SMOOTH SURFACE APPROXIMATION BY A LOCAL METHOD
OF INTERPOLATION AT SCATTERED POINTS

RICHARD FRANKE

Final Report for Period January - March 1978
Approved for Public Release; distributed unlimited
Prepared for: Chief of Naval Research
Arlington, VA 22217

78 06 15 063
The work reported herein was supported by the Foundation Research Program of the Naval Postgraduate School with funds provided by the Chief of Naval Research.

Reproduction of all or part of this report is authorized.

This report was prepared by:

RICHARD FRANKE
Associate Professor
Department of Mathematics

Reviewed by: CARROLL O. WILDE, Chairman
Department of Mathematics

Released by: W. M. TOLLES
(Acting) Dean of Research
This report describes a computer program which constructs a surface passing through a set of data points \((x_k,y_k,f_k), k=1,...,n\). It is based on previous work of the author (JIMA 19 (1977) 471-482), but uses a somewhat different approach which takes advantage of the nature of the approximations used and incorporates experience gained in the ensuing period. The surfaces are defined for all \((x,y)\) points and have continuous second partial derivatives.
1. The Interpolation Problem

Given the set of data points \((x_k,y_k,f_k)\), it is desired to construct a function \(F(x,y)\), such that \(F(x_k,y_k) = f_k\), \(k = 1,\ldots,n\). For large sets of data, it is desirable for the method to be local, that is, the value of \(F(x,y)\) depends only on the value of \(f_k\) at nearby points \((x_k,y_k)\).

This problem is receiving a great deal of attention and discussions of it and proposed methods can be found in [1], [2], [4], [7], and [8].

2. The Interpolation Scheme

This is a local method, the general idea having been discussed in [4]. The basic idea is to construct local interpolants, \(F(x,y)\), which are then weighted by functions \(W(x,y)\) having limited support to obtain the function

\[
F(x,y) = \sum_{k} W(x,y)F(x,y)/\sum_{k} W(x,y)
\]

The details are fully discussed in the reference, but the important fact is that \(F(x,y)\) will take on the value \(f_k\) at \((x_k,y_k)\) if for each \(k\) where \(W(x_k,y_k) = 0\), \(F(x_k,y_k) = f_k\). In the referenced paper, the weight functions \(W(x,y)\) were taken to be of the form

\[
W(x,y) = \begin{cases} 
1 - 3(d/r)^2 + 2(d/r)^3 & , \quad d \leq r \\
0 & , \quad d > r 
\end{cases}
\]

where \(r\) is the radius of the smallest circle centered at \((x_k,y_k)\) which contains a given fixed number of data points, and \(d\) is the distance from \((x,y)\) to \((x_k,y_k)\).

This scheme, used with local interpolants, \(F(x,y)\), which were taken to be certain optimal approximations, yielded reasonably good results. However, the computational burden was rather high. This was due in part to a great deal of overlap in the regions where the \(W(x,y)\) are nonzero. In addition, the approxi-
imation is defined only on the union of the circles around each of the data points, which could cause problems.

One of the advantages of using optimal approximations is that the basis functions are generated by the \((x_k, y_k)\) and the system is automatically nonsingular. Within some limitations, the resulting equations can easily be solved by Cholesky decomposition [5]. This opens the way for the present approach, which is to choose rectangular regions on which weight functions are non-zero, thus being able to carefully govern the amount of overlap with a resulting decrease in the necessary computations, as well as simplification of the weight functions.

2.1. Weight Functions

With these ideas in mind, we are now ready to describe the present selection of regions over which the weight functions are non-zero. These regions will be rectangles defined by the following parameters. Let \(n_x\) and \(n_y\) be given positive integers and let finite values of \(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_{n_x}\) and \(\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_{n_y}\) be given. For convenience in this section we let

\[
\tilde{x}_0 = \tilde{y}_0 = -\infty \quad \text{and} \quad \tilde{x}_{n_x+1} = \tilde{y}_{n_y+1} = \infty .
\]

For each \(i = 1, \ldots, n_x\) and \(j = 1, \ldots, n_y\), let \(R_{ij}\) denote the rectangle \((\tilde{x}_{i-1}, \tilde{x}_{i+1}) \times (\tilde{y}_{j-1}, \tilde{y}_{j+1})\).

Let \(H_5(s) = 1 - s^3(6s^2 - 15s + 10)\), the Hermite quintic satisfying \(H_5(0) = 1, H_5'(0) = H_5''(0) = H_5'(1) = H_5''(1) = 0\). We then define functions which are piecewise quintics with continuous second derivatives which have the property that they are non-zero on two intervals and satisfy

\[
V_i(x_j) = \delta_{ij} , \quad i = 1, \ldots, n_x , \quad j = 0, 1, \ldots, n_x + 1
\]

\[
U_j(y_i) = \delta_{ij} , \quad j = 1, \ldots, n_y , \quad i = 0, 1, \ldots, n_y + 1
\]
In particular,

\[ V_1(x) = \begin{cases} 
1 & , x < \tilde{x}_1 \\
\frac{x - \tilde{x}_1}{\tilde{x}_2 - \tilde{x}_1} & , \tilde{x}_1 \leq x < \tilde{x}_2 \\
0 & , x \geq \tilde{x}_2 
\end{cases} \]

\[ V_i(x) = \begin{cases} 
0 & , x < \tilde{x}_{i-1} \\
1 - V_{i-1}(x) & , \tilde{x}_{i-1} \leq x < \tilde{x}_i \\
\frac{x - \tilde{x}_i}{\tilde{x}_{i+1} - \tilde{x}_i} & , \tilde{x}_i \leq x < \tilde{x}_{i+1} \\
0 & , x \geq \tilde{x}_{i+1} 
\end{cases} \]

for \( i = 2, \ldots, n_x - 1 \), and

\[ V_{n_x}(x) = \begin{cases} 
0 & , x < \tilde{x}_{n_x-1} \\
1 - V_{n_x-1}(x) & , \tilde{x}_{n_x-1} \leq x < \tilde{x}_{n_x} \\
1 & , x \geq \tilde{x}_{n_x} 
\end{cases} \]

The \( U_j(Y) \) are dual. Then, if we define

\[ W_{ij}(x,y) = V_i(x)U_j(Y), \quad i = 1, \ldots, n_x, \quad j = 1, \ldots, n_y, \]

it is easily observed that the function \( W_{ij}(x,y) \) has support \( \text{Cl}(R_{ij}) \) and that the functions form a partition of unity for the plane, i.e.,

\[ \sum_{i,j} W_{ij}(x,y) = 1 \quad \text{for all } (x,y). \]

These properties allow the construction of the interpolation function (1) to proceed easily since any point \((x,y)\) is in at most four \( R_{ij} \), and the denominator of (1) is always \( \equiv 1 \), allowing us to write

\[ F(x,y) = \sum_{i,j} W_{ij}(x,y)Q_{ij}(x,y). \]

We again emphasize that at most four terms in the sum are non-zero.
The appropriate choice of $x_i$ and $y_j$ as well as $n_x$ and $n_y$ depend on the data as well as the choice of local interpolating functions $Q_{ij}(x,y)$. For this reason we defer discussion of the selection of these grid lines until after we discuss the choice of $Q_{ij}(x,y)$.

2.2. Local Interpolation Functions

The only restriction on the local interpolation functions $Q_{ij}(x,y)$ are that they interpolate all data points in $R_{ij}$ and that they are defined for all $(x,y)$ in $R_{ij}$. Polynomials sometimes fail to satisfy these conditions. The use of optimal approximations in Sard corner spaces has been investigated [5], and for small numbers of data points, the approximations can be computed in straightforward fashion. One possible defect in such approximations is their lack of polynomial precision: even constants are not approximated exactly. With only a slight complication this can be overcome, since by a theorem of Barnhill and Gregory [3], the boolean sum operator $B \oplus L$ has the interpolation properties of $B$ and the function precision of $L$. Here we are thinking of $Bf$ as the optimal approximation in $B_{[2,2]}$ while $Lf$ is the least squares fit by a linear function.

The implemented version of the program embodies three options: (1) Use optimal approximations in $B_{[2,2]}$ as the local interpolation functions; (2) Use the least squares linear approximation instead of an interpolation function; and (3) Use the optimal approximation in $B_{[2,2]}$ boolean sum the least squares linear approximation. The second option yields a surface which in general does not interpolate the given data. The third option is achieved computationally as $(B \oplus L)f = (B + L - BL)f = Lf + B(f - Lf)$.

The use of the boolean sum has a desirable effect in that it removes much of the effect of linear transformations of the data on the overall approximation.
However, for complete consistency with respect to translation and change of the measure of distance, each rectangle

\[ [\tilde{x}_{i-1}, \tilde{x}_{i+1}] \times [\tilde{y}_{j-1}, \tilde{y}_{j+1}] \]

is transformed to the unit square for the optimal approximation. For these purposes, we take \( \tilde{x}_0 = \min x_k \) and \( \tilde{x}_{n+1} = \max x_k \), and the dual in \( y \).

The base point \((a, b)\) is taken to be \((0, 0)\) for all \(i, j \geq 2\), while it is taken to be \((1, 1)\) for \(i = j = 1\), \((0, 1)\) for \(i \geq 2, j = 1\), and \((1, 0)\) for \(i = 1, j \geq 2\). This yields lines of discontinuity in the second derivatives which are nowhere interior to the support regions for the weight functions \( W_{ij} \), thus assuring continuous second derivatives in the overall approximation.

The overall approximation is invariant with respect to linear transformations which leave the directions of the axes unchanged. Since lines of discontinuities in the third derivatives occur along horizontal and vertical lines the approximation is not invariant with respect to rotations.

The points associated with \( R_{ij} \) include all the points in the closure of \( R_{ij} \). Because approximation by a linear function requires at least three points, a parameter \( \text{MINPTS} \), is used to assure that at least \( \text{MINPTS} \) points are selected for each \( R_{ij} \). If extra points are required, they are taken as the closest points in the sup norm, distance being measured after

\[ [x_{i-1}, x_{i+1}] \times [y_{j-1}, y_{j+1}] \]

has been transformed onto \([0, 1]^2\). Presently \( \text{MINPTS} \) is set to three and this has been satisfactory. It is easily changed, if desired or necessary. For example, if some \( R_{ij} \) has only three colinear points associated with it, the scheme will fail under options (2) or (3). Then one must either increase the value of \( \text{MINPTS} \) or use option (1).
2.3. Selection of Grid Lines

It is desirable to have automatic selection of grid lines, that is, values of $\tilde{x}_i$ and $\tilde{y}_j$. This should be accomplished in some manner which results in rectangles $R_{ij}$ which contain approximately equal numbers of points. For data which is poorly distributed this may not be possible. However, for somewhat uniformly distributed points the process we describe here works quite well.

The selection of the grid lines is determined by one parameter, called NPPR, for "number of points per rectangle." The grid lines are then chosen so that there will be approximately NPPR points in each rectangle, $R_{ij}$. If there are additional points added to certain rectangles to make up MINPTS points the average may be higher. The average is, of course, dependent on the data set.

Equal numbers of grid lines are chosen in each direction, that is $n_x = n_y$. Because we want NPPR points per rectangle, each subrectangle $(\tilde{x}_i, \tilde{x}_{i+1}) \times (\tilde{y}_j, \tilde{y}_{j+1})$ should have $\frac{1}{4}$ NPPR points. Thus we want to choose $n_x = n_y$ so that $(n_x + 1)^2 \cdot \frac{1}{4} \text{NPPR} = n$, the total number of data points. Thus, we take $n_x$ to be the nearest integer to $(4n/\text{NPPR})^{1/2} - 1$.

Grid lines, that is $\tilde{x}_i$ and $\tilde{y}_j$ values, are now determined by choosing these values so that approximately $n/(n_x + 1)$ points occur in each $(\tilde{x}_i, \tilde{x}_{i+1})$ and each $(\tilde{y}_j, \tilde{y}_{j+1})$. Specifically, let $\hat{x}_k$ denote the values of $x_k$ given in nondecreasing order, then $x_i = \hat{x}_k$, where $k$ is the integer nearest in$/ (n_x + 1)$ for $i = 1, 2, \ldots, n_x$. The selection in $y$ is dual.
3.0. Implementation

The scheme is implemented in a set of subprograms, only one of which is normally referenced by the user. The hierarchy of subprograms is given in figure 1. A brief description of them, according to level, follows.

![Diagram of subprograms](image)

### 3.1.1. User's Program

The user's program must provide the data points, \((x_k, y_k, f_k), k = 1, ..., n\), as well as the points \(x_0, y_0\) for the grid of points at which the interpolation function is to be evaluated. In addition the user's program must provide workspace arrays, IWK and WK, and an array FO for the returned function values. The amount of storage required in the arrays IWK and WK are not known a priori, but are estimated as follows.

For IWK, approximately \(4n(1 + 1/NPPR)\) locations are required. This has generally proven to be an overestimate. For WK, the storage required depends on the type of local approximation used and is approximately:

*This is a Cholesky decomposition equation solver in the IMSL Library, which itself references several other subroutines from the IMSL Library.*
for MODE = 1, \(4(n + \sqrt{n/NPPR})\);
for MODE = 2, \(4(3n/NPPR + \sqrt{n/NPPR})\); and
for MODE = 3, \(4(n + 3n/NPPR + \sqrt{n/NPPR})\).

As with the estimates for IWK, these estimates have proven to generally be overestimates. The precise number of storage locations required in each array are returned to the user's program by the principal subroutine.

Under the usual option, the user specifies \(NPPR > 0\). If the user wishes to specify the grid lines, \(NPPR\) is set to zero, and additional input in the arrays IWK and WK is necessary. In the array IWK, the user specifies \(n_x\) in IWK(1) and \(n_y\) in IWK(2). The grid lines are specified in the array WK, according to the following.

\(WK(1)\) is \(\min_{k} x_k\),
\(WK(2),\ldots,WK(n_x + 1)\) are the vertical grid values \(\tilde{x}_1,\ldots,\tilde{x}_{n_x}\), in increasing order,
\(WK(n_x + 2)\) is \(\max_{k} x_k\),
\(WK(n_x + 3)\) is \(\min_{k} y_k\),
\(WK(n_x + 4),\ldots,WK(n_x + n_y + 3)\) are the horizontal grid values \(\tilde{y}_1,\ldots,\tilde{y}_{n_y}\) in increasing order, and
\(WK(n_x + n_y + 4)\) is \(\max_{k} y_k\).

3.2.1. Subroutine LOB 22

This subroutine provides the interface between the user and the set of subroutines which implement the method. Generally LOB 22 sets up storage locations in the arrays IWK and WK, determines parameters required by other subroutines, and calls subroutines to (1) generate the grid (if necessary), (2) determine the interpolation points for each rectangle, (3) compute coefficients for the local interpolating functions, and finally, (4) to evaluate the interpolation function on the desired grid of points.
3.3.1. Subroutine GRID

This subroutine selects the values of $\tilde{x}_i$ and $\tilde{y}_j$ in accordance with the discussion in Section 2.3.

3.3.2. Subroutine LOCLIP

This subroutine determines the local interpolation points for each $R_{ij}$ in accordance with the last paragraph of Section 2.2.

3.3.3. Subroutine CLOB 22

This subroutine computes the coefficients in the least squares plane (MODE = 2 or 3) and the coefficients in the optimal approximation (MODE = 1 or 3) or each of the rectangles $R_{ij}$. In the present implementation the IMSL Cholesky decomposition equation solver LEQT1P is used. This could be replaced at a facility where IMSL is not available, although according to IMSL policy, LEQT1P (and associated subroutines) can be used as part of this package at any facility. Because of the short single precision word length on the IBM 360/370 series computers, on which this program was implemented, the coefficients for the system of equations for the optimal approximation are generated in double precision. On computers with a longer word length, the double precision variables in this routine can be safely removed. Other double precision statements occur in EVLB 22 and function K, which must also be removed.

3.3.4. Subroutine EVLB 22

This subroutine evaluates the approximation (2) on the set of points $(x_0_i, y_0_j)$ as specified by the user, and returns the values in F0. As noted above, the double precision variables in this subroutine should be removed on computers with longer word length than the IBM 360/370 series.
3.4.1. **Function K**

This function evaluates the representers for point evaluation functionals in $B_{2,2}$. For evaluation at $(u,v)$, base point at $(a,b)$, the representor, as a function of $(s,t)$ [6] is

$$
K(a,b;u,v,s,t) = g_2(a;u,s)g_2(b;v,t),
$$

where

$$
g_2(a;u,s) = G_2(a;u,s) = (s-u)^3 + (u-a)(s-a) + (u-a)(s-a)^2 + (s-a)^3
$$

for $a \leq u$, and

$$
g_2(a;u,s) = G_2(-a;-u,-s) \text{ for } u < a.
$$

The arguments of this function are all single precision, but because of the short word length of the IBM 360/370 computers, all calculations are performed in double precision, and the returned value is double precision. On computers with longer word lengths these calculations can be done in single precision.

4.0. **Examples**

The method has been applied to a number of sets of data with good results. Figures 3 - 5 show test surfaces and results of applying the method for each of the options for local approximations. The three surfaces are described by

(C) $F(x,y) = \tanh (y - x) + 1$,

(S) $F(x,y) = 3/2[\cos(3/5(y - 1)) + 5/4]/[1 + (x-4)^2]$, and

(E) $F(x,y) = 9(3/4 \exp(-(x-3)^2 - (y-3)^2) + \exp(-(x-7)^2 - (y-10)^2)) - \frac{1}{5} \exp(-(x - 5)^2 - (y - 8)^2) + \frac{1}{2} \exp(-(x-8)^2 - (y-4)^2)$,

respectively. The 100 interpolation points were chosen at random within a unit square centered at $(i,j)$ for $i,j = 1,2,\ldots,10$. The points are shown in Figure 2 as +’s, with the convex hull shown by dashed lines, while the square
on which the resulting interpolation functions were evaluated, is given by the solid lines. The diagonal line shows the direction toward the viewing point.

There does not appear to be a great deal of difference between the optimal approximation and the optimal approximation boolean sum least squares plane. Generally the latter option has slightly smaller errors and slightly less noticeable defects. Gross defects in the approximations can generally be traced to a lack of data in that particular part of the region.

The effect of varying the parameter NPPR is shown in Figures 6 - 14. Some general observations are possible from this set of views. Most apparent is the fact that option (2), the least squares plane fit as the local approximation does not appear to lead to very good results. In general, however, the smaller value of NPPR gives better results, visually, and usually better accuracy, too.

The choice of NPPR = 6 for options (1) and (3) appears to be a reasonable one. For surfaces with sharp gradients, as in Figure 6, it appears that localizing the behavior as much as possible with a smaller value of NPPR is the best strategy. For smooth surfaces, such as in Figure 9 and 12 it appears the opposite is true, where NPPR = 8 seems to lead to the best results.

The storage and timing results are given in Table 1. The storage refers to requirements of the two workspace arrays provided by the user. The timing is for calculation of the 1089 points generated for the plots. The program was run under the Fortran H compiler on the IBM 360 model 67 at the Naval Postgraduate School. Computation times are dependent on external factors and may vary from run to run.
5.0. Acknowledgements

During the first half of 1977 the author was a Visiting Associate Professor at the University of Utah. Interactions with Professor R. E. Barnhill and his students on the subject of surface approximation proved to be fruitful. The kernel of a number of ideas in the present scheme germinated during that time. Thanks also go to Rosemary E. Chang of Sandia Laboratories (Livermore) who first undertook to run the program on a CDC computer. Improvements in the program description and the test program were a result of those efforts.
<table>
<thead>
<tr>
<th>NPPR</th>
<th>MODE</th>
<th>NIWK</th>
<th>NWK</th>
<th>Approx. Time (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>437</td>
<td>375</td>
<td>7.9</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>437</td>
<td>265</td>
<td>1.3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>437</td>
<td>618</td>
<td>8.2</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>380</td>
<td>346</td>
<td>10.9</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>380</td>
<td>165</td>
<td>1.1</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>380</td>
<td>493</td>
<td>11.1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>351</td>
<td>328</td>
<td>12.9</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>351</td>
<td>124</td>
<td>1.0</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>351</td>
<td>436</td>
<td>13.1</td>
</tr>
</tbody>
</table>

Table 1
Test Surface
Cliff Function

Mode = 1, $E_{\text{max}} = 0.468$
$E_{\text{rms}} = 0.0263$
$E_{\text{mean}} = 0.0526$

Mode = 2, $E_{\text{max}} = 0.283$
$E_{\text{rms}} = 0.0523$
$E_{\text{mean}} = 0.0864$

Mode = 3, $E_{\text{max}} = 0.466$
$E_{\text{rms}} = 0.0257$
$E_{\text{mean}} = 0.0527$

Figure 3 (NPPR = 6)
Test Surface

Mode = 1, \( E_{\text{max}} = 0.187 \)
\( E_{\text{rms}} = 0.0156 \)
\( E_{\text{mean}} = 0.0273 \)

Mode = 2, \( E_{\text{max}} = 0.389 \)
\( E_{\text{rms}} = 0.0495 \)
\( E_{\text{mean}} = 0.0739 \)

Saddle Function

Mode = 3, \( E_{\text{max}} = 0.178 \)
\( E_{\text{rms}} = 0.0148 \)
\( E_{\text{mean}} = 0.0265 \)

Figure 4 (NPPR = 6)
Test Surface

Mode = 1, $E_{\text{max}} = 0.974$
$E_{\text{rms}} = 0.0929$
$E_{\text{mean}} = 0.169$

Mode = 2, $E_{\text{max}} = 0.216$
$E_{\text{rms}} = 0.209$
$E_{\text{mean}} = 0.366$

Mode = 3, $E_{\text{max}} = 0.827$
$E_{\text{rms}} = 0.0757$
$E_{\text{mean}} = 0.133$

Figure 5 (NPPR = 6)
Figure 6 (Mode = 1)
Test Surface
Cliff Function

NPPR = 4, $E_{\text{max}} = 0.336$
$E_{\text{rms}} = 0.0435$
$E_{\text{mean}} = 0.0797$

NPPR = 6, $E_{\text{max}} = 0.283$
$E_{\text{rms}} = 0.0523$
$E_{\text{mean}} = 0.0864$

NPPR = 8, $E_{\text{max}} = 0.375$
$E_{\text{rms}} = 0.0692$
$E_{\text{mean}} = 0.113$

Figure 7 (Mode = 2)
Test Surface
Cliff Function

NPPR = 4, \( E_{\text{max}} = .261 \)
\( E_{\text{rms}} = .0246 \)
\( E_{\text{mean}} = .0500 \)

NPPR = 6, \( E_{\text{max}} = .468 \)
\( E_{\text{rms}} = .0263 \)
\( E_{\text{mean}} = .0526 \)

NPPR = 8, \( E_{\text{max}} = .462 \)
\( E_{\text{rms}} = .0304 \)
\( E_{\text{mean}} = .0622 \)

Figure 8 (Mode = 3)
Test Surface
Saddle Function

NPPR = 4, \( E_{\text{max}} = 0.208 \)
\( E_{\text{rms}} = 0.0249 \)
\( E_{\text{mean}} = 0.0398 \)

NPPR = 6, \( E_{\text{max}} = 0.187 \)
\( E_{\text{rms}} = 0.0156 \)
\( E_{\text{mean}} = 0.0273 \)

NPPR = 8, \( E_{\text{max}} = 0.154 \)
\( E_{\text{rms}} = 0.0118 \)
\( E_{\text{mean}} = 0.0211 \)

Figure 9 (Mode = 1)
Figure 10 (Mode = 2)

Test Surface
Saddle Function.

NPPR = 4, $E_{\text{max}} = 0.247$
$E_{\text{rms}} = 0.0338$
$E_{\text{mean}} = 0.0487$

NPPR = 6, $E_{\text{max}} = 0.309$
$E_{\text{rms}} = 0.0495$
$E_{\text{mean}} = 0.0739$

NPPR = 8, $E_{\text{max}} = 0.336$
$E_{\text{rms}} = 0.0565$
$E_{\text{mean}} = 0.0803$
Test Surface
Saddle Function

NPPR = 4, $E_{\text{max}} = 0.244$
$E_{\text{rms}} = 0.0211$
$E_{\text{mean}} = 0.0363$

NPPR = 6, $E_{\text{max}} = 0.178$
$E_{\text{rms}} = 0.0148$
$E_{\text{mean}} = 0.0265$

NPPR = 8, $E_{\text{max}} = 0.148$
$E_{\text{rms}} = 0.0115$
$E_{\text{mean}} = 0.0202$

Figure 11 (Mode = 3)
Test Surface
Exponentials

NPPR = 4, \( E_{\text{max}} = 1.20 \)
\( E_{\text{rms}} = 0.128 \)
\( E_{\text{mean}} = 0.227 \)

NPPR = 6, \( E_{\text{max}} = 0.974 \)
\( E_{\text{rms}} = 0.0929 \)
\( E_{\text{mean}} = 0.169 \)

NPPR = 8, \( E_{\text{max}} = 0.703 \)
\( E_{\text{rms}} = 0.0779 \)
\( E_{\text{mean}} = 0.129 \)

Figure 12 (Mode = 1)
Test Surface

Exponentials

NPPR = 4, $E_{\text{max}} = 1.29$

$E_{\text{rms}} = 0.162$

$E_{\text{mean}} = 0.265$

NPPR = 6, $E_{\text{max}} = 2.16$

$E_{\text{rms}} = 0.209$

$E_{\text{mean}} = 0.366$

NPPR = 8, $E_{\text{max}} = 2.04$

$E_{\text{rms}} = 0.251$

$E_{\text{mean}} = 0.398$

Figure 13 (Mode = 2)

-27-
Test Surface
Exponentials

NPPR = 4, \( E_{\text{max}} = 1.30 \)
\( E_{\text{rms}} = 0.191 \)
\( E_{\text{mean}} = 0.189 \)

NPPR = 6, \( E_{\text{max}} = 0.827 \)
\( E_{\text{rms}} = 0.0757 \)
\( E_{\text{mean}} = 0.133 \)

NPPR = 8, \( E_{\text{max}} = 0.748 \)
\( E_{\text{rms}} = 0.0778 \)
\( E_{\text{mean}} = 0.131 \)

Figure 14 (Mode = 3)
Appendix: Program Listings and Sample Output
DIMENSION X(25), Y(25), F(25), XC(3), YQ(3), FO(3,3), FAC(3,3,4), TST00010
1 E(3,3), IWK(100), WK(120)
DATA NOUT/, 2.557525042, 1.156302503, 0.343660482,
1 1.686594904, 0.729563464, 0.211667614,
2 0.377497943, 0.343550958, 0.393714350,
3 2.658641228, 1.522739473, 0.682157263,
4 1.754474583, 0.877577736, 0.284611789,
5 0.933759918, 0.413208351, 0.109657470,
6 2.572645288, 1.158049264, 0.348002564,
7 1.694989136, 0.724123412, 0.214329225,
8 0.736691672, 0.341271688, 0.093442275,
9 2.376248312, 1.107179709, 0.346220775,
A 1.62663927, 0.74742886, 0.217782524,
B 0.71195600, 0.324808684, 0.097759180 /

C
100 F(K) = 4.*EXP(-(X(K)**2+Y(K)**2)**2)

C
DO 100 I=1,5
DO 100 J=1,5
K = K+1
X(K) = I-1+J/FLOAT(K)
Y(K) = J-1/FLOAT(K)
K = K+1
100 F(K) = 4.*EXP(-(X(K)**2+Y(K)**2)**2)
END
WK(5) = 0.0
WK(6) = 1.25
WK(7) = 2.9
WK(8) = 3.8
WK(9) = 4.96
NWK = 120
NIWK = 100
MODE = 3
CALL LOB22 (MODE, O, NP, X, Y, F, 3, X0, 3, YO, IWK, NIWK, WK, NWK, FO, KER)
C
DO 180 I=I,3
DO 180 J=1,3
E(I,J) = FO(I,J) - FAC(I,J,4)
C
WRITE (NOUT,2) MODE, KER, NIWK, NWK
WRITE (NOUT,1) FO, E
STOP
C
1 FORMAT (/7X,15HFUNCTION VALUES,3(/3F20.6)/7X,50HDEVIATIONS (THESEI
1 VALUES REPRESENT ROUNDOFF ERROR)/(3F20.6))
2 FORMAT (43H) THE VALUES OF MODE, KER, NIWK, AND NWK ARE, 415
END

TST00490
TST00500
TST00510
TST00520
TST00530
TST00540
TST00550
TST00560
TST00570
TST00580
TST00590
TST00600
TST00610
TST00620
TST00630
TST00640
TST00650
TST00660
TST00670
TST00680
TST00690
TST00700
SUBROUTINE LOB22 (MODE, NPPI, NPI, XI, YI, FI, NXC, XO, NYO, YO, IWK, NIWK, WK, NWK, F0, KER) 

1

THIS SUBROUTINE SERVES AS A USER INTERFACE TO THE SET OF SUBROUTINES THAT IMPLEMENT FRANK'S METHOD OF SURFACE INTERPOLATION. RECTANGULAR REGIONS ARE USED WITH PRODUCT QUINTIC HERMITE WEIGHT FUNCTIONS. THE RECTANGLES ARE CHOSEN IN AN ATTEMPT TO OBTAIN ABOUT NPPI POINTS IN EACH REGION. THE SAME NUMBER OF GRID LINES IS USED IN EACH DIRECTION. LOCAL INTERPOLATION FUNCTION ARE EITHER OPTIMAL APPROXIMATIONS IN NDX SPACE BY CORNER 2 OR OPTIMAL APPROXIMATIONS IN A CORNER 2 BOOLEAN SUM THE LEAST SQUARES PLANES, OR FOR APPROXIMATION RATHER THAN INTERPOLATION, THE LEAST SQUARES PLANES.

THE ARGUMENTS ARE AS FOLLOWS.

MODE - INPUT.

= 1, INDICATES THE STATUS OF THE CALCULATION.

SET UP THE PROBLEM, USE OPTIMAL APPROXIMATIONS, AND RETURN THE GRID OF INTERPOLATED POINTS INDICATED BY NXC, XO, NYO, YC IN FO.

= 2, SET UP THE PROBLEM, USE THE LEAST SQUARES PLANE FOR THE LOCAL APPROXIMATION, AND RETURN THE GRID OF INTERPOLATED POINTS INDICATED BY NXC, XO, NYO, YC IN FO.

= 3, SET UP THE PROBLEM, USE THE OPTIMAL APPROXIMATIONS BOOLEAN SUM THE LEAST SQUARES PLANES FOR THE LOCAL APPROXIMATIONS, AND RETURN THE GRID OF INTERPOLATED POINTS INDICATED BY NXC, XO, NYO, YO IN FO.

= 4, THE PROBLEM HAS BEEN SET UP PREVIOUSLY, CALCULATE THE GRID OF INTERPOLATED POINTS INDICATED BY NXC, XO, NYO, YO IN FO. THE PROGRAM ASSUMES THAT ALL ARGUMENTS EXCEPT NXC, XO, NYO, YO, FO, AND MODE ARE UNCHANGED FROM THE PREVIOUS CALL. THE SUGGESTED VALUE IS SIX. SHOULD BE AT LEAST FOUR. VALUES LARGER THAN TEN ARE NOT ADVISED. IF THE USER WISHES TO SPECIFY HIS OWN GRID LINES, X TILDA AND Y TILDA, HE MAY DO SO BY SETTING NPPI = 0 AND SETTING NECESSARY VALUES IN THE ARRAYS IWK AND WK.

NPPI - INPUT.

NUMBER OF INPUT DATA POINTS.

XI - INPUT.

THE DATA POINTS (XI, YI, FI), I = 1,..., NPI.

FI - INPUT.

THE NUMBER OF XO VALUES AT WHICH THE INTERPOLATION FUNCTION IS TO BE CALCULATED.
XO - INPUT.  THE VALUES OF X AT WHICH THE INTERPOLATION FUNCTION IS TO BE CALCULATED.

Y0 - INPUT.  THE VALUES OF Y AT WHICH THE INTERPOLATION FUNCTION IS TO BE CALCULATED.

IWK - INPUT AND OUTPUT.  THIS ARRAY IS OUTPUT WHEN MODE = 1, 2, OR 3, AND IS INPUT WHEN MODE = 4.  THIS MUST BE AN ARRAY DIMENSIONED APPROXIMATELY 4*NPI+1 FOR NPPR = 6, THIS MEANS ABOUT 5*NPI.  WHEN NPPR IS INPUT AS ZERO THE USER MUST SPECIFY THE NUMBER OF VERTICAL GRID LINES (THE NUMBER OF X TILDA VALUES) IN IWK(1) AND THE NUMBER OF HORIZONTAL GRID LINES (THE NUMBER OF Y TILDA VALUES) IN IWK(2).

NIWK - INPUT AND OUTPUT.  WHEN MODE = 1, 2, OR 3 THIS MUST BE SET TO THE NUMBER OF LOCATIONS RESERVED FOR THE ARRAY IWK.  ON ENTRY IT IS SET TO THE ACTUAL NUMBER OF LOCATIONS REQUIRED FOR THE ARRAY.  IF THE ARRAY IWK IS NOT DIMENSIONED LARGE ENOUGH IT MAY CAUSE A FAILURE SINCE THE ARRAY BOUNDS WILL BE EXCEEDED IN SUBROUTINE LOCIP.

WK - INPUT AND OUTPUT.  THIS ARRAY IS OUTPUT WHEN MODE = 1, 2, OR 3, AND IS INPUT WHEN MODE = 4.  THIS MUST BE AN Array DIMENSIONED APPROXIMATELY AS FOLLOWS.  FOR NPPR = 1, 4*(NPI+SQR(NPI/NPPR)).  FOR NPPR = 2, 4*(NPI+SQR(NPI/NPPR)).  FOR NPPR = 3, 4*(NPI+3*SQR(NPI/NPPR)).  FOR NPPR = 6, THIS MEANS APPROXIMATELY 4*NPI+1.6*SQR(NPI).  FOR NPPR = 1, 4*NPI+1.6*SQR(NPI).  FOR NPPR = 2, 7.6*SQR(NPI).  FOR NPPR = 3, 6.6*NPI+1.6*SQR(NPI).

WHEN NPPR IS INPUT AS ZERO THE USER MUST SPECIFY THE VALUES OF X TILDA AND Y TILDA AS FOLLOWS: WK(1) IS MIN(X1), WK(2), ..., WK(NXG+1) ARE THE VALUES OF X TILDA IN INCREASING ORDER, AND WK(NXG+2) IS MAX(X1).  WK(NXG+3) IS MIN(Y1), WK(NXG+4), ..., WK(NXG+NYG+3) ARE THE VALUES OF Y TILDA IN INCREASING ORDER.

NWK - INPUT AND OUTPUT.  ON ENTRY WITH MODE = 1, 2, OR 3 THIS MUST BE THE NUMBER OF LOCATIONS RESERVED FOR THE ARRAY WK.  ON OUTPUT IT IS SET TO THE ACTUAL NUMBER OF LOCATIONS REQUIRED FOR THE ARRAY.

FO - OUTPUT.  VALUES OF THE INTERPOLATION FUNCTION AT THE GRID POINTS INDICATED BY NXY, XO, YO, XC, YC.  FO IS ASSUMED TO BE DIMENSIONED (NXY, NXY) IN THE CALLING PROGRAM.
KER - OUTPUT RETURN INDICATOR.
= 0, NORMAL RETURN.
= -1, PROBLEM HAS NOT BEEN PREVIOUSLY SET UP (LOB 22
CALLED WITH MODE = 1, 2 OR 3).
= 1, ERROR RETURN FROM CLOB22, SINGULAR MATRIX IN THE
LEAST SQUARES FIT.
= 2, ERROR RETURN FROM CLOB22, SINGULAR MATRIX IN THE
OPTIMAL FIT.
= 3, ERROR RETURN FROM CLOB22, SOME RECTANGLE (I,J)
HAS MORE THAN THE ALLOWED NUMBER OF POINTS
ASSOCIATED WITH IT.
= 4, PREVIOUS ERROR RETURN FROM CLOB22 HAS NOT BEEN
CORRECTED.
= 5, IWK AND WK ARRAYS HAVE NOT BEEN DIMENSIONED
LARGE ENOUGH IN THE CALLING PROGRAM. REDIMEN-
SION IWK AND WK TO AT LEAST THE SIZE INDICATED
BY NIWK AND NWK, RESPECTIVELY.
= 6, MODE IS OUT OF RANGE.

DIMENSION XI(1:1), YI(1:1), FI(1:1), IWK(1:1), WK(1:1), XO(1:1), YO(1:1), FG(NXOLOB 1170
= 1:1)

DATA KERO/-1/
IF (MODE.LT.1.OR.MODE.GT.4) GO TO 220
KER = 0

ON INITIAL ENTRY MODE = 1, 2, OR 3, AND THE GRID LINES ARE SET UP, LOCAL INTERPOLATION POINTS ARE DETERMINED AND LOCAL APPROXIMATIONS ARE COMPUTED.

IF (MODE.EQ.4) GO TO 140
NWKIN = NWK
NIWKIN = NIWK
NXGWK = 1
NPWK = 3

IF (NPPR.LE.0) GO TO 100
NXG = SQRT(4.*FLOAT(NPI)/FLOAT(NPPR))-0.5
NYG = NXG
IWK(1) = NXG
IWK(2) = NYG
GOTO 120
100 NXG = IWK(1)
NYG = IWK(2)
120 IALWK = NXG*NYG+5
IABWK = IALWK
IF (MODE.NE.1) IABWK = IABWK+3*NXG*NYG
NYGWK = NXG+3
MPWK = NXG+NYG+4
IF (NPPR.GT.0) CALL GRID (XI,YI,NPI,NXG,WK(NXGWK),NYG,WK(NYGWK),WKLOB 1450
1(IALWK))

DETERMINE THE LOCAL INTERPOLATION POINTS FOR THE REGIONS.

CALL LOCLIP (NXG,WK(NXGWK),NYG,WK(NYGWK),NPI,XI,YI,IWK(NPWK),IWK(ML)
1PWK),WK(IALWK))

NWK = IABWK-1
IF (MODE.NE.2) NWK = NWK+IWK(MPWK-1)-1
NWK = NWG+NYG+2+IWK(MPWK-1)
IF (NIWK.GT.NWandin) GO TO 200
IF (NW.KT.NW) GO TO 200
MC = MODE

COMPUTE THE LOCAL APPROXIMATIONS.

CALL CLOB22 (MO,XI,YI,FI,NXG,WK(NXGWK),NYG,WK(NYGWK),IWK(NPWK),IWKLOB 1610
1(MPWK),WK(IALWK),WK(IABWK),IER)

KERO = IER
IF (IER.NE.0) GO TO 160
IF (KERO.NE.0) GO TO 180

COMPUTE THE FUNCTION VALUES ON THE DESIRED GRID OF POINTS.

CALL EVLB22 (MO,XI,YI,NXG,WK(NXGWK),NYG,WK(NYGWK),IWK(NPWK),IWK(MPL)
1WJ),WK(IALWK),WK(IABWK),NXO,XO,YC,YO,F0)

RETURN

ERROR RETURNS

160 KER = IER

RETURN

180 KER = 4
IF (KERO.LT.0) KER = -1
RETURN

200 KER = 5
RETURN

220 KER = 6
RETURN
END
SUBROUTINE GRID (X,Y,N,NX,XG,NY,YG,T)
C
C THIS SUBROUTINE PLACES A SET OF GRID LINES ON THE SET OF POINTS
C (X,Y), I=1,...,N. THIS IS DONE BY PLACING APPROXIMATELY EQUAL
C NUMBERS OF THEM WITHIN VERTICAL AND HORIZONTAL LINES.
C
C THE ARGUMENTS ARE AS FOLLOWS.
C
X,Y - INPUT. THE ARRAY OF (X,Y) POINTS.
N - INPUT. THE NUMBER OF (X,Y) POINTS.
NX - INPUT. THE DESIRED NUMBER OF VERTICAL GRID LINES.
XG - OUTPUT. THE COORDINATES OF THE VERTICAL GRID LINES.
NY - INPUT. THE DESIRED NUMBER OF HORIZONTAL GRID LINES.
YG - OUTPUT. THE COORDINATES OF THE HORIZONTAL GRID LINES.
T - WORK ARRAY OF DIMENSION AT LEAST N.
C
DIMENSION X(1), Y(1), XG(1), YG(1), T(1)
C
NMI = N-1
C
DO 100 I=1, N
  100 T(I) = X(I)
C
K = 1
FINC = FLOAT(N)/FLOAT(NX+1)
GO TO 220
C
120 DC 140 J=1, NX
  140 XG(J+1) = T(K)
C
XG(1) = T(1)
XG(NX+2) = T(N)
C
DO 160 I=1,N
  160 T(I) = Y(I)
C
K = 2
FINC = FLOAT(N)/FLOAT(NY+1)
GO TO 220
C
180 DO 200 J=1, NY
  200 YG(J+1) = T(K)
C
YG(1) = T(1)
YG(NY+2) = T(N)
RETURN
C
220 DC 260 I=1, NMI
C
IP1 = I + 1
DO 240 J = IP1, I(n)
IF (S(i) = T(i,j)) GO TO 240
END
C 240 CONTINUE
C 260 CONTINUE
GO TO (120, 180), K
SUBROUTINE LOCLIP (NXG, XG, NYG, YG, NPI, XI, YI, NP, MP, D)

C
C THIS SUBROUTINE DETERMINES THE LOCAL INTERPOLATION POINTS FOR THE
C GRID VERSION OF FRANKEL'S METHOD OF SURFACE INTERPOLATION.
C
C MINPTS POINTS ARE REQUIRED FOR EACH REGION. IF FEWER THAN MINPTS POINTS
C ARE FOUND IN THE REGION, THE NEXT CLOSEST POINTS (IN THE SUP NORM
C AFTER THE CURRENT RECTANGLE IS TRANSFORMED ONTO (0,1)) ARE USED. MINPTS IS
C SET TO 3, WHICH IS THE RECOMMENDED VALUE, ALTHOUGH IT MAY BE ALTERED.

C THE ARGUMENTS ARE AS FOLLOWS.

NXG - INPUT. NUMBER OF VERTICAL GRID LINES.
XG - INPUT. THE COORDINATES OF THE VERTICAL GRID LINES.
NYG - INPUT. NUMBER OF HORIZONTAL GRID LINES.
YG - INPUT. THE COORDINATES OF THE HORIZONTAL GRID LINES.
NPI - INPUT. THE NUMBER OF DATA POINTS.
XI - INPUT. THE DATA POINTS (XI,YI), I=1,...,NPI.
YI - INPUT. THE DATA POINTS (XI,YI), I=1,...,NPI.
NP - OUTPUT. AN ARRAY WHICH GIVES THE INITIAL SUBSCRIPT IN THE ARRAY MP AT WHICH THE SUBSCRIPTS FOR THE LOCAL INTERPOLATION POINTS ARE STORED.
MP - OUTPUT. AN ARRAY WHICH GIVES THE SUBSCRIPTS FOR THE LOCAL INTERPOLATION POINTS.
D - A WORK ARRAY OF DIMENSION AT LEAST NPI.

C DIMENSION XG(1), YG(1), XI(1), YI(1), NP(1), MP(1), D(1)
C DATA MINPTS/3/ IJ = 1
C NP(1) = 1 L = 0
C
C DO 200 J=1,NYG
C YGA = (YG(J)+YG(J+1))/2.
C DYG = YG(J+1)-YG(J)
C
C DO 180 I=1,NXG
C XGA = (XG(I)+XG(I+1))/2.
C DXG = XG(I+1)-XG(I)
C IJ = IJ+1

C DETERMINE THE POINTS IN THE (I,J)TH RECTANGLE.

C DC 120 NK=1,NPI
C IF (XI(NK).GT.XG(I+2).OR.XI(NK).LT.XG(I)) GC TO 100
C IF (YI(NK).GT.YG(J+2).OR.YI(NK).LT.YG(J)) GC TO 100

C
L = L+1
D(NK) = 1.E10
MP(L) = NK
GO TO 120
100 D(NK) = AMAX1(ABS(XI(NK)-XGA)/DXG,ABS(YI(NK)-YGA)/DYG)
120 CONTINUE
C
NP(IJ) = L+1
IF (NP(IJ)-NP(IJ-1).GE.MINPTS) GO TO 180
C
ADD THE CLOSEST POINTS IF THERE ARE LESS THAN MINPTS IN THE
RECTANGLE.
C
LM = MINPTS-(NP(IJ)-NP(IJ-1))
C
DO 160 II=1,LM
L = L+1
MP(L) = 1
DM = D(1)
C
DO 140 NK=2,NP
IF (D(NK).GE.DM) GO TO 140
DM = D(NK)
MP(L) = NK
140 CONTINUE
C
NK = MP(L)
160 D(NK) = 1.E10
C
NP(IJ) = L+1
180 CONTINUE
C
RETURN
END
SUBLROUTINE CLB22 (MODL, XI, YI, FI, NXG, XG, NYG, YG, NP, MP, AL, AB, IER)  CLO  10

THIS SUBROUTINE CONSTRUCTS THE LOCAL APPROXIMANTS FOR THE GRID
VERSION OF FRANKS'S METHOD. THE LOCAL APPROXIMATIONS MAY BE
EITHER OPTIMAL APPROXIMATIONS IN A CORNER 2,2, OR OPTIMAL
APPROXIMATIONS IN A CORNER 2,2 BOOLEAN SUM THE LEAST SQUARES
PLANE, OR FOR APPROXIMATION, RATHER THAN INTERPOLATION
THE LEAST SQUARES PLANE MAY BE SPECIFIED.
THE ARGUMENTS ARE AS FOLLOWS.

MODL - INPUT. SPECIFIES THE TYPE OF LOCAL APPROXIMATION
DESIRED.
= 1, USE THE OPTIMAL APPROXIMATION IN A CORNER 2,2.
= 2, USE THE LEAST SQUARES PLANE.
= 3, USE THE OPTIMAL APPROXIMATION IN A CORNER 2,2 BOOLEAN SUM THE LEAST SQUARES PLANE.

XI - INPUT. THE DATA POINTS (XI, YI, FI), I=1, NPI.

YI - INPUT. THE NUMBER OF VERTICAL GRID LINES.

NXG - INPUT. THE COORDINATES OF THE VERTICAL GRID LINES.

XG - INPUT. THE COORDINATES OF THE VERTICAL GRID LINES.

NYG - INPUT. THE NUMBER OF HORIZONTAL GRID LINES.

YG - INPUT. THE COORDINATES OF THE HORIZONTAL GRID LINES.

NP - INPUT. AN ARRAY WHICH GIVES THE INITIAL SUBSCRIPTS IN THE ARRAY MP AT WHICH THE SUBSCRIPTS FOR THE LOCAL INTERPOLATION POINTS ARE STORED.

MP - INPUT. AN ARRAY WHICH GIVES THE SUBSCRIPTS FOR THE LOCAL INTERPOLATION POINTS.

AL - INPUT. THE COEFFICIENTS FOR THE LEAST SQUARES FIT, WHEN MO DL = 2 OR 3.

AB - INPUT. THE COEFFICIENTS FOR THE OPTIMAL APPROXIMATION WHEN MO DL = 1 OR 3.

IER - OUTPUT. RETURN INDICATOR.
= 0, NORMAL RETURN.
= 1, SINGULAR MATRIX HAS BEEN DETECTED IN THE LEAST SQUARES FIT.
= 2, SINGULAR MATRIX HAS BEEN DETECTED IN THE OPTIMAL FIT, IN CASE OF A SINGULAR MATRIX, THE GRID VALUE (I,J) AND THE DATA POINTS ASSOCIATED WITH THAT RECTANGLE ARE PRINTED.
= 3, THE NUMBER OF POINTS ASSOCIATED WITH SOME RECTANGLE i,j IS BIGGER THAN PRESENTLY PERMITTED.
N*(NC*3)/2.
BECAUSE OF THE SHORT WORD LENGTH OF THE IBM 360/370 COMPUTERS CERTAIN VARIABLES ARE DECLARED AS DOUBLE PRECISION. THIS STATEMENT MAY BE SAFELY REPLACED WITH THE STATEMENT 'REAL K' WHEN THIS PROGRAM IS USED ON COMPUTERS WITH LONGER WORD LENGTHS.

DOUBLE PRECISION K, D1, D2, C
DIMENSION XI(I), YI(J), FI(I), NP(I), MP(I), AL(I), AB(I), XG(I), YG(J), C(230)
DATA NOUT, NC/6, 20/
IER = 0
IJ = 0
B = 1.

DC 260 J=1, NYG
DY = YG(J+2) - YG(J)
A = 1.

DC 240 I=1, NXG
DX = XG(I+2) - XG(I)
IJ = IJ+1
LEND = NP(IJ+1) - NP(IJ)
IF (LEND.GT.NC) GO TO 360
LNS = LENS*(LENS+1)/2
ALS = (IJ-1)*3
IF (MODL.EQ.1) GO TO 160

CALCULATE THE LEAST SQUARES PLANE

DC 100 LL=1, 9

100 C(LL) = 0.

C(1) = LENS

DO 120 I=1, LENS
MP = NP(IJ+1) - L - 1
KL = MP(MPSL)
XKL = XG(KL) - XG(I+1)
YKL = YG(KL) - YG(J+1)
C(2) = C(2) + XKL
C(3) = C(3) + XKL**2
C(4) = C(4) + YKL
C(5) = C(5) + YKL**2
C(6) = C(6) + YKL**2
C(7) = C(7) + FI(KL)
C(8) = C(8) + FI(KL)*XKL
12C C(9) = C(9) + FI(KL)*YKL
CALL LEQTIP (C,1,3,C(7),1,0,D1,D2,KER)
IF (KER.NE.0) GO TO 280

DO 140 LL=1,3
   IAL = IALS+LL
   AL(IAL) = C(LL+6)
   140

IF (MODL.EQ.2) GO TO 240

C   CALCULATE THE B CORNER 2,2 OPTIMAL APPROXIMATION
C
   KK = 0
   DO 200 LI=1,LEND
      MPI = NP(IJ)+LI-1
      KI = MP(MPI)
      XKI = (XI(KI)-XG(I))/DX
      YKI = (YI(KI)-YG(J))/DY
   200 CONTINUE

C   C(KK) = K(A,B,XKI,YKI,XKJ,YKJ)

C   LB = LBS+LI
   C(LB) = FII(KI)
   IF (MODL.EQ.1) GO TO 200
   XIK = XI(KI)-XG(I+1)
   YIK = YI(KI)-YG(J+1)
   C(LB) = C(LB)+(IALS+1)+ALIALS+2)*XIK+ALIALS+3)*YIK
200 CONTINUE

CALL LEQTIP (C,1,LEND,C(LBS+1),1,0,D1,D2,KER)
IF (KER.NE.0) GO TO 300

DO 220 LI=1,LEND
   IAB = NP(IJ)+LI-1
   LB = LBS+LI
   AEBIAB) = C(LB)
   220

A = 0.
240 CONTINUE
C
260 B = 0.
C RETURN
C ERROR RETURNS
C 280 IER = 1
C GC TO 320
C 300 IER = 2
C 320 WRITE (NOUT,1) I,J,IER
C MPST = MP(I,J)
C MPSE = MPST+LEND-1
C 340 DO 360 MPS=MPST,MPSE
C I = MP(MPS)
C WRITE (NOUT,2) I,XI(I),YI(I),FI(I)
C RETURN
C 360 IER = 3
C WRITE (NOUT,3) I,J,LEND,NC
C RETURN
C 1 FORMAT (46HOSINGULAR MATRIX DETECTED FOR GRID POINT I,J = 215,12H 1,
C IER = 12//40H THE DATA POINTS INVOLVED ARE AS FOLLOWS/)
C 1600
C 2 FORMAT (1X,15,3E15.6)
C 1680
C 3 FORMAT (15HOTHE RECTANGLE I3,1H, I3,5H HAS I3,27H POINTS ASSOCIACL
C 1690
C 1ED WITH IT /51H THE CURRENT LIMIT IMPOSED IN SUBROUTINE CLBC22 IS
C 1700
C 2,13//84H THIS MAY BE MODIFIED BY REDEFINING NC AND CHANGING THE DICL
C 1710
C 3MENSION OF C APPROPRIATELY)
C END
SUBROUTINE EVLB22 (MODL, XI, YI, NXG, YG, NO, MP, AL, AB,
      
1
      NXT, XO, NYO, YO, FO)

C THIS SUBROUTINE EVALUATES THE INTERPOLANT FOR THE GRID VERSION OF
   C FRANK'S METHOD. THE FUNCTION IS EVALUATED AT THE GRID OF POINTS
   C INDICATED BY NXX, XO, NYO, YO, AND THESE VALUES ARE RETURNED
   C IN THE ARRAY FO, WHICH IS ASSUMED TO BE DIMENSIONED (NXX,NYY).
   C
   C THE ARGUMENTS ARE AS FOLLOWS.
   C
   MODL- INPUT. SPECIFIES THE TYPE OF LOCAL APPROXIMATION USED.
      C = 1, USE THE OPTIMAL APPROXIMATION IN B CORNER 2,2.
      C = 2, USE THE LEAST SQUARES PLANE.
      C = 3, USE THE OPTIMAL APPROXIMATION IN B CORNER 2,2
      C BOOLEAN SUM THE LEAST SQUARES PLANE.
   XI - INPUT. THE DATA POINTS (XI, YI, FI), I = 1, ..., NPI.
   YI - INPUT. THE NUMBER OF VERTICAL GRID LINES.
   XG - INPUT. THE COORDINATES OF THE VERTICAL GRID LINES.
   YG - INPUT. THE NUMBER OF HORIZONTAL GRID LINES.
   NG - INPUT. THE COORDINATES OF THE HORIZONTAL GRID LINES.
   NP - INPUT. AN ARRAY WHICH GIVES THE INITIAL SUBSCRIPT IN
      C THE ARRAY MP AT WHICH THE SUBSCRIPTS FOR THE
      C LOCAL INTERPOLATION POINTS ARE STORED.
   MP - INPUT. AN ARRAY WHICH GIVES THE SUBSCRIPTS FOR THE
      C LOCAL INTERPOLATION POINTS.
   AL - INPUT. THE COEFFICIENTS FOR THE LEAST SQUARES PLANE.
   AB - INPUT. THE COEFFICIENTS FOR THE OPTIMAL APPROXIMATION
      C WHEN MODL = 1 OR 3.
   NXX - INPUT. THE NUMBER OF XG VALUES AT WHICH THE INTERPO-
      C LATION FUNCTION IS TO BE CALCULATED.
   XG - INPUT. THE VALUES OF X AT WHICH THE INTERPOLATION
      C FUNCTION IS TO BE CALCULATED.
   NYY - INPUT. THE NUMBER OF YG VALUES AT WHICH THE INTERPO-
      C LATION FUNCTION IS TO BE CALCULATED.
   YG - INPUT. THE VALUES OF Y AT WHICH THE INTERPOLATION
      C FUNCTION IS TO BE CALCULATED.
   FO - OUTPUT. VALUES OF THE INTERPOLATION FUNCTION AT THE
      C GRID POINTS INDICATED BY NXX, XO, NYY, YG.
      C FO IS ASSUMED TO BE DIMENSIONED (NXX,NYY) IN THE
      C CALLING PROGRAM.
   C
   DIMENSION XG(1), YG(1), XI(1), YI(1), NO(1), MP(1), AL(1),
   C 1AB(1), XO(1), YO(1), FO(NXX,1)
   C
   BECAUSE OF THE SHORT WORD LENGTH OF THE IBM 360/370 COMPUTERS
CERTAIN VARIABLES ARE DECLARED AS DOUBLE PRECISION. THIS STATE-
MENT MAY BE SAFELY REPLACED WITH THE STATEMENT "REAL K" WHEN
THIS PROGRAM IS USED ON COMPUTERS WITH LONGER WORD LENGTHS.

DOUBLE PRECISION K

ARITHMETIC STATEMENT FUNCTION FOR THE HERMITE QUINTIC.

H5(S) = 1. - S**3*( (6. * S - 15.) * S + 10.)
J = 1

DC 640 JC=1, NYO

DETERMINE THE LOCATION OF THE POINT YO IN TERMS OF THE SMALLEST
VALUE OF J SUCH THAT YO(JO) IS IN SOME RECTANGLE (I,J).

YY = YO(JO)
JJS = J+1
IF (YY.LT.YG(JJS)) JJS=1

DO 100 JJ=JJS,NYG
IF (YY.LT.YG(JJ+1)) GO TO 120
100 CONTINUE

J = NYG
GO TO 140

J = JJ-1
140 JD = 3
IF (JD.GE.1) GO TO 160
JC = 0
J = 1
GO TO 180
160 IF (JD.LT.NYG) GO TO 180
J = 6
180 DY = YG(J+2)-YG(J+1)
I = 1

DO 620 IC=1, NIO

DETERMINE THE LOCATION OF THE POINT XO IN TERMS OF THE SMALLEST
VALUE OF I SUCH THAT XO(I0) IS IN THE RECTANGLE (1,J).

IIS = I+1
XV = XO(I0)
IF (XV.LT.XG(IIS)) IIS=1

DC 200 II=IIS,NXG
IF (XV.LT.XG(II+1)) GO TO 220
200 CONTINUE
C
   I = NXG
   GO TO 240
220 I = I-1
240 IF (I.GE.1) GO TO 260
   IC = 1
   I = 1
   GO TO 280
260 IF (I.LT.NXG) GO TO 280
   ID = 3
280 DX = XG(I+2)-XG(I+1)
   KD = ID+JD
   A = 0.
   IF (I.EQ.1) A = 1.
   B = 0.
   IF (J.EQ.1) B = 1.
   GO TO (330,360,330,440,520,440,300,360,300), KD
   THIS IS FOR (XO(IO),YO(JO)) POINTS IN A SINGLE RECTANGLE (I,J)
300 FV = 0.
   IJ = (J-1)*NXG+I
   IAL = 3*IJ-2
   IF (MODL.EQ.2) GO TO 340
   LMAX = NP(IJ+1)-NP(IJ)
   DXA = XG(I+2)-XG(I)
   DY = YG(I+2)-YG(I)
   XVD = (XY-XG(I))/DXA
   YVD = (YV-YG(I))/DYA
   DC 320 L=1,LMAX
   MPS = NP(IJ)+L-1
   KI = MP(MPS)
   XK = (XI(KI)-XG(I))/DXA
   YK = (YI(KI)-YG(I))/DYA
   320 FV = FV+AB(MPS)*K(A,B,XK,YK,XVD,YVD)
   IF (MODL.NE.1) FV = FV+AL(I+1)+AL(I+1)*XV-XG(I+1)+AL(I+2)*YV-YG(J+1)
   GO TO 620
   THIS IS FOR XO(IO),YO(JO) POINTS WHICH ARE IN TWO RECTANGLES, (I,J) AND (I+1,J).
360 DY = YG(J+2)-YG(J)
   YVD = (YV-YG(J))/DYA
C

DC 420 IP=1,2
FC(IP) = 0.
IS = I+IP-1
IJ = (J-1)*NXG+IS
IAL = 3*I-J-2
IF (MODL.EQ.2) GO TO 400
DXA = XG(IS+2)-XG(IS)
XVD = (XV-XG(IS))/DXA
LMAX = NP(IJ+1)-NP(IJ)

C

DO 380 L=1,LMAX
MPS = NP(IJ)+L-1
KI = MP(MPS)
XKI = (XI(KI)-XG(IS))/DXA
YKI = (YI(KI)-YG(IJ))/DYA
380 FC(IP) = FC(IP)+AB(MPS)*K(A,B,XKI,YKI,XVD,YVD)

C

400 IF (MODL.NE.1) FC(IP)=FC(IP)+AL(IAL)+AL(IAL+1)*(XV-XG(IS+1))+AL(IAL+2)*(YV-YG(IJ+1))
A = 0.

420 CONTINUE

C

WI = H5((XV-XG(I+1))/DX)
FV = FC(1)*WI+(1.-WI)*FC(2)
GC TO 620

C

THIS IS FOR (XO(I0),YO(J0)) POINTS WHICH ARE IN TWG RECTANGLES,
(I,J) AND (I,J+1).

C

440 DXA = XG(I+2)-XG(I)
XVD = (XV-XG(I))/DXA

C

DG 500 JP=1,2
FC(JP) = 0.
JS = J+JP-1
IJ = (JS-1)*NXG+I
IAL = 3*I-J-2
IF (MODL.EQ.2) GO TO 480
DYA = YG(JS+2)-YG(JS)
YVD = (YV-YG(JS))/DYA
LMAX = NP(IJ+1)-NP(IJ)

C

DO 460 L=1,LMAX
MPS = NP(IJ)+L-1
KJ = MP(MPS)
XKJ = (XI(KJ)-XG(I))/DXA
YKJ = (YI(KJ)-YG(JS))/DYA

EVL 1450
EVL 1460
EVL 1470
EVL 1480
EVL 1490
EVL 1500
EVL 1510
EVL 1520
EVL 1530
EVL 1540
EVL 1550
EVL 1560
EVL 1570
EVL 1580
EVL 1590
EVL 1600
EVL 1610
EVL 1620
EVL 1630
EVL 1640
EVL 1650
EVL 1660
EVL 1670
EVL 1680
EVL 1690
EVL 1700
EVL 1710
EVL 1720
EVL 1730
EVL 1740
EVL 1750
EVL 1760
EVL 1770
EVL 1780
EVL 1790
EVL 1800
EVL 1810
EVL 1820
EVL 1830
EVL 1840
EVL 1850
EVL 1860
EVL 1870
EVL 1880
EVL 1890
EVL 1900
EVL 1910
EVL 1920
460 FC(JP) = FC(JP)+AB(MPS)*K(A,B,XKJ,YKJ,XVD,YVD)

480 IF (MODL NE 1) FC(JP)=FC(JP)+AL(IAL)+AL(IAL+1)*(XV-XG(I+1))+AL(IAL+2)*(YV-YG(JS+1))
   B = 0.

500 CONTINUE

C
UJ = HS((YV-YG(J+1))/DY)
FV = FC(I)*UJ+(1.-UJ)*FC(2)
GC TO 620

C
THIS IS FOR (XO(I),YO(J)) POINTS WHICH ARE IN FOUR RECTANGLES, (I,J), (I+1,J), (I,J+1), AND (I+1,J+1).

C
520 KFC =

C
DQ 600 JP=1,2
JS = J+JP-1
DYA = YG(JS+2)-YG(JS)
YVD = (YV-YG(JS))/DYA
A = 1.

C
DQ 580 IP=1,2
IS = I+IP-1
IJ = (JS-1)*NXG+IS
IAL = 3*I+2
KFC = KFC+1
FC(KFC) = 0.
IF (MODL EQ 2) GO TO 56)
IF (IS-GT 1) A = 0.
DXA = XG(IS+2)-XG(IS)
XVD = (XV-XG(IS))/DXA
LMAX = NP(I+1)-NP(IJ)

C
DC 540 L=1,LMAX
MPS = NP(IJ)+L-1
KI = MP(MPS)
XKI = (XI(KI)-XG(IS))/DXA
YKI = (YI(KI)-YG(JS))/DYA
540 FC(KFC) = FC(KFC)+AB(MPS)*K(A,B,XKJ,YKJ,XVD,YVD)

C
560 IF (MODL NE 1) FC(KFC)=FC(KFC)+AL(IAL)+AL(IAL+1)*(XV-XG(IS+1))+AL(IAL+2)*(YV-YG(JS+1))

580 CONTINUE

C
B = 0.

600 CONTINUE

C
WI = H5{(XV-XG{I+1})/DX}
UJ = H5{(YV-YG{J+1})/DY}
FV = WI*UJ*FC(1)+(1-UJ)*FC(2)+(1-WI)*UJ*FC(3)+(1-WI)*FC(4)
620 FO(IO,JO) = FV
C 640 CONTINUE
C
RETURN
END
FUNCTION K (A,B,U,V,S,T)

THIS FUNCTION EVALUATES THE REPRESENTER FOR THE POINT EVALUATION
(AT (U,V)) FUNCTIONAL FOR THE SARD CORNER SPACE B CORNER 2,2.

THE ARGUMENTS ARE AS FOLLOWS.

A,B - INPUT. THE BASE POINT (A,B) FOR THE SARD SPACE B CORNER 2,2.

U,V - INPUT. THE LINEAR FUNCTIONAL IS EVALUATION AT (U,V).

S,T - INPUT. THE REPRESENTER IS EVALUATED AT (S,T).

BECAUSE OF THE SHORT WORD LENGTH OF THE IBM 360/370 COMPUTERS CERTAIN VARIABLES ARE DECLARED AS DOUBLE PRECISION. THIS STATEMENT MAY BE SAFELY REPLACED WITH THE STATEMENT REAL K WHEN THIS PROGRAM IS USED ON COMPUTERS WITH LONGER WORD LENGTHS.

DOUBLE PRECISION K,SMA,UMA,USMA,X,TRMS,GP1,GP2

SS = S
UL = U
AA = A

100 SMA = SS-AA
UMA = UU-AA
USMA = SMA*UMA
TRMS = 0.
IF (USMA.LE.0.) GO TO 160
IF (UMA.GE.0.) GO TO 120
UMA = -UMA
SMA = -SMA

120 IF (SMA.LE.UMA) GO TO 140
X = SMA
UMA = X
TRMS = SMA/2.*(USMA-SMA**2/3.)

140 GP2 = 1.+USMA+TRMS
GO TO (180,230), KVAR

180 GP1 = GP2
SS = T
UL = V
AA = B
KVAR = 2
GO TO 100

200 K = GP1*GP2
RETURN
END
<table>
<thead>
<tr>
<th>Function Values</th>
<th>Deviations (These values represent roundoff error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.557540</td>
<td>-0.000015 -0.000009 0.000039</td>
</tr>
<tr>
<td>1.686603</td>
<td>0.000042 0.000020 0.000038</td>
</tr>
<tr>
<td>0.737488</td>
<td>0.000052 0.000050 0.000009</td>
</tr>
</tbody>
</table>

The values of MODE, KER, NIWK, and NWK are 1 0 79 77

<table>
<thead>
<tr>
<th>Function Values</th>
<th>Deviations (These values represent roundoff error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.658678</td>
<td>-0.000003 0.000001 0.000001</td>
</tr>
<tr>
<td>1.754473</td>
<td>0.000002 0.000000 -0.000000</td>
</tr>
<tr>
<td>0.933759</td>
<td>0.000001 3.333339</td>
</tr>
</tbody>
</table>

The values of MODE, KER, NIWK, and NWK are 2 0 79 37

<table>
<thead>
<tr>
<th>Function Values</th>
<th>Deviations (These values represent roundoff error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.572663</td>
<td>-0.000018 -0.000019 0.000026</td>
</tr>
<tr>
<td>1.695004</td>
<td>0.000033 -0.000033 0.000002</td>
</tr>
<tr>
<td>0.736699</td>
<td>0.333339 -0.000066 -0.000017</td>
</tr>
</tbody>
</table>

The values of MODE, KER, NIWK, and NWK are 3 0 79 104

<table>
<thead>
<tr>
<th>Function Values</th>
<th>Deviations (These values represent roundoff error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.576339</td>
<td>0.000092 -0.000033 0.000026</td>
</tr>
<tr>
<td>1.626697</td>
<td>-0.000019 -0.000033 0.000023</td>
</tr>
<tr>
<td>0.711982</td>
<td>-0.000096 -0.000066 0.000017</td>
</tr>
</tbody>
</table>

The values of MODE, KER, NIWK, and NWK are 3 0 58 76
REFERENCES


## DISTRIBUTION LIST

<table>
<thead>
<tr>
<th>Address</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defense Documentation Center, Cameron Station, Alexandria, VA 22314</td>
<td>2</td>
</tr>
<tr>
<td>Dudley Knox Library, Naval Postgraduate School, Monterey, CA 93940</td>
<td>2</td>
</tr>
<tr>
<td>Dean of Research, Naval Postgraduate School, Monterey, CA 93940</td>
<td>2</td>
</tr>
<tr>
<td>Department of Mathematics, C. O. Wilde, Chairman, 1</td>
<td>1</td>
</tr>
<tr>
<td>Professor C. Comstock</td>
<td>1</td>
</tr>
<tr>
<td>Professor E. Faulkner</td>
<td>1</td>
</tr>
<tr>
<td>Professor A. Schoenstadt</td>
<td>1</td>
</tr>
<tr>
<td>Professor R. Franke</td>
<td>5</td>
</tr>
<tr>
<td>Naval Postgraduate School, Monterey, CA 93940</td>
<td></td>
</tr>
<tr>
<td>Dr. Richard Lau, Office of Naval Research, Office of Naval Research</td>
<td>1</td>
</tr>
<tr>
<td>1030 East Green St., Pasadena, CA 91106</td>
<td></td>
</tr>
<tr>
<td>Professor R. E. Barnhill, Department of Mathematics, University of Utah</td>
<td>1</td>
</tr>
<tr>
<td>Salt Lake City, UT 84112</td>
<td></td>
</tr>
<tr>
<td>Professor G. M. Nielson, Department of Mathematics, Arizona State</td>
<td>1</td>
</tr>
<tr>
<td>University, Tempe, AZ 85281</td>
<td></td>
</tr>
<tr>
<td>Chief of Naval Research, ATT: Mathematics Program, Arlington, VA 22217</td>
<td>2</td>
</tr>
<tr>
<td>Rosemary E. Chang, Sandia Laboratories, Applied Mathematics, Division 2</td>
<td>1</td>
</tr>
<tr>
<td>8235, Livermore, CA 94550</td>
<td></td>
</tr>
<tr>
<td>Hiroshi Akima, Office of Telecommunications, Department of Commerce</td>
<td>1</td>
</tr>
<tr>
<td>Boulder, CO 80302</td>
<td></td>
</tr>
</tbody>
</table>