Title
Bicubic Hermite Interpolation Code

Authors
Dickinson, R P, Jr.
Carlson, R E
Fritsch, F N

Publication Date
1978

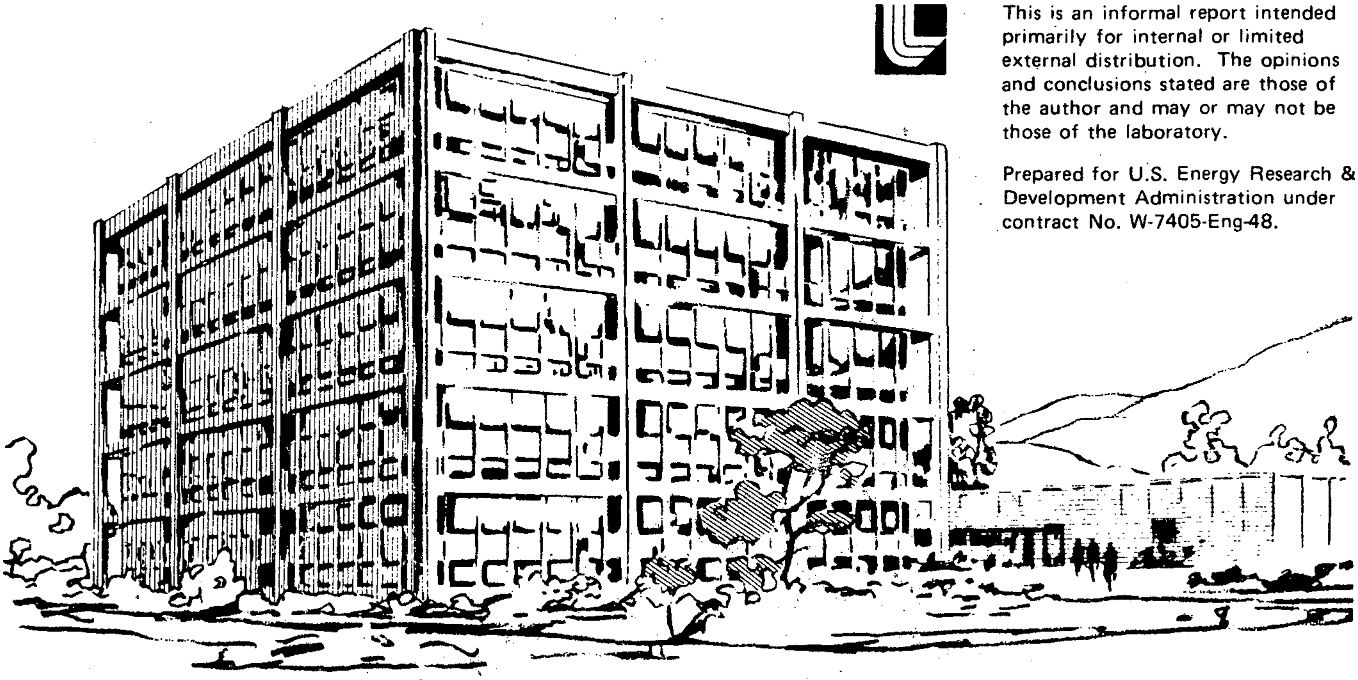
UCID-17623 REV.1 C.1

Lawrence Livermore Laboratory

BIHI: Bicubic Hermite Interpolation Code

R.P. Dickinson, Jr.
R.E. Carlson
F.N. Fritsch

January 1978



Prepared for U.S. Energy Research & Development Administration under contract No. W-7405-Eng-48.

DISCLAIMER

BIHI: Bicubic Hermite Interpolation Code

R.P. Dickinson, Jr.*
R.E. Carlson†
F.N. Fritsch

CONTENTS

NOTICE: This report supercedes UCID-17623, by the same title.

*Current address: Chabot College, Valley Campus, Livermore, CA 94550.
†Current address: Grove City College, Grove City, PA 16127

BIHI: Bicubic Hermite Interpolation Code

ABSTRACT

BIHI is a bivariate interpolation code which constructs a pice-wise bicubic function that interpolates to a given set of data values arranged over a rectangular grid.

1. INTRODUCTION

Over the past two years several bivariate interpolation codes have been developed by the Numerical Mathematics Group (NMG) for solving problems in surface approximation. These codes use smooth approximating functions such as splines, blending functions, and splines in tension which are quite accurate if the surfaces are "well behaved". For surfaces which are not so well behaved - such as those which rise nearly vertically and abruptly flatten out - the smooth approximations tend to overshoot in the neighborhood of the abrupt change. As a result, the approximation tends to oscillate in this subregion. This phenomenon is referred to as "ringing".

In certain equation of state (EOS) problems the ringing described above is unacceptable because the approximant is not monotonic. BIHI was developed in an effort to produce a monotonic approximation while preserving the accuracy of the higher order techniques used in the other codes. The technique used in BIHI does not guarantee monotonicity, but it does represent a significant improvement over the techniques which use smoother functions.

In BIHI a bicubic Hermite surface is constructed which interpolates to the given data. Partial derivatives at the mesh points (which are needed to construct the surface) are approximated using a three point difference formula. In Section 2 we discuss the relevant interpolation and approximation results. In Section 3 the code itself is described in detail. Numerical results for a sample problem are presented in Section 4.

The source code for BIHI is available in the form of an ORDER input deck in Photostore file

.295701:NMS:PROTOLIB:FNF:E1:BIHI.

The current NMG consultant for this code is F.N. Fritsch, ext. 2-4275.

2. INTERPOLATION AND APPROXIMATION RESULTS

BIHI was developed to solve the following bivariate interpolation problem. Let R be a rectangular domain partitioned as shown in Figure 1.

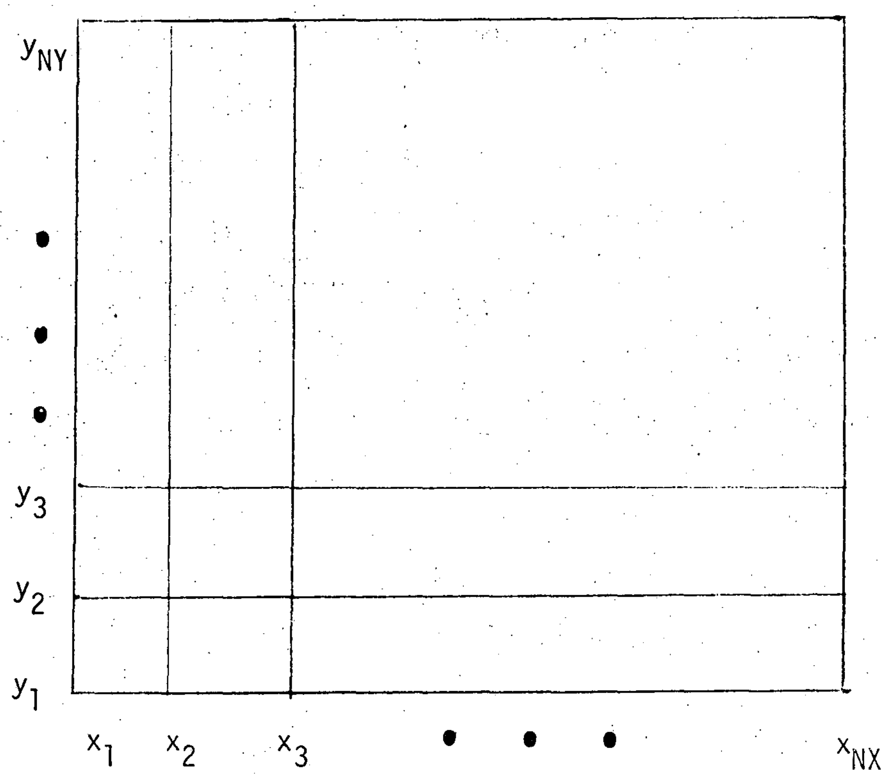


Figure 1. Partitioned Rectangular Domain.

It is assumed that data values are known at the mesh points. The data may be interpreted as points on a surface defined by a function $f(x,y)$ with domain R. The problem is to construct a function $u(x,y)$ such that $u(x,y)$ is a "good" approximation to $f(x,y)$, and $u(x,y)$, interpolates to the given data; i.e.,

$$(1)\quad u(x_i,y_j) = f(x_i,y_j) \qquad 1\le i\le NX,\ 1\le j\le NY.$$

In BIHI the function $u(x,y)$ is a piecewise bicubic Hermite polynomial. The usual bicubic Hermite interpolant of a function $f(x,y)$ not only interpolates function values as in (1), but also interpolates derivatives of $f(x,y)$. Specifically

$$\left.\begin{aligned}
&(2)\quad \frac{\partial u}{\partial x}(x_i,y_j) = \frac{\partial f}{\partial x}(x_i,y_j)\\
&(3)\quad \frac{\partial u}{\partial y}(x_i,y_j) = \frac{\partial f}{\partial y}(x_i,y_j)\\
&(4)\quad \frac{\partial^2 u}{\partial x\partial y}(x_i,y_j) = \frac{\partial^2 f}{\partial x\partial y}(x_i,y_j)
\end{aligned}\right\}\quad \begin{aligned}&1\le i\le NX\\ &1\le j\le NY\end{aligned}$$

It is well known that if $f(x,y) \varepsilon\, C^4\,[R]$, then the bicubic Hermite interpolant of $f(x,y)$ defined by (1)-(4) is a fourth order approximation to $f(x,y)$.*

However, in most bivariate interpolation problems, values of the partial derivatives of $f(x,y)$ required in (2)-(4) are not known.

*That is, there exists a constant K such that $||u-f||\le ch^4$, where h is the maximum mesh interval.

In BIHI these derivatives are approximated from the function values by a three point difference formula (see Appendix A). Once all derivatives in (2)-(4) have been approximated, a bicubic Hermite interpolant of f(x,y) can be constructed. In this case, because the derivations are only approximated, u(x,y) is only a third order approximation to $f(x,y) \in C^4 [R]$.

In order to evaluate u(x,y) at some point $(x',y') \in R$, the subrectangle containing (x',y') is located and in that subrectangle u(x,y) is written in the form

$$(5) \quad u(x,y) = \sum_{k=1}^{4} \sum_{\ell=1}^{4} a_{k\ell} H_k(x) G_\ell(y)$$

where the $H_k(x)$ and $G_\ell(y)$ are Hermite basic functions (see Ref. 1) and the $a_{k\ell}$ are the appropriate values derived from (1)-(4). This representation provides for the rapid evaluation of function values and derivatives of u(x,y). For completeness, we note that the first partial derivatives and cross-derivative of u(x,y) are given by:

$$(6) \quad \frac{\partial u}{\partial x}(x,y) = \sum_{k} \sum_{\ell} a_{k\ell} H_k'(x) G_\ell(y);$$

$$(7) \quad \frac{\partial u}{\partial y}(x,y) = \sum_{k} \sum_{\ell} a_{k\ell} H_k(x) G_\ell'(y);$$

$$(8) \quad \frac{\partial^2 u}{\partial x \partial y}(x,y) = \sum_{k} \sum_{\ell} a_{k\ell} H_k'(x) G_\ell'(y).$$

3. CODE DESCRIPTION

The BIHI main code is intended to be a prototype for user-tailored applications of bicubic Hermite interpolation. It is a driver for the two basic mathematical subroutines SETBIH, which computes the approximate derivatives needed to completely determine the bicubic Hermite interpolant u and SEVALH, which evaluates u at an arbitrary point. The latter two routines are designed to be used independently of the driver, if desired.

To be compatible with the array names actually used in the program, we shall use the following notation throughout this section:

$$\begin{aligned} F &= f ; \\ FX &\cong \partial f/\partial x ; \\ FY &\cong \partial f/\partial y ; \\ FXY &\cong \partial^2 f/\partial x \partial y. \end{aligned}$$

3.1 The Driver.

3.1.1. Code Structure.

Initially the driver calls GEN1 and GEN2. These routines read and check data from the user supplied disk file INBIH (described below) and print out the input values. After this initialization phase, SETBIH is called to calculate the arrays of approximate derivative values which completely describe the fit. If requested, the next routine called is OUTNOD, which produces a table of function values and various partial derivatives at the user's initial data mesh. Finally, SEVALH, the fit evaluator, is called to produce a table of function values and partial derivatives on a second user-specified uniform x, y-net. This evaluator allows extrapolation, and a flag is printed which indicates the type of extrapolation. Note: It would be an easy matter to modify this portion of the driver to put out the interpolated function values in a form suitable for input to a 3D plot package.

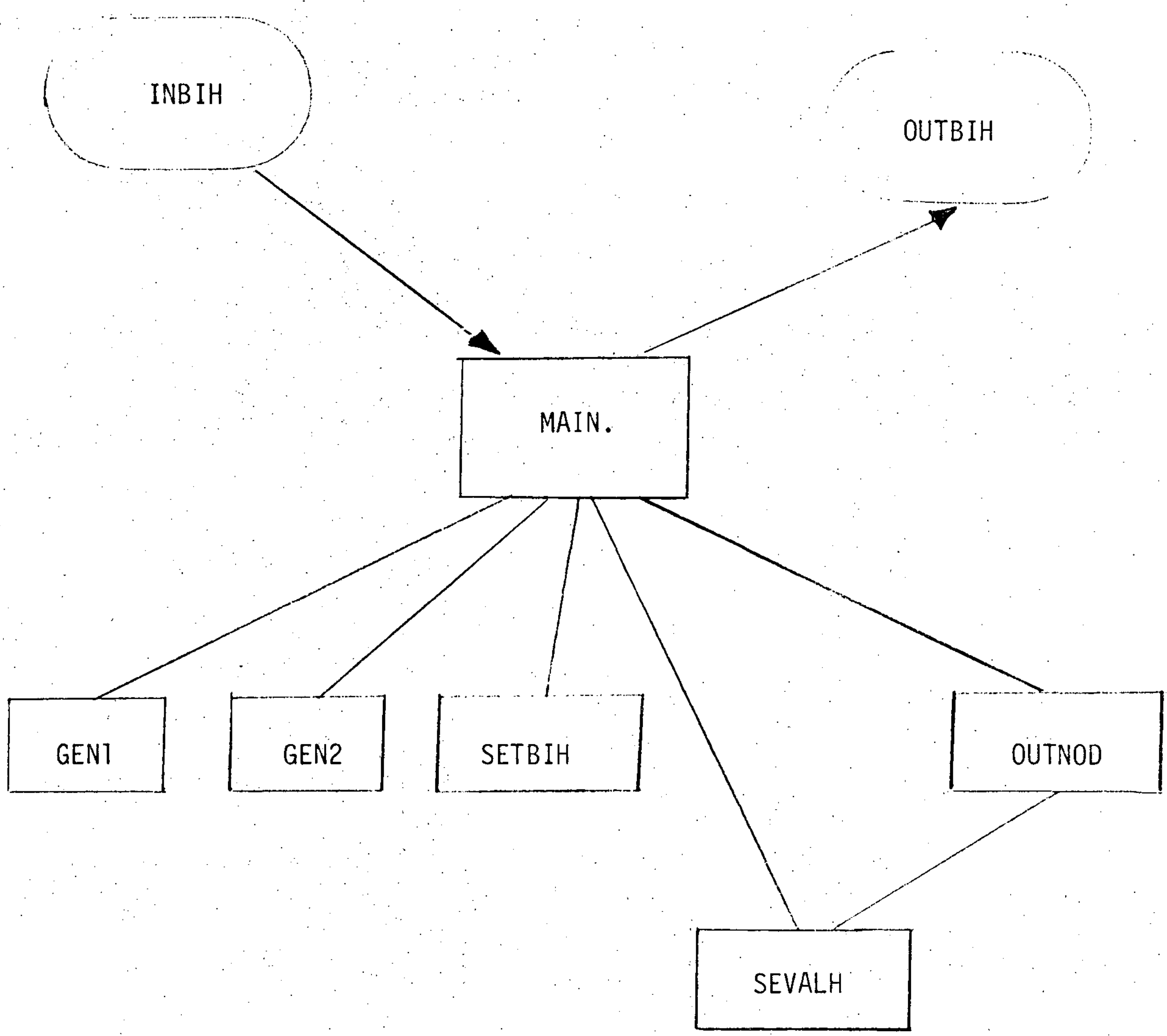


Figure 2. Structure of BIHI Driver.

3.1.2. Input. All data are read from the user supplied disk file INBIH. This section describes the format of this file in detail.

Line 1: NX, N, Y, ILOG1, ILOG2, (MFLAG(k), k = 1,4), IOPT as read with a 9I5 format.

As above, NX and NY are the number of x and y mesh lines, respectively. As presently dimensioned, these must satisfy $3 \leq NX, NY \leq 100$.

ILOG1 and ILOG2 are flags used for EOS applications. If ILOG1 is nonzero, logarithms of the independent variables are taken before calling SETBIH. Specifically, if ILOG1 = 0, then no logarithms are taken. If ILOG1 = 1, then logarithms of the x values are taken. If ILOG1 = 2, then logarithms of the y values are taken. If ILOG1 = 3, then logarithms are taken of both x and y values. ILOG2 is similar to ILOG1 but it applies to the dependent variable. If ILOG2 = 1, then logarithms are taken of the given function values, f, before fitting, and if ILOG2 = 0, then no logarithms are taken.

MFLAG is an array of four values which control boundary derivatives over R. MFLAG attempts to help preserve monotone behavior of a function when the data is monotone. The range of values for this flag is $-1 \leq \text{MFLAG}(k) \leq 1$.

MFLAG(k) = 0: No motonicity control.

MFLAG(k) = +1: Assume monotone increasing at boundary k and use a two point derivative approximation* if the three point formula produces a nonpositive result.

MFLAG(k) = -1: Assume monotone decreasing at boundary k; use two point formula if necessary.

The index k has the following meaning:

k = 1 for left boundary ($x = x_1$)

k = 2 for right boundary ($x = x_{NX}$)

k = 3 for bottom boundary ($y = y_1$)

k = 4 for top boundary ($y = y_{NY}$)

This flag is perhaps best understood by example. Suppose one knows that when x is held fixed and y is varied, starting at y_1, the function is monotone increasing. To approximate $\partial f/\partial y$ at the boundary $y = y_1$, the normal procedure is to fit a parabola through $f(x_i,y_1)$, $f(x_i,y_2)$, and $f(x_i,y_3)$. Then $FY(x_i,y_1)$, the approximate derivate, is set equal to the derivative of this parabola at y_1. In certain cases, even though the data is monotone increasing, the derivative from this three point formula may be negative*. This could be avoided by using the (theoretically less accurate) two point formula†

$$FY = (f(x_i,y_2) - f(x_i,y_1)) / (y_2 - y_1).$$

This is precisely what is done in SETBIH if MFLAG(3) = +1 and the three point formula gives a zero or negative value.

<u>Some suggested values for MFLAG</u>:

(a) If f is known to be monotone increasing in both variables, set MFLAG(k) = +1 for k = 1, ..., 4.

*Note, however, that the formula used to approximate interior derivatives automatically produces positive values for monotone increasing data.

†Which will necessarily be positive if the data is strictly increasing.

(b) If f is known to be monotone decreasing in both variables, set MFLAG(k) = -1 for k = 1, ..., 4.

(c) If f is something like a paraboloid, with a minimum interior to R, a reasonable setting might be MFLAG(1) = -1, MFLAG(2) = +1, MFLAG(3) = -1, MFLAG(4) = +1.

(d) If no special structure is present, then set all values of MFLAG to zero.

IOPT may be 0 or 1. If IOPT = 1, then OUTNOD is called to produce a table of F, FX, FY, and FXY at the points of the input mesh. (Incidently, OUTNOD prepares this table by calling the evaluator SEVALH.) If IOPT = 0, then OUTNOD is not called.

Line 2 through Line NX*NY+1:

These cards contain the X, Y, F-values in a 3E14.7 format. The data are read as follows:

((X(i), Y(j), F(i,j), j = 1, NY), i = 1, NX)

All of the above data are read in GEN1. If the user wished to modify the way in which the data is read, a simple modification could be made here. The formatting used was designed for a specific EOS surface fit.

The Last Line: NXP, NYP, XP1, YP1, DXP, DYP

as read with the format 2I3, 6X, 4E12.5.

This line is read in GEN2. It specifies a uniform rectangular net over which the interpolating function and its partial derivatives will be calculated. The x, y points are given by:

x = XVAL = XP1 + (i-1)*DXP   i = 1, ..., NXP

y = YVAL = YP1 + (j-1)*DYP   j = 1, ..., NYP

If ILOG1 is nonzero, the approximate x or y value of both will be logged before calling the evaluator.

Note: If logarithms have been used to transform the data, then the partial derivatives calculated will be derivatives of the logged quantities and not the derivatives of the original surface. Formulas can be found which approximate the derivatives of the original surface given the calculated derivatives, but these are not included in BIHI.

3.1.3. Output. The output from BIHI is written in the BIHI-created file OUTBIH. The output consists of three parts:

(1) The input data is written out. (This is done in GEN1 and GEN2.)

(2) If IOPT = 1, OUTNOD produces a table of F, FX, FY, and FXY over the interpolation mesh. This is done by calling the evaluator SEVALH.

(3) A table is produced (within the main code) of F, FX, FY, and FXY over a uniform rectangular domain, as specified by the user's input NXP, NYP, XP1, YP1, DXP, and DYP (read by GEN2). An extrapolation flag having four possible values is also printed. The table below explains this flag.

   (a) IEXT = 0: interpolation; that is $X(1) \leq XVAL \leq X(NX)$ and $Y(1) \leq YVAL \leq Y(NY)$.

   (b) IEXT = 1: extrapolation with respect to X only. That is, XVAL < X(1) or XVAL > X(NX).

   (c) IEXT = 2: extrapolation with respect to Y only.

   (d) IEXT = 3: extrapolation with respect to both X and Y.

3.1.4. Data Restrictions and Error Exits. Both the driver and the mathematical algorithm have conditions which must be satisfied by the data for successful execution. These conditions are summarized here, and an indication is given as to the type of error message produced if the conditions are not met.

(a) Conditions of GEN1:

(1) $0 \leq$ ILOP1 ≤ 3.

(2) $0 \leq$ ILOP2 ≤ 1.

(3) $3 \leq$ NX ≤ 100 .

(4) $3 \leq$ NY ≤ 100 .

(5) $0 \leq$ IOPT ≤ 1 .

(6) $-1 \leq$ MFLAG(k) ≤ 1, for k = 1, ..., 4.

(7) The points X(i), Y(j) must lie on a rectangular mesh. The data are being read a line at a time:

X(i), Y(j), F(i,j).

In order for the data to lie on a rectangular mesh, all values read for X(i), for a fixed i, must be equal, and all values for Y(j), for a fixed j, must be equal. For example, if NX = 3 and NY = 2, we would have as cards:

Card 2: X(1) Y(1) F(1,9)

Card 3: X(1) Y(2) F(1,2)

Card 4: X(2) Y(1) F(2,1)

Card 5: X(2) Y(2) F(2,2)

Card 6: X(3) Y(1) F(3,1)

Card 7: X(3) Y(2) F(3,2)

The number read in for X(1) on cards 2 and 3 must be the same. Similarly for X(2) on 4 and 5, etc. Also, the same number must be read in for Y(1) on cards 2, 4, and 6. Similarly for Y(2).

(8) If ILOG1 $\neq 0$ or ILOG2 $\neq 0$, then the appropriate quantity to be logged must be a positive number. For example, if ILOG1 = 1, then we must have that $X(i) > 0$ for $i = 1, \ldots; NX$.

If any of the above conditions is not met, the driver will exit with a message that one of the conditions was not satisfied in GEN1.

(b) Conditions of GEN2:

The only possible error here is if ILOG1 is nonzero and a generated XVAL or YVAL is negative or zero. Suppose ILOG1 = 1 (x's are logged); if some XVAL to be generated is negative or zero then exit will occur with an error message indicating this and saying that the exit occurred in GEN2. Recall that in this case we would log XVAL before calling the evaluator. Similar remarks apply to Y. Note that, since this exit occurs in GEN2, neither SETBIH nor SEVALH will have been called yet.

(c) Conditions of SETBIH:

If all the checks in GEN1 and GEN2 have been successfully passed by the user's data, the only additional requirement imposed by SETBIH is that the x and y arrays must both be strictly increasing. (See SETBIH description, below, for the reasons for this restriction.)

Mathematically, we must have

$$X(i-1) < X(i), \text{ for } i = 2, \ldots, NX,$$

and

$$Y(j-1) < Y(j), \text{ for } j = 2, \ldots, NY.$$

If any of these conditions fails, SETBIH will return a nonzero value of IER, which will cause the driver to exit after printing an appropriate error message. Printed values of IER have the following meaning:

IER = 1: X array not strictly increasing.

IER = 2: Y array not strictly increasing.

IER = 3: Neither array is strictly increasing.

NOTICE: The reader who does not wish to learn about the details of the mathematical subroutines SETBIH and SEVALH may now skip forward to the numerical example, Section 4.

3.2. SETBIH

SETBIH performs the first part of the mathematical algorithm, the evaluation of the approximations FX, FY, FXY to the derivatives $\partial f/\partial x$, $\partial f/\partial y$, $\partial^2 f/\partial x \partial y$ (respectively).

3.2.1. Code Structure. After checking for valid input, SETBIH calls SMCHEK to check that the X and Y arrays are both strictly monotone increasing. If all checks pass, SETBIH then calls the one-dimensional derivative approximation routine DAPROX three times to set up FX, FY, and FXY. DAPROX evaluates the interior derivatives internally, but calls FUNDA3 for the endpoint derivative approximations.

The strict monotonicity condition is required both to insure that division by zero does not occur in DAPROX and to insure proper operation of the search routine called by the evaluator SEVALH.

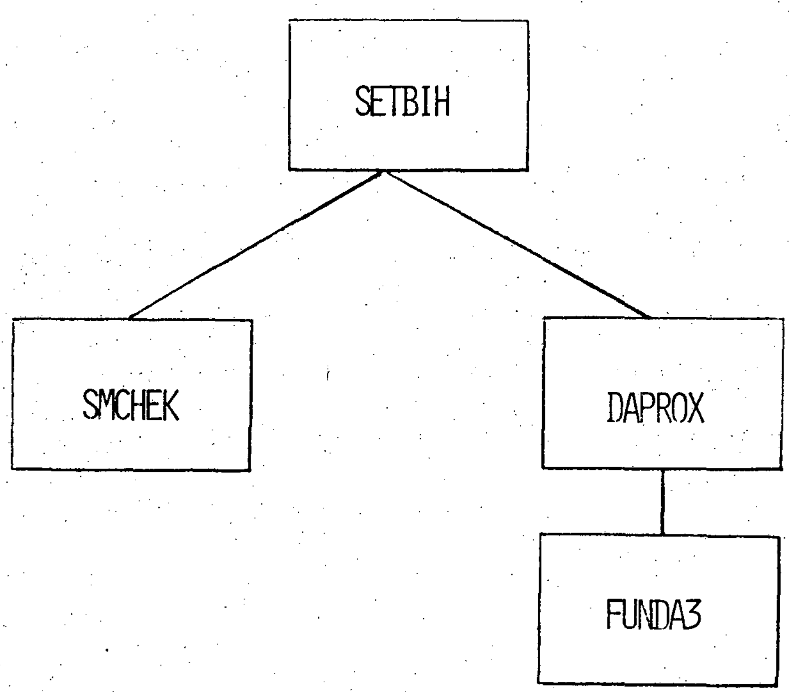


Figure 3. Structure of SETBIH

3.2.2. Calling Sequence. SETBIH is used as follows:

CALL SETBIH (MX, NX, NY, MFLAG, X, Y, F, W, IER)

On input:

MX is the row dimension (maximum value for the first index) of the array F.

NX,NY are the number of elements in the X and Y arrays, respectively. (Restrictions: $3 \le NX \le MX$, $3 \le NY$.)

MFLAG is the integer array of endpoint monotonicity controls discussed in section 3.1.2, above. It must be dimensioned at least 4. (Restrictions: $-1 \le MFLAG(k) \le +1$, $k = 1, \ldots, 4$.)

X,Y Are arrays of independent variable values defining a rectangular mesh R. (Restrictions: Both arrays must be strictly increasing.)

F Is the two-dimensional array of data values:

$$F(i,j) = f(X(i), Y(j)), \; i = 1, \ldots, NX, \; j = 1, \ldots, NY.$$

On Output:

W Contains the approximate derivatives FX, FY, FXY needed to define a bicubic Hermite interpolant to f on R. It must be dimensioned at least 3*MX*NY. The derivatives are stored in W as though they were two-dimensional arrays of the same structure as F:

```
FX  in W(1)          through W(  MX*NY)
FY  in W(MX*NY+1)    through W(2*MX*NY)
FXY in W(2*MX*NY+1)  through W(3*MX*NY)
```

IER Is an error flag. If IER $\neq 0$, one or more of the above restrictions on the input variables has been violated. In this case, W has not been changed.

IER = 1: The X array is not strictly increasing; that is, for some i, $X(i-1) \geq X(i)$.

IER = 2: The Y array is not strictly increasing; that is, for some j, $Y(j-1) \geq Y(j)$.

IER = 3: Neither the X nor the Y array is strictly increasing.

IER = 4: One or more of the following conditions has been violated: $3 \leq NX$, $3 \leq NY$, $-1 \leq MFLAG(k) \leq +1$ for $k = 1, \ldots, 4$.

3.3. SEVALH

Given the derivative approximations from SETBIH (or the actual derivative values, if known), SEVALH evaluates the bicubic Hermite approximant u at an arbitrary point (x,y) = (XVAL,YVAL).

3.3.1. Code Structure. SEVALH first locates the x- and y-intervals containing the evaluation point (XVAL,YVAL). This is done by two calls to a one-dimensional search routine SEARCH. Extrapolation is allowed by using the first cubic for values less than X(1) or Y(1), or the last cubic for values greater than X(NX) or Y(NY). NOTE: This search procedure is not especially efficient. The user who anticipates making heavy use of SEVALH should contact the NMG consultant for ideas on improving the efficiency of the search for his/her particular application.

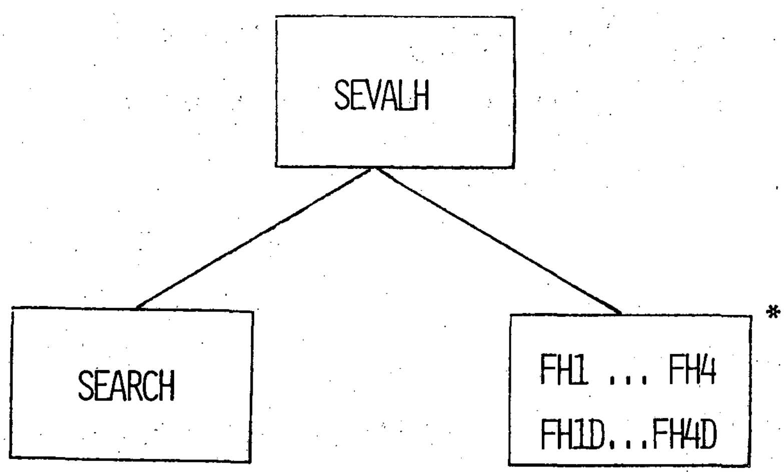


Figure 4. Structure of SEVALH

* This box represents the four one-dimensional Hermite basis functions FHi and their derivatives FHiD.

After the search, the appropriate one-dimensional Hermite basis functions and their derivatives are evaluated and stored in local arrays. In the notation of Section 2,

$$\left.\begin{array}{ll} AC(k,\ell) = a_{k,\ell} & \\ B1(k) = H_k(XVAL), & B2(k) = H_k'(XVAL) \\ C1(\ell) = G_\ell(YVAL), & C2(\ell) = G_\ell'(XVAL) \end{array}\right\} \begin{array}{l} k = 1, \ldots, 4 \\ \ell = 1, \ldots, 4 \end{array}$$

The same functions FH1, ..., FH4, FH1D, ..., FH4D are used here as in UNI (see Ref. 2). Finally, the approximant and its derivatives are evaluated via equations (5) - (8) of Section 2.

3.3.2. Calling Sequence. SEVALH assumes that the W array has been set up as described in Section 3.2.2, either by the user or by SETBIH. The numbers MX, NX, NY and the X, Y, F, and W arrays must not be changed from those used in the call to SETBIH. For each desired evaluation point, SEVALH is called as follows:

```
CALL SEVALH (MX, NX, NY, X, Y, F, W, XVAL, YVAL, V, IEXT)
```

On input:

MX, ..., F are the same as the correspondingly-named inputs to SETBIH.

W is the array of derivative values as output by SETBIH.

XVAL,YVAL are the x- and y-components of the point at which the interpolant is to be evaluated.

On output:

V is set to the values of the interpolant and its derivatives. It must be dimensioned at least 4.

$V(1) = F(XVAL,YVAL) = u(XVAL,YVAL)$

$V(2) = FX(XVAL,YVAL) = \partial u/\partial x\ (XVAL,YVAL)$

V(3) = FY(XVAL,YVAL) = $\partial u/\partial y$ (XVAL,YVAL)

V(4) = FXY(XVAL,YVAL) = $\partial^2 u/\partial x \partial y$ (XVAL,YVAL)

IEXT is an extrapolation flag, with the following meanings:

IEXT = 0: No extrapolation; i.e., X(1) $\leq$ XVAL $\leq$ X(NX), Y(1) $\leq$ YVAL $\leq$ Y(NY).

IEXT = 1: Extrapolation in x; i.e., XVAL < X(1) or XVAL > X(NX).

IEXT = 2: Extrapolation in y; i.e., YVAL < Y(1) or YVAL > Y(NY).

IEXT = 3: Extrapolation in both x and y.

This may be interpreted by the calling program either as an error flag or a warning flag, but the authors strongly caution against the use of BIHI for extrapolation much beyond the boundaries of R. Figure 5 shows pictorially the regions corresponding to various values of IEXT.

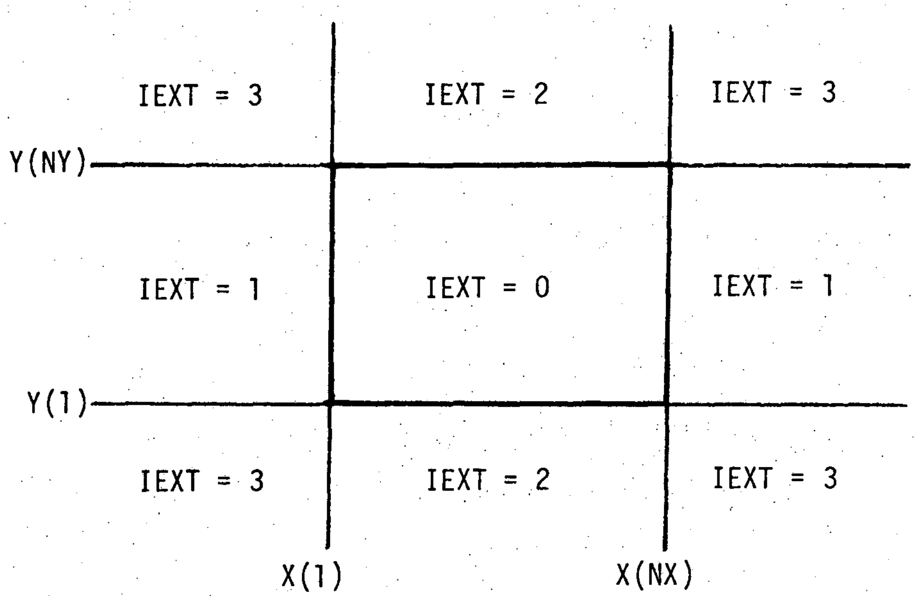


Figure 5. Correspondence between Regions and IEXT Values

4. NUMERICAL EXAMPLE

A sample data set is provided with BIHI. We include in Appendix B a listing of this file and its associated output. These data were generated from the function

$$f(x,y) = xy^2 + 5$$

on the rectangular mesh R defined by

$$x_i = X1 + (i-1)*DX \qquad i = 1, \ldots, NX \quad (NX = 4)$$
$$y_j = Y1 + (j-1)*DY \qquad j = 1, \ldots, NY \quad (NY = 3)$$

with X1 = 2., DX = 2., Y1 = 3., DY = 2. This mesh is illustrated in Figure 6. For completeness, we note that the true partial derivatives of this function are

$$\frac{\partial f}{\partial x}(x,y) = y^2 \quad ;$$

$$\frac{\partial f}{\partial y}(x,y) = 2xy \;;$$

$$\frac{\partial^2 f}{\partial x \partial y}(x,y) = 2y.$$

The net over which the interpolant and its derivatives are calculated is defined by

NXP = 5, XP1 = 1., DXP = 2.,

NYP = 4, XP1 = 2., DYP = 2.

This choice will test the extrapolation features in BIHI. Figure 6 shows the relation between R and the evaluation net. R is given by solid lines; the evaluation net by dashed lines.

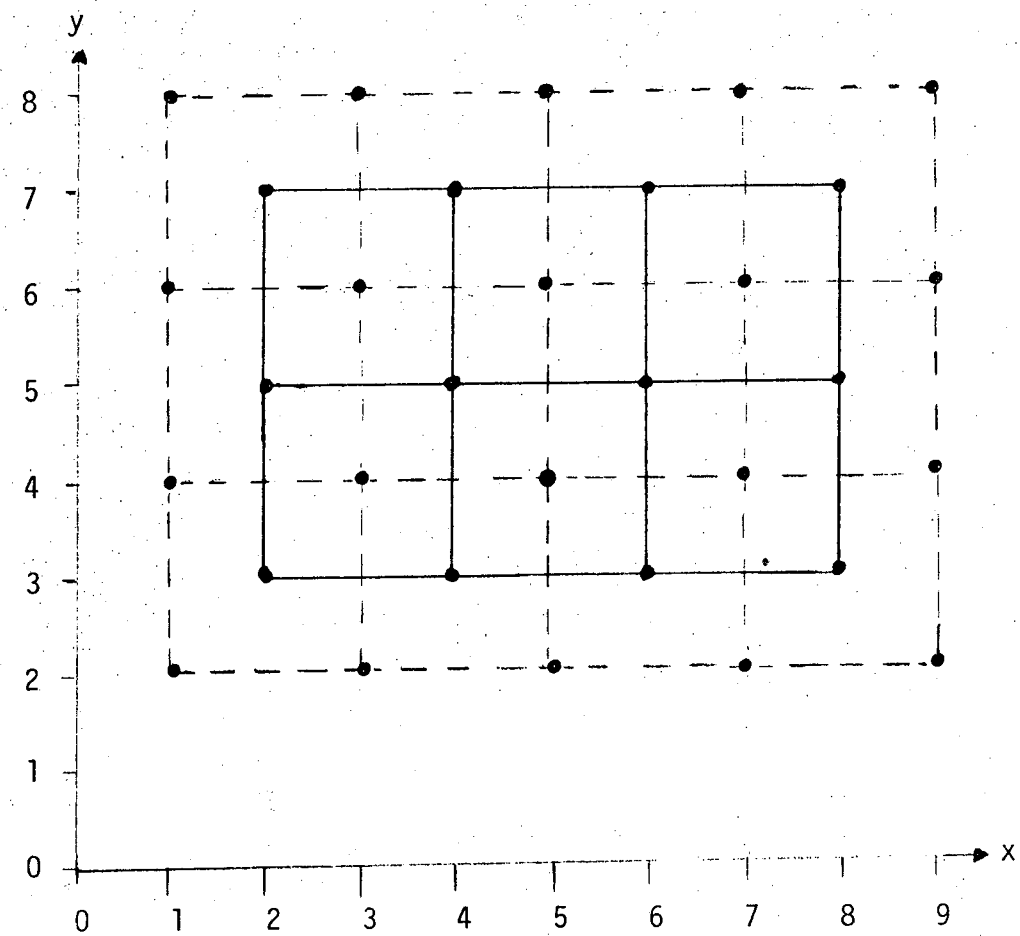


Figure 6. Data and Interpolation Meshes
for Sample Problem

The flags for this run are as follows:

ILOG1	= 0	(no logarithms)
ILOG2	= 0	
MFLAG(k)	= 0	for k = 1, ..., 4 (no monotonicity control)
IOPT	= 1	(Print interpolated values on original mesh.)

Since BIHI is exact for biquadratic functions, we expect the values of F, FX, FY, and FXY as calculated by BIHI to be equal to those computed from exact functions, given at the beginning of this section.

REFERENCES

1. Carlson, R. E., Piecewise Polynomial Functions, LLL Report UCID-17059 (January 1976).

2. Dickinson, R. P., Jr., and R. E. Carlson, UNI: Univariate Hermite and Spline Interpolation Code, LLL Report UCID-30140 (July 1976).

APPENDIX A. Three/Point Difference Formulas

In this section we derive the three point difference formulas used in BIHI. Let (x_1,y_1), (x_2,y_2) and (x_3,y_3) be three points in the plane as shown in Figure A-1.

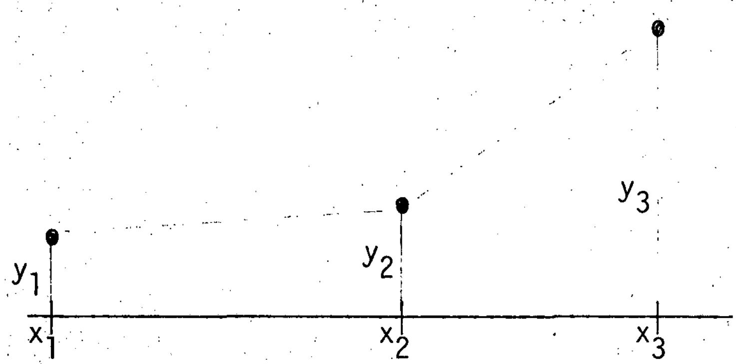


Figure A-1.

There is a unique quadratic polynomial q(x) which passes through these three points. By differentiating q(x) we get an approximation to y'(x). At each interior mesh point we use

$$(1)\quad y'(x_2) \approx q'(x_2) = \left[\frac{y_3-y_2}{x_3-x_2}\right]\left[\frac{x_2-x_1}{x_3-x_1}\right] + \left[\frac{y_2-y_1}{x_2-x_1}\right]\left[\frac{x_3-x_2}{x_3-x_1}\right] .$$

Note that this is the weighted average of the forward and backward differences.

At boundary mesh points it is necessary to approximate $y'(x_1)$ or $y'(x_3)$. These formulas are given below.

$$(2)\quad y'(x_1) \approx q'(x_1) = \left[\frac{y_2-y_1}{x_2-x_1}\right]\left[\frac{2(x_2-x_1)+(x_3-x_2)}{x_3-x_1}\right] + \left[\frac{y_3-y_2}{x_3-x_2}\right]\left[\frac{-(x_2-x_1)}{x_3-x_1}\right] .$$

$$(3)\ y'(x_3) \approx q'(x_3) = \left[\frac{y_2-y_1}{x_2-x_1}\right]\left[\frac{-(x_3-x_2)}{x_3-x_1}\right] + \left[\frac{y_3-y_2}{x_3-x_2}\right]\left[\frac{2(x_3-x_2)+(x_2-x_1)}{x_3-x_1}\right] .$$

Each of the above values of $q'(x_i)$ is a second order approximation to $y'(x_i)$.

APPENDIX B. Sample Problem

Below is a listing of the sample input file described in Section 4. The following pages contain the output produced by BIHI from these data.

```
     4       3       0       0       0       0       0       0       1
2.0000000E+00 3.0000000E+00 2.3000000E+01
2.0000000E+00 5.0000000E+00 5.5000000E+01
2.0000000E+00 7.0000000E+00 1.0300000E+02
4.0000000E+00 3.0000000E+00 4.1000000E+01
4.0000000E+00 5.0000000E+00 1.0500000E+02
4.0000000E+00 7.0000000E+00 2.0100000E+02
6.0000000E+00 3.0000000E+00 5.9000000E+01
6.0000000E+00 5.0000000E+00 1.5500000E+02
6.0000000E+00 7.0000000E+00 2.9900000E+02
8.0000000E+00 3.0000000E+00 7.7000000E+01
8.0000000E+00 5.0000000E+00 2.0500000E+02
8.0000000E+00 7.0000000E+00 3.9700000E+02
 5   4          1.00000E+00 2.00000E+00 2.00000E+00 2.00000E+00
```

1

BICUBIC HERMITE INTERPOLATION CODE

INTERPOLATION PARAMETERS:

NX = 4 NY = 3 ILOG1 = 0 ILOG2 = 0 IOPT = 1

MFLAG = 0 0 0 0

INPUT DATA:

	X(1)	X(2)	X(3)	X(4)
	2.0000000E+00	4.0000000E+00	6.0000000E+00	8.0000000E+00
Y(1) = 3.0000000E+00	2.3000000E+01	4.1000000E+01	5.9000000E+01	7.7000000E+01
Y(2) = 5.0000000E+00	5.5000000E+01	1.0500000E+02	1.5500000E+02	2.0500000E+02
Y(3) = 7.0000000E+00	1.0300000E+02	2.0100000E+02	2.9900000E+02	3.9700000E+02

OUTPUT MESH PARAMETERS:

NXP = 5 NYP = 4 XP1 = 1.00000E+00 YP1 = 2.00000E+00
DXP = 2.00000E+00 DYP = 2.00000E+00

1

VALUES OF INTERPOLANT AND DERIVATIVES ON INPUT MESH

X	Y	F	FX	FY	FXY
2.00000E+00	3.00000E+00	2.30000E+01	9.00000E+00	1.20000E+01	6.00000E+00
2.00000E+00	5.00000E+00	5.50000E+01	2.50000E+01	2.00000E+01	1.00000E+01
2.00000E+00	7.00000E+00	1.03000E+02	4.90000E+01	2.80000E+01	1.40000E+01
4.00000E+00	3.00000E+00	4.10000E+01	9.00000E+00	2.40000E+01	6.00000E+00
4.00000E+00	5.00000E+00	1.05000E+02	2.50000E+01	4.00000E+01	1.00000E+01
4.00000E+00	7.00000E+00	2.01000E+02	4.90000E+01	5.60000E+01	1.40000E+01
6.00000E+00	3.00000E+00	5.90000E+01	9.00000E+00	3.60000E+01	6.00000E+00
6.00000E+00	5.00000E+00	1.55000E+02	2.50000E+01	6.00000E+01	1.00000E+01
6.00000E+00	7.00000E+00	2.99000E+02	4.90000E+01	8.40000E+01	1.40000E+01
8.00000E+00	3.00000E+00	7.70000E+01	9.00000E+00	4.80000E+01	6.00000E+00
8.00000E+00	5.00000E+00	2.05000E+02	2.50000E+01	8.00000E+01	1.00000E+01
8.00000E+00	7.00000E+00	3.97000E+02	4.90000E+01	1.12000E+02	1.40000E+01

TABLE OF INTERPOLATED OR EXTRAPOLATED VALUES

X	Y	F	FX	FY	FXY	IEXT
1.00000E+00	2.00000E+00	9.00000E+00	4.00000E+00	4.00000E+00	4.00000E+00	3
1.00000E+00	4.00000E+00	2.10000E+01	1.60000E+01	8.00000E+00	8.00000E+00	1
1.00000E+00	6.00000E+00	4.10000E+01	3.60000E+01	1.20000E+01	1.20000E+01	1
1.00000E+00	8.00000E+00	6.90000E+01	6.40000E+01	1.60000E+01	1.60000E+01	3
3.00000E+00	2.00000E+00	1.70000E+01	4.00000E+00	1.20000E+01	4.00000E+00	2
3.00000E+00	4.00000E+00	5.30000E+01	1.60000E+01	2.40000E+01	8.00000E+00	0
3.00000E+00	6.00000E+00	1.13000E+02	3.60000E+01	3.60000E+01	1.20000E+01	0
3.00000E+00	8.00000E+00	1.97000E+02	6.40000E+01	4.80000E+01	1.60000E+01	2
5.00000E+00	2.00000E+00	2.50000E+01	4.00000E+00	2.00000E+01	4.00000E+00	2
5.00000E+00	4.00000E+00	8.50000E+01	1.60000E+01	4.00000E+01	8.00000E+00	0
5.00000E+00	6.00000E+00	1.85000E+02	3.60000E+01	6.00000E+01	1.20000E+01	0
5.00000E+00	8.00000E+00	3.25000E+02	6.40000E+01	8.00000E+01	1.60000E+01	2
7.00000E+00	2.00000E+00	3.30000E+01	4.00000E+00	2.80000E+01	4.00000E+00	2
7.00000E+00	4.00000E+00	1.17000E+02	1.60000E+01	5.60000E+01	8.00000E+00	0
7.00000E+00	6.00000E+00	2.57000E+02	3.60000E+01	8.40000E+01	1.20000E+01	0
7.00000E+00	8.00000E+00	4.53000E+02	6.40000E+01	1.12000E+02	1.60000E+01	2
9.00000E+00	2.00000E+00	4.10000E+01	4.00000E+00	3.60000E+01	4.00000E+00	3
9.00000E+00	4.00000E+00	1.49000E+02	1.60000E+01	7.20000E+01	8.00000E+00	1
9.00000E+00	6.00000E+00	3.29000E+02	3.60000E+01	1.08000E+02	1.20000E+01	1
9.00000E+00	8.00000E+00	5.81000E+02	6.40000E+01	1.44000E+02	1.60000E+01	3

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161
Price: Printed Copy $; Microfiche $3.00

Page Range	Domestic Price	Page Range	Domestic Price
001-025	$ 4.00	326-350	$12.00
026-050	4.50	351-375	12.50
051-075	5.25	376-400	13.00
076-100	6.00	401-425	13.25
101-125	6.50	426-450	14.00
126-150	7.25	451-475	14.50
151-175	8.00	476-500	15.00
176-200	9.00	501-525	15.25
201-225	9.25	526-550	15.50
226-250	9.50	551-575	16.25
251-275	10.75	576-600	16.50
276-300	11.00	601-up	[1]
301-325	11.75		

[1] Add $2.50 for each additional 100 page increment from 601 pages up.

Technical Information Department
LAWRENCE LIVERMORE LABORATORY
University of California | Livermore, California | 94550