UNCLASSIFIED

A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES
DEFENCE RESEARCH ESTABLISHMENT ATLANTIC
DARTMOUTH (NOVA SCOTIA) JUNE 85 DREA-TM-85/205

END

10-85
A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES

David Hally
 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.

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

CENTRE DE RECHERCHES POUR LA DÉFENSE ATLANTIQUE
9 GROVE STREET
C.P. 1012
DARTMOUTH, N.É.
B2Y 3Z7
A NEW LIBRARY OF SUBROUTINES
FOR
CALCULATING SMOOTHING SPLINES

David Hally
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.

[Signature]

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

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

TELEPHONE 1603/428-3100
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;
4) The new subroutines have, as input, an extra set of weights, $\delta_i$, $i=1,N$, 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.
Résumé

L'ERDA possède maintenant deux bibliothèques contenant des sous-programmes pour calculer des splines : IMSL et BSPLIN. Une nouvelle bibliothèque a été mise sur pied pour compléter les programmes IMSL 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 dérivée 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 \( \delta_i \), \( i=1, N \), qui déterminent la raideur de la spline entre deux noeuds.

Les nouveaux sous-programmes ont initialement 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

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>i</td>
</tr>
<tr>
<td>NOTATION</td>
<td>vi</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2. THE BSMTH ALGORITHM</td>
<td>2</td>
</tr>
<tr>
<td>2.1 Implementation of Smoothing</td>
<td>2</td>
</tr>
<tr>
<td>2.2 Minimization of $G^s$ for given $a$</td>
<td>4</td>
</tr>
<tr>
<td>2.3 Evaluation of $P_{nj}, R_{nj}$, and $v_n$</td>
<td>5</td>
</tr>
<tr>
<td>2.4 Calculation of $X_2$</td>
<td>6</td>
</tr>
<tr>
<td>2.5 The iteration for $p$</td>
<td>7</td>
</tr>
<tr>
<td>2.6 Examples of splines calculated by BSMTH</td>
<td>8</td>
</tr>
<tr>
<td>3. CALCULATION OF INPUT VALUES FOR THE SPLINE $X_2$</td>
<td>9</td>
</tr>
<tr>
<td>4. DEFAULT VALUES FOR THE STIFFNESS WEIGHTS</td>
<td>12</td>
</tr>
<tr>
<td>5. A PARAMETRIC SMOOTHING SPLINE</td>
<td>13</td>
</tr>
<tr>
<td>6. CONCLUDING REMARKS</td>
<td>13</td>
</tr>
</tbody>
</table>

Appendix

<table>
<thead>
<tr>
<th>A. USER'S GUIDES</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1 BSMCRV: User's Guide</td>
<td>14</td>
</tr>
<tr>
<td>B. SUBROUTINE LISTINGS</td>
<td>28</td>
</tr>
<tr>
<td>B.1 BSMCRV</td>
<td>28</td>
</tr>
<tr>
<td>B.2 BSMTH</td>
<td>32</td>
</tr>
<tr>
<td>B.3 NEWWTI</td>
<td>37</td>
</tr>
<tr>
<td>B.4 PARDIF</td>
<td>39</td>
</tr>
<tr>
<td>B.5 PRERR</td>
<td>40</td>
</tr>
<tr>
<td>B.6 SETUPP</td>
<td>43</td>
</tr>
<tr>
<td>B.7 SETUPR</td>
<td>45</td>
</tr>
<tr>
<td>B.8 SMODAV</td>
<td>47</td>
</tr>
</tbody>
</table>
NOTATION

- $B_{n,k}$ - The $n$th B-spline of order $k$ (Section 2.1)
- $D_{n,j}^{(m)}$ - 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)
- $P_{ph}$ - 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)
$y_n^2$ - See equation (2.20)
$y_n$ - Ordinate of the $n^{th}$ data point (Section 2.1)
$y_n^*$ - See equation (2.21)
$\alpha$ - Parameter which balances the relative values of $F$ and $X^2$ (Section 2.1)
$\beta_n$ - The $n^{th}$ spline coefficient (Section 2.1)
$\beta_n^*$ - Approximation to $\beta_n$ used for numerically stable determination of the $X^2$ of a spline (Section 2.4)
$\delta_n$ - The stiffness weight corresponding to the interval between the $(n+k-1)^{th}$ and the $(n+k)^{th}$ knot (Section 2.1)
$\delta_n$ - The Kronecker delta (Section 3)
$\epsilon_n$ - The actual error of the $n^{th}$ data point (Section 3)
$\sigma$ - See equation (3.3)
$X^2$ - The chi-square of the spline (Section 1)
$X^2_{n_{hi}}$ - The chi-square of the spline corresponding to the p-value $p_{hi}$ (Section 2.5)
$X^2_{n_{lo}}$ - 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 \(x^2\) 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, \(\delta_i\), \(i=1,N\), 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 \(\delta_i\) is increased. If the spline is to bend sharply in a different region, the appropriate \(\delta_i\) is
Appendix A

= 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, $S = SMFACT\times NPT\times PRERR^2$. The default value is 1.

Via COMMON /INTEXP/

JDER : The value of JDER used by BSMTH. The integral of the square of the JDER$^{th}$ 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 values 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 NK1 < N + K + max(0,K-2*JDER)

= 3, if IWK < max(NK1,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
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

<table>
<thead>
<tr>
<th>NPT</th>
<th>The number of data points.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>An array of length NPT containing the data point abscissae in ascending order.</td>
</tr>
<tr>
<td>Y</td>
<td>An array of length NPT containing the data point ordinates.</td>
</tr>
<tr>
<td>E</td>
<td>The errors of the data points. The smaller the error the closer the spline will come to that point.</td>
</tr>
<tr>
<td>N</td>
<td>The number of B-splines used to represent the spline.</td>
</tr>
<tr>
<td>K</td>
<td>The order of the spline.</td>
</tr>
<tr>
<td>NKT1</td>
<td>(N + K + \max(0,K-2*JDER)) (see below for a definition of JDER)</td>
</tr>
<tr>
<td>T</td>
<td>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.</td>
</tr>
<tr>
<td>IER</td>
<td>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).</td>
</tr>
</tbody>
</table>
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 parametrization 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 $s_n$ at each data point by

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

(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 performed 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 change the 'stiffness' of the spline at specific locations via the stiffness weights, $\delta_n$, 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, $\delta_n$, 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, $\delta_n$, $n = 1, N-k+1$. Two subroutines are provided which calculate reasonable values for the parameters. The first subroutine, WTIBEG, uses the data points to calculate the $\delta_n$. The second, NEWWTI, uses the spline coefficients of a previously spline approximation of the data to calculate new values for $\delta_n$.

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

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

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

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

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

$$
\int_{t_{n+k-1}}^{t_{n+k}} [f(m)(x)]^2 dx \approx \frac{1}{4} (t_{n+k} - t_{n+k-1}) [(f(m)(t_{n+k}))^2 + f(m)(t_{n+k})f(m)(t_{n+k-1}) + [f(m)(t_{n+k-1})]^2]
$$

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 choices 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.
Section 3

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 \( N \),

\[
\sum_{j=1}^{N} D_{nj}^{(m)} y_j \approx \sum_{j=1}^{N} D_{nj}^{(m)} \epsilon_j \approx \sum_{j=1}^{N} D_{nj}^{(m)} \epsilon_j = \sum_{j=1}^{N} D_{nj}^{(m)} \epsilon_j \sigma^2 \approx g_n^{(m)} \sigma^2 \tag{3.9}
\]

so that an estimate for \( \sigma^2 \) is

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

\( D_{nj}^{(m)} \epsilon_j \) is easily calculated from

\[
D_{nj}^{(m)} \epsilon_j = \sum_{k=1}^{N} D_{mk}^{(m)} \delta_{kj} \epsilon_j \tag{3.11}
\]

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

Thus, in order to estimate \( \sigma^2 \), 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 \( \sum_{j=1}^{N-m} D_{nj}^{(m)} \). If dominated by the errors, these values should be distributed 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.

$$<w_n> = \sigma \text{ for all } n \quad (3.3)$$

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 $x^2$ of the "best" spline is

$$x^2 = \sum_{n=1}^{N_p} \frac{e_n^2}{w_n^2} = \sum_{n=1}^{N_p} w_n^2 \approx Np^2 \quad (3.4)$$

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 $g$, defined iteratively by

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

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

where $\delta_{nj}$ 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 $\xi$ in $(x_n,x_{n+m})$ such that

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

Thus, from equation (3.1) one obtains

$$\sum_{j=1}^{N} D_{nj}^{(m)} y_j = \frac{f^{(m)}(\xi)}{m!} \cdot \sum_{j=1}^{N} D_{nj}^{(m)} \xi_j \quad (3.8)$$
Section 2.6

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

\[
y_n = f(x_n) + \epsilon_n
\]

(3.1)

\( \epsilon_n \) 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 \( \epsilon_n \) are not.

The actual errors \( \epsilon_n \) may be expressed

\[
\epsilon_n = w_n e_n
\]

(3.2)
Section 2.5

2) \( S < X^2 \): \( p_1 \) is too high. Therefore, set \( p_{hi} = 1 \), \( X_{hi}^2 = X^2_1 \) 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 \( X_{lo}^2 = X^2_2 \). \( p_3 \) is now determined such that \( (p_3, S) \) lies on the straight line interpolating \( (p_{lo}, X_{lo}^2) \) and \( (p_{hi}, X_{hi}^2) \).

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

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

1) If \( |S - X_{n}^2| < S/10 \), the iteration terminates.

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

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

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 routine 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 \( x = 16 \). The small crosses below the curve indicate 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
Section 2.4

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

2.5 The iteration for \(p\)

The spline calculated by \text{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 \(a\) in equation (2.7) until \(|X^2 - S| < S/10\). In practice, \text{BSMTH} iterates over \(p\), defined in equation (2.13) rather than \(a\).

As \(a\) increases from 0 to 1, the \(X^2\) of the spline minimizing \(G^x\) increases from some minimum value to some maximum value. However, although the linear system of equation (2.12) is theoretically invertible for any \(a\) 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 \(a\) approaches 0 or 1, there will be numerical difficulties in the inversion of equation (2.12). For this reason, the allowed range of \(a\), and therefore \(p\) is restricted. The upper and lower limits for \(p\) are denoted \(p_{\text{min}}\) and \(p_{\text{max}}\) respectively. \(p_{\text{min}}\) is given the default value of 0.001 and \(p_{\text{max}}\) the default value of 1000. These values have been found adequate to circumvent any numerical difficulties when using \text{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 \(n^\text{th}\) 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 \(p_{\text{max}}\). Again there are two cases:

i) \(S > X^2_2\): \(p_2\) is still too low. However, \(p\) cannot be increased above \(p_{\text{max}}\). 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 \((p_{hi},X^2_{hi})\).
Section 2.4

2.4 Calculation of $\chi^2$

Using equations (2.1) and (2.2), the $\chi^2$ of the spline may be expressed in terms of $Y^2$, $v$, $P_{ij}$ and $\beta_i$:

$$\chi^2 = \sum_{i=1}^{N} \sum_{j=1}^{N} R_{ij} \beta^2_{ij} - 2 \sum_{i=1}^{N} v^2_{i} + Y^2 \quad (2.16)$$

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

$$\chi^2 = \sum_{n=1}^{N} \sum_{j=1}^{N} R_{nj} \gamma^2_{nj} - 2 \sum_{n=1}^{N} v^2_{n} \gamma_{n} + Y^2 \quad (2.17)$$

where

$$\gamma_{n} = \beta_{n} - \beta^*_n \quad (2.18)$$

$$v^2_{n} = \sum_{n=1}^{N} \frac{(y_{n} - y^*_n)B_{nj}(x)}{e^2_{n}} \quad (2.20)$$

$$y^2_{n} = \sum_{n=1}^{N} \frac{(y_{n} - y^*_n)^2}{e^2_{n}} \quad (2.21)$$

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

$$\beta_n = f(t^*_n) \quad (2.22)$$

where

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

(see Reference 1, pp.171). BSMTH chooses $\beta^*_n$ so that $(t^*_n \beta^*_n)$ lies on the piecewise linear curve interpolating the data points, which has breakpoints at the data points.
Section 2.3

2.3 Evaluation of \( P_{nj} \), \( R_{nj} \), and \( v_n \)

The matrix \( P_{nj} \) is evaluated in the subroutine SETUP. Since \( B_{nk}(x) \) is a piecewise polynomial of order \( k \), the integrals in the definition of \( P_{nj} \) can be evaluated by a series of integrations by parts.

\[
P_{nj} = \sum_{p=1}^{N_p} \sum_{q=1}^{k-m} (-1)^{n-1} f_{n,k}(x_p) f_{n+q-1,k}(x_p) \tag{2.14}
\]

If \( k > 2m \), then \( m-q \) will become negative. By convention \( B_{nk}^{(q)}(x) \), for \( q < 0 \), is defined to be the \( q \)th integral of \( B_{nk}(x) \). 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 \((0.9999t_i + 0.0001t_{i+1})\) and \((0.0001t_i + 0.9999t_{i+1})\) for each knot interval.

In practice, \( P_{ij} \) in equation (2.12) is replaced by \( P_{ij}/\Delta \), where \( \Delta \) 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 \( \Delta \) can be absorbed into a redefinition of \( \alpha \). \( \Delta \) is defined by

\[
\Delta = \frac{(t_n - t_1)}{(N-k+1)2^{m-1}} \tag{2.15}
\]

The matrix \( R_{nj} \) is evaluated in the subroutine SETUP making use of the BSPLIN library subroutine BSPLVB to evaluate \( B_{nk}(x_n) \). This subroutine is a modification of the subroutine L2APPR in the BSPLIN library.
2.2 Minimization of $G^*$ for given $\alpha$

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

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

(2.7)

where

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

(2.8)

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

(2.9)

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

(2.10)

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

(2.11)

$G^*$ is minimized with respect to the spline coefficients, $\beta_n$, when

$$\sum_{j=1}^{N} (R_{nj} + P_{nj}) \beta_j = \nu_n$$

(2.12)

where

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

(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).
Section 2.1

\(X^2\) 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

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

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

\[
G = \alpha X^2 + (1-\alpha)F
\]

for given \(\alpha\). Since \(G\) is quadratic in the spline coefficients \(\beta_n\), this amounts to the solution of a linear system. An iteration is then done to find the value of \(\alpha\) 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

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

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 \(t_n, n=1,N\).

The coefficients \(\delta_n\) 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 \(\delta_n\) will be large. In regions where the spline is expected to have high curvature, the \(\delta_n\) will be small.

The required spline is found by minimizing the functional

\[
G^* = \alpha X^2 + (1-\alpha)F^*
\]
decreased. Two subroutines, WTBEG and WTNIEW (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 and later considered by Schoenberg, Reinsch and de Boor. The idea is to define two functionals dependent quadratically on the spline coefficients for which one is solving. One functional is the $X^2$ of the spline curve,

$$X^2 = \sum_{n=1}^{N_p} \left[ \frac{y_n - f(x_n)}{e_n} \right]^2$$  \hspace{1cm} (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

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

The functions $f_n(x)$ are the basis functions for the linear space of spline functions. The
= 5, if 0.9\times x^2 of the spline of the data point abscissae is greater than the value predicted by PRERR.

= 6, if 1.1\times 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 = \text{max}(NKT1,K*2)

WK: An array of length 4*IWK
Appendix A

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

<table>
<thead>
<tr>
<th>J</th>
<th>X(J)</th>
<th>Y(J)</th>
<th>E(J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.24</td>
<td>-0.09</td>
<td>0.3</td>
</tr>
<tr>
<td>2</td>
<td>0.86</td>
<td>0.33</td>
<td>0.3</td>
</tr>
<tr>
<td>3</td>
<td>0.28</td>
<td>0.18</td>
<td>0.3</td>
</tr>
<tr>
<td>4</td>
<td>0.31</td>
<td>0.35</td>
<td>0.3</td>
</tr>
<tr>
<td>5</td>
<td>0.43</td>
<td>0.47</td>
<td>0.3</td>
</tr>
<tr>
<td>6</td>
<td>0.45</td>
<td>0.63</td>
<td>0.3</td>
</tr>
<tr>
<td>7</td>
<td>0.39</td>
<td>0.77</td>
<td>0.3</td>
</tr>
<tr>
<td>8</td>
<td>0.38</td>
<td>0.86</td>
<td>0.3</td>
</tr>
<tr>
<td>9</td>
<td>0.28</td>
<td>0.94</td>
<td>0.3</td>
</tr>
<tr>
<td>10</td>
<td>0.11</td>
<td>0.97</td>
<td>0.3</td>
</tr>
<tr>
<td>11</td>
<td>0.06</td>
<td>1.03</td>
<td>0.3</td>
</tr>
<tr>
<td>12</td>
<td>-0.08</td>
<td>1.02</td>
<td>0.3</td>
</tr>
<tr>
<td>13</td>
<td>-0.17</td>
<td>1.00</td>
<td>0.3</td>
</tr>
<tr>
<td>14</td>
<td>-0.23</td>
<td>0.97</td>
<td>0.3</td>
</tr>
<tr>
<td>15</td>
<td>-0.31</td>
<td>0.91</td>
<td>0.3</td>
</tr>
<tr>
<td>16</td>
<td>-0.29</td>
<td>0.80</td>
<td>0.3</td>
</tr>
<tr>
<td>17</td>
<td>-0.33</td>
<td>0.70</td>
<td>0.3</td>
</tr>
<tr>
<td>18</td>
<td>-0.38</td>
<td>0.59</td>
<td>0.3</td>
</tr>
<tr>
<td>19</td>
<td>-0.15</td>
<td>0.45</td>
<td>0.3</td>
</tr>
<tr>
<td>20</td>
<td>0.86</td>
<td>0.33</td>
<td>0.3</td>
</tr>
<tr>
<td>21</td>
<td>0.26</td>
<td>0.11</td>
<td>0.3</td>
</tr>
<tr>
<td>22</td>
<td>0.66</td>
<td>0.03</td>
<td>0.3</td>
</tr>
</tbody>
</table>

The spline coefficients and the fractional arc length values returned by BSMCRV are
The values returned via COMMON / CHISQ / are

\[ \text{XSQX} = 0.76300E-01, \text{XSQY} = 0.11647, \text{SX} = 0.71920E-01, \text{SY} = 0.11766 \]
Appendix A

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(l),l=1,NKT which has chi-square of S with respect to the data points X(l),Y(l),l=1,NPT, and which has as small a JDER^th 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 JDER^th 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 JDER^th 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*K)

= 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 < 0.9*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*K)

WK : An array of length 4*IWK
Appendix A

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

\( S = 10.0, \ JOER = 2, \ NPT = 10, \ K = 4, \ N = 10, \ NKT = 14 \)

<table>
<thead>
<tr>
<th>( J )</th>
<th>( X(J) )</th>
<th>( Y(J) )</th>
<th>( E(J) )</th>
<th>( T(J) )</th>
<th>( WTI(J) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.0</td>
<td>0.0</td>
<td>0.005</td>
<td>7.0</td>
<td>0.51183E-02</td>
</tr>
<tr>
<td>2</td>
<td>8.0</td>
<td>13.5</td>
<td>0.5</td>
<td>7.0</td>
<td>0.55789</td>
</tr>
<tr>
<td>3</td>
<td>9.0</td>
<td>15.5</td>
<td>0.5</td>
<td>7.0</td>
<td>1.1778</td>
</tr>
<tr>
<td>4</td>
<td>10.0</td>
<td>14.5</td>
<td>0.5</td>
<td>7.0</td>
<td>1.1778</td>
</tr>
<tr>
<td>5</td>
<td>11.0</td>
<td>15.5</td>
<td>0.5</td>
<td>9.0</td>
<td>2.6500</td>
</tr>
<tr>
<td>6</td>
<td>12.0</td>
<td>15.0</td>
<td>0.5</td>
<td>10.0</td>
<td>0.88333</td>
</tr>
<tr>
<td>7</td>
<td>13.0</td>
<td>14.5</td>
<td>0.5</td>
<td>11.0</td>
<td>0.15407E-01</td>
</tr>
<tr>
<td>8</td>
<td>14.0</td>
<td>15.0</td>
<td>0.5</td>
<td>12.0</td>
<td>16.1</td>
</tr>
<tr>
<td>9</td>
<td>15.0</td>
<td>13.5</td>
<td>0.5</td>
<td>13.0</td>
<td>16.1</td>
</tr>
<tr>
<td>10</td>
<td>15.0</td>
<td>0.0</td>
<td>0.005</td>
<td>14.0</td>
<td>16.1</td>
</tr>
</tbody>
</table>

The spline coefficients obtained were

<table>
<thead>
<tr>
<th>( J )</th>
<th>( BCDEF(J) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.3195889E-04</td>
</tr>
<tr>
<td>2</td>
<td>14.9338</td>
</tr>
<tr>
<td>3</td>
<td>14.9338</td>
</tr>
<tr>
<td>4</td>
<td>15.00996</td>
</tr>
<tr>
<td>5</td>
<td>15.11922</td>
</tr>
<tr>
<td>6</td>
<td>15.11922</td>
</tr>
<tr>
<td>7</td>
<td>15.11922</td>
</tr>
<tr>
<td>8</td>
<td>14.9338</td>
</tr>
<tr>
<td>9</td>
<td>13.10889</td>
</tr>
<tr>
<td>10</td>
<td>-2.075629</td>
</tr>
</tbody>
</table>

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 $\delta_i$ 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, $\delta_i$. 

Appendix A

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

\[ NOLD = 4 , \ \text{NKOLD} = 14 , \ \text{NKTNEW} = 14 , \ \text{NWTI} = 7 , \ \text{JDER} = 2 \]

<table>
<thead>
<tr>
<th></th>
<th>TOLD(J)</th>
<th>TNEW(J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.0</td>
<td>7.0</td>
</tr>
<tr>
<td>2</td>
<td>7.0</td>
<td>7.0</td>
</tr>
<tr>
<td>3</td>
<td>7.0</td>
<td>7.0</td>
</tr>
<tr>
<td>4</td>
<td>9.0</td>
<td>9.0</td>
</tr>
<tr>
<td>5</td>
<td>10.0</td>
<td>10.0</td>
</tr>
<tr>
<td>6</td>
<td>11.0</td>
<td>11.0</td>
</tr>
<tr>
<td>7</td>
<td>12.0</td>
<td>12.0</td>
</tr>
<tr>
<td>8</td>
<td>13.0</td>
<td>13.0</td>
</tr>
<tr>
<td>9</td>
<td>14.0</td>
<td>14.0</td>
</tr>
<tr>
<td>10</td>
<td>15.1</td>
<td>16.1</td>
</tr>
<tr>
<td>11</td>
<td>16.1</td>
<td>16.1</td>
</tr>
<tr>
<td>12</td>
<td>16.1</td>
<td>16.1</td>
</tr>
<tr>
<td>13</td>
<td>16.1</td>
<td>16.1</td>
</tr>
<tr>
<td>14</td>
<td>16.1</td>
<td>16.1</td>
</tr>
</tbody>
</table>

The stiffness weights obtained were

<table>
<thead>
<tr>
<th></th>
<th>WTI(J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.10E-02</td>
</tr>
<tr>
<td>2</td>
<td>8.56E-01</td>
</tr>
<tr>
<td>3</td>
<td>1.15E-01</td>
</tr>
<tr>
<td>4</td>
<td>8.5E-01</td>
</tr>
<tr>
<td>5</td>
<td>4.8E-05</td>
</tr>
<tr>
<td>6</td>
<td>8.15E-01</td>
</tr>
<tr>
<td>7</td>
<td>8.10E-02</td>
</tr>
</tbody>
</table>
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: \( S = NPT \times \text{PRERR}^2 \).

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 \( 2 \times \text{IMAX} \). 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
Appendix A

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 = 48, IMAX = 5

<table>
<thead>
<tr>
<th>J</th>
<th>X(J)</th>
<th>Y(J)</th>
<th>J</th>
<th>X(J)</th>
<th>Y(J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.00</td>
<td>1507.89</td>
<td>21</td>
<td>22.70</td>
<td>1482.57</td>
</tr>
<tr>
<td>2</td>
<td>3.63</td>
<td>1507.85</td>
<td>22</td>
<td>23.00</td>
<td>1481.83</td>
</tr>
<tr>
<td>3</td>
<td>7.26</td>
<td>1507.81</td>
<td>23</td>
<td>23.50</td>
<td>1480.34</td>
</tr>
<tr>
<td>4</td>
<td>10.90</td>
<td>1507.77</td>
<td>24</td>
<td>24.20</td>
<td>1478.83</td>
</tr>
<tr>
<td>5</td>
<td>12.20</td>
<td>1507.17</td>
<td>25</td>
<td>26.10</td>
<td>1476.17</td>
</tr>
<tr>
<td>6</td>
<td>13.70</td>
<td>1505.82</td>
<td>26</td>
<td>27.00</td>
<td>1475.02</td>
</tr>
<tr>
<td>7</td>
<td>14.10</td>
<td>1502.49</td>
<td>27</td>
<td>28.10</td>
<td>1472.68</td>
</tr>
<tr>
<td>8</td>
<td>14.50</td>
<td>1501.53</td>
<td>28</td>
<td>29.80</td>
<td>1470.30</td>
</tr>
<tr>
<td>9</td>
<td>14.80</td>
<td>1499.58</td>
<td>29</td>
<td>29.90</td>
<td>1468.70</td>
</tr>
<tr>
<td>10</td>
<td>15.20</td>
<td>1498.27</td>
<td>30</td>
<td>30.60</td>
<td>1467.92</td>
</tr>
<tr>
<td>11</td>
<td>15.38</td>
<td>1496.94</td>
<td>31</td>
<td>44.00</td>
<td>1463.25</td>
</tr>
<tr>
<td>12</td>
<td>15.40</td>
<td>1496.93</td>
<td>32</td>
<td>52.70</td>
<td>1464.54</td>
</tr>
<tr>
<td>13</td>
<td>15.70</td>
<td>1494.92</td>
<td>33</td>
<td>58.40</td>
<td>1466.09</td>
</tr>
<tr>
<td>14</td>
<td>16.60</td>
<td>1492.87</td>
<td>34</td>
<td>65.20</td>
<td>1465.74</td>
</tr>
<tr>
<td>15</td>
<td>16.80</td>
<td>1491.84</td>
<td>35</td>
<td>74.50</td>
<td>1475.43</td>
</tr>
<tr>
<td>16</td>
<td>17.30</td>
<td>1490.89</td>
<td>36</td>
<td>80.30</td>
<td>1478.08</td>
</tr>
<tr>
<td>17</td>
<td>18.40</td>
<td>1488.33</td>
<td>37</td>
<td>94.50</td>
<td>1482.65</td>
</tr>
<tr>
<td>18</td>
<td>28.90</td>
<td>1486.22</td>
<td>38</td>
<td>110.00</td>
<td>1487.18</td>
</tr>
<tr>
<td>19</td>
<td>21.80</td>
<td>1484.77</td>
<td>39</td>
<td>119.10</td>
<td>1489.40</td>
</tr>
<tr>
<td>20</td>
<td>22.50</td>
<td>1483.31</td>
<td>40</td>
<td>158.10</td>
<td>1491.40</td>
</tr>
</tbody>
</table>

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

\[
\begin{array}{ccc}
NPT & = & 10 \\
NKT & = & 14 \\
NWTI & = & 7
\end{array}
\]

\[
\begin{array}{cccc}
J & X(J) & Y(J) & T(J) \\
1 & 7.0 & 0.0 & 7.0 \\
2 & 8.0 & 13.5 & 7.0 \\
3 & 9.0 & 15.5 & 7.0 \\
4 & 10.0 & 14.5 & 7.0 \\
5 & 11.0 & 15.5 & 9.0 \\
6 & 12.0 & 15.0 & 10.0 \\
7 & 13.0 & 14.5 & 11.0 \\
8 & 14.0 & 15.0 & 12.0 \\
9 & 15.0 & 13.5 & 13.0 \\
10 & 16.0 & 0.0 & 14.0 \\
11 & & & 16.1 \\
12 & & & 16.1 \\
13 & & & 16.1 \\
14 & & & 16.1 \\
\end{array}
\]

The stiffness weights obtained were

\[
\begin{array}{c}
J \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array}
\begin{array}{c}
WTI(J) \\
0.51183E-02 \\
0.55789 \\
1.1778 \\
1.1778 \\
2.6588 \\
0.88333 \\
0.15407E-01 \\
\end{array}
\]
SUBROUTINE BSMCRV

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

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 fractional arc-length along the
spline. An approximation for the arc length at each point is
obtained from the distances between the points. BSMTH is used to
spline the abscissae and the ordinates. PRERR is used to
determine a smoothing factor for the splines and WTBEG is used
to determine stiffness weights.

AUTHOR: David Hally, May 1981

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

CALLS PRERR,BSMTH,WTBEG

INPUT:

VIA SUBROUTINE ARGUMENTS:

NPT : The no. of data points
X : An array of length NPT containing the data point
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.
Appendix B

N : The no. of B-splines.
K : The order of the spline.
NKT1 = N+K+\text{max}(0,K-2,JDER)
T : An array of length NKT1 the first N+K elements of which contain the knot sequence. The variable used to parametrize the curve is the arc length divided by the total length of the curve. Thus the knots must span the interval [0,1]. Default gives a uniform distribution of knots over this interval.

IER : 0 : If defaults are desired
1 : If defaults are not desired

COMMON /NODFLT/

IMAX : 2*IMAX is the max. no. of divided differences allowed to find the error (used in function PRERR). Default value is 5
SMFACT : See comments below

COMMON /INTEXP/

JDER : The integral of the square of the JDER-th derivative of the spline is minimized (subject to the constraint that XSG=S). Default value is 2. JDER must not exceed K-1

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)

OUTPUT:
IER : 0 : Iteration converged
1 : If JDER > K-1
2 : If NKT1 < N+K+\text{max}(0,K-2,JDER)
3 : If IWK < \text{max}(NKT1,K+2)
4 : If iteration for P1 in BSMTH did not converge during the spline of the X-values
5 : If the chi-square of the spline of the X-values returned by BSMTH > 1.1*S (i.e. PMAX in BSMTH is too small)
6 : If the chi-square of the spline of the X-values returned by BSMTH < .9*S (i.e. PMIN in BSMTH is too large)
7,8,9 As for IER=4,5,6, respectively, but for the spline of the Y-values
BCOEFX : Array of length N containing the B-spline coefs. for the X-values of the curve
SUBROUTINE SETUPP(NPT,E,JD,ENK,NK,WTI,PA,PA1,EB1,EB11,A)
C-------------------------------------------------------------------
C
C SETUPP calculates the matrix P (see Ref. Manual)
C
CAUTHOR: David Hally, May. 1981
C
CCALLED by BSMTH
C
CCALLS SMODAV
C
C from BSPLIN library BVALUE,BSPLVD
C-------------------------------------------------------------------

REAL T(NK),P(K,N),A(K,K),EB(K,K),EB1(K,K),EB11(NK),EB111(NK),EB1111(NK),
*  E(NPT),A1(KK),T1,T2,H,H1
INTEGER NK,JD,ENK,NN,MM,MMAX,II,IIJ,IIJ1,IIK,IIK1

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)

DO 20 I=IK
20  T(J)=T(N+K)+T(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,WT1))

C An iteration over the intervals between knots is begun.

DO 18 I=K,N
18  IF(T(I)=T(I+1))GO TO 180

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,PA1,EB11,K)
CALL BSPLVD(T,K,T2,I,PA1,EB11,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/2.0) GO TO 130
IF (J.GE.IMAX-2) GO TO 120
GO TO 70
120 SDEV=(D/2.-NSGNCH)/SQRT(D)

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

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

RETURN
END
REAL X(N), Y(N), E(N), WK(N), G(N, IMAX), DEV, 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)=S8

DO 20 J=1,N
   WK(J)=0.0
   WK(1)=E(I)
   DO 20 J=1,IMAX
   20 G(I,J)=WK(J)

IMAX-MINO(N/2, IMAX)
SDEV=8.8
PRERR=0.8
IF(IFLAG.EQ.1)GO TO 50
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.

DO 68 I=1,N
   WK(I)=Y(I)
   J=0
   J2=0
C The no. of sign changes in the divided differences is determined.

NSGNCH=0
I=J+1
DO 88 K=I+1,N-J
   IF(WK(I).LE.WK(K))100,98,118
98 CONTINUE
100 NSGNCH=NSGNCH+1
110 I=K
   IF(I.LT.N-J)GO TO 88
   D=N-J2-1

Appendix B
B.5 PRERR

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

This subroutine calculates the mean error in the data points (X,Y) by taking divided differences until the no. of sign changes in the I-th divided difference is that expected from random data. The error is then determined by assuming that the contribution from the smooth curve underlying the data is negligible.

AUTHOR: David Hall, Jan. 1981

USAGE:
EXECUTE mainpgm.BLPLIN:HELLYP/LIB

CALLS PARDIF

INPUT:

N   = No. of data points
X   = An array of length N containing the data point abscissae in ascending order.
Y   = An array of length N containing the data point ordinates.
E   = An array of length N containing the relative errors of the data points. The absolute errors are obtained by multiplying the returned value of PRERR by the relative errors.
IFLAG = 0, If calculation is to be done from scratch
       = 1, If IMAX,X,E, and G have the same value as in the previous call
G   = Array of dimensions N,IMAX. G(I,J) is the expectation value of the J-th divided difference given an error of E(I) in the I-th data point. If IER=0 G is calculated; otherwise it is assumed known.
IMAX = 2*IMAX is the max. no. of divided differences allowed. IMAX = 5 is suggested

OUTPUT:

PRERR = The calculated mean error in the data

VIA COMMON / CERR /

SDEV : The no. of sign changes in the divided difference used to calculate PRERR is greater than that expected for random data less SDEV standard deviations

WORK SPACE:

WK(1) OF DIMENSION N
SUBROUTINE PARDIF(N,X,F,J,IMIN,IMAX)

C------------------------------------------------------------
C
C PARDIF calculates the divided difference of the data points
C (X(I),F(I)), I=IMIN,IMAX. To avoid over- or underflows the X
C intervals are normalized by the factor H=(X(N)-X(1))/N. This is
C of no consequence in PRERR since only ratios of partial diff-
C erences are of significance.
C
C AUTHOR: David Hally, May. 1981
C
C CALLED by PRERR
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
      F(I)=H*(F(I+1)-F(I))/(X(I+IT)-X(I-J))
10   DO 20 I=IMAX-2,IMIN,-1
      F(I+1)=F(I)
20   F(IMIN)=8.8
     F(IMAX)=8.8
RETURN
END
B2=BVALUE(TOLD,BCDEF,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.WMIN) WTI(IW) = 1.0E-03
     20 IF(DUMMY.LE.WMAX) WTI(IW) = 1.0E+03
    RETURN

30 DO 40 IW=1,NWTI
   WTI(IW)=1.0
40 RETURN
END
B.3 NEWWTI

SUBROUTINE NEWWTI (NOLD, BCOEF, NKOLD, TOLD, NKNEW, TNEW, WTI, JOER)

C NEWWTI uses the previous spline fit to predict values for the
integral weights WTI for use in BSMTH.

AUTHOR: David Hally, Aug. 1981

USAGE:
EXECUTE mainpgm,BSPLIN:HLLYSPLIB

CALLS SMODAV
from BSPLIN library : BVALUE

INPUT:
NOLD = No. of B-splines for old spline fit
BCOEF = Array of length N containing the B-spline coefficients
        for the old spline fit
NKOLD = No. of knots for the old spline fit
TOLD = Array of length NK containing the knots for the old
        spline fit
NKNEW = No. of knots for the new spline fit
TNEW = Array of length NK containing the knots for the new
        spline fit
WTI = No. of integral weights for the new spline fit
JOER = The order of derivative minimized by BSMTH

OUTPUT:
WTI = Array of length NWTI containing the integral weights

REAL BCOEF (NOLD), TOLD (NKOLD), TNEW (NKNEW), WTI (NWTI),
* B1, B2, T1, T2, WM, WMAX, XIAV, DUMMY
INTEGER NOLD, NKOLD, NKNEW, KOLD, KNEW, NWTI, JOER, I, IW

KOLD = NKOLD - NOLD
KNEW = (NKNEW - NWTI + 1) / 2
IW = 0

C On each knot interval the integral of the square of the JOER-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 = .8881 * TNEW (I + KNEW - 1) + .1118 * TNEW (I + KNEW)
  B1 = BVALUE (TOLD, BCOEF, NOLD, KOLD, T1, JOER)
  B2 = BVALUE (TOLD, BCOEF, NOLD, KOLD, T2, JOER)
  WM = WTI (I) * B1 * B2
  WMAX = MAX (WM, WMAX)
  XIAV = XIAV + WM
  DUMMY = DUMMY
10 CONTINUE
$\Phi_1 = P_1$
$P_1 = 1 \times P_{LO} + .9 \times \Phi_1$
GO TO 220

C Similarly, if $P_1$ is very close to $P_{LO}$, it is possible that $XS_Q < XS_{LO}$
C In this case $P_{LO}$ is set to $P_1$, $XS_{LO}$ to $XS_Q$ and $P_1$ to $.9 \times P_1, +1\times \Phi_1$

178 IF $(XS_Q,GT,XS_{LO})$ GO TO 188
$XS_{LO} = XS_Q$
$P_{LO} = P_1$
$P_1 = .9 \times P_{LO} + .1 \times \Phi_1$
GO TO 220

188 IF ((S-XS_{LO}) LT. (XS_{HI}-S)) GO TO 198
$P_2 = (P_1 - \Phi_1) \times (S-XS_{HI}) / (XS_{HI}-XS_Q) + \Phi_1$
GO TO 200

198 $P_2 = (P_1 - P_{LO}) \times (S-XS_{LO}) / (XS_{LO}-XS_{LO}) + P_{LO}$

200 IF ($P_2, LT.P_1$) GO TO 210
$P_{LO} = P_1$
$XS_{LO} = XS_Q$
$P_1 = P_2$
IF $(P_1, GT.PHI) P_1 = (P_{LO} + PHI) / 2.$
GO TO 220

210 $PH_1 = P_1$
$XS_{HI} = XS_Q$
$P_1 = P_2$
IF $(P_1, LT.PLO) P_1 = (P_{LO} + PHI) / 2.$

220 CONTINUE$\quad IER=4$

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

230 DO 240 I=1,N
240 $BCOEF(I) = BCOEF(I) + 4K(I,1)$
RETURN
END
Appendix B

90  BCOEF(I)=BCOEF(I)-WK(I,1)
     XSQ=XSQ(NK,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 XSQL0 or XSQHI.
C If XSQL0 < S or XSQL0 > 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
     XSQL0=XSQ
     PLO=P1
     P1=PMAX
     GO TO 220
120  XSQLHI=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  XSQLHI=XSQ
     PHI=P1
     GO TO 150
140  IF(S.GE.XSQ)GO TO 145
     IER=6
     GO TO 230
145  XSQL0=XSQ
     PLO=P1
150  P1=(P1-PLO)*(S-XSQL0)/(XSQHI-XSQL0)+PLO
     GO TO 220

C It is possible that due to numerical inaccuracy in the evaluation
C of XSQ, that XSQL0<XSQHI. This would normally only occur if P1 is
C very close to PHI. Hence PHI is set to P1, XSQLHI to XSQ and
C P1 to .1ePLO+.3ePHI!
160  IF(XSQL.LT.XSQLHI)GO TO 170
     XSQLHI=XSQ

50
IER=1
RETURN
10 IF(NKT1.GE.NKT+MAX0(0,K-2*$JOER))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 SETUPP and SETUPR respectively.
30 IF(IER.EQ.0)GO TO 40
CALL SETUPP(NPT,E,JOER,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 50 I=1,N-1
  DYSQ=0.
  DO 50 J=1,K-1
    DYSQ=DYSQ+T(I,J)
  END
  DYSQ=DYSQ/FLOAT(K-1)
  CALL INTERV(X,NPT,DYSQ,LEFT,IFLAG)
  IF(MFLAG.EQ.1)LEFT=NPT-1
  DYSQ=(DYSQ-X(LEFT))/X(LEFT+1)-X(LEFT))
50 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 1.5* of S
P1=1.
XSQLO=0.0
XSQH1=0.0
DO 220 I=1,30
  DO 70 J=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*BCEF-VCT is solved by first finding the
C Cholesky factorization of R, then by solving for BCEF.
CALL BCHFAC(R,K,N,WK(1,4))
DO 80 I=1,N
80 BCEF(I)=WK(I,2)
CALL BCHSLV(R,K,N,BCEF)

C The chi-square of the solution is determined.
DO 90 I=1,N
over this region. These weights are relative, i.e., changing all the weights by a constant factor will not affect the resulting spline.

IER = 0, if JoER = 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).

IER = 1, if P is to be recalculated.

VIA COMMON / PLIMS /:

PMIN = Min. allowed value of P1 (See comment describing iteration for correct chi-square). Default is 1.E-03.

PMAX = Max. allowed value of P1. Default is 1.E+03.

OUTPUT:

IER = 0, Calculation has been successful.

IER = 1, if JoER > K-1.

IER = 2, if NKT1 < N*K+max(0,K-2*JoER).

IER = 3, if IWK < max(NKT1,K**2).

IER = 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.

IER = 5, if the chi-square of the spline > 1.1*5.

IER = 6, if the chi-square of the spline < .5*5.

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

VIA COMMON / CHISO /:

XSO = the chi-square 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.

-------------------------------------------------------------------

REAL BCOEF(N),T(NKT1),WTI(N),R(K,N,3),WK(IWK,4),
* X(NPT),Y(NPT),E(NPT),
* PL,P2,PHI,PLD,XSO,XSOHI,XSOLO,YSQ,ALF,DYSQ,S
INTEGER N,K,NKT,NK,NK1,JoER,NPT,NFLAG,LEFT,LEFTIWK,IER,IT,I,J

COMMON / CHISO / XSO,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(JOER.LT.K)GO TO 10

NKT=W*K
B.2 BSMTH

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

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 JDER-th derivative
as possible.

AUTHOR: David Hally, May 1981

USAGE:
EXECUTE mainpgm,BSPRIN,ZLYSP/LIB,BSPLIN:BSPLIN/LIB

CALLS SETUPQ,SETUPR,XSQC
from BSPLIN library: INTERV,BCHFAC,BCHSLY

INPUT:

S : The chi-square of the spline with respect to the data
will be within 15% 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 JDER-th derivative
of the spline is minimized (subject to the con-
straint 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 no. 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 no. of B-splines.
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
SETUP.

WTI : An array of length N of which only the first N-K+1
elements are used (rather than passing in an other-
wise superfluous argument). WTI(I) is a weight
for the integral of the square of the JDER-th deriv-
ative of the spline between T(I+K-1) and T(I+K). The
larger WTI(I) the smaller the integral will be.
The error in the X-values are found by calling the function PRERR and the integral weights, WTI, by calling WTIBEG. They are splined using BSMTH using fractional arc length to parametrize the data points. Similarly for the Y-values.

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

default value for SMFACT is 1.0.

158   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,1WK,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,1WK,WK,IER)
   RETURN
END
BCOEFY = Array of length N containing the B-spline coefs. for the Y-values of the curve
ARCL(I) = Arc length at the I-th data point/total length of curve
VIA COMMON /CHISQ /
XSQX = Chi-square of the spline of the abscissae
XSQY = Chi-square of the spline of the ordinates
SX = Required Chi-square of the abscissae (as determined by PRERR)
SY = Required Chi-square of the ordinates (as determined by PRERR)
VIA COMMON /CRVLTH /
SNEWL = The total arc length of the curve
WORK SPACE :
R : An array of length 3*K*N
IWK = max(NKT1,K1,K1*2)
W : An array of length 4*IWK
G : An array of dimensions NPT,IMAX (used by PRERR)
WTI : Array of length N used for the integral weights for BSMTH

-------------------------------------------------------------------
REAL X(NPT),Y(NPT),E(NPT),ARCL(NPT),BCOEFY(N),BCOEFX(N),
* T(NKT1),WTI(N),G(NPT,IMAX),W(K,K,N,3),
* SMFACT
INTEGER NPT,NK1,NKT1,NWT1,NK1,IER,IMAX,K1,IW,ID
COMMON /NODFLT/ SMFACT,IMAX
COMMON /INTEXP/ JOER
COMMON /CHISQ/ XSQY,XSQX,SY,SX
NWT1=N-K+1
C ARCL(I),I=1,N is initialized by connecting the data points with 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 taken from the COMMON block /NODFLT/ and JOER from COMMON /INTEXP/
IF (IER,NE.0) GO TO 150
SMFACT=1.0
The elements of $P$ are determined by integration by parts.

```
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+J
   WK(IKJ)=1.
   IF(J.EQ.K)GO TO 58
   DO 48 L=IKJ+1,I+1
     WK(L)=0.0
     IF(J.EQ.K)GO TO 60
     DO 48 L=IKJ+1,I+1
       WK(L)=WK(L)+WK(L)*(T(L,KM)-T(L))/FLOAT(KM)
       IF(L.LE.1)WK(L)=WK(L)+WK(L-1)
     CONTINUE
   CONTINUE
A1(J,M)=BVALUE(T,WK,N,K+M,T2,0)
A(J,M)=BVALUE(T,WK,N,K+M,T1,0)
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
       IKJ=I+J
       HI=H
       IF(MMAX.LT.1)GO TO 110
       DO 100 M=1,MMAX
          P(LJ1,IKJ)=P(LJ1,IKJ)+HI*WT(I1)*(DB1(L,JDER-M+1)*
            DB1(L,JDER+M)*DB1(L,JDER-M+1)*DB1(L,JDER+M)+
            MMAX*HI-A(L,M)*DB1(J,JDER+MMAX+M))
          HI=HI
        CONTINUE
    CONTINUE
110 IF(K.LE.2*JDER)GO TO 130
100 DO 120 M=1,K-2*JDER
    P(LJ1,IKJ)=P(LJ1,IKJ)+HI*WT(I1)*(A1(L,M)*DB1(J,JDER+
      MMAX+M)-A(L,M)*DB1(J,JDER+MMAX+M))
120 HI=HI
130 CONTINUE
140 CONTINUE
RETURN
END
```
SUBROUTINE SETUPR (NKT, T, K, NPT, X, Y, E, YSQ, Y1, VCT, VCT1)
C-------------------------------------------------------------
C
C The matrix R, the arrays Y1, VCT and VCT1, and the number YSQ are calculated
C (SETUPR is based closely on L2APPR by Carl de Boor, in
A Practical Guide to Splines, p. 255)
C
AUTHOR: David Hally, May, 1981
C
CALLED BY BSMT
C CALLS from Bsplin library BSPLVB
C
REAL T(NKT), R(K, N), VCT(N), BIATX(20), X(NPT), Y(NPT), E(NPT), DW,
* 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 30 LL=1, NPT
   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 38
30 CONTINUE

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
      MM=MM+1
      DYSQ=DYSQ-BIATX(MM)*Y1(LEFT-K+MM)
   50 CONTINUE
   DO 70 MM=1, K
      MM=MM+1
      MM=MM+1
      DYSQ=DYSQ-BIATX(MM)*Y1(LEFT-K+MM)
      DYSQ=DYSQ-DYSQ
   70 CONTINUE
   VCT1(J)=VCT1(J)+DYSQ*OW
   VCT(J)=VCT(J)+DYSQ*OW
i=1

DO 60 JJ=MM,K
   R(I,J) = B1ATX(JJ) * D4 + R(I,J)
   I = I + 1
60 CONTINUE

CONTINUE
70 CONTINUE
   YSQ = (DYSQ/E(LL))**2 + YSQ
80 CONTINUE
RETURN
END
B.8 SMODAV

FUNCTION SMODAV(NPT,X)
C-----------------------------------------------------------------
C
C
C SMODAV returns a modal average of the numbers in X
C
C AUTHOR : David Hally, Aug. 1981
C
C USAGE : EXECUTE main-pgm,BSPLIN:HLLYSP/LIB
C
C INPUT :
C
NPT  = No. of values to be averaged
C X    = Array of length NPT containing values to be averaged
C
C RETURNS:
C
SMODAV = Modal average of the values in X
C-----------------------------------------------------------------
REAL X(NPT),XBOX(11),SUMBOX(10),XMIN,XMAX,XRATIO,SCALE,SMODAV
INTEGER IBOX(181,NPT,ISUM,NBGX,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=MAX1(X(I),XMAX)
  XMIN=MIN1(X(I),XMIN)
10 CONTINUE

IF(XMIN.EQ.1.0E+30)GO TO 910
XRATIO=XMAX/XMIN
NBOX=ALOG10(XRATIO)/ALOG10(FLOAT(NPT))
NBOX=MAX8(NBOX,18,NPT/5)
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
  XBOX(I)=XBOX(I-1)*SCALE
DO 60 I=1,NPT
   DO 40 J=2,NBOX+1
      IF(X(I)).LE.XBOX(J) GO TO 50
   40 SUMBOX(J)=SUMBOX(J-1)+X(I)
   50 IBOX(J-I)=IBOX(J-I)+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(1)/IBOX(1)
   RETURN
   90 SMODAV=0.0
   RETURN
END
Appendix B

B.9 WTIBEG

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

C-------------------------------------------------------------
C WTIBEG uses the data points to predict values for the integral
C weights WTI for use in BSMT.
C
AUTHOR: David Hally, Aug. 1981
C
USAGE:
EXECUTE mainpgm,BSPLIN:HLLYSP/LIB
C
INPUT:
C NPT = No. of data points
C X = Array of length NPT containing data point abscissae
C Y = Array of length NPT containing data point ordinates
C NKT = No. of knots
C T = Array of length NKT containing the knots
C NWTI = No. of integral weights (= no. of B-splines - order of spline +1)
C
OUTPUT:
C WTI = Array of length NWTI containing the integral weights
C-------------------------------------------------------------

REAL X(NPT),Y(NPT),T(NKT),WTI(NWTI),
  * HL,HR,TL,TR,D1YL,D1YR,D2YDL,D2YOR,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))
D2YOR=D2YDL
D2YTL=D2YDL
SLOPE=0.0
The next interval of interest is the interval from the right end of the current interval to the next data point or the next knot, whichever occurs first. The contribution to \( Wi \) from this interval is determined.

\[
\begin{align*}
10 & \quad TR = \text{MIN}(T(I+1), X(ID+1)) \\
& \quad D2YTR = D2YDL + \text{SLOPE} \times (TR - X(ID)) \\
& \quad \text{WTI} (I1) = \text{WTI} (I1) + \text{HL} \times (D2YDL + D2YTR + D2YTR) / 3. \\
& \quad \text{IF}(I1 \text{EQ} NWT1 + K - 1) \text{GO TO 50} \\
& \quad T1 = TR \\
& \quad D2YDL = D2YTR \\
& \quad \text{IF}(TR \text{NE} X(ID+1)) \text{GO TO 30} \\
& \quad ID = ID + 1 \\
& \quad D2YDL = D2YDR \\
& \quad D1YL = D1YR \\
& \quad \text{HL = HR} \\
& \quad \text{IF}(ID, GT, NPT - 2) \text{GO TO 20} \\
& \quad HR = X(ID + 2) - X(ID + 1) \\
& \quad D1YR = (Y(ID + 2) - Y(ID + 1)) / HR \\
& \quad D2YDR = (D1YR - D1YL) / (HR + HL) \\
& \quad 20 \quad \text{SLOPE} = (D2YDR - D2YDL) / HL \\
& \quad 30 \quad \text{IF}(TR \text{NE} T(I+1)) \text{GO TO 10} \\
& \quad IT = IT + 1 \\
& \quad I1 = I1 + 1 \\
& \quad \text{WTI} (I1) = 0, 0 \\
& \quad \text{IF}(IT(1), NE, T(I)) \text{GO TO 10} \\
& \quad IT = IT + 1 \\
& \quad I1 = I1 + 1 \\
& \quad \text{WTI} (I1) = 0, 0 \\
& \quad \text{GO TO 40} \\
\end{align*}
\]

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

\[
\begin{align*}
50 & \quad \text{WTIAV} = \text{SUM} (\text{WTI}, 100) \\
& \quad \text{IF}(\text{WTIAV} \text{EQ} 0, 0) \text{GO TO 70} \\
& \quad \text{WMIN} = \text{WTIAV} - 1.0E-03 \\
& \quad \text{WMAX} = \text{WTIAV} + 1.0E+03 \\
& \quad \text{DO} 50 \quad I1 = 1, NWTI \\
& \quad \text{DUMMY} = \text{WTI}(I1) \\
& \quad \text{IF}((\text{WTI}(I1) \text{GT} \text{WMIN}) \text{AND} \text{AND} (\text{WTI}(I1) \text{LT} \text{WMAX})) \text{WTI}(I1) = \text{WTIAV} / \text{WTI}(I1) \\
& \quad 50 \quad \text{IF} \text{DUMMY} \text{GT} \text{WMAX} \text{WTI}(I1) = 1.0E-03 \\
& \quad \text{IF} \text{DUMMY} \text{LE} \text{WMIN} \text{WTI}(I1) = 1.0E+03 \\
& \quad \text{RETURN} \\
60 & \quad \text{DO} 50 \quad I1 = 1, NWTI \\
& \quad \text{WTI}(I1) = 1.0 \\
& \quad \text{RETURN} \\
END
\end{align*}
\]
Appendix B

B.10 XSQC

FUNCTION XSQC(N,K,BCOEF,R,VCT,WK,YSQ)
----------------------------------------------------
C XSQC calculates the chi-square:
C XSQ = SUM( (Y(I) - SUM( (BCOEF(J) - Y1(J)) * BJ(X(I))) )**2 * E(I) )
C By subtracting Y1 from BCOEF one keeps the numbers fairly small
C thus avoiding round-off error.
C
C AUTHOR: David Hall, May, 1981
C CALLED by BSMTH
C
-------------------------------------------------------------------
REAL BCOEF(N),R(K,N),VCT(N),WK(N),YSQ,XSQC,YSQ
INTEGER I,N,K,J,J1

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 J=1,N-I+1
30 CONTINUE
DO 30 J=1,N-I+1
J1=I+J-1
WK(J)=WK(J)+R(I,J)*BCOEF(J1)
IF(I.EQ.1)GO TO 38
WK(J1)=WK(J1)+R(I,J)*BCOEF(J)
38 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


A NEW LIBRARY OF SUBROUTINES FOR CALCULATING SMOOTHING SPLINES

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.
4) The new subroutines have, as input, an extra set of weights, \( \delta_i \), \( i = 1, N \), 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.
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 the security reclassification group number. The three groups are defined in Appendix 16 of the DRS 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-scare-letter abbreviation in parentheses immediately following the title.

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

5. AUTHORS: 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 approved for publication of the document.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal report 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 sponsoring 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 number(s) of the document by which the document will be classified and controlled by the originating activity. This number must be unique to the document.

9b. OTHER DOCUMENT NUMBER(S): If the document has been assigned any other document numbers (other than the original 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 statements such as:

1. "Qualified requesters may obtain copies of this document from their defense 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), (SD), (CI), (RI), or (UI).

The length of the abstract should be limited to 20 single-spaced standard typewritten lines; 7 1/2 inches long.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a document and should 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.