INTERPOLATION AND QUASI-INTERPOLATION USING TRIANGULAR SPLINES

A thesis submitted to
Lakehead University
In Partial Fulfillment of the Requirements
for the Degree of
Master of Science

by
Athanase (Tom) C. Tsekouras

1973

D-2738

THESES
M.Sc.
1973
.T88
c.1

199886

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor P. O. Frederickson, for his advice and encouragement during the preparation of this paper.

ABSTRACT

This thesis is a study of the applications of Regular Triangular Splines to the bivariate approximation problem. Our primary interest will concern the numerical approximation of functions in two variables, and implementation of the mathematical theories as working programs on high speed computers.

Chapter I contains background information on the history and development of bivariate (multivariate) approximation theory. Chapter II contains the statement of the specific problems we are concerned with in this thesis.

In Chapter III we present the major ideas related to triangular splines. A convenient coordinate system called barycentric coordinates is introduced as well as the basic element of the space $S_h^{4,2}$ of quartic splines.

In Chapter IV we consider the problem of transforming the theory of triangular spline Interpolation, as developed by Professor P. Frederickson, to a working computer program. An algorithm is given and numerical results of its use when applied to certain functions.

Chapter V contains a Quasi-interpolation scheme using triangular splines, an error analysis of it and numerical results of its implementation. Treating this scheme we were confronted with the extrapolation problem. Consequently, we discuss certain extrapolation schemes.

Finally, Chapter VI contains some remarks on interpolation and quasi-interpolation and another algorithm for interpolation is presented.

TABLE OF CONTENTS

CHAPTER I

HISTORICAL SURVEY

1.1 The Bivariate Interpolation Problem

The problem of the numerical approximation of functions in two variables (or more) is relatively new compared to the corresponding one for one variable. Except for certain direct generalizations of the latter case it was only during the past two or three decades that a considerable degree of development took place. There are several reasons why the development of this problem was delayed. In our opinion the following two are the major ones.

A. Size of the problem. The size of the problem increases for two dimensions essentially from the computational point of view. Roughly we would say that the size of the bivariate problem is n^2 where the corresponding one dimensional problem is of size n. If, for example, we think of $f(x)$ as a smooth and well behaved function when it is adequately approximated by a polynomial of degree 99, with 100 coefficients, then we must consider $f(x,y)$ as smooth and well behaved when it is approximated by a polynomial of degree 99 in each variable, which requires 10,000 coefficients. Obviously, in the latter case we are confronted with a tremendous problem as far as handling 10,000 coefficients is concerned. Besides, the determination of these coefficients requires the inversion of a 100 by 100 matrix in the line case, but a 10,000 by 10,000 matrix in the two dimensional case. It is only recently, with the development of high-speed computers, with their huge storage capacity, that it has been feasible for numerical analysts to attempt these problems.

B. Variety of regions. In the univariate case we have, when practical algorithms are under considerations, exactly one region to consider, the finite interval. It has exactly two boundary points. In the bivariate case, though, we have a great variety of regions and boundaries. The contrast is even stronger when we consider the points at which information about the function f is given. In the one dimensional case these points divide the interval into subintervals, for they are simply ordered. In the two dimensional case a wide variety of patterns might be considered, and it is not obvious which is optimal. The problem of construction of approximating formulas, as well as the problem of finding error estimates or convenient expressions for remainders of approximating formulas, becomes much more difficult with the variety of regions.

In the next section we establish some notation to be used in the rest of this paper. Section 1.3 contains a selection of classical bivariate approximation methods, most of which are extensions of the corresponding one dimensional methods. Sections 1.4 through 1.6 discuss briefly several approximation methods developed during the past decade or so, almost exclusively with the use of piecewise polynomial functions (splines).

1.2 Notation

We will be concerned with the approximation of a continuous real valued function defined on a bounded domain Ω in R^2 with boundary $\partial\Omega$. For simplicity we will assume that Ω is a subset of the square $[0,1] \times [0,1]$. If $\alpha = (k,\ell)$ is a 2-tuple with non-negative integer components we define the differential operator

$$D^{\alpha} = D^{(k,\ell)} = \frac{\partial^k}{\partial x^k} \frac{\partial^{\ell}}{\partial y^{\ell}} \qquad (1.2.1)$$

with $|\alpha| = k + \ell$.

We will denote by $C^{(m,n)}(\Omega)$ the space of all functions f such that $D^{(k,\ell)}f$ exists and is continuous for all $0 \le k \le m$, $0 \le \ell \le n$; and by $C^s(\Omega)$ the space of all functions f such that $D^{\alpha}f$ exists and is continuous for all α, $|\alpha| \le s$, on the set Ω. Denote by $P_{m,n}(\Omega)$ the space of polynomials of degree not exceeding m in x and n in y; and by $P_k(\Omega)$ the space of polynomials of degree k in x,y. Thus $f \in P_k(\Omega)$ if and only if $f \in C^{k+1}(\Omega)$ and $D^{\alpha}f = 0$ if $|\alpha| = k + 1$.

The notion of approximation becomes precise only when we establish a measure of deviation of the approximating function from the given function. This measure of deviation is required to satisfy the properties of norm. We assume that the reader is familiar with the basic theory of normed linear spaces, finitely or infinitely dimensioned. We will denote by $L_p(\Omega)$, $p \ge 1$ the completion of the space $C(\Omega)$ under the norm $||\cdot||_p$ defined as follows

$$||f||_p = (\iint_\Omega (f(x,y))^p)^{1/p}, \qquad 1 \le p < \infty \tag{1.2.2}$$

and

$$||f||_\infty = \underset{(x,y)\,\varepsilon\,\Omega}{\ell.u.b}|f(x,y)| \tag{1.2.3}$$

Use will also be made of the Sobolev space $W_p^s(\Omega)$. This is the completion of the space $C^s(\Omega)$ under the norm $||\cdot||_{p,s}$ defined by

$$||f||_{p,s} = \sum_{|\alpha|\le s} ||D^\alpha f||_p, \qquad 1 \le p \le \infty \tag{1.2.4}$$

1.3 Classical Bivariate Approximation Methods

The most familiar example of bivariate approximation is of course the first $(m+1) \times (m+2)/2$ terms in the Taylor expansion of $f(x,y)$ about (x_0,y_0). That is, for every function f defined on an open and convex set Ω with $f\ \varepsilon\ C^{m+1}(\Omega)$ and (x_0,y_0), $(x,y)\ \varepsilon\ \Omega$, we have

$$f(x,y) = f(x_0,y_0) + \sum_{1\le k+\ell\le m} D^{(k,\ell)} f(x_0,y_0) \frac{(x-x_0)^k}{k!} \frac{(y-y_0)^\ell}{\ell!} + R_m \tag{1.3.1}$$

where R_m is the remainder of the approximation and is bounded by

$$\sup_{(\xi,\eta)} \sum_{k+\ell=m+1} \left|D^{(k,\ell)} f(\xi,\eta) \frac{(x-x_0)^k}{k!} \frac{(y-y_0)^\ell}{\ell!}\right| \tag{1.3.2}$$

with (ξ,η) situated on the line joining (x_0,y_0) to (x,y), [26, 16.5.1]. However, there are certain disadvantages to this method. It is hard, for example, to obtain satisfactory error estimates for the remainder expression. If f has partial derivatives of all orders on Ω then the Taylor's series

$$f(x_0,y_0) + \sum_{1\leq k+\ell} D^{(k,\ell)} f(x_0,y_0) \frac{(x-x_0)^k}{k!} \frac{(y-y_0)^\ell}{\ell!} \qquad (1.3.1)'$$

may fail to converge for any (x,y) other than (x_0,y_0). Even if the series does converge for a particular (x,y), then it still may be that the series (1.3.1)' does not converge to the value $f(x,y)$. A well known example of such a function is

$$f(x,y) = \exp(-1/(x^2+y^2))$$

Partial derivatives of all orders are continuous everywhere, but at the point $(0,0)$ all vanish.

In contrast to the Taylor theorem, the well known Weierstrass approximation theorem assures us we can approximate uniformly functions which are merely continuous. This theorem holds for functions in n variables [29, theorem 8]. We will state it for the bivariate case.

Theorem (Weierstrass). *If* Ω *is closed and bounded and* $f \in C(\Omega)$, *then for every* $\varepsilon > 0$ *there exist* $m = m(\varepsilon)$, $n = n(\varepsilon)$ *and a polynomial* $p \in P_{m,n}$ *such that*

$$||f-p||_\infty < \varepsilon \qquad (1.3.3)$$

If Ω is the space $[0,1] \times [0,1]$, then the generalized Bernstein polynomials are given by

$$P_{m,n}(x,y) = \sum_{k=0}^{m} \sum_{\ell=0}^{n} \binom{m}{k}\binom{n}{\ell} f\left(\frac{k}{m}, \frac{\ell}{n}\right) x^k (1-x)^{m-k} y^\ell (1-y)^{n-\ell} \qquad (1.3.4)$$

The proof is almost the same as in the one dimensional case [30, Theorem 2.1].

Stancu [35, Section 8] presents a simple expression for the remainder of the approximation of f by Bernstein's polynomials (1.3.4), when $D^{(2,0)}f$, $D^{(0,2)}f$ and $D^{(2,2)}f$ are continuous on Ω. He proves that the remainder in this case is given by

$$R(f) = -\frac{x(1-x)}{2m} D^{(2,0)} f(x,\eta) - \frac{y(1-y)}{2n} D^{(0,2)} f(\xi,y)$$

$$- \frac{x(1-x)y(1-y)}{4mn} D^{(2,2)} f(\xi,\eta) \qquad (1.3.5)$$

for some point $(\xi,\eta) \in \Omega$.

The problem of interpolation appears to present a larger variety of treatment. The most commonly used methods for deriving polynomial interpolation formulas are those of extending to two variables the formulas of Newton, Lagrange, Bessel, Everett, etc., in which forward, central and backward differences are used.

Let us, for example, generate the Newton's interpolation formula with divided differences when the data points are arranged at the intersections of a rectangular mesh, $\{(x_i,y_j), i = 0, 1, \ldots, m;$

$j = 0, 1, \ldots, n\}$ [36, Section 19]. The divided difference with respect to x, formed with the arguments $x_0, x_1, \ldots, x_k$, and with respect to y, formed with the arguments $y_0, y_1, \ldots, y_\ell$ is of the order $k + \ell$ and we denote it by

$$f_{k\ell} = f(x_0, x_1, \ldots, x_k; y_0, y_1, \ldots, y_\ell).$$

We also write

$$X_k(x) = (x-x_0)\cdots(x-x_{k-1}); \; Y_\ell(y) = (y-y_0)\cdots(y-y_{\ell-1})$$

with $X_0 = Y_0 = 1$.

For fixed y applying Newton's one dimensional formula we get

$$f(x,y) = \sum_{k=0}^{m} X_k(x) f(x_0,x_1,\ldots,x_k;y) + X_{m+1}(x) f(x,x_0,\ldots,x_\eta;y)$$

and by the same formula

$$f(x_0,x_1,\ldots,x_k;y) = \sum_{\ell=0}^{\eta} Y_\ell(y) f_{k\ell} + Y_{\eta+1} f(x_0,x_1,\ldots,x_m;y,y_0,\ldots y_\eta)$$

so that

$$f(x,y) = p(x,y) + R \qquad (1.3.6)$$

where $p(x,y) = \sum_{k=0}^{m} \sum_{\ell=0}^{n} X_k(x) Y_\ell(y) f_{k\ell}$ is the interpolating polynomial and R the remainder, given

$$R = X_{m+1}(x) f(x,x_0,\ldots,x_m; y) + Y_{n+1}(y) f(x; y_0,y_1,\ldots,y_\eta,y)$$

$$- X_{m+1}(x) Y_{n+1}(y) f(x,x_0,\ldots,x_m; y,y_0,\ldots,y_n) \qquad (1.3.7)$$

It is easy to see that $p(x,y)$ interpolates $f(x,y)$. For if (x_i,y_j), $0 \leq i \leq m$, $0 \leq j \leq n$, is one of the data points then $R = 0$ since $X_{m+1}(x_i) = Y_{n+1}(y_j) = 0$. We also have

$$\begin{aligned} p(x_i,y_j) &= \sum_{k=0}^{m} \sum_{\ell=0}^{n} X_k(x_i)Y_\ell(y_j)f_{k\ell} \\ &= \sum_{\ell=0}^{n} Y_\ell(y_j) \sum_{k=0}^{m} X_k(x_i)f_{k\ell} \\ &= \sum_{\ell=0}^{n} Y_\ell(y_j)f(x_i;\, y_0,y_1,\ldots,y_\ell) \\ &= f(x_i,y_j). \end{aligned}$$

Similarly one can obtain the other kinds of polynomial interpolating formulas.

Approximation of periodic functions in two variables is also an easy extension. The idea is to approximate f by trigonometric polynomials in each variable. If $f(x,y)$ is a periodic continuous function of period 2π in both variables, then, given any $\varepsilon > 0$, there exist $m = n(\varepsilon)$, $n = n(\varepsilon)$ and a trigonometric polynomial $T_{m,n}(x,y)$ such that

$$||f(x,y) - T_{m,n}(x,y)||_\infty < \varepsilon \qquad (1.3.8)$$

[28, p. 87].

The theoretical development of least-squares techniques encounters no difficulty when extended to functions of two or more variables.

Theorems related to such techniques have been proved in even more general spaces [16, Chapter I].

Let $f(x,y)$ be a function defined on $[-1,1] \times [-1,1]$. We consider [16] sets of functions $\{g_i(x),\ i = 1, 2, \ldots, m\}$, $\{h_j(y),\ j = 1, 2, \ldots, n\}$ and tensor products of them $\{g_i h_j,\ 1 \le i \le m,\ 1 \le j \le n\}$. It is easily proved [31] that if $\{g_i\}$ and $\{h_j\}$ are orthonormal sets of functions so are $\{g_i h_j\}$. We now consider an approximation of f by the linear form

$$L(f) = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} g_i(x) h_j(y) \tag{1.3.9}$$

with the functionals a_{ij} to be determined so that $||f-L(f)||_2$ is minimized. Although this extension is theoretically easy, difficulties arise when one attempts to compute the coefficients a_{ij} in order to evaluate $L(f)$.

In methods discussed so far, the extension of the approximation theory from the one dimensional case to the bivariate one presents no major theoretical difficulties. However, the generalization of the Tchebycheff theory to functions of more than one variable fails to follow this pattern. The basic problem is that there are no Tchebycheff sets of functions of more than one variable. Hence, by the absence of these the bivariate Tchebycheff theory is considerably changed.

The existence of best Tchebycheff approximations is easy to prove. However, these may not be uniquely determined.

More on multivariate Tchebycheff approximation can be found in the work of Rice [31], [32].

1.4 Piecewise Polynomial Approximation Methods

During the past two decades many numerical analysts interested in approximation problems were attracted by spline functions. A spline [24] is a mechanical device, used by draftsmen to draw a smooth curve, consisting of a strip or rod of some flexible material constrained to pass through specified points. The mathematical spline is a piecewise polynomial of degree n with continuous $(n-1)$st derivative. We are all familiar with linear splines (i.e. broken line joining tabular points) which are inefficient for approximation problems. Cubic splines $(n=3)$ are closely related to the draftsman's spline and are the simplest ones for which good results have been obtained.

Consider the partitioned interval $[x_0,x_N]$

$$x_0 < x_1 < \dots < x_N$$

with the following set of ordinates prescribed

$$y_0, y_1, \dots, y_N.$$

We seek a function $C(x) \in C^2[x_0,x_N]$ such that

$$C(x_i) = y_i, \quad i = 0, 1, \dots, N$$

and C coincides with a cubic in each subinterval $[x_{i-1},x_i]$. We write $C''(x_i) = M_i$. Then from the linearity of the second derivative on $[x_{i-1},x_i]$, $i = 1, 2, \ldots, N$ we get

$$C''(x) = M_{j-1} \frac{x_i - x}{h_i} + M_i \frac{x - x_{i-1}}{h_i} \tag{1.4.1}$$

where $h_i = x_i - x_{i-1}$.

Integrating twice (1.4.1), we obtain

$$C(x) = M_{i-1} \frac{(x_i - x)^3}{6h_i} + M_i \frac{(x - x_{i-1})^3}{6h_i} + (y_{i-1} - \frac{M_{i-1}h_i^2}{6}) \frac{x_i - x}{h_i}$$
$$+ (y_i - \frac{M_i h_i^2}{6}) \frac{x - x_{i-1}}{h_i} \tag{1.4.2}$$

This is the cubic spline on the interval $[x_{i-1},x_i]$. The quantities M_i, $i = 0, 1, \ldots, N$ can be easily evaluated [see 1, Chapter 21] from continuity conditions of C, C', and C'' on the mesh points, and additional end conditions.

Cubic splines have been extensively studied during the past two decades as well as generalizations of them. Most of this material has been standardized [1], [24] and [33]. Shumaker's recent bibliography on applications, generalizations or extensions of cubic splines, is a list of more than 800 references.

The theory of splines has also been extended to several dimensions. The extension to bicubic splines was initiated by Birkhoff and Garabedian [5] and elaborated upon by deBoor [9] and Ahlberg, Nilson, Walsh [1]. If f is a function given together with its normal derivative $\frac{\partial f}{\partial n}$ on the boundary of a rectangular domain in the (x,y)-plane and values of f are also given on a rectangular grid of points (x_i, y_j), $1 \le i \le m$, $1 \le j \le n$, the authors in [5] succeed in fitting a smooth and analytically simple surface inside each rectangle $R_{ij}: x_{i-1} < x < x_i$, $y_{j-1} < y < y_j$. The edges of the rectangles R_{ij} are pieces of one dimensional cubic splines $f(x_i,y)$ and $f(x,y_j)$ passing through the given points, along the coordinate lines $x = x_i$, $y = y_j$.

However, the first truly successful extension was made by deBoor [9]. He proved both the existence and uniqueness of certain bicubic splines of interpolation. These are tensor products of one dimensional cubic splines $\phi_m(x)$, $\psi_n(y)$.

$$\sum_{m=0}^{I} \sum_{n=0}^{J} \beta_{mn}\phi_m(x)\psi_n(y) \tag{1.4.3}$$

Now given the values

$$\begin{aligned}
f_{ij} &= f(x_i,y_j) & i &= 0, 1, \ldots, I; \quad j = 0, 1, \ldots, J, \\
p_{ij} &= D^{(1,0)}f(x_iy_j) & i &= 0, I; \quad j = 0, 1, \ldots, J, \\
g_{ij} &= D^{(0,1)}f(x_iy_j) & i &= 0, 1, \ldots, I; \quad j = 0, J, \\
s_{ij} &= D^{(1,1)}f(x_iy_j) & i &= 0, I; \quad j = 0, J.
\end{aligned} \tag{1.4.4}$$

the author proves (Theorem 2) that there exists exactly one bicubic spline $f(x,y)$ of the form (1.4.3) which satisfies (1.4.4).

The bicubic spline interpolation scheme of deBoor, described above, has become the classic scheme for rectangular regions. Carlson and Hall in a series of three recent papers [11], [12] and [13] treat the same problem in more general rectangular polygons, right triangles and L-shaped domains respectively. They also give sharp error bounds, the best known so far. The main result proved in the first of them is the following: Let (R,π) be a uniformly partitioned rectangle such that each mesh line contains an even number of mesh points. Assume that the values (1.4.4) are given. Then there exists a function $V_f \varepsilon S_1^2(R,\pi)$, the space of bicubic splines over (R,π), such that

$$||D^{(k,\ell)}(V_f - f)||_\infty = O(h^{4-(k+\ell)}), \quad 0 \le k, \quad \ell \le 2 \tag{1.4.3}$$

and

$$D^{(r,s)}V_f = D^{(r,s)}f$$

at the respective values and points specified in (1.4.4).

Analogous results are proved in [12] and [13].

1.5 Blending-function Approximation Techniques

Another family of methods for bivariate interpolation is that treated in [6], [7], [20], [21], [22], [23], etc. This is interpolation on curve networks; i.e. these methods interpolate to functions given along mesh lines. They are usually referred to as "blending-function" techniques. The first work in this direction was done by Coons. Forest elaborated on it and Gordon extended and generalized it. Coon's "patch surface" methods solve the following problem: Given the values of the $p - 1$ normal derivatives, for any fixed integer p, along the four edges of the unit square construct a surface which interpolates all of these conditions. Birkhoff and Gordon [7] characterized these surfaces by proving the following result: The Coon's surface $V(x,y)$ which interpolates $f(x,y)$ and the first $p - 1$ normal derivatives $\frac{\partial^{p-1} f}{\partial_\eta^{p-1}}$ on the boundary of $[0,1] \times [0,1]$ is the unique solution to the generalized "draftsman's" equation

$$D^{(2p,2p)} f(x,y) = 0 \qquad (1.5.1)$$

It is understood that Coon's blending-techniques are local in the sense that interpolation on larger networks is obtained by joining surface patches.

In contrast to these, Gordon [23] presents blending function schemes which are global. The blending functions in this case are two sets of functions

$$\{\phi_i(x)\}_{i=0}^{M} \quad \text{and} \quad \{\psi_j(y)\}_{j=0}^{N}$$

which play the role of blending together the various curves comprising the network. The following main result (Theorem 3.1) is proved in [23].

Let $F(x,y)$ be an arbitrary bivariate function and $\{\phi_i(x)\}_{i=0}^{M}$, $\{\psi_j(y)\}_{j=0}^{N}$ be two families of blending functions such that

$$\phi_i(x_k) = \begin{cases} 0 & \text{if } k \neq i \\ 1 & \text{if } k = i \end{cases}$$

$$\psi_j(y_\ell) = \begin{cases} 0 & \text{if } \ell \neq j \\ 1 & \text{if } \ell = j \end{cases}$$

Then the function $V(x,y)$ given by

$$V(x,y) = \sum_{i=0}^{M} f(x_i,y)\phi_i(x) + \sum_{j=0}^{M} f(x,y_i)\psi_j(y) - \sum_{i=0}^{M}\sum_{j=0}^{N} f(x_i,y_j)\phi_i(x)\psi_j(y) \tag{1.5.2}$$

interpolates $f(x,y)$ along the lines $x = x_i$ $(i = 0, 1, \ldots, M)$ and $y = y_j$ $(j = 0, 1, \ldots, N)$.

This method has been applied by General Motors for the construction of dies for auto bodies.

Barnhill, Birkhoff and Gordon [6] give another blending scheme treating the interpolation problem of constructing a smooth function which assumes given boundary values and derivatives on the edges of a triangle T. This scheme is built up from projectors P_i which interpolate in the x, y, z - space between parallels to the ith side of T. Additionally for any values $p = 1, 2, \ldots$ the interpolating function $f(p_i)f$ interpolates to $f \varepsilon C^{3p}(T)$ and to the first $p - 1$ derivatives of f, and if $f \varepsilon C^{4p}(T)$ the error of approximation is of order $O(h^{3p})$, when h is the diameter of T.

1.6 Other Bivariate Approximation Methods

In this section we consider interpolation formulae which are particularly useful in the variational solution of boundary value problems.

Birkhoff, Schultz and Varga [8] present a general theory of Hermite interpolation on a rectangle. They obtain sharp errors for derivatives of interpolation errors for functions in one variable and extend these results to bivariate functions defined on rectangular domains R. The interpolated functions f belong to the space

$$S^{p,r}(R) = \{f(x,y) \,|\, D^{(p-i,i)}f \in L_r(R),\ \ 0 \le i \le p;\ D^{(i,j)}f \in C^0(R),$$
$$0 \le i + j \le p\}$$

If $P^{\rho}_{2m-1,2m-1}(x,y)$ is the piecewise Hermite Interpolate of $f(x,y) \in S^{p,r}$, $p \ge 2m$ and $\rho = \pi \times \pi'$ a partition of R, with $\bar{\pi} = \max\{\pi,\pi'\}$, they prove that

$$||D^{(k,\ell)}(f-P^{\rho}_{2m-1,2m-1})||_r \le K(\bar{\pi})^{2m-k-\ell} \qquad (1.6.1)$$

for all $0 \le k$, $\ell \le m$ with $0 \le k + \ell \le 2m - 1$, K some constant.

Zlamal [38], [39] proved interpolation theorems on a triangle for polynomials of the second, third and fifth degree. He considers all triangulations of the domain Ω and he proves that given 6 values at the vertices of the triangle and the mid points of its sides, there exists a quadratic polynomial

$$p(x,y) = a_1 + a_2x + a_3y + a_4x^2 + a_5xy + a_6y^2 \tag{1.6.2}$$

assuming the above given values at the corresponding points, which is uniquely determined. Constructing such polynomials for each triangle it turns out that we obtain a continuous and piecewise differentiable function. For cubic interpolation formulae he assumes 10 values to be given on each triangle. The values of the function and its first derivatives at the three vertices and as tenth the value of f at the center of mass of the triangle.

Zenisek [37] generalizes the above methods to higher degree interpolating polynomials.

Many other papers were written with significant results in this direction during the past few years and many others are still appearing in the literature. We single out Hall [25] and Babuska [2], [3] and [4].

CHAPTER II

NOTATION AND PROBLEM FORMULATION

Introduction. In the bicubic Interpolation scheme developed by deBoor, the rectangulation of the domain under discussion is of significant importance as bicubic splines have rectangular domains of definition. In our case, of Triangular Splines, triangulations and triangles are the basic elements of the discussion. The next section contains definitions regarding these notions, as we recall them from [18], and leading to the statement of our problem.

2.1 Notation and Definitions

Definition 2.1.1

By a triangulation of the plane we mean a covering of the plane by arbitrary triangles such that the open triangles are disjoint, the union of closed triangles is the plane and any two adjacent triangles have a common side.

We will denote by $\tau(R^2)$ the set of all triangulations of the plane.

Definition 2.1.2

By a regular triangulation $T \varepsilon \tau(R^2)$ we mean a triangulation of the plane which is invariant under any translation of the plane which takes one vertex into another. The restriction of T to

$\Omega = [0,1] \times [0,1]$ will be a regular triangulation of Ω if the boundary of Ω does not intersect any open triangle of T.

Consider now a uniform partition of $[0,1]$ of mesh h. This will give rise to a square mesh in Ω [see Fig. 2.1, a] which together with the indicated diagonals of the subsquares gives rise to a regular triangulation of Ω. We denote this by T_n if $1/h = n$. In Fig. 2.1, b we have a different mesh size on each axis. We denote by $T_{m,n}$ this triangulation if we have m respectively n subintervals in the x respectively y axis

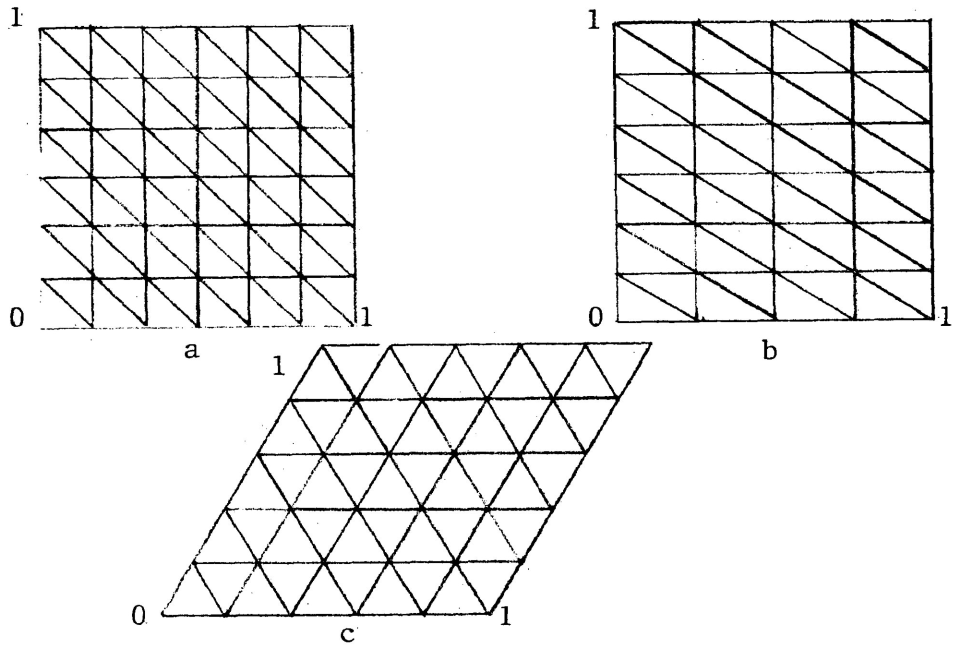


Fig. 2.1

We are especially interested in regular triangulations in which every triangle is equilateral [see Fig. 2.1, c] because most of our computations are simplified. In this case the square domain takes the form of a skewed parallelogram.

When we speak of a triangulation T in the rest of this paper we mean a regular triangulation. We will also denote by L_T the set of vertices of all the triangles of the triangulation T.

Definition 2.1.3

Let T be a triangulation of Ω. We denote by $S^{n,q}(T)$ the set of all functions f such that

$$D^{(q-i,i)}f \in C(\Omega), \qquad 0 \leq i \leq q$$

and $f \in P_n$ on each open triangle of T. Any function in $S^{n,q}(T)$ will be referred to as a *regular triangular spline.*

In this paper we will exclusively work with the space $S^{4,2}$, referring to its elements as quartic splines.

Definition 2.1.4

An element B of $S^{n,q}(T)$ is called a *normal basic spline* if it has compact support and

$$\sum_{(x,y)\in L_T} B(x,y) = 1$$

We define the convolution $b*g$ of any real-valued function g defined on L_T with b, a function defined in Ω, by the equation

$$(b*g)(x_1,x_2) = \sum_{(y_1,y_2)\in L_T} b(x_1-y_1,x_2-y_2)g(y_1,y_2) \qquad (2.1.1)$$

We recall one more definition from [18].

Definition 2.1.5

We say that the normal basic spline $B \in S^{n,q}(T)$, $n \geq 1$, $q \geq 0$ is k-exact if $B * g$ is in P_k whenever g is the restriction to L_T of an element of P_k.

2.2 Problem Formulation

The questions we are concerned with in this thesis are interpolation and quasi-interpolation, using triangular splines.

The periodic interpolation problem was treated by P. Frederickson [18]. Existence and under additional conditions uniqueness theorems were proved and error bounds were given.

We are interested in constructing and carrying out an algorithm which solves the above problem; i.e. given the function f defined in Ω and periodic, with period 1 in both variables and the space $S^{n,q}(T)$, T_h a triangulation of Ω, find a spline $s \in S^{n,q}(T)$ which interpolates f on L_T.

In Chapter V a local approximation scheme is introduced called quasi-interpolation. This scheme is defined so as to be of the same order of approximation as interpolation is.

We are concerned first in doing an error analysis of this scheme and next in constructing and carrying out an algorithm that solves it. Subsequently, the previous considerations led to the natural problem of seeking the existence of any relation between interpolation and quasi-interpolation. This was done in Chapter VI.

Since most of our work was the construction and application of algorithms we found it a necessary part of our problem, to introduce ideas and convenient ways of bridging the gap between theoretic developments and actual computation. This is done in Chapter III. P. Frederickson [18] introduced most of these ideas, but only to an extent that served his theoretical development.

CHAPTER III

COORDINATE SYSTEM, NORMAL BASIC QUARTIC

3.1 Barycentric coordinates

As it will become clear in the rest of this thesis, the most convenient coordinate system associated with T_h, in describing triangular splines and computations involving them, is the barycentric coordinate system. By barycentric coordinates for the triangle with vertices A, B and C we mean the three affine functions α, β and γ defined by

$$\alpha(A) = \beta(B) = \gamma(C) = 1 \tag{3.1.1}$$

and

$$\alpha(\overline{BC}) = \beta(\overline{AC}) = \gamma(\overline{AB}) = 0$$

i.e. each takes the value 1 at one vertex and vanishes on the opposite side.

It is obvious now that

$$\alpha + \beta + \gamma = 1 \tag{3.1.2}$$

holds at the three vertices of the triangle. Therefore it holds at every point of the triangle. Consequently, this identity reduces the number of the three functions α, β and γ required to determine any point on the triangle ABC to only two of them.

In order to see how the barycentric coordinates facilitate the treatment of triangular splines we must consider them in connection with a global coordinate system. Let, as in Fig. 3.1, x, y be the global coordinate system in such a way that two of the sides of the equilateral triangles constituting the regular triangulation T_h, $h = \frac{1}{n}$ of the domain Ω' under consideration, are parallel to x, y. Let also

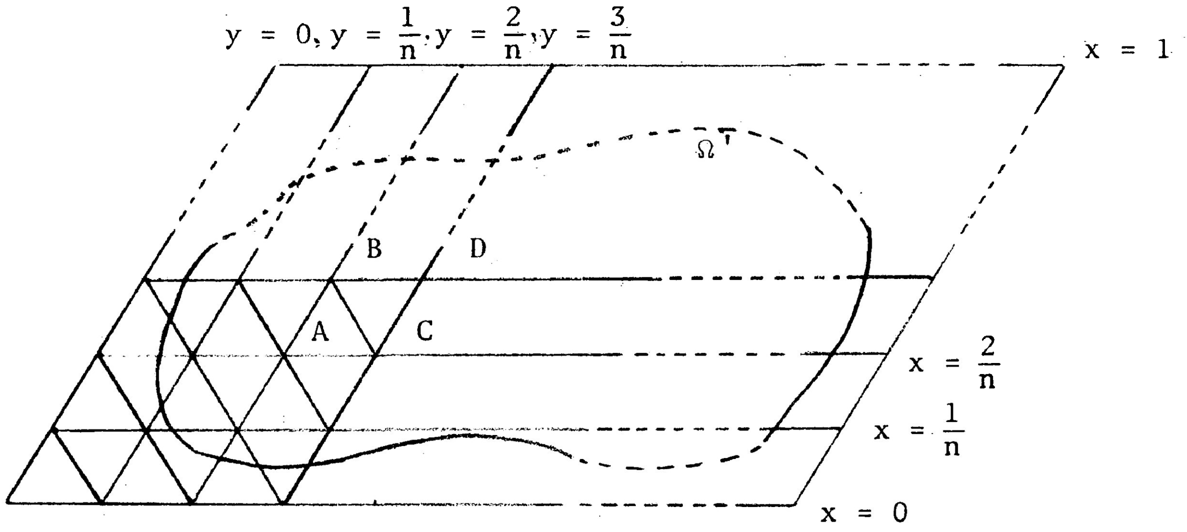


Fig. 3.1

$P(x,y)$ be a point in the triangle ABC. Consider now the numbers

$$i = [nx], \qquad j = [ny] \tag{3.1.3}$$

with $[\cdot]$ denoting the integer part function. It is easy to see that the point $(\frac{i}{n}, \frac{j}{n})$ is A, and the barycentric coordinates of the point P in ABC are

$$\gamma = nx - i$$
$$\beta = ny - j \tag{3.1.4}$$

and

$$\alpha = 1 - nx - ny + i + j$$

If the point $P(x,y)$ is in the triangle BCD its barycentric co-ordinates are

$$\beta = 1 - (nx-i)$$
$$\gamma = 1 - (ny-j) \tag{3.1.5}$$

and

$$\delta = nx + ny - i - j - 1$$

Consider now the integer

$$k = [nx+ny-i-j] \tag{3.1.6}$$

which indicates on which of the two above triangles the point P lies. If $k = 0$ respectively 1 the point is in the triangle ABC respectively BCD.

From the above discussion we conclude the following: Given a point (x,y) on Ω' there is a triangle with vertices

$$(\frac{i+1}{n}, \frac{j}{n}), \quad (\frac{i}{n}, \frac{j+1}{n}) \quad \text{and} \quad (\frac{i+k}{n}, \frac{j+k}{n})$$

where the point lies, with i, j and k given by (3.1.3) and (3.1.6). Additionally if $k = 0$ respectively $k = 1$ its barycentric coordiantes on the triangle it belongs to are given by (3.1.4) respectively by (3.1.5). The barycentric coordinate γ in (3.1.4) and β in (3.1.5) are given as functions only of x. We define

$$u = \begin{cases} nx - i & \text{if } k = 0 \\ 1 - (nx-i) & \text{if } k = 1 \end{cases} \tag{3.1.7}$$

and similarly

$$\upsilon = \begin{cases} ny - j & \text{if } k = 0 \\ 1 - (ny-j) & \text{if } k = 1 \end{cases} \tag{3.1.8}$$

The pair (u,v) provides two of the barycentric coordinates of the point (x,y) in the triangle it belongs. We will refer to these two as the *basic barycentric coordinates*.

Let us now consider a triangular spline $f \varepsilon S^{\ell,q}$ and defined on Ω. f is a piecewise polynomial and $f \varepsilon P_{\ell}$ on each triangle. Thus f will have an expression of the form

$$f(x,y) = \sum_{t=0}^{\ell} \sum_{s=0}^{t} b_{s+t(t+1)/2}\, x^{t-s} y^{s} \tag{3.1.9}$$

on each triangle. Using the basic barycentric coordinates of each triangle (3.1.7) and (3.1.8) the spline can be given by the unique expression

$$f'(u,v) = \sum_{t=0}^{\ell} \sum_{s=0}^{t} a_{s+t(t+1)/2}\, u^{t-s} v^{s} \qquad (3.1.9)'$$

on each triangle. For computation purposes we employ the following notation of the spline: We think of it as a four dimensional array F such that

$$F(i,j,k,\cdot) = (a_0, a_1, a_2, \ldots, a_{(\ell+1)(\ell+2)/2}) \tag{3.1.10}$$

with $(a_0, a_1, a_2, \ldots, a_{(\ell+1)(\ell+2)/2})$ the vector of the coefficients of the expression (3.1.9)' when the spline f is given by that expression on the triangle specified by the triple of integers (i,j,k) given by (3.1.3) and (3.1.6). It is clear now that

$$f'(u,v) = F(i,j,k,\cdot)\cdot V_{u,v} \qquad (3.1.11)$$

where (for $\ell = 4$)

$$V_{u,v} = (1,u,v,u^2,uv,v^2,u^3,u^2v,uv^2,v^3,u^4,u^3v,u^2v^2,uv^3,v^4) \qquad (3.1.12)$$

For later reference we derive here the formulas that give the first and second normal derivative of the function $f(\alpha,\beta,\gamma)$ defined on the equilateral triangle ABC of side h (see fig. 3.2). For the normal derivative of the function f along any side we have

$$\frac{\partial f}{\partial n} = \frac{\partial f}{\partial \alpha}\frac{\partial \alpha}{\partial n} + \frac{\partial f}{\partial \beta}\frac{\partial \beta}{\partial n} + \frac{\partial f}{\partial \gamma}\frac{\partial \gamma}{\partial n} .$$

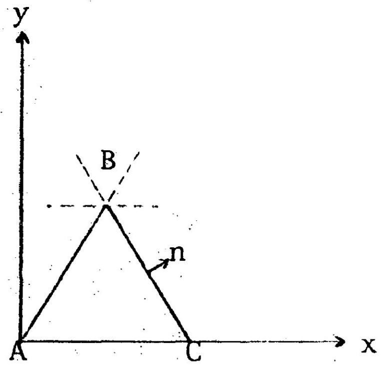


Fig. 3.2

But along the side BC

$$\frac{\partial \alpha}{\partial n} = -\frac{2}{\sqrt{3}h} \quad \text{and} \quad \frac{\partial \beta}{\partial n} = \frac{\partial \gamma}{\partial n} = \frac{1}{\sqrt{3}h} .$$

Hence

$$\frac{\partial f}{\partial n} = \frac{2}{\sqrt{3}h}\left(-\frac{\partial f}{\partial \alpha} + \frac{1}{2}\frac{\partial f}{\partial \beta} + \frac{1}{2}\frac{\partial f}{\partial \gamma}\right) \qquad (3.1.13)$$

Using these formulas we can get the second normal derivative along the side BC

$$\frac{\partial^2 f}{\partial n^2} = \frac{4}{3h^2}\left[\frac{\partial^2 f}{\partial a^2} + \frac{1}{4}\frac{\partial^2 f}{\partial \beta^2} + \frac{1}{4}\frac{\partial^2 f}{\partial \gamma^2} - \frac{\partial^2 f}{\partial \alpha \partial \beta} - \frac{\partial^2 f}{\partial \alpha \partial \gamma} + \frac{1}{2}\frac{\partial^2 f}{\partial \beta \partial \gamma}\right] \qquad (3.1.14)$$

Clearly now, interchanging the role of α, β respectively α, γ we can obtain the normal derivatives along the side AC respectively AB.

3.2 The Basic Quartic Spline

In this section we present the basic quartic spline. Since our subsequent work involves exclusively quartic splines a good understanding of this section is suggested.

Let T_n be a regular triangulation of R^2 consisting of equilateral triangles, as in Fig. 2.1.c. The basic quartic spline in $S^{4,2}$, denoted by Q, as introduced by Professor Frederickson, has hexagonal support of side 2h. In Fig. 3.3 this spline is shown centered at the point A = (2,2). In this case its support is the convex hull of the points $(2,2) + 2h\omega_s$, $s = 1, 2, \ldots, 6$ where

$$\omega_s = (\cos(2\pi s/6), \ \sin(2\pi s/6)) \qquad (3.2.1)$$

In the same Figure the triangles which consist the support of the basic quartic spline have been labeled by the triple of integers (i,j,k) as described in the previous section. The point (0,0) is the origin of the global coordinate system. Professor Frederickson [18] has given Q on the triangles ABC, BCD and BED. Using the notation of (3.1.10) we have

$$\begin{aligned} 12Q(2,2,0,\cdot) &= (6,0,0,-12,-12,-12,8,12,12,8,-1,-2,0,-2,-1) \\ 12Q(2,2,1,\cdot) &= (0,0,0,0,0,0,2,6,6,2,-1,-2,0,-2,-1) \qquad (3.2.2) \\ 12Q(2,3,0,\cdot) &= (1,-2,-4,0,6,6,2,0,-6,-4,-1,-2,0,2,1) \end{aligned}$$

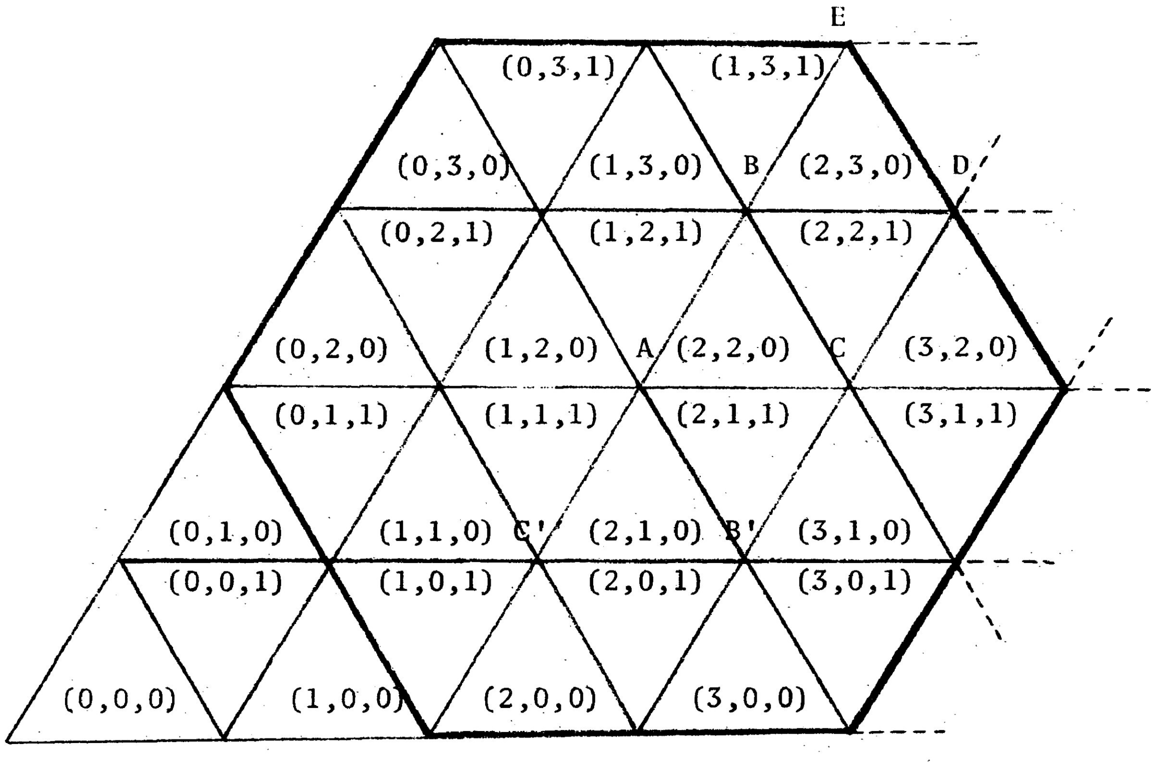


Fig. 3.3

Evaluation of Q in the remaining 21 triangles of its support is the problem we consider now. The fact that Q is symmetric with respect to all symmetries of the hexagon, which is its support, will be used.

Notice that there are three classes of triangles in the support of Q. Those occupying the inside hexagon of side h, those having one vertex on the boundary and those having one side on the boundary. The triangles ABC, BCD, and BED on which Q is given by (3.2.2) are representatives of these three classes.

Let us now generate Q on AB'C' being in the class of ABC. We consider the point R in AB'C' and its symmetric P in ABC with respect to one of the axis of symmetry of the hexagonal support. We denote the barycentric coordinates of the point P in ABC respectively R in $AB'C'$ by α, β and γ respectively α', β', γ'. Their basic coordinates are γ, β respectively β', α'. Besides we have

$$\alpha = \alpha', \quad \gamma = \beta' \quad \text{and} \quad \beta = \gamma' = 1 - \alpha' - \beta'.$$

Hence

$$\begin{aligned} Q(R) = Q(P) &= Q(2,2,0,\cdot)\cdot V_{\gamma,\beta} \\ &= Q(2,2,0,\cdot)\cdot V_{\beta',\gamma'} \\ &= Q(2,2,0,\cdot)\cdot V_{\beta',1-\alpha'-\beta'} \qquad (3.2.3) \\ &= Q(2,2,0,\cdot)\cdot M\cdot V_{\beta',\alpha'} \\ &\equiv Q(2,1,0,\cdot)\cdot V_{\beta',\alpha'} \end{aligned}$$

where M is a matrix such that

$$V_{\beta',1-\alpha'-\beta'} = M\cdot V_{\beta',\alpha'}$$

It turns out that the matrix

$$M = ROT\cdot REF$$

where

$$\mathrm{ROT} = \left[\begin{array}{ccccccccccccccc}
1 & 0 & 0 & & & & & & & & & & & & \\
0 & 0 & 1 & & & & & & & & & & & & \\
1 & -1 & -1 & & & & & & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 1 & & & & & 0 & & & & \\
0 & 0 & 1 & 0 & -1 & -1 & & & & & & & & & \\
1 & -2 & -2 & 1 & 2 & 1 & & & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & & & & & \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & & & & & \\
0 & 0 & 1 & 0 & -2 & -2 & 0 & 1 & 2 & 1 & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -2 & -2 & 0 & 0 & 1 & 2 & 1 \\
0 & 0 & 1 & 0 & -3 & -3 & 0 & 3 & 6 & 3 & 0 & -1 & -3 & -3 & -1 \\
1 & -4 & -4 & 6 & 12 & 6 & -4 & -12 & -12 & -4 & 1 & 4 & 6 & 4 & 1
\end{array}\right] \qquad (3.2.4)$$

and

$$\mathrm{REF} = \left[\begin{array}{ccccccccccccccc}
1 & & & & & & & & & & & & & & \\
 & 0 & 1 & & & & & & & & & & & & \\
 & 1 & 0 & & & & & & & & & & & & \\
 & & & 0 & 0 & 1 & & & & 0 & & & & & \\
 & & & 0 & 1 & 0 & & & & & & & & & \\
 & & & 1 & 0 & 0 & & & & & & & & & \\
 & & & & & & 0 & 0 & 0 & 1 & & & & & \\
 & & & & & & 0 & 0 & 1 & 0 & & & & & \\
 & & & & & & 0 & 1 & 0 & 0 & & & & & \\
 & & & & & & 1 & 0 & 0 & 0 & & & & & \\
 & & & 0 & & & & & & & 0 & 0 & 0 & 0 & 1 \\
 & & & & & & & & & & 0 & 0 & 0 & 1 & 0 \\
 & & & & & & & & & & 0 & 0 & 1 & 0 & 0 \\
 & & & & & & & & & & 0 & 1 & 0 & 0 & 0 \\
 & & & & & & & & & & 1 & 0 & 0 & 0 & 0
\end{array}\right] \qquad (3.2.5)$$

The above two matrices ROT and REF are not any random ones. We had anticipated that simple transformations on the coordinates would give the result (3.2.3). By this result we mean obtaining Q at the triangle (2,1,0)

$$Q(2,1,0,\cdot) = Q(2,2,0,\cdot)\cdot M.$$

The above transformations ROT and REF are rotation and reflection of the coordiantes. By rotation of the coordinates α, β and γ (clockwise rotation about the center of mass of the triangle ABC) we have

$$\alpha \rightarrow \beta, \quad \beta \rightarrow \gamma \quad \text{and} \quad \gamma \rightarrow \alpha = 1 - \beta - \gamma \tag{3.2.6}$$

and by reflection (about the axis vertical to BC through the vertex A) we have

$$\beta \rightarrow \gamma, \quad \gamma \rightarrow \beta \quad \text{and} \quad \alpha \rightarrow \alpha. \tag{3.2.7}$$

Having now a polynomial $f(\gamma,\beta) = F(i,j,k,\cdot)\cdot V_{\gamma,\beta}$ and applying the transformations (3.2.6) respectively (3.2.7) on its coordiantes we get

$$\begin{aligned} f'(1-\beta-\gamma,\gamma) &= F(i,j,k,\cdot)\cdot V_{1-\beta-\gamma,\gamma} \\ &= F(i,j,k,\cdot)\cdot \text{ROT}\cdot V_{\gamma,\beta} \end{aligned}$$

respectively

$$\begin{aligned} f''(\beta,\gamma) &= F(i,j,k,\cdot)\cdot V_{\beta,\gamma} \\ &= F(i,j,k,\cdot)\cdot \text{REF}\cdot V_{\gamma,\beta}. \end{aligned}$$

REMARK: We note that $(ROT)^3 = I$ and $(REF)^2 = I$, where I is the identity matrix. This corresponds to the fact that if we apply three rotations or two reflections on the coordinates, the polynomial remains unchanged. Thus the operators ROT and REF generate a group G of six elements

$$\text{ROT},\quad \text{REF},\quad \text{ROT}\cdot\text{REF},\quad \text{REF}\cdot\text{ROT},\quad \text{ROT}\cdot\text{ROT},\quad I \tag{3.2.8}$$

These six elements of G are precisely the operators M we need in order to generate Q on the remaining 21 triangles. Starting with $Q(2,2,0,\cdot)$ or $Q(2,2,1,\cdot)$ we generate Q on the other five triangles belonging to the class of the triangles ABC or BCD. In the class of BED there are twelve triangles. However, the six elements of G generate Q on all of them starting with $Q(2,3,0,\cdot)$.

In Appendix I we list an A.P.L./360 function QUARTICSPL which creates Q in all of its support. It is stored as a (4,4,2,15) array and is also displayed in the same Appendix as QUARTIC.

Differentiability of Q is easily checked. Obviously it is twice differentiable on the inside of each triangle. Applying formulas (3.1.13) and (3.1.14) we can establish that Q is twice differentiable everywhere on its support. For later reference we note that Q has the value of 1/2 at the center of its support and 1/12 at the six other lattice points inside the boundary.

3.3 Evaluation of Splines

The purpose of this section is mainly to describe a subroutine which carries out the evaluation of splines. This subroutine called EVAL has been written in FORTRAN and displayed in Appendix II. It is designed to evaluate a quartic spline at a point of its domain and when this domain is a rectangle.

Let $T_{m,n}$ be the triangulation of the rectangle where the quartic spline is defined. L_T will consist of the points (x_i,y_j), $0 \leq i \leq m$, $0 \leq j \leq n$. The spline is passed into the subroutine in its array form i.e. as a four dimensional array SPL(ID,JD,2,15) where ID = M, JD = N. It is easy now to see, from formula (3.1.11), that in order to evaluate the spline at the point (x,y) we need determine the triangle which the point lies in and the two barycentric coordinates of the point in that triangle. In the subroutine this information is obtained as follows: The pair of numbers X and Y $(X = (x-x_0)/(x_m-x_0)$, $Y = (y-y_0)/(y_n-y_0))$ is passed to the subroutine. It is easy to see now from the discussion in 3.1 that the integers

$$K = [A]$$

$$L = [B]$$

and

$$M = [A+B-K-L]$$

where $A = X \cdot ID$ and $B = Y \cdot JD$, determine the triangle, while

$$U = |M-(A-K)|$$

and

$$V = |M-(B-L)|$$

are the barycentric coordinates of the point in the specified triangle.

Proper functions are employed in FORTRAN to perform the above operations.

The subroutine can also be easily adapted for the evaluation of higher order splines.

CHAPTER IV

INTERPOLATION OF PERIODIC FUNCTIONS

4.1 Quartic Interpolation Operator

As it was stated in 2.2 our task in this chapter will be a demonstration of the feasibility of constructing an algorithm that solves the periodic interpolation problem. The description of the algorithm will be given and numerical results of its implementation will be presented.

We are going to make use of $S_h^{4,2}$ as the space of interpolating splines. Quartic splines, being piecewise polynomials of degree 4, could be of more practical value than any other splines. The work of section 3.3, where the construction of the basic normal quartic spline was done, will be very helpful in the numerical part of this chapter.

The function we are going to interpolate will be in the space $C^m(R^2)$, $m > 4$ and periodic on R^2 with period 1 in both variables. The fact that any function defined on a bounded subset of R^2 can be extended to a periodic function on R^2, with period set a rectangle covering its domain, allows us to use periodic splines as interpolating splines for such functions. On the other hand the choice of the period set to be the domain $\Omega = [0,1] \times [0,1]$ was made only for convenience in computations.

Let T_h, $h > 0$ be a regular triangulation of Ω. We will denote by $S(f)$ the quartic spline interpolant of f; i.e., we have

$$S(f) \in S_h^{4,2}(\Omega)$$

and (4.1.1)

$$S(f)(x,y) = f(x,y), \quad (x,y) \in L_T$$

We denote by f° and Q° the restriction to L_T of f and Q respectively. The spline interpolant $S(f)$ of f is given [18] by

$$S(f) = Q * \tilde{f} \qquad (4.1.2)$$

where $\tilde{f}$ is the solution of the system

$$f^\circ = Q^\circ * \tilde{f} \qquad (4.1.3)$$

According to (2.1.1)

$$S(f)(x_1,x_2) = \sum_{(y_1,y_2)\in L_T} Q(x_1-y_1,x_2-y_2)\tilde{f}(y_1,y_2) \qquad (4.1.4)$$

Additionally the quartic spline operator S is 3-exact [18], (see definition 2.1.5).

4.2 Mathematical description of the algorithm

It is clear from (4.1.2) that the construction of the interpolating spline requires two operations: The solution of the system (4.1.3) to obtain $\tilde{f}$ and the convolution of $\tilde{f}$ with the normal basic quartic spline Q.

The solution of the system $f^\circ = Q^\circ * \tilde{f}$ is not an easy task. We are faced with a tremendous problem. In order to show the magnitude of it we will consider the mesh size $h = 1/32$ (the smallest we use in our implementation later on). Obviously, f is given in 1024 points in this case and the system (4.1.3) has 1024 equations with 1024 unknowns.

It is really not feasible to invert a 1024×1024 matrix by Gauss elimination. The number of operations (multiplications and divisions) needed to triangularize an $n \times n$ matrix and solve the triangular system is [27]

$$\frac{n(n^2-1)}{3} \quad \text{and} \quad \frac{n(n+1)}{2} \tag{4.2.1}$$

respectively. When $n = 1024$ these numbers are slightly larger than 3.5×10^8 and 5.2×10^5. The size of the matrix also discourages us of considering some well known iterative schemes although we have to deal with a sparse matrix.

It seems that the Fourier transform provides an optimum way for solving the system under discussion. We define the Fourier transform [18] of the function g defined on L_T by

$$g^\wedge(x) = h^2\sqrt{3}/2 \sum_{y \in L_T} e^{-2\pi i x \cdot y} g(y) \tag{4.2.2}$$

Obviously, this is an extension of the usual Fourier transform. It follows immediately that the inverse Fourier transform of g is given by

$$g^{\vee}(x) = g^{\wedge}(-x) \tag{4.2.3}$$

Furthermore, the operation of convolution transforms into pointwise multiplication by means of

$$(b*g)^{\wedge} = (2/h^2\sqrt{3})b^{\wedge}\cdot g^{\wedge} \tag{4.2.4}$$

Using formulas (4.2.2) and (4.2.4) now, equation (4.1.3) gives

$$f^{\sim} = [f^{\circ\wedge}/(2Q^{\circ\wedge}/h^2\sqrt{3})]^{\vee} \tag{4.2.5}$$

In order to evaluate the above formula we only need take the Fourier transform of f° and Q° and the inverse transform of their pointwise division. The number of operations required to carry out the Fourier transform of f° or Q° using the Cooley-Tukey algorithm, and when $n = r^m$ [15], is

$$rm\log_r n. \tag{4.2.6}$$

Hence in the above case where $n = 2^{10}$ the whole number of operations required to evaluate the expression (4.2.5) is

$$\begin{aligned} 3(2.n.10) + n &= 61.n \\ &< 6.25 \times 10^4 \end{aligned}$$

After we obtain the solution $f^{\sim}$ to the system (4.1.3) the next thing for the construction of the spline is to carry out the convolution of Q with $\tilde{f}$. This is done with the use of formula (4.1.4).

In Chapter VI we develop an iterative scheme for the solution of the system (4.1.3). We leave the discussion of it to that chapter because we need introduce quasi-interpolation first.

4.3 Implementation of the algorithm

Let T_n, $n > 1$ be a regular triangulation of Ω (Ω considered in the skewed form of Fig. 2.1.c) and $f°$ be given at the lattice L_T.

From the computational point of view and in order to construct the spline interpolant (4.1.2) we need two subroutines which will carry out the two operations described in the previous section, namely the fourier transform of $f°$, $Q°$ and $f°^{\wedge}/Q°^{\wedge}$ and the convolution of $\tilde{f}$ with Q. We name them FASTF and CONVOLP respectivley. The first one was taken from [34] and translated into Fortran. Passing $f°$ respectively $Q°$, in the form of a one-dimensional array into the subroutine it returns $f°^{\wedge}$ respectively $Q°^{\wedge}$. Note that $Q°$ is 0 everywhere except

$$Q°(0,0) = \frac{1}{2} \text{ and } Q°(0,1) = Q°(1,0) = Q°(0,n) \quad Q°(1,n) = Q°(n,0) = Q°(n,1) = \frac{1}{12} .$$

After $\tilde{f}$ has been evaluated, and we have a main program which would do the pointwise division of $f°$ and $Q°$, the subroutine CONVOLP is called (see Appendix III).

Input for CONVOLP

FT1L	$\tilde{f}$ as an $n \times n$ array.
Q	The basic quartic spline as a $4 \times 4 \times 2 \times 15$ array. It is being read into the program as displayed in Appendix I. (Note that in this Appendix Q is multiplied by a factor of 12).
ID,JD	The number of subintervals we want to construct the interpolating spline in the direction x and y respectively $1 \leq$ ID, JD $\leq n$.

Output of CONVOLP

SPL	the interpolating spline as an ID $\times$ JD $\times 2 \times 15$ array.

From (4.1.4) we note that, in order to construct the interpolating spline SPL on the boundary triangles of Ω, we need values of $\tilde{f}$ on lattice points outside Ω. But since f is periodic so is $\tilde{f}$. In the subroutine CONVOLP instead of extending FTL to an $(n+2) \times (n+2)$ array to include the required values of $\tilde{f}$, we employ the modulo function $(\mathrm{mod}(n+1))$ on the subscripts of FT1L.

Additional subroutines that we used in the program are: FUNCT, REORDER, EVAL and NORM. Subroutine FUNCT created f° at the lattice points L_T for the test function we used.

REORDER was used in connection with FASTF and it was again taken from [34]. It permutes data from reverse binary to normal order. EVAL was described in section 3.3. Finally, NORM subroutine was

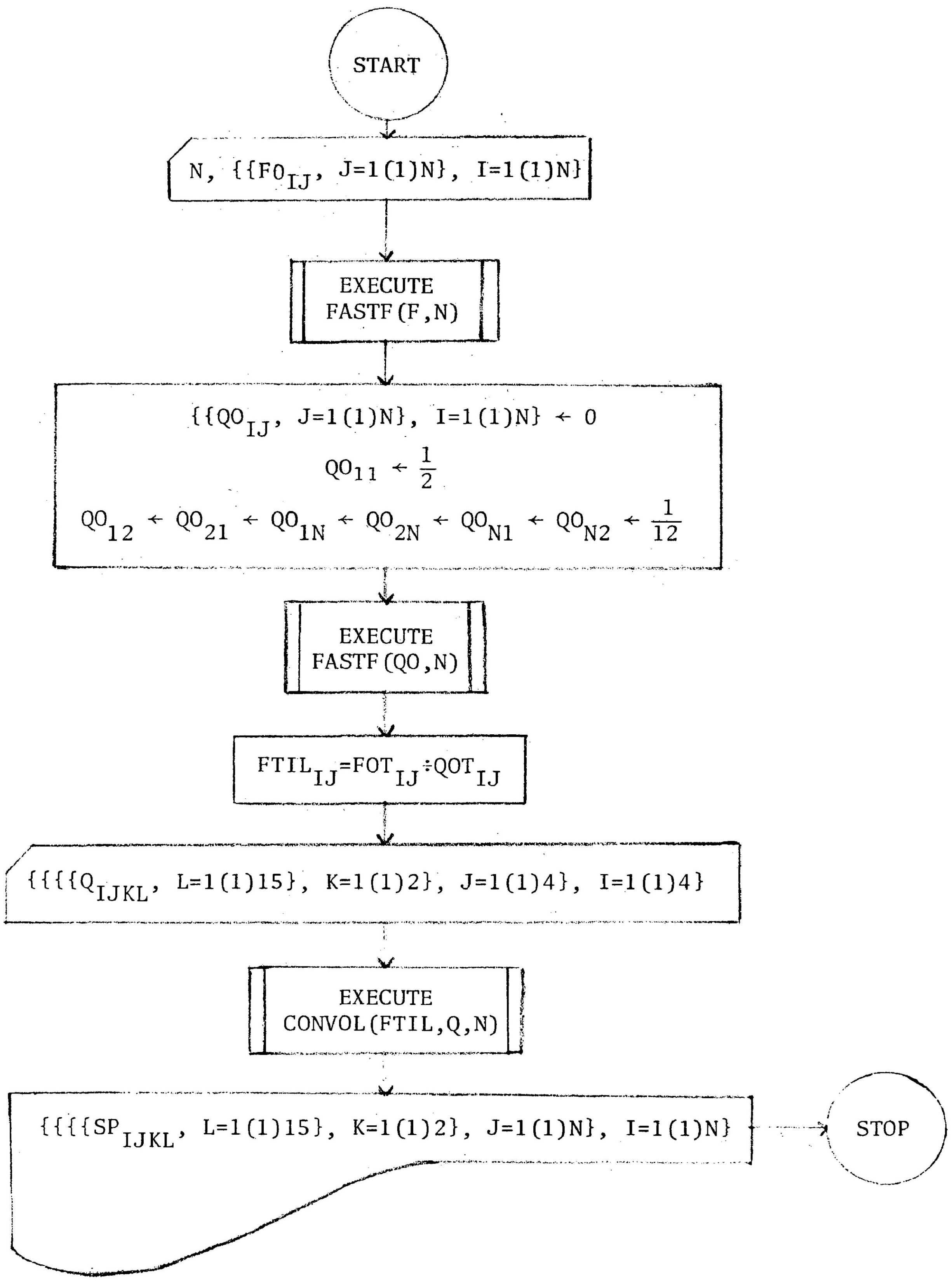

START
N, {{FO_{IJ}, J=1(1)N}, I=1(1)N}
EXECUTE FASTF(F,N)
{{QO_{IJ}, J=1(1)N}, I=1(1)N} $\leftarrow$ 0
$QO_{11} \leftarrow \frac{1}{2}$
$QO_{12} \leftarrow QO_{21} \leftarrow QO_{1N} \leftarrow QO_{2N} \leftarrow QO_{N1} \leftarrow QO_{N2} \leftarrow \frac{1}{12}$
EXECUTE FASTF(QO,N)
$FTIL_{IJ}=FOT_{IJ} \div QOT_{IJ}$
{{{{Q_{IJKL}, L=1(1)15}, K=1(1)2}, J=1(1)4}, I=1(1)4}
EXECUTE CONVOL(FTIL,Q,N)
{{{{SP_{IJKL}, L=1(1)15}, K=1(1)2}, J=1(1)N}, I=1(1)N}
STOP

Fig. 4.1

used to calculate the L_1, L_2 and L_∞ norms of the error function $e(x,y) = f(x,y) - S(f)(x,y)$; i.e., calling FUNCT and EVAL at the same set of points $\{(x_i,y_j),\ 1 \le i \le k,\ 1 \le j \le \ell\}$ we produced an error array which we passed to the subroutine NORM to compute the above norms.

An illustration in flow chart form of the main routine is given in Fig. 4.1. The correspondence between parameters used in the flow chart (program) and actual ones is the following

Q	for basic quartic
FO	>> f°
QO	>> Q°
N	>> n
FOT	>> $f^{\circ\wedge}$
QOT	>> $Q^{\circ\wedge}$
FTIL	>> $\tilde{f}$

4.4 Numerical results

The test function we used implementing the above algorithm was

$$f(x,y) = (1-\cos 2\pi x)\cdot(1-\cos 2\pi y)/4 \qquad (4.4.1)$$

which is periodic on R^2 with period 1 in both variables.

Three different mesh sizes were tried, namely $\frac{1}{8}$, $\frac{1}{16}$ and $\frac{1}{32}$; (n=8, 16 and 32). The error function was evaluated at the $4 \times n^2$ points

$$(\frac{k}{n} + \frac{1}{4n}, \frac{\ell}{n} + \frac{1}{4n}),\ (\frac{k}{n} + \frac{1}{4n}, \frac{\ell}{n} + \frac{3}{4n}),\ (\frac{k}{n} + \frac{3}{4n}, \frac{\ell}{n} + \frac{1}{4n}) \text{ and } (\frac{k}{n} + \frac{3}{4n}, \frac{\ell}{n} + \frac{3}{4n}) \tag{4.4.2}$$

with $0 \leq k,\ \ell \leq n - 1$

Table 4.1 contains the norms L_1, L_2 and L_∞ obtained.

Table 4.1

L_p \ h	1/8	1/16	1/32
L_1	5.379915×10^{-4}	2.652338×10^{-5}	1.549839×10^{-6}
L_2	1.825893×10^{-3}	9.587575×10^{-5}	5.731885×10^{-6}
L_∞	6.673183×10^{-4}	3.295835×10^{-5}	1.936131×10^{-6}

The above numerical results compare fairly well with the theoretical ones obtained by Professor Frederickson. He proved in [18] that if $f \in C^m(R^2)$, $m > 4$, the order of approximation by quartic splines is 4. Thus one should expect the error to decrease by a factor of 16 when the mesh size is cut down by half. As a matter of fact the data of table 4.1 give even larger factors.

All the above calculations were carried out in double precision arithmetic on the IBM/360 computer at Lakehead University.

CHAPTER V

QUASI-INTERPOLATION USING QUARTIC SPLINES

5.1 Introduction

Our motivation for this Chapter comes from the paper [10] by deBoor, C. and Fix, G. J., and [19] by Frederickson. We outline the problem treated by the authors in [10]: Let Ω be a region in R^n, π a partition of R^n into rectangles and S_π^k $(k \geq 1)$ the corresponding space of spline functions of degree $k - 1$. The authors explicitely construct for each function in $W_\infty^k(\Omega)$ a spline $F_\pi f \in S_\pi^k$ which they call the spline interpolant of f with the following properties:

(i) F_π is local in the sense that the value of $F_\pi f$ at a point x depends only on the values of f in a uniformly small neighbourhood of x.

(ii) F_π reproduces polynomials; $F_\pi(x^\gamma) = x^\gamma, |\gamma| < k$. Finally they prove that this approximation scheme is of order k.

$$F_\pi f - f = \mathcal{O}(|\pi|^k) \tag{5.1.1}$$

In the rest of this chapter we derive a Quasi-interpolation scheme using Quartic Triangular splines.

5.2 Quartic Quasi-Interpolant Operator

Let T_h be a triangulation of the plane in which every triangle is equilateral with the origin being a point in L_T. For $f \varepsilon C^n(R^2)$, $n \geq 4$ we shall determine a spline $S_h f \varepsilon S^{4,2}$ with properties

(i)' S_h is local as F_π in (i)

(ii)' S_h reproduces polynomials $p \varepsilon P_3$.

Moreover, it will be of the form

$$(S_h f)(x) = \sum_{y \varepsilon L_T} f^{\sim}(y)\, Q(x-y) \tag{5.2.1}$$

where $x = (x_1, x_2)$, $y = (y_1, y_2)$ and

$$f^{\sim}(y) = H(f)(y) \tag{5.2.2}$$

$$= \mu_0 f(y) + \sum_{s=1}^{6} \mu_s f(y + h\omega_s)$$

with the constants μ_s, $s = 0, 1, \ldots, 6$ invariant with respect to x and h, and ω_s given as in (3.2.1).

In section 5.3 we will prove that

$$||S_h f - f||_\infty \leq K|f|_4 h^4 \tag{5.2.3}$$

where K is a constant and $|f|_4$ is the Sobolev seminorm defined by

$$|f|_4 = \sup_{x \varepsilon \Omega} \sup_{\theta} \{|\sum_{k=0}^{4} \binom{4}{k} \cos^k\theta \sin^{4-k}\theta D^{k,4-k} f(x)|\} \tag{5.2.4}$$

Property (i)' is obviously satisfied by (5.2.1). We will determine the constants μ_s, $s = 0, 1, \ldots, 6$ so that property (ii) is satisfied as well. This can be done in a straightforward algebraic way. Let $p(x_1,x_2) \in P_3$.

$$p(x_1,x_2) = a_0+a_1x_1+a_2x_2+a_3x_1^2+a_4x_1x_2+ a_5x_2^2+a_6x_1^3+a_7x_1^2x_2+a_8x_1x_2^2+a_9x_2^3 \quad (5.2.5)$$

We want

$$S_hp(x) = \sum_{y\varepsilon L_T} p^{\sim}(x)\ Q(y-x) = p(x)$$

or by 5.2.2

$$p(x) = \sum_{y\varepsilon L_T} (\mu_0 p(x)+ \sum_{s=1}^{6} \mu_s p(x+h\omega_s))\ Q(y-x) \qquad (5.2.6)$$

Since the constants μ_s are invariant with respect to x and h, equation (5.2.6) is valid when $x = (0,0)$ and $h = 1$. Hence,

$$p(0,0) = a_0 = \sum_{y\varepsilon L_{T_1}} (\mu_0 a_0+ \sum_{s=1}^{6} \mu_s p(\omega_s))Q(y)$$

Taking into consideration that $Q(0,0) = \frac{1}{2}$, $Q(\omega_s) = \frac{1}{12}$ $s = 1, 2, \ldots, 6$ and $Q(y) = 0$ for any other $y \varepsilon L_T$, when Q is centered at $(0,0)$, after equating the coefficients a_j, $j = 0, 1, \ldots, 9$ in the above equation, we obtain the following system of equations (omitting the two redundant ones)

$$\begin{aligned}
\mu_0 + \mu_1 + \mu_2 + \mu_3 + \mu_4 + \mu_5 + \mu_6 &= 1 \\
\mu_2 + \mu_3 - \mu_5 - \mu_6 &= 0 \\
- \mu_1 - \mu_2 + \mu_4 + \mu_5 &= 0 \\
\mu_0 + \mu_1 + 4\mu_2 + 4\mu_3 + \mu_4 + 4\mu_5 + 4\mu_6 &= 0 \\
\mu_0 + \mu_1 + 7\mu_2 + \mu_3 + \mu_4 + 7\mu_5 + \mu_6 &= 0 \\
\mu_0 + 4\mu_1 + 4\mu_2 + \mu_3 + 4\mu_4 + 4\mu_5 + \mu_6 &= 0 \\
\mu_1 + 5\mu_2 + \mu_3 - \mu_4 - 5\mu_5 - \mu_6 &= 0 \\
24\mu_1 + 24\mu_2 + \mu_3 - 24\mu_4 - 24\mu_5 - \mu_6 &= 0
\end{aligned} \tag{5.2.7}$$

The nontrivial solution of the last seven equations of this system is

$$\mu_0 = 3k, \ \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 = \mu_6 = -\frac{k}{6}, \quad k \in R$$

Substitution in the first one gives the following solution

$$\begin{aligned}
\mu_0 &= 3/2 \\
\mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 = \mu_6 &= -1/12
\end{aligned} \tag{5.2.8}$$

With these values formula (5.2.2) becomes

$$f^{\sim}(y) = H(f)(y) = 3/2\ f(y) - 1/12 \sum_{s=1}^{6} f(y+h\omega_s) \tag{5.2.9}$$

It is clear that given f on the lattice points L_T, $f^{\sim}$ can be easily derived by this formula. Let us now turn again to the finite domain Ω. Obviously, given f on the grid points L_T of a regular triangulation T_h of Ω, formula (5.2.9) cannot give $f^{\sim}$ on the boundary grid points. This would require values of f on grid points outside Ω. Additionally, if one is to construct $S_h f$ over the whole region Ω he would need $f^{\sim}$ (as we point out on 5.3) outside Ω as well. If f is periodic the above problem is not hard to overcome. Actually in this case as we shall see in section 5.4 the quasi-interpolation scheme is much easier than interpolation from the computation point of view.

However, in the general case one is led naturally to <u>extrapolation</u> of $f^{\sim}$.

Our next section is devoted to an error analysis of the Quasi-interpolation scheme.

5.3 <u>Error Analysis</u>

Let $f \in C^n(\Omega)$, $n \geq 4$ and T_h be a regular triangulation of Ω. We assume that all fourth order derivatives of f are bounded on Ω and denote by M the seminorm $|f|_4$ of f defined by (5.2.4). We denote by $S_h f$ the quasi-interpolant of f on Ω as given by (5.2.1).

<u>Theorem</u>: Given f with the above assumptions then

$$||f-S_h f||_\infty \leq 0.420\ Mh^4 \tag{5.3.1}$$

Proof: Given any point $x = (x_1,x_2) \in \Omega$ we choose a point $y_1 = (y_{11},y_{12}) \in L_T$ such that

$$|x-y_1| = \min\{|x-y|:y\in L_T\} \tag{5.3.2}$$

(see Fig. 5.1). Therefore, the point x lies in a hexagon of side $\frac{\sqrt{3}}{3}h$ centered at y_1. Our error analysis will be based on the assumption that x lies in one of the twelve right triangles making up the above hexagon. Let this be the triangle ABC (see Fig. 5.1) which we will denote by U. As it will become clear, the choice of any one of these triangles will give the same result. This is due to the symmetry of the seminorm $|\cdot|_4$ and the symmetry of the basic normal quartic spline Q.

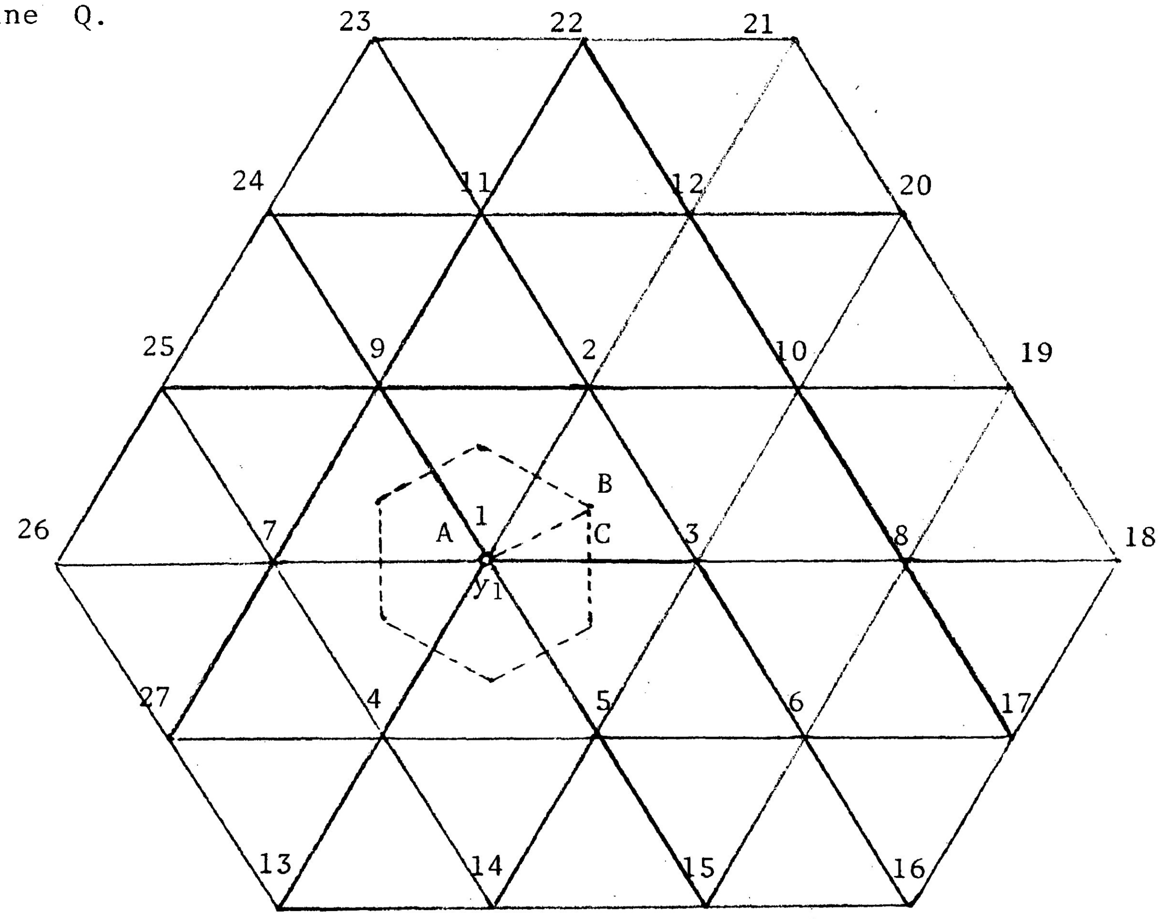


Fig. 5.1

Let us denote by $f^*(z)$ the third order Taylor's expansion of f about x. Then

$$f(z) = f^*(z) + R(z) \tag{5.3.3}$$

with

$$f^*(z) = \sum_{|\alpha|=0}^{3} D^\alpha f(x)(x-z)^\alpha/\alpha! \tag{5.3.4}$$

and

$$R(z) = \sum_{|\alpha|=4} D^\alpha f(x+\theta(x-z))(x-z)^\alpha/\alpha! \tag{5.3.5}$$

for some θ, $0 < \theta < 1$.

The error between the function f and its quasi-interpolant at the point x is

$$\begin{aligned} f(x) - (S_h f)(x) &= f(x) - f^*(x) + f^*(x) - (S_h f^*)(x) + (S_h f^*)(x) - (S_h f)(x) \\ &= (S_h f^*)(x) - (S_h f)(x) \\ &= (S_h R)(x) \end{aligned} \tag{5.3.6}$$

since $f(x) = f^*(x)$ from (5.3.3) and $f^*(x) = (S_h f^*)(x)$ because the operator S_h reproduces polynomials of degree 3.

Expanding (5.3.6), using (5.2.1) and (5.2.2), we get

$$\begin{aligned} |f(x)-(S_h f)(x)| &= \Big| \sum_{y\in L_T} \Big(\frac{3}{2} R(y) - \frac{1}{12} \sum_{s=1}^{6} R(y+\omega_s)\Big) Q(x-y) \Big| \\ &= \Big| \sum_{y\in L_T} R(y) \Big(\frac{3}{2} Q(x-y) - \frac{1}{12} \sum_{s=1}^{6} Q(x+\omega_s-y)\Big) \Big| \\ &\le \sum_{y\in L_T} |R(y)| \Big(\frac{3}{2} Q(x-y) + \frac{1}{12} \sum_{s=1}^{6} Q(x+\omega_s-y)\Big) \end{aligned} \tag{5.3.7}$$

In order to obtain a bound for this expression we need bounds for $R(y)$ and $Q(x-y)$ when y is a lattice point. From (5.3.5) we get

$$\begin{aligned} |R(z)| &\leq \left|\sum_{k=0}^{4} D^{(k,4-k)} f(z')(x_1-y_{11})^k (x_2-y_{12})^{4-k}/k!(4-k)!\right| \\ &\leq \frac{1}{4!}\left|\sum_{k=0}^{4} \binom{4}{k} D^{(k,4-k)} f(z') \frac{(x_1-y_{11})^k (x_2-y_{12})^{4-k}}{|x-z|^4}\right| |x-z|^4 \\ &\leq \frac{M}{4!} |x-z|^4 \end{aligned} \tag{5.3.8}$$

where $z' = x + \theta(x-z)$ and $|x-z|$ is the distance between the points x and z. Now if we define

$$r(z) = \sup_{x \varepsilon U} |x-z|^4 \tag{5.3.9}$$

we get

$$|R(z)| \leq \frac{M}{4!} r(z) \tag{5.3.10}$$

Using this inequality we can have bounds for $R(y)$, $y \varepsilon L_T$. Obviously $R(y)$ can be nonzero for every $y \varepsilon L_T$. But the values of Q are zero when the distance between x and y or $y - \omega_s$ is greater than 2h. There are 27 points $y_i \varepsilon L_T$, $i = 1, 2, \ldots, 27$ such that the distance $|x-y_i|$ or $|x-(y_i-\omega_s)|$ for some x, is less than 2h. In Fig. 5.1 these points have been labeled by the numbers i, $i = 1, 2, \ldots, 27$ instead of y_i. If we consider a point $y \varepsilon L_T$ outside the support of these 27 points, then the basic quartic spline centered at that point or any of the six lattice points surrounding it will be zero at any point $x \varepsilon U$.

For these 27 points we can obtain numerical values of $r(y_i)$, $i = 1, 2, \ldots, 27$ [5.3.10] and

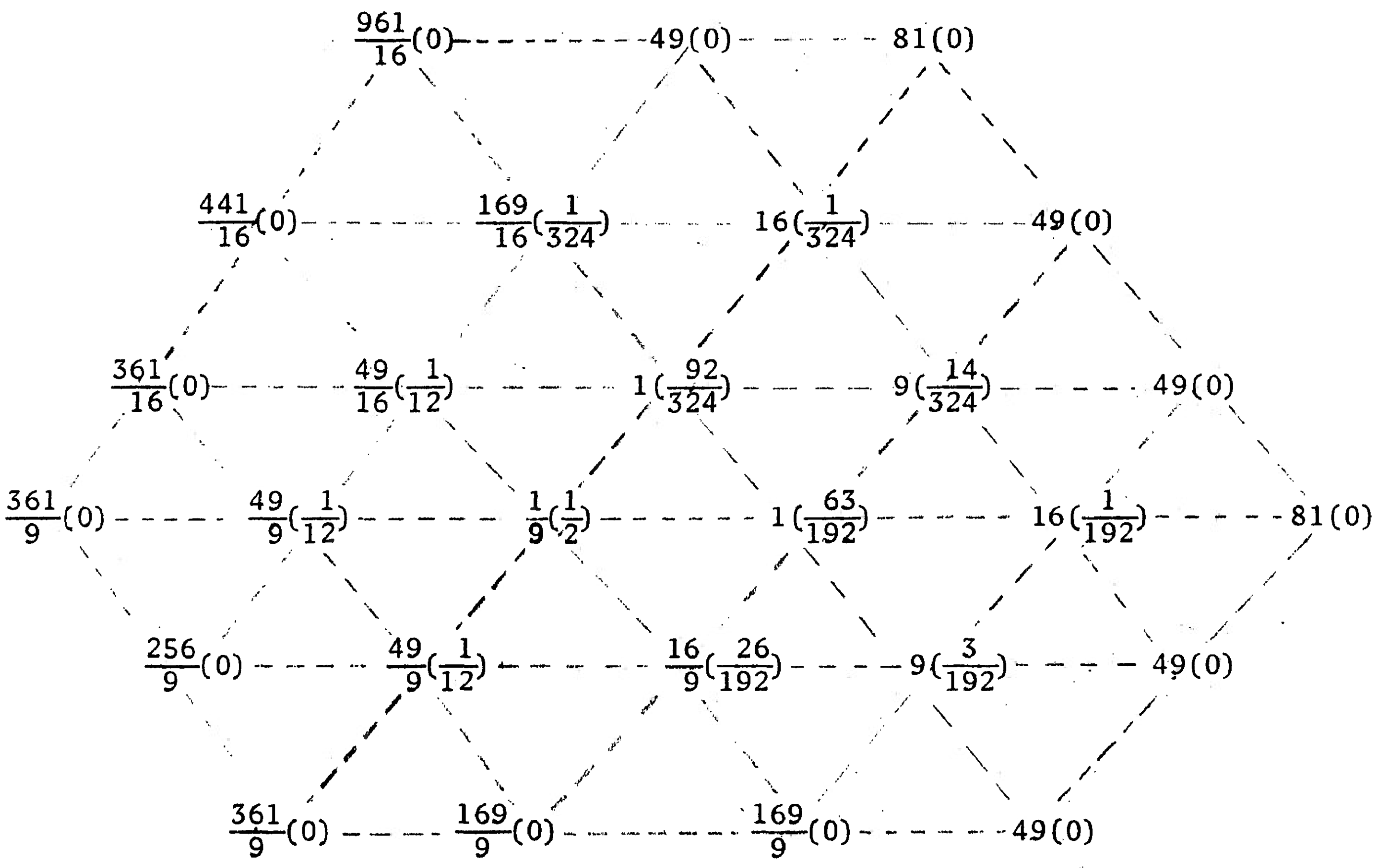
$$q(y_i) = \sup_{x \in U} Q(x-y_i), \; i = 1,2,\ldots,27 \qquad (5.3.11)$$

When, for example, $i = 1$

$$r(y_i) = (\frac{\sqrt{3}}{3} h)^4 = \frac{1}{9} h^4$$

and $q(y_1) = \frac{1}{2}$.

In the following display we give all the constants A_i such that $r(y_i) = A_i h^4$ and the constants $q(y_i)$ (in parenthesis) corresponding to the 27 points i of Fig. 5.1.



Using the above data for $R(y)$ and $Q(x-y)$, we calculated the expression (5.3.7) with an A.P.L. function to get

$$|f(x)-(S_h f)(x)| \le 0.420 \, Mh^4$$

Hence the proof of the Theorem is complete.

The above error analysis is valid when $f^{\sim}$ is exact. However, as we mentioned in the previous section, construction of $S_h f$ over the whole region Ω requires values of $f^{\sim}$ on and outside the boundary of Ω. If these values are not available exactly (see section 5.5), the above error anlaysis is valid only on an interior sub-region of Ω whose distance from the boundary of Ω is $2h$.

5.4 An Algorithm for Periodic Quasi-Interpolation and Numerical Results

The spline interpolant $S_h f$ of Quasi-Interpolation is much easier to compute than the one for interpolation. The two major steps again are evaluation of $f^{\sim}$ and convolution of $f^{\sim}$ with Q. However, $f^{\sim}$ given explicitely by

$$f^{\sim}(y) = 3/2 \, f(y) - 1/12 \sum_{s=1}^{6} f(y+h\omega_s), \quad y \in L_T$$

provides no difficulty in its computation especially in the periodic case we examine here.

Suppose that the period set of f is Ω and f° is given on L_T of the triangulation T_n of Ω. As we mention in (4.3) if $f^{\sim}$

is given on this same set of points, then the subroutine CONVOLP constructs $S_{1/n}f$ over all of Ω.

In order to evaluate $f^{\sim}$ on the above set of points we extend f° by periodicity to a set of $(n+2) \times (n+2)$ points. This in the program means extending the array of $n \times n$ points to an array of $(n+2) \times (n+2)$ points in such a way that the first and the $(n+2)$-nd columns respectively rows of the new array are equal to the nth and the first columns respectively rows of the previous array. In Fig. 5.2 we give a flow chart of the main program we used. FO is the given $n \times n$ array f° while FON the extended $(n+2) \times (n+2)$ one.

Although quasi-interpolation and interpolation are of the same order of approximation, quasi-interpolation is less accurate than interpolation. This is our penalty for less computational work and was found experimentally. We applied interpolation and quasi-interpolation to the periodic function (4.4.1), given on Ω at the same lattice points. Table 5.1 lists the norms obtained using quasi-interpolation for the above function, corresponding to the norms of table 4.1 with the error function evaluated again at the set of points (4.4.2). The subroutines FUNCT, EVAL and NORM are being used here without change.

Table 5.1

L_p \ h	1/8	1/16	1/32
L_1	6.401972507×10^{-3}	4.806234811×10^{-4}	3.146364383×10^{-5}
L_2	7.344676049×10^{-3}	5.490275040×10^{-4}	3.588237149×10^{-5}
L_∞	1.588539084×10^{-2}	1.177734939×10^{-3}	7.691211366×10^{-5}

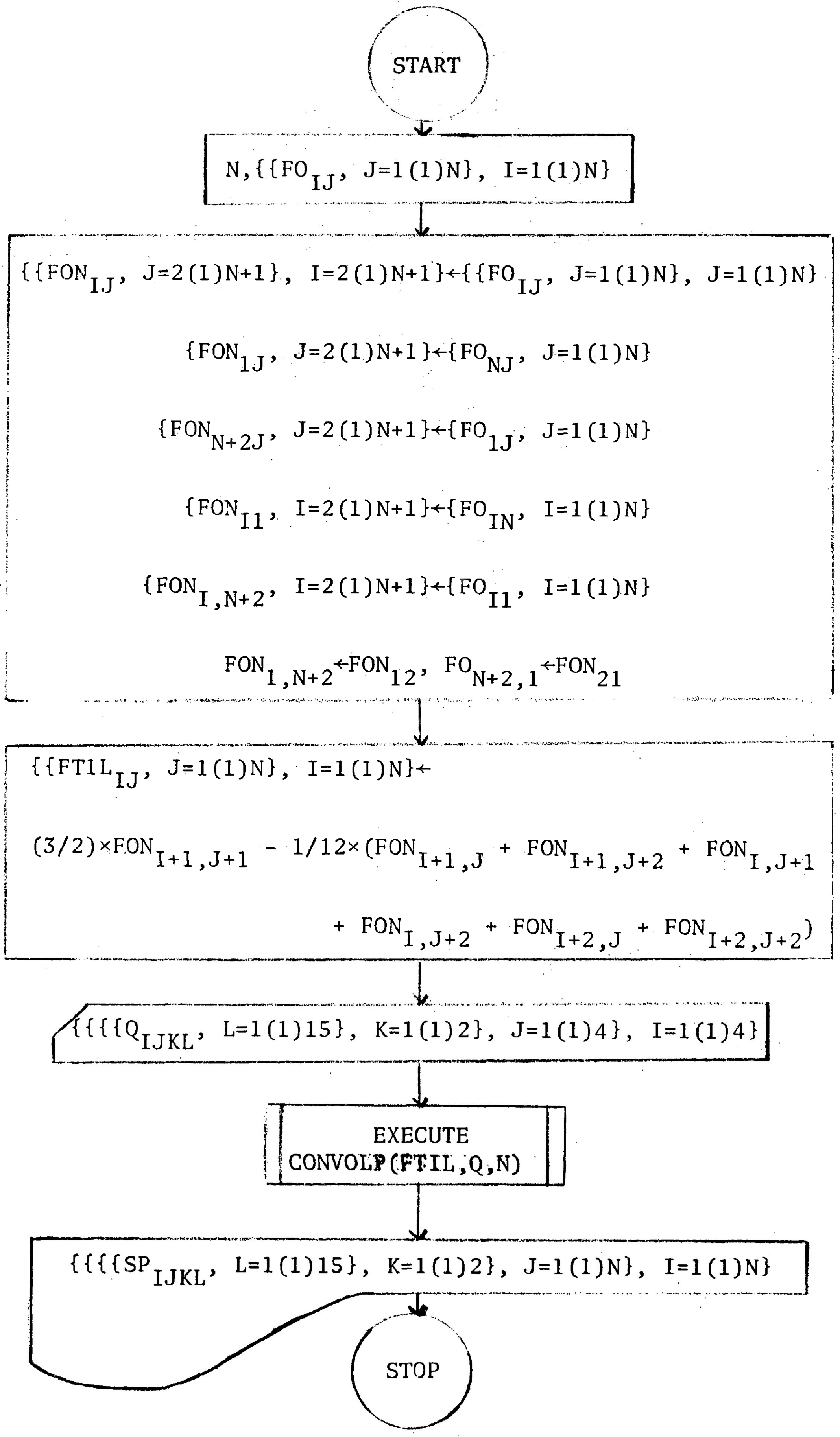

START
$N, \{\{FO_{IJ}, J=1(1)N\}, I=1(1)N\}$
$\{\{FON_{IJ}, J=2(1)N+1\}, I=2(1)N+1\} \leftarrow \{\{FO_{IJ}, J=1(1)N\}, J=1(1)N\}$
$\{FON_{1J}, J=2(1)N+1\} \leftarrow \{FO_{NJ}, J=1(1)N\}$
$\{FON_{N+2J}, J=2(1)N+1\} \leftarrow \{FO_{1J}, J=1(1)N\}$
$\{FON_{I1}, I=2(1)N+1\} \leftarrow \{FO_{IN}, I=1(1)N\}$
$\{FON_{I,N+2}, I=2(1)N+1\} \leftarrow \{FO_{I1}, I=1(1)N\}$
$FON_{1,N+2} \leftarrow FON_{12}, FO_{N+2,1} \leftarrow FON_{21}$
$\{\{FTIL_{IJ}, J=1(1)N\}, I=1(1)N\} \leftarrow$
$(3/2) \times FON_{I+1,J+1} - 1/12 \times (FON_{I+1,J} + FON_{I+1,J+2} + FON_{I,J+1}$
$+ FON_{I,J+2} + FON_{I+2,J} + FON_{I+2,J+2})$
$\{\{\{\{Q_{IJKL}, L=1(1)15\}, K=1(1)2\}, J=1(1)4\}, I=1(1)4\}$
EXECUTE
CONVOLP(FTIL,Q,N)
$\{\{\{\{SP_{IJKL}, L=1(1)15\}, K=1(1)2\}, J=1(1)N\}, I=1(1)N\}$
STOP

Fig. 5.2

5.5 Extrapolation

It became clear from the discussion in sections 5.2 and 5.3 that the construction of the quasi-interpolant of f on Ω requires values of $f^{\sim}$ on the boundary lattice points and lattice points outside Ω. If f is not periodic these values of $f^{\sim}$ are not available exactly. In this section we seek an approximate solution of this problem by extrapolation.

Let Ω be a finite domain and T_h a regular triangulation of it. Let also the function $f \in C^n(\Omega)$, $n \geq 4$. Motivated by Section 5.2 we will seek a linear operator G with the following property:

(i) G will be exact for polynomials of degree three; i.e. $(Gf)(y) = f^{\sim}(y)$, $y \in L_T$ when $f \in P_3$.

We will also require that G is local.

An existence theorem for extrapolating operators of bivariate functions $f \in C^m(\Omega)$ was proved by P. Frederickson [19]. Here we will make use of a ten point scheme A, B, C, D, E, F, H, K, L and M to entrapolate $f^{\sim}$ at the points A, B, C, D, P, Q, R, etc. [see Fig. 5.3]. This triangular scheme fits very well our quasi-interpolation problem on rectangular domains. Obviously, it fits it equally well on triangular, hexagonal, or L-shaped domains.

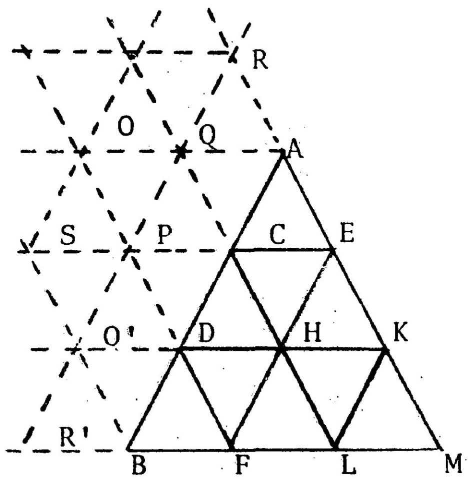


Fig. 5.3

Starting with the point P we define the extrapolated value of $f^{\sim}$ at P by

$$(Gf)(P) = \mu_1 f(A) + \mu_2 f(B) + \mu_3 f(C) + \mu_4 f(D) + \mu_5 f(E) + \mu_6 f(F) + \mu_7 f(H) + \mu_8 f(K) + \mu_9 f(L) + \mu_{10} f(M) \tag{5.5.1}$$

with the functionals μ_i, $i=1, 2, \ldots, 10$ to be evaluated so that property (i) would be satisfied. Assume (for ease of computation) that P is the origin, and f is the polynomial (5.2.6). From property (i) and equation (5.2.10) we have

$$(Gp)(P) = p^{\sim}(P) = 3/2p(P) - 1/12(p(Q)+p(C)+p(D)+p(Q')+p(S)+p(O))$$

Hence (5.5.1) becomes

$$\begin{aligned}(Gp)(0,0) &= 3/2p(0,0)-1/12(p(0,1)+p(1,0)+p(1,-1)+p(0,-1)+p(-1,0)+p(-1,1)) \\ &= \mu_1 p(1,1) + \mu_2 p(1,-2) + \cdots + \mu_{10} p(4,-2).\end{aligned}$$

Equating corresponding coefficients a_i, $i=0, 1, \ldots, 9$ in this identity we obtain the following system

$$M\mu_p = b_p \tag{5.5.2}$$

where $\mu_p = (\mu_1, \mu_2, \ldots, \mu_{10})$, $b_p = (1, 0, 0, -1/3, 1/6, -1/3, 0, 0, 0, 0)$ and

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 2 & 2 & 2 & 3 & 3 & 4 \\ 1 & -2 & 0 & -1 & 0 & -2 & -1 & -1 & -2 & -2 \\ 1 & 1 & 1 & 1 & 4 & 4 & 4 & 9 & 9 & 16 \\ 1 & -2 & 0 & -1 & 0 & -4 & -2 & -3 & -6 & -8 \\ 1 & 4 & 0 & 1 & 0 & 4 & 1 & 1 & 4 & 4 \\ 1 & 1 & 1 & 1 & 8 & 8 & 8 & 27 & 27 & 64 \\ 1 & -2 & 0 & -1 & 0 & -8 & -4 & -9 & -18 & -32 \\ 1 & 4 & 0 & 1 & 0 & 8 & 2 & 3 & 12 & 16 \\ 1 & -8 & 0 & -1 & 0 & -8 & -1 & -1 & -8 & -8 \end{bmatrix}$$

The solution of the system (5.5.2) is

$$\mu_P = (-\frac{1}{6}, -\frac{1}{6}, \frac{23}{12}, \frac{23}{12}, -\frac{7}{12}, -\frac{7}{12}, -\frac{7}{12}, \frac{17}{12}, \frac{17}{12}, -\frac{2}{3}) \qquad (5.5.3)$$

All the above computations were carried out in APL/360. The polynomials

$$p(i,j) = a_0+a_1i+a_2j+a_3i^2+a_4ij+a_5j^2+a_6i^3+a_7i^2j+a_8ij^2+a_9j^3$$

are easily manipulated when they are given as the product of the vectors

$$A = (a_0,a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8,a_9)$$

and $V_{i,j}$ as in (3.1.12).

Now in order to obtain μ_Q, μ_R, μ_A, etc. we need the corresponding vectors b_Q, b_R, b_A, etc. But we have

$$b_Q = \frac{3}{2} V_{1,0} - \frac{1}{12} (V_{0,0}+V_{0,1}+V_{1,1}+V_{1,0}+V_{1,-1}+V_{0,-1})$$

and similarly for the others. Consequently

$$\mu_Q = M^{-1} b_Q .$$

The values of μ_i, $i=1, 2, \ldots, 10$ we obtained for these points are

$$\mu_Q = (\frac{2}{3}, 0, \frac{13}{4}, -\frac{5}{12}, -\frac{29}{12}, \frac{1}{4}, -\frac{5}{2}, \frac{9}{4}, \frac{7}{12}, -\frac{2}{3}) \qquad (5.5.4)$$

$$\mu_R = (\frac{7}{2}, \frac{1}{6}, \frac{7}{12}, -\frac{3}{4}, -\frac{21}{4}, \frac{1}{12}, \frac{1}{2}, \frac{37}{12}, -\frac{1}{4}, -\frac{2}{3}) \qquad (5.5.5)$$

$$\mu_A = (\frac{2}{3}, \frac{1}{6}, \frac{5}{12}, -\frac{7}{12}, \frac{5}{12}, -\frac{1}{12}, \frac{1}{2}, -\frac{7}{12}, -\frac{1}{12}, \frac{1}{6}) \qquad (5.5.6)$$

$$\mu_C = (-\frac{1}{6}, 0, \frac{13}{12}, -\frac{1}{4}, \frac{1}{4}, \frac{1}{12}, \frac{1}{2}, -\frac{5}{12}, -\frac{1}{4}, \frac{1}{6}) \qquad (5.5.7)$$

By symmetry now we get $\mu_{Q'}$, $\mu_{R'}$, μ_B and μ_D. We have for example

$$\mu_B = (\frac{1}{6}, \frac{2}{3}, -\frac{7}{12}, \frac{5}{12}, -\frac{1}{12}, \frac{5}{12}, \frac{1}{2}, -\frac{1}{12}, -\frac{7}{12}, \frac{1}{6})$$

having always in mind the point at which each of the functionals μ_i operates.

The order of approximation of the above extrapolation schemes is 4, when the function f is four times differentiable on a domain Ω' such that $\Omega \subset \Omega'$ and the point at which we extrapolate $\tilde{f}$ is in Ω'. This comes from the fact that G is exact for polynomials of degree three.

5.6 Non-periodic Quasi-interpolation. Algorithm and Numerical Results

In this section we construct and carry out an algorithm for quasi-interpolation on a rectangular domain when the function under consideration is not (necessarily) periodic. The results of section 5.4 are applied for the evaluation of $f^{\sim}$.

We will consider again the rectangular domain Ω, $T\alpha$ regular triangulation of Ω such that the number of the lattice points L_T in Ω is $m \times n$ (with a uniform mesh in both directions). We will assume $f \varepsilon C^n(R^2)$, $n \geq 4$ and f° given on the above $m \times n$ lattice points of Ω. In Fig. 5.4 we illustrate Ω as the

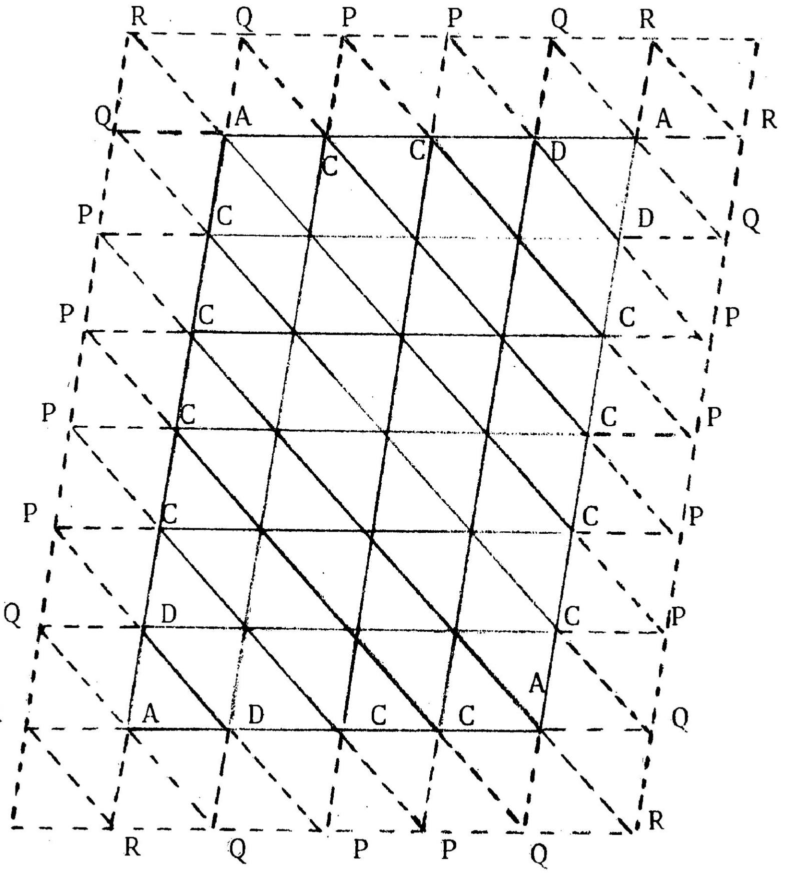


Fig. 5.4

parallelogram with the unbroken lines. We will refer to the extended parallelogram by Ω'.

In order to construct the spline quasi-interpolant Sf of f on Ω we need $\tilde{f}$ at all lattice points of Ω' (except the lower left corner and the upper right corner). Using formula (5.2.10) we can evaluate it at all inside lattice points of Ω. For the boundary points of Ω and the outside ones we will make use of the extrapolating schemes described in 5.5. For all the points labeled by P in Fig. 5.4 we use the scheme found for the point P in Fig. 5.3 with coefficients (5.5.3). For the points denoted by C we can use either the C-scheme or the D-scheme of Fig. 5.3. We consider the average of these two and obtain a twelve point scheme (see Fig. 5.5) with coefficients in the same order as the points in the Fig. 5.5.

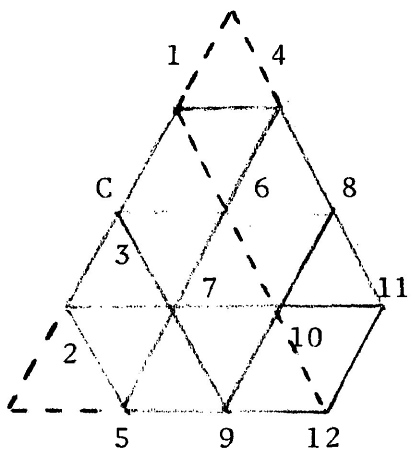


Fig. 5.5

$$\left(-\frac{5}{24}, -\frac{5}{24}, \frac{26}{24}, \frac{1}{24}, \frac{1}{24}, \frac{9}{24}, \frac{9}{24}, -\frac{3}{24}, -\frac{3}{24}, -\frac{10}{24}, \frac{2}{24}, \frac{2}{24}\right) \qquad (5.6.1)$$

There are still 7 points in the lower left and upper right corner respectively, and 4 points in each one of the other two corners at which $f^{\sim}$ should be evaluated at. We labeled them with the corresponding letter scheme of Fig. 5.3.

In Appendix IV we list the subroutine EQUASI which carries out all the above evaluation of $f^{\sim}$. The array FU = f° and the dimensions IF = m, JF = n are passed to the subroutine and the array FT = $f^{\sim}$ with dimensions IFT = IF + 2, JFT = JF + 2 is returned. First $f^{\sim}$ is evaluated on the inside points. Then specifying appropriate parameters it evaluates, using the same program segment, the P and C points of the two leftmost and rightmost columns. The same is done for the lower two and upper two rows, for the two 7 point corners and for the two 4 point corners. We would like to mention at this point that when subroutine EQUASI is called the array FT should be initialized to 0. This is because the subroutine was made to serve an iterative scheme we are going to describe in Chapter VI. In Appendix V we list the subroutine CONVOL which carries out the convolution of $f^{\sim}$ with Q in the non-periodic case described in this section. Q and FTL = $f^{\sim}$ (an $(m+1)\times(n+1)$ array) as evaluated by EQUASI together with the dimensions of the spline ID = m - 1, JD = n - 1 are passed to the subroutine. It returns the spline SPLN as an

ID × JD × 2 × 15 array. In our experiments we used the functions

$$f(x,y) = (1-\cos 2\pi x)\ (1-\cos 2\pi y) \tag{5.5.2}$$

and

$$f(x,y) = e^{x+y} \tag{5.5.3}$$

In Table 5.2 respectively 5.3 we give the norms L_1, L_2 and L_∞ of the error between the function (5.5.2) respectively (5.5.3) and its quasi-interpolant for two pairs of (m,n). The error is computed at the $(4\times m) \times (4\times n)$ points $(\frac{k}{4m}, \frac{\ell}{4n})$, $0 \le k \le 4m - 1$. $0 \le \ell \le 4n - 1$.

Table 5.2

$L_p \backslash {}^m n$	m = 7, n = 9	m = 15, n = 17
L_1	0.63000×10^{-5}	0.49587×10^{-6}
L_2	0.82900×10^{-5}	0.63170×10^{-6}
L_∞	0.35740×10^{-4}	0.25033×10^{-5}

Table 5.3

$L_p \backslash {}^m n$	m = 7, n = 9	m = 15, n = 17
L_2	0.21827×10^{-4}	0.15183×10^{-5}
L_1	0.24188×10^{-4}	0.16758×10^{-5}
L_∞	0.73887×10^{-4}	0.43018×10^{-5}

CHAPTER VI

REMARKS ON INTERPOLATION AND QUASI-INTERPOLATION

6.1 Relation of Quasi-Interpolation to Interpolation

The piecewise quartic interpolating spline as well as the quasi-interpolating one, were given by

$$S = Q * f^{\sim} \tag{6.1.1}$$

where Q is the normal basic Quartic spline and $f^{\sim}$ was the solution of $f^{\circ} = f^{\sim} * Q^{\circ}$ and $f^{\sim} = H(f^{\circ})$ respectively.

In Chapter 5 we noticed that the quasi-interpolation operator had two strong advantages over the interpolation one. The first of these is ease of computation: indeed, there is no large system of equations to be solved as the operator H is local. The second advantage is theoretical very strong error estimates are easy to obtain in the case of quasi-interpolation as was done in Section 5.3.

A natural question that arises now is the following: can one start with quasi-interpolation coefficients $f^{\sim}$ and improve to get the interpolation ones? Or at least get an approximation to them?

Let us consider the function $f^{\sim}$ (4.2.5) for interpolation

$$f^{\sim} = [f^{\circ\wedge}/(2Q^{\circ\wedge}/h^2\sqrt{3})]^{\vee} \tag{6.1.2}$$

and the corresponding one (5.2.8) for quasi-interpolation which we will denote by $f^{\sim\prime}$

$$f^{\sim\prime} = a * f^{\circ} \tag{6.1.3}$$

where the function a is 0 everywhere except

$$a(0) = \frac{3}{2} \quad \text{and} \quad a(\omega_s) = -\frac{1}{12}, \quad s = 1, 2, \ldots, 6$$

We also know that

$$Q(0) = \frac{1}{2} \quad \text{and} \quad Q(h\omega_s) = \frac{1}{12}, \quad s = 1, 2, \ldots, 6$$

and is zero everywhere else.

Consider now the discrete Laplacian ∇^2 given by

$$\nabla^2(0) = -4 \quad \text{and} \quad \nabla^2(h\omega_s) = \frac{2}{3}, \quad s = 1, 2, \ldots, 6 \tag{6.1.4}$$

and zero everywhere else.

It is clear that

$$Q^{\circ} = I + \frac{1}{8}\nabla^2 \tag{6.1.5}$$

and

$$a = I - \frac{1}{8}\nabla^2 \tag{6.1.6}$$

with I being the identity

If we set

$$E = \frac{1}{8}\nabla^2 \tag{6.1.7}$$

then (6.1.5) and (6.1.6) become

$$Q^\circ = I + E \quad \text{and} \quad a = I - E \tag{6.1.8}$$

Equation (6.1.3) now becomes

$$f^{\sim\prime} = f^\circ * (I-E) \tag{6.1.9}$$

and from equation 4.2.4, letting $k = \frac{\sqrt{3}h^2}{2}$ equation (6.1.2) may be written

$$\begin{aligned} f^\sim &= k[f^{\circ\wedge}/Q^{\circ\wedge}] \\ &= f^\circ * (1/Q^{\circ\wedge})^\vee \\ &= f^\circ * (1/(I+E^\wedge)^\vee \end{aligned} \tag{6.1.10}$$

Now, letting $E^n = E * E^{n-1}$, we have

$$(1/(I+E)^\wedge)^\vee = I - E + E^2 - E^3 + \ldots \tag{6.1.11}$$

Before we prove the validity of this expansion we notice that $f^{\sim\prime}$ is the convolution of f° with the first two terms of the above series and

$$\begin{aligned} f^\sim &= f^\circ * (I-E) + f^\circ * (E^2 - E^3 + \ldots) \\ &= f^{\sim\prime} + f^\circ * (E^2 - E^3 + \ldots) \end{aligned} \tag{6.1.12}$$

Another thing that becomes clear from (6.1.11) is the 3-exactness of the spline interpolant. Indeed, the contribution of the terms

$$E^n = \frac{1}{8^n} \nabla^{2n}, \quad n = 2, 3, \ldots$$

in (6.1.12), when f is a polynomial of degree three or less, will be zero.

The convergence of the series in (6.1.11) can be verified as follows: First we rewrite it

$$I - E + E^2 - E^3 + \dots = (I-E)*(I+E^2+E^4+\dots).$$

So, we need prove that the series

$$I + E^2 + E^4 + \dots$$

converges. We consider the term $E^2 = E*E$. The support of E^2 is a hexagon of side 2h, and

$$E^2(0) = \frac{21}{72}, \quad E^2(h\omega_s) = -\frac{5}{72}, \quad E^2(2h\omega_s) = \frac{1}{144},$$

while at the rest six lattice points of the hexagon E^2 takes the value of $\frac{1}{72}$.

Now if we define

$$|E^n| = \sum_{y\varepsilon L_T} |E^n(y)|$$

then

$$\begin{aligned}|E^{2n}| &= \sum_{y\varepsilon L_T} |E^{2n}(y)| \\ &= \sum_{y\varepsilon L_T} \Big| \sum_{z\varepsilon L_T} E^2(y)E^{2(n-1)}(y-z)\Big| \\ &\le |E^2|\cdot|E^{2(n-1)}|\end{aligned}$$

and

$$|E^2| = \frac{5}{6}.$$

Hence

$$\begin{aligned}|I+E^2+E^4+\dots| &\le |I| + |E^2| + |E^4| + \dots\\ &\le 1 + \frac{5}{6} + (\frac{5}{6})^2 + \dots\\ &= 6\end{aligned}$$

Obviously

$$|E^n|_\infty \le |E^n|$$

hence convergence of the series at any point $y \varepsilon L_T$ is also trivial.

Now in order to obtain the equality in (6.1.11) we first note that

$$I\hat{} = k$$

because $f^\circ\hat{} = (f^\circ * I)\hat{} = \frac{1}{k} \cdot f^\circ\hat{} \cdot I\hat{}$. Secondly, if we take the Fourier transform of the left hand side and multiply by $(I+E)\hat{}$ we get 1. Hence, starting with the right hand side we have

$$\begin{aligned}(I+E)\hat{}\cdot(I-E+E^2-E^3+\dots)\hat{} &= \frac{1}{k}\,[(I+E)*(I-E+E^2+\dots)]\hat{}\\ &= \frac{1}{k} \cdot I\hat{} = 1\end{aligned}$$

Hence the expansion (6.1.11) is valid.

6.2 An Iterative Scheme for Interpolation. Algorithm and Numerical Results.

Our numerical results in Chapter 5 show that quasi-interpolation is a fairly good approximation to interpolation. From (6.1.12) we notice that taking more terms in the remaining series and convoluting them with f° we will get an even better approximation to interpolation. Our object now will be to devise an iterative scheme suitable for computation. The expansion (6.1.11) may be written

$$\begin{aligned} I - E + E^2 - E^3 + \cdots &= (I-E)*(I+E^2+E^4+\cdots) \\ &= (I-E) + (I-E)*(E^2+E^3+\cdots) \\ &= (I-E) + (I-E)*E^2*(I+E^2+E^3+\cdots) \\ &= (I-E) + (I-E)*E^2 + (I-E)*E^2*E^2*(I+E^2+E^4+\cdots) \end{aligned}$$

Thus, the iterative scheme is

$$f^{\sim\prime}_{i+1} = f^{\sim\prime} + f^{\sim\prime}_i * E^2, \quad i = 1, 2, \ldots \tag{6.2.1}$$

with $f^{\sim\prime}_1 = f^{\sim\prime}$.

Moreover, since

$$Q^\circ * a = I - E^2 \quad \text{and} \quad f^\circ * a = f^{\sim\prime}$$

the right hand side of (6.2.1) takes the following form

$$\begin{aligned} f^{\sim\prime} + f^{\sim\prime}_i * E^2 &= f^\circ * a + f^{\sim\prime}_i - f^{\sim\prime}_i * Q^\circ * a \\ &= f^{\sim\prime}_i + (f^\circ - f^{\sim\prime}_i * Q^\circ) * a \\ &= f^{\sim\prime}_i + H(f^\circ - f^{\sim\prime}_i * Q^\circ) \end{aligned}$$

Hence, the iterative scheme may be written

$$f^{\sim\prime}_{i+1} = f^{\sim\prime}_{i} + H(e), \quad i = 1, 2, \ldots \tag{6.2.2}$$

with

$$e = f^{\circ} - f^{\sim\prime}_{i} * Q^{\circ} \tag{6.2.3}$$

and

$$f^{\sim\prime}_{1} = f^{\sim\prime}.$$

The implementation of this scheme is fairly straightforward. We consider the non-periodic case in order to incorporate the subroutine EQUASI described in section 5.5. Our iterative scheme is being carried out by the subroutine ITERAT displayed in Appendix VI. The $m \times n$ array f° is passed to ITERAT. This subroutine internally calls subroutine EQUASI to evaluate $f^{\sim\prime}$ ($(m+2)\times(n+2)$ array). $f^{\sim\prime}$ is convoluted with Q° and produces an $m \times n$ array which is subtracted from f° to give e according to (6.2.3). Subroutine EQUASI is called again to evaluate $H(e)$. The iteration continues as long as the number of iterations is less than prespecified number and the norm $||e||_{\infty}$ remains greater than some small number (10^{-8} was used in our experiments).

Using functions (5.5.2) and (5.5.3) we obtain the following tables 6.1 and 6.2, corresponding to tables 5.2 and 5.3, applying the above iterative scheme.

Table 6.1

$L_p \diagdown {}^m_n$	m = 7, 19 = 9	m = 15, n = 19
L_1	$0.24983 \cdot 10^{-5}$	0.10139×10^{-6}
L_2	$0.55870 \cdot 10^{-5}$	0.22747×10^{-6}
L_∞	$0.41858 \cdot 10^{-4}$	0.29026×10^{-5}

Table 6.2

$L_p \diagdown {}^m_n$	m = 7, n = 9	m = 15, n = 19
L_1	0.17767×10^{-5}	0.26304×10^{-6}
L_2	0.36487×10^{-5}	0.33179×10^{-6}
L_∞	0.59297×10^{-4}	0.26106×10^{-5}

Remark. The L_∞ norm in table 6.1 is greater than the corresponding one in table 5.2. There is a simple reason for that: As it was expected the largest error in table 5.2 (Norm L_∞) was at a point close to the boundary of Ω. Then carrying out the iterative scheme, $f_i^{\sim\prime}$ converges to $f^\sim$ on the internal lattice points of Ω where $f^{\sim\prime}$ is exact. However, the values of $f_i^{\sim\prime}$ on the rest of the lattice points are perturbed. This gave rise to a higher error on a boundary point (table 6.1). Another reason is the fact that for the function under consideration the seminorm $|f|_4$ is very large $|f|_4 > 1500$.

A P P E N D I X   I

In this appendix we list the APL function QUARTICSPL which gives Q (see section 3.2) on all of its support. QABC, QBCD and QBED are the vectors (3.3.1)'. In statement [4] another function GROUP is executed. This function creates the matrices ROT and REF, and we omit it. The four dimensions of Quartic run from 0-3, 0-3, 0-1 and 0-14. Following QUARTICSPL we display the whole Q.

```
                    ∇ QUARTICSPL

  [1]  QABC←6,0,0,(-12),(-12),8,12,12,8,(-1),(-2),0,(-2),(-1)
  [2]  QBCD←0,0,0,0,0,0,2,6,6,2,(-1),(-2),0,(-2),(-1)
  [3]  QBED←1,(-2),(-4),0,6,6,2,0,(-6),(-4),(-1),(-2),0,2,1
  [4]  GROUP
  [5]  QUARTIC← 4 4 2 15 ρ0
  [6]  QUARTIC[2;2;0;]←QABC
  [7]  QUARTIC[2;2;1;]←QBCD
  [8]  QUARTIC[2;3;0;]←QBED
  [9]  QUARTIC[2;1;0;]←QABC+.×ROT+.×REF
 [10]  QUARTIC[1;1;1;]←QABC
 [11]  QUARTIC[1;2;1;]←QABC+.×ROT+.×REF
 [12]  QUARTIC[1;2;0;]←QABC+.×REF+.×ROT
 [13]  QUARTIC[2;1;1;]←QABC+.×ROT
 [14]  QUARTIC[2;0;1;]←QBCD+.×ROT+.×REF
```

```
[15]   QUARTIC[2;0;1;]←QBCD+.×ROT+.×REF
[16]   QUARTIC[1;1;0;]←QBCD
[17]   QUARTIC[1;3;0;]←QBCD+.×ROT+.×REF
[18]   QUARTIC[3;1;0;]←QBCD+.×ROT
[19]   QUARTIC[3;0;0;]←QBED+.×REF+.×ROT+.×REF
[20]   QUARTIC[0;2;0;]←QBED+.×ROT
[21]   QUARTIC[3;2;0;]←QBED+.×REF
[22]   QUARTIC[2;0;0;]←QBED+.×ROT+.×REF
[23]   QUARTIC[0;3;0;]←QBED+.×REF+.×ROT
[24]   QUARTIC[1;3;1;]←QBED+.×ROT+.×REF
[25]   QUARTIC[3;1;1;]←QBED+.×ROT
[26]   QUARTIC[1;0;1;]←QBED
[27]   QUARTIC[0;3;1;]←QBED+.×REF+.×ROT+.×REF
[28]   QUARTIC[0;1;1;]←QBED+.×REF
[29]   QUARTIC[3;0;1;]←QBED+.×REF+.×ROT
     ∇
```

```
                      QUARTIC

 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0

 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 1 ¯4 ¯2  6  6  0 ¯4 ¯6  0  2  1  2  0 ¯2 ¯1
```

```
  0   0   0   0   0   0   0   0   0   0   1   2   0   0   0
  1  ¯2   2   0  ¯6   0   2   6   0  ¯4  ¯1  ¯2   0   4   2

  0   0   0   0   0   0   2   0   0   0  ¯1  ¯2   0   0   0
  0   0   0   0   0   0   0   0   0   2   0   0   0  ¯2  ¯1

  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  1  ¯2  ¯4   0   6   6   2   0  ¯6  ¯4  ¯1  ¯2   0   2   1

  0   0   0   0   0   0   2   6   6   2  ¯1  ¯2   0  ¯2  ¯1
  6   0   0 ¯12 ¯12 ¯12   8  12  12   8  ¯1  ¯2   0  ¯2  ¯1

  1   4   2   6   6   0  ¯4  ¯6 ¯12  ¯4  ¯1  ¯2   0   4   2
  1   2   4   0   6   6  ¯4 ¯12  ¯6  ¯4   2   4   0  ¯2  ¯1

  1   2  ¯2   0  ¯6   0  ¯4   0   6   2   2   4   0  ¯2  ¯1
  0   0   0   0   0   0   0   0   0   0   0   0   0   2   1

  0   0   0   0   0   0   0   0   0   0   0   0   0   2   1
  1   2  ¯2   0  ¯6   0  ¯4   0   6   2   2   4   0  ¯2  ¯1

  1   2   4   0   6   6  ¯4 ¯12  ¯6  ¯4   2   4   0  ¯2  ¯1
  1   4   2   6   6   0  ¯4  ¯6 ¯12  ¯4  ¯1  ¯2   0   4   2

  6   0   0 ¯12 ¯12 ¯12   8  12  12   8  ¯1  ¯2   0  ¯2  ¯1
  0   0   0   0   0   0   2   6   6   2  ¯1  ¯2   0  ¯2  ¯1

  1  ¯2  ¯4   0   6   6   2   0  ¯6  ¯4  ¯1  ¯2   0   2   1
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

```
 0   0   0   0   0   0   0   0   0   2   0   0   0  ¯2  ¯1
 0   0   0   0   0   0   2   0   0   0  ¯1  ¯2   0   0   0

 1  ¯2   2   0  ¯6   0   2   6   0  ¯4  ¯1  ¯2   0   4   2
 0   0   0   0   0   0   0   0   0   0   1   2   0   0   0

 1  ¯4  ¯2   6   6   0  ¯4  ¯6   0   2   1   2   0  ¯2  ¯1
 0   0   0   0   0   0   0   0   0   0   0   0   0   0   0

 0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

A P P E N D I X   I I

In this appendix we list the subroutine EVAL used for the evaluation of Quartic splines. A detailed description of this subroutine was given in Section 3.3.

```
SUBROUTINE EVAL(SPL,X.Y,SPXY,ID,JD)
IMPLICIT REAL*8 (A-H,O-Z),INTEGER*4 (I-N)
DIMENSION SPL(ID,JD,2,15)
A=X*ID
B=Y*JD
IJ=1
II=IDINT(A)
JJ=IDINT(B)
A2=A-DFLOAT(II)
B2=B-DFLOAT(JJ)
AB=A2+B2
IF (AB.LE.1DO) IJ=0
A1=DABS(DFLOAT(IJ)-A2)
B1=DABS(DFLOAT(IJ)-B2)
IF (II.LE.O) II=0
IF (II.GE.ID) II=ID-1
IF (JJ.LE.O) JJ=0
IF (JJ.GE.JD) JJ=JD-1
K=II+1
L=JJ+1
M=IJ+1
```

```
      SPXY=SPL(K,L,M,1)+B1*(SPL(K,L,M,3)+B1*(SPL(K,L,M,6)+
     B1*(SPL(K,L,M,10)+B1*SPL(K,L,M,15))))+A1*(SPL(K,L,M,2)+B1*
    *(SPL(K,L,M,5)+B1*(SPL(K,L,M,9)+B1*SPL(K,L,M,14)))+A1*(SPL
    *(K,L,M,4)+B1*(SPL(K,L,M,8)+B1*SPL(K.L,M,13))+A1*(SPL(K,L,
    *M,7)+B1*SPL(K,L,M,12)+A1*SPL(K,L,M,11))))
     RETURN
     END
```

A P P E N D I X   I I I

In this appendix we list subroutine CONVOLP used for periodic interpolation and quasi-interpolation. The description of this subroutine was given in Section 4.3.

```
      SUBROUTINE CONVOLP(SP,FTIL,Q,ID,JD,IFT,JFT)
      REAL*8 SP(ID,JD,2,15),FTIL(IFT,JFT)
      INTEGER Q(4,4,2,15)
      IID=ID+1
      JJD=JD+1
      DO 10 I=1,ID
      DO 10 J=1,JD
      DO 10 K=1,2
      DO 10 L=1,15
10    SP(I,J,K,L)=ODO
      DO 20 I=1,ID
      DO 20 J=1,JD
      DO 20 II=1,4
      DO 20 JJ=1,4
      IM=MOD(II-(I+2),IID)
      JM=MOD(JJ-(J+2),JJD)
      DO 20 K=1,2
      DO 20 L=1,15
20    SP(I,J,K,L)=(FTIL(IM,JM)*Q(II,JJ,K,L))/12+SP(I,J,K,L)
      RETURN
      END
```

A P P E N D I X   I V

In this appendix we list subroutine EQUASI, It was described almost satisfactorily in section 5.6.

```
      SUBROUTINE EQUASI(FU,FT,IF,JF,IFT,JFT)
      IMPLICIT REAL*8(A-H,O-Z),INTEGER*4(I-N)
      DIMENSION FU(IF,JF),FT(IFT,JFT)
      DO 310 I=3,IF
      DO 310 J=3,JF
  310 FT(I,J)=(18*FU(I-1,J-1)-(FU(I,J-1)+FU(I-1,J)+FU(I-2,J)+
     *FU(I-2,J-1)+FU(I-1,J-2)+FU(I,J-2)))/12+FT(I,J)
      K=1
      L=2
      M=3
      N=4
      II=3
      JA=1
      JB=2
      I1=0
      I2=1
      I4=3
      L1=3
      L2=IF-1
  320 DO 330 I=L1,L2
      I1=I1+1
      I2=I2+1
      I4=I4+1
      II=II+1
      FT(II,JB)=(-5*(FU(I2,K)+FU(I4,K))+26*FU(I,K)+FU(I4,L)+
     *FU(I1,L)+9*(FU(I,L)+FU(I2,L))-3*(FU(I,M)+FU(I1,M))-10*
     *FU(I2,M)+2*(FU(I2,N)+FU(I1,N)))/24+FT(II,JB)
  330 FT(II,JA)=(-2*(FU(II,K)+FU(I4,K))+23*(FU(I,K)+FU(I2,K))-
     *7*(FU(I,L)+FU(I1,L))-42*FU(I2,L)+17*(FU(I2,M)+FU(I1,M))-
     *8*FU(I1,N))/12+FT(II,JA)
      IF (K-1) 340,340,350
  340 K=JF
      L=JF-1
      M=JF-2
      N=JF-3
      II=2
      JA=JFT
      JB=JFT-1
      I1=3
      I2=2
      I4=0
      L1=2
      L2=IF-2
      GO TO 320
  350 K=1
      L=2
      M=3
      N=4
      JJ=3
      IA=1
      IB=2
      J1=0
      J2=1
      J4=3
      L1=3
      L2=JF-1
  360 DO 370 J=L1,L2
      J1=J1+1
      J4=J4+1
      JJ=JJ+1
      FT(IB,JJ)=(-5*(FU(K,J2)+FU(K,J4))+26*FU(K,J)+FU(L,J1   FU(L,
     *J4)+9*(FU(L,J2)+FU(L,J))-3*(FU(M,J)+FU(M,J1))-10*FU(M,J2)+2
     **(FU(N,J2)+FU(N,J1)))/24+FT(IB,JJ)
  370 FT(IA,JJ)=(-2*(FU(K,J1)+FU(K,J4))+23*(FU(K,J)+FU(K,J2))-7*
     *(FU(L,J)+FU(L,J1))-42*FU(L,J2)+17*(FU(M,J1)+FU(M,J2))-8*
     *FU(N,J1))/12+FT(IA,JJ)
      IF (K-1) 380,380,390
  380 K=IF
      L=IF-1
      M=IF-2
      N=IF-3
      JJ=2
      IA=IFT
      IB=IFT-1
      J1=3
      J2=2
      J4=0
      L1=2
      L2=JF-2
      GO TO 360
  390 K=1
      I1=1
      I2=2
      I3=3
      I4=4
      J1=1
      J2=2
      J3=3
      J4=4
      L=0
      M=2
  400 A=FU(I1,J1)
      B=FU(I1,J2)
      C=FU(I1,J3)
      D=FU(I1,J4)
      E=FU(I2,J1)
```

```
     F=FU(I2,J2)
     G=FU(I2,J3)
     H=FU(I3,J1)
     O=FU(I3,J2)
     P=FU(I4,J1)
     FT(I1+L,J1+M)=(8*A+39*B-5*C-29*E-30*F+3*G+27*H+7*O-8*P)/12+
    *FT(I1+L,J1+M)
     FT(I1+L,J1+K)=(42*A+7*B-9*C+2*D-63*E+6*F+G+37*H-3*O-8*P)/12+
    *FT(I1+L,J1+K)
     FT(I1+K,J1+M)=(-2*A+13*B-3*C+3*E+6*F+G-5*H-3*O+2*P)/12+FT
    *(I1+K,J1+M)
     FT(I1+K,J1+K)=(8*A+5*B-7*C+2*D+5*E+6*F-G-7*H-O+2*P)/12+FT
    *(I1+K,J1+K)
     FT(I1+K,J1+L)=(42*A-63*B+37*C-8*D+7*E+6*F-3*G-9*H+O+2*P)/12+
    *FT(I1+K,J1+L)
     FT(I1+M,J1+K)=(-2*A+3*B-5*C+2*D+13*E+6*F-3*(G+H)+O)/12+FT(I1+
    *M,J1+K)
     FT(I1+M,J1+L)=(8*A-29*B+27*C-8*D+39*E-30*F+7*G-5*H+3*O)/12+
    *FT(I1+M,J1+L)
     IF (M) 420,420,410
 410 I1=IF
     J1=JF
     I2=IF-1
     J2=JF-1
     I3=IF-2
```

```
      J3=JF-2
      I4=IF-3
      J4=JF-3
      M=0
      L=2
      GO TO 400
  420 I1=IF
      J1=1
      J2=2
      I2=IF-1
      I3=IF-2
      J3=3
      I4=IF-3
      J4=4
      L=0
      M=2
  430 A=FU(I1,J1)
      B=FU(I2,J1)
      C=FU(I3,J1)
      D=FU(I4,J1)
      E=FU(I2,J2)
      F=FU(I3,J2)
      G=FU(I4,J2)
      H=FU(I3,J3)
      O=FU(I4,J3)
      P=FU(I4,J4)
      Q=FU(I1,J2)
      R=FU(I1,J3)
      S=FU(I1,J4)
      T=FU(I2,J3)
      U=FU(I2,J4)
      V=FU(I3,J4)
      FT(I1+K,J1+K)=(16*A+5*(B+Q)+10*E-7*(C+R)+6*(F+T)-14*H+2*(
     *D+S)+4*P-(G+O+U+V))/24+FT(I1+K,J1+K)
      FT(I1+K,J1+L)=(8*A+39*B-5*C-29*E-30*F+3*G+27*H+7*O-8*P)/12+
     *FT(I1+K,J1+L)
      FT(I1+M,J1+K)=(8*A+39*Q-5*R-29*E-30*T+3*U+27*H+7*V-8*P)/12+
     *FT(I1+M,J1+K)
      FT(I1+M,J1+L)=(84*A+7*(B+Q)-126*E-9*(C+R)+6*(F+T)+74*H+2*
     *(D+S)+G+U-3*(O+V)-16*P)/24+FT(I1+M,J1+L)
      IF (M) 450,450,440
  440 I1=1
     J1=JF
     I2=2
     J2=JF-1
     I3=3
     J3=JF-2
     I4=4
     J4=JF-3
     L=2
```

```
      M=0
      GO TO 430
  450 FT(1,1)=ODO
      FT(IFT,JFT)=ODO
      RETURN
      END
```

A P P E N D I X   V

Here we list the version of the CONVOL subroutine which we used in the non-periodic case of interpolation and quasi-interpolation. This is much more polished than the one we listed in Appendix III. It is easier to program convolution thinking of Q as having rectangular support. However, this introduces unnecessary computation which we avoided here.

```
      SUBROUTINE CONVOL(SPLN,FTL,IQ,ID,JD,IFT,JFT)
      IMPLICIT REAL*8(A-H,O-Z),INTEGER*4(I-N)
      DIMENSION SPLN(ID,JD,2,15),FTL(IFT,JFT),IQ(4,4,2,15)
      DO 570 K=1,2
      DO 570 IB=1,4
      JL=MAX0(1,5-(IB+K))
      JU=MIN0(4,8-(IB+K))
      DO 570 JB=JL,JU
      IF ((IB.EQ.1).AND.(JB.EQ.JL)) GO TO 530
  500 IF ((IB.EQ.4).AND.(JB.EQ.JU)) GO TO 550
  510 DO 520 I=1,ID
      DO 520 J=1,JD
      ITL=I+4-IB
      JTL=J+4-JB
      DO 520 L=1,15
  520 SPLN(I,J,K,L)=FTL(ITL,JTL)*IQ(IB,JB,K,L)+SPLN(I,J,K,L)
      GO TO 570
  530 DO 540 I=1,ID
```

```
      ITL=I+4-IB
      DO 540 J=1,JD
      JTL=J+4-JB
      DO 540 L=1,15
  540 SPLN(I,J,K,L)=FTL(ITL,JTL)*IQ(IB,JB,K,L)
      GO TO 570
  550 DO 560 I=1,ID
      ITL=I+4-IB
      DO 560 J=1,JD
      JTL=J+4-JB
      DO 560 L=1,15
  560 SPLN(I,J,K,L)=(FTL(ITL,JTL)*IQ(IB,JB,K,L)+SPLN(I,J,K,L))/12
  570 CONTINUE
      RETURN
      END
```

A P P E N D I X V I

In this appendix we list subroutine ITERAT. The input of this subroutine is FI = f° with its dimensions IF and JF. FTI = $f^\sim$ with dimensions IFT and JFT is returned. Subroutine EQUASI is called once to give the initial $f^{\sim\prime}$. If the number of iterations ITN is specified to be 0, $f^{\sim\prime}$ is returned. Otherwise using ERR = e as was described in 6.2 we iterate to get a better approximation to $f^\sim$.

```
      SUBROUTINE ITERAT(FI,FTI,ERR,IF,JF,IFT,JFT,ITN)
      IMPLICIT REAL*8(A-H,O-Z),INTEGER*4(I-N)
      DIMENSION FI(IF,JF),FTI(IFT,JFT),ERR(IF,JF)
      K=0
      CALL EQUASI(FI,FTI,IF,JF,IFT,JFT)
  100 K=K+1
      IF (ITN-K) 140,110,110
  110 DO 120 I=1,IF
      DO 120 J=1,JF
  120 ERR(I,J)=FI(I,J)-(6*FTI(I+1,J+1)+FTI(I+1,J+2)+FTI(I,J+2)+
      FTI(I,J+1)+FTI(I+1,J)+FTI(I+2,J+1))/12
      AINF=ODO
      DO 130 I=1,IF
      DO 130 J=1,JF
      E=DABS(ERR(I,J))
  130 AINF=DMAX1(AINF,E)
```

```
      IF (AINF.LE.1D-8) GO TO 140
      CALL EQUASI(ERR,FTI,IF,JF,IFT,JFT)
      GO TO 100
  140 RETURN
      END
```

BIBLIOGRAPHY

[1] AHLBERG, J. H., NILSON, E. N., WALSH, J. L. "The Theory of Splines and Their Applications". Academic Press 1967.

[2] BABUSKA, I. "Approximation by Hill Functions". Comm. Math. Univ. Carolinae 11, 4 (1970).

[3] BABUSKA, I. "A Remark to the Finite Element Method". Comm. Math. Univ. Carolinae 12, 2 (1971) 367-375.

[4] BABUSKA, I. "The Finite Element Method for Infinite Domains". Math. Comp. 26, 117 (1972) 1-11.

[5] BIRKHOFF, G., GARABEDIAN, H. "Smooth Surface Interpolation". J. Math. Phys., 39 (1960), 353-68.

[6] BIRKHOFF, G., BARNHILL, R. E., GORDON, W. J. "Smooth Interpolation in Triangles". GMR-1064, February 9, 1971.

[7] BIRKHOFF, G., GORDON, W. J. "The Draftsman's and Related Equations". J. of Approx. Theory 1 (1968) 199-208.

[8] BIRKHOFF, G., SCHULTZ, M. H., VARGA, R. S. "Piecewise Hermite Interpolation in One and Two Variables with Applications to Partial Differential Equations". Numer. Math., 11 (1968), 232-256.

[9] deBOOR, C. "Bicubic Spline Interpolation". J. Math. Phys., 41 (1962), 212-218.

[10] deBOOR, C., FIX, G. J. "Spline Interpolation by Quasi-interpolants". J. Approx. Theory-to appear.

[11] CARLSON, R. E., HALL, C. A. "On Piecewise Polynomial Interpolation in Rectangular Polygons". J. Appr. Theory, 4 (1971) 37-53.

[12] CARLSON, R. E., HALL, C. A. "Bicubic Spline Interpolation and Approximation in Right Triangles". J. Appr. Theory, to appear.

[13] CARLSON, R. E., HALL, C. A. "Bicubic Spline interpolation in L-shaped Domains". J. Appr. Theory, to appear.

[14] CHENEY, E. W. "Introduction to Approximation Theory". McGraw-Hill Book Company, New York 1966.

[15] COOLEY, J. W., TUKEY, J. W. "An Algorithm for the Machine Calculation of Complex Fourier Series". Math. Comput. 19, 90 (1965), 297-301.

[16] DAVIS, P. J. "Interpolation and Approximation". Blaisdell Publishing Company, New York 1965.

[17] FERGUSON, J. C. "Multivariable Curve Interpolation". J. Assoc. Comp. Math., 11 (1964), 221-228.

[18] FREDERICKSON, P. O. "Generalized Triangular Splines". Technical Report, Lakehead University, 1970.

[19] FREDERICKSON, P. O. "Quasi-interpolation, extrapolation, and Approximation on the Plane". Proc. of the Manitoba Conference on Numer. Analysis. October 7-9, 1971.

[20] GORDON, W. J. "Free-form Surface Interpolation Through Curve Networks". CMR-921, September 1969.

[21] GORDON, W. J. "Spline-Blended Surface Interpolation Through Curve Networks". J. Math. Mech., 18 (1969), 931-952.

[22] GORDON, W. J. "Bivariate interpolation through curve Networks". Notices of AMS, 15 (1968).

[23] GORDON, W. J. "Blending-function Methods of Bivariate and Multivariate Interpolation and Approximation". SIAM J. Numer. Anal., 8 (1971), 158-177.

[24] GREVILLE, T. N. E. "Theory and Applications of Spline Functions". Academic Press, New York 1969.

[25] HALL, C. A. "Bicubic Interpolation over Triangles". J. Math. Mech., 19 (1969), 1-11.

[26] HILLE, E. "Analysis". Blaisdell Publishing Company 1966.

[27] ISAACSON, E., KELLER, H. B. "Analysis of Numerical Methods". John Wiley & Sons, Inc., New York 1966.

[28] LORENTZ, G. G. "Approximation of Functions". Holt, Rinehart and Winston". New York 1966.

[29] MEINARDUS, G. "Approximation of Functions: Theory and Numerical Methods". Springer-Verlag New York Inc. 1967.

[30] RALSTON, A. "A First Course in Numerical Analysis". McGraw Hill Book Company.

[31] RICE, J. R. "The Approximation of Functions". Volume 1, 2. Addison-Wesley Publishing Company, Inc., 1969.

[32] RICE, J. T. "Tchebycheff Approximation in Several Variables". Trans. Amer. Math. Soc., 109, pp. 444-466.

[33] SCHOENBERG, I. J. "Approximations with Special Emphasis on Spline Functions". Academic Press 1969.

[34] SINGLETON, R. C. "Algol Procedures for the Fast Fourier Transform". Comm. ACM 11 (Nov. 1968), 773-777.

[35] STANCU, D. D. "The Remainder of Certain Linear Approximation Formulas in Two Variables". J. SIAM Numer. Anal. Ser. B. 1 (1964) 137-163.

[36] STEFFENSEN, J. F. "Interpolation". Chelsea Publishing Company New York, 1950.

[37] ZENISEK, A. "Interpolation Polynomials on the Triangle". Numer. Math. 15 (1970) 283-296.

[38] ZLAMAL, M. "On the Finite Element Method". Numer. Math. 12 (1968) 394-409.

[39] ZLAMAL, M. "A Finite Element Procedure of the Second Order of Accuracy". Numer. Math. 14 (1970) 394-402.