

AD-762 419

**FORTTRAN SUBROUTINES FOR BICUBIC SPLINE
INTERPOLATION**

John J. Cornyn

**Naval Research Laboratory
Washington, D.C.**

June 1973

DISTRIBUTED BY:

NTIS

**National Technical Information Service
U. S. DEPARTMENT OF COMMERCE
5285 Port Royal Road, Springfield Va. 22151**

AD 762419

NRL Memorandum Report 2596
NRL Computer Bulletin 32

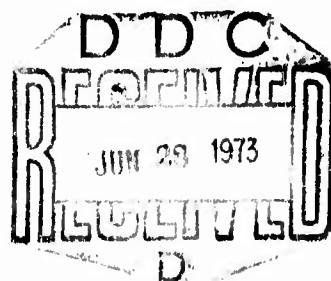
Fortran Subroutines for Bicubic Spline Interpolation

JOHN J. CORNYN

*Information Processing Systems Branch
Communications Sciences Division*

June 1973

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
U.S. Department of Commerce
Springfield, MA 01104



NAVAL RESEARCH LABORATORY
Washington, D.C.

Approved for public release; distribution unlimited.

47

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

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

1. ORIGINATING ACTIVITY (Corporate author) Naval Research Laboratory Washington, D.C. 20375		2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	
		7b. GROUP ----	
3. REPORT TITLE FORTRAN SUBROUTINES FOR BICUBIC SPLINE INTERPOLATION			
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) This is a final report on one phase of a continuing problem.			
5. AUTHOR(S) (First name, middle initial, last name) John J. Cornyn			
6. REPORT DATE June 1973		7a. TOTAL NO. OF PAGES 4647	7b. NO. OF REFS 11
8a. CONTRACT OR GRANT NO. NRL General and Administrative Function S01-38		9a. ORIGINATOR'S REPORT NUMBER(S) NRL Memorandum Report 2596	
b. PROJECT NO.		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) NRL Computer Bulletin 32	
c.			
d.			
10. DISTRIBUTION STATEMENT Approved for public release; distribution unlimited.			
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY Department of the Navy (Office of Naval Research) Washington, D.C. 20360	
13. ABSTRACT Two Fortran subroutines (BICUB1 and BICUB2) which perform bicubic spline interpolation of a tabulated function of two variables are described. Given the values $X(1), \dots, X(N)$ and $Y(1), \dots, Y(M)$ of two independent variables and the corresponding function values $\{U(I,J)=f(X(I), Y(J))\}$, $I=1, \dots, N$ and $J=1, \dots, M$ and certain normal derivatives (optional) along the boundaries of the x-y mesh, BICUB1 estimates the derivatives f_x, f_y, and f_{xy} at each (I,J) mesh point. If the normal derivatives along the mesh boundaries are unknown, BICUB1 estimates them using a moving third order two dimensional Lagrange interpolating polynomial. Given the coordinates (XPT,YPT) and the derivatives calculated by BICUB1, BICUB2 obtains the coefficients of the bicubic polynomial for the rectangular region of the mesh containing (XPT,YPT) and estimates the functional value $UPT=f(XPT,YPT)$. In effect, the routines pass a twice continuously differentiable piecewise bicubic polynomial, $u(x,y) \in C^2$, through the given functional values.			

DD FORM 1 NOV 65 1473

(PAGE 6)

S/N 0101-807-6801

UNCLASSIFIED

Security Classification

Security Classification

DD FORM 1473 (BACK)
(PAGE 2)

Security Classification

CONTENTS

Abstract	ii
Problem Status	ii
Authorization	ii
1. IDENTIFICATION	1
2. PURPOSE	2
3. USAGE	6
4. METHOD OR ALGORITHM	14
5. SOURCE LANGUAGE LISTING	18
6. COMPARISON	34
7. TEST METHOD AND RESULTS	34
8. REMARKS	42
9. ACKNOWLEDGMENTS	42

ABSTRACT

Two Fortran subroutines (BICUB1 and BICUB2) which perform bicubic spline interpolation of a tabulated function of two variables are described. Given the values $X(1), \dots, X(N)$ and $Y(1), \dots, Y(M)$ of two independent variables and the corresponding function values $\{U(I,J)=f(X(I), Y(J))\}$, $I=1, \dots, N$ and $J=1, \dots, M$ and certain normal derivatives (optional) along the boundaries of the x-y mesh, BICUB1 estimates the derivatives f_x , f_y , and f_{xy} at each (I,J) mesh point. If the normal derivatives along the mesh boundaries are unknown, BICUB1 estimates them using a moving third order two dimensional Lagrange interpolating polynomial. Given the coordinates (XPT,YPT) and the derivatives calculated by BICUB1, BICUB2 obtains the coefficients of the bicubic polynomial for the rectangular region of the mesh containing (XPT,YPT) and estimates the functional value $UPT=f(XPT,YPT)$. In effect, the routines pass a twice continuously differentiable piecewise bicubic polynomial, $u(x,y) \in C^2$, through the given functional values.

PROBLEM STATUS

This is a final report on one phase of a continuing problem.

AUTHORIZATION

NRL Problem S01-38

1.0 IDENTIFICATION

1.1 Title

Bicubic Spline Interpolation

1.2 Identification Name

E1-NRL-BICUBIC

1.3 Classification Code

E1-Interpolation and Approximations, Curve
Fitting

1.4 RCC Identification Number

E1001000

1.5 Entry Points

BICUB1
BICUB2

1.6 Programming Language

Language: 3600/3800 FORTRAN

Routine type: Subroutines

Operating System: DRUM SCOPE 2.1

1.7 Computer and Configuration

CDC 3800

1.8 Contributor or Programmer

John J. Cornyn¹
Information Processing Systems Branch (Code 5493)
Communications Sciences Division

¹Formerly with the Large Aperture Systems Branch, Code 8160,
Acoustics Division

1.9 Contributing Organization

NRL - Naval Research Laboratory,
Washington, D. C., 20375

1.10 Program Availability

1.10.1 Submittal: Program write-up, Fortran
source deck, source listing

1.10.2 On File: RCC Program Library

1.11 Verification

Several third degree polynomials were used to test BICUBIC; answers were good to at least nine significant figures. Higher degree polynomials were also used. Then, as expected, the results did not compare as well with the true values.

1.12 Date

26 February 1973

2.0 PURPOSE

2.1 Description of Routines

Let the values u_{ij} of a function $u(x,y)$ over a two dimensional domain be given at the mesh points (x_i, y_j) where $i = 1, \dots, N$; $j = 1, \dots, M$.

- (1) The first problem considered is the estimation of the normal derivatives along the boundaries of the mesh assuming they are unknown. In Figure 1, squares designate locations at which one needs to know the normal derivatives with respect to x , $p_{ij} = u_x(x_i, y_j)$. Circles designate locations at which one needs to know the normal derivatives with respect to y , $q_{ij} = u_y(x_i, y_j)$. Squares imbedded in circles designate locations where the normal derivatives with respect to both x and y , $S_{ij} = u_{xy}(x_i, y_j)$, are required, in addition to p_{ij} and q_{ij} . A solution to this problem will be given in the form of subroutines EDGES and LAGRAN.

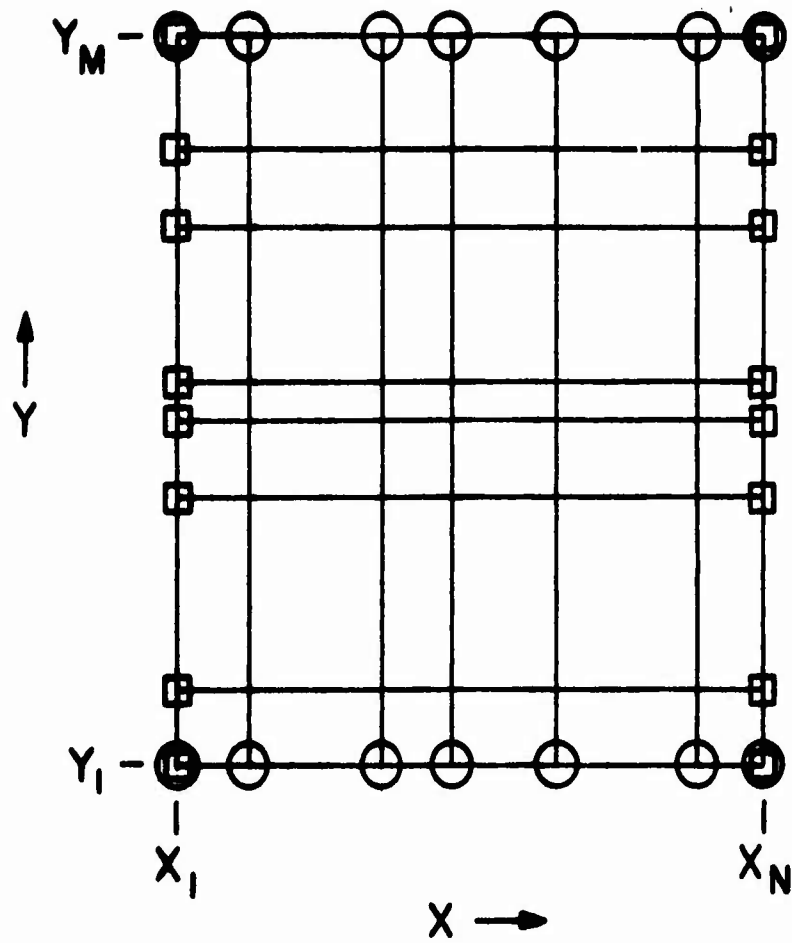


Figure 1

- (2) The second problem is, assuming the normal derivatives along the boundaries as discussed above have been given or estimated, to fit a "smooth function" $u(x,y) \in C^2$ (twice continuously differentiable) through these values.

The bicubic spline interpolation routines BICUB1, BICUB2, GETBP, AND SOLVIT described later implement a bicubic spline interpolation technique² [1] (see Section 3.15) which yields a piecewise bicubic polynomial $u(x,y)$. This function is defined in each rectangular cell,

$$R_{ij} : x_{i-1} \leq x \leq x_i ; y_{j-1} \leq y \leq y_j , \quad (1)$$

of the grid as

$$u(x,y) = C_{ij}(x,y) = \sum_{m,n=0}^3 \gamma_{mn}^{ij} (x-x_{i-1})^m (y-y_{j-1})^n \quad (2)$$

where $(x,y) \in R_{ij}$.

Individual Subroutine Functions

BICUB1 - A subroutine for calculating the normal derivatives at each mesh point.

BICUB2 - A subroutine for interpolating a value, $u(x,y)$, at any point (x,y) within the region subtended by the mesh.

EDGES - A subroutine for estimating the required normal derivatives along the boundaries assuming they have not been given, using a moving third order two dimensional Lagrange interpolating polynomial.

2. Note that Eq.(10) of [1] contains a typographical error and should read

$A(\Delta x_{i-1})K_{ij} A^T(\Delta y_{j-1}) = \Gamma_{ij}$, where T indicates a transpose. Also, the 8th character on line 4, p212 of [1] should be I and not 1.

GETBP - A subroutine for calculating the two dimensional difference arrays b and b' of Eq.(15) of [1].

LAGRAN - A subroutine for determining the value of a two dimensional Lagrange interpolating polynomial of degree m in x and n in y and its derivatives with respect to x , with respect to y , and with respect to both x and y at any intersection of a two dimensional mesh defined by $(m+1)$ levels of x_i , $i = a, \dots, a+m$ and $(n+1)$ levels of y_j , $j = c, \dots, c+n$. See Eq.(3).

SOLVIT - A subroutine for solving a linear system using Gaussian elimination as illustrated in Eqs.(15) and (16) of [1].

2.2 Problem Background

In the development of models of certain physical phenomena it is frequently useful to obtain a smooth functional representation of a quantity which is known at only a discrete set of points over a two-dimensional domain. Frequently, it is required that this function have continuous first and second derivatives. Once such a representation is obtained, it is possible to differentiate and integrate it in closed form.

A major problem with a Lagrange interpolating polynomial defined over a $N \times M$ mesh is that it must maintain $(N-2)$ in x and $(M-2)$ in y continuous derivatives and still pass through all of the data points. These requirements can and generally do lead to many large and unrealistic mountains and valleys in the interpolation surface, i.e., large interpolation errors. This problem is significantly reduced, but not completely eliminated by the bicubic spline (See Section 3.13).

A secondary, but still significant, problem is that when a large number of points is used, the evaluation and calculation of the Lagrange interpolating polynomial is costly and unreliable ([6] , p. 231). For further discussion see [8].

3.0 USAGE

3.1 Calling Sequence or Operational Procedure

BICUB1(N,M,NFAULT,X,Y,U,P,Q,S,MAX,W1,W2,W3,W4
W5,W6,W7)

BICUB2(XPT,YPT,UPT,NERROR,N,M,X,Y,U,P,Q,S)

3.2 Arguments, Parameters, and/or Initial Conditions

N -- number of x points at which the function was observed.

(N $\geq$ 4). TYPE INTEGER.

M -- number of y points at which the function was observed.

(M $\geq$ 4). TYPE INTEGER.

NFAULT -- a parameter that must be set to 1 if subroutine BICUB1 is to call subroutine EDGES to calculate the "required" normal derivatives along the boundaries of the mesh. If NFAULT is not set to 1, subroutine BICUB1 assumes the normal derivatives for the boundaries have already been entered into arrays, P,Q, and S, by the user's calling program. TYPE INTEGER.

The "required" normal derivatives are:

P(I,J) for I=1 and N; J=1 to M.

Q(J,I) for J=1 and M; I=1 to N.

S(J,I) for I=1 and N; J=1 and M.

- X -- the vector of distinct values of the first independent variable arranged in ascending order; (x_i for $i = 1$ to N). The minimum length of X is 4 and the maximum length is determined by the amount of core available. DIMENSION X(N). TYPE REAL.
- Y -- the vector of distinct values of the second independent variable arranged in ascending order. (y_j for $j = 1$ to M). The minimum length of Y is 4 and the maximum length is determined by the amount of core available. DIMENSION Y(M). TYPE REAL.
- U -- the matrix of function values corresponding to X and Y, i.e., $U(I,J)$ is u_{ij} of Section 2.1. DIMENSION U(N,M). TYPE REAL.
- P -- the matrix of normal derivatives with respect to x corresponding to X and Y; i.e., $P(I,J)$ is $U_x(X_i, Y_j)$. If NDFault is not 1, $P(I,J)$ for ($I = 1$ and N ; $J = 1$ to M) are required input. If NDFault is set to 1, no values of P are required as they will be calculated by subroutine EDGES. DIMENSION P(N,M). TYPE REAL.
- Q -- the matrix of normal derivatives with respect to y corresponding to X and Y; i.e., $Q(J,I)$ is $U_y(X_i, Y_j)$. Note the inversion of the J and I indices in Q. If NDFault is not 1, $Q(J,I)$ for ($J = 1$ and M ; $I = 1$ to N) are required input. If NDFault is set to 1, no values of Q are required as they will be calculated by subroutine EDGES. DIMENSION Q(M,N). TYPE REAL.
- S -- the matrix of normal derivatives with respect to both x and y corresponding to X and Y; i.e., $S(J,I)$ is $U_{xy}(X_i, Y_j)$. Note the inversion of indices I and J in S. If NDFault is not 1, $S(J,I)$ for ($I = 1$ and N ; $J = 1$ and M) are required input. If NDFault is set to 1, no values of S are required as they will be calculated by subroutine EDGES. DIMENSION S(M,N). TYPE REAL.

MAX -- the greater of N and M. TYPE INTEGER.

W1,...,W7 -- seven arrays which are used as working areas by subroutine BICUB1. Each of these arrays must be dimensioned to MAX words in the user's program. The user does not assign values to these arrays. TYPE REAL.

XPT -- the X coordinate of the point to be interpolated. TYPE REAL.

YPT -- the Y coordinate of the point to be interpolated. TYPE REAL.

UPT -- the interpolated value to be obtained. TYPE REAL.

NERROR -- an error indicator. If the point (XPT,YPT) does not lie within the mesh, NERROR will be set to 1 by BICUB2; otherwise it will remain set to 0. IF NERROR is returned as 1, the interpolated value is set to -0.0 and an error message is printed. See Section 3.5. TYPE INTEGER.

3.3 Space Required (Decimal and Octal)

3.3.1. Unique Storage (exclusive of system library)

<u>Subroutine</u>	<u>Decimal</u>	<u>Octal</u>
BICUB1	1083	2073
BICUB2	570	1072
EDGES	474	732
GETBP	183	267
LAGRAN	588	1114
SOLVIT	129	201
TOTAL	3027	5723

To interpolate a function over an N X M mesh the user's program is required to dimension the arrays X,Y,U,P,Q,S, and W1 through W7. These arrays require a total of

$4NM + N + M + 7K$ words,
where K is the greater of N and M.

3.3.2. Common Blocks

None.

3.3.3. Temporary Storage

Once subroutine BICUB1 has been called and the elements of the U,P,Q,S arrays have been determined, the space occupied by the working arrays W1,W2,...,W7 can be used for other purposes. That is, the 7K term of the above equation may be deleted.

3.4 Messages and Instructions to the Operator

None.

3.5 Error Returns, Messages, and Codes

Subroutine BICUB1 may print out the following error messages:

"ERROR - THE Y VECTOR IS NOT ARRANGED PROPERLY.
ERROR DETECTED BY BICUB1.
THE Y VECTOR IS (listed)"

"ERROR - THE X VECTOR IS NOT ARRANGED PROPERLY.
ERROR DETECTED BY BICUB1.
THE X VECTOR IS (listed)."

"ERROR - THE PARAMETER MAX OF SUB. BICUB1 WAS SET
TO __. IT SHOULD BE __."

"ERROR - THE X VECTOR HAS __POINTS AND THE MINIMUM
ALLOWED IS 4."

"ERROR - THE Y VECTOR HAS __POINTS AND THE MINIMUM
ALLOWED IS 4."

Subroutine BICUB2 will print out one or more of the following messages if an attempt is made to interpolate a point beyond the boundaries of the mesh:

"ERROR - XPT OUT OF BOUNDS
DETECTED BY SUB. BICUB2"

"ERROR - YPT OUT OF BOUNDS
DETECTED BY BICUB2"

3.6 Informative Messages to the User

None.

3.7 Input

No data are input. See Section 3.2.

3.8 Output

- (1) Completion of the P, Q and S arrays.
- (2) The value of the function $u(x,y)$ for any given (x,y) within the domain.

3.9 Formats

Not applicable.

3.10 External Routines and Symbols

BICUB1	- EDGES	(deck)
	GETBP	"
	SOLVIT	"
EDGES	- LAGRAN	"

3.11 Timing

The time required by subroutine BICUB1 is dependent upon the mesh size and whether or not the normal derivatives along the boundary are known. In the example of Section 7.0 BICUB1 took approximately 23 milliseconds for a 5 x 6 mesh when the boundary derivatives were known and approximately 135 milliseconds when they were unknown.

The time required for a call to BICUB2 is dependent on the mesh size. In the example of Section 7.0 an average call took about 3 milliseconds.

These time estimates should be considered very rough because of the method used to obtain them and the inaccuracies of the timing function (TIMELEFT) used.

3.12 Accuracy

An excellent discussion of the errors involved in bicubic spline interpolation is given in the paper by G. Birkhoff and C. de Boor [8]. Here, we will simply mention that for the special case of a 4 x 4 mesh, the interpolated values $u(x,y)$ will agree exactly, as they must, with those obtained from a third order two dimensional Lagrange interpolating polynomial. In this case, the remainder term is well known [3]. The final accuracy is dependent upon both discretion and rounding errors. A rough order of magnitude for these errors may be obtained from Section 7.0. The reader is referred to [4 and 5] for further discussion.

3.13 Cautions to Users

If the values $[u_{ij}]$ are highly variable along i or j , the interpolation surface may be forced to have unusually high mountains and deep valleys in order to maintain two continuous derivatives and still pass through all the data points. In fact, some interpolated values of u may be so large or so small as to be physically unrealistic. Whether or not this is the case will depend on the particular problem. The author has found that plotting several interpolated values, between and together with the given values, along a fixed direction in the x - y plane is helpful in detecting such conditions. In any event, care should be taken as the interpolation process could cause a physical model to generate faulty predictions.

3.14 Program Deck Structure

7 JOB card
9

7 FTN card
9

Users program (containing calls to BICUB1 and BICUB2)

Subroutine BICUB1	}	E ₁ - NRL - BICUBIC
Subroutine BICUB2		
Subroutine EDGES		
Subroutine GETBP		
Subroutine LAGRAN		
Subroutine SOLVIT		
SCOPE card		

7 LOAD card
9

7 RUN card
9
EØF

3.15 References

- [1]. C.de Boor, "Bicubic Spline Interpolation", J. of Mathematics and Physics, 41, 212-218 (1962).
- [2]. B. Carnahan, H. Luther, and J. Wilkes, Applied Numerical Methods, (Wiley and Sons, New York, 1969), Chapter 1.
- [3]. B. Carnahan, et. al., p65, problems 1.35 and 1.38.
- [4]. J. H. Wilkinson, Rounding Errors in Algebraic Processes, (Prentice Hall, N. J., 1968).
- [5]. J. M. Ortega, Numerical Analysis (A Second Course). (Academic Press, New York, 1972) Chapters 1, 7 and 9.
- [6]. C.de Boor and S. D. Conte, Elementary Numerical Analysis: An Algorithmic Approach, 2nd ed., (McGraw Hill, New York, 1972). Pages 231-240 describe one dimensional cubic spline interpolation.
- [7]. C. Price, "Table Lookup Techniques", Computing Surveys, 3, No. 2 (June 1971) pp53-56.

- [8]. G. Birkhoff and C. R. de Boor, "Piecewise Polynomial Interpolation and Approximation", in Approximation of Functions, H. Garabedian (editor), (Elsevier Publishing Co., Amsterdam, 1965).
- [9]. C. de Boor, Private communication.
- [10]. H. Späth, "Algorithm 10, Two Dimensional Smooth Interpolation", Computing 4, 178-182 (1962). (In German).
- [11]. H. Späth, "Correction to Algorithm 10", Computing 8, 200-201 (1971). (In German).

4.0 METHOD OR ALGORITHM

4.1 Subroutine BICUB1

We begin by considering the problem of estimating the required boundary derivatives (see Section 3.2) under the assumption that they are unknown.

Consider a 3rd order two dimensional Lagrange interpolating polynomial over a moving (a and c are variable) 4 x 4 submesh, i.e.,

$$v(x,y) = \sum_{i=a}^b \sum_{j=c}^d X_i(x) Y_j(y) u_{ij} \quad (3)$$

where $b = a + 3$, $d = c + 3$,

u_{ij} is defined in Section 2.1 ,

$$X_i(x) = \prod_{\substack{k=a \\ k \neq i}}^b \frac{x - x_k}{x_i - x_k} ,$$

$$Y_j(y) = \prod_{\substack{k=c \\ k \neq j}}^d \frac{y - y_k}{y_j - y_k} .$$

Differentiating Eq. (3) we can obtain closed form expressions for $v_x(x,y)$, $v_y(x,y)$, and $v_{xy}(x,y)$.

Subroutine LAGRAN can evaluate these expressions at any mesh point.

Basically, subroutine EDGES moves the 4 x 4 submesh of Eq. (3) along the boundaries of Figure 1 while calling subroutine LAGRAN to obtain the required normal derivatives.

Once the required boundary derivatives are obtained, the rest of the derivatives, p_{ij} , q_{ij} , and s_{ij} , are obtained for each ij mesh point by using the algorithm described in [1] , pages 217-218.

4.2 Subroutine BICUB2

To interpolate a value, $u(x,y)$, at the point (x,y) , subroutine BICUB2 begins by performing a binary search to determine the ij rectangle in which the point lies. A binary search [7] is used on the assumption that for most problems the X and Y vectors will be large enough to exceed the break even point between sequential and binary searches (about 50 points).

Once i and j are determined, a cubic Hermite basis is used to evaluate $u(x,y)$. That is,

$$u(x,y) = \sum_{r,s=1}^4 Q_{rs}^{ij} \phi_r(x,h) \psi_s(y,k) \quad (4)$$

where

$$Q^{ij} = \begin{bmatrix} u_{i-1,j-1} & u_{i-1,j} & q_{i-1,j-1} & q_{i-1,j} \\ u_{i,j-1} & u_{i,j} & q_{i,j-1} & q_{i,j} \\ p_{i-1,j-1} & p_{i-1,j} & s_{i-1,j-1} & s_{i-1,j} \\ p_{i,j-1} & p_{i,j} & s_{i,j-1} & s_{i,j} \end{bmatrix} \quad (5)$$

$$h = x_i - x_{i-1} \quad (6)$$

$$x' = x - x_{i-1} \quad (7)$$

$$k = y_j - y_{j-1} \quad (8)$$

$$y' = y - y_{j-1} \quad (9)$$

$$\phi_1(x,h) = 1 + \left(\frac{x'}{h}\right)^2 \left(\frac{2x'}{h} - 3\right) \quad (10)$$

$$\begin{aligned} \phi_2(x,h) &= \left(\frac{x'}{h}\right) \left(3 - \frac{2x'}{h}\right) \\ &= 1 - \phi_1(x,h) \end{aligned} \quad (11)$$

$$\phi_3(x,h) = \left(\frac{x'}{h}\right) \frac{(h-x')^2}{h} \quad (12)$$

$$\begin{aligned} \phi_4(x,h) &= \left(\frac{x'}{h}\right)^2 (x' - h) \\ &= -\phi_3(h-x) \end{aligned} \quad (13)$$

The functions ψ_s , for $s = 1$ to 4, are obtained by replacing ϕ , x' , and h by ψ , y' , and k respectively in Eqs. (10) through (13).

This procedure requires the storage of four values (u_{ij} , p_{ij} , q_{ij} , and S_{ij}) for each mesh point.

And the evaluation of $u(x,y)$ by BICUB2 requires 32 additions/subtractions and 27 multiplications/divisions. Assuming a multiplication/division is equivalent in time to three additions/subtractions, this results in 113 "operations".

An alternative approach, not taken in this report, would be to convert to a local power basis.

In particular, calculate the 16 values of γ_{mn}^{ij} (see Eq.(2)) for each ij rectangle, as described in [1], and store them for each ij rectangle of the mesh. This would require approximately four times as much storage as the above method. Also about 52 additions/subtractions and 76 multiplications/divisions (270 "operations") would be required to obtain the 16 coefficients γ_{mn}^{ij} , for $m,n = 0$ through 3.

The advantage of this approach is that only about 19 additions/subtractions and 15 multiplications/divisions (64 "operations") would be required by BICUB2 for the evaluation. This suggests that, for very fine mesh evaluations, in which every bicubic polynomial is evaluated on the average six or more times, it is more efficient to obtain the local power basis coefficients, γ_{mn}^{ij} , for the entire mesh once and save them. Of course, this results in a rather severe penalty in storage.

In contrast, by using the Hermite basis, as we've done here, evaluation costs slightly more work but considerably less storage.

In summary, it seems best for most applications to save only the partials at the mesh points and use the Hermite basis approach.

5.0 SOURCE LANGUAGE LISTING

```

IDENT NUMBER E1001000
TITLE = BICUBIC SPLINE INTERPOLATION
IDENT NAME = E1-NRL-BICUBIC
LANGUAGE = FORTRAN
COMPUTER = CDC-3600

CONTRIBUTOR = JOHN IJ CORNYN
               CODE 5493
               INFORMATION PROCESSING SYSTEMS BRANCH
               COMMUNICATIONS SCIENCES DIVISION
ORGANIZATION = NRL = NAVAL RESEARCH LABORATORY
               WASHINGTON, D.C. 20375

DATE = FEBRUARY 1973

*****
PURPOSE

SUBROUTINES BICUB1 AND BICUB2 PERFORM BICUBIC SPLINE
INTERPOLATION OF A TABULATED FUNCTION OF TWO VARIABLES.

*****
REFERENCE = DE BOOR, C. = BICUBIC SPLINE INTERPOLATION
              IJ. OF MATHEMATICS AND PHYSICS,
              VOL 41, PP 212-218, (1962)

*****
SUBROUTINE CALLING ORDER ---

THE USER'S PROGRAM CALLS SUBROUTINES BICUB1 AND/OR BICUB2.
SUBROUTINE BICUB1 CALLS SUBROUTINES EDGES, GETOP AND SOLVIT.
SUBROUTINE EDGES CALLS SUBROUTINE LAGRAN.
SUBROUTINES BICUB2, GETOP, SOLVIT, AND LAGRAN DO NOT CALL
ANY OTHER SUBROUTINES.

*****
*****
DICTIONARY FOR BICUBIC SPLINE INTERPOLATION
*****
DICTIONARY CODE = THE SECOND LETTER OF A LINE OF THE DICTIONARY
                  INDICATES THE TYPE OF ENTRY BEING DESCRIBED.
                  THIS CODE IS AS FOLLOWS
DICTIONARY CODE  TYPE OF VARIABLE
CA              ABBREVIATION
CI              INTEGER VARIABLE
CIA            INTEGER VARIABLE ARRAY
CR              REAL VARIABLE
CRA            REAL VARIABLE ARRAY
CS              ROUTINE
CT              WORD FROM TEXT

FOLLOWING THE DICTIONARY CODE IS A NUMBER WHICH INDICATES THE

```


C	SUBROUTINE NUMBER(S) IN WHICH THE ENTRY APPEARS	570
C	SUBROUTINE NAME SUBROUTINE NO.	580
C	BICUB1 1	590
C	BICUB2 2	600
C	EDGES 3	610
C	GETOP 4	620
C	LAGRAN 6	630
C	SOLVIT 7	640
C	CHECK 8	650
C		660
C	FOR EXAMPLE, CR2,3 MEANS THE ENTRY APPEARS IN SUBROUTINES	670
C	BICUB2 AND EDGES AND IS A REAL VARIABLE,	680
C		690
C		700
C	*****	710
C	CS1,8 BICUB1 • A SUBROUTINE FOR CALCULATING THE NORMAL	720
C	DERIVATIVES AT EACH MESH POINT,	730
C	CS2,8 BICUB2 • A SUBROUTINE FOR INTERPOLATING A VALUE	740
C	UPT AT ANY POINT (XPT,YPT) WITHIN OR ON	750
C	THE MESH,	760
C	CRA1,4,7 BIM1(I) • HOLDS THE ELEMENT B(I,I-1) OF EQ.15 OF REF 1	770
C	CRA1,4,7 BIP1(I) • HOLDS THE ELEMENT B(I,I+1) OF EQ.15 OF REF 1	780
C	CRA1,4,7 BP(I) • HOLDS THE ELEMENT B(I,I) OF EQ. 15 OF REF 1,	790
C	THE ARRAYS BIM1,BIP1, AND BP ARE USED TO HOLD	800
C	THE B AND B PRIME DIFFERENCE ARRAYS GIVEN	810
C	IN THE REFERENCE, THESE ARRAYS ARE USED FOR	820
C	GAUSSIAN ELIMINATION,	830
C	CS8 CHECK • A SAMPLE PROGRAM TO CHECK OUT THE	840
C	BICUBIC SPLINE PACKAGE	850
C	CR2 CX1 • X DEPENDENT TEMPORARY VARIABLE	860
C	CR2 CX2 • AN X DEPENDENT TEMPORARY VARIABLE	870
C	CR2 CX3 • AN X DEPENDENT TEMPORARY VARIABLE	880
C	CR2 CY1 • A Y DEPENDENT TEMPORARY VARIABLE	890
C	CR2 CY2 • A Y DEPENDENT TEMPORARY VARIABLE	900
C	CR2 CY3 • A Y DEPENDENT TEMPORARY VARIABLE	910
C	CRA1,7 D(I) • CORRESPONDS TO THE D MATRIX OF EQ. 13 OF THE	920
C	REFERENCE	930
C	CR6 DENOM • A DENOMINATOR	940
C	CR6 DIFF • DIFFERENCE BETWEEN INTERPOLATED AND EXACT VALUES	950
C	CRA1,7 DP(I) • CORRESPONDS TO THE D PRIME VECTOR OF EQ.16	960
C	OF THE REF.	970
C	CR6 DPT • THE ANSWER GIVEN BY LAGRAN	980
C	CRA1 DS(I) • AN ARRAY FOR HOLDING DIFFERENCES IN SOLVING	990
C	EQ 12 AND 14 OF REF.(DEBORR)	1000
C	CR4 DXL • DELTA X LEFT	1010
C	CR4 DXR • DELTA X RIGHT	1020
C	CS1,3 EDGES • A SUBROUTINE FOR ESTIMATING THE REQUIRED	1030
C	NORMAL DERIVATIVES ALONG THE MESH BOUNDARIES	1040
C	ASSUMING THEY HAVE NOT BEEN GIVEN, USING	1050
C	A THIRD ORDER TWO DIMENSIONAL LAGRANGE	1060
C	INTERPOLATING POLYNOMIAL	1070
C	CS1,4 GETOP • A SUBROUTINE FOR CALCULATING THE TWO	1080
C	DIMENSIONAL DIFFERENCE ARRAYS B AND B PRIME	1090
C	OF EQ 15 OF REF.(DEBORR)	1100
C	CI1,8 I • INDEX, GENERALLY USED FOR THE X ARRAY	1110
C	CI1,4,2,7 IM1 • I MINUS ONE	1120

C11,4,7	IP1	• I PLUS ONE	1130
C14	IP2	• I PLUS 2	1140
C11=3,6=7,8	J	• INDEX GENERALLY USED FOR THE Y ARRAY	1190
C11,2	JM1	• J MINUS ONE	1140
C11	JP1	• J PLUS ONE	1170
C14,8	K	• AN INDEX	1190
C12	KH	• UPPER LIMIT FOR BINARY SEARCH	1190
C12	KL	• LOWER LIMIT FOR BINARY SEARCH	1200
C11,6	L	• A COUNTER	1210
C11	L1	• L PLUS ONE	1220
C83,6	LAGRAN	• A SUBROUTINE FOR DETERMINING THE VALUE OF A TWO DIMENSIONAL LAGRANGE INTERPOLATING POLYNOMIAL OF ARBITRARY DEGREE IN X AND Y, AND ITS DERIVATIVES WITH RESPECT TO X, WITH RESPECT TO Y, AND WITH RESPECT TO BOTH X AND Y AT ANY INTERSECTION POINT OF A X AND Y AT ANY INTERSECTION POINT OF A TWO DIMENSIONAL MESH.	1230 1240 1250 1260 1270 1280 1290 1300
C11=3, 6,8	M	• THE NUMBER OF Y POINTS AT WHICH THE FUNCTION WAS OBSERVED, MUST BE GREATER THAN 3,	1310 1320
C11,8	MAX	• THE GREATER OF N AND M,	1330
C			1340
C11	MAXS	• THE VALUE MAX SHOULD HAVE BEEN SET TO,	1350
C13,6	MF	• NUMBER OF THE FINAL POINT ON Y AXIS TO BE USED IN LAGRANGE INTERPOLATION	1360 1370
C			1380
C11,8	MM1	• M MINUS ONE	1390
C11	MM2	• M MINUS TWO	1390
C11	MMIN	• THE SMALLEST VALUE M CAN TAKE,	1400
C16	MPT	• THE VALUE OF THE Y VECTOR TO BE USED IN LAGRANGE INTERPOLATION	1410 1420
C			1430
C13,6	MS	• NUMBER OF THE STARTING POINT ON Y AXIS TO BE USED IN LAGRANGE INTERPOLATION	1440
C			1450
C11=3, 6,8	N	• THE NUMBER OF X POINTS AT WHICH THE FUNCTION WAS OBSERVED,	1460
C			1470
C17,4	N	• NUMBER OF ELEMENTS IN LINEAR SYSTEM	1480
C11,8	NDFAULT	• A PARAMETER WHICH MUST BE SET TO 1 IF SUBROUTINE BICUB1 IS TO CALL EDGES TO CALCULATE THE REQUIRED NORMAL DERIVATIVES ALONG THE BOUNDARIES OF THE MESH, IF NDFAULT IS NOT SET TO 1, BICUB1 ASSUMES THE NORMAL DERIVATIVES FOR THE BOUNDARIES HAVE ALREADY BEEN ENTERED INTO ARRAYS P,Q, AND S BY THE USERS CALLING PROGRAM,	1490 1500 1510 1520 1530 1540
C			1550
C12,8	NERROR	• AN ERROR INDICATOR, IF THE POINT (XPT,YPT) DOES NOT LIE WITHIN THE MESH, NERROR WILL BE SET TO 1 BY BICUB2 OTHERWISE IT WILL REMAIN SET TO 0, IF NERROR IS RETURNED AS 1, AN INTERPOLATED VALUE IS NOT COMPUTED,	1560 1570 1580 1590
C			1600
C13,6	NF	• NUMBER OF THE FINAL POINT ON THE X AXIS TO BE USED IN LAGRANGE INTERPOLATION,	1610
C			1620
C11,4,7,8	NM1	• N MINUS ONE	1630
C11,4	NM2	• N MINUS TWO	1640
C14	NM3	• N MINUS 3	1650
C11	NMIN	• THE SMALLEST VALUE N CAN TAKE	1660
C16	NPT	• THE VALUE OF THE X VECTOR TO BE USED IN LAGRANGE INTERPOLATION, XPT=X(NPT)	1670
C			1680
C13,6	NS	• NUMBER OF THE STARTING POINT ON THE X AXIS TO	

C			BE USED IN LAGRANGE INTERPOLATION,	1690
C16	NTYPE		• SET TO 1 IF LAGRAN IS TO INTERPOLATE A	1700
C			VALUE OF THE FUNCTION AT THE (NPT,MPY)MESH	1710
C			POINT,	1720
C			SET TO 2 TO GET PARTIAL DERIVATIVE W/R TO X,	1730
C			SET TO 3 TO GET PARTIAL DERIVATIVE W/R TO Y,	1740
C			SET TO 4 TO GET PARTIAL DERIVATIVE W/R TO	1750
C			BOTH X AND Y,	1760
CRA1=3, 6.0	P		• THE NORMAL DERIVATIVES OF U WITH RESPECT TO X	1770
C				1780
CR2	P11J1		• P(1=1,J=1)	1790
CR2	P1J1		• P(1,J=1)	1800
CRA1=3, 6.0	O		• THE NORMAL DERIVATIVES OF U WITH RESPECT TO Y,	1810
CR1	R		• A TEMPORARY VARIABLE USED IN FORMING	1820
C			THE C MATRIX	1830
CS0	RANF		• A CDC 3000 RANDOM NUMBER GENERATOR	1840
C			GENERATES UNIFORMLY DISTRIBUTED RANDOM	1850
C			NUMBERS BETWEEN 0 AND 1	1860
CR1	RINV		• THE INVERSE OF R	1870
CRA1=3, 6.0	S		• THE NORMAL DERIVATIVES OF U WITH RESPECT TO	1880
C			BOTH X AND Y,	1890
CS9,1	SOLVIT		• A SUBROUTINE FOR SOLVING A LINEAR SYSTEM	1900
C			USING GAUSSIAN ELIMINATION AS ILLUSTRATED	1910
C			IN BC, 15 AND 16 OF REF (DEBOOR),	1920
CRA1	STEMP		• A TEMPORARY ARRAY USED IN SOLVING FOR S	1930
CR6	SUM		• A SUMMATION	1940
CR6	SUMX		• A SUMMATION ALONG X	1950
CR6	SUMY		• A SUMMATION ALONG Y	1960
CR6	T		• VARIABLE FOR TIME CALCULATION	1970
CR2	T1		• TEMPORARY VARIABLE IN EVALUATION	1980
CR2	T1M1		• TEMPORARY VARIABLE IN EVALUATION	1990
CS0	T1MELEFT		• A CDC-300 SYSTEM LIBRARY FUNCTION	2000
C			GIVING THE NUMBER OF SECONDS LEFT UNTIL THE	2010
C			TIME LIMIT FOR THE JOB WILL BE REACHED,	2020
CR2	TP1		• TEMPORARY VARIABLE IN EVALUATION	2030
CR2	TP1M1		• TEMPORARY VARIABLE IN EVALUATION	2040
CRA1=3, 6.0	U		• THE ARRAY OF FUNCTION VALUES CORRESPONDING	2050
C			TO X AND Y,	2060
CR6	UEXACT		• VALUE OF U OBTAINED BY EVALUATING THE	2070
C			ARBITRARY POLYNOMIAL AT THE COORDINATES (X1,Y1)	2080
CR2	U11J1		• U(1=1,J=1)	2090
CR2	U1J1		• U(1,J=1)	2100
CR6	UINT		• VALUE OF U OBTAINED BY BICUBIC SPLINE	2110
C			INTERPOLATION AT THE COORDINATES (X1,Y1)	2120
CR2	UPT		• THE INTERPOLATED VALUE AT (XPT,YPT)	2130
CRA4	V		• A VECTOR OF VALUES	2140
CR6	W1		• WORKING ARRAYS DIMENSIONED TO	2150
CR6	W2		MAX WORDS IN THE USERS PROGRAM,	2160
CR6	W3			2170
CR6	.			2180
CR6	.			2190
CR6	W7			2200
CRA1=3, 6.0	X		• THE VECTOR OF DISTINCT VALUES OF THE FIRST	2210
C			INDEPENDENT VARIABLE ARRANGED IN ASCENDING	2220
C			ORDER, THE MINIMUM LENGTH OF X IS 4 AND THE	2230
C			MAXIMUM LENGTH IS DETERMINED BY THE AMOUNT	2240

C		OF CORE AVAILABLE,	2250
CR1	X32	• X(3)MINUS X(2)	2260
CR6,6	XI	• THE ITH VALUE OF X	2270
CR2	XIM1	• X(I-1)	2280
CR2	XX10H	• TEMPORARY VARIABLE	2290
CR1	XN12	• X(NM1)=X(NM2)	2300
CR6	XP	• X(NPT)	2310
CR2	XPT	• THE X CO-ORDINATE OF THE POINT TO BE	2320
C		INTERPOLATED,	2330
CRA1=3, 6.8 Y		• THE VECTOR OF DISTINCT VALUES OF THE SECOND	2340
C		INDEPENDENT VARIABLE ARRANGED IN ASCENDING	2350
C		ORDER, THE MINIMUM LENGTH OF Y IS 4	2360
C		AND THE MAXIMUM LENGTH IS DETERMINED BY	2370
C		THE AMOUNT OF CORE AVAILABLE,	2380
CR1	Y32	• Y(3)=Y(2)	2390
CR6	YI	• THE ITH VALUE OF Y	2400
CR6	YJ	• Y(IJ)	2410
CR2	YJM1	• Y(IJ-1)	2420
CR1	YM12	• Y(MM1)=Y(MM2)	2430
CR2	YMY10K	• TEMPORARY VARIABLE	2440
CR6	YP	• Y(NPT)	2450
CR2	YPT	• THE Y CO-ORDINATE OF THE POINT TO BE	2460
C		INTERPOLATED,	2470
CRA7	Z	• THE SOLUTION VECTOR FOR THE LINEAR SYSTEM	2480
C			2490
C	*****		2500
C	*****		2510
C			2520
	SUBROUTINE BICUB1(N,M,NDF,AULT,X,Y,U,P,Q,S,MAX,DP,BIP1,BIM1,D,DS,		2530
	X STMP,DP)		2540
C			2550
C	BICUBIC SPLINE INTERPOLATION		2560
C	THIS SUBROUTINE CALCULATES THE PARTIAL DERIVATIVES FOR THE MESH		2570
C			2580
	DIMENSION X(N),Y(M),L(N,M),P(N,M),Q(M,N),S(M,N)		2590
	DIMENSION BP(MAX),BIP1(MAX),BIM1(MAX),D(MAX),DS(MAX),STMP(MAX),		2600
	X DP(MAX)		2610
C			2620
	DATA(NM,NB4),(MMIN=4)		2630
C			2640
C	CHECK TO SEE IF THE MAX PARAMETER WAS SET CORRECTLY		2650
C			2660
C			2670
C	MAXS IS WHAT MAX SHOULD BE		2680
C			2690
	MAXS=N		2700
	IF(N,LT,M) MAXS=M		2710
	IF(MAXS,NE,MAX) GO TO 900		2720
C			2730
C			2740
C	DETERMINE WHETHER THE X AND Y VECTORS ARE WITHIN LIMITS		2750
C			2760
C			2770
	IF(N,LT, NMIN) GO TO 905		2780
	IF(M,LT, MMIN) GO TO 907		2790
	NM1=N-1		2800

NM2=2	2810
NM1=1	2820
NM2=2	2830
C	2840
C DETERMINE WHETHER THE X AND Y ARRAYS CONTAIN DISTINCT ELEMENTS AND	2850
C ARE ARRANGED IN ASCENDING ORDER.	2860
C	2870
DO 200 I=1,NM1	2880
IF(X(I),GE,X(I+1)) GO TO 911	2890
200 CONTINUE	2900
C	2910
DO 210 J=1,NM1	2920
IF(Y(J),GE,Y(J+1)) GO TO 912	2930
210 CONTINUE	2940
C	2950
C DETERMINE THE EDGE BOUNDARIES FOR P,Q,AND S IF REQUESTED	2960
C	2970
IF(INDFAULT,EO,1) CALL EDGES(N,M,X,Y,U,P,Q,S)	2980
C	2990
C GET THE DIFFERENCE ARRAY,D, FOR THE X VECTOR AND ALSO	3000
C GET THE S PRIME ARRAY,SP, FROM THE S ARRAY	3010
C	3020
CALL GETBP(N,X,SP,BIP1,BIM1)	3030
C	3040
C NOW SOLVE FOR THE PARTIALS W/R TO X WHICH WERE NOT GIVEN	3050
C	3060
X32=X(3)-X(2)	3070
XN12=X(NM1)-X(NM2)	3080
DO 30 J=1,M	3090
C	3100
C SET UP THE D VECTOR FOR EQ(11) OF REF	3110
C	3120
DO 35 IM1=1,NM2	3130
IP1=1	3140
IP1=1	3150
R=(X(IP1)-X(IM1))/(X(IP1)-X(1))	3160
RINV=1/R	3170
D(IM1)=3,0(R*(U(IP1,J)-U(1,J))+RINV*(U(1,J)-U(IM1,J)))	3180
IF(J,EO,1,OR,J,EO,M) 34,35	3190
C	3200
34 D(IM1)=3,0(R*(Q(J,IP1)-Q(J,1))+RINV*(Q(J,1)-Q(J,IM1)))	3210
C	3220
C NOTE Q AND S ARRAYS ARE STORED AS Q(J,I) AND S(J,I) RATHER THAN	3230
C Q(I,J) BECAUSE OF FORTRAN CONVENTIONS FOR STORING ARRAYS COLUMNWISE	3240
C	3250
35 CONTINUE	3260
C	3270
C ADD ADDITIONAL TERMS TO THE FIRST AND LAST ELEMENTS OF THE D VECTOR	3280
C	3290
D(1)=D(1)+X32*P(1,J)	3300
D(NM2)=D(NM2)+XN12*P(N,J)	3310
C	3320
C	3330
C NOW SOLVE LINEAR SYSTEMS FOR EQ(11) OF REFERENCE,	3340
C	3350
CALL SOLVIT(NM2,P(2,J),D,SP,BIP1,BIM1,DP)	3360

C	IF(J, EQ, 1, OR, J, EQ, M) 37,30	3370
C		3380
C	ADD ADDITIONAL TERMS TO THE DS ARRAY	3390
C		3400
	37 DS(1)=DS(1)+X32*S(J,1)	3410
	DS(NM2)=DS(NM2)+XN12*S(J,N)	3420
C		3430
C	NOW SOLVE LINEAR SYSTEM FOR EQ(12)	3440
C		3450
	CALL SOLVIT(NM2,STEMP,DS,BP,BIP1,BIM1,DP)	3460
C		3470
C	MOVE VALUES FROM THE TEMPORARY ARRAY INTO THE PARTIAL	3480
C	ARRAY FOR THE CROSS TERMS, WE DO THIS BECAUSE OF FORTRAN ARRAY	3490
C	STORAGE CONVENTIONS	3500
C		3510
	DO 10 L=1,NM2	3520
	L1=L+1	3530
	S(J,L1)=STEMP(L)	3540
	10 CONTINUE	3550
C		3560
	30 CONTINUE	3570
C		3580
C	GET THE DIFFERENCE ARRAY ,B, FOR THE Y VECTOR AND THE B PRIME ARRAY	3590
C		3600
	CALL GETBP(M,Y,BP,BIP1,BIM1)	3610
C		3620
C	NOW GET THE PARTIALS WITH RESPECT TO Y	3630
C	WHICH WERE NOT GIVEN, I, E, MEMBERS OF THE Q ARRAY WHICH WERE NOT	3640
C	SPECIFIED	3650
C		3660
	Y32=Y(3)-Y(2)	3670
	YM12=Y(MM1)-Y(MM2)	3680
	DO 40 I=1,N	3690
C		3700
C	SET UP THE D MATRIX FOR EQ(13) OF REF	3710
C		3720
	DO 45 JM1=1,MM2	3730
	JBJM1=1	3740
	JP1=J+1	3750
	R=(Y(J)-Y(JM1))/(Y(JP1)-Y(J))	3760
	RINV=1/R	3770
	D(JM1)=3,0(R(U(I,JP1)-U(I,J))+RINV(U(I,J)-U(I,JM1)))	3780
	DS(JM1)=3,0(R(P(I,JP1)-P(I,J))+RINV(P(I,J)-P(I,JM1)))	3790
	45 CONTINUE	3800
C		3810
	D(1)=D(1)+Y32*Q(1,1)	3820
	D(MM2)=D(MM2)+YM12*Q(M,1)	3830
C		3840
	DS(1)=DS(1)+Y32*S(1,1)	3850
	DS(MM2)=DS(MM2)+YM12*S(M,1)	3860
C		3870
C	NOW CAN SOLVE THE LINEAR SYSTEM OF EQ(13) FOR THIS I	3880
C		3890
	CALL SOLVIT(MM2,Q(2,1),D,BP,BIP1,BIM1,DP)	3900
C		3910
		3920

C NOW SOLVE THE LINEAR SYSTEMS OF EQ(14) FOR THIS I	3930
C	3940
CALL SOLVIT(MM2,S(2,1),DS,BP,BIP1,BIM1,DP)	3990
40 CONTINUE	3940
C	3970
C	3980
RETURN	3990
C	4000
900 PRINT 903,MAX,MAXS	4010
STOP	4020
905 PRINT 906,N,NMIN	4030
STOP	4040
907 PRINT 908,M,MMIN	4050
STOP	4060
911 PRINT 915,(X(I),I=1,N)	4070
STOP	4080
912 PRINT 916,(Y(J),J=1,M)	4090
STOP	4100
C	4110
C -----FORMAT STATEMENTS -----	4120
C	4130
903 FORMAT(10X,'ERROR-THE PARAMETER MAX OF SUB, BICUB1 WAS SET TO ',	4140
X 15,' IT SHOULD BE ',15)	4150
906 FORMAT(10X,'ERROR-THE X VECTOR HAS ',13,	4160
X 'POINTS AND THE MINIMUM ALLOWED IS ',13)	4170
908 FORMAT(10X,'ERROR-THE Y VECTOR HAS ',13,	4180
X 'POINTS AND THE MINIMUM ALLOWED IS ',13)	4190
915 FORMAT(10X,'ERROR-THE X VECTOR IS NOT ARRANGED PROPERLY',//,	4200
X 10X,'ERROR DETECTED BY BICUB1',/,	4210
X 10X,'THE X VECTOR IS',/,	4220
X(5(2X,F12,4)))	4230
916 FORMAT(10X,'ERROR-THE Y VECTOR IS NOT ARRANGED PROPERLY',//,	4240
X 10X,'ERROR DETECTED BY BICUB1',/,	4250
X 10X,'THE Y VECTOR IS',/,	4260
X(5(2X,F12,4)))	4270
C	4280
END	4290

SUBROUTINE DICUB2(XPT,YPT,UP,T,NERROR,N,M,X,Y,U,P,Q,S)	4300
C	4310
C SUBROUTINE TO INTERPOLATE A VALUE,UP,T, AT LOCATION (XPT,YPT),	4320
C THIS SUBROUTINE ASSUMES THAT THE X AND Y VECTORS ARE LARGE ENOUGH	4330
C TO MAKE A BINARY SEARCH TECHNIQUE SUPERIOR TO A	4340
C SEQUENTIAL SEARCH TECHNIQUE,	4350
C NERROR IS SET TO 1 IF (XPT,YPT) IS OUTSIDE THE PROPER DOMAIN	4360
C	4370
DIMENSION X(N),Y(M),U(N,M),P(N,M),Q(M,N),S(M,N)	4380
C	4390
NERROR=0	4400
C	4410
C CONDUCT A BINARY SEARCH FOR I	4420
C	4430
KH=1	4440
KL=1	4450
2 J=(KL+KH)/2	4460
3 IF(X(J)-XPT) 6,7,11	4470
11 IF(XPT-X(J-1)) 4,10,7	4480
4 KH=J	4490
5 IF(KH-KL=1) 9,9,2	4500
6 KL=J	4510
7 00 TO 5	4520
9 IF(XPT-X(KH)) 10,8,13	4530
8 J=KH	4540
7 00 TO 7	4550
10 IF(XPT-X(KL)) 13,14,8	4560
14 J=KL	4570
7 00 TO 7	4580
18 J=J+1	4590
7 00 TO 7	4600
13 NERROR=1	4610
PRINT 12	4620
UP=0.0	4630
RETURN	4640
C	4650
C CONDUCT A BINARY SEARCH FOR J	4660
C	4670
7 KH=M	4680
KL=1	4690
20 J=(KL+KH)/2	4700
30 IF(Y(J)-YPT) 60,70,110	4710
110 IF(YPT-Y(J-1)) 40,100,70	4720
40 KH=J	4730
50 IF(KH-KL=1) 90,90,20	4740
60 KL=J	4750
7 00 TO 50	4760
90 IF(YPT-Y(KH)) 100,80,130	4770
80 J=KH	4780
7 00 TO 70	4790
100 IF(YPT-Y(KL)) 130,140,80	4800
140 J=KL	4810
7 00 TO 70	4820
180 J=J+1	4830
7 00 TO 70	4840
130 NERROR=1	4850


```

      PRINT 23
      UPT=0.0
      RETURN

C
C
C USE A CUBIC HERMITE BASIS TO EVALUATE THE BICUBIC POLYNOMIAL
C AT (XPT,YPT) :
C (XPT,YPT) LIES WITHIN THE RECTANGLE BOUNDED BY X(I),X(I+1),Y(J),
C AND Y(J+1).
C
70 IM1=1
   JM1=J
   XI=X(I)
   YJ=Y(J)
   XM1=X(IM1)
   YM1=Y(JM1)
   XHX10H=(XPT-XIM1)/(XI-XIM1)
   YHY10K=(YPT-YJM1)/(YJ-YJM1)
   CX10(3,2)=XHX10H*XHX10H**2
   CY10(3,2)=YHY10K*YHY10K**2
   CX20(XI,XPT)=XHX10H
   CY20(YJ,YPT)=YHY10K
   CX30=XHX10H**3
   CY30=YHY10K**3
   CX20CX20CX30
   CY20CY20CY30
   U1J1=U(I,JM1)
   U1J10U(IM1,JM1)
   P1J1=P(I,JM1)
   P1J10P(IM1,JM1)
   TIM1=U1J1*CY10(U(IM1,J)=U1J1)+CY20(JM1,IM1)=CY30(J,IM1)
   TI=U1J1*CY10(U(I,J)=U1J1)+CY20(JM1,I)=CY30(J,I)
   TPIM1P1J1*CY10(P(IM1,J)=P1J1)+CY20(JM1,IM1)=CY30(J,IM1)
   TP10P1J1*CY10(P(I,J)=P1J1)+CY20(JM1,I)=CY30(J,I)
   UPT=TIM1*CX10(TI=TIM1)+CX20*TPIM1=CX30*TP1
C
      RETURN
C
C *****
C      FORMAT STATEMENTS
C *****
C
12 FORMAT(10X,'ERROR=XPT OUT OF BOUNDS',/,
  X 10X,'DETECTED BY SUB, BICUB2')
23 FORMAT(10X,'ERROR=YPT OUT OF BOUNDS',/,
  X 10X,'DETECTED BY BICUB2')
C
      END

```

```

4860
4870
4880
4890
4900
4910
4920
4930
4940
4950
4960
4970
4980
4990
5000
5010
5020
5030
5040
5050
5060
5070
5080
5090
5100
5110
5120
5130
5140
5150
5160
5170
5180
5190
5200
5210
5220
5230
5240
5250
5260
5270
5280
5290
5300
5310
5320
5330

```

SUBROUTINE EDGES(N,M,X,Y,U,P,Q,S)	5340
C	5350
C IF THE USER REQUESTED DEFAULT EDGE CONDITIONS	5360
C THIS SUBROUTINE WILL DETERMINE THEM	5370
C USING A LAGRANGE INTERPOLATING POLYNOMIAL OF ORDER 3 X 3	5380
C	5390
C	5400
C THIS SUBROUTINE IS CALLED BY SUBROUTINE BICUR1.	5410
C THIS SUBROUTINE CALLS SUBROUTINE LAGRAN,	5420
C	5430
DIMENSION X(N),Y(M),L(N,M),P(N,M),Q(M,N),S(M,N)	5440
C	5450
C	5460
C GET PARTIALS WITH RESPECT TO X ALONG EDGES	5470
C	5480
C	5490
C DETERMINE THE LOCATION OF THE 4 X 4 GRID FOR THE LAGRANGE POLYNOMIAL	5500
C	5510
DO 10 J=1,M	5520
MS=J-2	5530
IF(MS .GT. (M-3)) MS=M-3	5540
IF(MS .LT. 1) MS=1	5550
MF=MS+3	5560
CALL LAGRAN(2,1,4,MS,MF,P(1,J),1,J,N,M,X,Y,U,P,Q,S)	5570
CALL LAGRAN(2,N-3,N,MS,MF,P(N,J),N,J,N,M,X,Y,U,P,Q,S)	5580
10 CONTINUE	5590
C	5600
C GET PARTIALS WITH RESPECT TO Y ALONG EDGES	5610
C	5620
DO 20 I=1,N	5630
NS=I-2	5640
IF(NS .GT. (N-3)) NS=N-3	5650
IF(NS .LT. 1) NS=1	5660
NF=NS+3	5670
CALL LAGRAN(3,NS,NF,1,4,C(1,I),1,1,N,M,X,Y,U,P,Q,S)	5680
CALL LAGRAN(3,NS,NF,M-3,M,C(M,I),I,M,N,M,X,Y,U,P,Q,S)	5690
20 CONTINUE	5700
C	5710
C GET PARTIALS WITH RESPECT TO X AND Y AT CORNERS	5720
C	5730
CALL LAGRAN(4,1,4,1,4,S(1,1),1,1,N,M,X,Y,U,P,Q,S)	5740
CALL LAGRAN(4,N-3,N,1,4,S(1,N),N,1,N,M,X,Y,U,P,Q,S)	5750
CALL LAGRAN(4,1,4,M-3,M,S(M,1),1,M,N,M,X,Y,U,P,Q,S)	5760
CALL LAGRAN(4,N-3,N,M-3,M,S(M,N),N,M,N,M,X,Y,U,P,Q,S)	5770
RETURN	5780
END	5790

SUBROUTINE GETOP(N,V,DP,BIP1,BIM1)	5800
C	5810
C SUBROUTINE TO GET THE 2 DIMENSIONAL DIFFERENCE ARRAY,B, OF EQ(15) OF	5820
C REFERENCE AND ALSO THE B PRIME ARRAY FOR GAUSSIAN ELIMINATION	5830
C	5840
C THIS SUBROUTINE IS CALLED BY SUBROUTINE DICUB1.	5850
C	5860
C	5870
C TO SAVE STORAGE WE	5880
C STORE ELEMENT B(1,1) IN BP(1)	5890
C STORE ELEMENT B(1,1+1) IN BIP1(1)	5900
C STORE ELEMENT B(1,1+1) IN BIM1(1)	5910
C THIS REDUCES THE SPACE REQUIRED FOR THE B AND B PRIME ARRAYS	5920
C FROM 2*N+2 TO 3*N WORDS	5930
C	5940
DIMENSION BP(N),BIP1(N),BIM1(N),V(N)	5950
C	5960
NM1=N-1	5970
NM2=N-2	5980
NM3=N-3	5990
DXR=V(2)-V(1)	6000
BP(1)=2*(DXR*(V(3)-V(2)))	6010
BIP1(1)=DXR	6020
DXL=V(N)-V(NM1)	6030
BIM1(NM2)=DXL	6040
BP(NM2)=2*(DXL*(V(NM1)-V(NM2)))	6050
IF(N,EO,4) GO TO 11	6060
C	6070
DO 10 I=2,NM3	6080
IM1=I-1	6090
IP1=I-1	6100
IP2=IP1+1	6110
DXL=V(IP2)-V(IP1)	6120
DXR=V(IP1)-V(I)	6130
BIM1(I)=DXL	6140
BP(I)=2*(DXL+DXR)	6150
BIP1(I)=DXR	6160
10 CONTINUE	6170
C	6180
C	6190
C NOW DETERMINE THE B PRIME MATRIX	6200
C	6210
11 DO 20 I=2,NM2	6220
IM1=I-1	6230
BP(I)=BP(I)-BIM1(I)*BIP1(IM1)/BP(IM1)	6240
20 CONTINUE	6250
RETURN	6260
END	6270

SUBROUTINE LAGRAN(NTYPE,AS,NP,MS,MP,DPT,NPT,MPT,N,M,X,Y,U,P,Q,S)	6280
C	6290
C	6300
C SUBROUTINE FOR INTERPOLATING THE VALUE OF A FUNCTION AND ITS	6310
C DERIVATIVES WITH RESPECT TO X, WITH RESPECT TO Y, AND WITH RESPECT TO	6320
C X AND Y AT THE MESH POINTS USING A LAGRANGE INTERPOLATING	6330
C POLYNOMIAL OF ARBITRARY ORDER	6340
C	6350
C THIS SUBROUTINE IS CALLED BY SUBROUTINE EDGES,	6360
C	6370
C	6380
C NS= NUMBER OF STARTING POINT ALONG THE X AXIS	6390
C NP= FINAL POINT ALONG THE X AXIS	6400
C MS= STARTING POINT ALONG THE Y AXIS	6410
C MP= FINAL POINT ALONG THE Y AXIS	6420
C INTERPOLATION IS CARRIED OUT OVER THESE POINTS	6430
C	6440
C	6450
C SUBROUTINE LAGRAN HAS 16 PARAMETERS IN ITS CALLING SEQUENCE	6460
C	6470
C	6480
C NTYPE=1 GET FUNCTION ITSELF	6490
C NTYPE=2 GET PARTIAL WITH RESPECT TO X	6500
C NTYPE=3 GET PARTIAL WITH RESPECT TO Y	6510
C NTYPE=4 GET PARTIAL WITH RESPECT TO X AND Y	6520
C	6530
C	6540
DIMENSION X(N),Y(M),L(N,M),P(N,M),Q(M,N),S(M,N)	6550
C	6560
XP=X(NPT)	6570
YP=Y(MPT)	6580
DPT=0,	6590
GO TO(10,20,30,40),NTYPE	6600
C	6610
C CALCULATE THE VALUE OF THE FUNCTION AT THE MESH POINT (NPT,MPT)	6620
C	6630
10 DPT=U(NPT,MPT)	6640
RETURN	6650
C	6660
C	6670
C CALCULATE THE FIRST PARTIAL WITH RESPECT TO X	6680
C AT THE MESH POINT (NPT,MPT)	6690
C	6700
20 DO 211 I=NS,NP	6710
SUM=0,	6720
DO 219 L=MS,MP	6730
IF(L.EQ. I) GO TO 215	6740
PROD=1.	6750
DO 212 K=NS,NP	6760
IF(K.EQ. I ,OR, K.EQ. L) GO TO 212	6770
PROD=PROD*(XP-X(K))	6780
212 CONTINUE	6790
SUM=SUM+PROD	6800
219 CONTINUE	6810
DENOM=1.	6820
XI=X(I)	6830

DO 213 K=NS,NF	6840
IF(K,EO,1) GO TO 213	6850
DENOM=DENOM*(X1-X(K))	6860
213 CONTINUE	6870
DPT=DPT,SUM/DENOM=L(1,MPT)	6880
211 CONTINUE	6890
RETURN	6900
C	6910
C CALCULATE THE FIRST PARTIAL WITH RESPECT TO Y	6920
C AT THE MESH POINT (NPT,MPT)	6930
C	6940
30 DO 311 J=MS,MF	6950
SUM=0	6960
DO 312 L=MS,MF	6970
IF(L,EO,J) GO TO 315	6980
PROD=1	6990
DO 312 K=MS,MF	7000
IF(K,EO,J,OR,K,EO,L) GO TO 312	7010
PROD=PROD*(YJ-Y(K))	7020
312 CONTINUE	7030
SUM=SUM,PROD	7040
315 CONTINUE	7050
DENOM=1	7060
YJ=Y(J)	7070
DO 313 K=MS,MF	7080
IF(K,EO,J) GO TO 313	7090
DENOM=DENOM*(YJ-Y(K))	7100
313 CONTINUE	7110
DPT=DPT,SUM/DENOM=L(NPT,J)	7120
311 CONTINUE	7130
RETURN	7140
C	7150
C CALCULATE THE PARTIAL WITH RESPECT TO X AND Y	7160
C AT THE MESH POINT (NPT,MPT)	7170
C	7180
40 DO 41 I=NS,NF	7190
SUMX=0	7200
DO 43 L=NS,NF	7210
IF(L,EO,1) GO TO 43	7220
PROD=1	7230
DO 44 K=NS,NF	7240
IF(K,EO,1,OR,K,EO,L) GO TO 44	7250
PROD=PROD*(XP-X(K))	7260
44 CONTINUE	7270
SUMX=SUMX,PROD	7280
43 CONTINUE	7290
DENOM=1	7300
X1=X(1)	7310
DO 45 K=NS,NF	7320
IF(K,EO,1) GO TO 45	7330
DENOM=DENOM*(X1-X(K))	7340
45 CONTINUE	7350
SUMX=SUMX/DENOM	7360
DO 42 J=MS,MF	7370
SUM=0	7380
DO 433 L=MS,MF	7390

IF(L,EO, J) GO TO 433	7400
PROD=1.	7410
DO 434 K=MS,MF	7420
IF(K,EO, J,OR, K,EC, L) GO TO 434	7430
PROD=PROD*(YF-Y(K))	7440
434 CONTINUE	7450
SUMY=SUMY+PROD	7460
433 CONTINUE	7470
DENOM=1.	7480
YJ=Y(J)	7490
DO 435 K=MS,MF	7500
IF(K,EO, J) GO TO 435	7510
DENOM=DENOM*(YJ-Y(K))	7520
435 CONTINUE	7530
SUMY=SUMY/DENOM	7540
DPT=DPT+SUMY*SUMX*L(I,J)	7550
42 CONTINUE	7560
41 CONTINUE	7570
RETURN	7580
C	7590
END	7600

SUBROUTINE SOLVIT(N,Z,C,BP,BIP1,BIM1,DP)	7610
C	7620
C	7630
C THIS ROUTINE IS CALLED BY SUBROUTINE DICUB1	7640
C	7650
DIMENSION BP(N),BIP1(N),BIM1(N),D(N),Z(N),DP(N)	7660
C	7670
C COMPUTE THE D PRIME VECTOR, SEE EQ(16) OF REF	7680
C	7690
DP(1)=D(1)	7700
DO 10 I=2,N	7710
IM1=I-1	7720
DP(I)=D(I)-BIM1(I)*DP(IM1)/BP(IM1)	7730
10 CONTINUE	7740
C	7750
C OBTAIN SOLUTION BY RECURSION RELATION OF EQ(17) , SEE REF	7760
C	7770
Z(N)=DP(N)/BP(N)	7780
NM1=N-1	7790
DO 20 J=1,NM1	7800
IN=J	7810
IP1=I+1	7820
Z(I)=(DP(I)-BIP1(I)*Z(IP1))/BP(I)	7830
20 CONTINUE	7840
RETURN	7850
END	7860

6.0 COMPARISONS

For most quantities defined over a two dimensional mesh, where N and M are greater than 4, the bicubic spline generates a more physically plausible interpolation surface than a two dimensional Lagrange interpolation polynomial over the same mesh. When N and M are equal to 4 the bicubic spline and Lagrange interpolating polynomial are identical.

No comparisons have been made with any other programs.

7.0 TEST METHODS AND RESULTS

The following program, CHECK, illustrates the use of the routines.

CHECK begins by setting up an arbitrary 5 x 6 mesh using a data statement to define the X and Y arrays. Next a third order two dimensional polynomial, $U(I,J)$, having arbitrary coefficients is evaluated at each of the mesh points. Since NDFault is initially set to zero, CHECK is required to supply the normal derivatives along the boundaries. The equations used in statements 300, 301, and 302 were obtained by differentiating the polynomial $U(I,J)$. Next subroutine BICUB1 is called to complete the P, Q, and S arrays and the results are printed. Following this, 30 random coordinates are generated over the x-y mesh using a uniform random number generator available in the CDC-3800 system library and the polynomial $U(I,J)$ is evaluated at each point (UEXACT). Also subroutine BICUB2 is called to interpolate a value at each point. Then the x-y coordinates of the 30 points are printed out. Since the arbitrary polynomial is third order in x and y, the interpolated and exact values should be the same. The fact that the elements of the difference column are essentially zero (neglecting rounding errors) indicates that this is indeed the case. Next the P, Q, and S arrays are zeroed and the parameter NDFault is set to 1, indicating that BICUB1 should determine the edge conditions rather than assume they are supplied by CHECK, and the above procedure is repeated. Again the difference column is essentially zero. In addition, the P, Q, and S arrays determined by BICUB1 agree almost exactly with those obtained by CHECK.

PROGRAM CHECK	7870
C	7880
C SAMPLE PROGRAM TO ILLUSTRATE THE USE OF THE	7890
C CUBIC SPLINE INTERPOLATION PACKAGE	7900
C	7910
C SETTING UP AN ARBITRARY MESH	7920
C	7930
DATA(N=5),(M=6),(MAX=6)	7940
DATA(X=1.,2.,5.,2.,75.,3.,5.), (Y=1.,1.,5.,2.,25.,4.,5.,7.,3)	7950
C	7960
DIMENSION X(5),Y(6),L(5,6),P(5,6),Q(6,5),S(6,5)	7970
C	7980
C SET THE DIMENSION OF EACH OF THE SEVEN WORK AREAS TO MAX,	7990
C	8000
DIMENSION W1(6),W2(6),W3(6),W4(6),W5(6),W6(6),W7(6)	8010
C	8020
C EVALUATE AN ARBITRARY POLYNOMIAL AT EACH MESH POINT	8030
C	8040
DO 12 I=1,N	8050
XI=X(I)	8060
DO 20 J=1,M	8070
YI=Y(J)	8080
U(I,J)=3.*XI+15.*XI+17.*XI**2+83.*XI**3	8090
X*YI*(457+26.*XI+18.*XI**2+19.*XI**3)	8100
X *YI**2*(34.+6.*XI+13.*XI**2+43.*XI**3)	8110
X *YI**3*(47.+21.*XI+15.*XI**2+2.*XI**3)	8120
20 CONTINUE	8130
12 CONTINUE	8140
C	8150
C	8160
C IF NDFAU1T IS SET TO 0 THE EDGE CONDITIONS WILL BE INPUT TO BICUB1.	8170
C IF NDFAU1T IS SET TO 1 BICUB1 WILL CALCULATE THE EDGE CONDITIONS,	8180
C	8190
NDFAU1T=0	8200
C	8210
C CALCULATE EXACT EDGE CONDITIONS FOR TEST EQUATION	8220
C	8230
NM1=N-1	8240
MM1=M-1	8250
DO 300 I=1,N,NM1	8260
XI=X(I)	8270
C	8280
C GET NORMAL DERIVATIVES WITH RESPECT TO X	8290
C	8300
DO 300 J=1,M	8310
YI=Y(J)	8320
300 P(I,J)=15.+34.*XI+249.*XI**2	8330
X *YI*(26.+36.*XI+57.*XI**2)	8340
X *YI**2*(6.+26.*XI+129.*XI**2)	8350
X *YI**3*(21.+30.*XI+6.*XI**2)	8360
C	8370
C	8380
C GET NORMAL DERIVATIVES WITH RESPECT TO Y ALONG EDGE	8390
C	8400
DO 301 J=1,M,MM1	8410
YI=Y(J)	8420

DO 301 I=1,N	8430
X=X(I)	8440
301 Q(J,I)=45.*26.*X I *18.*X I **2*19.*X I **3	8490
X *2.*Y I *(34.*6.*X I *13.*X I **2*43.*X I **3)	8460
X *3.*Y I **2*(47.*21.*X I *15.*X I **2*2.*X I **3)	8470
C	8480
C GET NORMAL DERIVATIVES WITH RESPECT TO BOTH X AND Y AT EACH CORNER OF	8490
C MESH,	8500
C	8510
DO 302 I=1,N,NM1	8520
X=X(I)	8530
DO 302 J=1,M,MM1	8540
Y=Y(J)	8550
302 S(J,I)=26.*36.*X I *57.*X I **2	8560
X *2.*Y I *(6.*26.*X I *129.*X I **2)	8570
X *3.*Y I **2*(21.*30.*X I *6.*X I **2)	8580
C	8590
C	8600
C ESTIMATE THE AMOUNT OF TIME REQUIRED FOR A CALL TO BICUB1	8610
C	8620
200 T=TIMELEFT(1)	8630
C	8640
C COMPLETE THE P,Q, AND S ARRAYS , THAT IS, DETERMINE NORMAL DERIVATIVES	8650
C AT EACH MESH POINT,	8660
C	8670
C *****	8680
CALL BICUB1(N,M,NDFALLT,X,Y,U,P,Q,S,MAX,W1,W2,W3,W4,W5,W6,W7)	8690
C *****	8700
C	8710
C	8720
C CALCULATE TIME SPENT IN MILLISECONDS	8730
C	8740
T=(T-TIMELEFT(1))*1000,	8750
PRINT 3	8760
DO 100 I=1,N	8770
DO 110 J=1,M	8780
PRINT 1,1,I,J,X(I),Y(J),U(I,J),P(I,J),Q(I,J),S(J,I)	8790
110 CONTINUE	8800
100 CONTINUE	8810
PRINT 4,T	8820
PRINT 2	8830
C	8840
C USE SYSTEM RANDOM NUMBER GENERATOR TO	8850
C GENERATE RANDOM COORDINATES OVER THE X-Y PLANE	8860
C	8870
DO 10 K=1,30	8880
X(=RANF(=1))*(X(N)-X(1))+X(1)+.3*RANF(=1)	8890
Y(=RANF(=1))*(Y(M)-Y(1))+Y(1)+.3*RANF(=1)	8900
UBXACT=3.*15.*X I *17.*X I **2*83.*X I **3	8910
X *Y I *(457.*26.*X I *18.*X I **2*19.*X I **3)	8920
X *2.*Y I **2*(34.*6.*X I *13.*X I **2*43.*X I **3)	8930
X *3.*Y I **3*(47.*21.*X I *15.*X I **2*2.*X I **3)	8940
C	8950
C	8960
C ESTIMATE THE AMOUNT OF TIME REQUIRED FOR THE CALL TO BICUB2	8970
C	8980

FTN5,5A

```

      T=TIMELEFT(1)
C INTERPOLATE AT VALUE AT (XI,YI)
C .....
      CALL BICUB2(XI,YI,LINT,NERROR,N,M,X,Y,U,P,Q,S)
C .....
C
      T=(T-TIMELEFT(1))*1000,
      DIFF=UINT-UEXACT
      PRINT 1,K,XI,YI,UEXACT,UINT,DIFF,NERROR,T
10 CONTINUE
C
C SEE IF WE HAVE COMPLETED THE SECOND PASS
C
      IF(NDFAULT.EQ.1) STOP
C
C REDEFINE NDFAULT SO THAT THE EDGE CONDITIONS WILL BE CALCULATED
C AND ZERO OUT THE P,Q, AND S ARRAYS,
C
      NDFAULT=1
      DO 501 J=1,M
      DO 500 I=1,N
      P(I,J)=Q(J,I)=S(J,I)=0,
500 CONTINUE
501 CONTINUE
C REPEAT ASSUMING BOUNDARY CONDITIONS ARE NOT GIVEN,
C HERE WE USE THE LAGRANGE INTERPOLATING POLYNOMIAL TO DETERMINE THE
C EDGE CONDITIONS,
C
      GO TO 200
C
C FORMAT STATEMENTS .....
C
1 FORMAT(10X,15,2X,4(F12,4,2X),E16,5,2X,15,2X,F12,4)
2 FORMAT(1X,/,
X 14X,0K,9X,0X,0,12X,0Y,0,9X,0UEXACT,0X,0UINT,0,12X,
X 0DIFFERENCE,0X,0NERROR,0X,0TIME,/)
3 FORMAT(1X,/,3X,0I,0J,0,9X,0X,0,12X,0Y,0,12X,0U,0,12X,0P,0,12X,0Q,0,
X 12X,0S,/)
4 FORMAT(10X,0THE CALL TO BICUB1 TOOK APPROX,0,F12,4,0 MSEC,0)
111 FORMAT(1X,2(13,1X),6(E12,4,1X))
C
      END

```

I	J	X	Y	U	P	Q	S
1	1	1.0000+000	1.0000+000	4.0700+002	6.3500+002	5.5500+002	6.1200+002
1	2	1.0000+000	1.5000+000	7.8287+002	1.0311+003	9.6975+002	9.6675+002
1	3	1.0000+000	2.2500+000	1.8152+003	2.0301+003	1.8309+003	1.7092+003
1	4	1.0000+000	4.5000+000	1.0294+004	9.2879+003	6.1358+003	5.0308+003
1	5	1.0000+000	5.0000+000	1.3683+004	1.2043+004	7.4430+003	6.0040+003
1	6	1.0000+000	7.5000+000	3.9089+004	3.1920+004	1.5099+004	1.1582+004
2	1	2.5000+000	1.0000+000	2.9896+003	3.1393+003	2.7971+003	2.6273+003
2	2	2.5000+000	1.5000+000	4.7852+003	4.7890+003	4.4411+003	4.0051+003
2	3	2.5000+000	2.2500+000	9.2302+003	8.6805+003	7.5385+003	6.4474+003
2	4	2.5000+000	4.5000+000	4.0481+004	3.3711+004	2.1377+004	1.6478+004
2	5	2.5000+000	5.0000+000	5.2156+004	4.2636+004	2.5378+004	1.9257+004
2	6	2.5000+000	7.5000+000	1.3531+005	1.0379+005	4.8121+004	3.4623+004
3	1	2.7500+000	1.0000+000	3.8496+003	3.7496+003	3.5133+003	3.1088+003
3	2	2.7500+000	1.5000+000	6.0943+003	5.6375+003	5.5305+003	4.7202+003
3	3	2.7500+000	2.2500+000	1.1596+004	1.0270+004	9.2870+003	7.5559+003
3	4	2.7500+000	4.5000+000	4.9609+004	3.9385+004	2.5817+004	1.9078+004
3	5	2.7500+000	5.0000+000	6.3687+004	4.9708+004	3.0562+004	2.2252+004
3	6	2.7500+000	7.5000+000	1.6327+005	1.2008+005	5.7408+004	3.9731+004
4	1	3.0000+000	1.0000+000	4.8690+003	4.4150+003	4.3550+003	3.6320+003
4	2	3.0000+000	1.5000+000	7.6406+003	6.6866+003	6.8063+003	5.4958+003
4	3	3.0000+000	2.2500+000	1.4376+004	1.1996+004	1.1324+004	8.7554+003
4	4	3.0000+000	4.5000+000	6.0212+004	4.5516+004	3.0932+004	2.1876+004
4	5	3.0000+000	5.0000+000	7.7057+004	5.7343+004	3.6523+004	2.5472+004
4	6	3.0000+000	7.5000+000	1.9546+005	1.3761+005	6.8017+004	4.5203+004
5	1	5.0000+000	1.0000+000	2.0419+004	1.1723+004	1.6859+004	9.3160+003
5	2	5.0000+000	1.5000+000	3.0969+004	1.7502+004	2.5537+004	1.3881+004
5	3	5.0000+000	2.2500+000	5.5659+004	3.0751+004	4.0739+004	2.1631+004
5	4	5.0000+000	4.5000+000	2.1190+005	1.1106+005	1.0208+005	5.1381+004
5	5	5.0000+000	5.0000+000	2.6710+005	1.3871+005	1.1891+005	5.9316+004
5	6	5.0000+000	7.5000+000	6.4221+005	3.2230+005	2.1137+005	1.0202+005

THE CALL TO BICUB1 TOOK APPROX. 23.0000 MSEC.

K	XI	YI	UEXACT	UINY	DIFFERENCE	NERRR	TIME
1	4.6801	7.3000	544727.7631	544727.7631	0.00000000	0	3.0000
2	3.9544	7.3000	362712.4656	362712.4656	7.62939006	0	3.0000
ERROR-XPT OUT OF BOUNDS DETECTED BY SUB. BICUB2							
3	0.9417	7.3000	37263.9349	-0.0000	-3.728390004	1	15.0000
ERROR-XPT OUT OF BOUNDS DETECTED BY SUB. BICUB2							
4	5.1564	7.3000	693996.9885	-0.0000	-6.939970005	1	14.0000
5	2.4001	7.3000	125257.4696	125257.4696	-1.907350006	0	3.0000
6	4.7003	7.3000	550557.2826	550557.2826	1.525880005	0	3.0000
7	3.8947	7.3000	349989.5102	349989.5102	0.000000000	0	3.0000
8	3.2720	7.3000	235644.8094	235644.8094	0.000000000	0	3.0000
9	2.3545	7.3000	120867.6676	120867.6676	-3.814700006	0	4.0000
10	3.4317	7.3000	261904.0295	261904.0295	3.814700006	0	3.0000
11	4.6961	7.3000	549341.0577	549341.0577	-3.051760005	0	3.0000
12	1.5016	7.3000	59680.5639	59680.5639	0.000000000	0	3.0000
ERROR-XPT OUT OF BOUNDS DETECTED BY SUB. BICUB2							
13	0.9623	7.3000	37908.1032	-0.0000	-3.790810004	1	14.0000
14	2.7008	7.3000	157443.8451	157443.8451	-3.814700006	0	3.0000
15	2.9853	7.3000	193437.7735	193437.7735	0.000000000	0	3.0000
16	4.9281	7.3000	619325.4995	619325.4995	-1.525880005	0	3.0000
ERROR-XPT OUT OF BOUNDS DETECTED BY SUB. BICUB2							
17	5.4112	7.3000	784423.6679	-0.0000	-7.844240005	1	14.0000
18	3.4676	7.3000	268078.4195	268078.4195	0.000000000	0	4.0000
19	2.5798	7.3000	143800.8440	143800.8440	-3.814700006	0	3.0000
20	2.0562	7.3000	95141.0646	95141.0646	3.814700006	0	3.0000
21	4.0475	7.3000	383210.2274	383210.2274	0.000000000	0	3.0000
22	4.3128	7.3000	446125.0424	446125.0424	7.629390006	0	4.0000
23	1.6377	7.3000	67029.4197	67029.4197	1.907350006	0	3.0000
24	3.2053	7.3000	225272.4446	225272.4446	-3.814700006	0	3.0000
ERROR-XPT OUT OF BOUNDS DETECTED BY SUB. BICUB2							
25	5.5974	7.3000	855451.8719	-0.0000	-8.554520005	1	13.0000
ERROR-XPT OUT OF BOUNDS DETECTED BY SUB. BICUB2							
26	5.3811	7.3000	773367.9365	-0.0000	-7.733680005	1	14.0000
27	1.1284	7.3000	43467.2707	43467.2707	0.000000000	0	4.0000
28	2.7850	7.3000	167520.5473	167520.5473	-3.814700006	0	4.0000
29	4.3106	7.3000	445564.0880	445564.0880	7.629390006	0	3.0000
30	2.6334	7.3000	149722.1129	149722.1129	-3.814700006	0	3.0000

I	J	X	Y	U	P	Q	S
1	1	1.0000+000	1.0000+000	4.0700+002	6.3500+002	5.5500+002	6.1200+002
1	2	1.0000+000	1.5000+000	7.8287+002	1.0311+003	9.6975+002	9.8673+002
1	3	1.0000+000	2.2500+000	1.8152+003	2.0301+003	1.8309+003	1.7092+003
1	4	1.0000+000	4.5000+000	1.0294+004	9.2879+003	6.1358+003	5.0308+003
1	5	1.0000+000	5.0000+000	1.3683+004	1.2043+004	7.4430+003	6.0040+003
1	6	1.0000+000	7.5000+000	3.9089+004	3.1920+004	1.5099+004	1.1582+004
2	1	2.5000+000	1.0000+000	2.9896+003	3.1393+003	2.7971+003	2.6272+003
2	2	2.5000+000	1.5000+000	4.7852+003	4.7890+003	4.4411+003	4.0051+003
2	3	2.5000+000	2.2500+000	9.2302+003	6.6803+003	7.5385+003	6.4474+003
2	4	2.5000+000	4.5000+000	4.0481+004	3.3711+004	2.1377+004	1.6478+004
2	5	2.5000+000	5.0000+000	5.2156+004	4.2636+004	2.5378+004	1.9257+004
2	6	2.5000+000	7.5000+000	1.3531+005	1.0379+005	4.8121+004	3.4623+004
3	1	2.7500+000	1.0000+000	3.8496+003	3.7496+003	3.5133+003	3.1088+003
3	2	2.7500+000	1.5000+000	6.0943+003	5.6975+003	5.5305+003	4.7202+003
3	3	2.7500+000	2.2500+000	1.1596+004	1.0270+004	9.2870+003	7.5559+003
3	4	2.7500+000	4.5000+000	4.9609+004	3.9385+004	2.5817+004	1.9078+004
3	5	2.7500+000	5.0000+000	6.3687+004	4.9708+004	3.0562+004	2.2522+004
3	6	2.7500+000	7.5000+000	1.6327+005	1.2008+005	5.7408+004	3.9731+004
4	1	3.0000+000	1.0000+000	4.8690+003	4.4150+003	4.3550+003	3.6320+003
4	2	3.0000+000	1.5000+000	7.6406+003	6.6866+003	6.8063+003	5.4958+003
4	3	3.0000+000	2.2500+000	1.4376+004	1.1996+004	1.1324+004	8.7554+003
4	4	3.0000+000	4.5000+000	6.0212+004	4.5516+004	3.0932+004	2.1876+004
4	5	3.0000+000	5.0000+000	7.7057+004	5.7343+004	3.6523+004	2.5472+004
4	6	3.0000+000	7.5000+000	1.9546+005	1.3761+005	6.8017+004	4.5203+004
5	1	5.0000+000	1.0000+000	2.0419+004	1.1723+004	1.6859+004	9.3160+003
5	2	5.0000+000	1.5000+000	3.0969+004	1.7502+004	2.5537+004	1.3881+004
5	3	5.0000+000	2.2500+000	5.5659+004	3.0751+004	4.0739+004	2.1631+004
5	4	5.0000+000	4.5000+000	2.1190+005	1.1106+005	1.0208+005	5.1381+004
5	5	5.0000+000	5.0000+000	2.6710+005	1.3871+005	1.1891+005	5.9316+004
5	6	5.0000+000	7.5000+000	6.4221+005	3.2230+005	2.1137+005	1.0202+005

THE CALL TO BICUB1 T00K APPR0X. 135.0000 MSEC.

K	XI	YI	UEXACT	UINY	DIFFERENCE	NERRAR	TIME
1	3.6803	7.3000	346979.9898	346979.9899	9.15527-005	0	3.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
2	0.7365	7.3000	31767.4360	-0.0000	-3.17674-004	1	14.0000
3	3.7808	7.3000	326597.6141	326597.6142	6.86644-005	0	3.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
4	5.2111	7.3000	712790.2950	-0.0000	-7.12790-005	1	14.0000
5	2.7605	7.3000	164541.3793	164541.3793	-3.81470-006	0	2.0000
6	1.2772	7.3000	49278.4133	49278.4134	2.57492-005	0	4.0000
7	4.1136	7.3000	398260.6176	398260.6177	9.15527-005	0	3.0000
8	2.1946	7.3000	106446.5736	106446.5734	1.14441-005	0	4.0000
9	4.0813	7.3000	390873.5700	390873.5701	9.91821-005	0	3.0000
10	1.1850	7.3000	45579.5901	45579.5901	1.81198-005	0	2.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
11	5.2737	7.3000	734675.3059	-0.0000	-7.34675-005	1	14.0000
12	4.7600	7.3000	568042.4312	568042.4313	7.62939-005	0	4.0000
13	1.1182	7.3000	43096.8970	43096.8970	1.33514-005	0	2.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
14	5.2122	7.3000	713147.0529	-0.0000	-7.13147-005	1	14.0000
15	3.8593	7.3000	342596.3454	342596.3455	9.91821-005	0	3.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
16	0.7218	7.3000	31417.4948	-0.0000	-3.14175-004	1	14.0000
17	2.6614	7.3000	152901.5849	152901.5849	0.00000-000	0	4.0000
18	2.8192	7.3000	171750.5687	171750.5687	0.00000-000	0	2.0000
19	1.2153	7.3000	46760.5225	46760.5225	1.90735-005	0	2.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
20	5.6675	7.3000	883290.6428	-0.0000	-8.83291-005	1	14.0000
21	2.8152	7.3000	171242.4343	171242.4343	-3.81470-006	0	2.0000
22	3.2704	7.3000	235392.1745	235392.1746	1.52588-005	0	4.0000
23	2.7102	7.3000	158550.9106	158550.9106	0.00000-000	0	3.0000
24	3.3746	7.3000	252282.3114	252282.3114	2.67029-005	0	4.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
25	0.7928	7.3000	33154.9680	-0.0000	-3.31550-004	1	14.0000
ERROR-XPT OUT OF BEUNDS DETECTED BY SUB. BICUB2							
26	5.2357	7.3000	721317.0978	-0.0000	-7.21317-005	1	15.0000
27	1.1668	7.3000	44889.5823	44889.5823	1.62125-005	0	2.0000
28	2.9207	7.3000	184767.6106	184767.6106	0.00000-000	0	3.0000
29	3.2510	7.3000	232340.2660	232340.2660	1.14441-005	0	4.0000
30	2.3937	7.3000	124634.3653	124634.3653	1.90735-006	0	2.0000

8.0 REMARKS

Although the Lagrange interpolation procedure described above for obtaining the "required" boundaries derivatives has been implemented and works, a different and possibly better approach [9] to this problem consists, in the analogous one dimensional case, of not having a break-point [6] at the second and second to last data points. Implementation of this latter approach is left as an exercise for the interested reader.

For those readers who may be interested, a completely independently conceived and different set of ALGØL procedures for bicubic spline interpolation is given in references [10 and 11] (in German). Comparison of Späth's algorithms with those described earlier is left as another exercise.

As a further remark, it should be obvious that the procedure described in this report could be readily generalized to N - dimensional cubic spline interpolation, where N is greater than 2.

9.0 ACKNOWLEDGMENTS

We are indebted to Professor Carl de Boor, of the University of Wisconsin, for reviewing an earlier draft of this report and offering a number of beneficial suggestions.

Also, we would like to thank Mrs. Janet Mason, of the NRL Research Computation Center, for checking the program and providing some helpful suggestions.