-
C                MENTIONED APPROXIMATION CONDITION.
C                YOU CAN DESTINATE SUITABLE WEIGHTS BY CALLING
C                PROCEDURE "CUSMCW".
C
C
C        N.....: INTEGER,
C                DIMENSION OF  X, Y AND W.
C
C
C        NISIT,
C        NFSIT.: INTEGER,
C                FIRST AND LAST INDEX OF X, WHOSE DATA
C                Y(NISIT),...,Y(NFSIT) SHOULD BE CONSIDERED FOR
C                SMOOTHING.
C                IF NYS:= NS  OR  NDY:= NS  OR ND2:= NS  AND
C                ILNSIT NOT EQUAL 1, NISIT AND NFSIT WILL BE
C                CALCULATED SUITABLE BY PROCEDURE 'CUSMSP'.
C
C
C        SM....: REAL*8,  0.0 < SM,
C                BOUND OF SUMMED, WEIGHTED EXTEND OF THE
```

```
C                SMOOTHING SPLINE SP IN THE ABSCISSAE X(I),
C                SEE ABOVE.
C
C
C        IC....: INTEGER,
C                NUMBER OF COLUMNS OF THE COEFFICIENT MATRIX "C".
C                IC MUST BE SET AS
C                IC:= N-1
C
C
C        WK....: REAL*8, DIMENSION (7,NWK),
C                WORKING AREA,  NWK:= N + 2.
C
C
C        NWK...: INTEGER, LENGHT OF WK,
C                NWK MUST BE SET AS
C                NWK:= N + 2.
C
C
C*****************************************************************
C
C     OUTPUT:
C
C        CO....: REAL*8,  WITH LENGHT N,
C                FUNCTION VALUES OF THE SMOOTHING SPLINE
C                IN THE ABSCISSAE X(I), I. E.
C                CO(I):= SP( X(I) ).
C
C
C        C.....: REAL*8,  DIMENSION  C(IC,3)
C                COEFFICIENT MATRIX FOR SPLINE EVALUATION.
C                CALCULATION OF IT IS DONE IN PROCEDURE 'CUSMCO'.
C
C                WITH "CO" AND "C", YOU CAN EVALUATE EVERY FUNCTION
C                VALUE SP(Z) OF SPLINE SP FOR ABSCISSAE Z, WITH
C                X(I) <= Z < X(I+1)  AND  NISIT <= I <= NFSIT.
C                FOR  H:= Z- X(I)  IS
C                SP(Z) = CO + (( C(I,3)*H + C(I,2) )*H + C(I,1) )*H.
C                BY CALLING THE PROCEDURE "CUSPED" YOU CAN EVALUATE
C                EVERY FUNCTION VALUE OF SP.
C
C
C        IERROR: INTEGER,
C
C                := 0,      NO ERROR DETECTED,
C
C
C                 < -900,   FATAL ERROR  ( NO CALCULATION IS DONE )
C
C                := -996,   INPUT ABSCISSAE ARE NOT ORDERED, SO THAT
C                           X(1) < X(2) < ... < X(N).
C
C                := -997,   IC IS NOT  N-1,
C
C                := -998,   NISIT < 1      OR
```

```
C                          NFSIT > N       OR
C                          NFSIT < NISIT+2
C
C
C*****************************************************************
C
C      OPERATING SYSTEM:          CMS
C
C*****************************************************************
C
C      EXTERNAL PROCEDURES:       NONE
C
C*****************************************************************
C
C
C
C*****************************************************************
C
```

```
C
C*********************************************************************
C
      SUBROUTINE INDEXD( X, N, AI, BI, N1, N2 )
C
C*********************************************************************
C
C     PROCEDURE: INDEXD     VERSION: FORTRAN IV/77   DATE: 16.3. 82
C
C*********************************************************************
C
C     AUTHOR:  J. HELLAUER                                  PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        DETERMINATION OF THE INDICES  N1  AND  N2  ON THE SCALE X,
C        WHERE   X(1) <= X(N1) <= AI <= BI <= X(N2) <= X(N).
C
C*********************************************************************
C
C     LINKAGE:   CALL INDEXD( X, N, AI, BI, N1, N2 )
C
C*********************************************************************
C
C     INPUT:
C
C        X      :  REAL*8 ARRAY, DIMENSION  X(N),
C                  CONSISTS OF THE GRADUATION OF SCALE
C
C        N      :  INTEGER, DIMENSION OF X
C
C        AI, BI:   REAL*8,
C                  LEFT RESPECTIVELY RIGHT POINT OF SCALE DIVISION
C                  TO BE CONSIDERED
C
C---------------------------------------------------------------------
C
C     OUTPUT:
C
C        N1, N2:   START RESPECTIVELY END INDICES OF SUBDIVISION
C                  INTERVAL  (. X(N1) , X(N2) .), WHERE
C                  X(1) <= X(N1) <= AI <= BI <= X(N2) <=  X(N)
C
C*********************************************************************
C
C     OPERATING SYSTEM:  CMS
C
C*********************************************************************
C
C     EXTERNAL PROCEDURES: NONE
C
C*********************************************************************
C
```

```
C
      REAL*8  X(N), AI, BI
C
      I = N + 1
 1001 I = I - 1
      IF ( X(I) .GT. AI  .AND.  I .GT. 1 )     GOTO 1001
      N1 = I
C
      I = 0
 1002 I = I + 1
      IF ( X(I) .LT. BI  .AND.  I .LT. N )      GOTO 1002
      N2 = I
C
C
      RETURN
C
C     END OF INDEXD
C
      END
C
C*****************************************************************************
C
```

```
C
C*********************************************************************
C
      SUBROUTINE AVEDEV(Y,NDIM,N1,N2,NTRUE,AVERAG,DEVIAT,IERROR)
C
C*********************************************************************
C
C     PROCEDURE: AVEDEV     VERSION: FORTRAN IV/77     DATE: 6.8. 82
C
C*********************************************************************
C
C     AUTHOR: J. HELLAUER                                  PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        AVEDEV CALCULATES THE AVERAGE VALUE AVERAG AND THE STANDARD
C        DEVIATION 'DEVIAT' OF THE REAL*8-DATA FIELD Y(N1),..,Y(N2),
C        1 <= N1 <= N2 <= NDIM.
C
C*********************************************************************
C
C     LINKAGE:
C
C        CALL AVEDEV (Y,NDIM, N1, N2, NTRUE, AVERAG, DEVIAT,IERROR)
C
C*********************************************************************
C
C     INPUT:
C
C        Y.....: REAL*8,  DIMENSION NDIM,
C                DATA FIELD, WHOSE AVERAGE VALUE AND STANDARD
C                DEVIATION SHOULD BE CALCULATED
C
C        NDIM..: INTEGER,
C                DIMENSION OF Y.
C
C        N1, N2: INTEGER, 1 <= N1 <= N2 <= NDIM,
C                INDEX BOUNDARIES OF THE WANTED REGION. 'AVERAG' AND
C                'DEVIAT' ARE ONLY DESTINATED BY THE DATA VALUES
C                Y(N1),...,Y(N2).
C
C---------------------------------------------------------------------
C
C     OUTPUT:
C
C        NTRUE.: INTEGER,
C                NUMBER OF DATA ITEMS, NTRUE:= N2 - N1 + 1.
C                CAN BE USED FOR AN INDEX CHECK.
C
C        AVERAG: REAL*8,
C                MEAN VALUE OF THE DATA  Y(N1),...,Y(N2).
C                AVERAG:= SUM (Y(I)) / NTRUE.
C
```

```
C         DEVIAT: REAL*8,
C                 STANDARD DEVIATION OF THE DATA  Y(N1),...,Y(N2).
C                 DEVIAT:= SQRT( SUM( (Y(I) - AVERAG)**2 ) / NTRUE ).
C
C         IERROR: INTEGER,   ERROR FLAG,
C                 IERROR:=    0,  CALCULATION CORRECTLY FINISHED.
C                 IERROR:= -999,  ONE OR MORE OF THE INDICES  N1,
C                                 N2 OR NDIM ARE NOT SUITABLE FOR
C                                 CALCULATION.
C
C*****************************************************************
C
C      OPERATING SYSTEM...:  CMS
C
C*****************************************************************
C
C      EXTERNAL PROCEDURES:  NONE
C
C*****************************************************************
C
C      EXTERNAL FILES.....:  NONE
C
C*****************************************************************
C
C
       REAL*8  Y(NDIM), SX, SX2, NDIFFI, S, AVERAG, DEVIAT
C
       IERROR = 0
C
       IF( N1 .GE. 1  .AND. N1 .LE. NDIM  .AND.
      #    N2 .GE. N1 .AND. N2 .LE. NDIM        )     GOTO  1000
C
       IERROR = -999
C
       WRITE(6,10) N1, N2, NDIM
C
    10 FORMAT(' ERROR IN SBR AVEDEV!'/' N1 = ',I5,',   N2 = ',I5,
      #       ',    NDIM = ',I5/)
C
       RETURN
C
  1000 AVERAG = 0.0
       DEVIAT = 0.0
C
       DO 1100  I = N1, N2
       AVERAG = AVERAG + Y(I)
  1100 DEVIAT = DEVIAT + Y(I)*Y(I)
C
       NTRUE = N2 - N1 + 1
       NDIFFI = 1.0/DFLOAT(NTRUE)
C
       AVERAG = AVERAG * NDIFFI
       DEVIAT = DEVIAT*NDIFFI - AVERAG*AVERAG
       DEVIAT = DSQRT(DEVIAT)
C
```

```
C
      RETURN
C
C     END OF AVEDEV
C
      END
C
C*****************************************************************
C
```

```
C
C*********************************************************************
C
      SUBROUTINE SLIDEV( Y, NDIM, N1, N2, NP, NTRUE, DEVIAT, IERROR)
C
C*********************************************************************
C
C     PROCEDURE:  SLIDEV    VERSION: FORTRAN IV/77     DATE: 9.8. 82
C
C*********************************************************************
C
C     AUTHOR:     J. HELLAUER                                PHONE: 494
C
C*********************************************************************
C
C     FUNCTION:
C
C        FOR ALL INDICES I, I:= N1,...,N2,  SLIDEV CALCULATES THE
C        STANDARD DEVIATION DEVIAT(I), WHICH IS destinated BY USING
C        THE MAXIMAL (2*NP+1) NEIGHBOR POINTS  Y(IL),...,Y(IU), WITH
C        IL:= MAX( N1 , I-NP )  AND  IU:= MIN( N2 , I+NP ).
C
C        'DEVIAT(I)' IS CALLED THE SLIDING STANDARD DEVIATION WITH
C        MAXIMAL (2*NP+1) MEMBERS.
C
C*********************************************************************
C
C     LINKAGE:
C
C        CALL SLIDEV( Y, NDIM, N1, N2, NTRUE, NP, DEVIAT, IERROR )
C
C*********************************************************************
C
C     INPUT:
C
C        Y.....: REAL*8,  DIMENSION (NDIM),
C                DATA FIELD, WHOSE SLIDING DEVIATIONS SHOULD BE
C                CALCULATED.
C
C        NDIM..: INTEGER,
C                DIMENSION OF  Y  AND  DEVIAT.
C
C        N1, N2: INTEGER,
C                BOUDARY INDEX OF CONSIDERED DATA RANGE
C                Y(N1),...,Y(N2).
C
C
C        NP....: INTEGER,  0 <= NP <= (N2-N1+1),
C
C                HIGHEST NUMBER OF NEIGHBOR POINTS IN ONE DIRECTION,
C                WHICH SHOULD BE INTEGRATED FOR DETERMINATION OF
C                DEVIAT(I).
C                I.E. THE  2*NP+1  POINTS  Y(IU),...,Y(IO), WITH
C                IU:= MAX( I-NP , N1 )  AND  IO:= MIN( I+NP , N2 ),
C                INFLUENCE THE VALUES DEVIAT(I).
```

```
C
C               NP:= 0,
C               EFFECTES DEVIAT(I):= 0.0, FOR ALL I:= N1,..,N2.
C
C               NP:= N2-N1+1,  ===>  DEVIAT(I):= DEVCON, FOR
C               I:= N1,..,N2, WITH DEVCON IS THE STANDARD DEVIATION
C               OF DATA FIELD  Y(N1),..,Y(N2).
C
C-----------------------------------------------------------------
C
C     OUTPUT:
C
C        NTRUE.: INTEGER,
C                NUMBER OF NEEDED DATA POINTS,
C                NTRUE:= N2 - N1 + 1.
C
C        DEVIAT: REAL*8, DIMENSION(NDIM),
C                SLIDING STANDARD DEVIATION OF THE POINT Y(I) AND
C                ITS NEAREST  2*NP+1  NEIGHBOR POINTS.
C
C        IERROR: INTEGER, ERROR FLAG,
C                IERROR:=    0, NO ERROR OCCURED.
C                IERROR:= -999, N1, N2 OR NDIM WRONG PLACED,
C                               CALCULATION IMPOSSIBLE.
C                IERROR:= -888, NP < 0  IS GIVEN BY INPUT,
C                               FURTHER CALCULATION IMPOSSIBLE.
C
C
C*****************************************************************
C
C     OPERATING SYSTEM:     CMS
C
C*****************************************************************
C
C     EXTERNAL PROCEDURES:  AVEDEV (FORTRAN)
C
C*****************************************************************
C
C
      REAL*8  Y(NDIM), DEVIAT(NDIM), AVECON, DEVCON
C
      IERROR = 0
C
      IF ( N1 .GE. 1  .AND. N1  .LE. NDIM  .AND.
     #     N2 .GE. N1 .AND. N2  .LE. NDIM        )      GOTO 1
C
      IERROR = -999
C
      RETURN
C
    1 IF ( NP .GE. 0 )     GOTO 1000
C
      IERROR = -888
      RETURN
C
```

```
 1000 IF ( NP .GT. 0 )     GOTO 2000
C
      DO 1100 I = N1, N2
 1100 DEVIAT(I) = 0.0
C
      RETURN
C
C
 2000 NDIFF = N2 - N1
C
      IF ( NDIFF .GT. NP )     GOTO 3000
C
      CALL AVEDEV( Y,NDIM,N1,N2,NTRUE,AVECON,DEVCON, IERROR )
C
      DO 2100  I = N1, N2
 2100 DEVIAT(I) = DEVCON
C
      RETURN
C
C
 3000 CONTINUE
C
      DO 3100 I = N1, N2
C
      IL = MAX( N1, I-NP )
      IU = MIN( N2, I+NP )
C
      CALL AVEDEV( Y,NDIM,IL,IU,NO,AVECON,DEVIAT(I),IERROR )
C
 3100 CONTINUE
C
C
      RETURN
C
C     END OF SLIDEV
C
      END
```

4. Beispiele

4.1 Auswertung von Messdaten

Wie bereits erwähnt, wurden alle angeführten Approximationsalgorithmen bei der Messdatenauswertung am Plasmaexperiment W-VIIa im IPP getestet. Die bei der Messdiagnostik VUV-Spektroskopie erfassten Daten werden ins Rechenzentrum des IPP übertragen und dort ausgewertet. Die aufgenommenen Signale bestehen aus Photonen-Intensitäten, die von Plasmaverunreinigungen stammen. Aus den diskreten Sigalwerten, in den beigelegten Plots als Messpunkte zu erkennen, sollte eine mindestens einmal stetig differenzierbare Funktion rekonstruiert werden. Die Auswertungen erfolgten an der Siemens 7880 unter VM/370-CMS.

Wie erwartet erwiesen sich die Interpolationsverfahren zur Auswertung von Messdaten als ungeeignet, da diese aus physikalisch-statistischen Gründen starke Oszillationen aufweisen. Die relativ grossen Schwankungen der einzelnen aufeinanderfolgenden Datenpunkte erschwerte insbesonders die anschliessende Numerische Abelinversion zur Darstellung von Dichteprofilen. Mit einem speziellen Algorithmus, der auf der hier benutzten Splinedarstellung aufbaute, konnte dieses Auswerteverfahren verbessert bzw. anwendbar gemacht werden. Bei geeigneter Wahl der Eingabeparamter 'MODE', 'DELTAY' und 'DEVIAT' zeigten sich sehr gute Erfolge.

In den nachfolgenden Grafikplots sind die Rohdaten mit ihren davon abgeleiteten Interpolations- bzw Ausgleichssplines aufgezeichnet. Die jeweils vorgegebenen Glättungsparameter 'MODE', 'DELTAY' und 'DEV' sind hinzugefügt. Die Bedeutung der einzelnen Parameter entspricht der Terminologie von (3.2). Die Interpolations-Splines (MODE < 0) wurden mit den in (2.3) vorgestellten Algorithmen "CUSP**" berechnet. Das Programmpaket "CUSM**" diente zur Darstellung der Ausgleichssplines (MODE > 1).

CMS-Benutzer haben durch die Kommandos

```
CP LINK DATANALY 191 <vaddr> RR PUBLIC
ACcess <vaddr> <mode>
```

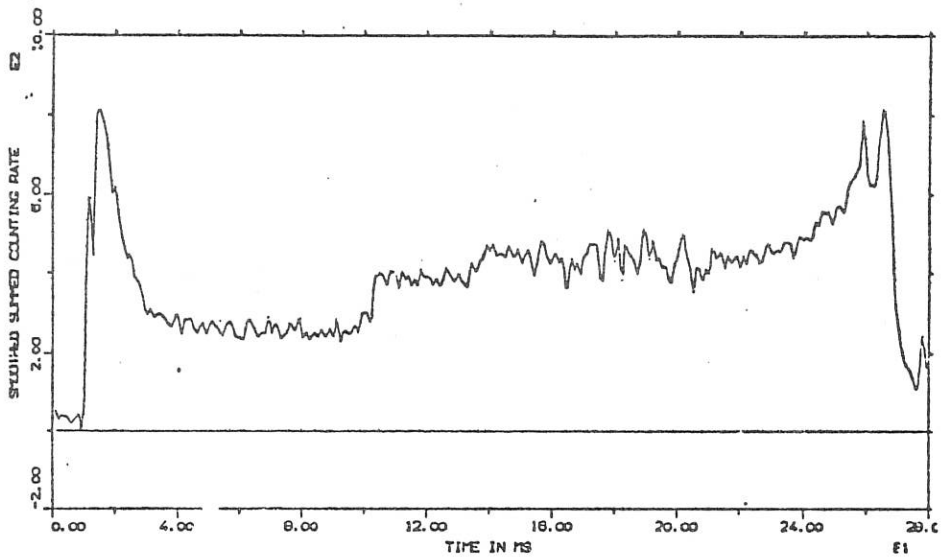
Lesezugriff zu den Files dieser VM und koennen durch das Kommando

```
GLOBAL TXTLIB DATANALY ...
```

die Object-Library in ihr Programm einbinden.

SHOTNUM:38678 CHANNEL: 4

I DELTAY = 0.00, DEV = 0.00 , MOD = -2



II, DELTAY = 0.00, DEV = 0.00 , MOD = -2

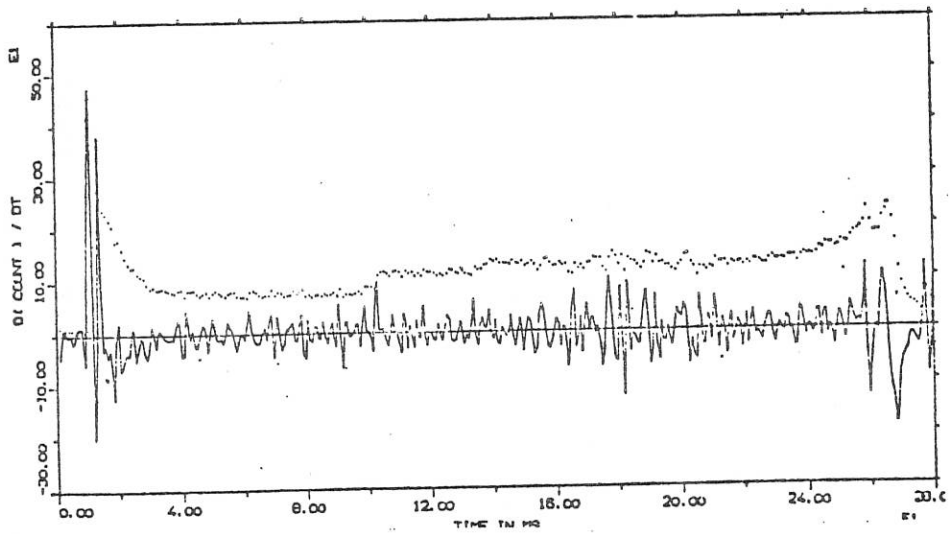


Abb.(4.1.1)

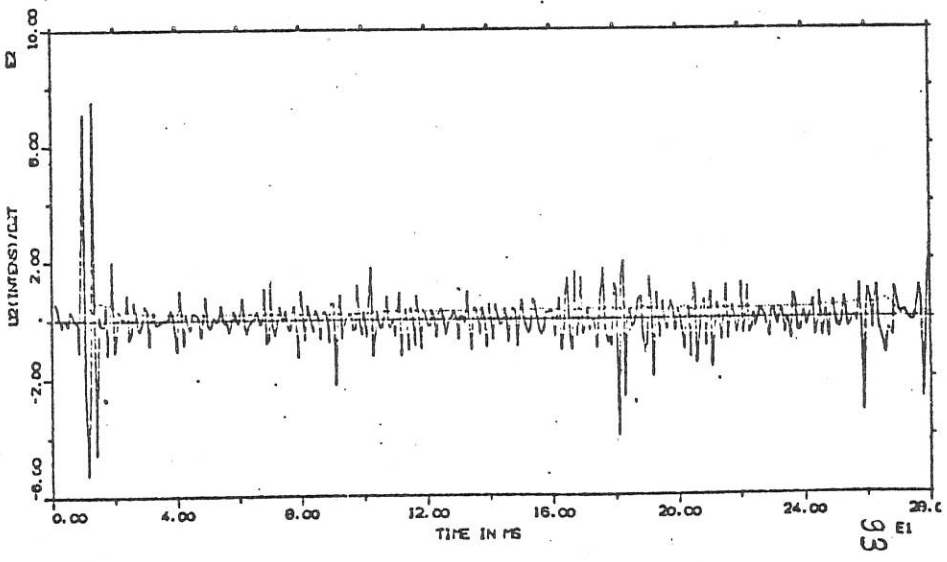
Kubische Spline-Interpolation von n:= 280 Messdaten

I - Die Oszillationen des Datenfeldes übertragen sich vollständig auf die Interpolationskurve und machen eine weitere sinnvolle Auswertung der Näherungslösung praktisch unmöglich.

II - Die Darstellung der 1. Ableitung der Spline-Interpolation wird durch die Oszillationen der Daten unbrauchbar. Man beachte die hohen absoluten Schwankungen von Ds:= ds/dt in aufeinander- folgenden Zeitpunkten t(i) und t(i+1), die physikalisch völlig unplausibel sind.

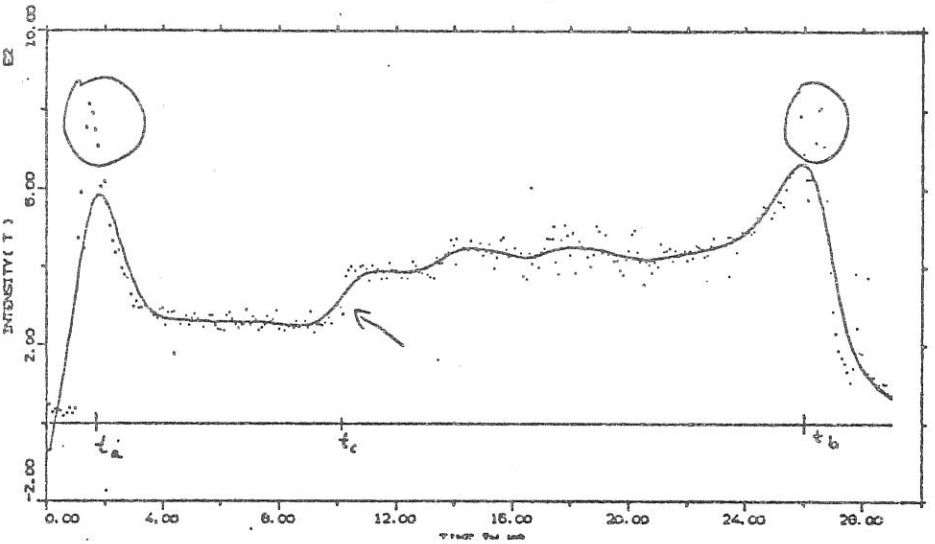
III - Bei der 2. Ableitung tritt dieser störende Effekt noch stärker in den Vordergrund.

III, DELTAY = 0.00, DEV = 0.00 , MOD = -2



SHOTNUM: 38678 CHANNEL: 4

I DELTAY = 0.00, DEV = 0.15 , MOD = 7



II DELTAY = 0.00, DEV = 0.15 , MOD = 7

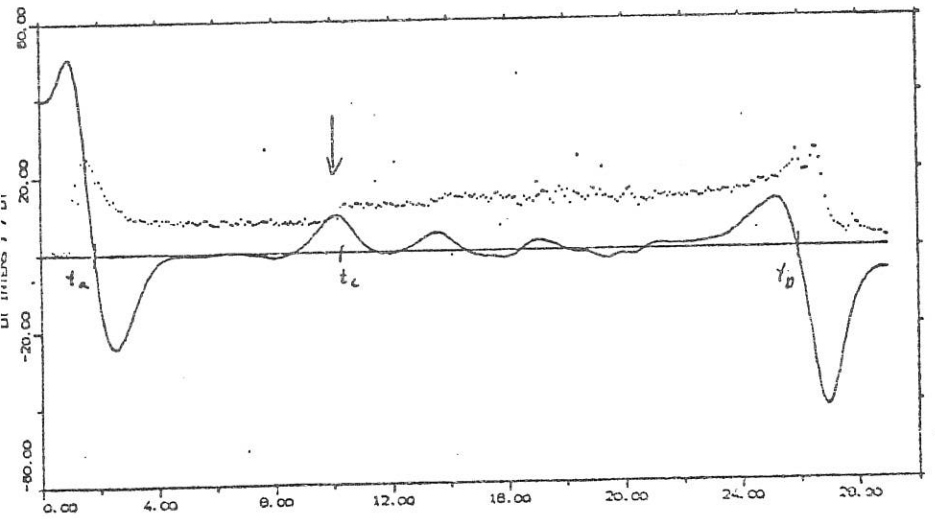


Abb.(4.1.2)

Ausgleichsspline zu den Messdaten von (4.1.1)

I - Ausgleichsspline s mit Datenpunkten. Die Glättungsparameter wurden den Eigenschaften des Datenfeldes angepasst. Von den Umgebungen um die Extremalpunkte ta und tb abgesehen, liegt die Approximationskurve innerhalb der jeweiligen Datenstreubreite.

II - Darstellung der 1. Ableitung von s. Deutlich wird das physikalisch relevante Ansteigen der Intensitätswerte in der Umgebung um tc erkennbar. Erstaunlich gut tritt die Symmetrie der Intensitatsfunktion in der Umgebung von ta zu der von tb zutage. Die relativ grosse Abweichung der Näherung s von den Datenpunkten bei ta und tb wirkt sich bei der Approximation der Ableitung absolut gesehen nur wenig aus.

III - Darstellung der 2. Ableitung von s. Man vergleiche die Absolutwerte ihrer Minima und Maxima mit denen von Abb.(4.1.1).

III DELTAY = 0.00, DEV = 0.15 , MOD = 7

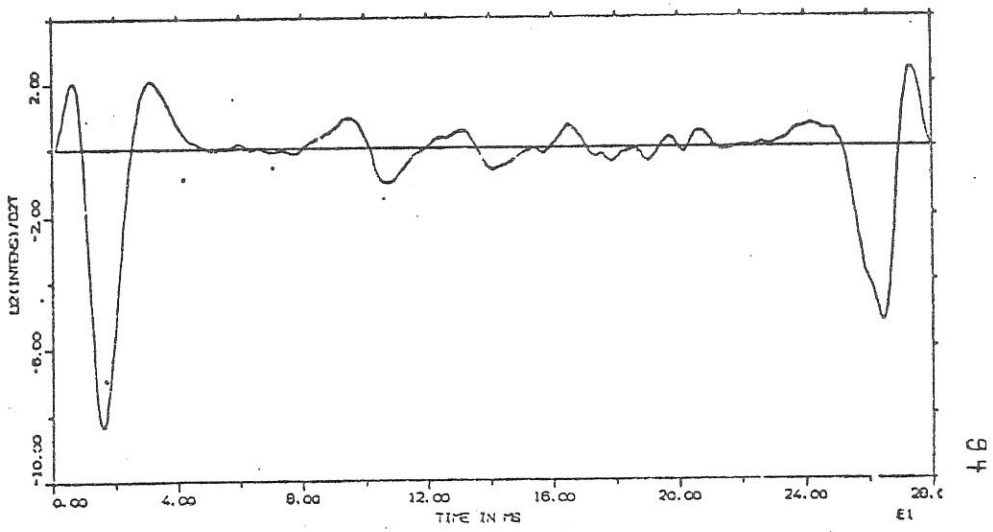


Abb.(4.1.3)

Ausgleichsspline zum Datenfeld (4.1.1) mit verschiedenen Glättungsparametern.

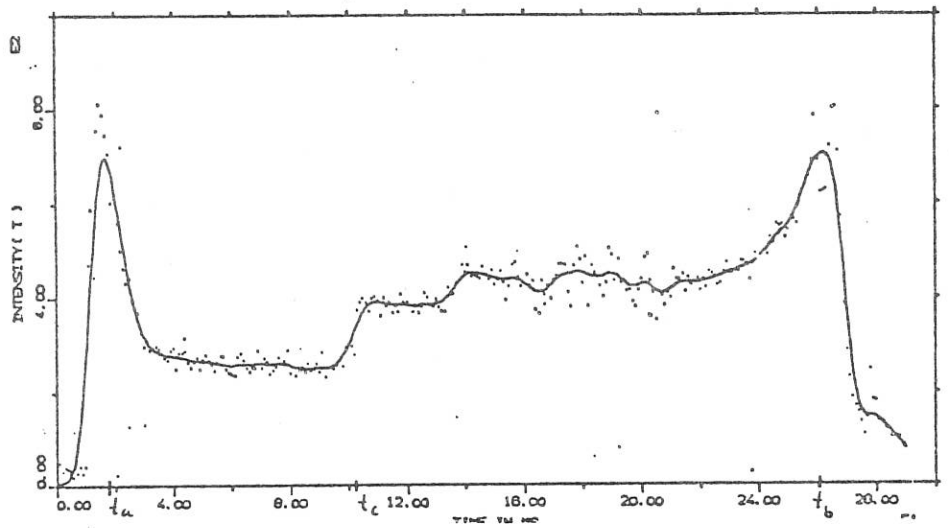
I - Die Verkleinerung des Steuerparameters DEV:= 0.10 (gegenüber 0.15) verbessert die Approximationseigenschaft des Ausgleichssplines s in den Extremalpunkten ta und tb, bedingt jedoch ein stärkeres Oszillieren der Näherung.

II - Bei Mode:= 13 wird fur ein y(i) die Standardabweichung aus den 6 jeweiligen Nachbarpunkten zur Bestimmung der w(i) herangezogen. Diese Gleitende Standardabweichung wird bei den Extremalpunkten ta und tb relativ gross und beeinflusst somit die Gewichte w(i) und damit die Fehlervorgabe des Splines in der Umgebung von t(i).

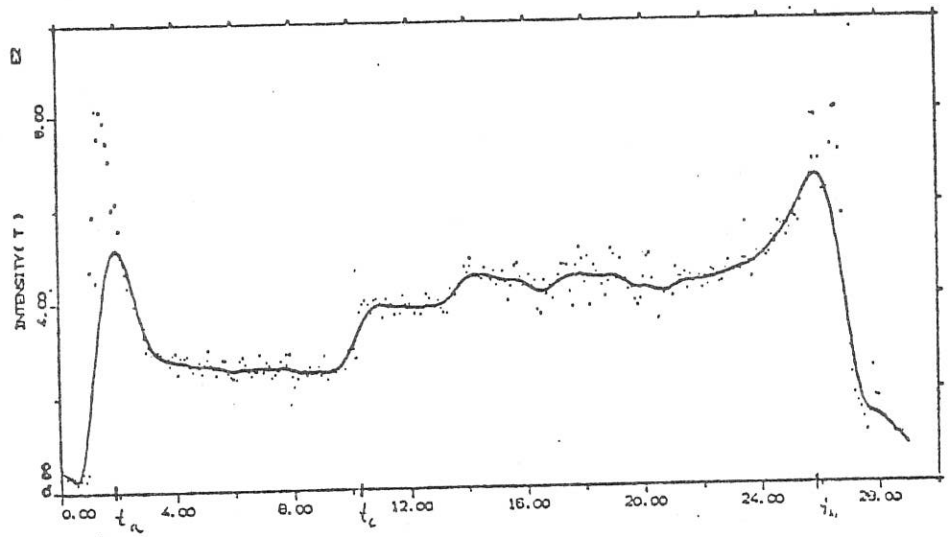
III - Bei beliebiger Vergrösserung der Gewichte w(i) erhält man wegen der Minimalvorgabe an die Krümmung von s als Ausgleichsspline eine Gerade, die aber im allgemeinen nicht mit der üblichen Regressionsgerade (1.1.2) übereinstimmt.

SHOTNUM: 38678 CHANNEL: 4

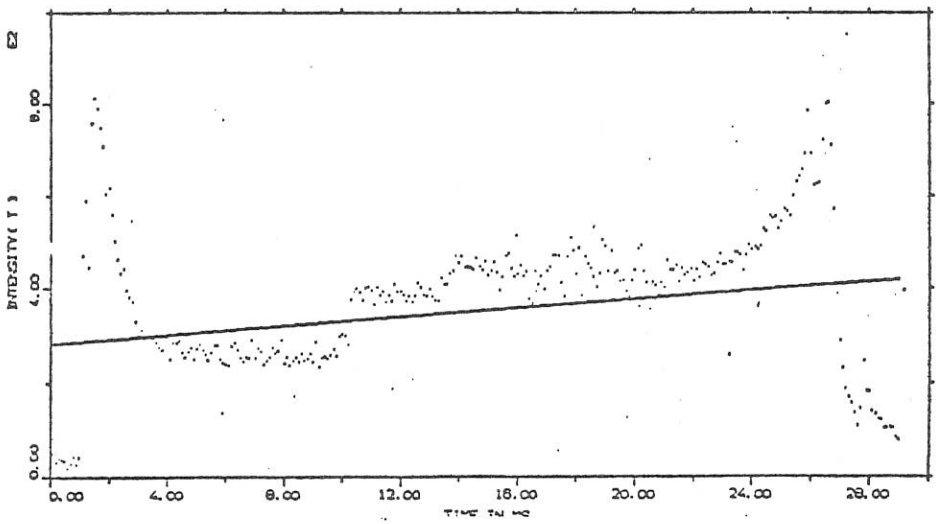
I DELTAY = 0.00, DEV = 0.10 , MOD = 7



II DELTAY = 0.05, DEV = 0.40 , MOD = 13

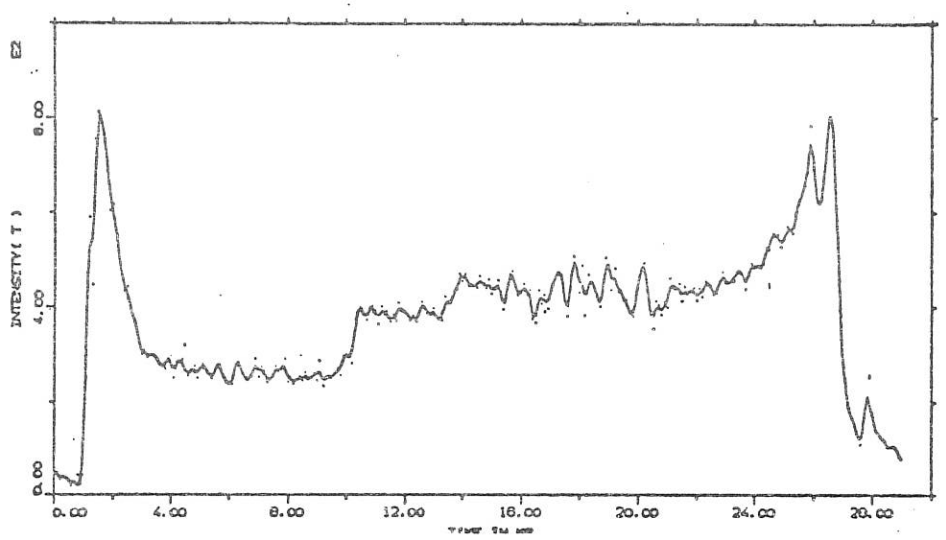


III DELTAY = 0.10, DEV = 0.30 , MOD = 7



SHOTNUM: 38678 CHANNEL: 4

I DELTAY = 0.00, DEV = 25.00 , MOD = 2



II DELTAY = 0.00, DEV = 50.00 , MOD = 2

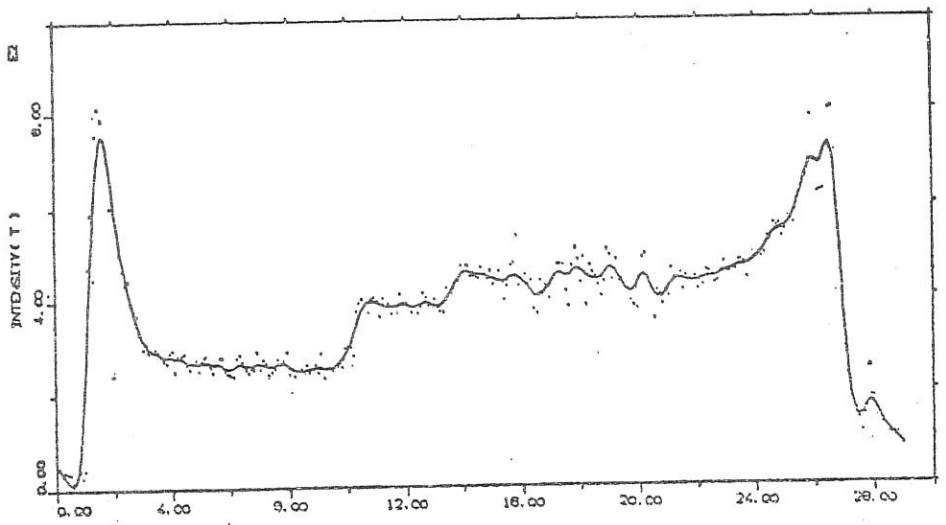


Abb.(4.1.4)

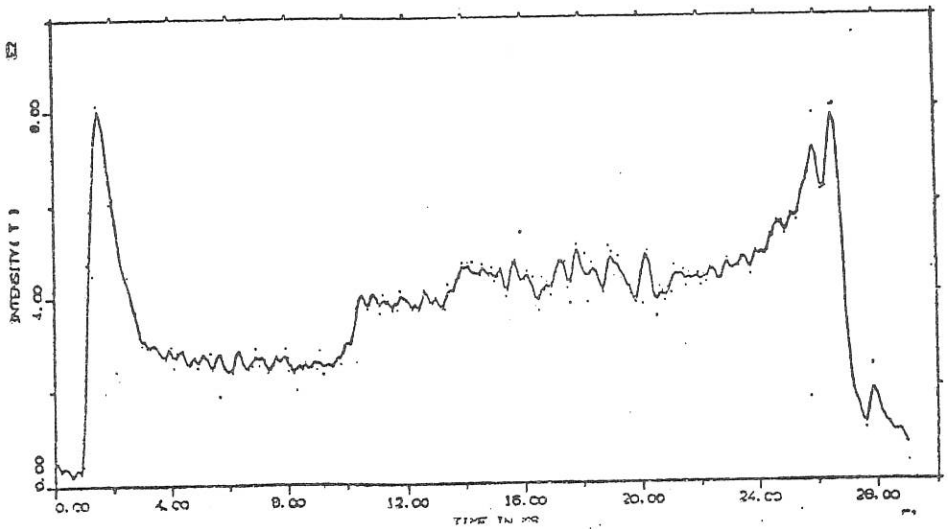
Ausgleichsplines zum Datenfeld (4.1.1) mit verschiedenen Glättungsparametern.

I - Diesem Ausgleichsspline lag die Annahme eines unabhängigen, statistischen und im Intervall [-25.0 , 25.0] gleichverteilten Messfehlers zugrunde. Die Schätzung erwies sich als zu klein; der Spline weist grosse Oszillationen auf.

II - Die Annahme eines Messfehlers im Bereich [-50.0 , 50.0] erwies sich als realistischer.

III - Die Vorgabe der Gewichte w(i) als halbe Standardabweichung der 3 Datenpunkte y(i-1), y(i), y(i+1) war offensichtlich zu klein. Der Ausgleichsspline s zeigt ein dem Interpolations-Spline ähnliches Verhalten.

III DELTAY = 0.00, DEV = 0.50 , MOD = 11



4.2 Approximation von analytischen Funktionen

Aufschlussreiche Vergleichstests mit Approximationsmethoden werden anhand von analytisch vorliegenden Funktionen vorgenommen, da ihre Funktions- und Ableitungswerte an jeder beliebigen Stelle zur Verfügung stehen und mit der Näherung verglichen werden können. Die Approximation von relativ glatten und differenzierbaren analytischen Funktionen mit Ausgleichssplines ist nicht sehr sinnvoll, hier liefern im allgemeinen Interpolationssplines genauere Ergebnisse.

Will man mit Hilfe von analytisch vorliegenden Funktionen Vergleichstest von Data-Smoothing-Verfahren durchführen, so muss man das sogenannte 'Rauschen' der Messdaten durch gezielte Störungen der ursprünglichen Funktionswerte simulieren. Dies erreicht man am einfachsten durch Addition von Pseudo-Zufallszahlen mit geeigneten Parameterwerten (wie Mittelwert und Standardabweichung).

Die zwei am häufigsten auftretenden Verteilungen von Messfehlern sind die N(m,d)-(Normal-)-Verteilung mit Mittelwert m und Standardabweichung d und die U(c,d)-(Gleich-)-Verteilung im Intervall [c,d]. Für diese beiden Fehlermodelle wurden in (3.2) Algorithmen zur Bestimmung der Gewichte w(i) angegeben. In den durchgeführten Vergleichstests wurden diese Fehlerverteilungen durch Addition entsprechend verteilter Zufallszahlen untersucht. Da die ursprüngliche Funktion f bekannt war, konnte man die Rekonstruierfähigkeit der Ausgleichsspline optimal testen und untereinander vergleichen. Von speziellem Interesse war dabei die Frage, inwieweit der Ausgleichsspline in der Lage ist, brauchbare Approximationen an Df bzw. D2f zu liefern.

Die Testergebnisse bei den künstlich gestörten, analytischen Funktionen lassen folgenden Schluss zu:

- Schon ein durchschnittlicher Messfehler von etwa 10% verhindert eine sinnvolle Approximation der Ableitungen durch Interpolationsmethoden.

- Bei geeigneter Wahl der Glättungsparameter liefert der nach (3.2) bestimmte Ausgleichsspline eine sehr gute Approximation und damit Rekonstruktion der Funktion f und deren Ableitung Df. Sogar die 2. Ableitung D2f wird gut approximiert.

- Erhöht man die Anzahl der Messdaten (hier Stützstellen), so werden die vorteilhaften Approximationseigenschaften des Ausgleichssplines gegenüber den Interpolationsnäherungen immer mehr erkennbar.

In den Abbildungen (4.2.1) bis (4.2.8) werden folgende Bezeichnungen benutzt:

N(m,d): Normalverteilte Zufallszahlen mit Mittelwert 0 und Standardabweichung d.

U(+-d): Im Intervall [-c,c] Gleichverteilte Zufallszahlen.

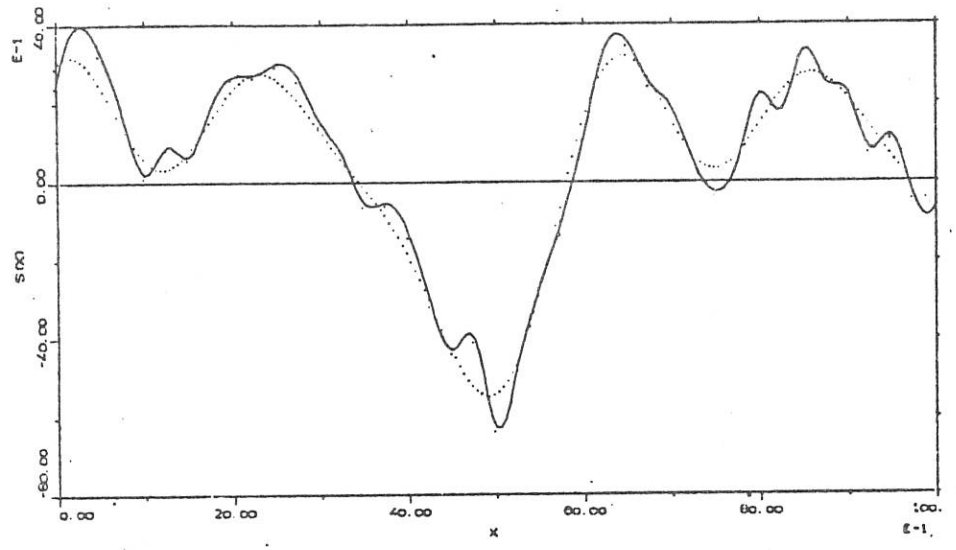
——— : Ausgleichs- oder Interpolationspline s bzw. Ds bzw. D2s.

.... : Ungestörte Funktion f bzw. Df bzw. D2f .

 : Gestörter Datenpunkt p(t(i),y(i))) mit 'Fehlerbalken' .
Die gesamte Höhe von 'I' entspricht der Grösse der jeweils benutzten Gewichtung w(i).

F(X) := 3*SIN(X)+2*SIN(2*X+PI/2)+COS(3*X)
+ N(0, 0.50)-RANDOM NUMBERS

I DELTAY = 0.00, DEV = 0.00 , MOD = -2



II DELTAY = 0.00, DEV = 0.40 , MOD = 4

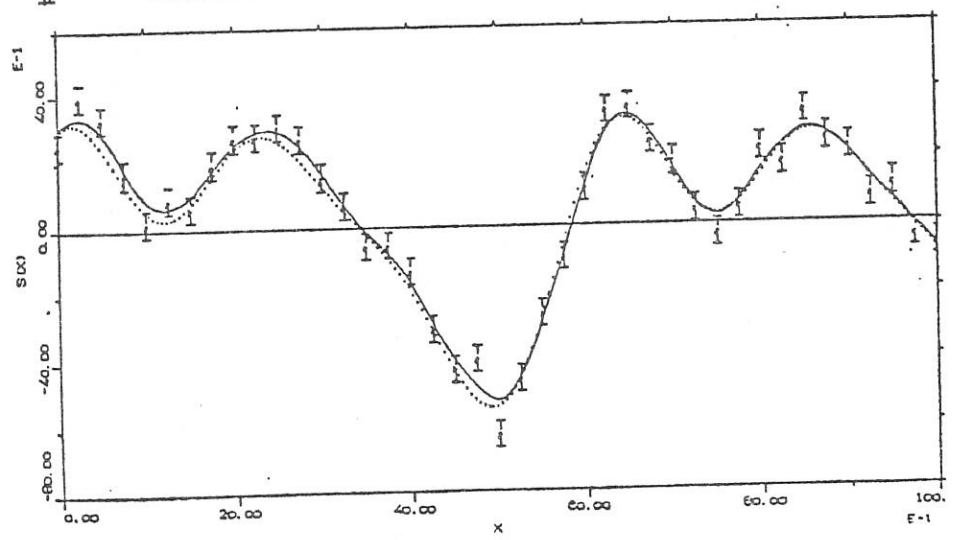


Abb.(4.2.1)

Approximation einer 'gestörten' Funktion mit Kubischen Interpolations-und Ausgleichssplines.

Die im Plot angebenen Funktion f wurde durch Addition von N(0,0.5)-Verteilten Zufallszahlen 'gestört'. Die künstliche Auslenkung simuliert einen durchschnittlichen Messfehler von etwa 15%.

Vergleich der Funktionswerte, n:= 40.

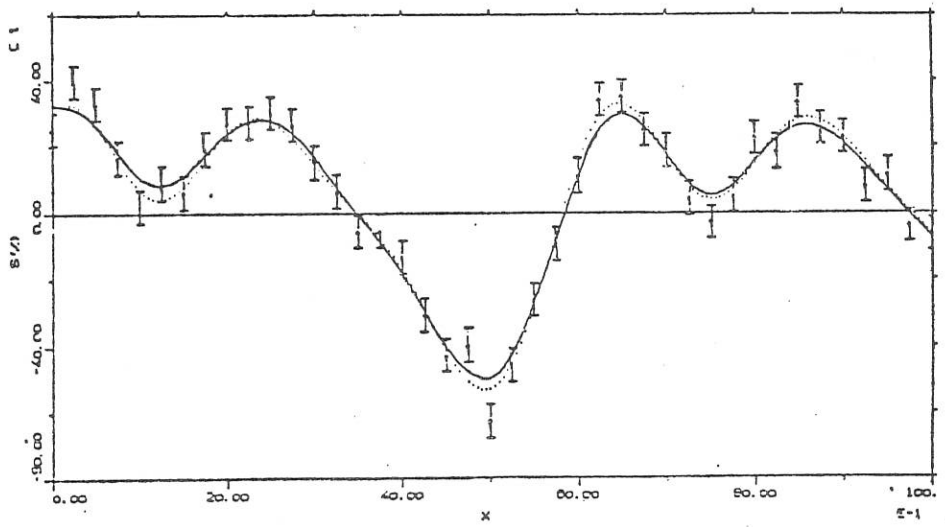
I - Kubische Spline-Interpolation der Datenpunkte mit Hermite-Randbedingungen (MODE:= -2).

II - Kubischer Ausgleichsspline nach (3.1) mit dem in (3.2) erklärten Messfehlermodell MODE:= 4 und dem Glättungsparameter DEVIAT:= 0.4.

III - Wie II, aber DEVIAT:= 0.5. Nach (3.2) sind diese Parameter optimal fur diese Messfehlerverteilung.

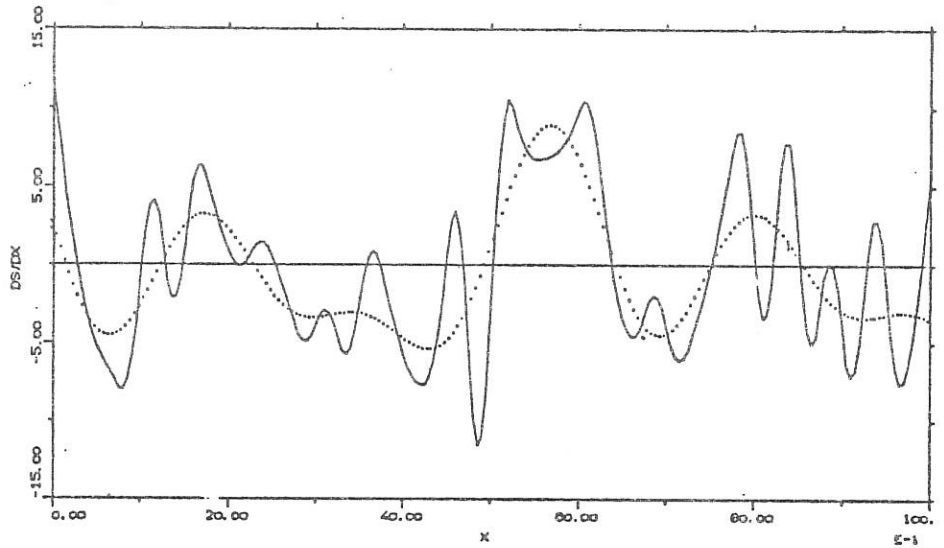
Durch Nachprüfung erhält man, dass beide Ausgleichssplines gegenüber dem Interpolationsspline eine (im Sinne der Tschebischev-Norm) wesentlich bessere Approximation der ungestörten Funktion f liefern.

III DELTAY = 0.00, DEV = 0.50 , MOD = 4



F(X) := 3*SIN(X)+2*SIN(2*X+PI/2)+COS(3*X)
+ N(0, 0.50)-RANDOM NUMBERS

I DELTAY = 0.00, DEV = 0.00 , MOD = -2



II DELTAY = 0.00, DEV = 0.40 , MOD = 4

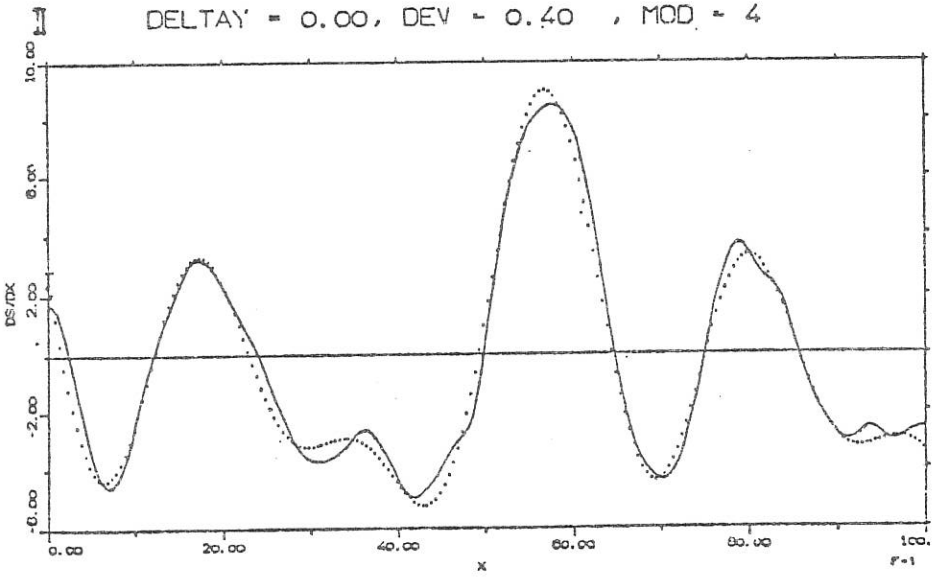


Abb.(4.2.2)

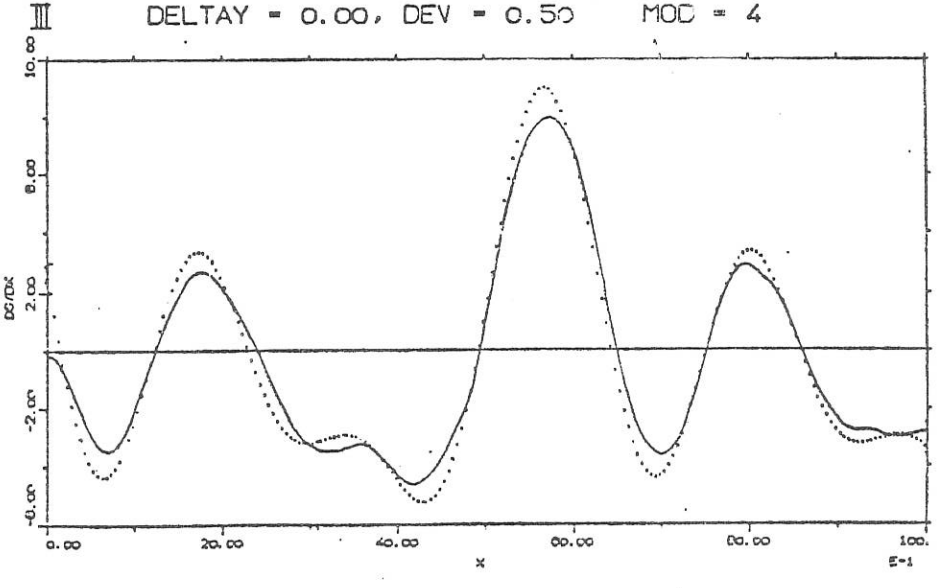
Vergleich der Ableitungswerte aus (4.2.1), n:= 40

I - Wie in (3.1) erwähnt, stören schon relativ kleine Oszillationen der Daten hauptsächlich die Ableitungen des interpolierenden Splines. Mit einer gewöhnlicher Polynominterpolation der Daten erhält man noch schlechtere Ergebnisse.

II - Die Ausgleichssplines liefern dagegen bei dieser Parametervorgabe eine sehr gute Rekonstruktion von f.

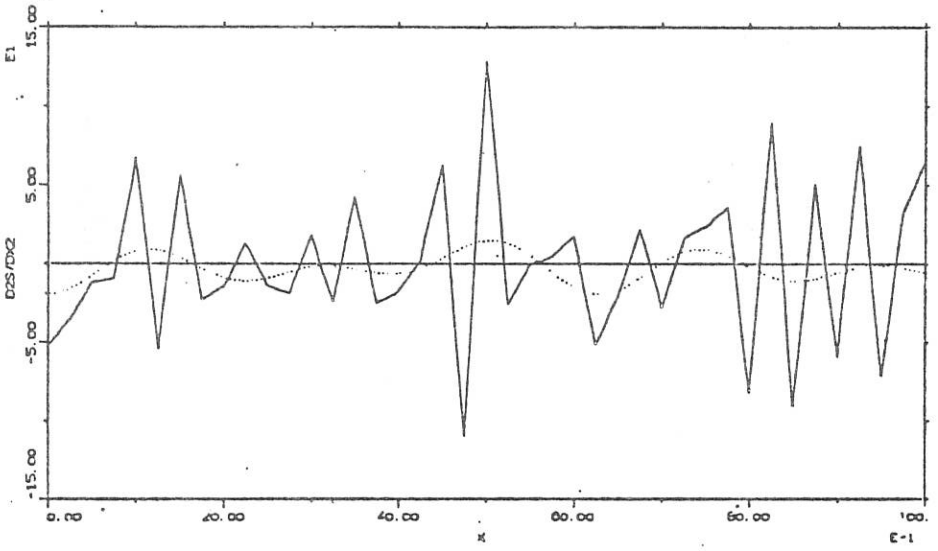
III

III DELTAY = 0.00, DEV = 0.50 MOD = 4



F(X) := 3*SIN(X)+2*SIN(2*X+PI/2)+COS(3*X)
+ N(0, 0.50)-RANDOM NUMBERS

I DELTAY = 0.00, DEV = 0.00 , MOD = -2



II DELTAY = 0.00, DEV = 0.40 , MOD = 4

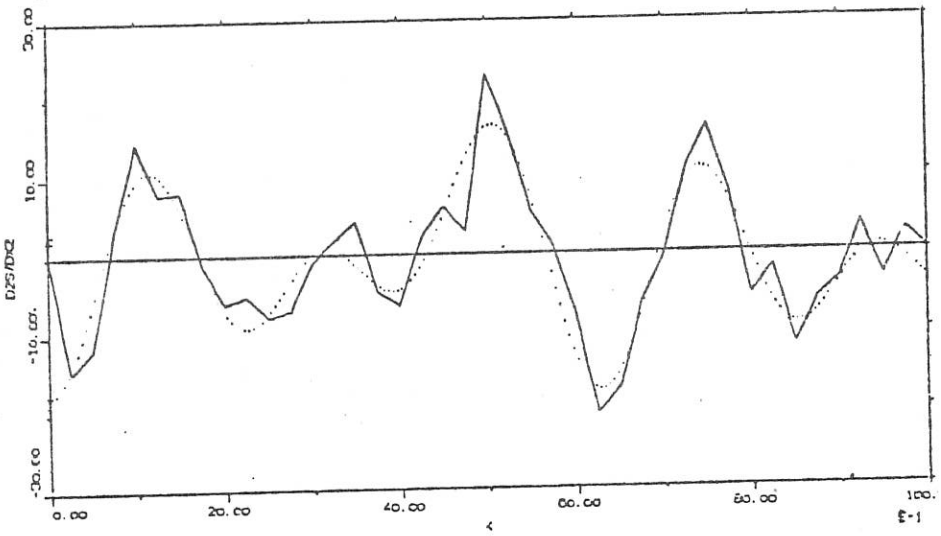


Abb.(4.2.3)

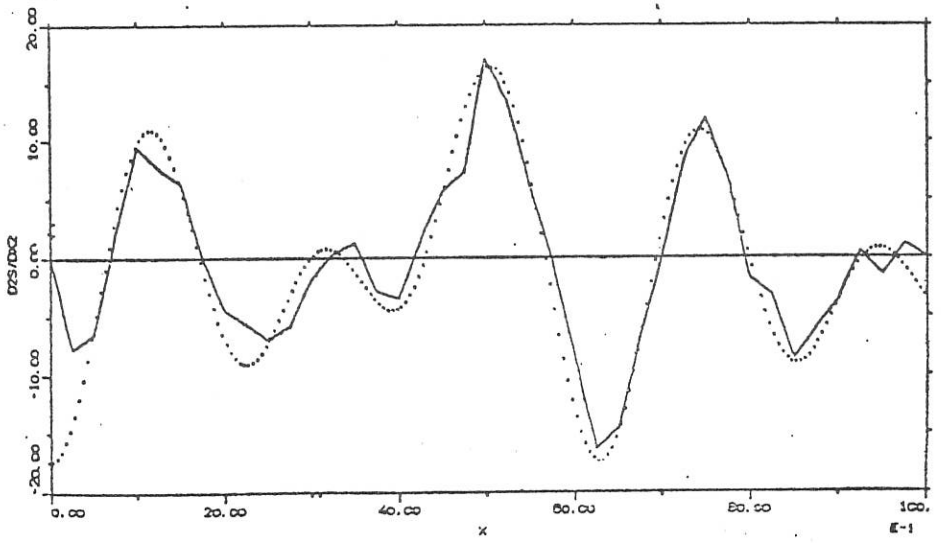
Vergleich der 2. Ableitungen von (4.2.1), n:= 40.

I - Wie erwartet zeigt die 2. Ableitung D2s der Kubischen Spline-Interpolierenden s keinerlei Übereinstimmung mehr mit der 2. Ableitung der Funktion f.

II - Obwohl D2s nicht mehr differenzierbar ist, rekonstruiert die 2. Ableitung des Ausgleichssplines D2f relativ
III gut.

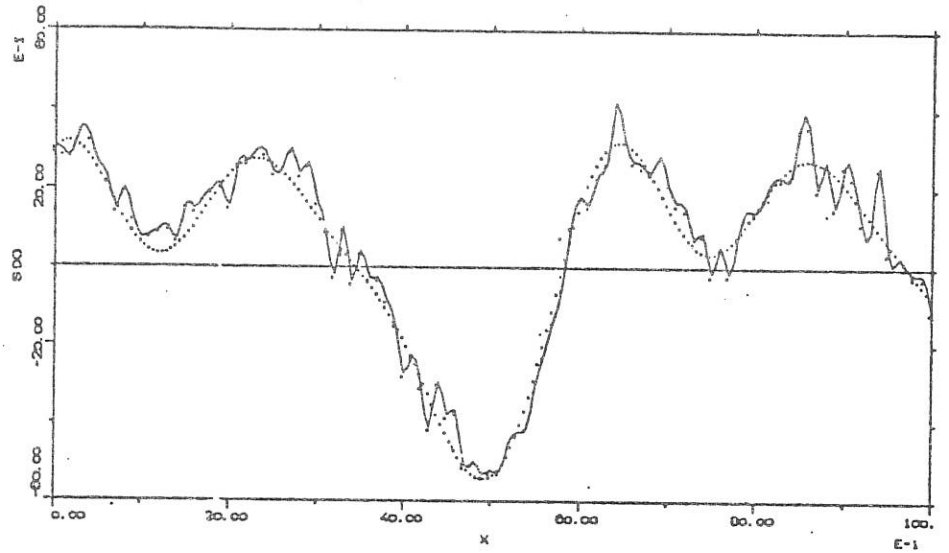
In dieser Darstellung sieht man sehr schön die Eigenschaft (2.2.1.1) bzw. (3.1.2.2) der Kubischen Splines. Da $s \in C^1$ ist D2s zwar stetig aber im allgemeinen nicht mehr differenzierbar.

III DELTAY = 0.00, DEV = 0.50 , MOD = 4



F(X):= 3*SIN(X)+2*SIN(2*X+PI/2)+COS(3*X)
+ N(0, 0.50)-RANDOM NUMBERS

I DELTAY = 0.00, DEV = 0.00 , MOD = -2



II DELTAY = 0.00, DEV = 0.50 , MOD = 4

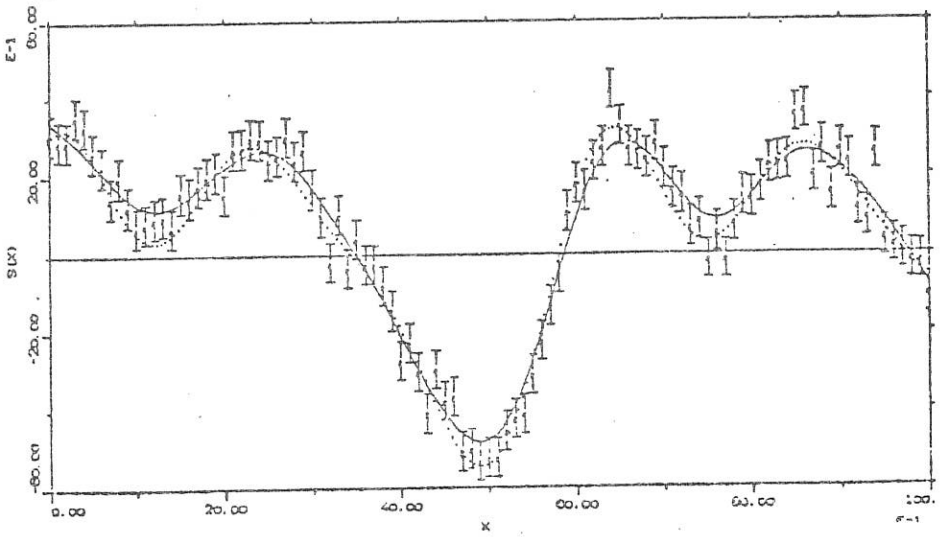


Abb.(4.2.4)

Approximation einer 'gestörten' Funktion mit Kubischen Interpolations-und Ausgleichssplines. Testfunktion f und Verteilung der Zufallszahlen sind wie in (4.2.1) gewählt.

Vergleich der Funktionswerte, n:= 100.

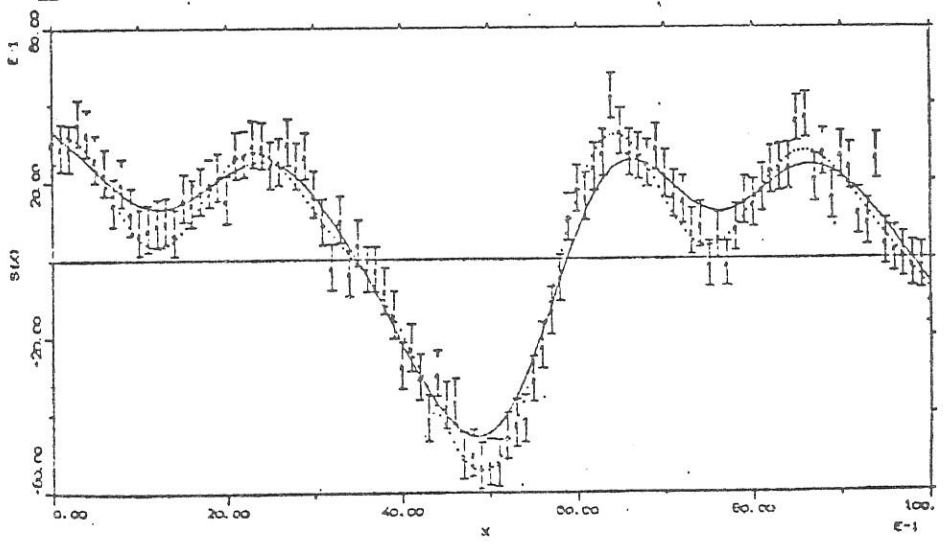
I - Kubische Spline-Interpolation der Datenpunkte mit Hermite-Randbedingungen (MODE:= -2).

II - Kubischer Ausgleichsspline nach (3.2) mit dem in (3.2) erklärten Messfehlermodell MODE:= 4 und dem Glättungsparameter DEVIAT:= 0.5. Nach (3.2) sind diese Parameter optimal fur diese Messfehlerverteilung.

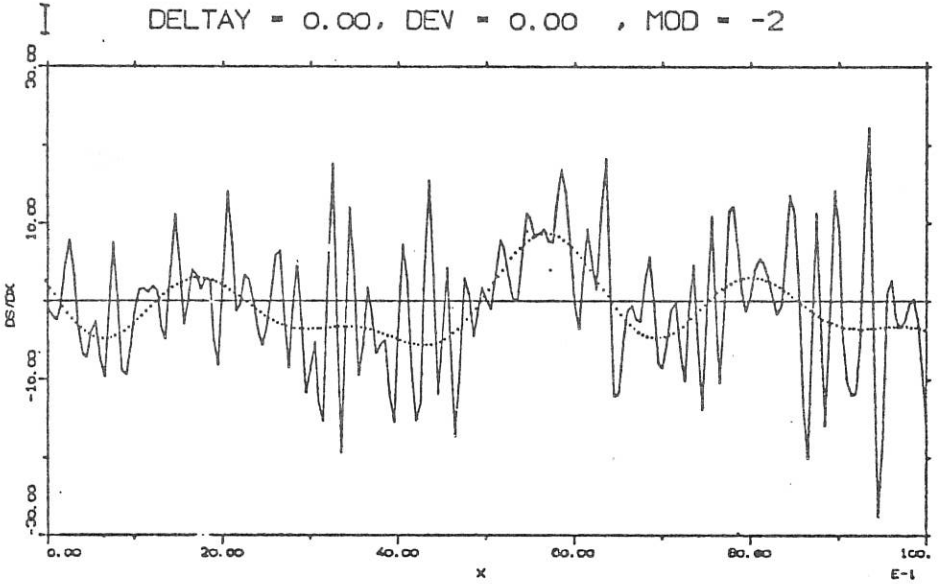
III - Wie II, aber DEVIAT:= 0.6.

Durch die Erhöhung der Datenmenge auf n:= 100 verschlechtern sich die Approximationseigenschaften der Ausgleichssplines nur geringfügig. Wie man deutlich sehen kann, verläuft der Ausgleichsspline s meistens sogar innerhalb des halben, mit 'I' bezeichneten und durch die Gewichte w(i) vorgegeben, Fehlerkorridors um die Datenpunkte y(i).

III DELTAY = 0.00, DEV = 0.60 , MOD = 4



F(X) := 3*SIN(X)+2*SIN(2*X+PI/2)+COS(3*X)
+ N(0, 0.50)-RANDOM NUMBERS



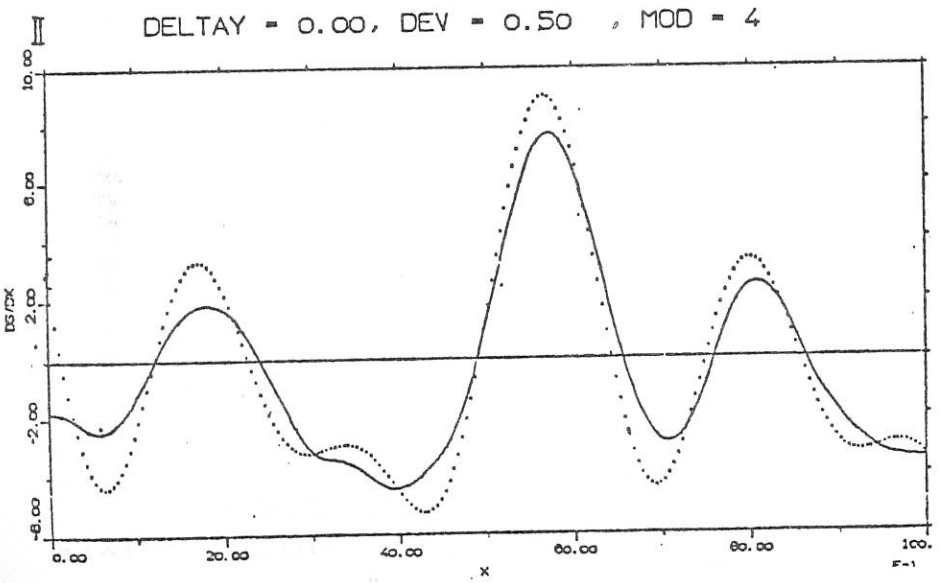


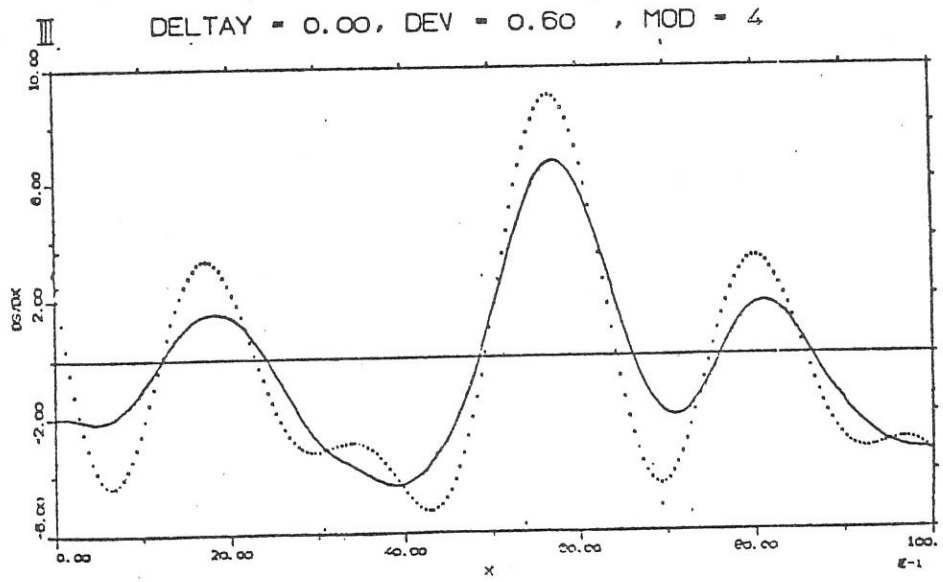
Abb.(4.2.5)

Vergleich der Ableitungswerte von (4.2.4), n:= 100.

I - Gegenüber (4.2.1) wird die Approximation der Ableitung Df durch den Interpolationsspline um etwa den Faktor 2 verschlechtert. Im Prinzip wird nur mehr die Ableitung der Störfunktion wiedergegeben.

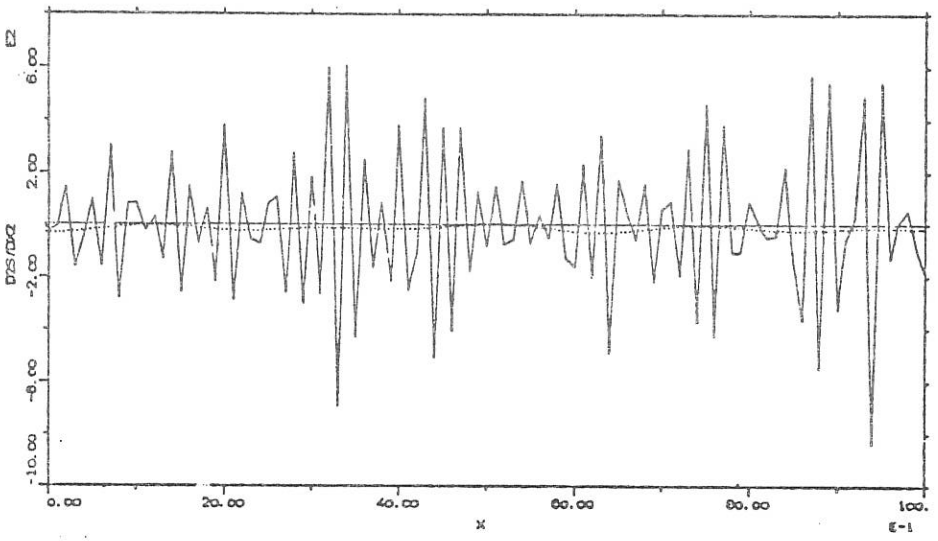
II - Vom Ausgleichsspline wird dagegen der Verlauf von Df sehr gut rekonstruiert. Gegenüber (4.2.2) ist keine Verschlechterung der Approximationsgüte erkennbar.

III - Eine leichte Vergrösserung der Gewichte gegenüber der theoretisch idealen Wahl der Glättungsparameter in II wirkt sich als leichte Verschlechterung in der Approximationsgüte aus.



F(X) := 3*SIN(X)+2*SIN(2*X+PI/2)+COS(3*X)
+ N(0, 0.50)-RANDOM NUMBERS

I DELTAY = 0.00, DEV = 0.00 , MOD = -2



II DELTAY = 0.00, DEV = 0.50 , MOD = 4

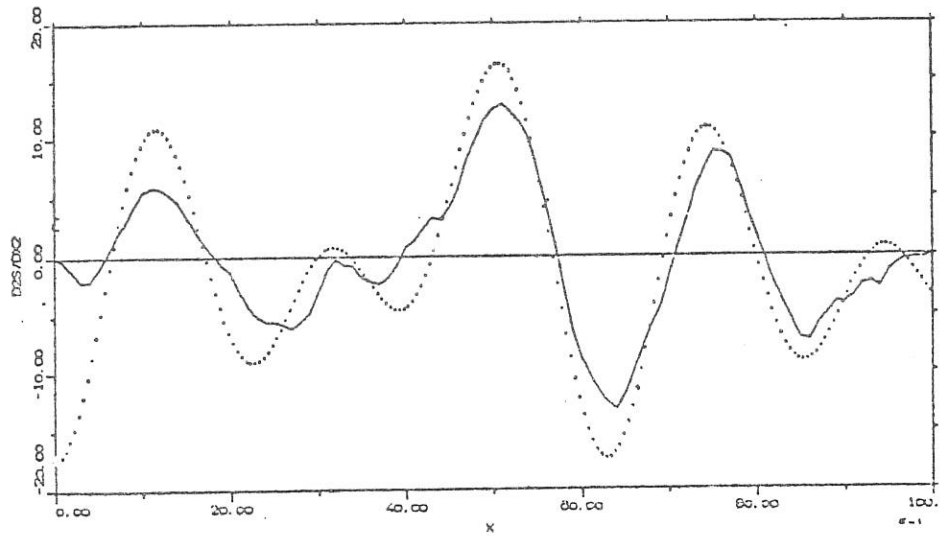


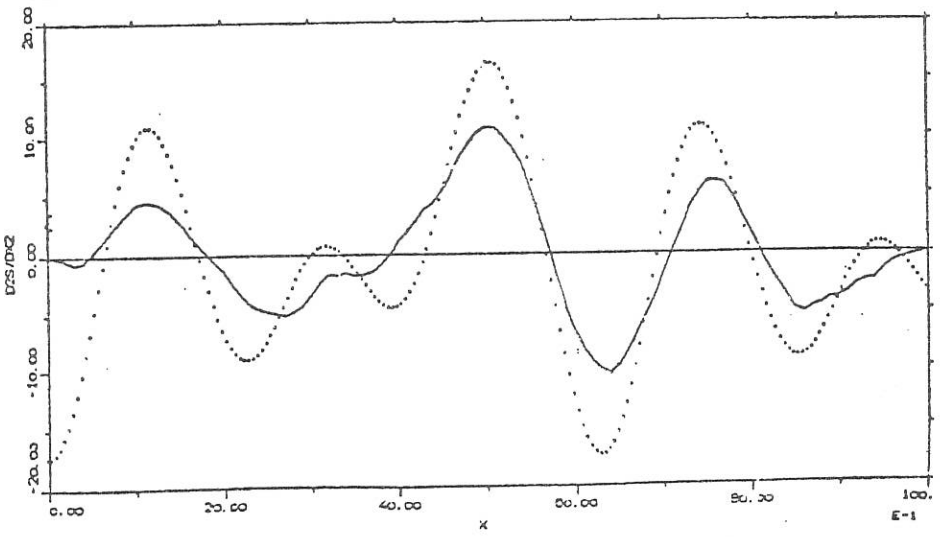
Abb.(4.2.6)

Vergleich der 2. Ableitungen von (4.2.4), n:= 100.

I - Die 2. Ableitung D2s des Kubischen Interpolationssplines s ist als Näherung an D2f völlig unbrauchbar. Die Differenz von D2s zu D2f beträgt im Schnitt etwa das 30-fache des Funktionswertes.

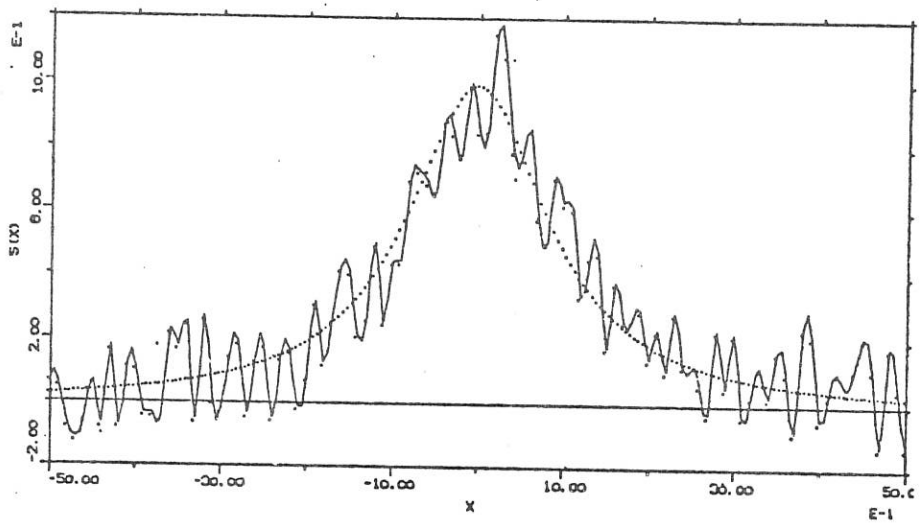
II - Die 2. Ableitungen D2s der Ausgleichssplines geben
III zumindest den groben Verlauf von D2f wieder. Die grössten Abweichungen treten hier an den Intervallenden t(1):= 0.0 und t(100):= 10.0 auf, da nach Definition (3.1.2.4) für die Endpunkte die Vorgaben D2s(0):= 0 und D2s(10.0):= 0 der Natürlichen Splines gelten.

III DELTAY = 0.00, DEV = 0.60 MOD = 4



F(X) := 1 / (1 + X*X)
+ U(+- 0.20)-RANDOM NUMBERS

I DELTAY = 0.00, DEV = 0.00 , MOD = -2



II DELTAY = 0.00, DEV = 0.00 , MOD = -2

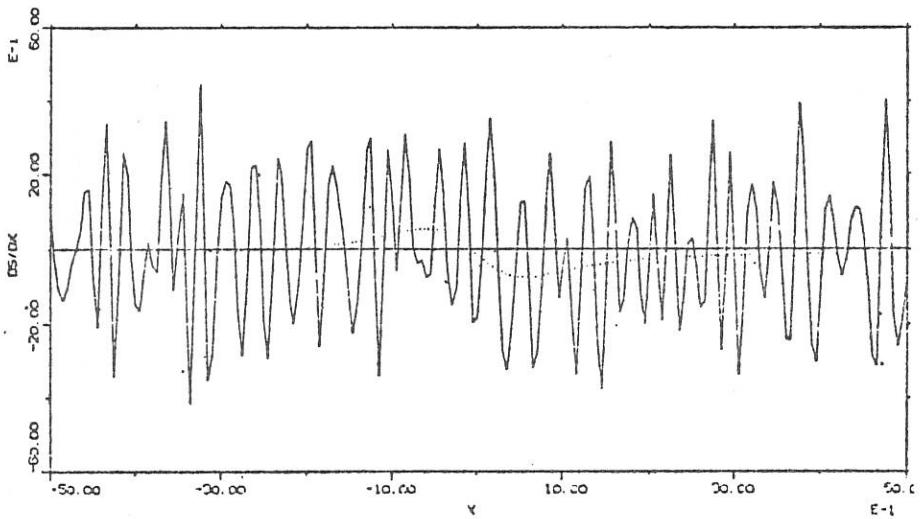


Abb.(4.2.7)

Kubische Spline-Interpolation einer stark 'gestörten' Funktion f.

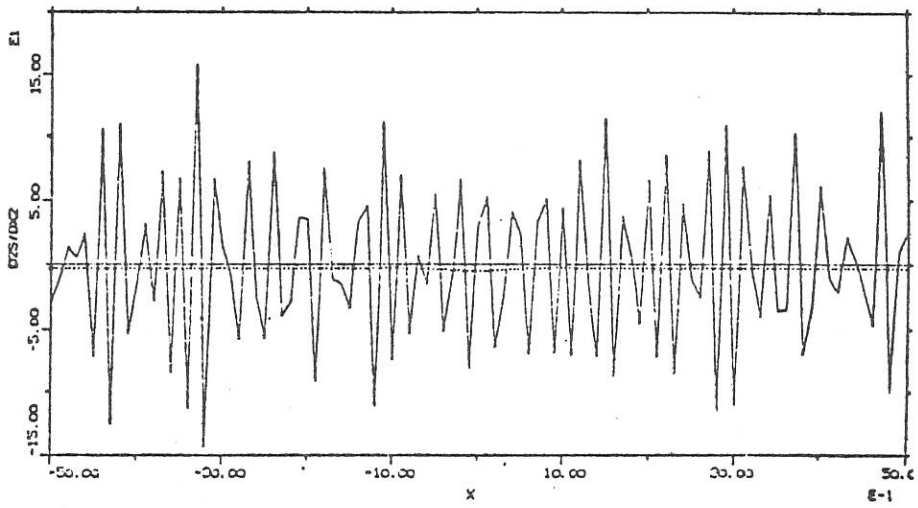
Die an n:= 100 aequidistanten Stützstellen gewonnenen Werte der beliebig oft differenzierbaren Funktion f wurden anschliessend durch Addition mit im Intervall [-0.20,0.20] Gleichverteilten Zufallszahlen gestört. Die künstliche Auslenkung der Funktionswerte entspricht hier in etwa einem mittleren Messfehler von 30%, wie er auch bei manchen Messverfahren angenommen werden muss.

I - Kubische Spline-Interpolation der künstlich verrauschten Daten.

II - Die Darstellung von Ds zeigt eindrucksvoll die Unbrauchbarkeit dieser Approximationsmethode bei so gearteten Daten.

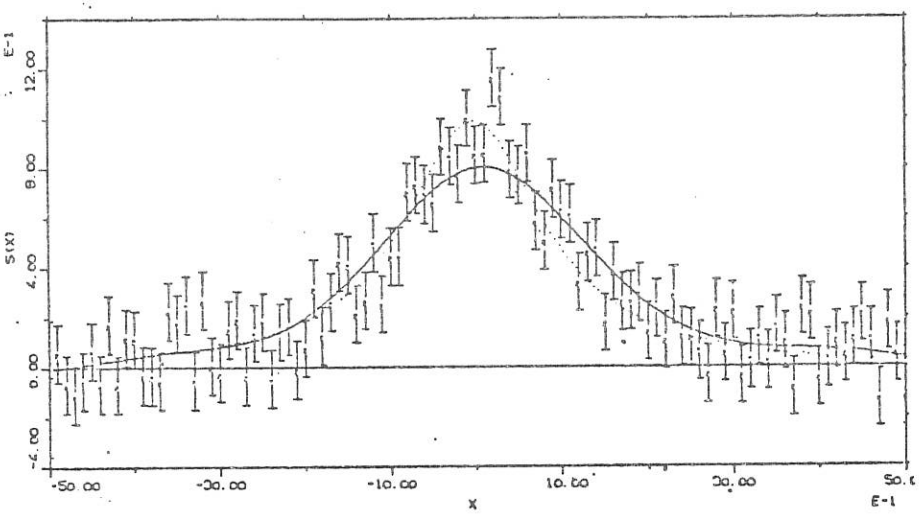
III - D2s gibt wie in (4.2.6) nur die 2. Ableitung der interpolierten 'Störfunktion' wieder.

III DELTAY = 0.00, DEV = 0.00 , MOD = -2



F(X) := 1 / (1 + X*X)
+ U(+- 0.20)-RANDOM NUMBERS

I DELTAY = 0.00, DEV = 0.20 , MOD = 2



II DELTAY = 0.00, DEV = 0.20 , MOD = 2

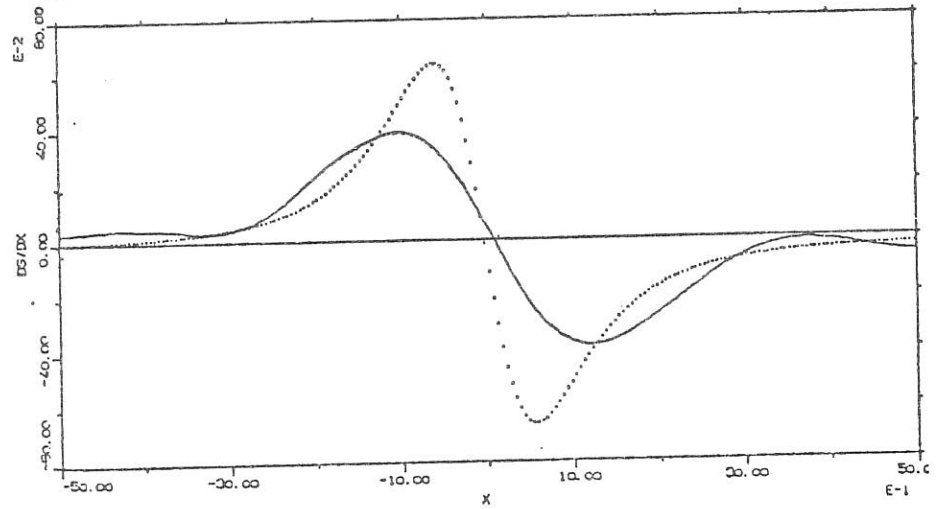


Abb.(4.2.8)

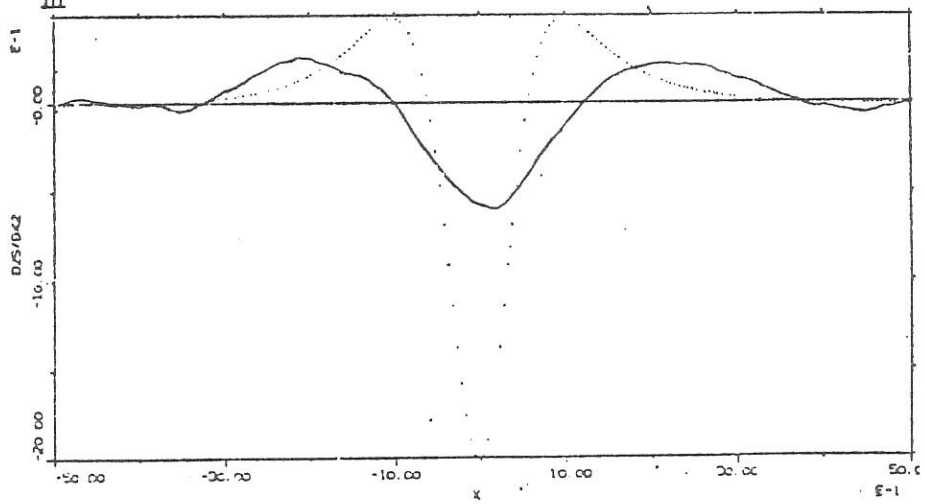
Approximation des Datenfeldes (4.2.7) mit einem Ausgleichsspline.

I - Die Glättungsparameter 'MODE' und 'DEVIAT' wurden nach der in (3.2) für diese Fehlerverteilung als optimal empfohlenen Grösse vorgegeben.

II - Trotz des relativ grossen Messfehlers wird die 2. Ableitung erstaunlich gut rekonstruiert. Beachtenswert ist die Erhaltung der Symmetrie.

III - Die Näherung D2s gibt noch relativ gut den Verlauf von D2f wieder. Selbst bei der Approximation der ungestörten Funktion f bietet die Darstellung von D2f in der Umgebung der 0 wegen der dortigen Steilheit einige Probleme.

III DELTAY = 0.00, DEV = 0.20 , MOD = 2



LITERATURVERZEICHNIS

[BÖH] Böhmer, K. : Spline-Funktionen
B. G. Teubner, Stuttgart, 1974

[CUF] Cuthbert, D. : Fitting Equations to Data
Fred, S. Wiley-Interscience,
Toronto, 1971

[LIN] Linz P. : Theoretical numerical analysis
John Wiley and Sons, New York, 1978

[RE1] Reinsch, Ch. : Smoothing by Spline Functions I
Numerische Mathematik 10,
177 - 183 (1967)

[RE2] ---------- : Smoothing by Spline Functions II
Numerische Mathematik 16,
451 - 454 (1971)

[RIC] Rice, J./ : Integrated Mean Squared Error of a
Rosenblatt, M. Smoothing Spline
Journal of Approximation Theory 33,
353 - 369 (1981)

[ROS] Rosenblatt, M.: Asymptotics and Representation of
Cubic Splines
Journal of Approximation Theory 17,
332 - 343 (1976)

[STO] Stoer, J. : Einführung in die Numerische
Mathematik I
Springer Verlag, Berlin, Heidelberg,
New York 1979

[VIE] Viergutz, H. : Interpolationsärgernisse
Technische Universitat Hannover,
Report Nr. 58, 1977

[WLO] Wloka, J. : Funktionalanalysis und Anwendungen
Walther de Gruyter & Co., Berlin 1971