OLD DOMINION UNIVERSITY RESEARCH FOUNDATION

DEPARTMENT OF MATHEMATICS
SCHOOL OF SCIENCE AND HEALTH PROFESSIONS
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA

FITTING MULTIDIMENSIONAL SPLINES USING STATISTICAL VARIABLE SELECTION TECHNIQUES

By

Dr. Patricia L. Smith, Principal Investigator

Final Report
For the period ending Oct. 18, 1982

Prepared for the
National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia 23665

Under
Grant NAG-1-223
Dr. James R. Schiess, Technical Monitor
Analysis and Computation Division

(NASA-CR-169363) FITTING MULTIDIMENSIONAL SPLINES USING STATISTICAL VARIABLE SELECTION TECHNIQUES Final Report, period ending 18 Oct. 1982 (Old Dominion Univ., Norfolk, Va.) 88 p HC A05/MF A01 CSCL 12A G3/65 N83-10867 Unclas 35499

October 1982

DEPARTMENT OF MATHEMATICS
SCHOOL OF SCIENCE AND HEALTH PROFESSIONS
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA

FITTING MULTIDIMENSIONAL SPLINES USING STATISTICAL VARIABLE SELECTION TECHNIQUES

By

Dr. Patricia L. Smith, Principal Investigator

Final Report
For the period ending Oct. 18, 1982

Prepared for the
National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia 23665

Under
Grant NAG-1-223
Dr. James R. Schiess, Technical Monitor
Analysis and Computation Division

Submitted by the
Old Dominion University Research Foundation
P.O. Box 6369
Norfolk, Virginia 23508-0369

October 1982

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (continued).

LIST OF FIGURES (continued)

LIST OF FIGURES (concluded)

FITTING MULTIDIMENSIONAL SPLINES USING STATISTICAL VARIABLE SELECTION TECHNIQUES

By

Patricia L. Smith[1]

SUMMARY

This report demonstrates the successful application of statistical variable selection techniques to fit splines. Major emphasis is given to knot selection, but order determination is also discussed. Two FORTRAN backward elimination programs using the B-spline basis were developed, and the one for knot elimination is compared in detail with two other spline-fitting methods and several statistical software packages. An example is also given for the two-variable case using a tensor product basis, with a theoretical discussion of the difficulties of their use.

1. INTRODUCTION

Polynomial splines have often been employed in modeling or data fitting when the functional form of the relationship between the dependent and independent variables is unknown. The major problem has been how to avoid under- or overfitting the data. A strictly mathematical approach is to add knots one at a time and move them around until the L_2 (or some other) norm of the errors is less than a preselected tolerance level (ref. 1). A major problem with this approach is that a good fit depends entirely on the subjective selection of the tolerance level. A fitting method which attempts to avoid this problem is the smoothing technique introduced by Reinsch (ref. 2), but it requires the experimenter to have good _a priori_ information about the data or the process which generated it. Both of these methods are currently feasible only for functions of a single variable.

A statistical approach to the curve-fitting problem using the method of cross-validation was introduced by Wahba and Wold (ref. 3). The major

[1] Assistant Professor, Department of Mathematics, Old Dominion University, Norfolk, Virginia 23508.

advantage of this procedure is its automation: no a priori information is needed. There are several disadvantages, however. Every data point is a knot so that the resulting functional form is difficult to use and interpret. In addition, if there are clearly identifiable trends in certain portions of the data such as linearity or sharp bends, this information is lost analytically even though it shows up when the spline is plotted. The practical use of this technique is also currently restricted to functions of one or two variables. The two variable case is considered in Wahba (ref. 4), with higher dimensions discussed in Wahba and Wendelberger (ref. 5).

Other statistical approaches to the variable knot spline problem have considered the knots as parameters in the model. However, this presents problems in finding the least squares sclution and in subsequent statistical estimation and testing procedures because the model is nonlinear. Traditional (ref. 6) as well as Bayesian (ref. 7) approaches have been investigated, but both are limited in scope and application. Further, in most cases, though the knot locations have been variable, their number has been fixed a priori by the analyst. Some exceptions are the works of Ertel and Fowlkes (ref. 8), Smith and Smith (ref. 9), and Agarwal and Studden (ref. 10), but, as with most other approaches mentioned above, they have not been developed to fit splines in several variables.

The technique investigated in this research is the use of variable selection procedures to fit splines. If a pool of knots is fixed in advance, then statistical linear models theory can be applied in a variable selection framework. There are four major advantages of the variable selection approach to fitting splines. First, variable selection procedures are essentially user independent (automatic) in their use of the F test as a stopping criterion. Second, they are widely available in statistical software. Third, final fits may have straightforward intepretations because of their simplicity or theoretical foundation. Fourth, regression diagnostics, such as outlier detection, may be performed. These advantages and other desirable properties are discussed in Section 4, along with a comparison of several methods and software.

The theory applies not only to splines in a single variable, but also to splines in several variable using a tensor product basis. However, as the careful and detailed development of this technique in the one-variable case is considered a crucial step to its use in several variables, discussion of the multivariate case is restricted to Section 7, and includes an example of its successful application to aerodynamic modeling.

The major emphasis of this report is the application of variable selection procedures for choosing the number and location of the knots for splines in a single variable of fixed order (= degree +1). A detailed discussion of this "knot selection" approach is given in Section 2 with examples, comparison of methods and software, and applications in Sections 3-5. Choosing the spline order with the number and location of the knots fixed is of less interest and considered in Section 6 only. FORTRAN programs which apply backward elimination in these two contexts were written as part of this research and discussed in Sections 2 and 6. Their documentation, flowcharts, and listings are given in the Appendix.

LIST OF SYMBOLS

a	angle of attack
a_i	breakpoint for angle of attack
A_{oi}, A_{1i}, A_ℓ	regression coefficients
b	sideslip angle
b_j	breakpoint for sideslip angle
B_{oj}, B_ℓ	regression coefficients
$B_i(x,y)$	two-variable spline basis element
C_0, C_1	regression coefficients
C^{-1}	class of discontinuous functions
$C^0, C^1, C^{k-2}, C^{k-3}$	functions with continuity class 0, 1, k-2, k-3
C_n	yawing moment coefficient

$C_{n_p}, C_{n_r}, C_{n_{\delta_a}}, C_{n_{\delta_r}}$	partial derivative of C_n with respect to p, r, δ_a, δ_r
C_z	vertical force coefficient
$C_{z_a}, C_{z_{\delta_e}}, C_{z_q}$	partial derivatives of C_z with respect to a, δ_e, q'
D_{ij}, D_ℓ	regression coefficients
$f, f^{(1)}, \ldots, f^{(k)}$	function and its first k derivatives
i	index
j	index
k	spline order (degree + 1)
ℓ, ℓ_1, ℓ_2	number of breakpoints
N	normal distribution
n	sample size
p'	nondimensional rolling velocity
q'	nondimensional pitch rate
Q	quantile function
r'	nondimensional yawing velocity
t_i	breakpoint
u	independent variable
u_1, u_2, u_3	number of breakpoints
x	independent variable
$x_0, \ldots, x_3, x_i$	breakpoints for x

y	dependent variable
$y_0,\ldots,y_3,y_j$	breakpoints for y
α	significance level
$\beta_0,\ldots,\beta_{32},\beta_1$	regression coefficierts
$\delta_a,\delta_\ell,\delta_r$	aileron, elevator, and rudder deflection
ϵ	random error
μ	mean
σ	standard deviation
$*_{ij}$	gridpoint (x_i,y_j)

Abbreviations:

KS	knot selection
MSE	mean squared error
SS	Smith-Smith
SSE	error sum of squares
WW	Wahba-Wold

2. THE KNOT SELECTION (KS) PROCEDURE

Statistical variable selection procedures can be used as a KS procedure to choose the number and location of knots in fitting splines. The "+" function basis is suitable for this, at least theoretically, because it is easily interpreted. Knots and knot multiplicities correspond to individual terms so that selection or deletion of terms is equivalent to selection or deletion of knots. The knots are thus selected indirectly. For example, a continuous linear spline with knots $t_1,\ldots,t_k$ may be written as $\beta_o + \beta_1 x + \Sigma_2^\ell \beta_i(x - t_i)_+$, where $u_+ = u$ for $u > 0$ and zero otherwise. Selection of the "spline term" $(x - t_j)_+$ is actually selection of the knot t_j. Because we don't know where the breakpoints should be, we provide as candidate variables a liberal number of spline terms, i.e., a pool of knots, more [illegible] expect or want to eventually use, and blanket the domain. [illegible], the [illegible]tual number and location of the knots used in the final model is unknown at the beginning in the sense that we are selecting from a larger set.

While "+" functions are easily defined in current statistical software packages and fit into the statistical hypothesis testing framework without modification (ref. 11), computational problems such as carry-over in round-off error and multicollinearity greatly restrict their use. As will be seen in Section 4, the backward elimination (stepdown) procedures are especially troublesome because all terms must be fit initially. An alternative is the use of the computationally advantageous B-spline basis (ref. 1). Unfortunately, it does not fit easily into the hypothesis testing framework and cannot be used in existing statistical software packages. There was thus a need for the development of a KS procedure using B-splines. Construction of hypotheses which are useful in B-spline regression, including testing the importance of knots, has been detailed in Smith (ref. 12). As part of this research, these results have been implemented in two FORTRAN computer programs, one of which accommodates the backward elimination of knots using the B-spline basis. Examples in Section 3 give the results of using this FORTRAN program, and comparisons with several statistical software packages, as well as with other statistical spline-fitting methods, are detailed in Section 4.

The use of variable selection is a sort of compromise between the techniques which use either fixed or variable knots. Its most important advantage, and one which makes possible all others, is that because the maximum number and location of the knots is fixed in advance, the statistical theory of general linear models applies. Consequently, the least squares solution is easily obtained at any given step, and hypothesis testing and interval estimation are straightforward. As mentioned earlier, details for using the B-spline basis are given in reference 12. The selection of knots can thus be accomplished through t tests. This fits exactly into the variable selection framework for (1) spline models in a single variable, (2) models in several variables with spline terms in one or more variables, and (3) models in several variables with tensor products defining higher dimensional splines. Also, trends in the data in one or more variables may be easily detected through the selection of a few knots. Several examples of this will be given in the next section. Further, in some experimental situations, models may be easily interpreted because the coefficients are physically meaningful, as in some examples in Sections 5 and 7.

3. EXAMPLES OF THE KS PROCEDURE

Four data sets were examined using the FORTRAN knot elimination program. The maximum number of continuity constraints allowed for any given order were imposed. The first data set, the Indy data, is rather simplistic but has appeared in the statistical literature several times in connection with curve-fitti[illegible] splines. It is a record of the average winning speeds at the I[illegible]polis 500 from 1911-1971, except for 1917-1918 and 1942-1945, during the two World Wars when the race was not run. Poirier (ref. 13) fit the data with a cubic spline with 2 knots, one each at the midpoint of the non-racing years. The data were coded so that x = year - 1910 with knots 7.5 and 33.5. The output and graphs from the knot elimination routine are shown in Figures 3.1 to 3.4, with circles around the function values of the knots. Using an F-table value of 8.0 (α = 0.01), the KS procedure eliminates both knots so that a cubic polynomial is adequate to fit the data. If a linear rather than a cubic spline is fit, only the knot at x = 7.5 can be eliminated (Figures 3.5 to 3.7).

The second example is noisy data generated from the function used in reference 3

$$f(x) = 4.26(e^{-x} - 4e^{-2x} + 3e^{-3x})$$

ORIGINAL PAGE IS
OF POOR QUALITY

INDY DATA

YEAR	Y	X
1911	74.530	1.
1912	78.720	2.
1913	75.931	3.
1914	82.470	4.
1915	89.840	5.
1916	84.000	6.
1919	88.050	9.
1920	88.620	10.
1921	89.620	11.
1922	94.480	12.
1923	90.950	13.
1924	98.230	14.
1925	101.130	15.
1926	95.904	16.
1927	97.545	17.
1928	99.482	18.
1929	97.585	19.
1930	100.448	20.
1931	96.629	21.
1932	104.114	22.
1933	104.162	23.
1934	104.863	24.
1935	106.240	25.
1936	109.069	26.
1937	113.580	27.
1938	117.200	28.
1939	115.035	29.
1940	114.277	30.
1941	115.117	31.
1946	114.820	36.
1947	116.338	37.
1948	119.814	38.
1949	121.327	39.
1950	124.002	40.
1951	126.244	41.
1952	128.922	42.
1953	128.740	43.
1954	130.840	44.
1955	128.209	45.
1956	129.490	46.
1957	135.601	47.
1958	133.791	48.
1959	138.857	49.
1960	138.767	50.
1961	139.130	51.
1962	140.293	52.
1963	143.137	53.
1964	147.350	54.
1965	151.338	55.
1966	144.317	56.
1967	151.207	57.
1968	152.882	58.
1969	156.867	59.
1970	155.749	60.
1971	157.735	61.

THE ORDER K = 4

THE # INTERVALS L = 3

THE DIMENSION N = 6

BREAKPOINTS	CONTINUITY CONDITIONS
0.00000000	0
7.50000000	3
33.50000000	3
62.00000000	

T	INDEX	
0.00000000	1	
0.00000000	2	
0.00000000	3	
0.00000000	4	
		KEND(1)= 4
7.50000000	5	
		KEND(2)= 5
33.50000000	6	
		KEND(3)= 6
62.00000000	7	
62.00000000	8	
62.00000000	9	
62.00000000	10	

L= 3 F-TABLE VALUE IS 8.00000000

SSE= 385.29008218 MSE= 7.86306290

F-RATIOS ARE:	BREAKPOINTS ARE
1.41909767	7.50000000
.32417463	33.50000000

BREAKPOINT 33.500 IS ELIMINATED

L= 2 F-TABLE VALUE IS 8.00000000

SSE= 387.83908771 MSE= 7.75678175

F-RATIOS ARE:	BREAKPOINTS ARE
1.11982607	7.50000000

BREAKPOINT 7.500 IS ELIMINATED

SSE= 396.52533411 MSE= 7.77500655

PROCEDURE TERMINATES WITH L= 1 AND K= 4

N	COEF	S.E.
1	74.74373319	1.56753075
2	110.79327327	3.10571455
3	113.63057381	3.16805157
4	161.44053672	1.53807401

Figure 3.1. Output for knot elimination. Indy data. Cubic spline.

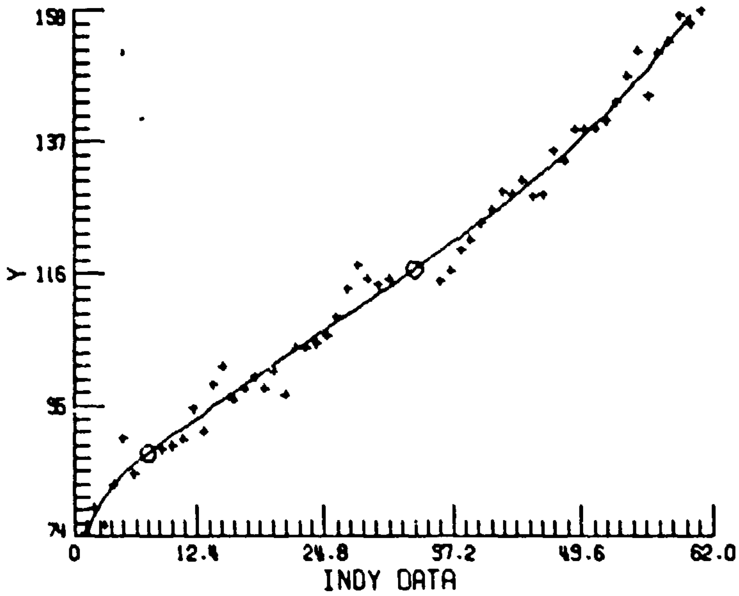

Figure 3.2. First step of knot elimination. Indy data. Cubic Spline.

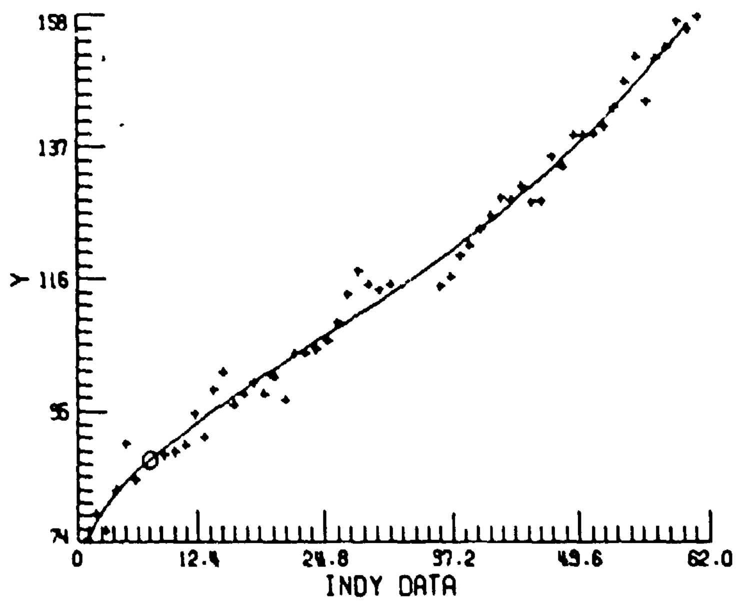

Figure 3.3. Second step of knot elimination. Indy data. Cubic spline.

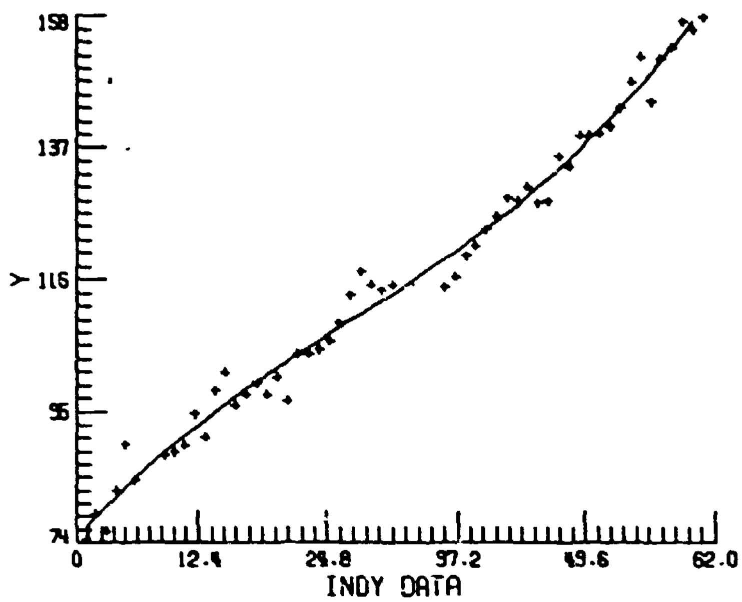

Figure 3.4. Third and final step of knot elimination. Indy data. Cubic spline.

```
L= 3      F-TABLE VALUE IS       8.00000000

SSE=    453.46808946     MSE=       8.89153115

F-RATIOS ARE:     BREAKPOINTS ARE
      5.60928518        7.50000000
     16.27042823       33.50000000

BREAKPOINT  7.500 IS ELIMINATED


L= 2      F-TABLE VALUE IS       8.00000000

SSE=    503.34322235     MSE=       9.67967735

F-RATIOS ARE:     BREAKPOINTS ARE
      9.83428353       33.50000000

NO BREAKPOINT CAN BE ELIMINATED


PROCEDURE TERMINATES WITH L=  2 AND K=  2

 N         COEF                  S.E.
 1      78.15096288          1.12854643
 2     115.82035775           .91553083
 3     157.04127664          1.14098079
```

Figure 3.5. Partial output for knot elimination. Indy data. Linear spline.

ORIG...
OF POOR ...TY

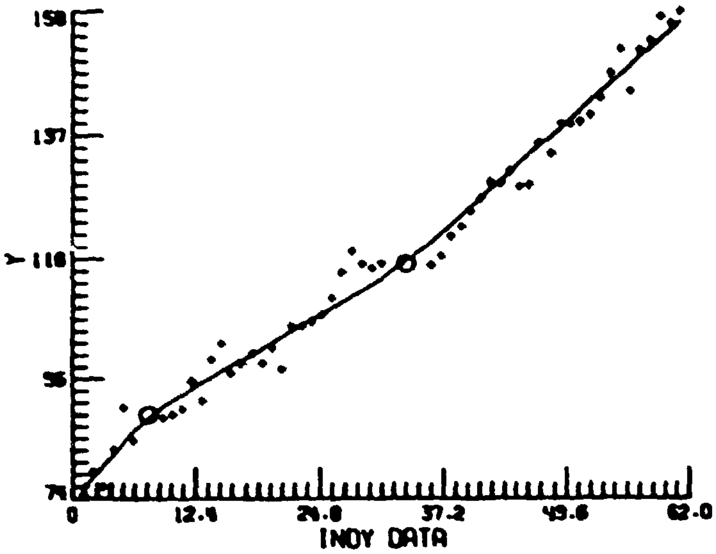

Figure 3.6. First step of knot elimination. Indy data. Linear spline.

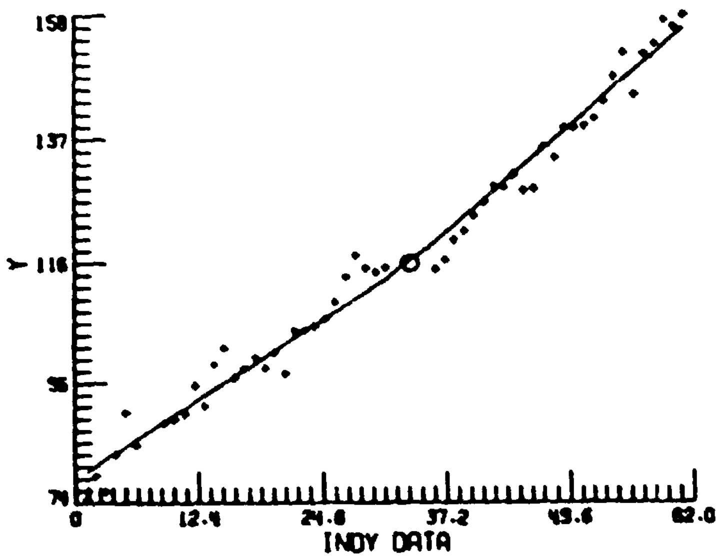

Figure 3.7. Second and final step of knot elimination. Indy data. Linear spline.

for $x \epsilon [0,3]$. For x starting at zero, we generated 100 data points at intervals of 1/32 up to 99/32 and added normal random noise, $N(\mu = 0, \sigma = .2)$, the value of σ the same as that used by Wahba and Wold (WW). A graph of the function and generated data is shown in Figure 3.8. Figures 3.9 to 3.27 show graphical results of the stepdown procedure for cubic splines starting with 19 equally spaced interior knots, and using an F-table value of 8.0. By examining this sequence of graphs, it becomes clear how the elimination of knots makes the spline smoother by making it less noise dependent.

An F-table value of 4.0 ($\alpha = 0.05$) rather than 8.0 results in stepdown terminating with 5 knots remaining (Fig. 3.23, p. 20). The latter fit is more data dependent and clearly inferior in terms of recovering the desired function. Use of the larger F value thus seems appropriate and keeps the procedure from terminating "prematurely." Graphs of starting and ending fits to the data, beginning with 39 interior knots, are shown in Figures 3.28 to 3.29, and the results are roughly the same as when 19 knots are used initially (Figure 3.27, p. 22). A phenomenon which occurs throughout most of these fits is the downward hook in the upper range of the x's due to a cluster of 3 data points. Figure 3.30 shows the conclusion of stepdown with those 3 points omitted and helps to illustrate the fact that different noise results in different fits.

The method used by Wahba and Wold to recover the function is a modification of the smoothing technique introduced by Reinsch (ref. 2). They use cross-validation to determine the smoothing parameter, and their resulting fit is shown in Figure 3.31. Referring again to Figure 3.27, p. 22, we see that the results of the two methods compare very favorably. A more detailed comparison of these methods and others is made in the next section.

Smith and Smith (SS) (ref. 9) examine a scaled version of the WW function, $f(x) = 4.26 \left(e^{-3.25x} - 4e^{-6.5x} + 3e^{-9.75x}\right)$ for $x \epsilon [0,1]$. A sample of size 600 equally spaced points was generated, and a variance of 0.039 (as in SS) was used for the normally distributed zero mean noise. Results from

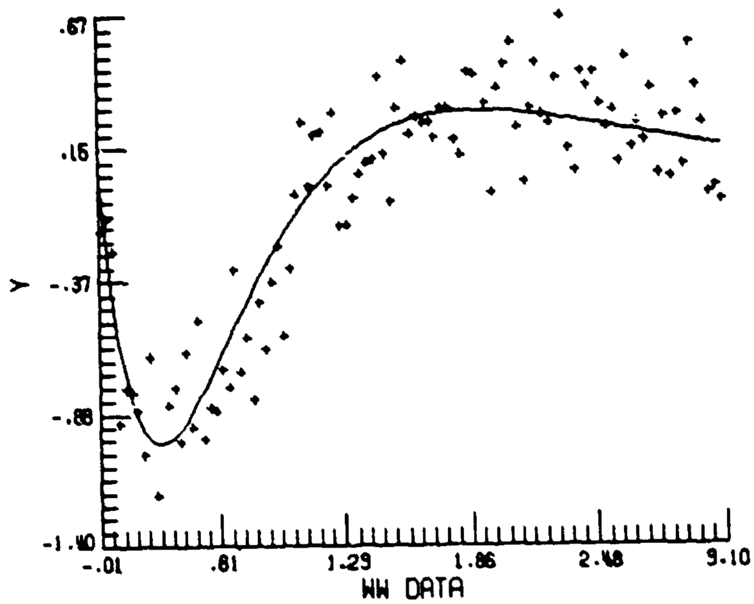

Figure 3.8. The Wahba-Wold (WW) function and data generated from it.

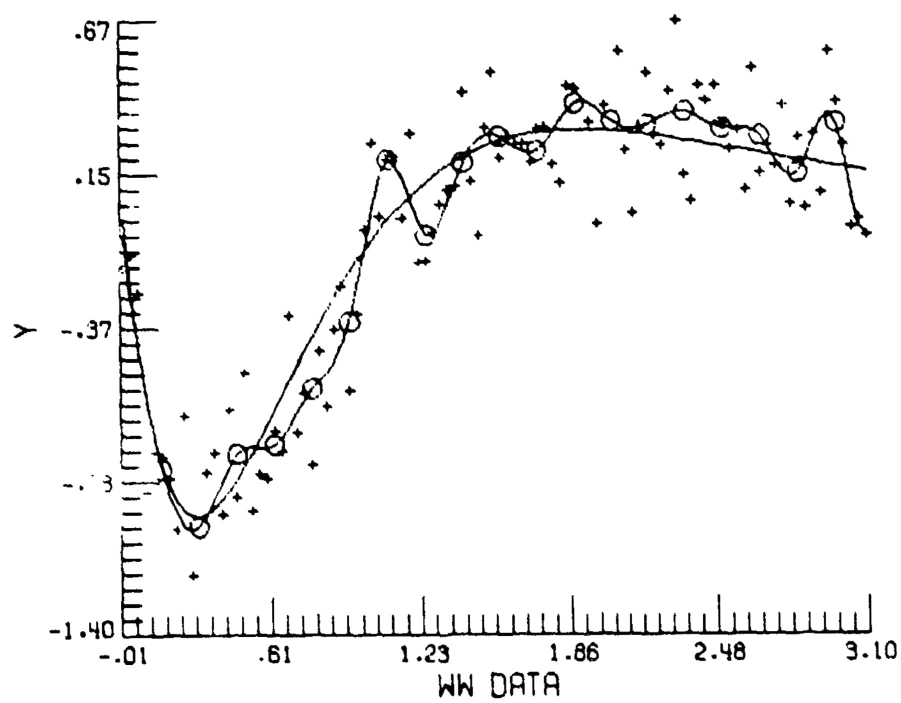

Figure 3.9. First step of knot elimination. WW data.

ORIGINAL PAGE IS
OF POOR QUALITY

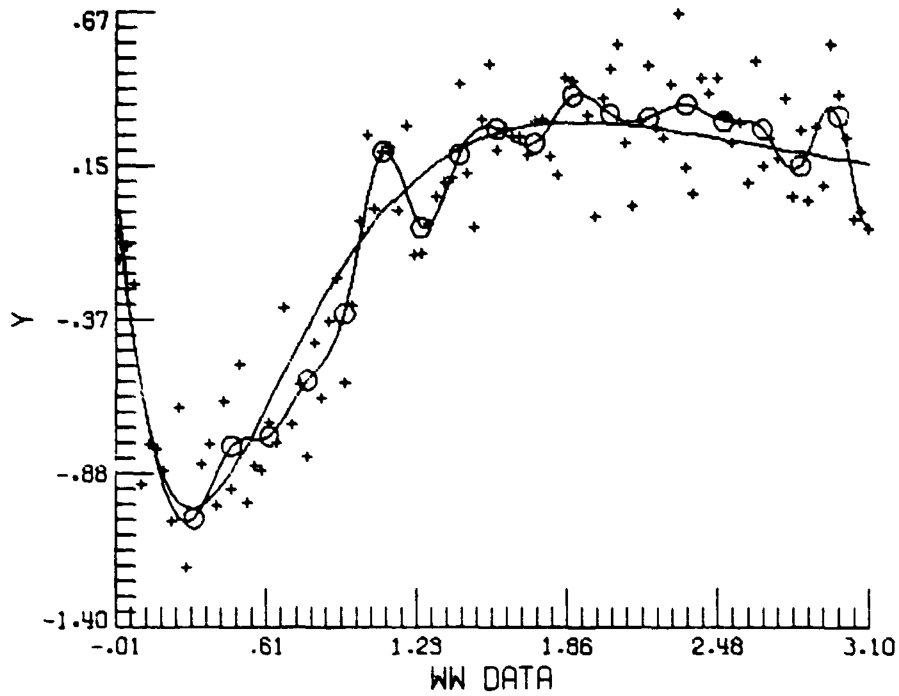

Figure 3.10. Second step of knot elimination. WW data.

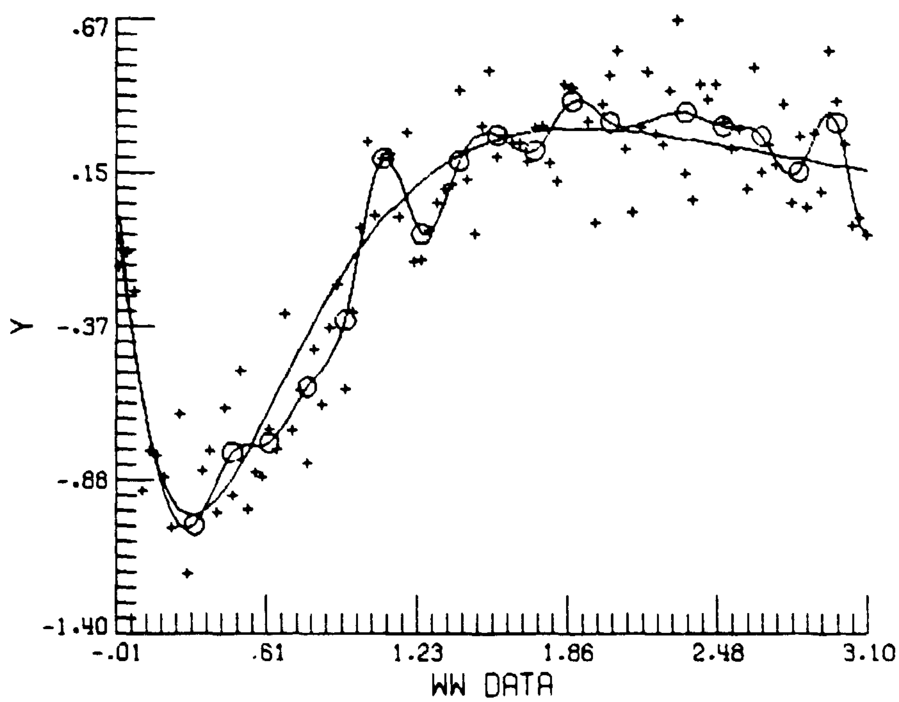

Figure 3.11. Third step of knot elimination. WW data.

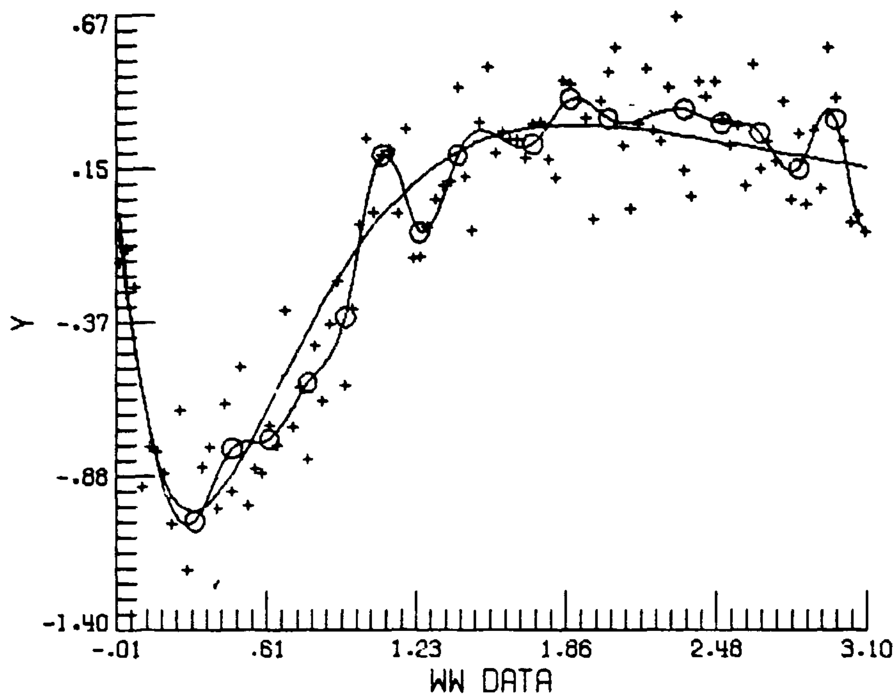

Figure 3.12. Fourth step of knot elimination. WW data.

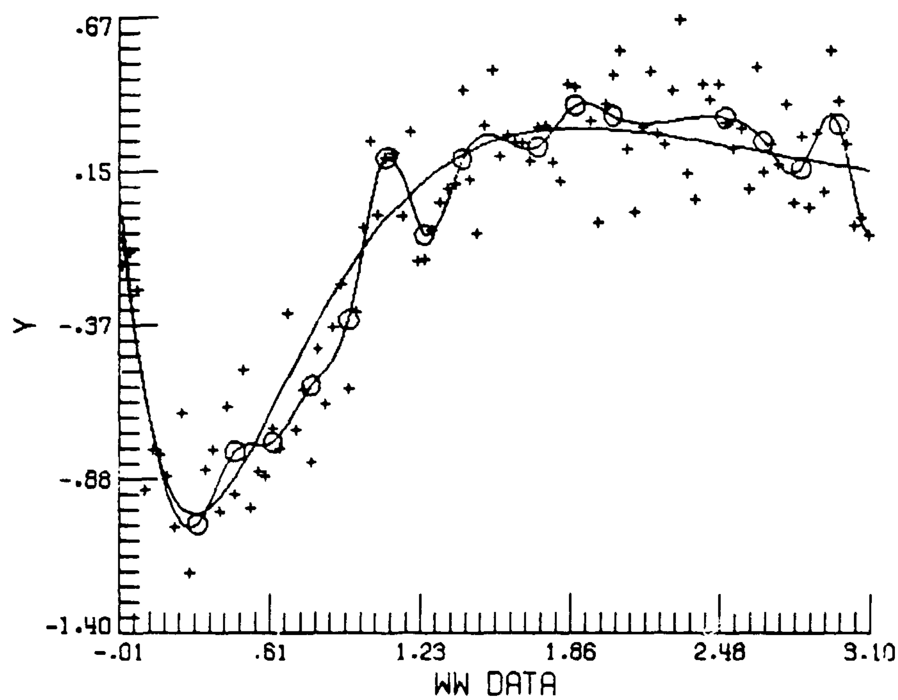

Figure 3.13. Fifth step of knot elimination. WW data.

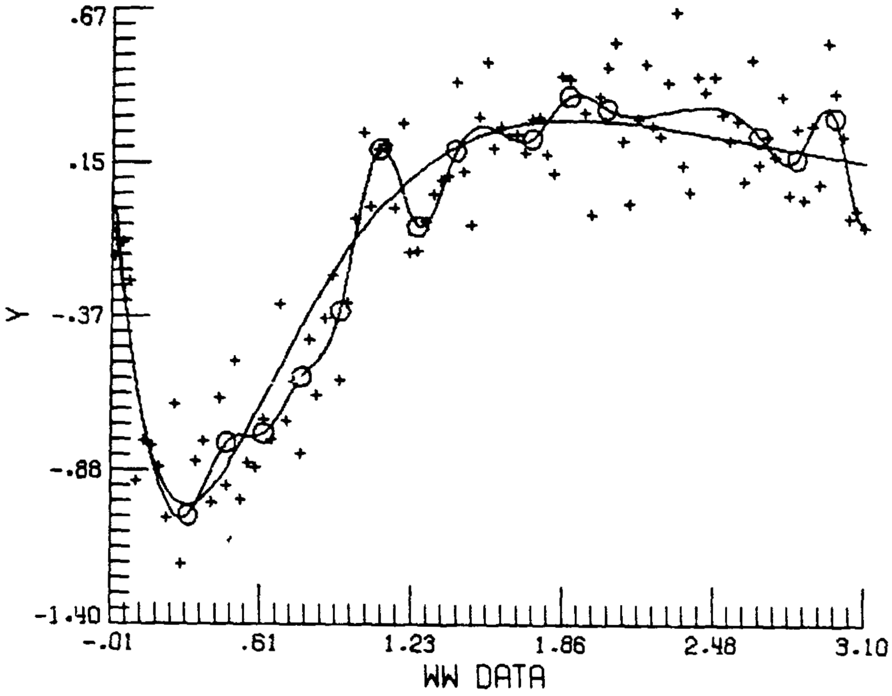

Figure 3.14. Sixth step of knot elimination. WW data.

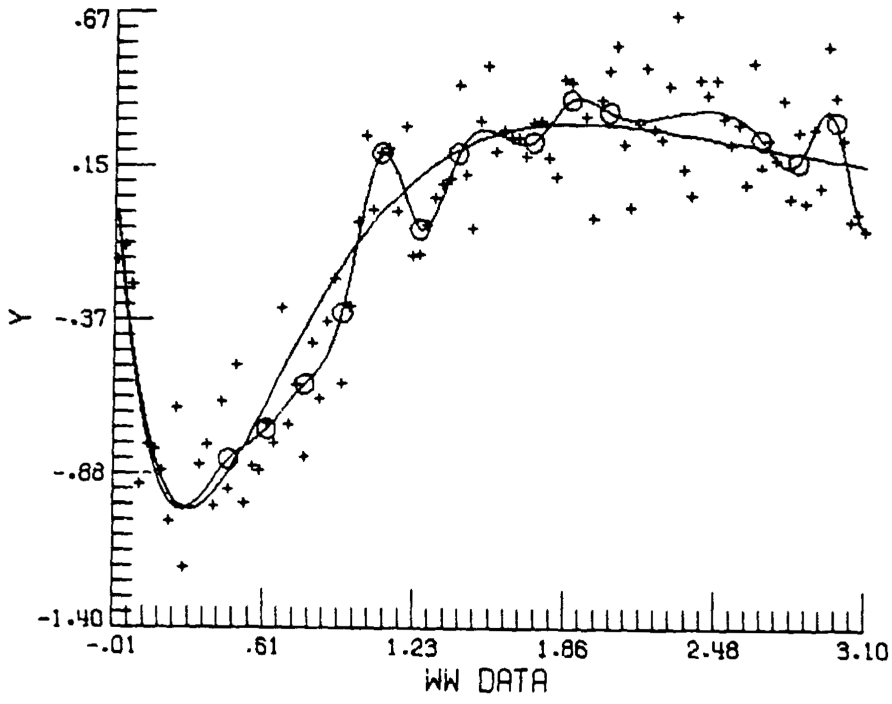

Figure 3.15. Seventh step of knot elimination. WW data.

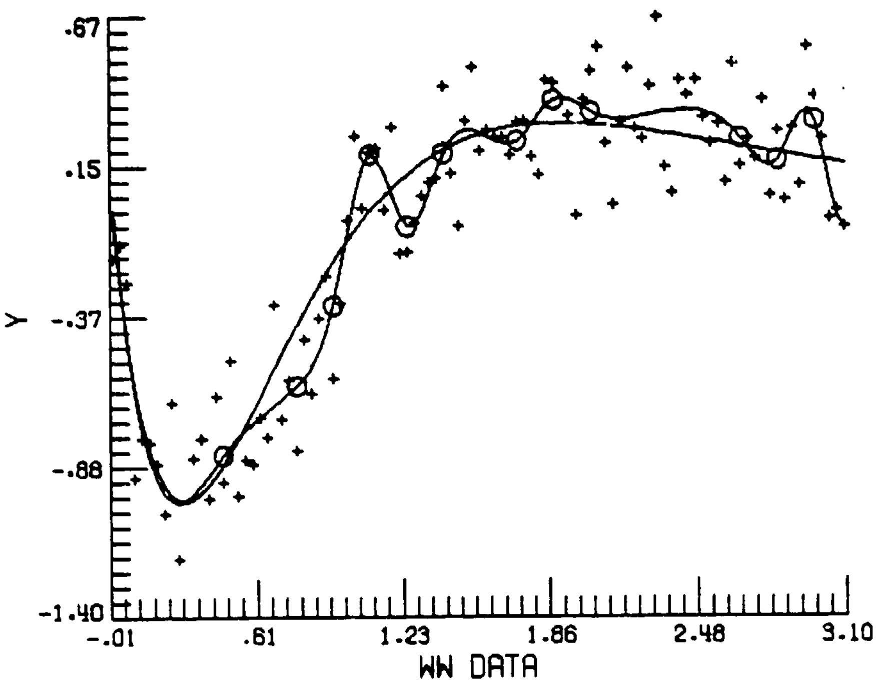

Figure 3.16. Eighth step of knot elimination. WW data.

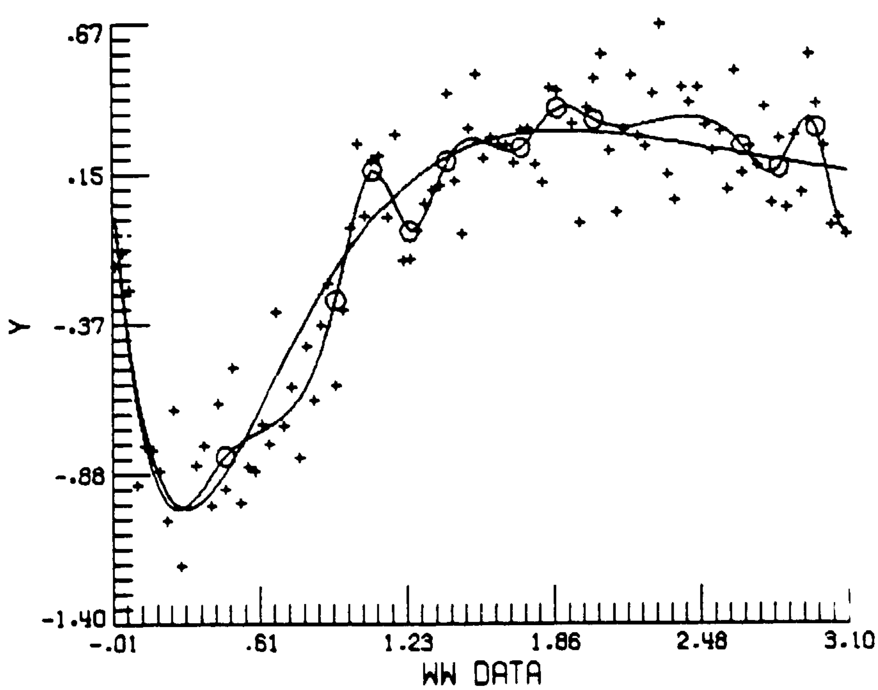

Figure 3.17. Ninth step of knot elimination. WW data.

OF POOR QUALITY

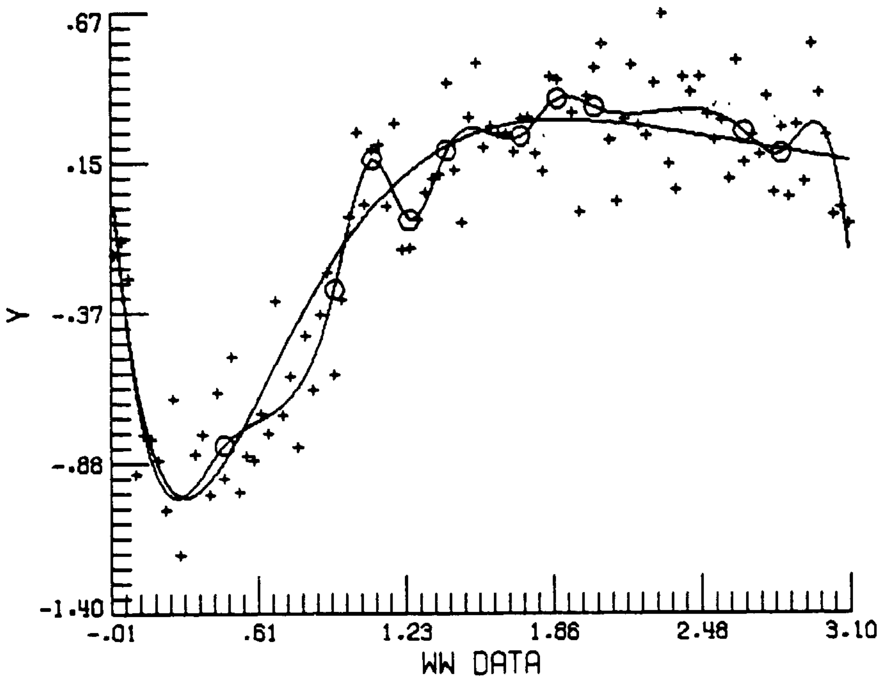

Figure 3.18. Tenth step of knot elimination. WW data.

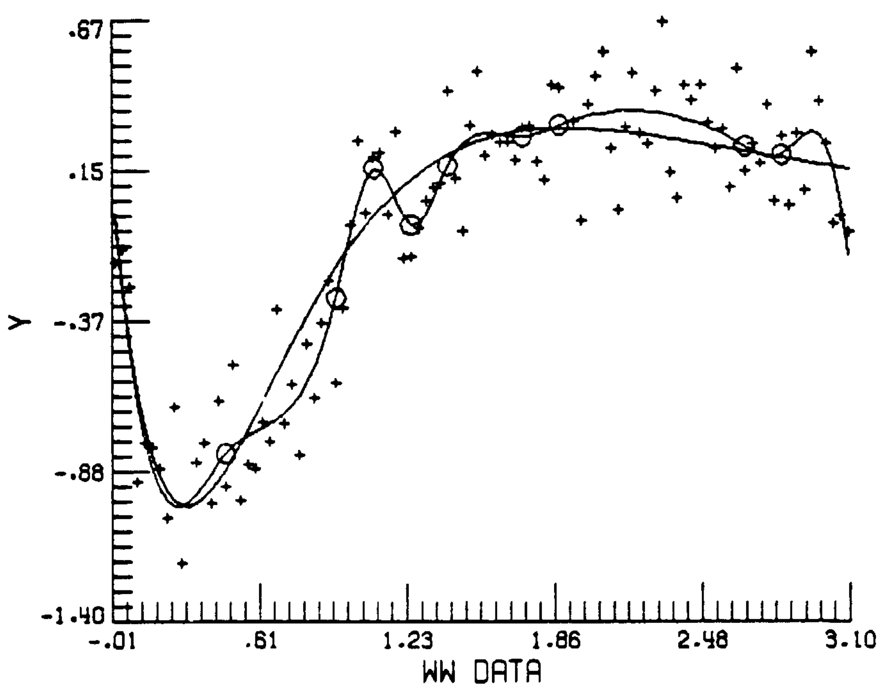

Figure 3.19. Eleventh step of knot elimination. WW data.

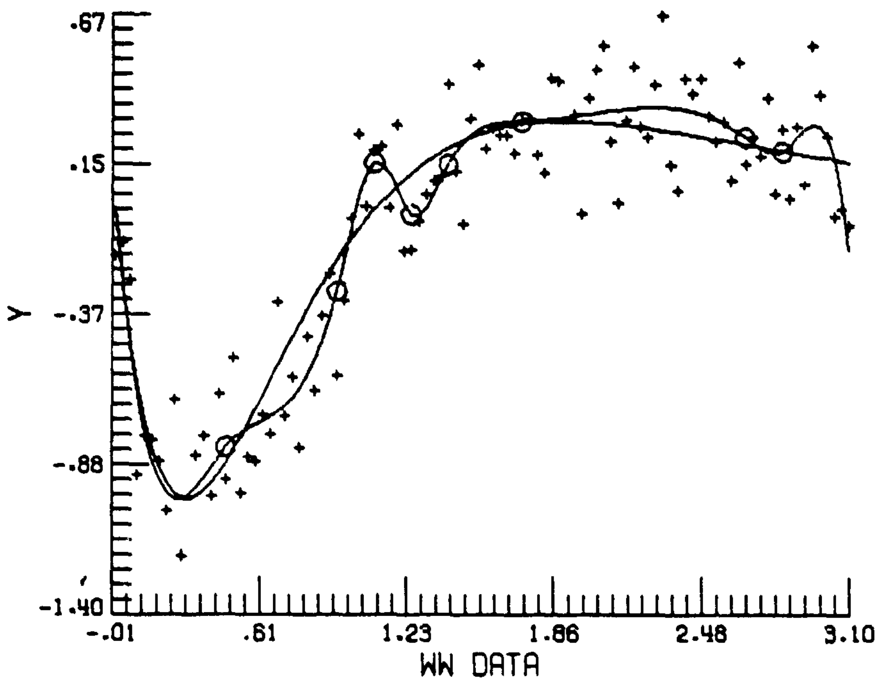

Figure 3.20. Twelfth step of knot elimination. WW data.

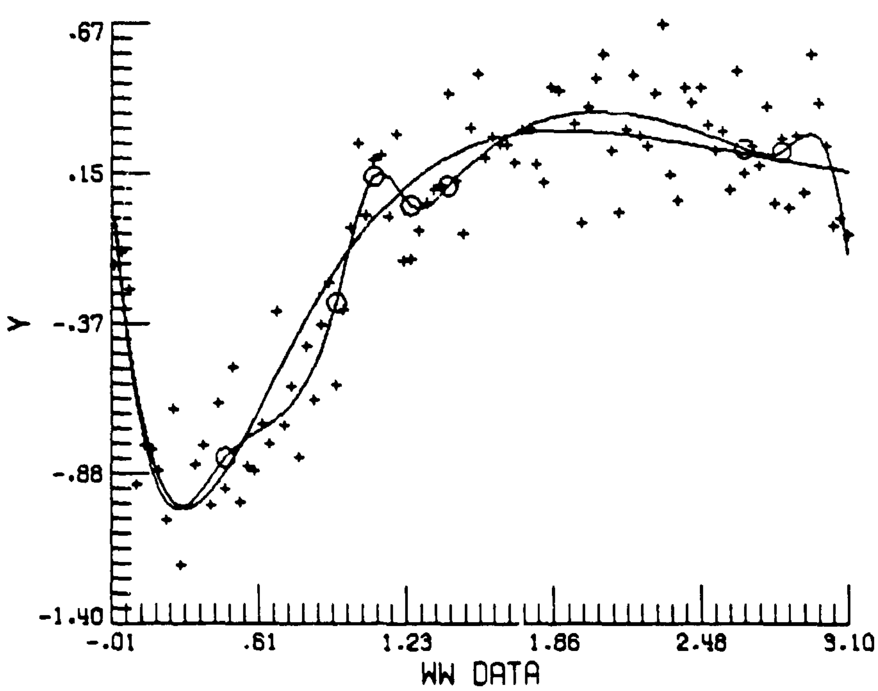

Figure 3 21. Thirteenth step of knot elimination. WW data.

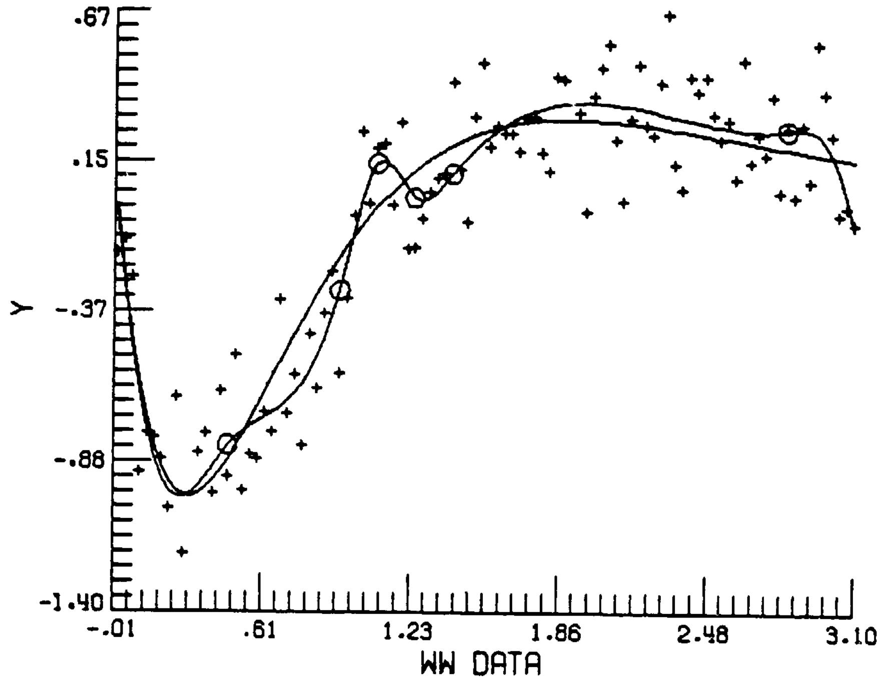

Figure 3.22. Fourteenth step of knot elimination. WW data.

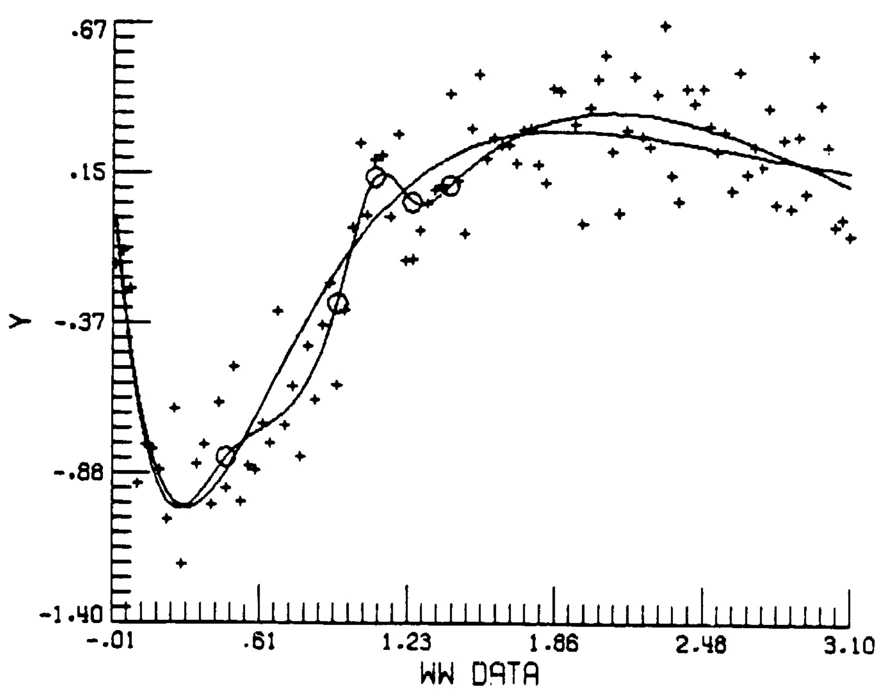

Figure 3.23. Fifteenth step of knot elimination. WW data.

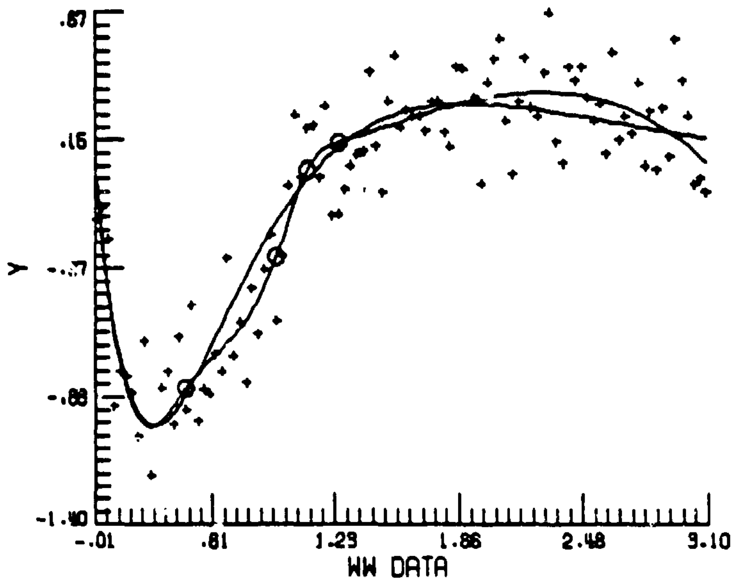

Figure 3.24. Sixteenth step of knot elimination. WW data.

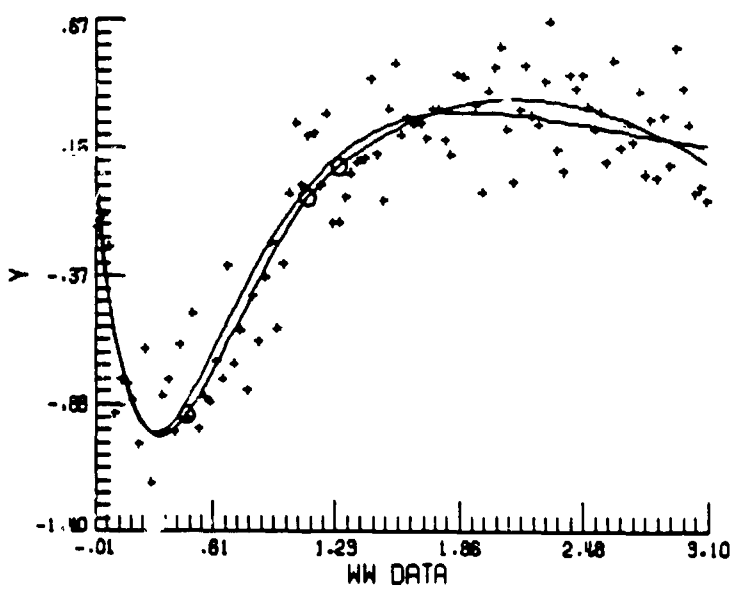

Figure 3.25. Seventeenth step of knot elimination. WW data.

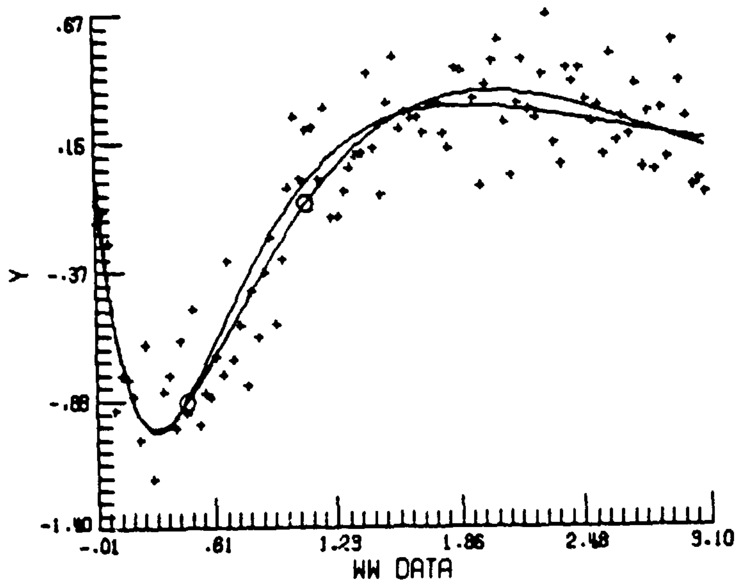

Figure 3.26. Eighteenth step of knot elimination. WW data.

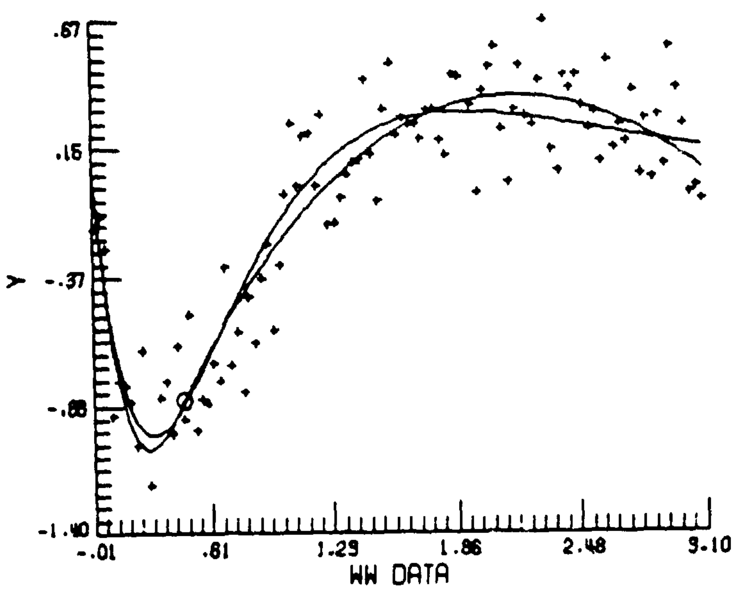

Figure 3.27. Nineteenth and final step of knot elimination. WW data.

ORIGINAL PAGE IS
OF POOR QUALITY

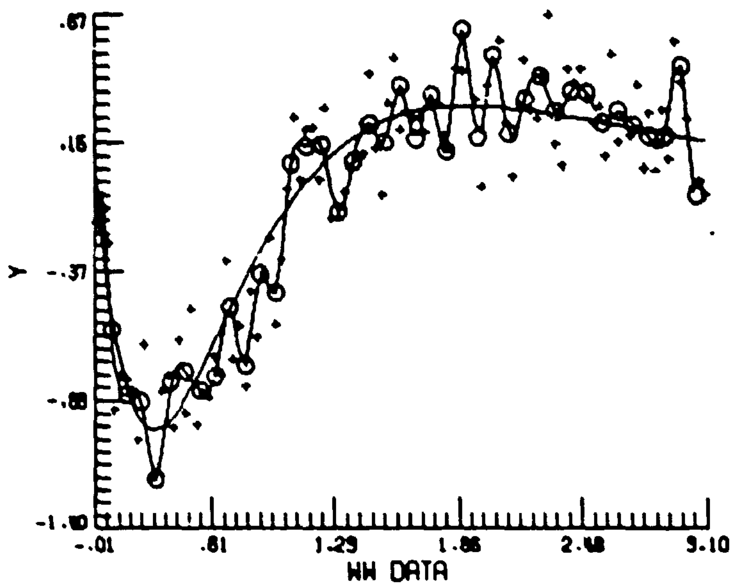

Figure 3.28. First step of knot elimination with 39 interior knots. WW data.

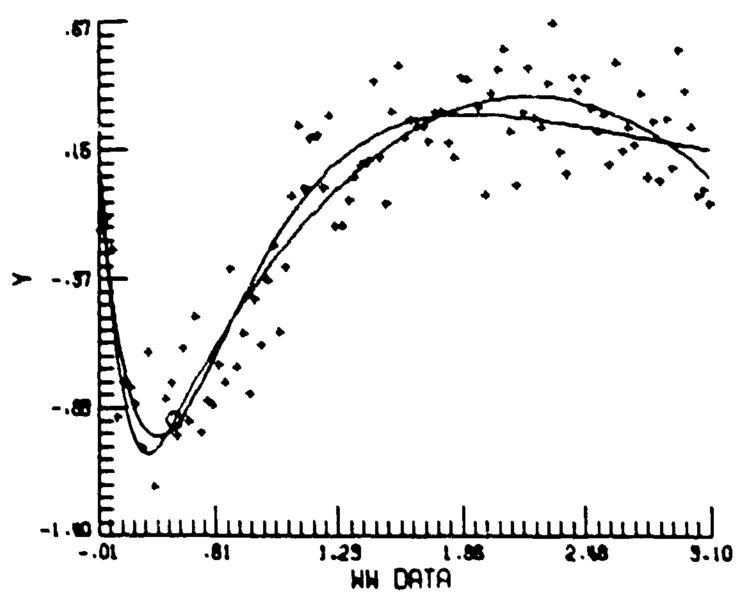

Figure 3.29. Final step of knot elimination from 39 knots. WW data.

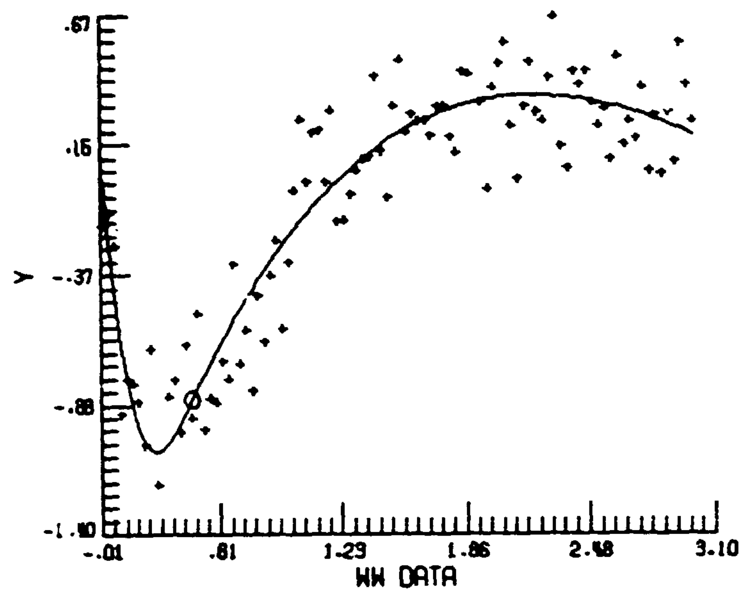

Figure 3.30. Final step of knot elimination with 3 data points in upper x-range omitted. WW data.

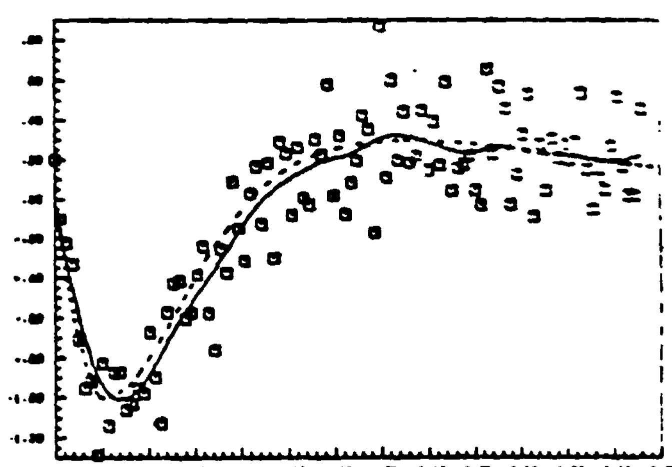

Figure 3.31. Spline fit obtained by cross-validation by Wahba and Wold. WW data.

two stepdown runs fitting cubic splines are shown in Figures 3.32 to 3.33, beginning with 19 and 49 knots. Three knots remain in Figure 3.32 with a slightly wigglier fit than that in Figure 3.33 with one remaining knot. These results show that a larger knot selection pool allows reduction to possibly a fewer number of final knots and a smoother fit, which, for simplicity, is more desirable.

Smith and Smith use asymptotic results to determine a stopping rule for adding knots one at a time to the model. Figure 3.34 shows their results using cubic splines overlaid on the true function. The data were not plotted so that the distinctions between the two functions would not be lost. Applying stepdown using these 9 initial knots resulted in Figure 3.35, a fit which smooths the wiggles visible in Figure 3.34. As seen in the two previous figures, however, using a larger pool of knots results in a smoother and more satisfactory recovery of the function. The SS method is compared in more detail to both the WW and KS methods in the next section.

The final function examined is $f(x) = \sin(x^2)$ for $x \in [0,4.5]$, which allows for more than two periods of the sine wave and gradually increases the frequency. Three hundred data points were used with $\sigma = .2$ for the normal noise. Beginning and ending cubic spline fits from a stepdown run are shown in Figures 3.36 to 3.37, starting with 19 interior knots and ending with 9. We note that more knots are needed for the final fit than for the functions previously discussed due to the increased curvature of the function. Most of the wiggliness in the initial spline fit occurs on the more gradual slope at the lower end of the x-range and is removed as knots are removed. This phenomenon also occurs on the "flat" portion of the SS and WW data.

In order to assess the effects of a lower noise level on the KS technique, random variables used for the noise on the WW function were generated using $\sigma = .1$ and $.05$. Final fits are shown in Figures 3.38 to 3.39, and referring back to Figure 3.27, p. 22, which shows results using $\sigma = .2$, we see that fitting data with a lower noise level results in more knots remaining at the end of the procedure. This tendency is especially striking when data from the function itself is fit, that is, when no noise is added so that to

ORIGINAL PAGE IS
OF POOR QUALITY

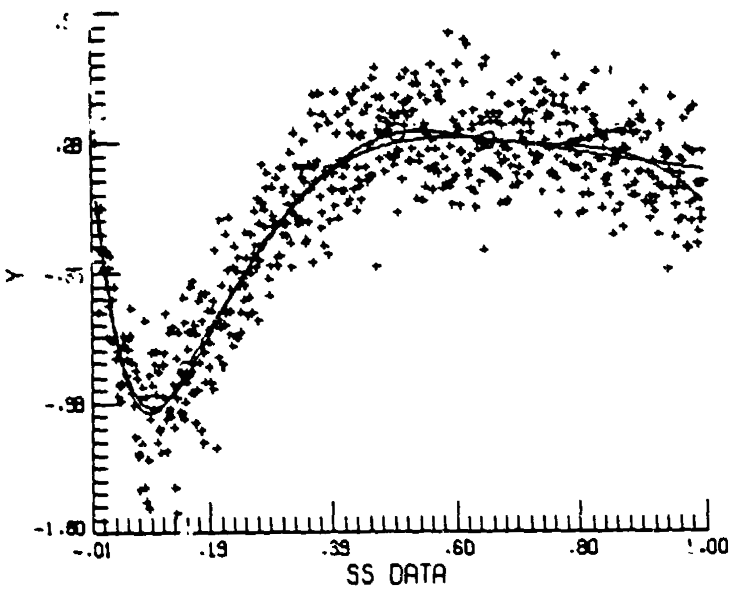

Figure 3.32. Final step of knot elimination from 19 knots. SS data.

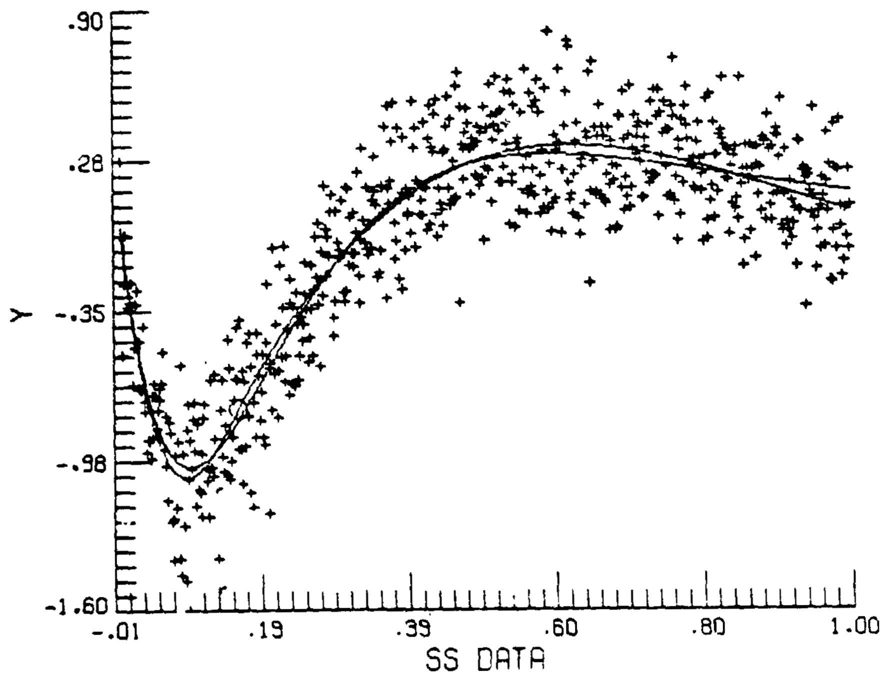

Figure 3.33. Final step of knot elimination from 49 knots. SS data.

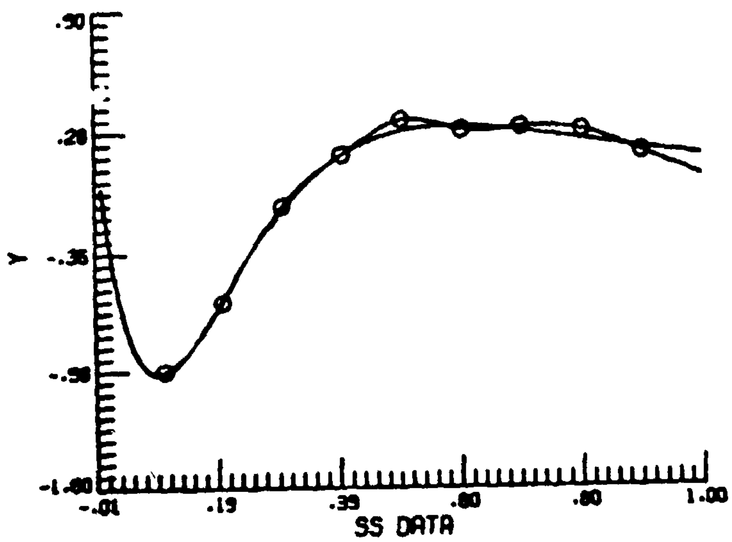

Figure 3.34. Cubic spline solution of Smith and Smith. SS data. (Actual data not shown.)

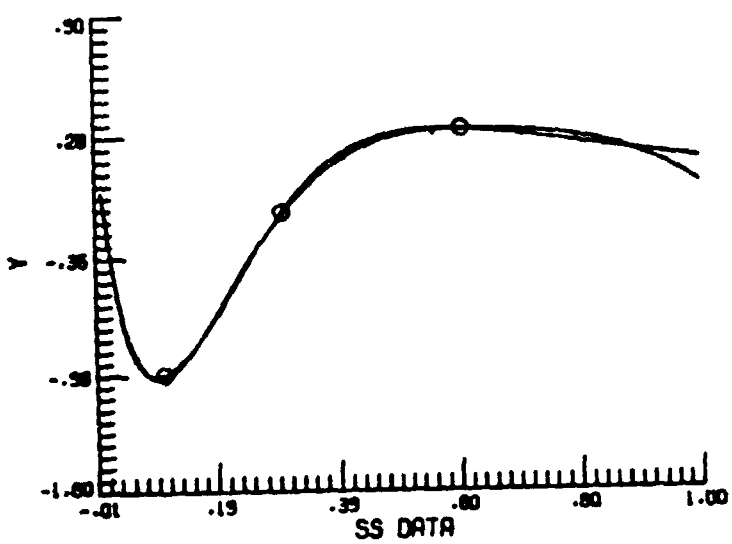

Figure 3.35. Final step of knot elimination from 9 knots. Cubic splines. SS data. (Actual data not shown.)

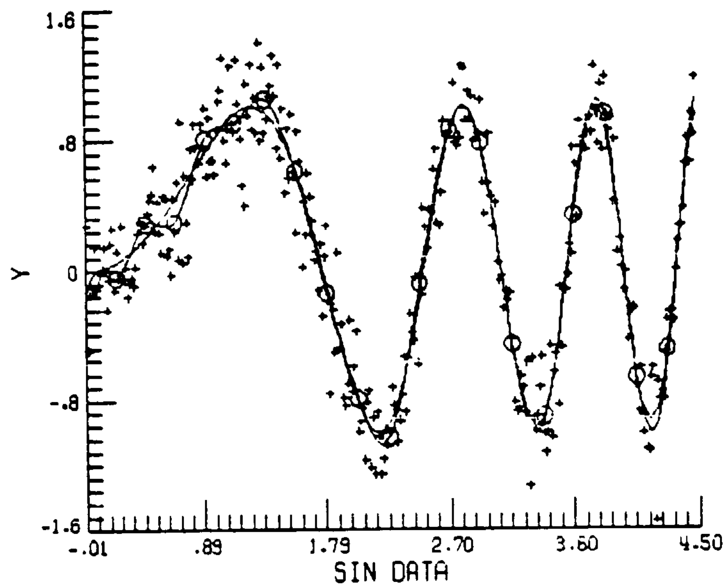

Figure 3.36. First step of knot elimination with 19 knots. True function is sin (x^2).

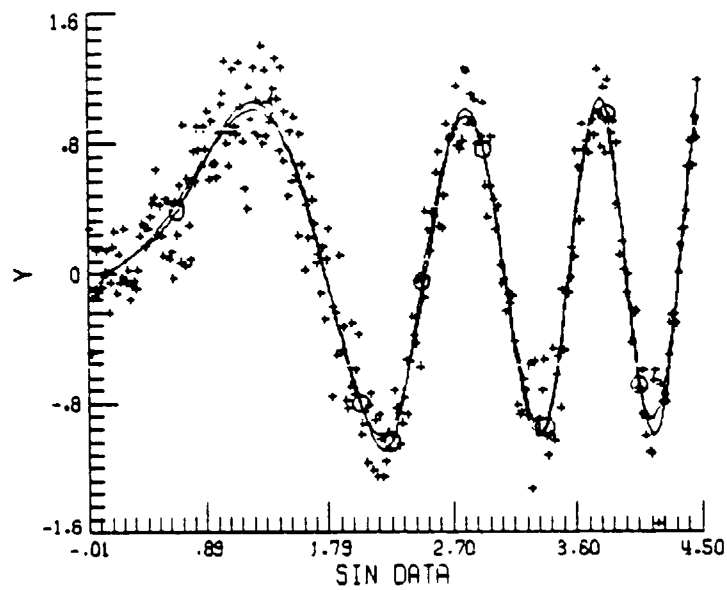

Figure 3.37. Final step of knot elimination from 19 knots. True function is sin (x^2).

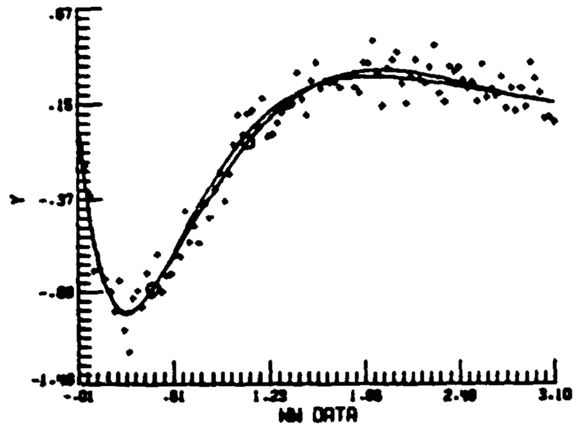

Figure 3.38. Final step of knot elimination from 19 knots with $\sigma = 0.1$ in the noise. WW data.

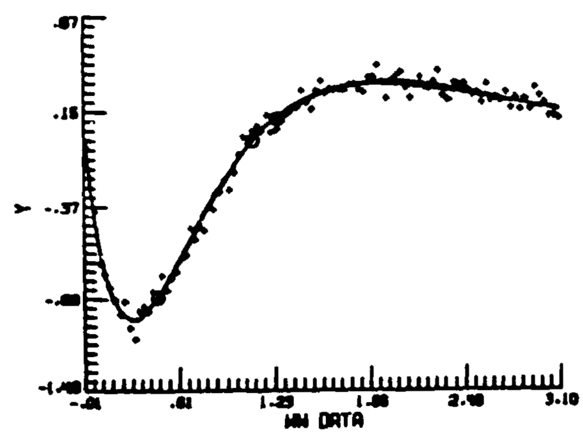

Figure 3.39. Final step of knot elimination from 19 knots with $\sigma = 0.05$ in the noise. WW data.

recover the function we actually need to interpolate. A stepdown from 19 knots results in a spline with 12 knots as shown in Figures 3.40 to 3.41. Both the true and fitted functions are graphed, but there is no perceptible difference between the two.

Wiggliness in data should be smoothed (i.e., ignored) if it is perceived as noise, but should be fit if it is perceived as trends in the underlying process. Thus, a danger in applying the KS technique is using too small or too large a pool of knots. The former problem is illustrated quite well in Figures 3.42 to 3.43, where noisy data generated from sin (x^2) is fit with the KS technique beginning with too few knots to allow the bending necessary to recover the function, especially near the third peak. It is interesting to see that the three knots eliminated were in the lower end of the x range where the underlying function is not wiggly. A comparison of Figures 3.37, p. 28, and 3.42 reveals that both have 9 knots, but a better fit is obtained from the one which began with 19 knots (Fig. 3.37): its 9 knots are more selectively and better placed.

4. COMPARISON OF METHODS AND SOFTWARE

In the previous section, two functions introduced in the literature (WW and SS) were examined using the FORTRAN knot elimination program. The purpose was to compare results, which we do in this section, in light of what we consider to be the most desirable properties of curve-fitting with splines. These are:

(1) good results;
(2) computational efficiency;
(3) diagnostics capabilities;
(4) user independence;
(5) ease of interpretation; and
(6) ease of use.

We also give in this section the results of using several statistical software packages on the Indy and WW data, fitting both linear and cubic splines.

ORIGINAL PAGE IS
OF POOR QUALITY

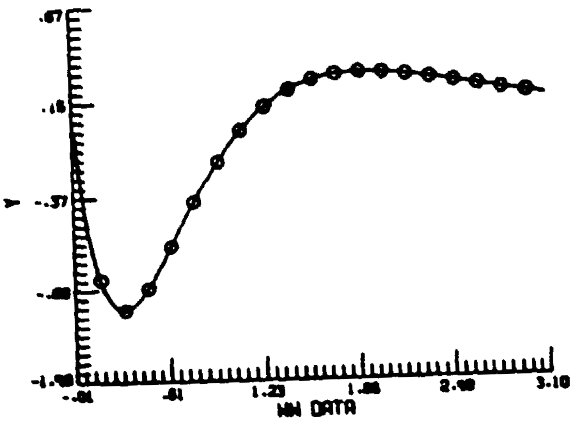

Figure 3.40. First step of knot elimination with 19 knots. No Noise. WW data.

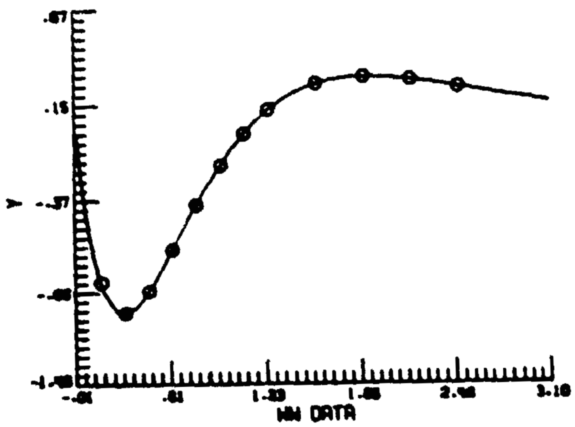

Figure 3.41. Final step of knot elimination from 19 knots. No noise. WW data.

ORIGINAL PAGE IS OF POOR QUALITY

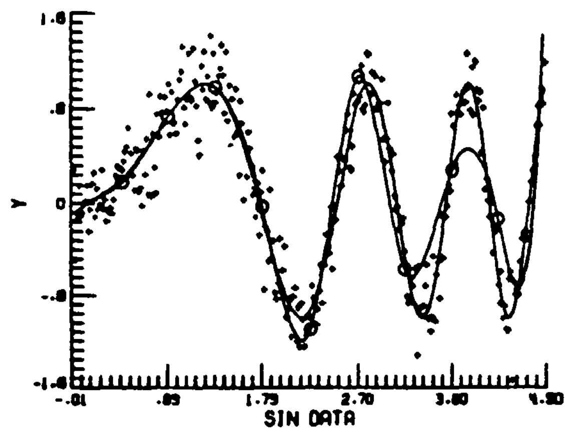

Figure 3.42. First step of knot elimination with 9 knots. True Function is sin (x^2).

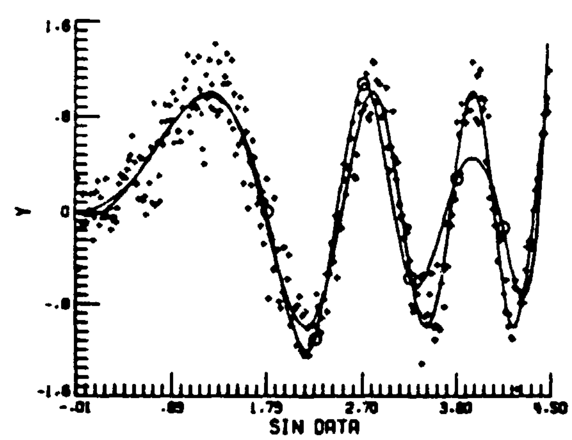

Figure 3.43. Final step of knot elimination from 9 knots. True function is sin (x^2).

Most statisticians have ready access to variable selection procedures, either in programs they have written themselves, or in widely available statistical software packages. Fitting splines through knot selection with these programs is a potential advantage of their use, which is realized only if good results are obtained. A summary of the results of using four such packages is given in Table 4.1: SAS (ref. 14), SPSS (ref. 15), MINITAB (ref. 16), and BMDP (ref. 17).

Table 4.1. Results of using variable selection techniques to fit splines with four statistical software packages.

		Stepwise		Stepdown	
		Indy	WW[1]	Indy	WW[1]
SAS	linear	✓	✓	✓	✓
	cubic	✓	✓	✓	✓
SPSS	linear	✓	✓	✓	✓
	cubic	✓	✓	✓	✓[2]
MINITAB	linear	✓	✓	✓	✓
	cubic	✓	X	X	X
BMDP[3]	linear	X	X	X	X
	cubic	X	X	X	X

[1] Selection pool of 19 interior knots.
[2] Numerical output has some inaccuracies, but overall results are correct.
[3] Tolerance cannot be made low enough to force entry of necessary terms.

In the case of stepwise procedures, accuracy was determined by comparing outputs for the various packages among themselves, while outputs for the stepdown procedures were compared with the FORTRAN B-spline knot elimination program. Results are surprisingly good considering the fact that the "+" function basis must be used. Entries marked with an "X" indicate failure to produce accurate results or, sometimes, any results at all due to high multicollinearity in the models or low tolerance, especially in stepdown.

The minimum tolerance allowed for SPSS, 10^{-12}, had to be used to force entry of some of the polynomial terms or to get results in stepdown. For BMDP, the tolerance of 0.01 for variable selection was not low enough to force entry of necess y terms to get results for any of the cases considered. As expected, less trouble was had with fewer knots (Indy data), lower egree (linear), and simpler models (stepwise). Stepdown gave accurate results in several cases even for a large number of knots, but there are limitations. For instance, computational problems were encountered by SAS for the cubic WW data with 39 knots. The final dels determined by stepwise and stepdown, however, were either identical or very similar. The occasional user of splines could thus safely rely on stepwise procedures from one of several packages to give good results.

Table 4.2 compares several spline-fitting methods: Wahba-Wold (WW), Smith-Smith (SS), and knot selection (KS). As the latter method may be implemented through several different computer programs, two statistical packages and the FORTRAN knot elimination routine are included. All methods give good results for the data examined, though as seen in earlier discussion, care must be taken when using the statistical packages, especially for stepdown. Their use of the "+" function makes them computationally inefficient and can cause severe problems. They are handy, however, for the occasional user as is the WW method which is available as an IMSL subroutine (ref. 18). The KS techniques depend on setting an α level for the hypothesis tests and specifying an initial pool of knots but are otherwise user independent. The WW method is "completely automatic," while the SS method depends on user application of the stopping criterion. The KS approach in general produces results which are sier to interpret.

Results from this section and from Section 3 show that splines fit by knot selection recover the underlying functions quite well and compare very favorably with the results of Wahba and Wold and improve upon those of Smith and Smith. Though somewhat simplistic, the knot selection approach provides an alternative to the method of cross-validation and offers a great computational savings. In addition, there is the possibility of analytic or physical interpretation in many modeling situations, an example of which is given in the next section.

Table 4.2. Comparison of spline-fitting techniques and software.

Desirable Properties	WW	SS	Knot Selection SAS	SPSS	B-Splines FORTRAN
Good results	✓	✓	✓	✓-	✓
Computational efficiency	✓	✓	X	X	✓
Diagnostics capabilities	X	X	✓	✓	✓
User independence	✓	✓-	✓-	✓-	✓-
Ease of interpretation	X	X	✓	✓	✓
Ease of use occasionally	✓	X	✓-[1]	✓	X

[1] SAS is available only on IBM-compatible machines.

5. SOME SPECIAL APPLICATIONS

Probably the most useful application of the KS technique is data-smoothing, and in Section 3 we saw several examples of recovering underlying functions from noisy data. A variation that is useful in simulation experiments is smoothing the sample quantile function (ref. 19). This function is a left-continuous step function defined as $Q(u) = x_{(i)}$ for $(i-1)/n < u \leq i/n$, where n is the sample size and $x_{(i)}$ is the i-th order statistic. Experimental conditions can be simulated by generating data which behaves like the original, and a smoothed sample quantile function provides a continuous distribution from which to draw the simulated data. An advantage of smoothing the sample quantile function, rather than its pseudo-inverse, the sample cumulative distribution function, is that the former always has domain $[0,1]$ regardless of the type of distribution.

Programming can thus be standardized, as, for example, in the determination of the original knot selection pool.

The KS technique is also useful in modeling. For example, stepwise regression has been applied successfully by Klein, Batterson, and Smith (ref. 20) to model flight data using splines. They use "+" function terms defined in the angle-of-attack variable in a Taylor series expansion of force and moment coefficients in order to model longitudinal motion of an airplane. One of their simple "spline-modified" Taylor series expansions of the vertical aerodynamic force coefficient C_z is given by

$$C_z = C_z(a)_{\substack{q'=0 \\ \delta_e=0}} + C_{z_q}(a)\, q' + C_{z_{\delta_e}}(a)\, \delta_e$$

where

$$C_z(a) = C_z(a = 0) + C_{z_a}\, a + \sum_{\ell=2}^{u_1} A_\ell (a - a_\ell)_+$$

$$C_{z_q}(a) = C_{z_q} + \sum_{\ell=2}^{u_2} B_\ell (a - a_\ell)_+^0$$

$$C_{z_{\delta_e}}(a) = C_{z_{\delta_e}} + \sum_{\ell=2}^{u_3} D_\ell (a - a_\ell)_+^0$$

and a is the angle of attack, q' is the nondimensional pitch rate, δ_e is the elevator deflection, $C_{z_a} = \partial C_z/\partial a$, $C_{z_q} = \partial C_z/\partial q'$, $C_{z_{\delta_e}} = \partial C_z/\partial \delta_e$. They then use stepwise regression to select terms, and thus knots, in the model. This spline representation preserves the concept of stability and control derivatives inherent in the usual Taylor series expansion of aerodynamic coefficients but has the advantage of providing a representation of C_z over an extended range of the angle of attack a. A global model over the observed range of a is thus obtained through the use of splines.

6. OTHER USES OF VARIABLE SELECTION PROCEDURES IN SPLINE REGRESSION

Thus far we have emphasized the use of variable selection to choose the number and location of knots. There are other possible, but perhaps less useful, "extensions" to spline regression of variable selection procedures based on polynomial or multiple regression models. In the latter cases, the purpose is to determine the polynomial degree and the important independent variables and interactions. This is accomplished by examining the contribution of individual terms in the model. With univariate spline models, however, there are several polynomial pieces, not just one, whose degrees may be examined, and, as seen previously, we may examine the importance of each knot. Also, one may wish to examine the continuity conditions at one or more breakpoints as in the example discussed by Smith (ref. 11). Thus, the complexity of the spline model over the polynomial model manifests itself in the greater number of ways the dimension of the spline parameter space may be altered. Splines in several variables present even more possible diversity since, for example, two-variable spline continuity occurs not across points but along lines connecting grid points.

While it might be nice to have a single software package which could perform any combination of these spline hypothesis tests, it is neither feasible nor desirable. The major reason is that variable order splines, i.e., splines with polynomial pieces of different degrees, have not been sufficiently researched by mathematicians to allow for the satisfactory construction in a general framework of a basis using either "+" functions or B-splines. Lowering or raising the degree of a single polynomial piece must be accomplished by applying restrictions to the model, and hypothesis tests must then use restricted least squares. In simple cases this may be straightforward (references 11 and 21), but in general the task is unmanageable. For example, the user is subject to hidden analytical errors as when the regression or hypothesis degrees of freedom are not equal to the number of restrictions because some restrictions are obtained automatically through linear combinations of others. While theoretically such dependencies can be checked, the usual methods would need some revision in the case of B-spline regres ion since hypotheses involve values of the fitted spline or its derivatives (ref. 12). In the case of the "+" function basis, most, but not all, of the individual terms are meaningful. However, the innocent yet indiscriminant selection or removal of terms through hypothesis tests can result in fits which are statistically valid yet nonsensical because they are uninterpretable in terms of polynomial degree or knot locations (ref. 11). Because of these various difficulties, it is reasonable to construct task-specific procedures.

The application of variable selection to knot selection, as in the examples in Section 3, is useful for smoothing data with a fixed order spline with maximum continuity conditions. In these cases the interest is not in the spline order but rather in determining the minimal number of knots deemed adequate to faithfully represent the data. Cubic splines are popular because of their low degree and second derivative continuity. The selective use of forward or backward algorithms in some statistical software packages using "+" functions (see Section 4), or the backward elimination FORTRAN program developed here using B-splines, may be used for this purpose.

Another possible "extension" of variable selection to splines is the determination of the polynomial degree while keeping the number and location of knots fixed, that is, not consider the knots as "variables" to be either entered or removed. Because of the difficulties with variable order splines discussed above, we must restrict ourselves to polynomial pieces of the same degree. Unfortunately, even further constraints are necessary for this version. The ideal situation would be to compare a maximally continuous $\left(C^{k-2}\right)$ k-th order spline, i.e., a k-th order spline with continuous f, $f^{(1)},\ldots f^{(k-2)}$, with a maximally continuous k-1-st order spline $\left(C^{k-3}\right)$. A formal test, however, is not possible. This can be easily seen by considering a specific example using the partial ordering of some spline models given in reference 11. Basis elements for C^0 and C^1 quadratic splines and for C^0 linear splines with one knot are shown in Fig. 6.1. A comparison of orders 3 and 2 (degrees 2 and 1) which retained maximum continuity conditions would require comparing the C^1 quadratic with the C^0 linear.

$$\begin{tikzcd} & \dfrac{C^0\ \text{quadratic}}{1,x,x^2,(x-t)_+^2,(x-t)_+} \arrow[dl] \arrow[dr] & \\ \dfrac{C^1\ \text{quadratic}}{1,x,x^2,(x-t)_+^2} & & \dfrac{C^0\ \text{linear}}{1,x,(x-t)_+} \end{tikzcd}$$

Figure 6.1. A partial ordering of some spline spaces.

Neither is a subspace of the other, however, so they cannot be formally compared (via testing). A solution of a sort is available if the C^0 quadratic and the C^0 linear are compared, since, as can be seen from the figure, the C^0 linear basis generates a subspace of the C^0 quadratic space.

In general, a test to compare spline orders can be made between splines of order k and $k-1$, both having continuity C^{k-3}. In the case of cubic splines, for example, we could allow continuity of the function and its first (but not second) derivative in order to determine whether the order could be reduced from 4 to 3 or increased from 3 to 4. Since a C^1 cubic has sufficient smoothness (at least to the eye), the procedure is not so objectionable. Considerably less satisfactory, however, are the cases for linear and quadratic splines. In comparing splines of order 3 and 2 as seen in Figure 6.1, the quadratic spline would be continuous but not its first derivative while in comparing splines of order 1 and 2, the linear spline would not even be continuous. Of course, the results of formal tests can be used in combination with informal comparison between SSE's of the models of interest to decide upon an acceptable model, and we recommend this approach.

A backward elimination FORTRAN program using B-splines has been developed for the purpose of reducing spline order using the nesting of some "sub-optimal" spaces as described above. Details for the appropriate B-spline hypothesis tests are given in reference 12. The listing, documentation and flowchart for the program are given in the Appendix, and we illustrate its use with the Indy data. While some statistical software packages could undoubtedly be used by defining "+" functions as in knot selection, no attempt was made to use them in this context. However, using the results of Section 4 as a guide, we surmise that several backward elimination procedures would be suspect while most forward selection algorithms should give fairly accurate results. Again, tolerance levels may have to be made small in order to force entry of certain terms.

Figure 6.2 gives the FORTRAN program output for stepdown order selection on the Indy data starting with a cubic spline (order 4) with the two knots in mid-WWI and WWII as in Section 3. The program compares splines of different order with the same continuity conditions, though other fits are given for information purposes. For this example, order reduction is made from cubic to quadratic to linear. Estimates of the B-spline coefficients and their standard errors are given for the spline of lowest order which can adequately fit the data, and the highest continuity conditions are imposed. For this case it is the C^0 linear.

A graphical display of these results is quite helpful, and Figure 6.3 shows a partial ordering of the relevant spline spaces along with hypothesis test results and SSE's from the program. The dotted lines indicate the stepdown comparisons we wish to make, while the solid lines indicate those we can actually make through formal comparisons (tests). The importance of user input into the variable selection process is becoming more widely recognized, and here especially, because the formal tests available are not exactly what we would like. Consequently, we recommend the use not only of the formal tests, but also of informal comparisons between SSE's (or MSE's) of competing models using a display such as Figure 6.3.

We illustrate this technique by going through Figure 6.3 step by step, and we will discover some interesting characteristics of splines along the way. We first observe that while a formal test is not possible between the C^2 cubic and the C^1 quadratic, it would not even be necessary since the C^1 quadratic has a smaller SSE than the C^2 cubic. A better fit is thus obtained with a lower degree! This phenomenon could never happen with polynomials, but such are the vagaries of splines. An informal comparison in

```
**********************************************************

THE SMOOTHEST SPLINE OF ORDER K= 4  WITH MAXIMUM CONTINUITY C 2
    HAS SSE=    385.29008218  AND MSE=      7.86306290

CAN ORDER K= 4  WITH SUB-MAXIMUM CONTINUITY C 1
    BE REDUCED TO ORDER K= 3  WITH MAXIMUM CONTINUTIY C 1 ?


YES.
FOR K= 4  AND C 1
    FTABLE VALUE =      4.00000000      OBSERVED F=        1.27322583
    SSE=    348.34966208  MSE=       7.41169494

**********************************************************

THE SMOOTHEST SPLINE OF ORDER K= 3  WITH MAXIMUM CONTINUITY C 1
    HAS SSE=    376.65994630  AND MSE=      7.53319893

CAN ORDER K= 3  WITH SUB-MAXIMUM CONTINUITY C 0
    BE REDUCED TO ORDER K= 2  WITH MAXIMUM CONTINUTIY C 0 ?


YES.
FOR K= 3  AND C 0
    FTABLE VALUE =      4.00000000      OBSERVED F=        3.69172563
    SSE=    368.45371260  MSE=       7.67611901

**********************************************************

THE SMOOTHEST SPLINE OF ORDER K= 2  WITH MAXIMUM CONTINUITY C 0
    HAS SSE=    453.46308846  AND MSE=      8.89153115

CAN ORDER K= 2  WITH SUB-MAXIMUM CONTINUITY C-1
    BE REDUCED TO ORDER K= 1  WITH MAXIMUM CONTINUTIY C-1 ?


NO.
FOR K= 2  AND C-1
    FTABLE VALUE =      4.00000000      OBSERVED F=      285.46925596
    SSE=    328.52409313  MSE=       6.70457333

PROCEDURE TERMINATES.   ***************************


PROCEDURE TERMINATES WITH L= 3; K= 2; C 0

 N         COEF             ST. ERR.
 1       73.61675883       2.19888373
 2       88.49577629       1.11724005
 3      114.97961700        .94655257
 4      157.47846182       1.10901397


********************** FURTHER INFORMATION ***********
    FOR K= 1  AND C-1
    SSE=   6070.37277252  MSE=     116.73793793
```

Figure 6.2. Output for order reductiion. Indy data.

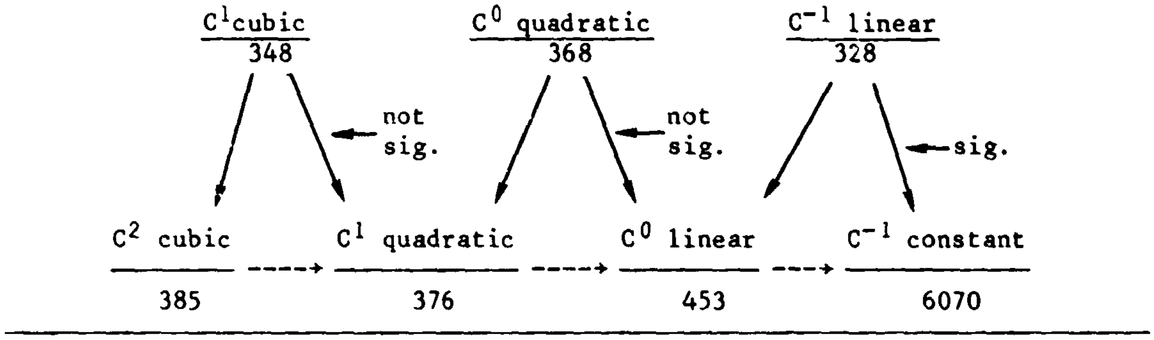

Figure 6.3. Partial ordering of spline spaces including SSE's and results of order reduction tests. Indy data.

in going from the C^1 quadratic to the C^0 linear reveals an increase of 77 in the SSE. While this increase cannot be formally judged insignificant, we may wish tc draw such a conclusion based on the results of the formal test which compares the C^0 quadratic with the C^0 linear: the larger increase of 85 is insignificant in that case. Having thus "safely" arrived at the C^0 linear, we must decide whether to further lower the order. The very large F value (285) from the program output which compares the C^{-1} linear and the C^{-1} constant splines reveals the importance of the linear trend. The big increase of 5742 in SSE from the C^{-1} linear to the C^{-1} constant is thus highly significant, and since the increase of 5617 from the C^0 linear to the C^{-1} constant (the desired comparison) is only slightly smaller, we conclude that the use of a constant spline fit is inadvisable.

7. SPLINES IN SEVERAL VARIABLES

A mathematical theory for splines in several variables is still developing, and a "satisfactory" basis even in two variables has not been found. However, tensor products of either "+" functions or B-splines can be used to form a spline basis in several variables. While a tensor product basis is somewhat clumsy and its interpretation difficult, we explain here some theoretical aspects of its use for the two variable case and give an example.

As in the example in Section 5, a spline-modified Taylor series expansion can be used to model aerodynamic force and moment coefficients. This time, however, Klein and Batterson (ref. 22) use splines in two variables, the angle of attack a and the sideslip angle b, to approximate the lateral force coefficient and the rolling and yawing moment coefficients. They use the yawing moment coefficient C_n as a typical example, and C_n can be expressed as

$$C_n = C_n(a,b)_{\delta_a = \delta_r = 0} + C_{n_p}(a)p' + C_{n_r} r'$$

$$p' = r' = 0 \tag{7.1}$$

$$+ C_{n_{\delta_a}}(\alpha)\,\delta_a + C_{n_{\delta_r}}(a)\,\delta_r$$

where p and r are the rolling and yawing velocity and δ_a and δ_r are the aileron and rudder deflection. They approximate the function $C_n(a,b)$ by

$$C_n(a,b) = C_o + C_1 b + \sum_{i=1}^{\ell 1} (A_{oi} + A_{1i} b)(a - a_i)_+^0$$

$$+ \sum_{j=1}^{\ell 2} B_{oj}(b = b_j)_+ + \sum_{i=1}^{\ell 1} \sum_{j=1}^{\ell 2} D_{ij}(b - b_j)_+ (a - a_i)_+^0 \tag{7.2}$$

while the remaining functions in (7.1) are approximated by splines in a alone. Results from a stepwise regression using these terms are not as good as in the one-variable case, and some fine-tuning remains.

From a theoretical point of view, the tensor product basis does not have the nice interpretation of knots and continuity constraints as in the one-variable case, even using "+" functions. There is, however, a one-to-one correspondence between two-variable "+" function terms and grid points, or nodes, and for this reason, we use the term node basis to refer to tensor products of the "+" function basis. As before, we use right-continuous "+" functions so that 0^0 is 1. Tensor products of B-splines may also be used to construct a basis for splines in two variables, and we shall see that the same advantages and disadvantages of the one-variable case carry over.

We discuss the simplest two-variable case in some detail: first order splines, i.e. step functions. Their application is somewhat limited, but there are several reasons for their detailed consideration. First and foremost, splines in two variables are difficult to envision and manipulate, and consideration of the simplest case, namely constants, is thus highly desirable. Second, as seen in the example above and in Section 5, the estimation of aerodynamic force and moment coefficients using a spline-modified Taylor series expansion reveals the importance of using constants from both interpretative and numerical points of view. Finally, the two-dimensional cumulative distribution function is a first order spline in two variables. Thus, the constant case, while limited, has already shown its usefulness.

We first discuss the node basis by way of example. Suppose breakpoints in the x variable occur at x_1, x_2, x_3 and in the y variab'e at y_1 and y_2 for data in $x_0 \le x < x_4$ and $y_0 \le y < y_3$. A "+" function basis of order 1 in the x variable is $(x - x_0)_+^0, \ldots, (x - x_3)_+^0$ and in the y variable is $(y - y_0)_+^0, \ldots, (y - y_2)_+^0$. The tensor product basis is formed by taking all the $4 \times 3 = 12$ products $(x - x_i)_+^0 (y - y_j)_+^0$, $i = 0, \ldots, 3$; $j = 0, \ldots, 2$. Each basis element in the two variables is thus a plane of height one bounded below by the line $y = y_j$ and on the left by the line $x = x_i$. Its support is thus a quadrant of a sort ( ). We call the intersection of these boundary lines, the corner of the quadrant, a node, denoted $*ij$. Figure 7.1 shows the relevant grid and nodes. Through any

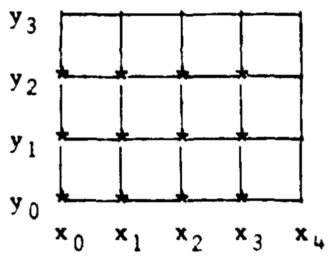

Figure 7.1. Nodes for a tensor product of "+" functions.

variable selection procedure, a model may be found whose terms are a subset of the 12 basis elements. Such a selection might result, for example, in the nodes shown in Figure 7.2 with the statistical model

$$f(x,y) = \beta_{00}x_{0}y_{0+} + \beta_{02}x_{0+}y_{2+} + \beta_{11}x_{1+}y_{1+}$$

$$+ \beta_{21}x_{2+}y_{1+} + \beta_{22}x_{2+}y_{2+} + \beta_{32}x_{3+}y_{2+} + \varepsilon,$$

where $x_{i+}y_{j+}$ is an abbreviation for $(x - x_i)_+^0 \, (y - y_j)_+^0$. We saw earlier the application of this technique to aerodynamic modeling.

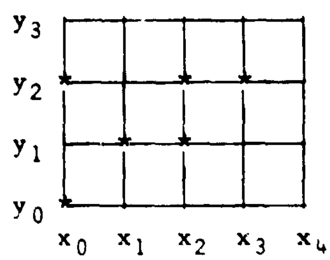

Figure 7.2. Nodes resulting after variable selection on a tensor product of "+" functions.

For splines of higher order, the same principles apply in forming the basis elements: they are the tensor product of one-variable "+" functions. Knot multiplicities in one variable result in node multiplicities in several variables. The absence or presence of a node or node multiplicity corresponds to the absence or presence of a certain basis element. There is thus some carry-over from the one-variable case in interpreting the role that basis elements play, and also in the fact that standard variable selection software may be used. The major drawback of this basis, as in the one-variable case, is computational. The basis elements do not have small support, so that roundoff errors get worse as computations increase.

The computational difficulties present in the node basis lead to consideration of tensor product B-splines. While the formulation of the latter basis is straightforward, its interpretation and use in model selection through hypothesis tests are not. The polynomial degree and importance of knots in modeling are considerations that carry over from one to several variables, and unfortunately, so do their difficulties when using B-splines.

To compare differences in the two-variable case between the node basis and B-spline basis, we consider a simple grid with nodes indicated (*) in Figure 7.3.

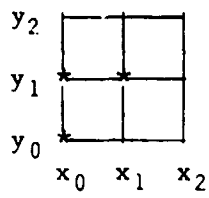

Figure 7.3. Nodes for model (7.3).

The statistical model for first order splines is thus

$$f(x,y) = \beta_{00}x_{0_+}y_{0_+} + \beta_{01}x_{0_+}y_{1_+} + B_{11}x_{1_+}y_{1_+} + \varepsilon. \tag{7.3}$$

The function is a "true" spline in both variables except when $y\varepsilon[y_0\ y_1)$, for then f is constant over $[x_0,x_2)$. If this model is represented with B-splines, each cell i is the support of a right-continuous plane which has height 1. Using the notation $B_i(x,y)$ for the basis element for each cell i, the model may be written

$$f(x,y) = \sum_{i=1}^{4} \beta_i B_i(x,y) + \varepsilon \quad \text{subject to} \quad \beta_1 = \beta_3.$$

This B-spline model is somewhat more complicated than the "+" function basis in its representation because of the model restrictions. It is also not obvious how to interpret the B-spline coefficients in terms of the presence or absence of nodes.

These simple examples illustrate that the "+" function terms are identifiable and meaningful on a grid as nodes, just as they correspond to knots in the one-variable case. They thus hold an advantage over the tensor product of B-splines from an interpretative point of view. As B-splines hold the computational edge, however, it would be desirable to identify the linear combinations of B-splines which correspond to the presence or absence of nodes. The interpretation and use of tensor product splines of higher order is more difficult and remains to be examined in detail.

8. SUGGESTIONS FOR FURTHER WORK

There are potential research areas for both the univariate and multivariate cases. In the univariate case, an efficient stepwise computer routine using B-splines could be developed. This would give the user the choice of forward and backward procedures with a computationally efficient basis. The use of knot selection to fit data with loops could be investigated, and approaching the problem using the parametric technique of Smith, Price, and Howser (ref. 23), seems feasible. The successful use of splines in two variables has already been demonstrated (Section 7), but further work remains such as investigating fits to known underlying functions like we have done in the one-variable case. Two-dimensional pictures in this case would be most helpful. Also, while the multivariate mathematical theory is still developing, interpretation of tensor-product bases from a statistical perspective could continue from that begun in Section 7.

REFERENCES

1. de Boor, Carl: A Practical Guide to Splines. Springer-Verlag, 1978.

2. Reinsch, C.H: Smoothing by Spline Functions. Numer. Math., Vol. 10, 1967, pp. 177-183.

3. Wahba, G.; and Wold, S.: A Completely Automatic French Curve: Fitting Spline Functions by Cross Validation. Comm. Statist., Vol. 4, No. 1, 1975, pp. 1-17.

4. Wahba, Grace: How to smooth curves and surfaces with splines and cross-validation. Technical Report No. 555, University of Wisconsin, 1979.

5. Wahba, Grace; and Windelberger, James: Some new mathematical methods for variational objective analysis using splines and cross-validation. Technical Report No. 578, University of Wisconsin, 1979.

6. Quandt, R.E.: Tests of Hypotheses that a Linear Regression System Obeys Two Separate Regimes. J. Amer. Stat. Assoc., Vol. 55, 1960, pp. 324-330.

7. Smith, A.F.M.; and Cook, D.G.: Straight Lines with a Change-point: A Bayesian Analysis of Some Renal Transplant Data. Appl. Statist., Vol. 29, 1980, pp. 180-189.

8. Ertel, James E.; and Fowlkes, Edward B.: Some Algorithms for Linear Splines and Piecewise Multiple Linear Regression. J. Amer. Stat. Assoc., Vol. 71, 1976, pp. 640-648.

9. Smith, Patricia L.; and Smith, Philip W.: Empirical Stopping Rules for Data Fitting by Splines. Proceedings of the 1979 Army Numerical Analysis and Computers Conference, ARO Report 79-3, 1979, pp. 449-456.

10. Agarwal, Girdhar G.; and Studden, W.J.: Asymptoti Integrated Mean Square Error Using Least Squares and Bias Minimizing Splines. Ann. Stat., Vol. 8, 1980, pp. 1307-1325.

11. Smith, Patricia L.: Splines as a Useful and Convenient Statistical Tool. The American Statistician, Vol. 33, 1979, pp. 57-62.

12. Smith, Patricia L.: Hypothesis Testing in B-spline Regression. Comm. Statist., Vol. 11, 1982, pp. 143-157.

13. Poirier, Dale J.: Piecewise Regression Using Cubic Splines. J. Amer. Stat. Assoc., Vol. 68, 1973, pp. 515-524.

14. SAS User's Guide. SAS Institute, Inc., P.O. Box 10066, Raleigh, NC 27605, 1979.

15. SPSS UPDATE 7-9. C. Hadlai Hull and Norman H. Nie editors. McGraw-Hill, 1981.

16. Ryan, T.A., Jr.; Joiner, B.L.; and Ryan, B.F.: MINITAB Reference Manual, Duxbury Press, 1981.

17. BMDP-79. Biomedical Computer Programs. W.J. Dixon and M.B. Brown editors. University of California Press, 1979.

18. IMSL Reference Manual Edition 9. IMSL, Inc., 7500 Bellaire Blvd., Houston, TX 77036-5085, 1982.

19. Schiess, James R.: Personal Communication. NASA Langley Research Center, Hampton, VA 23665, 1982.

20. Klein, V.; Batterson, J.G.; and Smith, P.L.: On the Determination of Airplane Model Structure From Flight Data. Presented at the 6th IFAC Symposium on Identification and System Parameter Estimation, Arlington, . June 7-11, 1982.

21. Gallant, A.R.; and Fuller, Wayne A.: Fitting Polynomial Regression Models Whose Join Points Have to be Estimated. J. Amer. Stat. Assoc., Vol. 68, 1973, pp. 144-147.

22. Klein, V.; and Batterson, J.G.: Determination of Airplane Aerodynamic Parameters From Flight Data at High Angles of Attack. Presented at the 13th Congress of the International Council of the Aeronautical Sciences, Seattle, WA, Aug. 22-27, 1982.

23. Smith, Robert E., Jr.; Price, Joseph M.; and Howser, Lona M.: A Smoothing Algorithm Using Cubic Spline Functions. NASA TN D-7397, 1974.

APPENDIX

Program Documentation

Two FORTRAN programs have been written which adapt stepdown procedures to B-spline regression. One program is for knot elimination while the other is for reducing the spline order. Theoretical details and appropriate references are given in Sections 2 and 6. The programs are written in FORTRAN 5 and have been implemented on both the ODU DEC-10 and the NASA/Langley CDC Cyber computers. Notation is patterned after that of de Boor (ref. 1), and definitions of parameters are given in the subroutine VL2NT, the second subroutine called. All necessary input is read in or specified in subroutine DAT: the data, sample size NDATA, (initial) spline order K = degree +1, (initial) interior breakpoints and endpoints BREAK(•), number of continuity conditions V(•) at the breakpoints, number of intervals L = # interior breakpoints +1, and tabled F value to be used in hypothesis tests. For equal spacing, the breakpoints and continuity conditions are most easily specified through a DO loop. Variables are dimensioned by one of three parameters (defined in comment statements) which are specified in the PARAMETER statement at the beginning of the main program.

Data must be interior to [BREAK(1), BREAK(L+1)]. For the Indy data, X max = 61, so we arbitrarily set BREAK(1) = 0 and BREAK(L+1) = 62. V(I) is the number of continuity constraints at BREAK(I). For example, V(1) = 0 means that the spline is discontinous at BREAK(1) while V(2) = 3 means there are 3 contiguous continuity conditions on the spline f at BREAK(2), i.e., f, f', and f" are all continuous at BREAK(2). Note that V(I) must be less than or equal to K-1 in order to have a "true" spline, not a polynomial, across BREAK(I). We always set V(1) = 0, though only for "symmetry" in the endpoint conditions, and V(L+1) need not be specified since it is never used nor referred to.

The subroutine FLAG is designed to catch user input errors which would otherwise cause the program to terminate abnormally or give inaccurate results which may or may not be obvious to the user. Sample output detecting errors in the input information of the Indy data is shown in Figure A.1.

ORIGINAL PAGE IS
OF POOR QUALITY

```
THE ORDER K =  4

THE # INTERVALS _ =  3

THE DIMENSION N =   5

    BREAKPOINTS    CONTINUITY CONDITIONS
     5.00000000               0
    33.50000000               4
     7.50000000               3
    62.00000000


     T              INDEX
     5.00000000       1
     5.00000000       2
     5.00000000       3
     5.00000000       4
                              KEND(  1)=    4
     7.50000000       5
                              KEND(  2)=    4
    62.00000000       6
    62.00000000       7
    62.00000000       8
    62.00000000       9

BREAKPOINTS MUST BE STRICTLY INCREASING.
 BREAKPOINT     33.50000000  IS NOT LESS THAN BREAKPOINT       7.50000000

THE NUMBER OF CONTINUITY CONDITIONS MUST BE STRICTLY
     LESS THAN THE SPLINE ORDER K.     V( 2)= 4
     AT BREAKPOINT      33.50000000  IS TOO LARGE.

X VALUE OUT OF RANGE.
 X(    1)=       1.00000000 IS NOT IN THE RANGE BREAK(1)=       5.00000000
    TO BREAK(LAST)=     62.00000000


**************************************************
STEPDOWN CANNOT PROCEED.  PROGRAM ABORTS.
```

Figure A.1. Sample output detecting input errors. Indy data.

Several lines in the programs are for plotting only. These are calls to the CDC system subroutines PSEUDO, INFOPLT, and CALPLT and the DO loop 10 which calculates the spline values at the knots.

For the knot elimination routine, input data and subsequently calculated information are printed by means of subroutines DAT1 and OUTNTS. This includes data values, spline order, number of intervals, dimension of the spline space, and knots. At each step of the procedure as indicated by the number of intervals L, the F-ratios for the importance of each breakpoint are given along with the SSE and MSE. If a breakpoint can be eliminated, it is specified and the procedure continues to stepdown. If no breakpoint can be eliminated, the resulting number of intervals and spline order are given along with a list of the values of the B-spline coefficients and their standard errors. Sample output appears in Figure 3.1, p. 8, in Section 3.

As in the knot elimination program, the subroutines DAT1 and OUTNTS of the order reduction routine print input data and subsequently calculated information. In addition, at each step, the printout gives the SSE's and MSE's for two splines of order K, one with continuity C^{K-2} and the other with continuity C^{K-3}. The hypothesis test is described in words with the results of the F test indicated. When further order reduction is not possible, estimates of the B-spline coefficients and their standard errors are given for the spline of lowest acceptable order with highest continuity imposed. Additional information is given by including the SSE and MSE of the next lowest order spline. Sample output for the Indy data appears in Figure 6.2, p. 41.

Flowcharts are given in Figures A.2 to A.3 followed by the program listings. A full listing of the knot elimination program from a CDC is given, including the subroutines of de Boor (ref 1) that are used. For the order reduction program we list only the main program and the subroutine SSHYP2, a variation of SSHYP appearing in the first program.

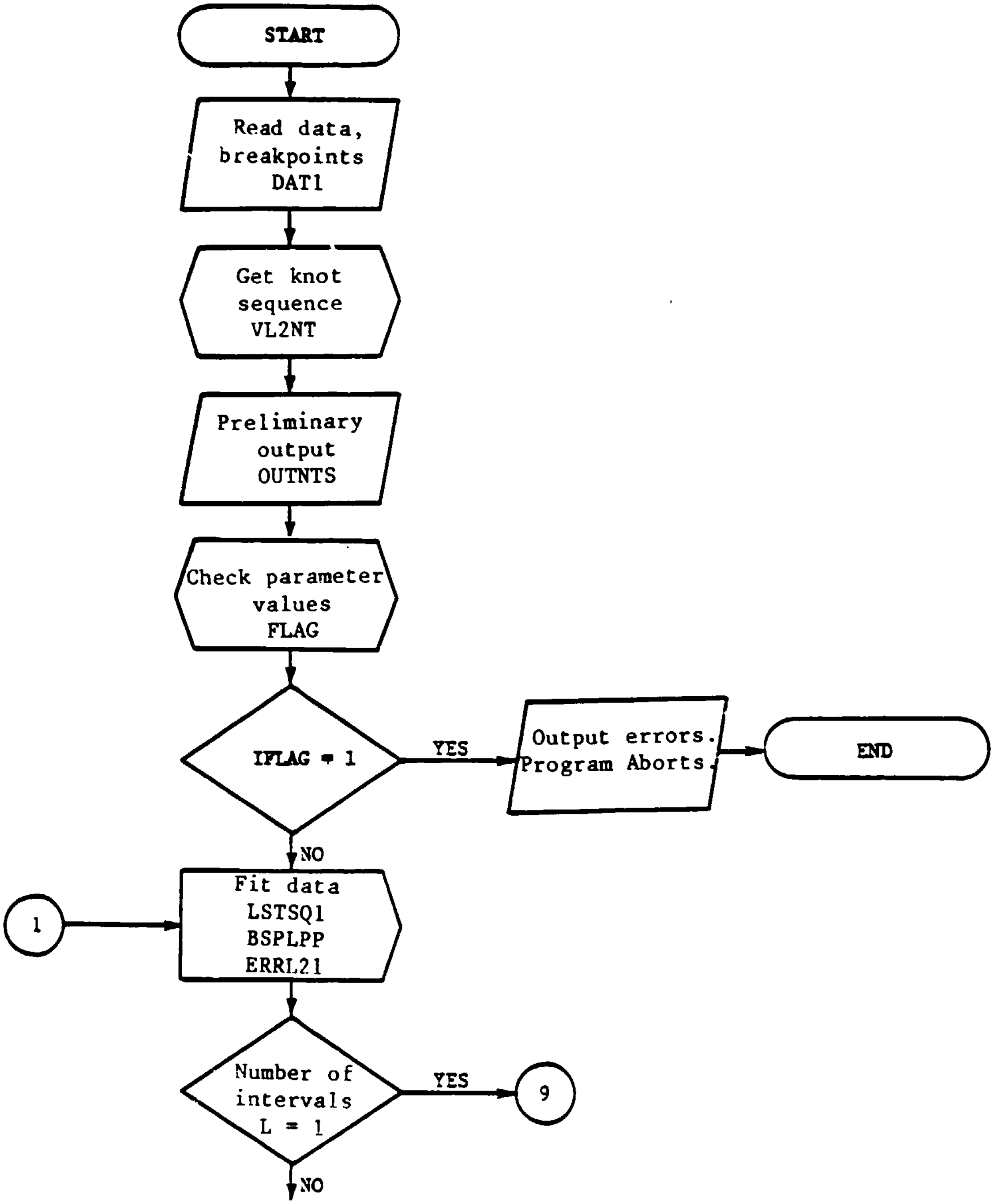

Figure A.2. Flowchart for knot elimination program.

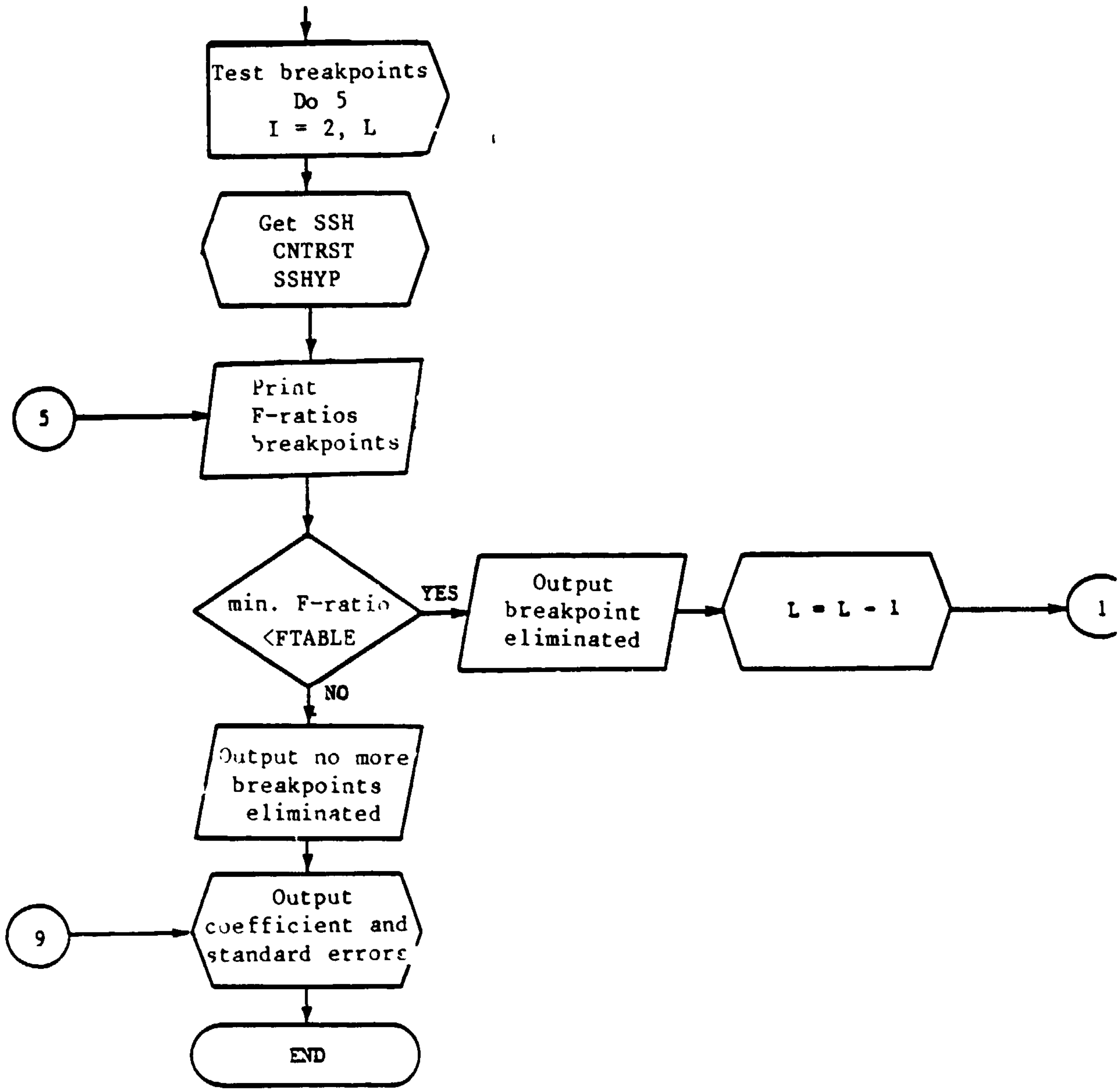

Figure A.2. (concluded).

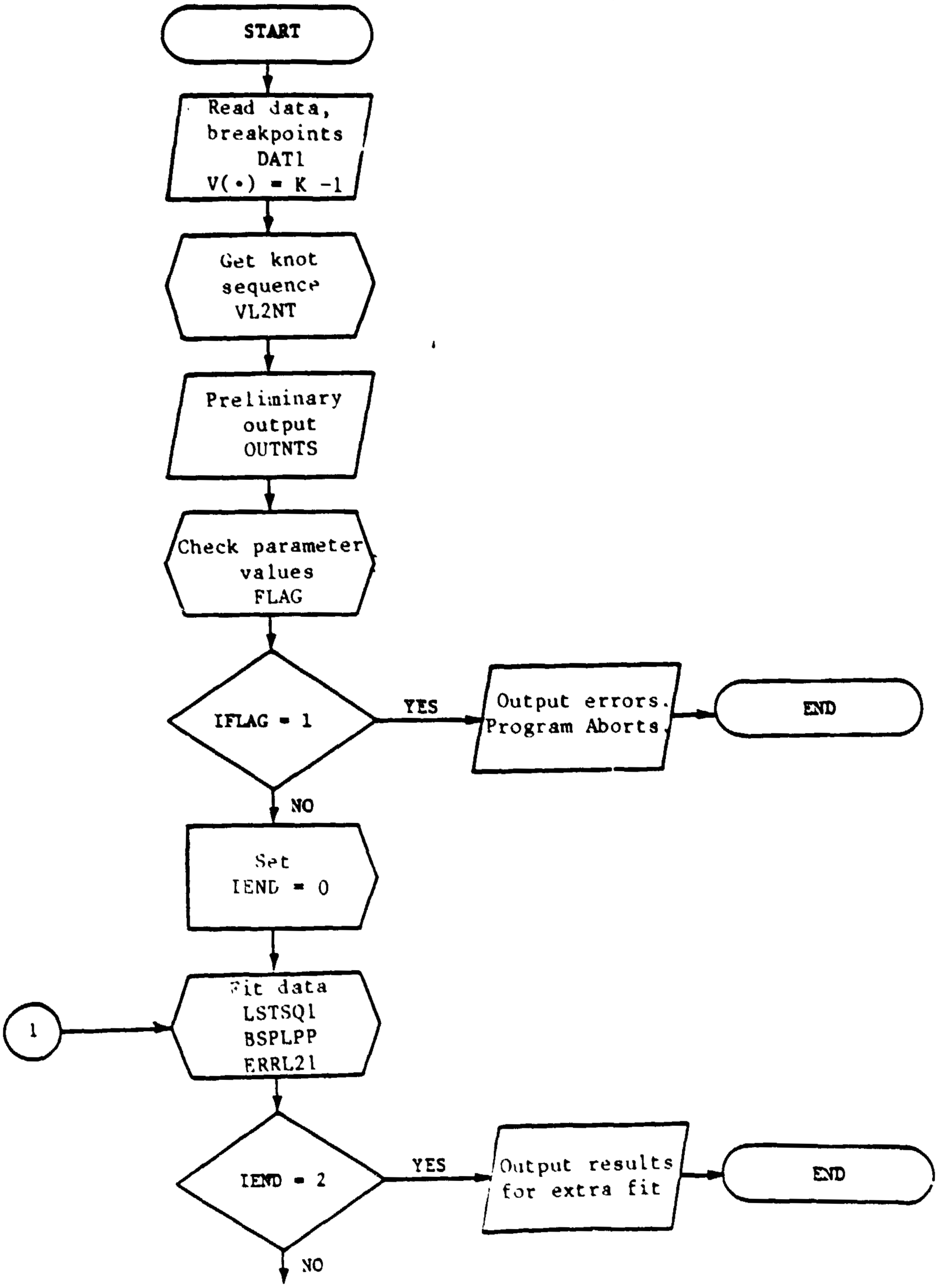

Figure A.3. Flowchart for order reduction program.

ORIGINAL PAGE IS
OF POOR QUALITY

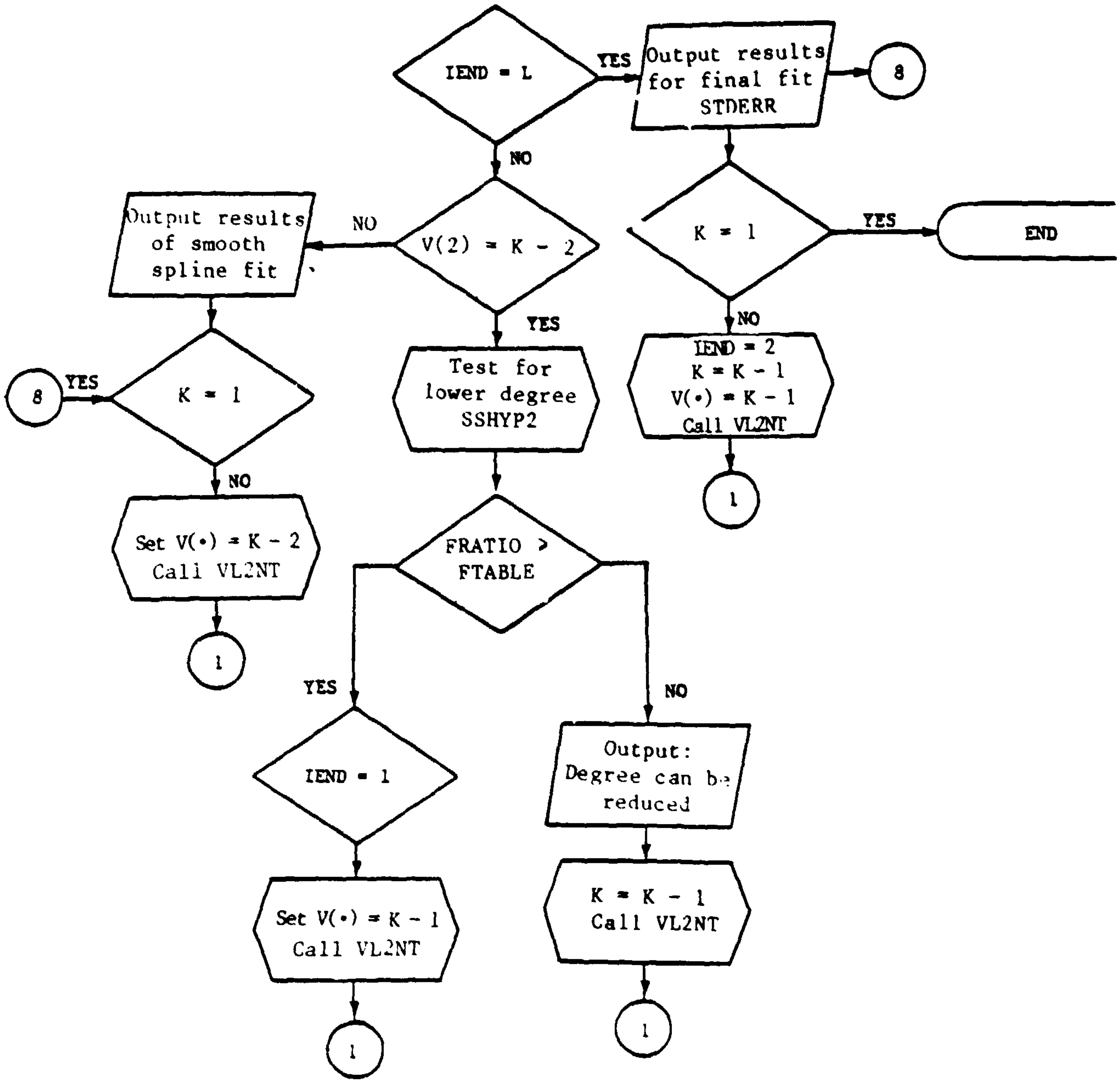

Figure A.3. (concluded).

OF POOR QUALITY

KNOT ELIMINATION PROGRAM LISTING

```
      PROGRAM XPLOT(INPUT,OUTPUT,TAPE6=OUTPUT,TAPE20,TAPE21,TAPE22)
C STEPDOWN FOR BREAKPOINT ELIMINATION FOR FIXED ORDER K.
C THE FUNCTION AND ITS FIRST K-2 DERIVATIVES MUST BE CONTINUOUS.
C
C NDMAX IS AT LEAST THE SAMPLE SIZE, NDATA.
C NMAX IS AT LEAST N.  WITH MAXIMUM CONTINUITY CONDITIONS,
C    N=L+K-1.  WITH NO CONTINUITY CONDITIONS, N=L*K.
C KTNMAX IS AT LEAST K*N.
C
C
      PARAMETER(NMAX=100,NDMAX=200,KTNMAX=2000)
      REAL BCOEF(NMAX), Q(KTNMAX), DIAG(KTNMAX), T(NDMAX)
     *   ,DCOEF(NMAX), BRT(NMAX), BLF(NMAX),F(NDMAX)
     *   ,CTRAST(NMAX),AA(NMAX,NMAX), ERROR(NDMAX)
     *   , MSE, MSH, SE(NMAX),FRATIO(NMAX)
     *   ,BB(NMAX,NMAX), LINV(NMAX,NMAX),FB(NMAX)
      INTEGER ERRDF,HDF,V,KEND(NMAX)
      COMMON /DATA/ NDATA, X(NDMAX), Y(NDMAX), FTABLE
      COMMON /APPROX/ BREAK(NMAX), COEF(KTNMAX), L, K, V(NMAX)

      ICOUNT=0
C ENTER DATA
      CALL DAT1(ICOUNT)

C GET THE KNOT SEQUENCE
      CALL VL2NT(BREAK,L,K,V,T,N,KEND)
C PRELIMINARY OUTPUT
      CALL OUTNTS(BREAK,V,L,T,N,K,KEND)

CHECK INPUT DATA
      IFLAG=0
      CALL FLAG(IFLAG,N)
      IF(IFLAG .EQ. 1) GO TO 25

      CALL PSEUDO

C TEST FOR CONTINUOUS K-1-ST DERIVATIVE AT EACH KNOT
      JDERIV=K-1
    1 FMIN=FTABLE

      LM1=L-1

C GET THE LEAST SQUARES FIT, I.E., THE B-SPLINE COEFFICIENTS
      CALL LSTSQ1(T,N,K,Q,DIAG,BCOEF)
C          LSTSQ1 CALLS BSPLVB, BCHFAC, AND BCHSLV

C GET SSE AND MSE
      ERRDF=NDATA-N
      CALL BSPLPP(T,BCOEF,N,K,DIAG,BREAK,COEF,L)
      CALL ERRL21(F,ERROR,ERRDF,SSE,MSE)
```

```
C       ERRL21 CALLS PPVALU WHICH CALLS INTERV
      LP1=L+1
      DO 10 I=1,LP1
         FB(I)=PPVALU(BREAK,COEF,L,K,BREAK(I),0)
   10    CONTINUE
C*****************   PLOTS   ******************************
      CALL INFOPLT(0,NDATA,X,1,F,1,0.,62.,74.,150.,1.,9,
     *             9HINDY DATA,1,1HY,0,5.,4.,.75,.75)
      CALL INFOPLT(0,NDATA,X,1,Y,1,0.,62.,74.,150.,1.,9,
     *             9HINDY DATA,1,1HY,22,5.,4.,.75,.75)
      CALL INFOPLT(1,LM1,BREAK(2),1,FB(2),1,0.,62.,74.,150.,1.,9,
     *             9HINDY DATA,1,1HY,1,5.,4.,.75,.75)


      IF(L .NE. 1) GO TO 12
      WRITE(20,11) SSE,MSE
   11 FORMAT(//' SSE=',F16.8,5X,'MSE=',F16.8)

                    GO TO 9

C TEST IMPORTANCE OF EACH BREAKPOINT
   12 WRITE(20,2) L,FTABLE,SSE,MSE
    2 FORMAT(///' L=',I3,5X,'F-TABLE VALUE IS',F16.8,//
     *     ' SSE=',F16.8,5X,'MSE=',F16.8//
     *   ' F-RATIOS ARE:     BREAKPOINTS ARE')

    3 DO 5 II=2,L
         ID=II
         CALL CNTRST(ID,JDERIV,N,K,L,T,BREAK,KEND,BRT,BLF,DCOEF
     *      ,CTRAST)
C                     CNTRST CALLS BCONT
         CALL SSHYP(BCOEF,CTRAST, Q,K,N,BRT,VAR SSH,MSH,HDF)
C                  SSHYP CALLS FORSUB
      FRATIO(II)=MSH/MSE
      WRITE(20,4) FRATIO(II),BREAK(II)
    4 FORMAT(2F16.8)
      IF(FRATIO(II) .GE. FMIN)    GO TO 5
      FMIN=FRATIO(II)
         KNOT=II
    5    CONTINUE

      IF (FMIN .LT. FTABLE)     GO TO 7
      WRITE(20,6)
    6 FORMAT(/' NO BREAKPOINT CAN BE ELIMINATED')
                                GO TO 9
    7 FMIN=FRATIO(KNOT)
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      WRITE(20,8) BREAK(KNOT)
    8 FORMAT(///' BREAKPOINT',F7.3,' IS ELIMINATED')


C RELABEL KNOT SEQUENCE T AS WELL AS BREAK, KEND, AND V
      CALL REKNOT(KEND,KNOT,N,K,L,T,V,BREAK)

                                        GO TO 1

C PRINT RESULTING COEFICIENTS, STANDARD DEVIATIONS, AND F-VALUES
    9 CALL STDERR(Q,BCOEF,K,N,L,MSE,DIAG,AA,SE,LINV)
C                 STDERR CALLS BCHINV AND MATVEC.


      CALL CALPLT(0.,0.,999)


   25 STOP
      END
C INDY DATA
       SUBROUTINE DAT1(ICOUNT)
COMMON STATEMENTS /DATA/ AND /APPROX/ ARE USED.
C
C THIS SUBROUTINE READS IN THE DATA AND GIVES THE NUMBER AND
C PLACEMENT OF THE KNOTS FOR THE FITTED SPLINE.

      PARAMETER (NMAX=100, NDMAX=200, KTNMAX=2000)
      INTEGER  V
      REAL  Y, X
      COMMON / DATA / NDATA, X(NDMAX), Y(NDMAX), FTABLE
      COMMON / APPROX / BREAK(NMAX), COEF(KTNMAX), L, K
     *                  , V(NMAX)

      NDATA = 55
      WRITE(20,5)
    5 FORMAT(' INDY DATA'//' YEAR     Y      X')
      DO 1 I=1,NDATA
         READ(21,4) YEAR,Y(I),X(I)
    4    FORMAT(I4,1X,F7.3,1X,F2.0)
         WRITE(20,2) YEAR,Y(I),X(I)
    2    FORMAT(I4,1X,F7.3,1X,F3.0)
    1    CONTINUE

C GIVE THE ORDER K AND NUMBER OF INTERVALS L
      K = 4
      L = 3
      FTABLE = 8.00

C GIVE THE BREAKPOINTS AND CONTINUITY CONSTRAINTS
      BREAK(1) =  0.
      BREAK(2) =  7.5
```

```
      BREAK(3) = 33.5
      BREAK(4) = 62.
      V(1) =0
      V(2) = 3
      V(3) = 3
                                            RETURN
      END
      SUBROUTINE VL2NT(BREAK,L,K,V,T,N,KEND)
COMPUTES THE KNOT SEQUENCE T AND DIMENSION N FROM THE BREAKPOINT
C SEQUENCE BREAK, GIVEN THE SPLINE ORDER K, THE NUMBER OF INTER-
C VALS L, AND THE NUMBER OF CONTINUITY CONDITIONS V(I) AT BREAK
C (I).
C
C****** I N P U T ******
C BREAK (1),...,BREAK(L+1)....THE BREAKPOINT SEQUENCE.
C L....THE NUMBER OF INTERVALS.
C K....THE ORDER OF THE SPLINE.
C V(2),...,V(L)....THE NUMBER OF CONTINUITY CONSTRAINTS AT
C                  BREAK(2),...,BREAK(L).
C
C****** O U T P U T ******
C T(1),...,T(N+K)....THE KNOT SEQUENCE.
C N....THE DIMENSION OF THE SPLINE SPACE OF ORDER K.
C KEND(I)....THE INDEX OF THE LARGEST KNOT EQUAL TO BREAK(I)
C
C****** M E T H O D ******
C THE FIRST K KNOTS ARE SET EQUAL TO BREAK(1). THE KNOTS ARE
C THEN SEQUENCED SO THAT K - V(I) KNOTS ARE AT BREAK(I) WITH
C KEND(I) EQUAL TO THE INDEX OF THE LARGEST KNOT AT BREAK(I).
C N IS SET EQUAL TO KEND(L) AND THE LAST K KNOTS T(N+1),...
C T(N+K) ARE SET EQUAL TO BREAK(L+1).
C
      INTEGER K,L,N,I,V(1),J,ISTART,ISTOP, KEND(1)
      REAL BREAK(1), T(1)
C SET THE FIRST K KNOTS EQUAL TO BREAK(1).
      DO 1 I = 1, K
    1    T(I) = BREAK(1)
C
C FIND THE INDEX KEND(I) OF THE LARGEST KNOT EQUAL TO BREAK(I).
      KEND(1) =  K
      DO 2 I = 2, L
    2    KEND(I) = KEND(I-1) + K - V(I)
C
C SET T(KEND(I-1) + 1) =...= T(KEND(I)) = BREAK(I).
      DO 10 I = 2, L
         ISTART = KEND(I-1) +1
         ISTOP = KEND(I)
         DO 11 J = ISTART, ISTOP
   11       T(J) = BREAK(I)
   10 CONTINUE
      N = KEND(L)
C
C SET THE LAST K KNOTS EQUAL TO BREAK(L+1).
```

```
      DO 20 I = 1, K
   20    T(N+I) = BREAK(L+1)
      RETURN
      END
      SUBROUTINE OUTNTS(BREAK,V,L,T,N,K,KEND)
C THIS SUBROUTINE IS FOR OUTPUTING ONLY.  IT OUTPUTS ALL
C CALLING ARGUMENTS A F T E R  VLZNT HAS BEEN CALLED.
C
C * * * * * *  I N P U T   AND  O U T P U T * * * * * *
C K....THE SPLINE ORDER
C L.... THE NUMBER OF INTERVALS
C N....THE DIMENSION OF THE SPLINE SPACE
C BREAK(1),...,BREAK(L+1)....THE BREAKPOINT SEQUENCE
C V(1),...V(L)....THE NUMBER OF CONTINUITY CONSTRAINTS AT
C                 BREAK(1),...BREAK(L)
C T(1),...T(N)....THE KNOT SEQUENCE
C KEND(1),...,KEND(L)....INDEX OF THE LARGEST KNOT EQUAL TO
C                        BREAK(1),...,BREAK(L)
C
      DIMENSION T(1), KEND(1), BREAK(1)
      INTEGER V(1)
      WRITE(20,40) K, L, N
   40 FORMAT(///' THE ORDER K = ', I3//' THE # INTERVALS L =', I3,
     *      //' THE DIMENSION N = ', I3)
      WRITE(20,41)
   41 FORMAT(///'      BREAKPOINTS', T20, ' CONTINUITY CONDITIONS')
      DO 45 J = 1, L
   45 WRITE(20,42) BREAK(J), V(J)
   42 FORMAT(F16.8,T30,I3)
      LP1 = L + 1
      WRITE(20,43) BREAK(LP1)
   43 FORMAT(F16.8)
      WRITE(20,8)
    8 FORMAT(///'       T              INDEX')
      NPK = N + K
      ICOUNT = 1
      INDEX = 1
      WRITE(20,5) T(1), INDEX
    5 FORMAT(F16.8, 5X, I3)
      DO 7 J = 2, NPK
         IF (T(J) .EQ. T(J-1))                GO TO 50
         WRITE(20,12) ICOUNT, KEND(ICOUNT), T(J), J
   12    FORMAT(T30, 'KEND(',I3,')= ',I3/T1,F16.8,5X,I3)
                                              GO TO 13
   50    WRITE(20,9) T(J), J
    9    FORMAT(F16.8,5X,I3)
         GO TO 7
   13    ICOUNT = ICOUNT + 1
    7    CONTINUE
                                              RETURN
      END
      SUBROUTINE FLAG(IFLAG,N)
C THIS SUBROUTINE CHECKS FOR
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
C    (1) BREAKPOINTS WHICH ARE NOT STRICTLY INCREASING;
C    (2) TOO MANY CONTINUITY CONDITIONS;
C    (3) K LARGER THAN 20
C    (4) X VALUES OUT OF RANGE OF THE FIRST AND LAST BREAKPOINTS.
C
      PARAMETER(NMAX=100,NDMAX=200,KTNMAX=2000)
      INTEGER V
      COMMON / DATA / NDATA, X(NDMAX), Y(NDMAX), FTABLE
      COMMON / APPROX / BREAK(NMAX),COEF(KTNMAX),L,K,V(NMAX)

      DO 1 I=1,L
        IP1=I+1
         IF(BREAK(I) .GE. BREAK(I+1))        GO TO 2
    1    CONTINUE

      GO TO 4

    2 WRITE(20,3) BREAK(I),BREAK(IP1)
    3 FORMAT(/' BREAKPOINTS MUST BE STRICTLY INCREASING.'/
     *        '  BREAKPOINT',F16.8,2X,'IS NOT LESS THAN BREAKPOINT',
     *        F16.8)
      IFLAG=1

    4 DO 5 I=1,L
         IF(V(I) .GE. K)               GO TO 6
    5    CONTINUE

      GO TO 20


    6 WRITE(20,7) I,V(I),BREAK(I)
    7 FORMAT(/' THE NUMBER OF CONTINUITY CONDITIONS MUST BE STRICTLY'/
     *         5X,' LESS THAN THE SPLINE ORDER K.    V(',I2,')=',I2/
     *        5X,' AT BREAKPOINT',F16.8,'  IS TOO LARGE.')
      IFLAG=1

   20 IF (K .GT. 20)  GO TO 8
      GO TO 10
    8 WRITE(20,9) K
    9 FORMAT(/' K=',I2,' IS TOO LARGE.'/' THE ORDER K MUST BE 20 OR',
     *      ' LESS.')
      IFLAG=1

   10 DO 11 I=1,NDATA
         IF(X(I) .LE. BREAK(1) .OR. X(I) .GE. BREAK(L+1))  GO TO 12
   11    CONTINUE

      GO TO 14

   12 WRITE(20,13) I,X(I),BREAK(1),BREAK(L+1)
   13 FORMAT(/' X VALUE OUT OF RANGE.'/'  X(',I4,')=',F16.8,
     *        ' IS NOT IN THE RANGE BREAK(1)=',F16.8/
     *        5X,'TO BREAK(LAST)=',F16.8)
```

```
   IFLAG=1

14 IF(N .GT. NDATA)  GO TO 16

   GO TO 18

16 WRITE(20,17) N, NDATA
17 FORMAT(/' THE DIMENSION N=',I2,
  *       ' IS GREATER THAN THE SAMPLE SIZE',I4,'.')
   IFLAG=1

18 KTN=K*N
   IF(NDMAX .LT. NDATA)  GO TO 19

   GO TO 22

19 WRITE(20,21) NDMAX,NDATA
21 FORMAT(/' CHECK PARAMETER STATEMENT.'/5X,' NDMAX=',I5,
  *       ' MUST NOT BE LESS THAN THE NUMBER OF DATA POINTS',
  *       I4,'.')
   IFLAG=1

22 IF (NMAX .LT. N)  GO TO 23

   GO TO 25

23 WRITE(20,24) N
24 FORMAT(/' CHECK PARAMETER STATEMENT.'/5X,' NMAX MUST NOT BE',
  *       ' LESS THAN N=',I5)
   IFLAG=1

25 IF(KTNMAX .LT. KTN)  GO TO 26

   GO TO 28
26 WRITE(20,27) KTN
27 FORMAT(/' CHECK PARAMETER STATEMENT.'/5X,' KTNMAX MUST NOT',
  *       ' BE LESS THAN KTN=',I4,'.')
   IFLAG=1

28 IF(IFLAG. EQ. 0)  GO TO 30

   WRITE(20,29)

29 FORMAT(///' ***************************************************'/
  *       ' STEPDOWN CANNOT PROCEED.  PROGRAM ABORTS.')

30                                                   RETURN
   END
```

```
      SUBROUTINE LSTSQ1(T,N K,Q,DIAG,BCOEF)
CALLS BSPLVB, BCHFAC, BCHSLV
C
COMMON STATEMENT DATA IS USED.
C
C THIS IS A MODIFICATION OF DE BOOR'S SUBROUTINE L2APPR,
C PAGE 255. IT INPUTS T,N,K, FINDS THE LEAST SQUARES
C APPROXIMATION TO THE DATA USING WORK ARRAYS Q AND DIAG,
C AND OUTPUTS THE B-SPLINE COEFFICIENTS BCOEF.
C
      PARAMETER(KMAX=20,NDMAX=200)
      REAL BCOEF(N), DIAG(N), Q(K,N), T(1), BIATX(KMAX)
      COMMON / DATA / NDATA, X(NDMAX), Y(NDMAX), FTABLE
C
      DO 7 J=1,N
         BCOEF(J) = 0.
         DO 7 I=1,K
    7       Q(I,J) = 0.
      LEFT = K
      LEFTMK = 0
      DO 20 LL=1,NDATA
C           LOCATE LEFT ST X(LL) IN (T(LEFT),T(LEFT+1))
   10    IF (LEFT .EQ. N)                GO TO 15
         IF (X(LL) .LT. T(LEFT+1))       GO TO 15
         LEFT = LEFT+1
         LEFTMK = LEFTMK+1
                                         GO TO 10
   15    CALL BSPLVB(T,K,1,X(LL),LEFT,BIATX)
         DO 20 MM=1,K
            DW = BIATX(MM)
            J = LEFTMK+MM
            BCOEF(J) = DW*Y(LL) + BCOEF(J)
            I=1
            DO 20 JJ=MM,K
               Q(I,J) = BIATX(JJ)*DW + Q(I,J)
   20          I = I+1
      CALL BCHFAC(Q,K,N,DIAG)
      CALL BCHSLV(Q,K,N,BCOEF)
                                          RETURN
      END
      SUBROUTINE BSPLVB(T,JHIGH,INDEX,X,LEFT,BIATX)
C CALCULATES THE VALUE OF ALL POSSIBLY NONZERO B-SPLINES AT X OF ORDER
C
C               JOUT=MAX(JHIGH,(J+1)*(INDEX-1))
C
C WITH KNOT SEQUENCE T.
C DE BOOR PAGE 134-135
      PARAMETER(JMAX=20)
      INTEGER INDEX,JHIGH,LEFT,    I,J,JP1
      REAL BIATX(JHIGH),T(1),X,    DELTAL(JMAX),DELTAR(JMAX),SAVED,TERM
C     DIMENSION BIATX(JOUT), T(LEFT+JOUT)
C CURRENT FORTRAN STANDARD MAKES IT IMPOSSIBLE TO SPECIFY THE LENGTH
C OF T AND OF BIATX PRECISELY WITHOUT THE INTRODUCTION OF OTHERWISE
```

```
C SUPERFLOUS ADDITIONAL ARGUMENTS.
      DATA J/1/
C     SAVE J,DELTAL,DELTAR (VALID IN FORTRAN 77)
C
                                       GOTO (10,20), INDEX
10    J=1
      BIATX(1)=1.
      IF (J.GE.JHIGH)                  GOTO 99
C
20       JP1=J+1
         DELTAR(J)=T(LEFT+J)-X
         DELTAL(J)=X-T(LEFT+1-J)
         SAVED=0.
         DO 26 I=1,J
            TERM=BIATX(I)/(DELTAR(I)+DELTAL(JP1-I))
            BIATX(I)=SAVED+DELTAR(I)*TERM
26          SAVED=DELTAL(JP1-I)*TERM
         BIATX(JP1)=SAVED
         J=JP1
         IF (J.LT.JHIGH)               GOTO 20
C
99                                     RETURN
      END
      SUBROUTINE BCHFAC (W, NBANDS, NROW, DIAG)
C
C CONSTRUCTS THE CHOLESKY FACTORIZATION C = L * D * L-TRANSPOSE.
C SEE DE BOOR P. 256
C
      INTEGER NBANDS, NROW,  I. IMAX, J, JMAX, N
      REAL W(NBANDS,NROW), DIAG(NROW), RATIO
      IF ( NROW .GT. 1 )               GO TO 9
      IF (W(1,1) .GT.0.) W(1,1) = 1./W(1,1)
                                        RETURN
C STORE DIAGONAL OF C IN DIAG.
    9 DO 10 N=1,NROW
   10    DIAG(N) = W(1,N)
C FACTORIZATION
      DO 20 N=1,NROW
         IF(W(1,N)+DIAG(N) .GT. DIAG(N))GO TO 15
         DO 14 J=1,NBANDS
   14       W(J,N) = 0.
                                        GO TO 20
   15    W(1,N) = 1./W(1,N)
         IMAX = MIN0(NBANDS-1,NROW - N)
         IF (IMAX .LT. 1)                   GO TO 20
         JMAX = IMAX
         DO 18 I=1,IMAX
            RATIO = W(I+1,N)*W(1,N)
            DO 17 J=1,JMAX
   17            W(J,N+I) = W(J,N+I) - W(J+I,N)*RATIO
            JMAX = JMAX - 1
   18       W(I+1,N) = RATIO
   20    CONTINUE
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
                                              RETURN
      END
      SUBROUTINE BCHSLV(W,NBANDS,NROW,B)
C SOLVES THE LINEAR SYSTEM C*X=B OF ORDER NROW FOR X
C PROVIDED W CONTAINS THE CHOLESKY FACTORIZATION FOR THE BANDED (SYM-
C METRIC) POSITIVE DEFINITE MATRIX C AS CONSTRUCTED IN THE SUBROUTINE
C BCHFAC (QUO VIDE).
C DEBOOR PAGE 258
      INTEGER NBANDS,NROW,    J,JMAX,N,NBNDM1
      REAL W(NBANDS,NROW),B(NROW)
      IF (NROW.GT.1)                       GOTO 21
      B(1)=B(1)*W(1,1)
                                           RETURN
C
C               FORWARD SUBSTITUTION. SOLVE L*Y=B FOR Y, STORE IN B.
21    NBNDM1=NBANDS-1
      DO 30 N=1,NROW
         JMAX=MIN0(NBNDM1,NROW-N)
         IF (JMAX.LT.1)                    GOTO 30
         DO 25 J=1,JMAX
25          B(J+N)=B(J+N)-W(J+1,N)*B(N)
30       CONTINUE
C
C     BACKSUBSTITUTION. SOLVE L-TRANSP.X=D**(-1)*Y FOR X, STORE IN B.
      DO 40 N=NROW,1,-1
         B(N)=B(N)*W(1,N)
         JMAX=MIN0(NBNDM1,NROW-N)
         IF (JMAX.LT.1)                    GOTO 40
         DO 35 J=1,JMAX
35          B(N)=B(N)-W(J+1,N)*B(J+N)
40       CONTINUE
                                           RETURN
      END
      SUBROUTINE BSPLPP (T,BCOEF,N,K,SCRTCH,BREAK,COEF,L)
C CALLS BSPLVB
C
C CONVERTS THE B-REPRESENTATION T, BCOEF,N, K OF SOME SPLINE INTO ITS
C PP-REPRESENTATION BREAK, COEF, L, K.
C DE BOOR PAGES 140-141
      PARAMETER(KMAX=20)
      INTEGER K,L,N,    I,J,JP1,KMJ,LEFT,LSOFAR
      REAL BCOEF(N),BREAK(1),COEF(K,1),T(1),    SCRTCH(K,K)
     *,                                         BIATX(KMAX),DIFF,FKMJ,SUM
C DIMENSION BREAK(L+1),COEF(K,L),T(N+K)
      LSOFAR=0
      BREAK(1)=T(K)
      DO 50 LEFT=K,N
C                         FIND THE NEXT NONTRIVIAL KNOT INTERVAL.
         IF (T(LEFT+1).EQ.T(LEFT))      GOTO 50
         LSOFAR=LSOFAR+1
         BREAK(LSOFAR+1)=T(LEFT+1)
         IF (K.GT.1)                    GOTO 9
         COEF(1,LSOFAR)=BCOEF(LEFT)
```

```
                                  GOTO 50
C        STORE THE K B-SPLINE COEFF'S RELEVANT TO CURRENT KNOT INTERVAL
C        IN SCRTCH(.,1).
9        DO 10 I=1,K
10            SCRTCH(I,1)=BCOEF(LEFT-K+I)
C
C        FOR J=1,...,K-1, COMPUTE THE K-J B-SPLINE COEFF'S RELEVANT TO
C        CURRENT KNOT INTERVAL FOR THE J-TH DERIVATIVE BY DIFFERENCING
C        THOSE FOR THE (J-1)ST DERIVATIVE, AND STORE IN SCRTCH(.,J+1).
         DO 20 JP1=2,K
            J=JP1-1
            KMJ=K-J
            FKMJ=FLOAT(KMJ)
            DO 20 I=1,KMJ
               DIFF=T(LEFT+I)-T(LEFT+I-KMJ)
               IF (DIFF.GT.0)  SCRTCH(I,JP1)=
     *                    ((SCRTCH(I+1,J)-SCRTCH(I,J))/DIFF)*FKMJ
20             CONTINUE
C
C FOR J=0,...,K-1, FIND THE VALUES AT T(LEFT) OF THE J+1
C B- SPLINES OF ORDER J+1 WHOSE SUPPORT CONTAINS THE CURRENT
C KNOT INTERVAL FROM THOSE OF ORDER J (IN BIATX), THEN COMBINE
C WITH THE B-SPLINE COEFF'S (IN SCRTCH(.,K-J)) FOUND EARLIER
C TO COMPUTE THE (K-J-1)ST DERIVATIVE AT T(LEFT) OF THE GIVEN
C SPLINE.
C   NOTE. IF THE REPEATED CALLS TO BSPLVB ARE THOUGHT TO GENERATE
C TOO MUCH OVERHEAD, THEN REPLACE THE FIRST CALL BY
C BIATX(1)=1.
C AND THE SUBSEQUENT CALL BY THE STATEMENT
C J=JP1-1
C FOLLOWED BY A DIRECT COPY OF THE LINES
C DELTAR(J)=T(LEFT+J)-X
C    ......
C BIATX(J+1)=SAVED
C FROM BSPLVB. DELTAL(KMAX) AND DELTAR(KMAX) WOULD HAVE TO
C APPEAR IN A DIMENSION STATEMENT, OF COURSE.
C
          CALL BSPLVB(T,1,1,T(LEFT),LEFT,BIATX)
          COEF(K,LSOFAR)=SCRTCH(1,K)
          DO 30 JP1=2,K
             CALL BSPLVB(T,JP1,2,T(LEFT),LEFT,BIATX)
             KMJ=K+1-JP1
             SUM=0.
             DO 28 I=1,JP1
28              SUM=BIATX(I)*SCRTCH(I,KMJ)+SUM
30           COEF(KMJ,LSOFAR)=SUM
50        CONTINUE
      L=LSOFAR
                                RETURN
      END
      SUBROUTINE ERRL21(FTAU, ERROR, ERRDF, SSE, MSE)
```

```
CALLS SUBPROGRAM PPVALU(INTERV)
C
C THIS SUBROUTINE COMPUTES THE ERROR SS AND MS.  IT IS A
C MODIFIED VERSION OF DE BOOR'S SUBROUTINE L2ERR, PAGE 261.
C MSE IS THE OUTPUTED MEAN SQUARED ERROR.
C
      PARAMETER(NMAX=100,NDMAX=200,KTNMAX=2000)
      INTEGER ERRDF, V
      REAL FTAU(1), ERROR(1), MSE, Y, X, BREAK, COEF
C DIMENSION FTAU(NDATA), ERROR(NDATA)
      COMMON / DATA / NDATA, X(NDMAX), Y(NDMAX), FTABLE
      COMMON / APPROX / BREAK(NMAX), COEF(KTNMAX), L, K
     *                   , V(NMAX)
C
      SSE=0.
      DO 10 LL=1,NDATA
          FTAU(LL) = PPVALU(BREAK,COEF,L,K,X(LL),0)
          ERROR(LL) = Y(LL) - FTAU(LL)
   10      SSE = SSE + ERROR(LL)**2
      MSE = SSE/ERRDF
                                              RETURN
      END
      REAL FUNCTION PPVALU(BREAK,COEF,L,K,X,JDERIV)
C CALLS 'INTERV'
C CALCULATES VALUE AT X OF JDERIV-TH DERIVATIVE OF PP FCT FROM PP-REPR
      INTEGER JDERIV,K,L,    I,M,NDUMMY
      REAL BREAK(L),COEF(K,L),X,    FMMJDR,H
      PPVALU=0.
      FMMJDR=K-JDERIV
C             DERIVATIVES OF ORDER K OR HIGHER AREARE IDENTICALLY ZERO.
      IF (FMMJDR.LE.0)                GOTO 99
C
C        FIND INDEX I OF LARGEST BREAKPOINT TO THE LEFT OF X.
      CALL INTERV(BREAK,L,X,I,NDUMMY)
C
C        EVALUATE JDERIV-TH DERIVATIVE OF I-TH POLYNOMIAL PIECE AT X.
      H=X-BREAK(I)
      DO 10 M=K,JDERIV+1,-1
         PPVALU=(PPVALU/FMMJDR)*H+COEF(M,I)
10       FMMJDR=FMMJDR-1
99                                    RETURN
      END
      SUBROUTINE INTERV(XT,LXT,X,LEFT,MFLAG)
C COMPUTES LEFT=MAX(I,1.LE.I.LE.LXT.AND.XT(I).LE.X)
C DE BOOR PAGE 92
      INTEGER LEFT,LXT,MFLAG,    IHI,ILO,ISTEP,MIDDLE
      REAL X,XT(LXT)
      DATA  ILO/1/
C     SAVE ILO (A VALID FORTRAN STATEMENT IN THE NEW  1977 STANDARD)
      IHI=ILO+1
      IF (IHI.LT.LXT)                    GOTO 20
         IF (X.GE.XT(LXT))               GOTO 110
         IF (LXT.LE.1)                   GOTO 90
```

```
         ILO=LXT-1
         IHI=LXT
C
20     IF (X.GE.XT(IHI))                   GOTO 40
       IF (X.GE.XT(ILO))                   GOTO 100
C
C            *****NOW X.LT.XT(ILO). DECREASE ILO TO CAPTURE X.
       ISTEP=1
31       IHI=ILO
         ILO=IHI-ISTEP
         IF (ILO.LE.1)                     GOTO 35
         IF (X.GE.XT(ILO))                 GOTO 50
         ISTTP=ISTEP*2
                                           GOTO 31
35     ILO=1
       IF (X.LT.XT(1))                     GOTO 90
                                           GOTO 50
C            *****NOW X.GE.XT(IHI). INCREASE IHI TO CAPTURE X.
40     ISTEP=1
41       ILO=IHI
         IHI=ILO+ISTEP
         IF (IHI.GE.LXT)                   GOTO 45
         IF (X.LT.XT(IHI))                 GOTO 50
         ISTEP=ISTEP*2
                                           GOTO 41
45     IF (X.GE.XT(LXT))                   GOTO 110
       IHI=LXT
C
C          *****NOW XT(ILO).LE.X.LT.XT(IHI). NARROW THE INTERVAL.
50     MIDDLE=(ILO+IHI)/2
       IF (MIDDLE.EQ.ILO)                  GOTO 100
C      NOTE. IT IS ASSUMED THAT MIDDLE=ILO IN CASE IHI=ILO+1.
       IF (X.LT.XT(MIDDLE))                 GOTO 53
         ILO=MIDDLE
                                           GOTO 50
53       IHI=MIDDLE
                                           GOTO 50
C*****SET OUTPUT AND RETURN.
90     MFLAG=-1
       LEFT=1
                                           RETURN
100    MFLAG=0
       LEFT=ILO
                                           RETURN
110    MFLAG=1
       LEFT=LXT
                                           RETURN
       END
       SUBROUTINE CNTRST(I, JDERIV, N, K, L, T, BREAK, KEND,
      *                  BRT, BLF, DCOEF, CTRAST)
CALLS BCONT
C
C FINDS THE CONTRAST COEFFICIENTS FOR TESTING CONTINUITY OF THE
```

```
C JDERIV-TH DERIVATIVE OF THE SPLINE FUNCTION AT BREAK(I).
C
C****** I N P U T ******
C L....NUMBER OF INTERVALS
C T(1),...,T(N+K).. .THE KNOT SEQUENCE
C I....THE INDEX OF THE BREAKPOINT OF INTEREST
C BREAK(1),...BREAK(L+1)....THE BREAKPOINT SEQUENCE
C JDERIV....NONNEGATIVE INTEGER GIVING THE ORDER OF THE DERI-
C           VATIVE TO BE EVALUATED
C KEND(1),...KEND(L)....INDEX OF THE LARGEST KNOT EQUAL TO
C                       BREAK(1),...,BREAK(L)
C N....DIMENSION OF SPLINE SPACE
C K....ORDER OF SPLINE
C BRT, BLF, DCOEF...WORK ARRAYS OF LENGTH N
C
C****** O U T P U T ******
C CTRAST(1),...,CTRAST(N)....THE CONTRAST COEFFICIENTS USED TO
C                            TEST CONTINUITY OF THE JDERIV-TH
C                            DERIVATIVE AT BREAK(I)
C
C****** M E T H O D ******
C THE FUNCTION SUBPROGRAM BCONT IS USED TO COMPUTE THE VALUE OF
C THE LEFT AND RIGHT LIMITS OF THE JDERIV-TH DERIVATIVE OF
C RELEVANT B-SPLINES AT BREAK(I).
C
      INTEGER KEND(1)
      REAL BRT(1), BLF(1), CTRAST(1), T(1),
     *     BREAK(1), DCOEF(1)

      DO 20 JJ = 1, N
   20   DCOEF(JJ) = 0.

      DO 10 J = 1, N
        DCOEF(J) = 1.

COMPUTE VALUE FOR RIGHT CONTINUITY
        IF (KEND(I)-K+1 .LE. J .AND. J .LE.KEND(I))  GO TO 30
        BRT(J) = 0.
                                                GO TO 40
   30   BRT(J) = BCONT(T,DCOEF,N,K,BREAK(I),KEND(I),
     *                          JDERIV)

COMPUTE VALUE FOR LEFT CONTINUITY
   40   IF(KEND(I-1)-K+1 .LE. J .AND. J .LE. KEND(I-1)) GO TO 50
        BLF(J) = 0.
                                                GO TO 60
   50   BLF(J) = BCONT(T,DCOEF,N,K,BREAK(I),
     *                          KEND(I-1),JDERIV)
C
COMPUTE DIFFERENCE OF THE LEFT AND RIGHT VALUES
   60   CTRAST(J) = BRT(J) - BLF(J)
        DCOEF(J) = 0.
   10 CONTINUE
```

```
                                                  RETURN
      END
      REAL FUNCTION BCONT(T,BCOEF,N,K,X,I,JDERIV)
CALCULATES VALUE AT X OF JDERIV-TH DERIVATIVE OF SPLINE FROM B-REP.
C THIS IS A MODIFIED VERSION OF DE BOOR'S SUBROUTINE BVALUE,
C PAGE 144. THE ONLY DIFFERENCE IS THAT THE LEFT-HAND KNOT
C INDEX I IS INPUTED RATHER THAN FOUND IN INTERV. CONSE-
C QUENTLY, LINE 10 IS MODIFIED TO INPUT I AND LINES 710 AND
C 720 ARE OMITTED. THE PURPOSE IS TO ALLOW EVALUATION AT
C BREAKPOINTS WITH LEFT (OR RIGHT) CONTINUITY.
      PARAMETER(KMAX=20)
      INTEGER JDERIV,K,N,    I,ILO,IMK,J,JC,JCMIN,JCMAX,JJ,KMJ,KM1,MFLAG
     *                     ,NMI
      REAL BCOEF(1),T(1),X,    AJ(KMAX),DL(KMAX),DR(KMAX),FKMJ
C     DIMENSION T(N+K)
      BCONT=0.
      IF (JDERIV.GE.K)                GOTO 99
C
C  *** IF K=1 (AND JDERIV=0), BCONT=BCOEF(I).
      KM1=K-1
      IF (KM1.GT.0)                   GOTO 1
      BCONT=BCOEF(I)
                                      GOTO 99
C
C  *** STORE THE K B-SPLINE COEFFICIENTS RELEVENT FOR THE KNOT INTERVAL
C       (T(I),T(I+1)) IN AJ(1),...,AJ(K) AND COMPUTE DL(J)=X-T(I+1-J),
C       DR(J)=T(I+J)-X, J=1,...,K-1. SET ANY OF THE AJ NOT OBTAINABLE
C       FROM INPUT TO ZERO. SET ANY T.S NOT OBTAINABLE EQUAL TO T(1) OR
C       TO T(N+K) APPROPRIATELY.
    1 JCMIN=1
      IMK=I-K
      IF (IMK.GE.0)                   GOTO 8
      JCMIN=1-IMK
      DO 5 J=1,I
    5    DL(J)=X-T(I+1-J)
      DO 6 J=I,KM1
         AJ(K-J)=0.
    6    DL(J)=DL(I)
                                      GOTO 10
    8 DO 9 J=1,KM1
    9    DL(J)=X-T(I+1-J)
C
   10 JCMAX=K
      NMI=N-I
      IF (NMI.GE.0)                   GOTO 18
      JCMAX=K+NMI
      DO 15 J=1,JCMAX
   15    DR(J)=T(I+J)-X
      DO 16 J=JCMAX,KM1
         AJ(J+1)=0.
   16    DR(J)=DR(JCMAX)
                                      GOTO 20
   18 DO 19 J=1,KM1
```

```
   19    DR(J)=T(I+J)-X
C
   20 DO 21 JC=JCMIN,JCMAX
   21    AJ(JC)=BCOEF(IMK+JC)
C
C                 *** DIFFERENCE THE  COEFFICIENTS JDERIV TIMES.
      IF (JDERIV.EQ.0)                   GOTO 30
      DO 23 J=1,JDERIV
         KMJ=K-J
         FKMJ=FLOAT(KMJ)
         ILO=KMJ
         DO 23 JJ=1,KMJ
            AJ(JJ)=((AJ(JJ+1)-AJ(JJ))/(DL(ILO)+DR(JJ)))*FKMJ
   23       ILO=ILO-1
C
C  *** COMPUTE VALUE AT X IN (T(I),T(I+1)) OF JDERIV-TH DERIVATIVE,
C      GIVEN ITS RELEVENT B-SPLINE COEFS IN AJ(1),...,AJ(K-JDERIV).
   30 IF (JDERIV.EQ.KM1)                 GOTO 39
      DO 33 J=JDERIV+1,KM1
         KMJ=K-J
         ILO=KMJ
         DO 33 JJ=1,KMJ
            AJ(JJ)=(AJ(JJ+1)*DL(ILO)+AJ(JJ)*DR(JJ))/(DL(ILO)+DR(JJ))
   33       ILO=ILO-1
   39 BCONT=AJ(1)
C
   99                                    RETURN
      END
C THIS IS FOR 1 DF HYPOTHESES.
      SUBROUTINE SSHYP(BCOEF,CTRAST,W,NBANDS,N,PVAR,VAR,
     *                 SSH, MSH, HDF)
CALLS FORSUB
C
C FINDS THE VARIANCE OF A CONTRAST AND THE MS FOR TESTING THAT
C THE CONTRAST IS ZERO.
C
C * * * * * * I N P U T * * * * * *
C LINV....THE INVERSE OF L OBTAINED FROM B C H I N V
C CTRAST.... THE CONTRAST VECTOR OBTAINED FROM C N T R S T
C BCOEF....THE B-SPLINE COEFFICIENTS
C W....THE MATRIX FROM B C H F A C CONTAINING D-INVERSE
C NBANDS....EQUALS K
C N....THE NUMBER OF ELEMENTS IN THE CONTRAST VECTOR--
C      ALSO THE DIMENSION OF THE SPLINE SPACE
C PVAR....WORK VECTOR OF LENGTH N EQUAL TO THE PRODUCT
C         W(1,.)*A, I.E. D-INV*LINV*CTRAST
C
C * * * * * * O U T P U T * * * * * *
C VAR....THE COEFFICIENT OF SIGMA-SQUARED IN THE VARIANCE OF THE
C        CONTRAST, I.E. THE PRODUCT CTRAST-TRANSPOSE*LINV-
C        TRANSPOSE*D-INV*LINV*CTRAST
C SSH, MSH, HDF....THE SS, MS, AND DF FOR THE HYPOTHESIS
C
```

```
C * * * * * * M E T H O D * * * * * *
C THE PRODUCT LINV*CTRAST IS OBTAINED THEN PREMULTIPLIED BY
C D-INV THEN THAT RESULT IS PREMULTIPLIED BY (LINV*CTRAST)-
C TRANSPOSE
C
      INTEGER HDF
      REAL NUM, MSH
      REAL  CTRAST(1), W(NBANDS,N), PVAR(N)
      REAL BCOEF(1)
      NUM= 0.
      DO 3 II=1,N
    3    NUM = NUM + (CTRAST(II)*BCOEF(II))
      CALL FORSUB(W,CTRAST,NBANDS,N)
      DO 1 II=1,N
    1    PVAR(II) = W(1,II)*CTRAST(II)
      VAR = 0.
      DO 2 II=1,N
    2    VAR = VAR + CTRAST(II)*PVAR(II)
      SSH = (NUM**2)/VAR
      MSH = SSH
      HDF = 1
                                                    RETURN
      END
       SUBROUTINE FORSUB(W,AA,NBANDS,NROW)
C
C SOLVES LY=AA FOR Y AND STORES IN AA
C
C * * * * * * I N P U T * * * * * *
C W...A MATRIX FED IN FROM B C H F A C AND CONTAINING IN ITS ROWS
C       THE DIAGONALS OF A P. D. SYMMETRIC MATRIX C
C NBANDS...THE BANDWIDTH OF C
C NROW...THE ORD OF C
C AA...THE VECTOR OF LENGTH NROW CONTAINING THE RIGHT HAND SIDE
C
C * * * * * * O U T P U T * * * * * *
C AA...THE VECTOR OF LENGTH NROW CONTAINING THE SOLUTION
C
C * * * * * * M E T H O D * * * * * *
C THE FORWARD SUBSTITUTION ROUTINE FROM DEBOOR'S BCHSLV IS USED
C
      REAL W(NBANDS, NROW), AA(NROW)

      IF (NROW.GT.1)    GO TO 21
      AA(1)=AA(1)*W(1,1)
      RETURN
   21 NBNDM1=NBANDS-1

      DO 30 N=1,NROW
         JMAX=MIN0(NBNDM1,NROW-N)
         IF (JMAX.LT.1)    GO TO 30
         DO 25 J=1,JMAX
   25       AA(J+N)=AA(J+N)-W(J+1,N)*AA(N)
   30 CONTINUE
```

```
      RETURN
      END
      SUBROUTINE REKNOT(KEND,KNOT,N,K,L,T,V,BREAK)
C RELABELS THE KNOT SEQUENCE T( ) BY OMITTING THE LEAST
C SIGNIFICANT KNOT, BREAK(KNOT)
C
C **************INPUT**********************
C KEND(I)...THE INDEX OF THE LARGEST KNOT EQUAL TO BREAK(I)
C KNOT...INDEX OF THE BREAKPOINT TO BE OMITTED
C N...DIMENSION OF THE (OLD) SPLINE SPACE
C K...ORDER OF THE SPLINE
C T... KNOT SEQUENCE
C V(I)..NUMBER OF CONTINUITY CONDITIONS AT BREAK(I)
C BREAK...BREAKPOINT SEQUENCE
C
C ***************OUTPUT******************
C N...DIMENSION OF (NEW) SPLINE SPACE WITH BREAK(KNOT) OMITTED
C T(1)..T(N)...(NEW) KNOT SEQUENCE WITH BREAK(KNOT) OMITTED
C
C ***************METHOD****************
C SINCE BREAK(KNOT)=T(KEND(KNOT-1)+1)=...=T(KEND(KNOT)), WE
C RELABEL ALL T'S BEYOND.
C
      DIMENSION KEND(1), T(1), BREAK(1)
      INTEGER V(1)
       I1=KEND(KNOT-1)+1
      I2=KEND(KNOT)+1
              J1=N+K-I2+1
      DO 1 KT=1,J1
         K1=KT-1
    1       T(I1+K1)=T(I2+K1)
       N=N-(K-V(KNOT))
              DO 2 II=KNOT,L
           BREAK(II)=BREAK(II+1)
           IF(II .EQ. L)          GO TO 2
           V(II)=V(II+1)
           KEND(II)=KEND(II-1)+K-V(II)
    2      CONTINUE
            L=L-1

      RETURN
      END
      SUBROUTINE STDERR(W,BCOEF,K,N,L,MSE,BB,AA,SE,LINV)
CALLS BCHINV AND MATVEC
C
C THIS SUBROUTINE COMPUTES THE STANDARD ERRORS OF THE B-SPLINE
C COEFFICIENTS AND OUTPUTS THEM.
C
      REAL W(K,N),BCOEF(N),MSE,BB(N,N),SE(N),LINV(N,N),AA(N,N)
      CALL BCHINV(W,K,N,LINV)
      WRITE(20,10) L,K
   10 FORMAT(///' PROCEDURE TERMINATES WITH L=',I3,' AND K=',I3//
```

```
     *   ' N        COEF                 S.E.')

      DO 11 II=1,N
         DO 11 JJ=1,N
   11       BB(JJ,II)=W(1,II)*LINV(II,JJ)

      CALL MATVEC(N,N,N,BB,LINV,AA)

      DO 13 II=1,N
         SE(II)=SQRT(AA(II,II)*MSE)
         WRITE(20,12) II,BCOEF(II),SE(II)
   12    FORMAT(I3,2F16.8)
   13    CONTINUE
                                                 RETURN
      END
      SUBROUTINE BCHINV (W, NBANDS, NROW, INV)
C FINDS L-INVERSE WHERE L IS THE LOWER TRIANGULAR MATRIX
C IN THE CHOLESKY FACTORIZATION OF THE BANDED SYMMETRIC P.D.
C MATRIX C AS CONSTRUCTED IN THE SUBROUTINE B C H F A C.
C SEE DE BOOR, P. 256
C
C****** I N P U T ******
C NROW.....IS THE ORDER OF THE MATRIX C.
C NBANDS.....IS THE BANDWIDTH OF C.
C W.....CONTAINS THE CHOLESKY FACTORIZATION OF C AS OUTPUT
C      FROM SUBROUTINE B C H F A C WITH ROWS 2 THROUGH NBANDS-1
C      CONTAINING THE NON-ZERO AND NON-UNIT DIAGONAL ENTRIES
C      OF L.
C
C****** O U T P U T ******
C INV.....THE INVERSE OF L.
C
C****** M E T H O D ******
C THE LINEAR SYSTEM L*L-INVERSE = IDENTITY IS SOLVED FOR
C L*INVERSE BY SUCCESSIVELY FINDING THE COLUMNS OF L-INVERSE
C USING THE FORWARD SUBSTITUTION ROUTINE IN B C H S L V.
C
      INTEGER NBANDS, NROW, J,JMAX,N,NBNDM1
      REAL W(NBANDS,NROW), INV(NROW,NROW)
      IF ( NROW .GT. 1 )                         GO TO 21
      INV(1,1) = 1.
                                                   RETURN
C
C STORE THE IDENTITY MATRIX IN INV
   21 DO 10 J=1,NROW
              DO 10 I=1,NROW
               IF (I .EQ. J)                       GO TO 20
                INV(I,J) = 0.
                 GO TO 10
   20            INV(I,J) = 1.
   10           CONTINUE
C
C NOW USE FORWARD SUBSTITUTION FROM B C H S L V.
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      NBNDM1 = NBANDS - 1
      DO 40 J=1,NROW
         DO 30 N=1,NROW
            JMAX = MIN0(NBNDM1,NROW-N)
            IF (JMAX .LT. 1)                GO TO 30
            DO 25 I = 1,JMAX
   25          INV(I+N,J) = INV(I+N,J) - W(I+1,N)*INV(N,J)
   30    CONTINUE
   40 CONTINUE
                                                      RETURN
      END
      SUBROUTINE MATVEC(N,NM,M,X,Y,Z)
C FINDS THE NXM MATRIX OR VECTOR Z WHICH IS THE PRODUCT
C X*Y WHERE X IS NXNM AND Y IS NMXM.
C
      REAL X(N,NM), Y(NM,M), Z(N,M)
      DO 1 I = 1,N
         DO 2 J = 1,M
            Z(I,J) = 0.
            DO 3 K=1,NM
    3          Z(I,J) = Z(I,J) + X(I,K)*Y(K,J)
    2    CONTINUE
    1 CONTINUE
                                                      RETURN
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

ORDER REDUCTION PROGRAM LISTING

```
      PROGRAM XPLT2(INPUT,OUTPUT,TAPE6=OUTPUT,TAPE20,TAPE21)
C STEPDOWN FOR REDUCING SPLINE ORDER (FOR ALL INTERVALS
C SIMULTANEOUSLY) WHILE KEEPING THE KNOTS FIXED AND ASSUMING
C K-2 CONTINUITY CONDITIONS
C
C
C NDMAX IS AT LEAST THE SAMPLE SIZE, NDATA.
C NMAX IS AT LEAST N.  WITH MAXIMUM CONTINUITY CONSTRAINTS,
C       N=L+K-1.  WITH NO CONTINUITY CONSTRAINTS, N=L*K.
C KTNMAX IS AT LEAST K*N.
C
      PARAMETER (NMAX=100,NDMAX=200,KTNMAX=2000)
      REAL BCOEF(NMAX),Q(KTNMAX),DIAG(KTNMAX),T(NDMAX)
     *    ,LINV(KTNMAX),DCOEF(NMAX),BRT(NMAX),BLF(NMAX)
     *    ,AA(NMAX),VAR(NMAX),B(NMAX),C(NMAX),ATRP(NMAX)
     *    ,F(NDMAX),ERROR(NDMAX),MSH,MSE,SE(NMAX)
     *    ,KMAT(NMAX,NMAX),FB(NMAX)
     *    ,WVAR(NMAX),CC(NMAX),CT(NMAX),CTRAST(NMAX)
      INTEGER ERRDF,HDF,V,KEND(NMAX)
      COMMON /DATA/ NDATA,X(NDMAX),Y(NDMAX),FTABLE
      COMMON /APPROX/ BREAK(NMAX),COEF(KTNMAX),L,K,V(NMAX)

      ICOUNT=0
C ENTER DATA
      CALL DAT1(ICOUNT)

C GET THE KNOT SEQUENCE
      CALL VL2NT(BREAK,L,K,V,T,N,KEND)

C PRELIMINARY OUTPUT
      CALL OUTNTS(BREAK,V,L,T,N,K,KEND)

CHECK INPUT DATA
      IFLAG=0
      CALL FLAG(IFLAG,N)
      IF(IFLAG .EQ. 1) GO TO 25

      CALL PSEUDO

      IEND=0
      LM1=L-1


C WE WILL TEST THAT THE K-1-ST DERIVATIVE IS ZERO IN ALL INTERVALS
C
C GET THE LEAST SQUARES FIT, I.E., THE B-SPLINE COEFFICIENTS.
    1 CALL LSTSQ1(T,N,K,Q,DIAG,BCOEF)
C         LSTSQ1 CALLS BSPLVB,BCHFAC,AND BCHSLV.
```

```
C GET SSE AND MSE
      ERRDF=NDATA-N
      CALL BSPLPP(T,BCOEF,N,K,DIAG,BREAK,COEF,L)
      CALL ERRL21(F,ERROR,ERRDF,SSE,MSE)
C         ERRL21 CALLS PPVALU WHICH CALLS INTERV.
      IF(IEND .EQ. 2)   GO TO 30

      LP1=L+1
      DO 10 I=1,LP1
   10    FB(I)=PPVALU(BREAK,COEF,L,K,BREAK(I),0)

C***************   PLOTS   ******************************
      CALL INFOPLT(0,NDATA,X,1,F,1,0.,62.,74.,158.,1.,
     *             9,9HINDY DATA,1,1HY,0,5.,4.,.75,.75)
      CALL INFOPLT(0,NDATA,X,1,Y,1,0.,62.,74.,158.,1.,
     *             9,9HINDY DATA,1,1HY,22,5.,4.,.75,.75)
      CALL INFOPLT(1,LM1,BREAK(2),1,FB(2),1,0.,62.,74.,158.,1.,
     *             9,9HINDY DATA,1,1HY,1,5.,4.,.75,.75)

      KM1=K-1
      KM2=K-2
      KM3=K-3
      IF(IEND.EQ.1) GO TO 8
      IF(V(2).EQ.KM2) GO TO 12
      WRITE(20,15) K,KM2,SSE,MSE
   15 FORMAT(//' ****************************************************',
     *       //' THE SMOOTHEST SPLINE OF ORDER K=',I2,2X,
     *       'WITH MAXIMUM CONTINUITY C',I2,2X/5X,
     *     'HAS SSE=',F16.8,2X,'AND MSE=',F16.8)
      IF(K.EQ.1) GO TO 8
      WRITE(20,16) K,KM3,KM1,KM3
   16 FORMAT(/' CAN ORDER K=',I2,2X,'WITH SUB-MAXIMUM CONTINUITY C',
     -       I2,2X/5X,'BE REDUCED TO ORDER K=',I2,2X,
     *       'WITH MAXIMUM CONTINUTIY C',I2,' ?')
      DO 18 II=2,L
   18    V(II)=K-2
      CALL VL2NT(BREAK,LM1,V,T,N,KEND)
      GO TO 1



C TEST FOR LOWER ORDER WITH THE HYPOTHESIS MATRIX KMAT.
   12 DO 80 JJ=1,N
   80    DCOEF(JJ) = 0.
      DO 2 III=1,N
         DCOEF(III) = 1.
         DO 81 II=1,L
            KMAT(II,III)=BCONT(T,DCOEF,N,K,BREAK(II),KEND(II),KM1)
            M=(II-1)*N+III
   81       CT(M)=KMAT(II,III)
         DCOEF(III) = 0.
```

```
   2    CONTINUE
        CALL SSHYP2(BCOEF,CT,Q,K,L,N,AA,DIAG,VAR,SSH,MSH,HDF,B,C
       *    ,ATRP,WVAR,CC,CTRAST)
C            SSHYP2 CALLS FORSUB AND MATVEC.
        FRATIO=MSH/MSE
        IF(FRATIO.GE.FTABLE) GO TO 5
        WRITE(20,3)
   3 FORMAT(///' YES.')
        WRITE(20,31) K,KM3,FTABLE,FRATIO,SSE,MSE
  31 FORMAT(' FOR K=',I2,2X,'AND C',I2,2X/5X,
       *        'FTABLE VALUE =',F16.8,5X,'OBSERVED F=',F16.8
       *        /5X,'SSE=',F16.8,2X,'MSE=',F16.8)
        K=K-1
        CALL VL2NT(BREAK,L,K,V,T,N,KEND)
        GO TO 1

   5 IEND=1
        WRITE(20,6)
   6 FORMAT(///' NO.')
        WRITE(20,31) K,KM3,FTABLE,FRATIO,SSE,MSE
        WRITE(20,32)
  32 FORMAT(///' PROCEDURE TERMINATES    *************************')
        DO 71 II=2,L
  71    V(II)=K-1
        CALL VL2NT(BREAK,L,K,V,T,N,KEND)
        GO TO 1

C PRINT RESULTING COEFFICIENTS AND STANDARD ERRORS.
   8 WRITE(20,13) L,K,KM2
  13 FORMAT(////' PROCEDURE TERMINATES WITH L=',I2,'; K=',I2,'; C',I2
             ///'  N          COEF                ST. ERR.')

        CALL STDERR(Q,BCOEF,K,N,L,MSE,DIAG,AA,SE,LINV)
C              STDERR CALLS BCHINV AND MATVEC.

        IF(K .EQ. 1)   GO TO 25
        IEND=2
        K=K-1
        DO 40 II=2,L
  40    V(II)=K-1
        CALL VL2NT(BREAK,L,K,V,T,N,KEND)
        GO TO 1
  30 WRITE(20,29) KM1,KM3,SSE,MSE
  29 FORMAT(///' ******************** FURTHER INFORMATION **********'/
       *        5X,'FOR K=',I2,2X,'AND C',I2,2X/
       *        5X,'SSE=',F16.8,2X,'MSE=',F16.8)

        CALL CALPLT(0.,0.,999)

  25 STOP

        END
```

```
      SUBROUTINE SSHYP2(BCOEF,CT,W,NBANDS,NCON,N,A
     *          ,PVAR,VAR,SSH,MSH,HDF,B,C,ATRP,WVAR,CC,CTRAST)
      INTEGER HDF,NCON,N,NBANDS
      REAL SSH
      REAL PVAR(NCON,N),A(NCON,N),W(NBANDS,N),CT(1)
      REAL VAR(NCON,NCON), B(N), C(NCON), ATRP(N,NCON)
      REAL MSH, WVAR(NCON,NCON), CTRAST(NCON,N), BCOEF(N),CC(N)
      DO 1 I=1,NCON
      DO 2 JJ=1,N
         M=JJ+(I-1)*N
        CC(JJ)=CT(M)
2         CTRAST(I,JJ)=CC(JJ)
        CALL FORSUB(W,CC,NBANDS,N)
        DO 3 J=1,N
3           A(I,J)=CC(J)
1      CONTINUE
      DO 4 II=1,NCON
        DO 4 JJ=1,N
4          PVAR(II,JJ)=W(1,JJ)*A(II,JJ)
      DO 5 I=1,N
        DO 5 J=1,NCON
5         ATRP(I,J)=A(J,I)
      CALL MATVEC(NCON,N,NCON,PVAR,ATRP,VAR)
      DO 6 I=1,NCON
            MM=NCON-I+1
        DO 6 J=1,MM
6         WVAR(I,J)=VAR(I+J-1,J)
      CALL BCHFAC (WVAR,NCON,NCON,DIAG)
      CALL MATVEC(NCON,N,1,CTRAST,BCOEF,B)
      CALL FORSUB(WVAR,B,NCON,NCON)
      DO 8 J=1,NCON
8       C(J)=WVAR(1,J)*B(J)
      CALL MATVEC(1,NCON,1,B,C,SSH)
      MSH=SSH/NCON
      HDF=NCON

      RETURN
     END
```