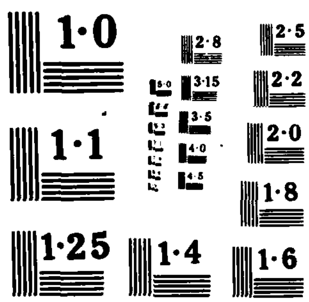
AD-A158 049 A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES(U) DEFENCE RESEARCH ESTABLISHMENT ATLANTIC DARTMOUTH (NOVA SCOTIA) D HALLY JUN 85 DREA-TM-85/205 1/1

UNCLASSIFIED F/G 9/2 NL

END
DATE
FILMED
10 85


1·0
2·8
2·5
3·15
2·2
5·0
3·5
1·1
4·0
2·0
4·5
1·8
1·25
1·4
1·6

(1)

National Defence
Research and Development Branch

Défense Nationale
Bureau de Recherche et Développment

AD-A158 049

TECHNICAL MEMORANDUM 85/205
JUNE 1985

A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES

David Hally

DTIC FILE COPY

Defence Research Establishment Atlantic

Centre de Recherches pour la Défense Atlantique

Canadä

JW

TERMS OF RELEASE

THIS DOCUMENT CONTAINS PROPRIETARY INFORMATION. IT IS FURNISHED IN CONFIDENCE AND WITH THE EXPRESS UNDERSTANDING THAT PROPRIETARY AND PATENT RIGHTS WILL BE RESPECTED.

PLEASE ADDRESS FURTHER ENQUIRIES TO:

THE CHIEF,
DEFENCE RESEARCH ESTABLISHMENT ATLANTIC,
P.O.BOX 1012, DARTMOUTH, N.S., B2Y 3Z7
CANADA.

A-1

DEFENCE RESEARCH ESTABLISHMENT ATLANTIC
9 GROVE STREET
P.O. BOX 1012
DARTMOUTH, N.S.
B2Y 3Z7

TELEPHONE
(902) 426-3100

CENTRE DE RECHERCHES POUR LA DÉFENSE ATLANTIQUE
9 GROVE STREET
C.P. 1012
DARTMOUTH, N.É.
B2Y 3Z7

(6)

National Defence
Research and Development Branch

Défense Nationale
Bureau de Recherche et Développment

AD-A158 049

TECHNICAL MEMORANDUM 85/205
JUNE 1985

A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES

David Hally

DTIC FILE COPY

Defence Research Establishment Atlantic

Centre de Recherches pour la Défense Atlantique

Canada

85 8 15 029

JW

TERMS OF RELEASE

THIS DOCUMENT CONTAINS PROPRIETARY INFORMATION. IT IS FURNISHED IN CONFIDENCE AND WITH THE EXPRESS UNDERSTANDING THAT PROPRIETARY AND PATENT RIGHTS WILL BE RESPECTED.

PLEASE ADDRESS FURTHER ENQUIRIES TO:

THE CHIEF,
DEFENCE RESEARCH ESTABLISHMENT ATLANTIC,
P.O.BOX 1012, DARTMOUTH, N.S., B2Y 3Z7
CANADA.

A-1

DEFENCE RESEARCH ESTABLISHMENT ATLANTIC
9 GROVE STREET
P.O. BOX 1012
DARTMOUTH, N.S.
B2Y 3Z7

TELEPHONE
(902) 426-3100

CENTRE DE RECHERCHES POUR LA DÉFENSE ATLANTIQUE
9 GROVE STREET
C.P. 1012
DARTMOUTH, N.E.
B2Y 3Z7

National Defence
Research and Development Branch

Défense Nationale
Bureau de Recherche et Développment

A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES

David Hally

JUNE 1985

Approved by B.F. Peters A/Director/Technology Division

DISTRIBUTION APPROVED BY

A/D/TD

TECHNICAL MEMORANDUM 85/205

Defence Research Establishment Atlantic

Centre de Recherches pour la Défense Atlantique

Canada

Abstract

~~DREA~~ has, at present, two libraries containing subroutines for calculating splines: IMSL and BSPLIN. A new library has been developed to supplement the IMSL and BSPLIN routines in the realm of smoothing splines. It is not self-contained, making frequent use of subroutines from the BSPLIN library.

The new subroutines offer several advantages over the smoothing spline subroutines in the IMSL and BSPLIN libraries:

1) The order of the spline may be picked by the user;

2) The second derivative of the spline is not constrained to be zero at its end-points;

3) The user of the new subroutines has freedom to choose the number and positions of the knots of the spline; and

4) The new subroutines have, as input, an extra set of weights, δ_i, i=1,N, which control the stiffness of the spline between each pair of knots.

The new subroutines were initially developed for use in ship hull approximation for the calculation of boundary layer growth on the hull. For this calculation one needs splines whose second derivatives are very well behaved. The additional control afforded by the new subroutines makes them far more suitable for this application than any of the subroutines currently available in either the IMSL or BSPLIN libraries.

Rèsumè

L'ERDA possède maintenant deux bibliothéques contentant des sous-programmes pour calculer des splines : IMSL et BSPLIN. Une nouvelle bibliothèque aètè mise sur pied pour compléter les programmes ISML et BSPLIN dans le domaine des splines de lissage. Elle n'est pas autonome, faisant souvent appel à des sous-programmes de la bibliothèque BSPLIN.

Les nouveaux sous-programmes offrent plusieurs avantages par rapport aux sous-programmes de splines de lissage des bibliothèques IMSL et BSPLIN.

(1) L'utilisateur peut choisir le degré de la spline.

(2) La deuxième derivée de la spline n'est pas forcément nulle à ses points extrêmes.

(3) L'utilisateur des nouveaux sous-programmes peut choisir le nombre et le lieu des noeuds de la spline.

(4) Les nouveaux sous-programmes acceptent en entrée un ensemble supplémentaire de coefficients de pondération δ_i, $i=1, N$, qui déterminent la raideur de la spline entre deux noeuds.

Les nouveaux sous-programmes ont intialement été mis au point pour l'approximation des coques de navire, notamment pour le calcul de la croissance des couches limites sur les coques. Pour ce dernier calcul, il faut utiliser des splines dont la deuxième dérivée est parfaitement définie. Par le contrôle accru qu'ils offrent, les nouveaux sous-programmes conviennent beaucoup mieux à cette application que tous les sous-programmes existants des bibliothèques IMSL ou BSPLIN.

Table of Contents

NOTATION

$B_{n,k}$	- The n^{th} B-spline of order k (Section 2.1)
$D_{nj}^{(m)}$	- The m^{th} divided difference operator (Section 3)
e_n	- Error of the n^{th} data point (Section 2.1)
$f(x)$	- The spline function (Section 2.1)
F	- The smoothing functional as used by Reinsch and de Boor (Section 2.1)
F^*	- The smoothing functional as used in BSMTH (Section 2.1)
$g_n^{(m)}$	- See equation (3.9)
G	- $pX^2 + (1-p)F$ (Section 2.1)
G^*	- $pX^2 + (1-p)F^*$ (Section 2.1)
k	- The order of the spline (Section 2.1)
m	- The derivative of the spline function used as a smoothing criterion (Section 2.1)
N	- The number of B-splines used in the spline (Section 2.1)
N_k	- The number of knots (Section 2.1)
N_p	- The number of data points (Section 2.1)
p	- Parameter which balances the relative values of F and X^2 (Section 2.1)
p_{hi}	- Value of p used in the iteration for p in BSMTH (Section 2.5)
p_{lo}	- Value of p used in the iteration for p in BSMTH (Section 2.5)
p_{hi}	- The value of p after the n^{th} iteration for p in BSMTH (Section 2.5)
S	- The value of the X^2 input by the user (Section 2.5)
s_n	- The arc length to the n^{th} data point (Section 5)

v_n - See equation (2.10)

v^*_n - See equation (2.19)

w_n - Error weights defined in equation (3.2)

x_n - Abscissa of the n^{th} data point (Section 2.1)

Y^2 - See equation (2.11)

Y^{*2} - See equation (2.20)

y_n - Ordinate of the n^{th} data point (Section 2.1)

y^*_n - See equation (2.21)

α - Parameter which balances the relative values of F and X^2 (Section 2.1)

β_n - The n^{th} spline coefficient (Section 2.1)

β^*_n - Approximation to β_n used for numerically stable determination of the X^2 of a spline (Section 2.4)

δ_n - The stiffness weight corresponding to the interval between the $(n+k-1)^{th}$ and the $(n+k)^{th}$ knot (Section 2.1)

δ_{nj} - The Kronecker delta (Section 3)

ϵ_n - The actual error of the n^{th} data point (Section 3)

σ - See equation (3.3)

X^2 - The chi-square of the spline (Section 1)

X^2_{hi} - The chi-square of the spline corresponding to the p-value p_{hi} (Section 2.5)

X^2_{lo} - The chi-square of the spline corresponding to the p-value p_{lo} (Section 2.5)

X^2_n - The chi-square of the spline corresponding to the p-value p_n (Section 2.5)

1 INTRODUCTION

DREA has, at present, two libraries containing subroutines for calculating splines: BSPLIN[1] and IMSL[2]. The library of subroutines presented here is intended to supplement the previous two. It is not self-contained, making frequent use of BSPLIN subroutines.

The subroutines presented here were developed because the smoothing spline routines available in IMSL and BSPLIN were found inadequate for smoothing data digitized from offset diagrams of ship hulls. The spline representations of the hulls were to be used in the calculation of hull boundary layer growth. For this application, it is necessary to have a spline representation of the hull whose second derivatives are very well behaved. The second derivatives of the hull representation cause accelerations in the fluid flow around the hull which in turn cause changes in the boundary layer growth. It was found that the spline subroutines in the IMSL and BSPLIN libraries could not be controlled sufficiently well that the boundary layer calculations would be unaffected by splining errors. In particular, the splines were unable to turn sharp corners (near the bilge, for example) sufficiently rapidly without either cutting the corner or having 'wiggles' on each side of the corner. Either result induced large errors in the second derivatives of the spline, the former underestimating the magnitudes of the second derivatives, the latter overestimating them. It was therefore necessary to develop new subroutines providing greater control over the splines and their derivatives.

The most fundamental subroutine in the new library is BSMTH. It is very similar in function to the IMSL subroutine ICSSCU (this is an implementation of a program originally written by Reinsch[3]) and the BSPLIN subroutine SMOOTH: given the χ^2 of the spline curve with respect to given data, a smooth spline approximating the data is determined by minimizing a functional which measures the 'lack of smoothness' of the spline. BSMTH, however, offers several advantages over the other two subroutines.

1) The order of the spline may be picked by the user. SMOOTH and ICSSCU are cubic splines only.

2) SMOOTH and ICSSCU constrain the second derivative of the spline to be zero at its end-points. BSMTH imposes no such constraint.

3) The user of BSMTH has freedom to choose the number and positions of the knots of the spline. SMOOTH and ICSSCU require exactly one knot at each data point. The freedom to choose the knots allows much greater control of the spline.

   When splining in two dimensions, control of the knots has additional consequences. For efficient approximation of two-dimensional data, the knots must form a rectangular lattice (see Reference 1, chapter 17, for example). ICSSCU and SMOOTH then require the data points to be in a rectangular lattice. With BSMTH this is no longer necessary.

4) BSMTH has, as input, an extra set of weights, δ_i, i=1,N, which control the stiffness of the spline between each pair of knots. If the spline is required to be flat in some region, then the appropriate δ_i is increased. If the spline is to bend sharply in a different region, the appropriate δ_i is

= 1, if P is to be recalculated.

Via COMMON / PLIMS /

PMIN = Minimum allowed value of p (See Section 2.5). Default is 1.0E-03

PMAX = Maximum allowed value of p. Default is 1.0E+03.

Via COMMON /NODFLT/

IMAX : 2*IMAX is the maximum number of divided differences allowed to find the error in function PRERR (See Section A.4). Default value is 5.

SMFACT: The value of the smoothing parameter used by BSMTH may be adjusted by using a value of SMFACT not equal to 1. The smoothing parameter used is, S = SMFACT*NPT*PRERR**2. The default value is 1.

Via COMMON /INTEXP/

JDER : The value of JDER used by BSMTH. The integral of the square of the JDERth derivative of the spline is minimized (subject to the constraint that XSQ = S). If smooth curves are desired a value of JDER = 2 is appropriate. JDER should be non-negative and less than K. The default value is 2.

DEFAULTS

If IER = 0 on input then
JDER = 2
SMFACT = 1.0
IMAX = 5
T(I) = (I-K)/(N-K+1), I=1,NKT i.e. knots are uniformly distributed in (0,1)

If IER = 1 on input, then the vales for JDER, SMFACT, IMAX and T(I) must be input by the user via the COMMON blocks /NODFLT/ and /INTEXP/.

OUTPUT

IER = 0, Calculation has been successful

= 1, If JDER > K - 1

= 2, If NKT1 < N + K + max(0,K-2*JDER)

= 3, If IWK < max(NKT1,K**2)

= 4, If more than 30 iterations are required to find the correct value for p in BSMTH when splining the data point abscissae. Indicates numerical difficulties in the solution of the linear system

Appendix A

USER'S GUIDES

Concise guides for the use of the spline subroutines are now given. The subroutines are listed alphabetically.

A.1 BSMCRV : User's Guide

SUBROUTINE BSMCRV(NPT,X,Y,E,N,K,NKT1,T,WTI,BCOEFX,BCOEFY,R,IWK,WK,ARCL,G,IER)

PURPOSE: Given data points (X(I),Y(I)), I=1,NPT BSMCRV *finds a smooth curve* approximating them by splining the abscissae and ordinates separately with respect to the arc-length along the spline. The arc length at each point is approximated from the distances between the points. BSMTH is used to spline the abscissae and the ordinates. The function PRERR is used to determine the smoothing parameter and the subroutine WTIBEG is used to determine the stiffness weights.

LANGUAGE: FORTRAN

USAGE: EXECUTE mainpgm,BSPLIN:HLLYSP/LIB,BSPLIN:BSPLIN/LIB

CALLS subroutines BSMTH, PRERR, WTIBEG

INPUT

NPT	:	The number of data points.
X	:	An array of length NPT containing the data point abscissae in ascending order.
Y	:	An array of length NPT *containing* the data point ordinates.
E	:	The errors of the data points. The smaller the error the closer the spline will come to that point.
N	:	The number of B-splines used to represent the spline.
K	:	The order of the spline.
NKT1	=	N + K + max(0,K-2*JDER) (see below for a definition of JDER)
T	:	An array of length NKT1 the first N+K elements of which contain the knot sequence (in ascending order). The remaining array elements are used in subroutine SETUPP.
IER	=	0, if JDER, T, WTI and the first N*K elements of R are as on the previous call to BSMTH (this means that the matrix P need not be recalculated).

5 A PARAMETRIC SMOOTHING SPLINE

It is often desired to approximate data by a smooth curve which is not necessarily a function. The spline approximation must be parametrized in some way. The choice of the parametriztion is important (see Reference 1, pp.316). It has been shown that any approximation of the arc length of the curve provides a good parametrization. It is usually sufficient to approximate the arc length from the distance between data points. The parametric spline is then calculated as follows.

1) Calculate the parameter s, at each data point by

$$s_n = s_{n-1} + ((y_n - y_{n-1})^2 + (y_n - y_{n-1})^2)^{1/2} \qquad (5.1)$$

2) Spline each of the data sets $\{(s_n,x_n),n=1,N\}$ and $\{(s_n,y_n),n=1,N\}$.

This is perfomed in the subroutine BSMCRV, which uses BSMTH to calculate each of the two sub-splines. Hence, BSMCRV calculates a smooth, parametric spline. The smoothing parameter for the calls to BSMTH is determined by the function PRERR and the stiffness weights are determined by the subroutine WTIBEG. In addition, the arc-length is normalized by the total length of the curve: that is, the parameter used is not the arc-length but the fractional arc length along the curve. Thus the parameter s varies between 0 and 1.

An example of a spline generated by BSMCRV is shown in Figure 11. Although the data points show a large amount of scatter, an excellent, smooth curve has been found to fit the data. Notice that the crossing of the curve over itself is of no consequence to BSMCRV.

6 CONCLUDING REMARKS

The computer subroutines presented in this memorandum extend the available libraries of spline subroutines at DREA. The versatility of BSMTH in comparison with the BSPLIN subroutine SMOOTH and the IMSL subroutine ICSSCU , make it suitable for use with a far greater variety of data sets. In particular, the ability to choose the spline order, the ability to vary the spline knots independent of the data points, and the ability to change the 'stiffness' of the spline at specific locations via the stiffness weights, δ_n, allow the user far greater control over the spline than is possible with SMOOTH or ICSSCU. Nor need the choice of inputs for BSMTH be overly difficult. The subroutines PRERR, WTIBEG and NEWWTI allow the user to generate reasonable sets of default values for the smoothing factor, S, and the stiffness weights, δ_n, input to BSMTH. Finally, the restriction that the data points be splined by a function is relaxed if one chooses to use the subroutine BSMCRV. Thus, the subroutine library provides a smoothing spline which provides, at once, both ease of use and great freedom and flexibility.

4 DEFAULT VALUES FOR THE STIFFNESS WEIGHTS

As with the smoothing parameter, it is often not convenient for the user to input the values for the stiffness weights, δ_n, $n = 1,N-k+1$. Two subroutines are provided which calculate reasonable values for the parameters. The first subroutine, WTIBEG, uses the data points to calculate the δ_n. The second, NEWWTI, uses the spline coefficients of a previously spline approximation of the data to calculate new values for δ_n.

Both subroutines use the same principle. Default values for the δ_n are chosen by setting δ_n equal to a predicted value for

$$\int_{t_n}^{t_{n+1}} \left[\frac{d^m f(x)}{dx^m}\right]^2 dx$$

The contributions from each knot interval to the functional F* are then nearly equal and the smoothing will not be dominated by one short segment of the curve.

In WTIBEG, it is assumed that m = 2. The second derivative of the spline in any knot interval may then be approximated by the second partial difference between data points near that knot interval. That is, if $x_{j-1} < x < x_{j+1}$ and $t_n < x < t_{n+1}$ then

$$f''(x) \approx \frac{1}{x_{j+1} - x_{j-1}}\left[\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}\right] \tag{4.1}$$

NEWWTI uses a previous spline approximation of the data to approximate the integral of the m^{th} spline derivatives in any knot interval. The m^{th} derivative of the spline is calculated at each of the knots and the integral approximated from the linear interpolation of these values. This yields the formula

$$\int_{t_{n+k-1}}^{t_{n+k}} [f^{(m)}(x)]^2 dx \approx ⅓(t_{n+k}-t_{n+k-1})([f^{(m)}(t_{n+k})]^2 + f^{(m)}(t_{n+k})f^{(m)}(t_{n+k-1}) + [f^{(m)}(t_{n+k-1})]^2) \tag{4.2}$$

If the $k \leq m+2$, this method is exact since the m^{th} derivative of the spline is then linear between the knots.

A demonstration of the ability of WTIBEG to choose appropriate choces for the stiffness weights is shown by the comparison of the splines in Figures 1 and 2. As explained in Section 2.6, the only effective difference in the calculations of these two splines is the variation in the stiffness weights.

If f is suitably smooth, then the first term on the right side of equation (3.8) remains small as m increases, while the second term increases rapidly. Thus, for sufficiently large m and N,

$$\sum_{j=1}^{N} D_{nj}^{(m)} y_j \approx \sum_{j=1}^{N} D_{nj}^{(m)} \epsilon_j \approx \sum_{j=1}^{N} D_{nj}^{(m)} \langle \epsilon_j \rangle = \sum_{j=1}^{N} D_{nj}^{(m)} e_j \sigma^2 \equiv g_n^{(m)} \sigma^2 \tag{3.9}$$

so that an estimate for σ^2 is

$$\sigma^2 \approx \frac{1}{N-m} \sum_{n=1}^{N-m} \left(\sum_{j=1}^{N} D_{nj}^{(m)} y_j \right)^2 / g_n^{(m)} \tag{3.10}$$

$D_{nj}^{(m)} e_j$ is easily calculated from

$$D_{nj}^{(m)} e_j = \sum_{k=1}^{N} D_{nk}^{(m)} \delta_{kj} e_j \tag{3.11}$$

That is, $D_{nj}^{(m)} e_j$ is the m^{th} divided difference of the data set $\{0,0,\ldots,e_j,\ldots 0,0\}$.

Thus, in order to estimate σ^2, and hence S, it is only necessary to have a method for determining a sufficiently large m. The domination of the divided differences by the errors is characterized by a large number of changes in sign between $\sum_{j=1}^{N-m} D_{nj}^{(m)} y_j$ and $\sum_{j=1}^{N-m} D_{n+1,j}^{(m)} y_j$. If dominated by the errors, these values should be distribute randomly so that, on average, one expects (N-m)/2 sign changes. Smooth data should have far fewer. The number of sign changes in the divided differences is therefore used as a criterion for determining when the error is dominant.

Figures 8, 9 and 10 demonstrate the ability of PRERR to calculate appropriate smoothing parameters. Figure 8 shows a data set obtained from measurements of the variation of sound speed with depth in the Atlantic Ocean, as splined by an ordinary cubic spline (the subroutine CUBSPL from the BSPLIN library was used). Figure 9 shows the same spline with the data points removed so that the curve may be seen more easily. It can be seen that the curve is not smooth, especially near x = 15. Figure 10 shows the same data splined using BSMTH with the smoothing parameter calculated by PRERR. The fit to the data is still excellent but the spline is now smooth.

If the relative magnitudes of the e_n accurately reflect the errors of the data collection process, then averaged over a large number of data sets the average values of each w_n will be equal.

$$\langle w_n \rangle = \sigma \quad \text{for all } n \tag{3.3}$$

Here angle brackets denote averaging over an ensemble of similar data sets.

Since f(x) is assumed to be a smooth, well-behaved curve, it should be possible to fit a spline curve to it with high accuracy. Hence, the χ^2 of the "best" spline is

$$\chi^2 = \sum_{n=1}^{N_p} \frac{\epsilon_n^2}{e_n^2} = \sum_{n=1}^{N_p} w_n^2 \approx N\sigma^2 \tag{3.4}$$

if it may be assumed that the errors in the data points are uncorrelated and that N_p is sufficiently large.

Let $\{g_j, j=1,N\}$ be any set of numbers. The m^{th} divided difference of $\{g_j\}$ is a linear transformation of the g_j defined iteratively by

$$D_{nj}^{(0)} = \delta_{nj} \tag{3.5}$$

$$\sum_{j=1}^{N} D_{nj}^{(m)} g_j = \sum_{j=1}^{N} \frac{(D_{n+1,j}^{(m-1)} - D_{n,j}^{(m-1)}) g_j}{x_{n+m} - x_n} \quad , \quad n = 1,N-m \tag{3.6}$$

where δ_{nj} is the Kronecker delta. By the Mean Value Theorem, if f(x) is a C^m function, then for any $\{x_j, j=1,N\}$ there is a ξ in (x_n, x_{n+m}) such that

$$\sum_{j=1}^{N} D_{nj}^{(m)} f(x_j) = \frac{f^{(m)}(\xi)}{m!} \tag{3.7}$$

Thus, from equation (3.1) one obtains

$$\sum_{j=1}^{N} D_{nj}^{(m)} y_j = \frac{f^{(m)}(\xi)}{m!} + \sum_{j=1}^{N} D_{nj}^{(m)} \epsilon_j \tag{3.8}$$

order of the spline is 4 and the smoothing exponent m is 2. Hence, the only difference between this spline and the spline calculated by SMOOTH arises from the effect of the stiffness weights. These weights have been decreased near $x = 8$ and $x = 15$ to allow the spline to bend rapidly there. The stiffness has also been increased between $x = 10$ and $x = 12$ to flatten the top of the curve. Notice the absence of wiggles. These stiffness weights were determined by the subroutine WTIBEG (See Section 4 and Appendix A.5).

For the spline shown in Figure 3, the order of the spline was increased to 6. In Figures 4 and 5, the second derivative of the SMOOTH spline and the sixth order BSMTH spline are shown, respectively. Since the SMOOTH spline is necessarily of fourth order, its second derivative is piecewise linear.

The spline shown in Figure 6 was obtained by decreasing the spline order to 3, and reducing the knots as shown. At the positions of the double knots, the spline need no longer have continuous derivative. Splines with discontinuous derivatives cannot be obtained from SMOOTH or ICSSCU. For certain data sets they are necessary to obtain an accurate fit: for example, when splining a ship hull with a chine. The spline shown in Figure 7 carries this idea one step further. At the triple knots, the spline is no longer continuous at all. While a use for a completely discontinuous spline may not be evident, this example does serve to illustrate the versatility of the subroutine BSMTH.

3 CALCULATION OF INPUT VALUES FOR THE SPLINE X^2

The smoothness of the splines determined by BSMTH, SMOOTH and ICSSCU is regulated by the input parameter S, the value of the X^2 of the resulting spline. It is often not convenient for the user to supply this input parameter, nor is an appropriate value likely to be known. In this section an algorithm is described which yields an appropriate value for the parameter S, given the set of data points to be splined and their associated errors and assuming that the errors are uncorrelated. Since statistical methods are used, the algorithm works best when there are more than 15 data points. The algorithm is implemented in the function subroutine PRERR.

Let (x_n,y_n), $n = 1,N_p$ be the data points and e_i, their associated errors. It is assumed that the data may be derived from some unknown "smooth" curve, $f(x)$, so that

$$y_n = f(x_n) + \epsilon_n \qquad (3.1)$$

ϵ_n is the actual error of the n^{th} data point. This must not be confused with e_n, which is the error of the n^{th} data point estimated by the collector of the data. The e_n are known ; the ϵ_n are not.

The actual errors ϵ_n may be expressed

$$\epsilon_n = w_n e_n \qquad (3.2)$$

2) $S < X^2_1$: p_1 is too high. Therefore, set $p_{hi} = 1$, $X^2_{hi} = X^2_1$ and $p_2 = p_{min}$. After X^2_2 is determined there are, again, two possibilities:

i) $S > X^2_2$: p_2 is too low. Set $p_{lo} = p_{min}$ and $X^2_{lo} = X^2_2$. p_3 is now determined such that (p_3,S) lies on the straight line interpolating (p_{lo}, X^2_{lo}) and (p_{hi},X^2_{hi}).

ii) $S < X^2_2$: p_2 is too high. However, p cannot be decreased below p_{min}. Therefore, the iteration terminates.

b) Once (p_{lo},X^2_{lo}), and (p_{hi},X^2_{hi}) have been determined the iteration proceeds as follows:

1) If $|S - X^2_n| < S/10$, the iteration terminates.

2) If $X^2_n - X^2_{lo} > X^2_{hi} - X^2_n$, then p_{n+1} is determined such that (p_{n+1},S) lies on the straight line interpolating (p_n,X^2_n) and (p_{hi},X^2_{hi}).

3) If $X^2_n - X^2_{lo} < X^2_{hi} - X^2_n$, then p_{n+1} is determined such that (p_{n+1},S) lies on the straight line interpolating (p_{lo},X^2_{lo}) and (p_n,X^2_n).

This procedure, though somewhat more complicated than the simple secant procedure used, for example, in the BSPLIN subroutine SMOOTH (see Reference 1, chapter 14), converges much more rapidly.

2.6 Examples of splines calculated by BSMTH

As examples of the versatility of BSMTH in comparison with the BSPLIN subroutine SMOOTH (the IMSL subroutine ICSSCU gives splines very similar to SMOOTH), a simple set of data points has been splined using both SMOOTH and BSMTH. The input values for the data point errors, e_i and the spline X^2 was the same in all cases. These inputs completely determine the spline calculated by SMOOTH. However, the versatility of BSMTH becomes apparent when one examines the many qualitatively different curves which can be made to fit the data using BSMTH. These curves are plotted in Figures 1 to 7.

Figure 1 shows the spline calculated by SMOOTH. Notice the wiggles caused by the inability of the spline to bend rapidly near the points x = 8 and x = 15. The small crosses below the curve indicte the positions of the breakpoints or knots of the spline. For SMOOTH, these are necessarily at the data point abscissae, with the exception of the second and next to last data point.

Figure 2 demonstrates the effect of the stiffness weights in BSMTH. The knots for this spline were placed at the data points (as are the breakpoints used by SMOOTH). The

v^*_i and Y^{*2} are calculated in the subroutine SETUPR during the calculation of R_{ij}. During the iteration for p, the X^2 is evaluated using equation (2.17) in the subroutine XSQC.

2.5 The iteration for p

The spline calculated by BSMTH is required to have a X^2 equal to S, a value input by the user. This is implemented by iterating over the value of α in equation (2.7) until $|X^2 - S| < S/10$. In practice, BSMTH iterates over p, defined in equation (2.13) rather than α.

As α increases from 0 to 1, the X^2 of the spline minimizing G^* increases from some minimum value to some maximum value. However, although the linear system of equation (2.12) is theoretically invertible for any α in (0,1), R_{ij} is not invertible, and , depending on the positions of the knots with respect to the data points (see Reference 1, chapter 13), P_{ij} might not be invertible either. Hence, as α approaches 0 or 1, there will be numerical difficulties in the inversion of equation (2.12). For this reason, the allowed range of α, and therefore p is restricted. The upper and lower limits for p are denoted p_{min} and p_{max}, respectively. p_{min} is given the default value of 0.001 and p_{max} the default value of 1000. These values have been found adequate to circumvent any numerical difficulties when using BSMTH, though they may be changed if desired.

The iteration for p is divided into two steps.

a) First, values of p and their corresponding X^2's are determined. These are denoted (p_{lo}, X^2_{lo}), and (p_{hi}, X^2_{hi}). They are determined as follows.

Let p_n denote the n^{th} value of p determined and X^2_n the corresponding X^2. The initial guess for p is $p_1 = 1$. The linear system of equation (2.12) is inverted, and the X^2 of the spline is evaluated. There are two possibilities:

1) $S > X^2_1$: In this case, p_1 is too low. Set $p_{lo} = 1$ and $X^2_{lo} = X^2_1$. p_2 is set to p_{max}. *Again there are two cases:*

i) $S > X^2_2$: p_2 is still too low. However, p cannot be increased above p_{max}. Therefore, the iteration terminates.

ii) $S < X^2_2$: p_2 is too high. Set $p_{hi} = p_2$ and $X^2_{hi} = X^2_2$. p_3 is now determined such that (p_3, S) lies on the straight line interpolating (p_{lo}, X^2_{lo}) and (p_{hi}, X^2_{hi}).

2.4 Calculation of X^2

Using equations (2.1) and (2.2), the X^2 of the spline may be expressed in terms of Y^2, v_i, P_{ij} and β_i:

$$X^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} R_{ij}\beta_i\beta_j - 2\sum_{i=1}^{N} v_i\beta_i + Y^2 \tag{2.16}$$

To calculate the spline by evaluating the terms in equation (2.16) poses numerical difficulties since the X^2 itself is generally much smaller than any of the three terms, so that round-off errors become large. To circumvent the problem, the X^2 is rewritten in the following form:

$$X^2 = \sum_{n=1}^{N}\sum_{j=1}^{N} R_{nj}\gamma_n\gamma_j - 2\sum_{n=1}^{N} v^*_n\gamma_n + Y^{*2} \tag{2.17}$$

where

$$\gamma_n = \beta_n - \beta^*_n \tag{2.18}$$

$$v^*_j = \sum_{n=1}^{N_p} \frac{(y_n - y^*_n)B_{j,k}(x_n)}{e_n^2}$$

$$Y^{*2} = \sum_{n=1}^{N_p} \frac{(y_n - y^*_n)^2}{e_n^2} \tag{2.20}$$

$$y^*_j = \sum_{n=1}^{N} \beta^*_n B_{n,k}(x_j) \tag{2.21}$$

and the β_n are some arbitrarily chosen coefficients. The evaluation of the X^2 using equation (2.17) is numerically well-behaved if $\beta_n \approx \beta^*_n$. The β^*_n are chosen using the fact that B-spline coefficients closely approximate the functions they represent. That is,

$$\beta_n \approx f(t^*) \tag{2.22}$$

where

$$t^*_n = \frac{(t_n + \dots + t_{n+k-1})}{k-1} \tag{2.23}$$

(see Reference 1, pp.171). BSMTH chooses β^*_n so that (t^*_n,β^*_n) lies on the piecewise linear curve interpolating the data points, which has breakpoints at the data points.

2.3 Evaluation of P_{nj}, R_{nj}, and v_n

The matrix P_{nj} is evaluated in the subroutine SETUPP. Since $B_{n,k}(x)$ is a piecewise polynomial of order k, the integrals in the definiton of P_{nj} can be evaluated by a series of integrations by parts.

$$P_{nj} = \sum_{p=1}^{N_p} \delta_i \sum_{q=1}^{k-m} (-1)^{q-1} B_{n,k}^{(m-q)}(x_p) B_{j,k}^{(m+q-1)}(x_p) \tag{2.14}$$

If $k > 2m$, then m-q will become negative. By convention $B_{j,k}^{(q)}(x)$, for $q < 0$, is defined to be the q^{th} integral of $B_{j,k}(x)$. The subroutine BSPLVD, from the BSPLIN library, is used to evaluate the derivatives of the B-splines. If $k > 2m$, integrals of the B-splines must also be calculated. This is most easily accomplished by calculating the coefficients of the knot sequence corresponding to the integral of each B-spline (see Reference 1, page 150) and then using the BSPLIN subroutine BVALUE to evaluate it. However, to calculate the spline coefficients, k-2m knots must be appended to the knot sequence. Thus the dimension of the array containing the knots is required to be $N_k + \max(0,k-2m)$.

Owing to the left continuity of the B-splines as implemented in the subroutines BSPLVD and BVALUE, and the discontinuity of the higher derivatives of the B-splines, they cannot be evaluated right at the knots. Instead, they are evaluated at $(0.9999t_i + .0001t_{i+1})$ and $(0.0001t_i + .9999t_{i+1})$ for each knot interval.

In practice, P_{ij} in equation (2.12) is replaced by P_{ij}/Δ, where Δ is a normalizing factor used to ensure that the elements of P_{ij} are of order 1. This averts unwanted overflows and underflows. It has no effect on the minimization of G^* as the factor Δ can be absorbed into a redefinition of α. Δ is defined by

$$\Delta = \frac{(t_{N_k} - t_1)}{(N-k+1)^{2m-1}} \tag{2.15}$$

The matrix R_{ij} is evaluated in the subroutine SETUPR making use of the BSPLIN library subroutine BSPLVB to evaluate $B_{i,k}(x_n)$. This subroutine is a modification of the subroutine L2APPR in the BSPLIN library.

for given α, and iterating over α until the spline has the required X^2.

2.2 Minimization of G^* for given α

Using equations (2.1), (2.2), (2.5), and (2.6), the functional G^* may be written in the following form:

$$G^* = \sum_{n=1}^{N}\sum_{j=1}^{N} ((1-\alpha)R_{nj} + \alpha P_{nj})\beta_n\beta_j - 2(1-p)\sum_{n=1}^{N} v_n\beta_n + (1-p)Y^2 \tag{2.7}$$

where

$$R_{pj} = \sum_{n=1}^{N_p} \frac{B_{p,k}(x_n)B_{j,k}(x_n)}{e_n^2} \tag{2.8}$$

$$P_{pj} = \sum_{n=1}^{N_p} \delta_p \int_{t_{p+k-1}}^{t_{p+k}} \frac{d^m}{dx^m} B_{p,k}(x_n) \frac{d^m}{dx^m} B_{j,k}(x_n) dx \tag{2.9}$$

$$v_j = \sum_{n=1}^{N_p} \frac{y_n B_{j,k}(x_n)}{e_n^2} \tag{2.10}$$

$$Y^2 = \sum_{n=1}^{N_p} \frac{y_n^2}{e_n^2} \tag{2.11}$$

G^* is minimized with respect to the spline coefficients, β_n, when

$$\sum_{j=1}^{N} (R_{nj} + pP_{nj})\beta_j = v_n \tag{2.12}$$

where

$$p = \frac{\alpha}{1-\alpha} \tag{2.13}$$

Since both P_{ij} and R_{ij} are symmetric, banded, positive definite matrices, the subroutines BCHFAC and BCHSLV in the BSPLIN library are appropriate for the solution of the linear system in equation (2.12).

X^2 measures the degree to which the curve approximates the data and is minimized when the curve interpolates the data.

The second functional is a measure of the smoothness of the spline function. Its definition relies on the observation that, for a smooth function, the average values of its high order derivatives will be considerably lower than those of a 'wiggly' function. Hence, one uses the functional

$$F = \int_{x_1}^{x_N} \left[\frac{d^2 f(x)}{dx^2} \right]^2 dx \tag{2.3}$$

as a measure of the smoothness of the spline.

The spline desired is that which has a given X^2 while minimizing F. In practice, this is found by finding the spline which minimizes the functional

$$G = \alpha X^2 + (1-\alpha)F \tag{2.4}$$

for given α. Since G is quadratic in the spline coefficients β_n, this amounts to the solution of a linear system. An iteration is then done to find the value of α for which X^2 has the required value. As implemented by Reinsch and de Boor, the splines are necessarily cubic, and the knots are constrained to be the data point abscissae, x_n, n=1,N.

A shortcoming of the above algorithm is that the second derivative of the spline is minimized even in places where one might expect it to be high: that is, where the data shows a pronounced bend. This problem has been avoided in BSMTH by generalizing the functional F to

$$F^* = \sum_{n=1}^{N-k+1} \delta_n \int_{t_n}^{t_{n+1}} \left[\frac{d^m f(x)}{dx^m} \right]^2 dx \tag{2.5}$$

The basis for the linear space of spline functions has been chosen to be the B-spline basis (see Reference 1, chapter 9). The n^{th} B-spline of order k is denoted $B_{n,k}(x)$ and the knots of the spline are denoted t_n, n = $1,N_k$.

The coefficients δ_n can be used to alter the 'stiffness' of the spline between the pair of knots,(t_{n-k+1},t_{n-k+2}). In regions where the spline curve is required to be very flat, the δ_n will be large. In regions where the spline is expected to have high curvature, the δ_n will be small.

The required spline is found by minimizing the functional

$$G^* = \alpha X^2 + (1-\alpha)F^* \tag{2.6}$$

decreased. Two subroutines, WTIBEG and WTINEW (see Section 4) are provided which calculate appropriate default values for the stiffness weights from the data points or from previous spline fits to the data, respectively.

5) SMOOTH and ICSSCU implement smoothing by minimizing the second derivative of the the spline. BSMTH allows one to choose the derivative which is to be minimized, again allowing more control over the character of the spline.

BSMTH does have the drawback that it is somewhat slower than the other subroutines, though usually at most by a factor of two. However, much of the extra time can often be made up by reducing the number of knots of the spline with no deterioration in the quality of the fit (the execution time is roughly proportional to the number of knots). Moreover, when splining in two dimensions, the time savings involved in having the data points independent of the knots far outweigh the slight inefficiency of BSMTH.

In the following sections the algorithms for each of the subroutines is discussed in detail. User's guides including sample runs of the subroutines are given in Appendix A. The computer code for each subroutine is given in Appendix B.

2 THE BSMTH ALGORITHM

2.1 Implementation of Smoothing

The technique used in ICSSCU and SMOOTH, for constructing a smooth spline curve through a given set of data is an extension of an algorithm first proposed by Whittaker[4] and later considered by Schoenberg[5], Reinsch[3] and de Boor[1]. The idea is to define two functionals dependent quadratically on the spline coefficients for which one is solving. One functional is the χ^2 of the spline curve,

$$\chi^2 = \sum_{n=1}^{N_p} \left[\frac{y_n - f(x_n)}{e_n}\right]^2 \tag{2.1}$$

where

(x_n, y_n), $n=1,N_p$; are the data points to be interpolated,

e_n is the error associated with the n-th data point, and

$$f(x) = \sum_{n=1}^{N} \beta_n f_n(x) \tag{2.2}$$

The functions $f_n(x)$ are the basis functions for the linear space of spline functions. The

= 5, If $0.9*X^2$ of the spline of the data point abscissae is greater than the value predicted by PRERR.

= 6, If $1.1*X^2$ of the spline of the data point abscissae is less than the value predicted by PRERR.

= 7, 8, 9 , As for IER = 4, 5, and 6 respectively, but for the spline of the data point ordinates.

BCOEFX : An array of length N containing the B-spline coefficients of the spline of the abscissae.

BCOEFY : An array of length N containing the B-spline coefficients of the spline of the ordinates.

ARCL : An array of length N containing the estimated arc-length at each data point.

Via COMMON / CHISQ /

XSQ = X^2 of the spline

WORK SPACE

WTI : An array of length N which is used to contain the stiffness weights as calculated by WTIBEG.

G : An array of length NPT*IMAX used as work space in the function PRERR.

R : An array of length 3*K*N

IWK = max(NKT1,K**2)

WK : An array of length 4*IWK

The following data has been splined using BSMCRV. The resulting spline has been plotted in Figure 11.

NPT = 22, N = 20, K = 4, NKT1 = 24, IWK = 24, IER = 0

J	X(J)	Y(J)	E(J)
1	-0.24	-0.09	0.3
2	0.06	0.03	0.3
3	0.20	0.18	0.3
4	0.31	0.39	0.3
5	0.43	0.47	0.3
6	0.45	0.63	0.3
7	0.39	0.77	0.3
8	0.38	0.86	0.3
9	0.29	0.94	0.3
10	0.11	0.97	0.3
11	0.06	1.03	0.3
12	-0.08	1.02	0.3
13	-0.17	1.00	0.3
14	-0.23	0.97	0.3
15	-0.31	0.91	0.3
16	-0.29	0.80	0.3
17	-0.33	0.70	0.3
18	-0.30	0.59	0.3
19	-0.15	0.45	0.3
20	0.06	0.33	0.3
21	0.26	0.11	0.3
22	0.66	0.03	0.3

The spline coefficients and the fractional arc length values returned by BSMCRV are

J	BCOEFX(J)	BCOEFY(J)	ARCL(J)
1	-0.35219	-0.20888	0.00000E+00
2	-0.19660	-0.10975	0.90250E-01
3	-0.39770E-01	-0.19646E-01	0.14756
4	0.10954	0.11428	0.21378
5	0.24699	0.27045	0.25406
6	0.37681	0.43389	0.29910
7	0.46363	0.61444	0.34165
8	0.40611	0.79660	0.36694
9	0.24539	0.97128	0.40057
10	0.56733E-01	1.0224	0.45154
11	-0.13699	1.0425	0.47336
12	-0.32258	0.88117	0.51256
13	-0.34446	0.69866	0.53832
14	-0.21200	0.53258	0.55705
15	-0.53392E-01	0.39341	0.58498
16	0.11140	0.26571	0.61621
17	0.27779	0.14252	0.64630
18	0.44669	0.43413E-01	0.67814
19	0.62444	0.14056E-01	0.73545
20	0.80006	-0.31815E-01	0.80301
21			0.88606
22			1.0000

The values returned via COMMON / CHISQ / are

XSQX = 0.76300E-01, XSQY = 0.11647, SX = 0.71920E-01, SY = 0.11766

A.2 BSMTH : User's Guide

SUBROUTINE BSMTH(S,JDER,NPT,X,Y,E,N,K,NKT1,T,WTI,BCOEF,R,IWK,WK,IER)

PURPOSE: BSMTH calculates the spline of order K, with knots T(I),I=1,NKT which has chi-square of S with respect to the data points X(I),Y(I),I=1,NPT, and which has as small a JDERth derivative as possible.

LANGUAGE: FORTRAN

USAGE: EXECUTE mainpgm,BSPLIN:HLLYSP/LIB,BSPLIN:BSPLIN/LIB

CALLS subroutines SETUPQ, SETUPR, XSQC, SMODAV and INTERV, BCHFAC, BCHSLV from the BSPLIN library

INPUT

S : The chi-square of the spline with respect to the data will be within 10% of S, if possible. As S is increased the spline becomes smoother but farther from the data points. Function PRERR can be used to give a value for S if a reasonable value is not known.

JDER : The integral of the square of the JDERth derivative of the spline is minimized (subject to the constraint that XSQ = S). If smooth curves are desired a value of JDER = 2 is appropriate. JDER should be non-negative and less than K.

NPT : The number of data points.

X : An array of length NPT containing the data point abscissae in ascending order.

Y : An array of length NPT containing the data point ordinates.

E : The errors of the data points. The smaller the error the closer the spline will come to that point.

N : The number of B-splines used to represent the spline.

K : The order of the spline.

NKT1 = N + K + max(0,K-2*JDER)

T : An array of length NKT1 the first N+K elements of which contain the knot sequence (in ascending order). The remaining array elements are used in subroutine SETUPP.

WTI : An array of length N of which only the first N-K+1 elements are used (rather than passing in an otherwise superfluous argument). WTI(I) is a weight for the integral of the square of the JDERth derivative of the spline between T(I+K-1) and T(I+K). The larger WTI(I) is the smoother the integral will be over this region. These weights are relative: i.e. changing all the WTI by a constant factor will not affect the resulting spline.

IER = 0, If JDER,T,WTI and the first N*K elements of R are as on the previous call to BSMTH (*this means that the matrix P need not be recalculated*)

= 1, if P is to be recalculated

Via COMMON / PLIMS /

PMIN = Minimum allowed value of p (See (Section 2.5)). Default is 1.0E-03

PMAX = Maximum allowed value of p. Default is 1.0E+03.

OUTPUT

IER = 0, Calculation has been successful

= 1, If JDER > K - 1

= 2, If NKT1 < N + K + max(0,K-2*JDER)

= 3, If IWK < max(NKT1,K**2)

= 4, If more than 30 iterations are required to find the correct value for P. Indicates numerical difficulties in the solution of the linear system

= 5, If the X^2 of the spline > 1.1*S

= 6, If the X^2 of the spline < .9*S

BCOEF : An array of length N containing the B-spline coefficients of the spline.

Via COMMON / CHISQ /

XSQ = X^2 of the spline

WORK SPACE

R : An array of length 3*K*N

IWK = max(NKT1,K**2)

WK : An array of length 4*IWK

The following input data has been splined using BSMTH. A plot of the spline is shown in Figure 2.

S = 10.0 , JDER = 2 , NPT = 10 , K = 4 , N = 10 , NKT = 14

J	X(J)	Y(J)	E(J)	T(J)	WTI(J)
1	7.0	0.0	0.005	7.0	0.51183E-02
2	8.0	13.5	0.5	7.0	0.55789
3	9.0	15.5	0.5	7.0	1.1778
4	10.0	14.5	0.5	7.0	1.1778
5	11.0	15.5	0.5	9.0	2.6500
6	12.0	15.0	0.5	10.0	0.88333
7	13.0	14.5	0.5	11.0	0.15407E-01
8	14.0	15.0	0.5	12.0	
9	15.0	13.5	0.5	13.0	
10	16.0	0.0	0.005	14.0	
11				16.1	
12				16.1	
13				16.1	
14				16.1	

The spline coefficients obtained were

J	BCOEF(J)
1	0.3195009E-04
2	14.03531
3	14.93398
4	15.00996
5	15.11922
6	15.09243
7	15.00040
8	14.90065
9	13.10809
10	-2.075629

and the X^2 of the spline was

XSQ = 9.9912

A.3 NEWWTI : User's Guide

SUBROUTINE NEWWTI(NOLD,BCOEF,NKTOLD,TOLD,NKTNEW,TNEW,NWTI,WTI,JDER)

PURPOSE: NEWWTI uses a previously calculated spline fit to predict values for the stiffness weights δ_i for use in BSMTH.

LANGUAGE: FORTRAN

USAGE: EXECUTE mainpgm,BSPLIN:HLLYSP/LIB, BSPLIN:BSPLIN/LIB

CALLS subroutines SMODAV and BVALUE from the BSPLIN library.

INPUT

- NOLD : Number of B-splines for old spline fit.
- BCOEF : Array of length N containing the B-spline coefficients for the old spline fit.
- NKTOLD : Number of knots for the old spline fit.
- TOLD : Array of length NKT containing the knots for the old spline fit.
- NKTNEW: Number of knots for the new spline fit.
- TNEW : Array of length NKT containing the knots for the new spline fit.
- NWTI : Number of stiffness weights for the new *spline fit*.
- JDER : The order of derivative minimized by BSMTH.

OUTPUT

- WTI : Array of length NWTI containing the stiffness weights, δ_i.

The following input data has been used to generate stiffness weights by NEWWTI. This data is the output data from the example in Section A.2.

NOLD = 4 , NKTOLD = 14 , NKTNEW = 14 , NWTI = 7 , JDER = 2

J	TOLD(J)	TNEW(J)
1	7.0	7.0
2	7.0	7.0
3	7.0	7.0
4	7.0	7.0
5	9.0	9.0
6	10.0	10.0
7	11.0	11.0
8	12.0	12.0
9	13.0	13.0
10	14.0	14.0
11	16.1	16.1
12	16.1	16.1
13	16.1	16.1
14	16.1	16.1

The stiffness weights obtained were

J	WTI(J)
1	0.10000E-02
2	0.56836E-01
3	1.1596
4	0.51772
5	4.8525
6	0.14507E-01
7	0.10000E-02

A.4 PRERR : User's Guide

FUNCTION PRERR(NPT,X,Y,E,IMAX,G,WK,IFLAG)

PURPOSE: This function calculates the mean error in the data points. The smoothing parameter used by BSMTH may then be determined by: S = NPT*PRERR**2.

LANGUAGE: FORTRAN

USAGE: EXECUTE mainpgm,BSPLIN:HLLYSP/LIB

CALLS subroutines PARDIF

INPUT

NPT	:	The number of data points.
X	:	An array of length NPT containing the data point abscissae in ascending order.
Y	:	An array of length NPT *containing the data point ordinates.*
E	:	The errors of the data points.
IMAX	:	The maximum number of partial differences taken is 2*IMAX. The suggested value for IMAX is 5.
IFLAG	=	0, If the calculation is to be done from scratch.
	=	1, If X, E and G have not been changed since the previous call.

OUTPUT

PRERR returns the mean error in the data points.

WORK SPACE

WK	:	An array of length NPT
G	:	An array of length N*IMAX

The mean error in the following data has been predicted by PRERR. Splines of this data are shown in Figure 8, 9, and 10 and are discussed in Section 3.

N = 40, IMAX = 5

J	X(J)	Y(J)	J	X(J)	Y(J)
1	0.00	1507.89	21	22.70	1482.57
2	3.63	1507.85	22	23.00	1481.83
3	7.26	1507.81	23	23.50	1480.34
4	10.90	1507.77	24	24.20	1478.83
5	12.20	1507.17	25	26.10	1476.17
6	13.70	1505.02	26	27.00	1475.02
7	14.10	1502.49	27	28.10	1472.68
8	14.50	1501.53	28	29.00	1470.30
9	14.80	1499.50	29	29.90	1468.70
10	15.20	1498.27	30	30.60	1467.92
11	15.30	1496.94	31	44.00	1463.25
12	15.40	1495.93	32	52.70	1464.54
13	15.70	1494.92	33	58.40	1466.09
14	16.60	1492.87	34	65.20	1469.74
15	16.80	1491.84	35	74.50	1475.43
16	17.30	1490.09	36	80.30	1478.08
17	18.40	1488.33	37	94.50	1482.65
18	20.90	1486.22	38	110.00	1487.18
19	21.80	1484.77	39	119.10	1489.40
20	22.50	1483.31	40	158.10	1491.40

PRERR returned the value 0.20361

A.5 WTIBEG : User's Guide

SUBROUTINE WTIBEG(NPT,X,Y,NKT,T,NWTI,WTI)

PURPOSE: WTIBEG uses the data points to calculate values for the stiffness weights δ_i for use in BSMTH.

LANGUAGE: FORTRAN

USAGE: EXECUTE mainpgm,BSPLIN:HLLYSP/LIB, BSPLIN:BSPLIN/LIB

CALLS subroutines SMODAV and BVALUE from the BSPLIN library.

INPUT

- NPT : The number of data points.
- X : An array of length NPT containing the data point abscissae in ascending order.
- Y : An array of length NPT containing the data point ordinates.
- NKT : Number of knots for the spline.
- T : Array of length NKT containing the knots for the spline.
- NWTI : Number of stiffness weights for the spline fit. NWTI = NKT-2*K+1 where K is the order of the spline.

OUTPUT

- WTI : Array of length NWTI containing the stiffness weights, δ_i.

The following input data has been used to generate stiffness weights in WTIBEG. The spline obtained from this data is discussed in Section 2.6 and is plotted in Figure 2.

NPT = 10 , NKT = 14 , NWTI = 7

J	X(J)	Y(J)	T(J)
1	7.0	0.0	7.0
2	8.0	13.5	7.0
3	9.0	15.5	7.0
4	10.0	14.5	7.0
5	11.0	15.5	9.0
6	12.0	15.0	10.0
7	13.0	14.5	11.0
8	14.0	15.0	12.0
9	15.0	13.5	13.0
10	16.0	0.0	14.0
11			16.1
12			16.1
13			16.1
14			16.1

The stiffness weights obtained were

J	WTI(J)
1	0.51183E-02
2	0.55789
3	1.1778
4	1.1778
5	2.6500
6	0.88333
7	0.15407E-01

Appendix B

SUBROUTINE LISTINGS

```
C     **********************************************************
C     *                                                        *
C     *   THESE COMPUTER SUBROUTINES ARE THE PROPERTY OF THE   *
C     *   CANADIAN DEPARTMENT OF NATIONAL DEFENCE ....         *
C     *                                                        *
C     *   THEY SHALL BE USED ONLY FOR PURPOSES AUTHORISED      *
C     *   BY THE DEPARTMENT ..........................         *
C     *                                                        *
C     *   THEY SHALL NOT BE DISCLOSED TO A THIRD PARTY         *
C     *   WITHOUT THE WRITTEN PERMISSION OF THE                *
C     *   DEPARTMENT ..............................            *
C     *                                                        *
C     **********************************************************
```

B.1 BSMCRV

```
      SUBROUTINE BSMCRV(NPT,X,Y,E,N,K,NKT1,T,WTI,BCOEFX,BCOEFY,
     *   R,IWK,WK,ARCL,G,IER)
C----------------------------------------------------------------------C
C                                                                      C
C  Given data points (X(I),Y(I)), I=1,NPT BSMCRV finds a smooth        C
C   curve approximating them by splining the abscissae and ordinates   C
C   separately with respect to the fractional arc-length along the     C
C   spline. An approximation for the arc length at each point is       C
C   obtained from the distances between the points.  BSMTH is used to C
C   spline the absissae and the  ordinates. PRERR is used to           C
C   determine a smoothing factor for the splines and WTIBEG is used    C
C   to determine stiffness weights.                                    C
C                                                                      C
C AUTHOR: David Hally , May 1981                                       C
C                                                                      C
C USAGE:                                                               C
C       EXECUTE mainpgm,BSPLIN:HLLYSP/LIB,BSPLIN:BSPLIN/LIB            C
C                                                                      C
C CALLS PRERR,BSMTH,WTIBEG                                             C
C                                                                      C
C INPUT:                                                               C
C                                                                      C
C   VIA SUBROUTINE ARGUMENTS:                                          C
C                                                                      C
C   NPT     : The no. of data points                                   C
C   X       : An array of length NPT containing the data point         C
C   Y       : An array of length NPT containing the data point         C
C                ordinates.                                            C
C   E       : The  errors of the data points. The smaller              C
C                the error the closer the spline will come             C
C                to that point.                                        C
```

```
C    N          : The no. of B-splines.                                  C
C    K          : The order of the spline.                               C
C    NKT1       = N+K+max(0,K-2*JDER)                                    C
C    T          : An array of length NKT1 the first N+K elements of      C
C                   which contain the knot sequence. The variable used   C
C                   to parametrize the curve is the arc length divided by C
C                   the total length of the curve. Thus the knots must   C
C                   span the interval [0,1]. Default gives a uniform     C
C                   distribution of knots over this interval.            C
C                                                                        C
C   IER         = 0  If defaults are desired                             C
C               = 1  If defaults are not desired                         C
C                                                                        C
C    COMMON /NODFLT/                                                     C
C                                                                        C
C   IMAX        : 2*IMAX is the max. no. of divided differences allowed  C
C                   to find the error (used in function PRERR). Default  C
C                   value is 5                                           C
C   SMFACT : See comments below                                          C
C                                                                        C
C   COMMON /INTEXP/ :                                                    C
C                                                                        C
C    JDER       : The integral of the square of the JDER-th derivative   C
C                   of the spline is minimized ( subject to the CON-     C
C                   straint that XSQ=S). Default value is 2 .            C
C                   JDER must not exceed K-1                             C
C   DEFAULTS:                                                            C
C                                                                        C
C    If IER = 0 on input then:                                           C
C     JDER    = 2                                                        C
C     SMFACT  = 1.0                                                      C
C     IMAX    = 5                                                        C
C     T(I)    = (I-K)/(N-K+1), I=1,NKT i.e. knots are uniformly          C
C                 distributed in (0,1)                                   C
C                                                                        C
C   OUTPUT:                                                              C
C                                                                        C
C    IER        = 0 , Iteration converged                                C
C               = 1 , If JDER > K-1                                      C
C               = 2 , If NKT1 < N+K+MAX(0,K-2*JDER)                      C
C               = 3 , If IWK < max(NKT1,K**2)                            C
C               = 4 , If iteration for P1 in BSMTH did not converge      C
C                       during the spline of the X-values                C
C               = 5 , If the chi-square of the spline of the X-values    C
C                       returned by BSMTH > 1.1*S    (i.e. PMAX in BSMTH C
C                       is too small)                                    C
C               = 6 , If the chi-square of the spline of the X-values    C
C                       returned by BSMTH < .9*S     (i.e. PMIN in BSMTH C
C                       is too large)                                    C
C               = 7,8,9 As for IER=4,5,6,respectively, but for the spline C
C                       of the Y-values                                  C
C    BCOEFX     = Array of length N containing the B-spline coefs. for   C
C                     the X-values of the curve                          C
```

B.6 SETUPP

```
      SUBROUTINE SETUPP(NPT,E,JDER,T,NKT,N,K,WTI,P,A,DB,DB1,WK,A1)
C-----------------------------------------------------------------------C
C                                                                       C
C SETUPP calculates the matrix P ( see Ref. Manual )                    C
C                                                                       C
C AUTHOR: David Hally , May. 1981                                       C
C                                                                       C
C CALLED by BSMTH                                                       C
C                                                                       C
C CALLS SMODAV                                                          C
C        from BSPLIN library BVALUE,BSPLVD                              C
C                                                                       C
C-----------------------------------------------------------------------C
      REAL T(NKT),P(K,N),A(K,K),DB(K,K),DB1(K,K),WTI(N),WK(NKT),
     *     E(NPT),A1(K,K),T1,T2,H,HI
      INTEGER NKT,JDER,N,K,MMAX,I,J,L,LJ1,IKJ,M,I1,KM

C A normalizing factor H is calculated. Normalization by H ensures that
C  most of the elements of P are of order 1.

      H=((T(N+K)-T(1))/FLOAT(N-K+1))**(2*JDER-1)

C P is initialized and extra points are added to the knot sequence
C  to allow the calculation of higher order B-splines if necessary
C  in the integration by parts.

      H=H/(SMODAV(NPT,E)**2*SMODAV(N-K+1,WTI))

      DO 10 I=1,K
         DO 10 J=1,N
10          P(I,J)=0.
      IF(N+K.EQ.NKT)GO TO 30
      DO 20 J=NKT,N+K+1,-1
20       T(J)=T(N+K)*1.0001
30    MMAX=MIN0(JDER,K-JDER)

C An iteration over the intervals between knots is begun.

      DO 140 I=K,N
         IF(T(I+1).EQ.T(I))GO TO 140
         I1=I-K+1

C The derivatives of the B-splines needed in the integration by parts
C  are calculated using BSPLVD. Due to the left continuity of BSPLVD
C  the derivatives are evaluated close to but not right at the knots.

         T1=.9999*T(I)+.0001*T(I+1)
         T2=.0001*T(I)+.9999*T(I+1)
         CALL BSPLVD(T,K,T1,I,A,DB,K)
         CALL BSPLVD(T,K,T2,I,A,DB1,K)

C The integrals of the B-splines needed in the integration by parts
```

```
        CALL PARDIF(N,X,WK,J,J+1,N-J)
        J=J+1
        J2=J2+2
        IF(NSGNCH.GT.D/2.)GO TO 130
        IF(J.GE.IMAX-2)GO TO 120
        GO TO 70

120     SDEV=(D/2.-NSGNCH)/SQRT(D)

C  The error is determined from the divided difference by taking the
C   root mean square of the divided difference values weighted by
C   the expected value for a unit error (given by G(I,J+1)).
C   anomalously high values are discarded and the resulting error
C   is corrected by multiplying prerr by 1.14

130     DEV=0.0
        DO 140 I=J+1,N-J
140        DEV=(WK(I)/G(I,J+1))**2+DEV
        IF(DEV.EQ.0.0)RETURN
        NM2J=N-J2
        PRERR=SQRT(DEV/FLOAT(NM2J))*2.0
        DO 150 I=J+1,N-J
           IF(ABS(WK(I)/G(I,J+1)).LT.PRERR)GO TO 150
           DEV=DEV-(WK(I)/G(I,J+1))**2
           NM2J=NM2J-1
150     CONTINUE
        PRERR=SQRT(DEV/FLOAT(NM2J))*1.14

        RETURN
        END
```

```
      REAL X(N),Y(N),E(N),WK(N),G(N,IMAX),DEV,D,PRERR
      INTEGER NSGNCH,NM2J,N,K,IMAX,J,J2,KMIN,KMAX,IFLAG,I

      COMMON /CERR/ SDEV

      IMAX=MIN0(N/2,IMAX)
      SDEV=0.0
      PRERR=0.0
      IF(IFLAG.EQ.1)GO TO 50

C  G is calculated.

      DO 10 I=1,N
         G(I,1)=E(I)
         DO 10 J=2,IMAX
10          G(I,J)=0.0
      DO 30 I=1,N
         DO 20 J=1,N
20          WK(J)=0.0
         WK(I)=E(I)
         DO 30 J=1,IMAX-1
            KMIN=MAX0(I-J-1,J)
            KMAX=MIN0(I+J+1,N-J+1)
            CALL PARDIF(N,X,WK,J-1,KMIN,KMAX)
            IF(I.GT.J)KMIN=KMIN+1
            IF(I+J.LT.N+1)KMAX=KMAX-1
            DO 30 K=KMIN,KMAX
30             G(K,J+1)=WK(K)**2+G(K,J+1)
      DO 40 J=1,IMAX-1
         DO 40 I=J+1,N-J
40          G(I,J+1)=SQRT(G(I,J+1))

C Divided differences are taken until the no. of sign changes is
C  greater than that expected for random data. If IMAX-2 iterations
C  occur first SDEV is set to the number of standard deviations
C  that NSGNCH is below its expected value.

50    DO 60 I=1,N
60       WK(I)=Y(I)
      J=0
      J2=0

C The no. of sign changes in the divided differences is determined.

70    NSGNCH=0
      I=J+1
80    DO 90 K=I+1,N-J
         IF(WK(I)*WK(K))100,90,110
90    CONTINUE
100   NSGNCH=NSGNCH+1
110   I=K
      IF(I.LT.N-J)GO TO 80

      D=N-J2-1
```

B.5 PRERR

```
      FUNCTION PRERR(N,X,Y,E,IMAX,G,WK,IFLAG)
C-----------------------------------------------------------------C
C                                                                 C
C  This subroutine calculates the mean error in the data points (X,Y) C
C   by taking divided differences until the no. of sign changes in  C
C   the I-th divided difference is that expected from random data.  C
C   The error is then determined by assuming that the contribution  C
C   from the smooth curve underlying the data is negligible.        C
C                                                                 C
C AUTHOR: David Hally , Jan. 1981                                 C
C                                                                 C
C USAGE:                                                          C
C       EXECUTE mainpgm,BSPLIN:HLLYSP/LIB                         C
C                                                                 C
C CALLS PARDIF                                                    C
C                                                                 C
C INPUT :                                                         C
C                                                                 C
C   N        = No. of data points                                 C
C   X        : An array of length N containing the data point     C
C                abscissae in ascending order.                    C
C   Y        : An array of length N containing the data point     C
C                ordinates.                                       C
C   E        : An array of length N containing the relative errors of C
C                the data points. The absolute errors are obtained by C
C                multiplying the returned value of PRERR by the   C
C                relative errors.                                 C
C   IFLAG    = 0 , If calculation is to be done from scratch      C
C            = 1 , If IMAX,X,E, and G have the same value as in the C
C                 previous call                                   C
C   G        = Array of dimensions N,IMAX. G(I,J) is the expectation C
C                 value of the J-th divided difference given an error C
C                 of E(I) in the I-th data point. If IER=0 G Is   C
C                 calculated ; otherwise it is assumed known.     C
C   IMAX     : 2*IMAX is the max. no. of divided differences allowed C
C             IMAX = 5 is suggested                               C
C                                                                 C
C OUTPUT :                                                        C
C                                                                 C
C   PRERR = The calculated mean error in the data                 C
C                                                                 C
C  VIA COMMON / CERR /                                            C
C                                                                 C
C   SDEV : The no. of sign changes in the divided difference used C
C             to calculate PRERR is greater than that expected for C
C             random data less SDEV standard deviations           C
C                                                                 C
C   WORK SPACE :                                                  C
C                                                                 C
C    WK(I) OF DIMENSION N                                         C
C                                                                 C
C-----------------------------------------------------------------C
```

B.4 PARDIF

```
      SUBROUTINE PARDIF(N,X,F,J,IMIN,IMAX)
C-----------------------------------------------------------------C
C                                                                 C
C PARDIF calculates the divided difference of the data points     C
C  (X(I),F(I)),I=IMIN,IMAX. To avoid over- or underflows the X    C
C  intervals are normalized by the factor H=(X(N)-X(1))/N . This is C
C  of no consequence in PRERR since only ratios of partial diff-  C
C  erences are of significance.                                   C
C                                                                 C
C AUTHOR: David Hally , May. 1981                                 C
C                                                                 C
C CALLED by PRERR                                                 C
C                                                                 C
C-----------------------------------------------------------------C
      REAL X(N),F(N),H
      INTEGER J,IMIN,IMAX,I,N,IT

      H=(X(N)-X(1))/FLOAT(N)
      DO 10 IT=1,2
         DO 10 I=IMIN,IMAX-IT
10          F(I)=H*(F(I+1)-F(I))/(X(I+IT)-X(I-J))
      DO 20 I=IMAX-2,IMIN,-1
20       F(I+1)=F(I)
      F(IMIN)=0.0
      F(IMAX)=0.0
      RETURN
      END
```

```
          B2=BVALUE(TOLD,BCOEF,NOLD,KOLD,T2,JDER)
          WTI(IW)=(B1*(B1+B2)+B2**2)*(TNEW(I+KNEW)-TNEW(I+KNEW-1))/3.
10      CONTINUE

C The modal average of WTI is determined and WTI(I) is set to
C  WTIAV/WTI(I)

        WTIAV=SMODAV(NWTI,WTI)
        IF(WTIAV.EQ.0.0)GO TO 30
        WMIN=WTIAV*1.0E-03
        WMAX=WTIAV*1.0E+03
        DO 20 IW=1,NWTI
          DUMMY=WTI(IW)
          IF((WTI(IW).GT.WMIN).AND.(WTI(IW).LT.WMAX))WTI(IW)=
     *                  WTIAV/WTI(IW)
            IF(DUMMY.GT.WMAX)WTI(IW)=1.0E-03
20        IF(DUMMY.LE.WMIN)WTI(IW)=1.0E+03
        RETURN

30      DO 40 IW=1,NWTI
40        WTI(IW)=1.0
        RETURN
        END
```

B.3 NEWWTI

```
      SUBROUTINE NEWWTI(NOLD,BCOEF,NKTOLD,TOLD,NKTNEW,TNEW,NWTI,WTI,
     *                           JDER)
C-----------------------------------------------------------------C
C                                                                 C
C NEWWTI uses the previous spline fit to predict values for the  C
C  integral weights WTI for use in BSMTH.                         C
C                                                                 C
C AUTHOR: David Hally , Aug. 1981                                 C
C                                                                 C
C USAGE:                                                          C
C       EXECUTE mainpgm,BSPLIN:HLLYSP/LIB                         C
C                                                                 C
C CALLS SMODAV                                                    C
C       from BSPLIN library : BVALUE                              C
C                                                                 C
C INPUT:                                                          C
C                                                                 C
C  NOLD    = No. of B-splines  for old spline fit                 C
C  BCOEF   = Array of length N containing the B-spline coefficients C
C             for the old spline fit                              C
C  NKTOLD  = No. of knots for the old spline fit                  C
C  TOLD    = Array of length NKT containing the knots for the old C
C             spline fit                                          C
C  NKTNEW  = No. of knots for the new spline fit                  C
C  TNEW    = Array of length NKT containing the knots for the new C
C             spline fit                                          C
C  NWTI    = No. of integral weights for the new spline fit       C
C  JDER    = The order of derivative minimized by BSMTH           C
C                                                                 C
C OUTPUT:                                                         C
C                                                                 C
C  WTI     = Array of length NWTI containing the integral weights C
C                                                                 C
C-----------------------------------------------------------------C
      REAL BCOEF(NOLD),TOLD(NKTOLD),TNEW(NKTNEW),WTI(NWTI),
     *     B1,B2,T1,T2,WMIN,WMAX,WTIAV,DUMMY
      INTEGER NOLD,NKTOLD,NKTNEW,KOLD,KNEW,NWTI,JDER,I,IW

      KOLD=NKTOLD-NOLD
      KNEW=(NKTNEW-NWTI+1)/2
      IW=0

C On each knot interval the integral of the square of the JDER-th
C  derivative of the given spline is approximated

      DO 10 I=1,NWTI
         IF(TNEW(I+KNEW-1).EQ.TNEW(I+KNEW))GO TO 10
         IW=IW+1
         T1=.9999*TNEW(I+KNEW-1)+.0001*TNEW(I+KNEW)
         T2=.0001*TNEW(I+KNEW-1)+.9999*TNEW(I+KNEW)
         B1=BVALUE(TOLD,BCOEF,NOLD,KOLD,T1,JDER)
```

```
           PHI=P1
           P1=.1*PLO+.9*PHI
           GO TO 220

C Similarly, if P1 is very close to PLO, it is possible that XSQ<XSQLO
C  In this case PLO is set to P1, XSQLO to XSQ and P1 to .9*P1+.1*PHI

170        IF(XSQ.GT.XSQLO)GO TO 180
           XSQLO=XSQ
           PLO=P1
           P1=.9*PLO+.1*PHI
           GO TO 220

180        IF((S-XSQLO).LT.(XSQHI-S))GO TO 190
           P2=(P1-PHI)*(S-XSQHI)/(XSQ-XSQHI)+PHI
           GO TO 200
190        P2=(P1-PLO)*(S-XSQLO)/(XSQ-XSQLO)+PLO

200        IF(P2.LT.P1)GO TO 210
           PLO=P1
           XSQLO=XSQ
           P1=P2
           IF(P1.GT.PHI)P1=(PLO+PHI)/2.
           GO TO 220

210        PHI=P1
           XSQHI=XSQ
           P1=P2
           IF(P1.LT.PLO)P1=(PLO+PHI)/2.
220     CONTINUE
        IER=4

C BCOEF is returned to its correct value (see comment before call to
C  XSQC).

230     DO 240 I=1,N
240        BCOEF(I)=BCOEF(I)+WK(I,1)
        RETURN
        END
```

```
90              BCOEF(I)=BCOEF(I)-WK(I,1)
           XSQ=XSQC(N,K,BCOEF,R(1,1,2),WK(1,3),WK(1,4),YSQ)

C If XSQ is within .1*S of S the iteration terminates.
C  The first value of XSQ calculated is for : P1=1. , then P1=PMIN or
C  P1=PMAX depending on whether S is less or greater than XSQ. The third
C  value of P1 is predicted by linear interpolation of the two known
C  points. The known P's and their corresponding XSQ's are then:
C  (PLO,XSQLO),(PHI,XSQHI), and (P1,XSQ) respectively. Subsequently
C  improved values of P1 are predicted by a linear interpolation
C  of (P1,XSQ) and either (PLO,XSQLO) or (PHI,XSQHI) depending on
C  whether S is closer to XSQLO or XSQHI.
C If XSQHI < S or XSQLO > S  initially the iteration terminates.

100        IF(ABS(S-XSQ).LT.S*.1)GO TO 230
           GO TO(110,130),IT
           GO TO 160

110        IF(S.LT.XSQ)GO TO 120
           XSQLO=XSQ
           PLO=P1
           P1=PMAX
           GO TO 220

120        XSQHI=XSQ
           PHI=P1
           P1=PMIN
           GO TO 220

130        IF(P1.EQ.PMIN)GO TO 140
           IF(S.LE.XSQ)GO TO 135
           IER=5
           GO TO 230

135        XSQHI=XSQ
           PHI=P1
           GO TO 150

140        IF(S.GE.XSQ)GO TO 145
           IER=6
           GO TO 230

145        XSQLO=XSQ
           PLO=P1
150        P1=(PHI-PLO)*(S-XSQLO)/(XSQHI-XSQLO)+PLO
           GO TO 220

C  It is possible that due to numerical inaccuracy in the evaluation
C   of XSQ, that XSQ>XSQHI. This would normally only occur if P1 is
C   very close to PHI. Hence PHI is set to P1, XSQHI to XSQ and
C   P1 to .1*PLO+.9*PHI

160        IF(XSQ.LT.XSQHI)GO TO 170
           XSQHI=XSQ
```

```
      IER=1
      RETURN
10    IF(NKT1.GE.NKT+MAX0(0,K-2*JDER))GO TO 20
      IER=2
      RETURN
20    IF((IWK.GE.NKT1).AND.(IWK.GE.K**2))GO TO 30
      IER=3
      RETURN

C The matrices P and R are calculated in
C  SETUPP and SETUPR respectively.

30    IF(IER.EQ.0)GO TO 40
      CALL SETUPP(NPT,E,JDER,T,NKT1,N,K,WTI,R(1,1,3),WK,WK(1,2),
     *     WK(1,3),WK(1,4),R)

C The array WK(.,1) is determined so that WK(.,1) approximates Y.
C  This is necessary for accurate calculation of XSQ.

40    IER=0
      DO 60 I=1,N-1
         DYSQ=0.
         DO 50 J=1,K-1
50          DYSQ=DYSQ+T(I+J)
         DYSQ=DYSQ/FLOAT(K-1)
         CALL INTERV(X,NPT,DYSQ,LEFT,MFLAG)
         IF(MFLAG.EQ.1)LEFT=NPT-1
         DYSQ=(DYSQ-X(LEFT))/(X(LEFT+1)-X(LEFT))
60       WK(I,1)=Y(LEFT)*(1.-DYSQ)+Y(LEFT+1)*DYSQ
      WK(N,1)=Y(NPT)
      CALL SETUPR(NKT,T,N,K,NPT,X,Y,E,YSQ,R(1,1,2),WK,WK(1,2),WK(1,3))

C An iteration is begun which changes P1 until XSQ is within
C .1*S of S

      P1=1.
      XSQLO=0.0
      XSQHI=0.0
      DO 220 IT=1,30
         DO 70 I=1,K
         DO 70 J=1,N-I+1
70          R(I,J,1)=P1*R(I,J,3)+R(I,J,2)

C The equation R*BCOEF=VCT is solved by first finding the
C Cholesky factorization of R, then by solving for BCOEF.

         CALL BCHFAC(R,K,N,WK(1,4))
         DO 80 I=1,N
80          BCOEF(I)=WK(I,2)
         CALL BCHSLV(R,K,N,BCOEF)

C The chi-square of the solution is determined.

         DO 90 I=1,N
```

```
C                 over this region. These weights are relative: i.e.   C
C                 changing all the WTI by a constant factor will not   C
C                 affect the resulting spline.                         C
C   IER      = 0 , If JDER,T,WTI and the first N*K elements of R are   C
C                    as on the previous call to BSMTH ( this means     C
C                    that the matrix P need not be recalculated )      C
C            = 1 , if P is to be recalculated                          C
C                                                                      C
C  VIA COMMON / PLIMS / :                                              C
C                                                                      C
C   PMIN     = Min. allowed value of P1 ( See comment describing       C
C                iteration for correct chi-square ).Default is 1.0E-03 C
C   PMAX     = Max. allowed value of P1. Default is 1.0E+03.           C
C                                                                      C
C OUTPUT:                                                              C
C                                                                      C
C   IER      = 0 , Calculation has been successful                     C
C            = 1 , If JDER > K-1                                       C
C            = 2 , If NKT1 < N+K+max(0,K-2*JDER)                       C
C            = 3 , If IWK  < max(NKT1,K**2)                            C
C            = 4 , If more than 30 iterations are required to find the C
C                  correct value for P. Indicates numerical difficul-  C
C                  ties in the solution of the linear system           C
C            = 5 , If the chi-square of the spline > 1.1*S             C
C            = 6 , If the chi-square of the spline < .9*S              C
C   BCOEF    : An array of length N containing the B-spline coeffi-    C
C                cients of the spline.                                 C
C                                                                      C
C  VIA COMMON / CHISQ / :                                              C
C                                                                      C
C   XSQ      = the chi-square of the spline                            C
C                                                                      C
C  WORK SPACE:                                                         C
C                                                                      C
C   R        : An array of length 3*K*N                                C
C   IWK      = max(NKT1,K**2)                                          C
C   WK       : An array of length 4*IWK                                C
C                                                                      C
C----------------------------------------------------------------------C
      REAL BCOEF(N),T(NKT1),WTI(N),R(K,N,3),WK(IWK,4),
     *   X(NPT),Y(NPT),E(NPT),
     *   P1,P2,PHI,PLO,XSQ,XSQHI,XSQLO,YSQ,ALF,DYSQ,S
      INTEGER N,K,NKT,NKT1,JDER,NPT,MFLAG,LEFT,IWK,IER,IT,I,J

      COMMON / CHISQ / XSQ,DUM(3)
      COMMON / PLIMS / PMIN,PMAX

      DATA PMIN / 1.0E-03 /,PMAX / 1.0E+03 /

C The input data is checked for simple errors

      NKT=N+K
      IF(JDER.LT.K)GO TO 10
```

B.2 BSMTH

```
      SUBROUTINE BSMTH(S,JDER,NPT,X,Y,E,N,K,NKT1,T,WTI,BCOEF,
     *   R,IWK,WK,IER)
C-----------------------------------------------------------------C
C                                                                 C
C BSMTH calculates the spline of order K, with knots T(I),I=1,NKT C
C  which has chi-square of S with respect to the data points      C
C  X(I),Y(I),I=1,NPT,  and which has as small a JDER-th derivative C
C  as possible.                                                   C
C                                                                 C
C AUTHOR: David Hally , May 1981                                  C
C                                                                 C
C USAGE:                                                          C
C       EXECUTE mainpgm,BSPLIN:HLLYSP/LIB,BSPLIN:BSPLIN/LIB       C
C                                                                 C
C CALLS SETUPQ,SETUPR,XSQC                                        C
C       from BSPLIN library: INTERV,BCHFAC,BCHSLV                 C
C                                                                 C
C INPUT:                                                          C
C                                                                 C
C   S       : The chi-square of the spline with respect to the data C
C               will be within 10% of S, if possible. As S is     C
C               increased the spline becomes smoother but farther C
C               from the data points. Function PRERR can be used  C
C               to give a value for S if a reasonable value is not C
C               known.                                            C
C   JDER    : The integral of the square of the JDER-th derivative C
C               of the spline is minimized ( subject to the con-  C
C               straint that XSQ=S). If smooth curves are desired C
C               a value of JDER=2 is appropriate. JDER should be  C
C               non-negative and less than K.                     C
C   NPT     : The no. of data points                              C
C   X       : An array of length NPT containing the data point   C
C               abscissae in ascending order.                     C
C   Y       : An array of length NPT containing the data point   C
C               ordinates.                                        C
C   E       : The  errors of the data points. The smaller         C
C               the error the closer the spline will come         C
C               to that point.                                    C
C   N       : The no. of B-splines.                               C
C   K       : The order of the spline.                            C
C   NKT1    = N+K+max(0,K-2*JDER)                                 C
C   T       : An array of length NKT1 the first N+K elements of  C
C               which contain the knot sequence (in ascending order). C
C               the remaining array elements are used in subroutine C
C               SETUPP.                                           C
C   WTI     : An array of length N of which only the first N-K+1  C
C               elements are used (rather than passing in an other- C
C               wise superfluous argument). WTI(I) is a weight    C
C               for the integral of the square of the JDER-th deriv- C
c               ative of the spline between T(I+K-1) and T(I+K). The C
C               larger WTI(I) is the smaller the integral will be C
```

```
          IMAX=5
          JDER=2
          DO 40 I=1,N+K
40           T(I)=FLOAT(I-K)/FLOAT(NWTI)

C  The error in the X-values are found by calling the function
C   PRERR and the integral weights, WTI, by calling WTIBEG.
C   They are splined using BSMTH using fractional arc length
C   to parametrize the data points. Similarly for the Y-values.

C   NOTE: The parameter SM to be used in BSMTH should be
C    expected to be NPT*PRERR**2.
C    However, due to the sensitivity of parametric splines to
C    data error, it has been found that slightly higher values of
C    SM sometimes give better results. SMFACT has been included as
C    a knob to increase (or decrease) SM : SM=SMFACT*NPT*PRERR**2 .
C    default value for SMFACT is 1.0.

150      IER=1
         CALL WTIBEG(NPT,ARCL,X,N+K,T,NWTI,WTI)
         SX=SMFACT*PRERR(NPT,ARCL,X,E,IMAX,G,WK,0)**2*FLOAT(NPT)
         CALL BSMTH(SX,JDER,NPT,ARCL,X,E,N,K,NKT1,T,WTI,BCOEFX,
     *          R,IWK,WK,IER)
         XSQX=XSQY
         IF((IER.NE.0).AND.(IER.LT.4))RETURN
         IER=1
         SY=SMFACT*PRERR(NPT,ARCL,Y,E,IMAX,G,WK,1)**2*FLOAT(NPT)
         CALL WTIBEG(NPT,ARCL,Y,N+K,T,NWTI,WTI)
         CALL BSMTH(SY,JDER,NPT,ARCL,Y,E,N,K,NKT1,T,WTI,BCOEFY,
     *          R,IWK,WK,IER)
         RETURN
         END
```

```
C   BCOEFY  = Array of length N containing the B-spline coefs. for    C
C                the Y-values of the curve                             C
C   ARCL(I) = Arc length at the I-th data point/total length of curve C
C                                                                      C
C  VIA COMMON / CHISQ /                                                C
C                                                                      C
C   XSQX = Chi-square of the spline of the abscissae                  C
C   XSQY = Chi-square of the spline of the ordinates                  C
C   SX   = Required Chi-square of the abscissae ( as determined by    C
C            PRERR )                                                   C
C   SY   = Required Chi-square of the ordinates ( as determined by    C
C            PRERR )                                                   C
C                                                                      C
C  VIA COMMON /CRVLTH/ :                                               C
C                                                                      C
C   SNEWL    : The total arc length of the curve                      C
C                                                                      C
C  WORK SPACE :                                                        C
C                                                                      C
C   R        : An array of length 3*K*N                               C
C   IWK      = max(NKT1,K**2)                                          C
C   WK       : An array of length 4*IWK                               C
C   G        : An array of dimensions NPT,IMAX (used by PRERR)        C
C   WTI      : Array of length N used for the integral weights for    C
C               BSMTH                                                  C
C                                                                      C
C                                                                      C
C---------------------------------------------------------------------C
      REAL X(NPT),Y(NPT),E(NPT),ARCL(NPT),BCOEFY(N),BCOEFX(N),
     *     T(NKT1),WTI(N),G(NPT,IMAX),WK(IWK,4),R(K,N,3),
     *     SMFACT
      INTEGER NPT,N,K,NKT1,NWTI,IWK,IER,IMAX,I,K1,IW,ID

      COMMON /NODFLT/ SMFACT,IMAX
      COMMON /INTEXP/ JDER
      COMMON /CHISQ/ XSQY,XSQX,SY,SX

      NWTI=N-K+1

C  ARCL(I),I=1,N is initialized by connecting the data points with
C   straight lines.

      ARCL(1)=0.0
      DO 10 I=2,NPT
10       ARCL(I)=ARCL(I-1)+SQRT((X(I)-X(I-1))**2+(Y(I)-Y(I-1))**2)
      OLDL=ARCL(NPT)
      DO 20 I=2,NPT
20       ARCL(I)=ARCL(I)/OLDL

C  If IER.NE.0 non-default values of SMFACT,IMAX and IMAX are
C   taken from the COMMON block /NODFLT/ and JDER from COMMON /INTEXP/

      IF(IER.NE.0)GO TO 150
         SMFACT=1.0
```

```
C  are calculated by calculating the coefs. of the knot sequence
C  corresponding to the integral of each B-spline and then calling
C  BVALUE to evaluate these at the appropriate points.

          IF(2*JDER.GE.K)GO TO 90
          DO 80 J=1,K
             IKJ=I-K+J
             WK(IKJ)=1.
             IF(J.EQ.K)GO TO 50
             DO 40 L=IKJ+1,I+1
40              WK(L)=0.0
50           DO 70 M=1,K-2*JDER
                KM=K+M-1
                DO 60 L=IKJ,I+1
                   WK(L)=WK(L)*(T(L+KM)-T(L))/FLOAT(KM)
                   IF(L.NE.1)WK(L)=WK(L)+WK(L-1)
60              CONTINUE
                A1(J,M)=BVALUE(T,WK,N,K+M,T2,0)
70              A(J,M)=BVALUE(T,WK,N,K+M,T1,0)
80           WK(IKJ)=0.0

C  The elements of P are determined by integration by parts.

90        DO 130 L=1,K
             DO 130 J=1,L
                LJ1=L-J+1
                IKJ=I-K+J
                HI=H
                IF(MMAX.LT.1)GO TO 110
                DO 100 M=1,MMAX
                   P(LJ1,IKJ)=P(LJ1,IKJ)+HI*WTI(I1)*(DB1(L,JDER-M+1)*
     *               DB1(J,JDER+M)-DB(L,JDER-M+1)*DB(J,JDER+M))
100                HI=-HI
110          IF(K.LE.2*JDER)GO TO 130
             DO 120 M=1,K-2*JDER
                P(LJ1,IKJ)=P(LJ1,IKJ)+HI*WTI(I1)*(A1(L,M)*DB1(J,JDER
     *                      +MMAX+M)-A(L,M)*DB(J,JDER+MMAX+M))
120             HI=-HI
130       CONTINUE
140    CONTINUE
       RETURN
       END
```

B.7 SETUPR

```
      SUBROUTINE SETUPR(NKT,T,N,K,NPT,X,Y,E,YSQ,R,Y1,VCT,VCT1)
C-----------------------------------------------------------------C
C                                                                 C
C The matrix R,the arrays Y1,VCT and VCT1, and the number YSQ are C
C  calculated                                                     C
C  ( SETUPR is based closely on L2APPR by Carl de Boor,in         C
C    A Practical Guide to Splines, p. 255)                        C
C                                                                 C
C AUTHOR: David Hally , May. 1981                                 C
C                                                                 C
C CALLED BY BSMTH                                                 C
C                                                                 C
C CALLS from BSPLIN library BSPLVB                                C
C                                                                 C
C-----------------------------------------------------------------C
      REAL T(NKT),R(K,N),VCT(N),BIATX(20),X(NPT),Y(NPT),E(NPT),DW,
     *     Y1(N),VCT1(N),YSQ,DYSQ
      INTEGER N,K,NKT,NPT,LEFT,LEFTMK,I,J,MM,JJ,LL
      YSQ=0.
      DO 20 J=1,N
         VCT1(J)=0.
         VCT(J)=0.
         DO 10 I=1,K
            R(I,J)=0.
10       CONTINUE
20    CONTINUE

C The LL-th data point is positioned within the knot sequence.

      LEFT=K
      LEFTMK=0
      DO 80 LL=1,NPT
30       IF(LEFT.EQ.N)GO TO 40
         IF(X(LL).LT.T(LEFT+1))GO TO 40
         LEFT=LEFT+1
         LEFTMK=LEFTMK+1
         GO TO 30

C R is calculated by calling BSPLVB to evaluate the B-splines at
C  the data points.

40       CALL BSPLVB(T,K,1,X(LL),LEFT,BIATX)
         DYSQ=Y(LL)
         DO 50 MM=1,K
            DYSQ=DYSQ-BIATX(MM)*Y1(LEFT-K+MM)
50       CONTINUE
         DO 70 MM=1,K
            DW=BIATX(MM)/E(LL)**2
            J=LEFTMK+MM
            VCT1(J)=VCT1(J)+DYSQ*DW
            VCT(J)=DW*Y(LL)+VCT(J)
```

```
            I=1
            DO 60 JJ=MM,K
               R(I,J)=BIATX(JJ)*DW+R(I,J)
               I=I+1
60          CONTINUE
70       CONTINUE
         YSQ=(DYSQ/E(LL))**2+YSQ
80    CONTINUE
      RETURN
      END
```

B.8 SMODAV

```
      FUNCTION SMODAV(NPT,X)
C---------------------------------------------------------------------C
C                                                                     C
C SMODAV returns a modal average of the numbers in X                  C
C                                                                     C
C AUTHOR : David Hally , Aug. 1981                                    C
C                                                                     C
C USAGE :                                                             C
C       EXECUTE main-pgm,BSPLIN:HLLYSP/LIB                            C
C                                                                     C
C INPUT :                                                             C
C                                                                     C
C  NPT   = No. of values to be averaged                               C
C  X     = Array of length NPT containing values to be averaged       C
C                                                                     C
C RETURNS:                                                            C
C                                                                     C
C  SMODAV = Modal average of the values in X                          C
C                                                                     C
C---------------------------------------------------------------------C
      REAL X(NPT),XBOX(11),SUMBOX(10),XMIN,XMAX,XRATIO,SCALE,SMODAV
      INTEGER IBOX(10),NPT,ISUM,NBOX,I,J

C The range of the values is found and broken into NBOX logarithmic
C  intervals, such that the ratio of the smallest to the largest
C  possible no. in each interval does not exceed NPT, but also
C  subject to the constraint 2 < NBOX < 11.

      XMIN=1.0E+30
      XMAX=1.0E-30
      DO 10 I=1,NPT
         IF(X(I).LE.0.0)GO TO 10
         XMAX=AMAX1(X(I),XMAX)
         XMIN=AMIN1(X(I),XMIN)
10    CONTINUE
      IF(XMIN.EQ.1.0E+30)GO TO 90
      XRATIO=XMAX/XMIN
      NBOX=ALOG10(XRATIO)/ALOG10(FLOAT(NPT))
      NBOX=MIN0(NBOX,10,NPT/5)
      NBOX=MAX0(3,NBOX)
      SCALE=XRATIO**(1./NBOX)
      XBOX(1)=XMIN
      XBOX(NBOX+1)=XMAX

C The no. of X-values within each interval is calculated

      DO 20 I=1,NBOX
         SUMBOX(I)=0.0
20       IBOX(I)=0
      DO 30 I=2,NBOX
30       XBOX(I)=XBOX(I-1)*SCALE
```

```
      DO 60 I=1,NPT
         DO 40 J=2,NBOX+1
40          IF(X(I).LE.XBOX(J))GO TO 50
50       SUMBOX(J-1)=SUMBOX(J-1)+X(I)
60       IBOX(J-1)=IBOX(J-1)+1

C Denote by Xmid the X-value such that there are an equal no. of X-values
C  both smaller and greater than Xmid. SMODAV is the average value of all
C  the X's in the interval containing Xmid.

      ISUM=0
      DO 70 I=1,NBOX
         ISUM=ISUM+IBOX(I)
         IF(ISUM.GE.NPT/2)GO TO 80
70    CONTINUE
80    SMODAV=SUMBOX(I)/IBOX(I)
      RETURN

90    SMODAV=0.0
      RETURN
      END
```

B.9 WTIBEG

```
      SUBROUTINE WTIBEG(NPT,X,Y,NKT,T,NWTI,WTI)
C-----------------------------------------------------------------C
C                                                                 C
C WTIBEG uses the data points to predict values for the integral  C
C  weights WTI for use in BSMTH.                                  C
C                                                                 C
C AUTHOR: David Hally , Aug. 1981                                 C
C                                                                 C
C USAGE:                                                          C
C       EXECUTE mainpgm,BSPLIN:HLLYSP/LIB                         C
C                                                                 C
C INPUT:                                                          C
C                                                                 C
C  NPT     = No. of data points                                   C
C  X       = Array of length NPT containing data point abscissae  C
C  Y       = Array of length NPT containing data point ordinates  C
C  NKT     = No. of knots                                         C
C  T       = Array of length NKT containing the knots             C
C  NWTI    = No. of integral weights ( = no. of B-splines -       C
C              order of spline +1 )                               C
C                                                                 C
C OUTPUT:                                                         C
C                                                                 C
C  WTI     = Array of length NWTI containing the integral weights C
C                                                                 C
C-----------------------------------------------------------------C
      REAL X(NPT),Y(NPT),T(NKT),WTI(NWTI),
     *     HL,HR,TL,TR,D1YL,D1YR,D2YDL,D2YDR,D2YTL,D2YTR,
     *     SLOPE,WMIN,WMAX,WTIAV,DUMMY
      INTEGER NPT,NWTI,NKT,K,ID,IT,IW

      K=(NKT-NWTI+1)/2

C 1st and 2nd derivatives at the first two data points and at the
C  end-points of the first knot interval are approximated by divided
C  differences.

      IT=K
      TL=T(K)
      ID=1
      IW=1
      WTI(1)=0.0
      HL=X(2)-X(1)
      HR=X(3)-X(2)
      D1YL=(Y(2)-Y(1))/HL
      D1YR=(Y(3)-Y(2))/HR
      D2YDL=(D1YR-D1YL)/(X(3)-X(1))
      D2YDR=D2YDL
      D2YTL=D2YDL
      SLOPE=0.0
```

```
C The next interval of interest is the interval from the right end of
C  the current interval to the next data point or the next knot,
C  whichever occurs first. The contribution to WTI from this interval
C  is determined.

10      TR=AMIN1(T(IT+1),X(ID+1))
        D2YTR=D2YDL+SLOPE*(TR-X(ID))
        WTI(IW)=WTI(IW)+HL*(D2YTL*(D2YTL+D2YTR)+D2YTR**2)/3.

        IF(IT.EQ.NWTI+K-1)GO TO 50
        TL=TR
        D2YTL=D2YTR
        IF(TR.NE.X(ID+1))GO TO 30
        ID=ID+1
        D2YDL=D2YDR
        D1YL=D1YR
        HL=HR
        IF(ID.GT.NPT-2)GO TO 20
        HR=X(ID+2)-X(ID+1)
        D1YR=(Y(ID+2)-Y(ID+1))/HR
        D2YDR=(D1YR-D1YL)/(HR+HL)
20      SLOPE=(D2YDR-D2YDL)/HL
30      IF(TR.NE.T(IT+1))GO TO 10
        IT=IT+1
        IW=IW+1
        WTI(IW)=0.0
40      IF(T(IT+1).NE.T(IT))GO TO 10
        IT=IT+1
        IW=IW+1
        WTI(IW)=0.0
        GO TO 40

C The WTI are normalized so that most of them are of order 1.

50      WTIAV=SMODAV(NWTI,WTI)
        IF(WTIAV.EQ.0.0)GO TO 70
        WMIN=WTIAV*1.0E-03
        WMAX=WTIAV*1.0E+03
        DO 60 IW=1,NWTI
           DUMMY=WTI(IW)
           IF((WTI(IW).GT.WMIN).AND.(WTI(IW).LT.WMAX))WTI(IW)=
     *                       WTIAV/WTI(IW)
             IF(DUMMY.GT.WMAX)WTI(IW)=1.0E-03
60         IF(DUMMY.LE.WMIN)WTI(IW)=1.0E+03
        RETURN
70      DO 80 IW=1,NWTI
80         WTI(IW)=1.0
        RETURN
        END
```

B.10 XSQC

```
      FUNCTION XSQC(N,K,BCOEF,R,VCT,WK.YSQ)
C-----------------------------------------------------------------C
C                                                                 C
C XSQC calculates the chi-square:                                 C
C  XSQ= SUM( (Y(I)- SUM( (BCOEF(J)-Y1(J))*BJ(X(I)) ))**2 *E(I) )  C
C  By subtracting Y1 from BCOEF one keeps the numbers fairly small C
C  thus avoiding round-off error.                                 C
C                                                                 C
C AUTHOR: David Hally , May. 1981                                 C
C                                                                 C
C CALLED by BSMTH                                                 C
C                                                                 C
C-----------------------------------------------------------------C
      REAL BCOEF(N),R(K,N),VCT(N),WK(N),XSQC,YSQ
      INTEGER I,N,K,J,IJ1

      XSQC=YSQ
      DO 10 I=1,N
10       XSQC=XSQC-2.*BCOEF(I)*VCT(I)
      DO 20 I=1,N
20       WK(I)=0.0
      DO 30 I=1,K
         DO 30 J=1,N-I+1
            IJ1=I+J-1
            WK(J)=WK(J)+R(I,J)*BCOEF(IJ1)
            IF(I.EQ.1)GO TO 30
            WK(IJ1)=WK(IJ1)+R(I,J)*BCOEF(J)
30       CONTINUE
      DO 40 I=1,N
40       XSQC=XSQC+WK(I)*BCOEF(I)
      RETURN
      END
```

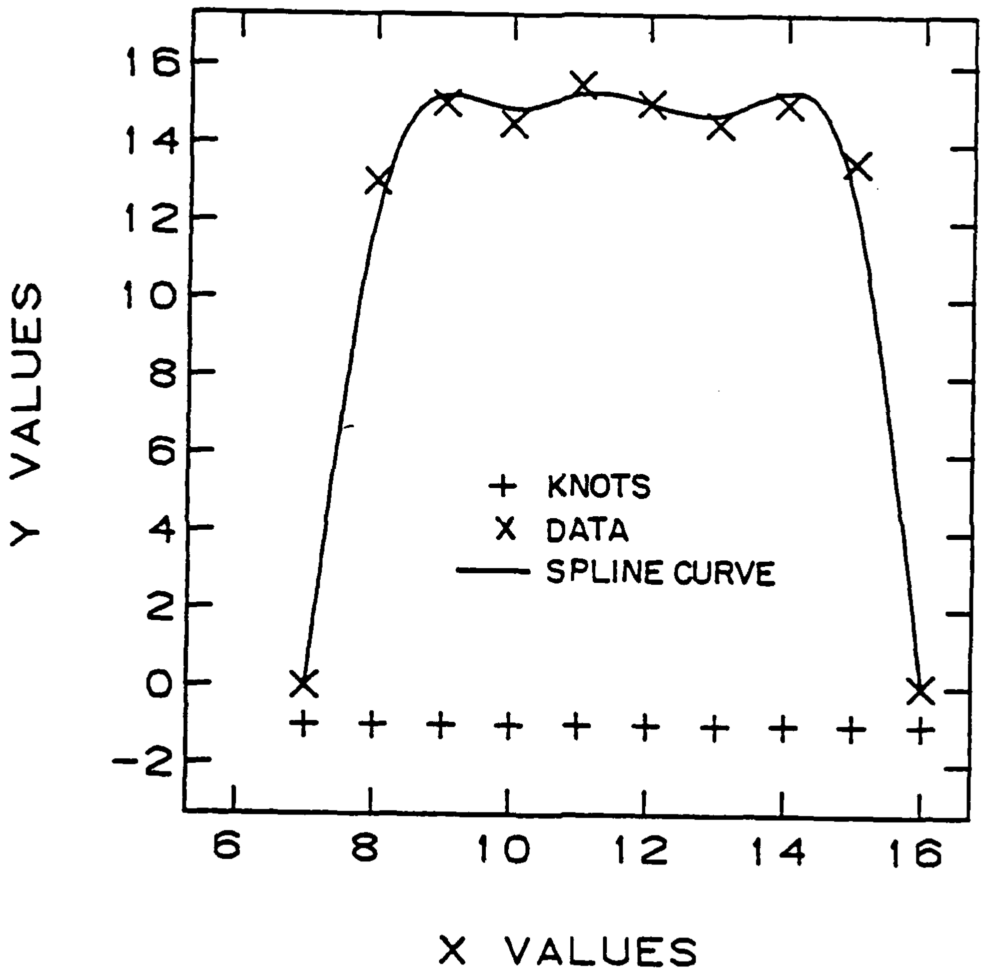


FIG. 1 SPLINE CALCULATED BY SMOOTH

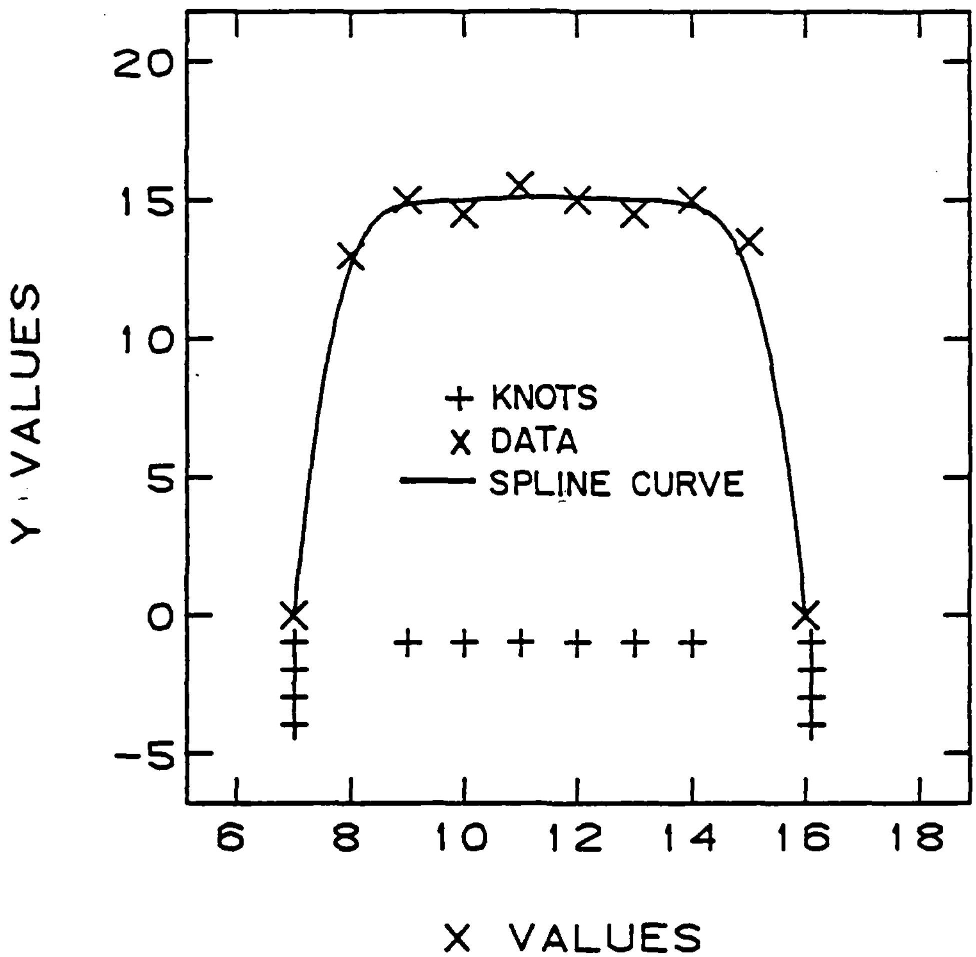


FIG. 2 SPLINE CALCULATED BY BSMTH: k = 4, STIFFNESS WEIGHTS FROM WTIBEG

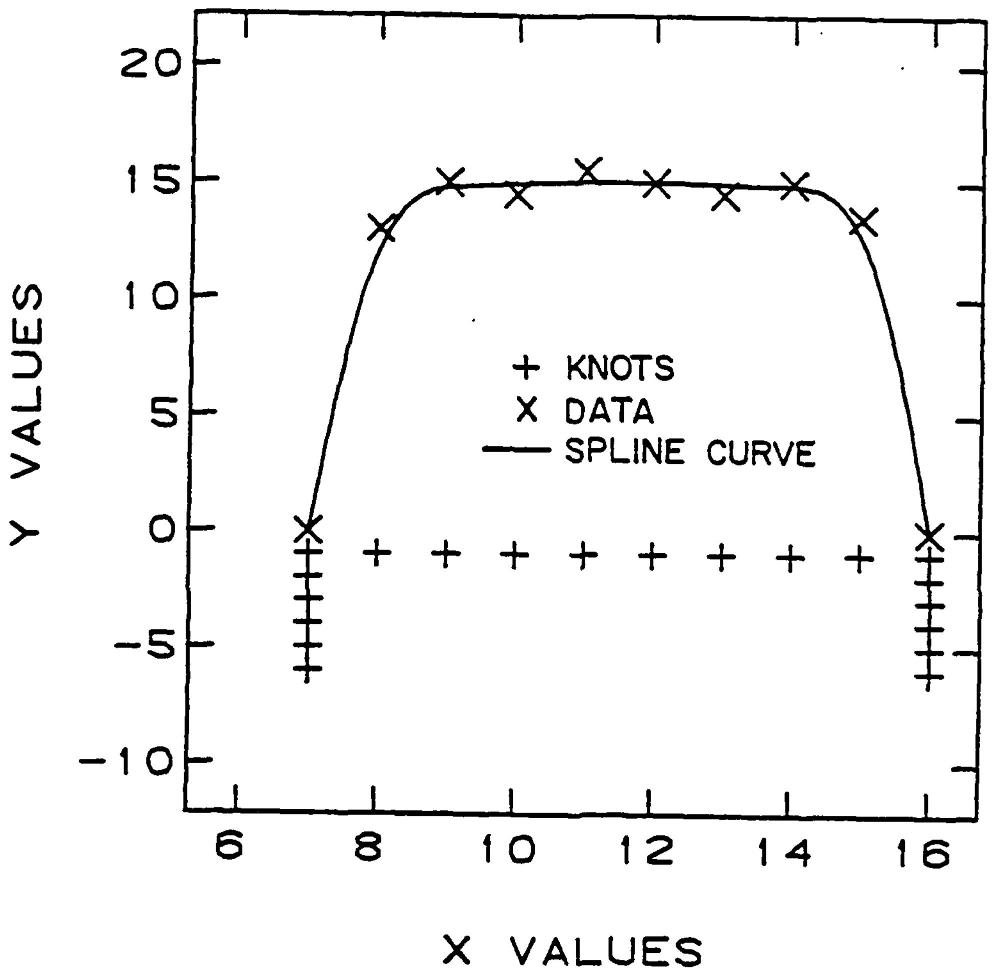


FIG. 3 SPLINE CALCULATED BY BSMTH: k = 6, STIFFNESS WEIGHTS FROM WTIBEG

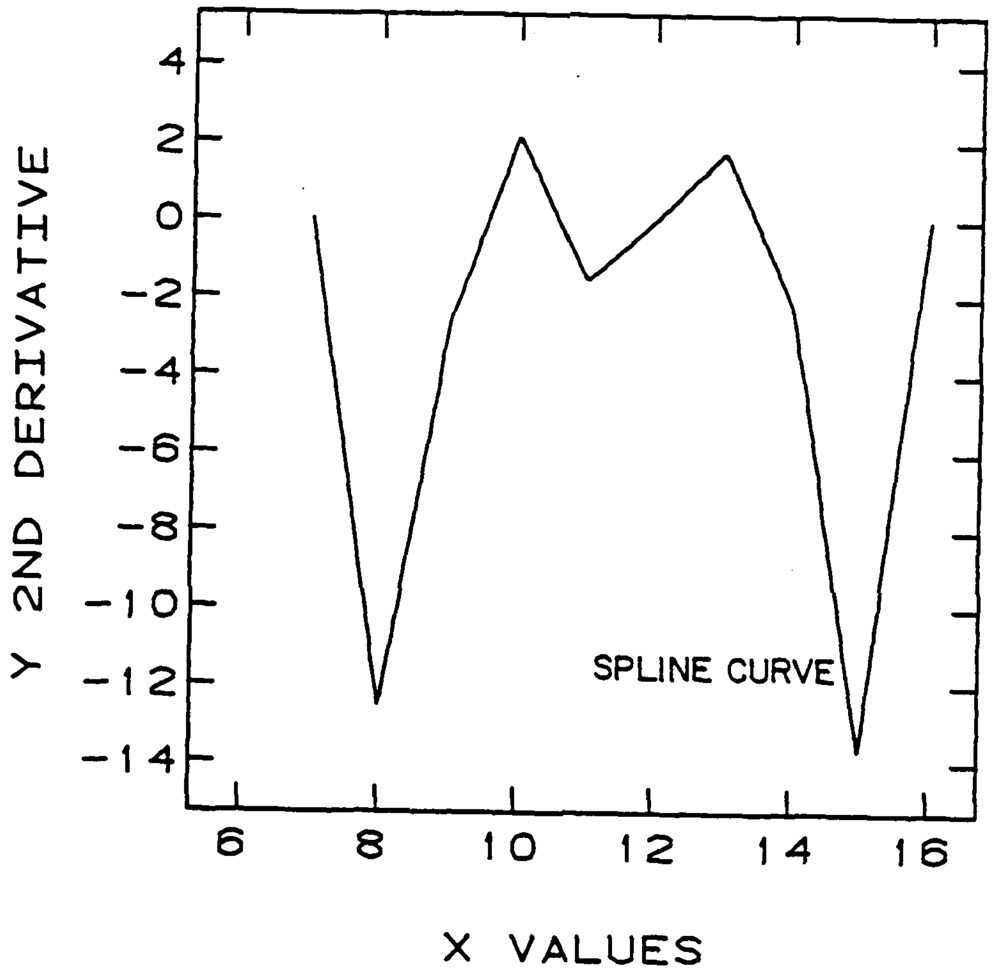


FIG. 4 2ND DERIVATIVE OF SPLINE CALCULATED BY SMOOTH

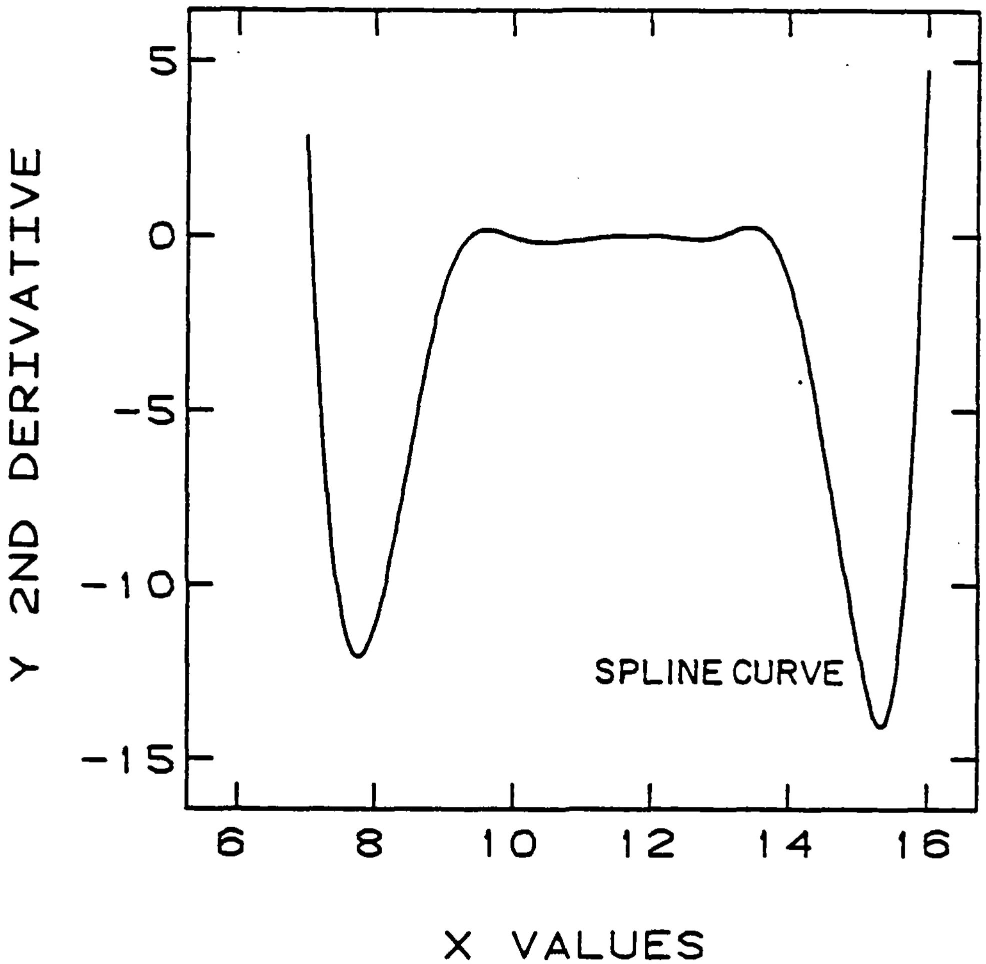


FIG. 5 2ND DERIVATIVE OF SPLINE CALCULATED BY BSMTH, k = 6

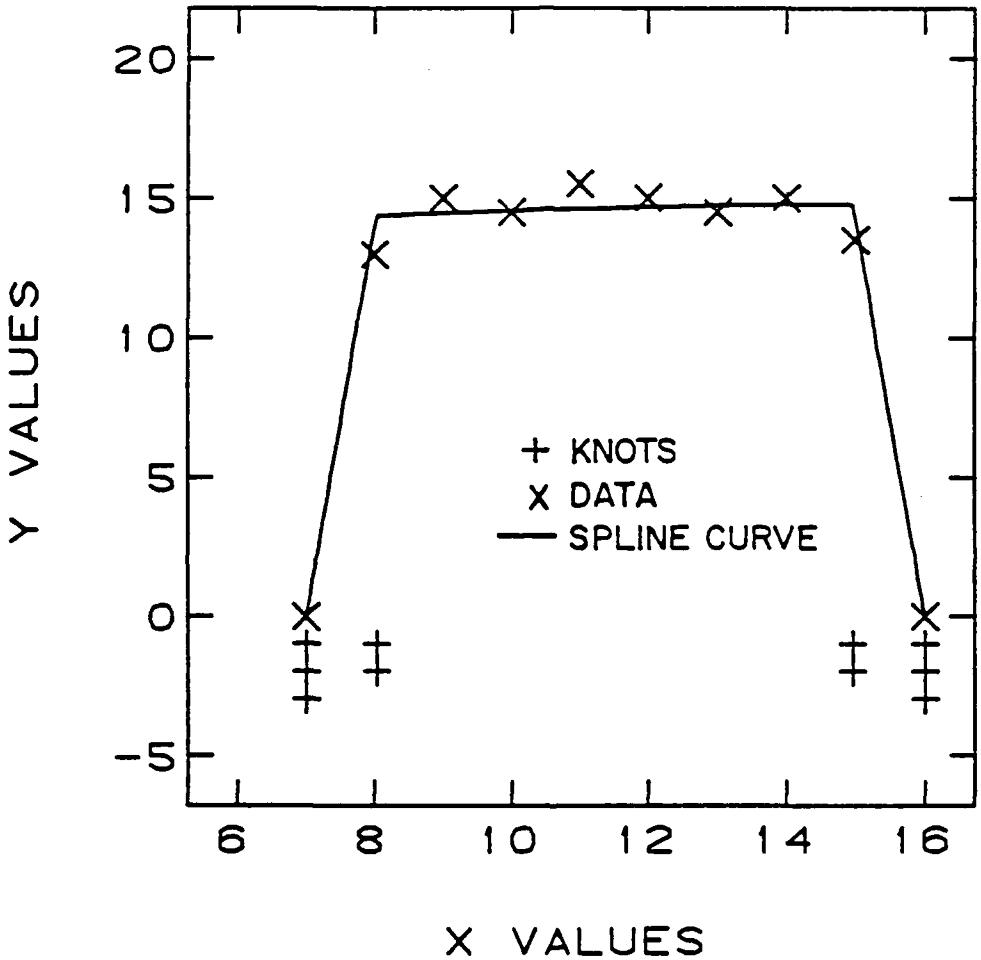


FIG. 6 SPLINE CALCULATED BY BSMTH, k = 3 DISCONTINUOUS DERIVATIVE

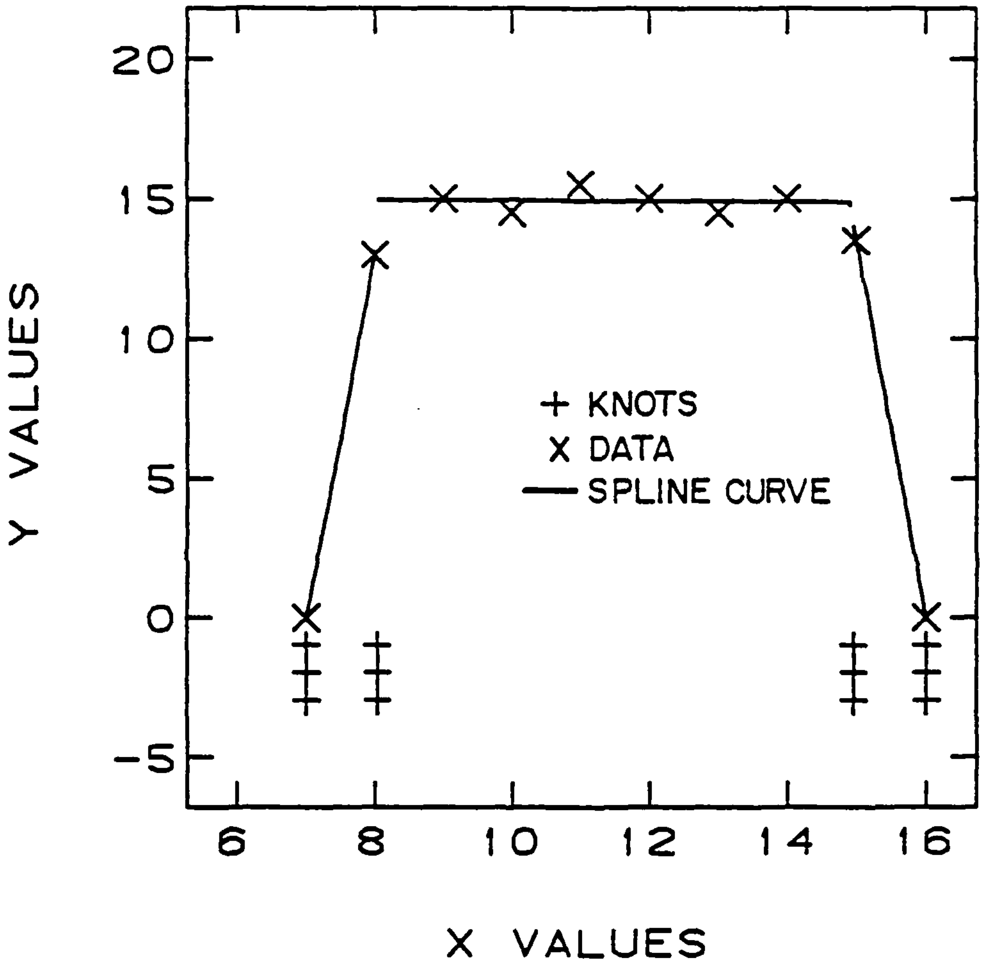


FIG. 7 SPLINE CALCULATED BY BSMTH : k = 3 DISCONTINUOUS SPLINE

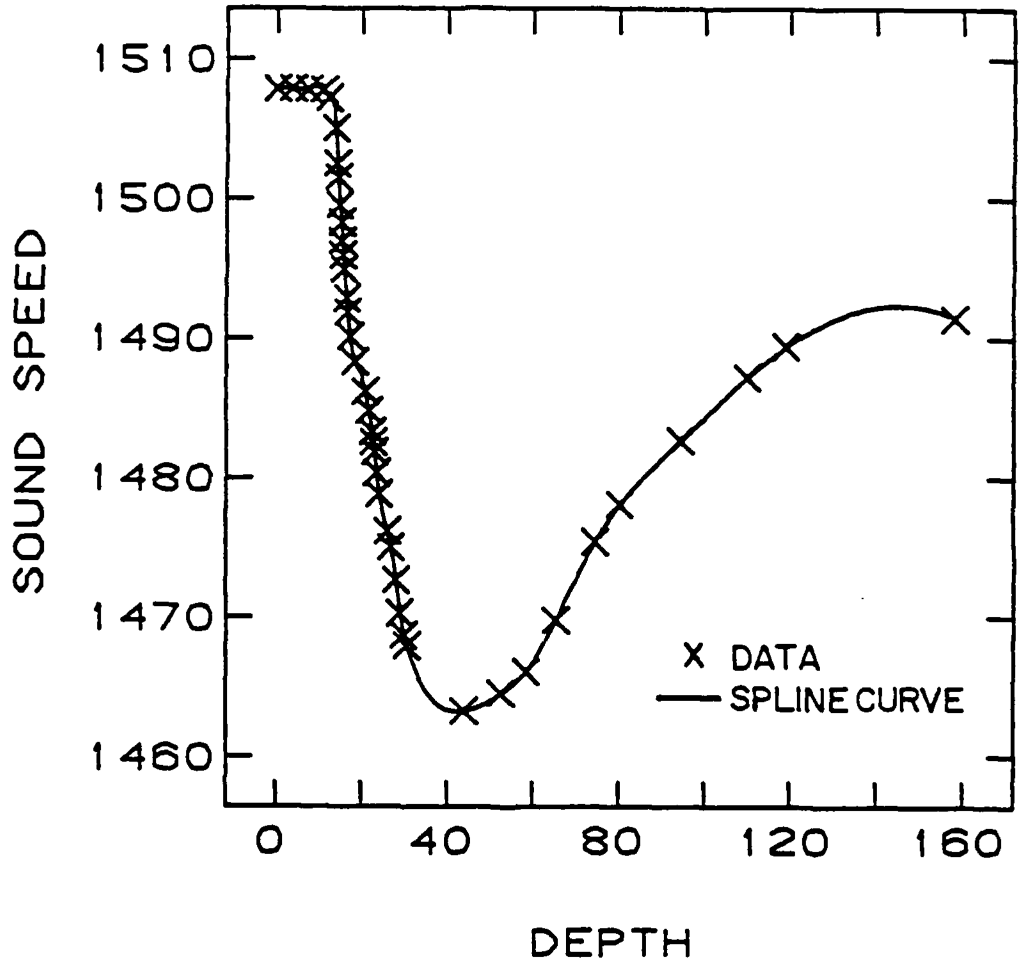


FIG. 8 SOUND SPEED PROFILE: DATA POINTS AND SPLINE CALCULATED BY CUBSPL

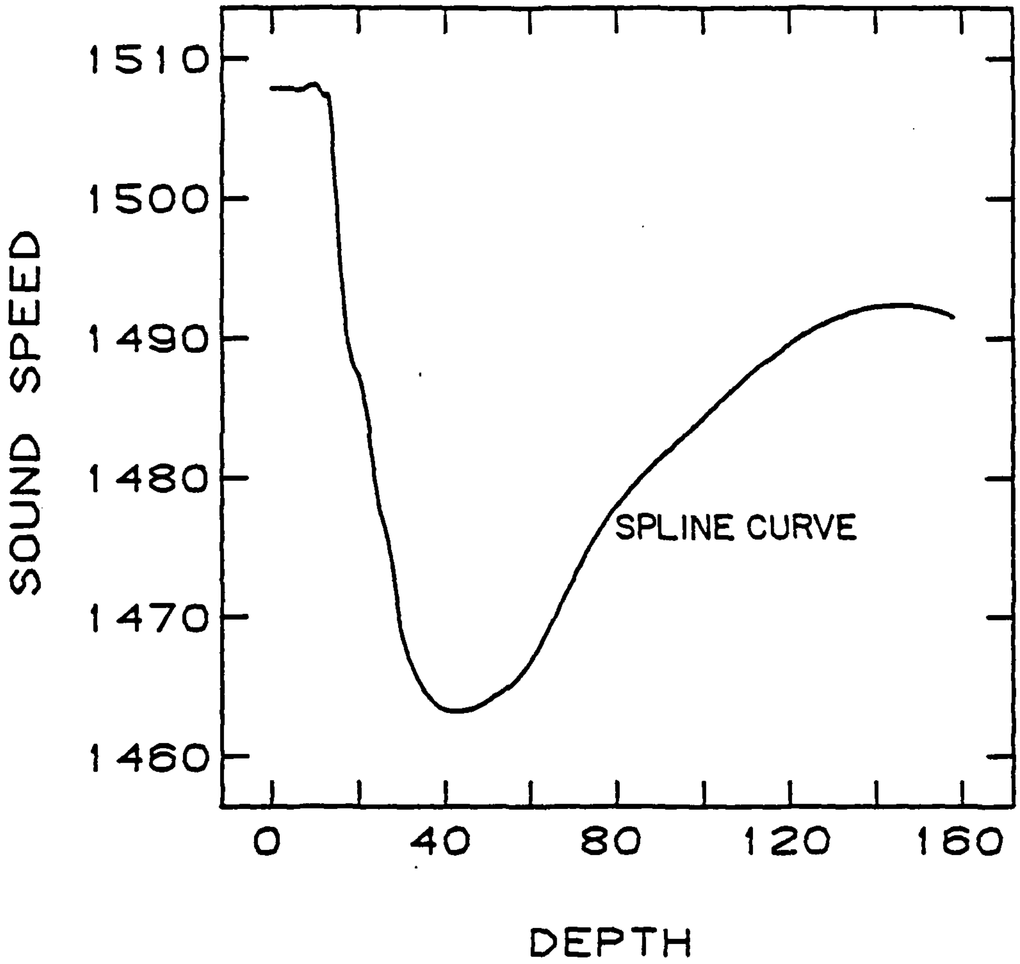


FIG. 9 SOUND SPEED PROFILE: SPLINE CALCULATED BY CUBSPL

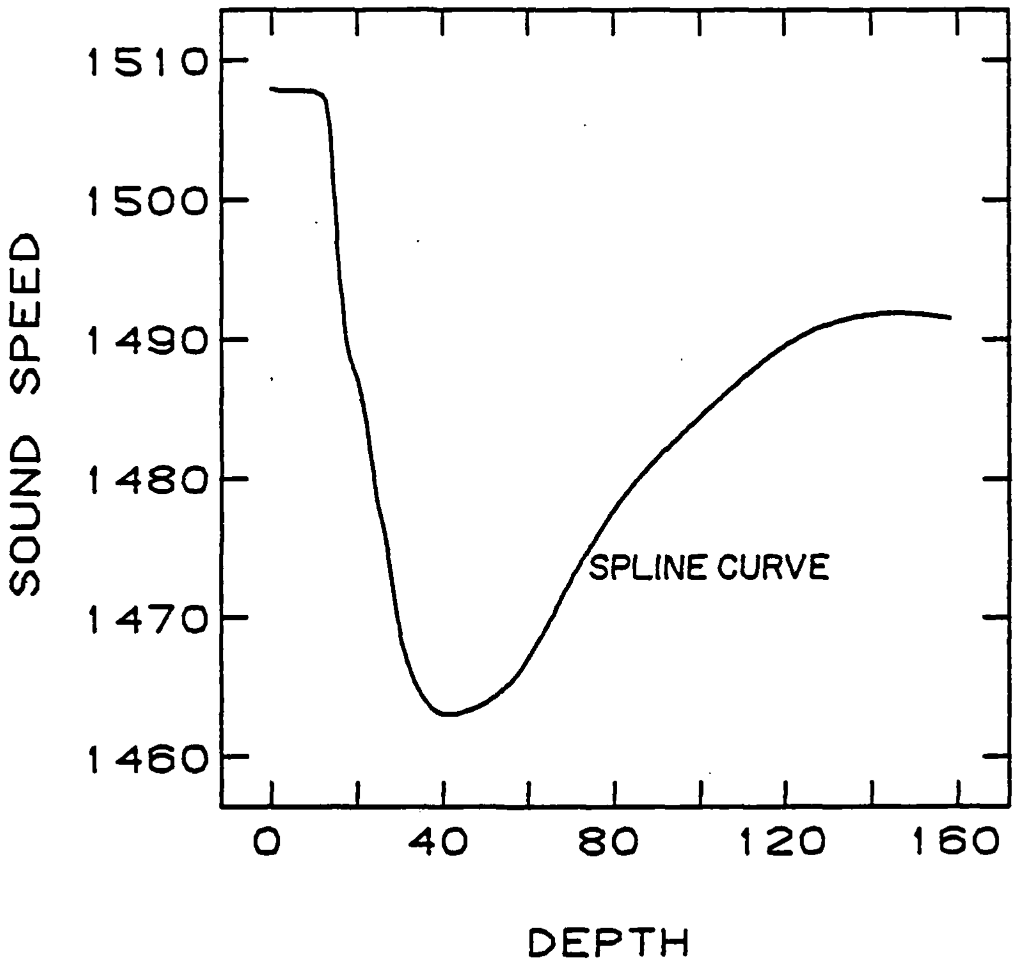


FIG. 10 SOUND SPEED PROFILE: SPLINE CALCULATED BY BSMTH. SMOOTHING PARAMETER FROM PRERR.

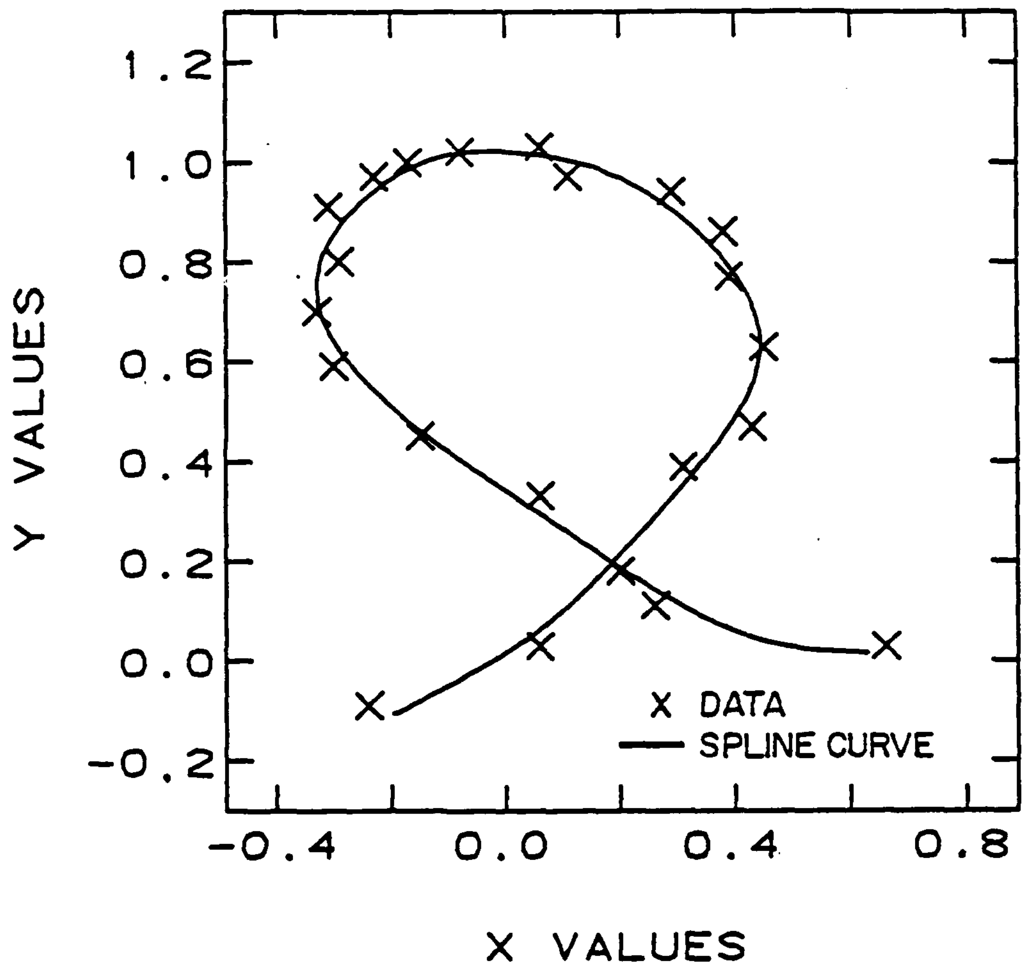


FIG. 11 SPLINE CALCULATED BY BSMCRV

References

1. de Boor,C.; A Practical Guide to Splines, Springer Verlag, New York (1978)

2. IMSL; "Interpolation, Approximation and Smoothing", IMSL Reference Manual, IMSL-LIB 0008, IMSL, Houston, Texas (1980)

3. Reinsch,C.H.; "Smoothing by Spline Functions",Numer. Math. 10, 177 (1967)

4. Whittaker,E.T.; "On a New Method of Graduation", Proc. Edinburgh Math. Soc., 41, 63 (1923)

5. Schoenberg,I.J.; "Spline Functions and the Problem of Graduation", Proc. Nat. Acad. Sci. 52, 947 (1964)

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATING ACTIVITY DEFENCE RESEARCH ESTABLISHMENT ATLANTIC	2a. DOCUMENT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP
3. DOCUMENT TITLE A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES	
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) Technical Memorandum	
5. AUTHOR(S) (Last name, first name, middle initial) Hally, David	
6. DOCUMENT DATE June 85	7a. TOTAL NO. OF PAGES 72 / 7b. NO. OF REFS 5
8a. PROJECT OR GRANT NO.	9a. ORIGINATOR'S DOCUMENT NUMBER(S) Technical Memorandum 85/205
8b. CONTRACT NO.	9b. OTHER DOCUMENT NO.(S) (Any other numbers that may be assigned this document)
10. DISTRIBUTION STATEMENT UNLIMITED	
11. SUPPLEMENTARY NOTES	12. SPONSORING ACTIVITY

13. ABSTRACT

DREA has, at present, two libraries containing subroutines for calculating splines: IMSL and BSPLIN. A new library has been developed to supplement the IMSL and BSPLIN routines in the realm of smoothing splines. It is not self-contained, making frequent use of subroutines from the BSPLIN library.

The new subroutines offer several advan-ages over the smoothing spline subroutines in the IMSL and BSPLIN libraries.

1) The order of the spline may be picked by the user.
2) The second derivative of the spline is not constrained to be zero at its end-points.
3) The user of the new subroutines has freedom to choose the number and positions of the knots of the spline.
4) The new subroutines have, as input, an extra set of weights, δ_i, i=1,N, which control the stiffness of the spline between each pair of knots.

The new subroutines were initially developed for use in ship hull approximation for the calculation of boundary layer growth on the hull. For this calculation one needs splines whose second derivatives are very well behaved. The additional control afforded by the new subroutines makes them far more suitable for this application than any of the subroutines currently available in either the IMSL or BSPLIN libraries.

PREVIOUS PAGE IS BLANK

UNCLASSIFIED

Security Classification

KEY WORDS

Splines
Smoothing
Interpolation
Computer programs

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the organization issuing the document.

2a. DOCUMENT SECURITY CLASSIFICATION: Enter the overall security classification of the document including special warning terms whenever applicable.

2b. GROUP: Enter security reclassification group number. The three groups are defined in Appendix 'M' of the DRB Security Regulations.

3. DOCUMENT TITLE: Enter the complete document title in all capital letters. Titles in all cases should be unclassified. If a sufficiently descriptive title cannot be selected without classification, show title classification with the usual one-capital-letter abbreviation in parentheses immediately following the title.

4. DESCRIPTIVE NOTES: Enter the category of document, e.g. technical report, technical note or technical letter. If appropriate, enter the type of document, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the document. Enter last name, first name, middle initial. If military, show rank. The name of the principal author is an absolute minimum requirement.

6. DOCUMENT DATE: Enter the date (month, year) of Establishment approval for publication of the document.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the document.

8a. PROJECT OR GRANT NUMBER: If appropriate, enter the applicable research and development project or grant number under which the document was written.

8b. CONTRACT NUMBER: If appropriate, enter the applicable number under which the document was written.

9a. ORIGINATOR'S DOCUMENT NUMBER(S): Enter the official document number by which the document will be identified and controlled by the originating activity. This number must be unique to this document.

9b. OTHER DOCUMENT NUMBER(S): If the document has been assigned any other document numbers (either by the originator or by the sponsor), also enter this number(s).

10. DISTRIBUTION STATEMENT: Enter any limitations on further dissemination of the document, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this document from their defence documentation center."

(2) "Announcement and dissemination of this document is not authorized without prior approval from originating activity."

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document, even though it may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall end with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (TS), (S), (C), (R), or (U).

The length of the abstract should be limited to 20 single-spaced standard typewritten lines; 7½ inches long.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a document and could be helpful in cataloging the document. Key words should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context.

DATE
ILMED
-8