A cubic and bicubic method for curve and surface estimation

Niki M Anagnostopoulou

University of Nevada, Las Vegas

Follow this and additional works at: https://digitalscholarship.unlv.edu/rtds

Repository Citation
https://digitalscholarship.unlv.edu/rtds/341

This Thesis is brought to you for free and open access by Digital Scholarship@UNLV. It has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI
University Microfilms International
A Bell & Howell Information Company
300 North Zeeb Road. Ann Arbor, MI 48106-1346 USA
313/761-4700  800/521-0600
A cubic and bicubic method for curve and surface estimation

Anagnostopoulou, Niki M., M.S.

University of Nevada, Las Vegas, 1994

Copyright ©1994 by Anagnostopoulou, Niki M. All rights reserved.
A CUBIC AND BICUBIC
METHOD FOR CURVE AND
SURFACE ESTIMATION

NIKI M. ANAGNOSTOPOULOU

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Mathematics
Department of Mathematical Sciences
University of Nevada, Las Vegas
May 1994
The Thesis of Niki M. Anagnostopoulou for the degree of Master of Science in Mathematics is approved.

Ashok K. Singh, Ph.D
Chairperson

Rohan J. Dalpatadu, Ph.D
Examinig Committee Member

Xin Li, Ph.D
Examinig Committee Member

Evangelos A. Yfantis
Graduate Faculty Representative

April 20/94

Dean of the Graduate College, Ronald W. Smith. Ph.D

University of Nevada. Las Vegas

May 1994
Abstract

A deterministic cubic and bicubic method for curve and surface estimation is presented. The method presented, does not assume that the surface to be estimated, based on a given set of data in the three dimensional space, has a continuous first or second derivatives. The given data does not have to be equidistant. Also, since the method leads to parametric equations for the patches of the surface, the estimating surface does not need to be a function. Parameters $h_{i,j}$ are used for continuity and stress.
# Tables of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>v</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>vii</td>
</tr>
<tr>
<td>CHAPTER 1</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Curve estimation</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Surface estimation</td>
<td>18</td>
</tr>
<tr>
<td>CHAPTER 2 Examples and Error discussion</td>
<td>33</td>
</tr>
<tr>
<td>2.1 First example</td>
<td>33</td>
</tr>
<tr>
<td>2.2 Second example</td>
<td>38</td>
</tr>
<tr>
<td>APPENDIX Functions programmed in C</td>
<td>42</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>53</td>
</tr>
</tbody>
</table>
List of Figures

FIGURE 1, Graph of $f_1 = 0.5 + 0.0000562(x^3 + y^3 + x^2y + y^2x)$, page 35

FIGURE 2, First estimation of $f_1$ without continuity requirements, page 35

FIGURE 3, Second estimation of $f_1$ with continuity requirements, page 35

FIGURE 4, Error between real function $f_1$ and first estimation, page 36

FIGURE 5, Error between real function $f_1$ and second estimation, page 36

FIGURE 6, Graph of $f_2 = 1 - 0.0008(x^2 + y^2 + xy)$, page 39

FIGURE 7, First estimation of $f_2$ without continuity requirements, page 39

FIGURE 8, Error between real function $f_2$ and first estimation, page 39

FIGURE 9, Zoom in estimation without continuity requirements, page 40
Acknowledgments

I would like to thank my Supervisor, Dr Ashok K. Singh, for his advises, and the Examining Committee Members, Dr. Rohan J. Dalpatadu, Dr. Xin Li and Dr. Evangelos A. Yfantis for their corrections and recommendations.

I would like also to thank Dr. Z. Psillakis for his encouragement and his support.
Chapter 1

1.1 Introduction

Over the past twenty-five years many interpolation algorithms, algorithms producing curves or surfaces passing through the data, for curve and surface estimation have been developed. These algorithms can be classified into three categories: Deterministic, Stochastic, and Chaotic.

The deterministic algorithms can be found in the literature under the name Splines [Barnhill and Riesenfeld '74], [Barnhill '85], [Barnhill and Boehm '83], [Barsky '88], [Akima '70], [Bartels et al '87], [Beck et al '86], [Coons '74], [de Boor '87], [de Casteljau '86], [Farin '83 '90], [Foley '86 '87 '87], [Lane '88], [Nielson '86 '87], [Salkauskas '84], [Sapidis '87], [Schumaker '81], [Shirman and Sequin '90], [Gregory '74], [Yfantis '93] and others. The deterministic interpolation algorithms for surface estimation require that the domain of the input data points constitutes a rectangular grid, and all the grids have to be equal. Also biquadratic, surface estimation algorithms require continuity of the first derivative, and bicubic splines require continuity of the second derivative.

Kriging requires that the data is sampled from a random process which is wide sense stationary or satisfies the intrinsic hypothesis, or if there is a trend, the residuals after removing the trend are second order stationary. Interpolation algorithms based on the theory of chaos give unpredictable answers, the generated curve or surface is continuous but has derivative nowhere, and due to their random nature every time
the method is applied to the same set of data it gives different results.

Another family of interpolation algorithms are the Stochastic interpolators. The stochastic interpolation algorithms minimize the mean square error, and assume the process the data come from is stationary, or it satisfies the intrinsic hypothesis. Stationarity, or satisfaction of the intrinsic hypothesis by the process the data come from is impossible to be proven. The stochastic algorithms can be found in the literature usually under the name Kriging [Journel '89], [David '77], [Olea '77], [Matheron '76], [Borgman and Frahme '76], [Yfantis et al. '87], [Journel and Huijbregts '78], [Armstrong '84], [Myers '82], [Davis '88], [Delfiner '76], [Englund '90], and others. The third class of interpolators, namely the ones based on the theory of chaos and fractals was developed recently [Carpenter '86], [Yfantis et al. '88] and others. Hybrid methods using stochastic and deterministic methods [Yfantis et al., 1992], or any combination of the above methods.

The method presented here, leads to a deterministic interpolation algorithm for curve and surface estimation, by extending the Bezier curve and surface estimation method.

In the Bezier curve and surface estimation method, in order to reproduce the shape of a curve, a control polygon which is defined by:

\[ b_i^r(t) = (1 - t)b_i^{r-1}(t) + tb_{i+1}^{r-1}(t) \]

where \( r = 1, \ldots, n \) and \( i = 1, \ldots, n-r \), given \( (b_0, \ldots, b_{n-1}) \) points of
$R^3$, and

$$b_i^r(0) = b_i,$$

is used, that somehow "exaggerates" the shape of the curve. In our method, we use a third degree Bezier curve, which passes through the end points $b_i$ and $b_{i+3}$, and in addition we require this curve to pass through the internal given points $b_{i+1}$ and $b_{i+2}$. Moreover, this method uses parameters $h_{i,j}$, which are the solution of a linear $m \times m$ system, giving continuity and stress.

However, the estimation does not require the solution of a linear system, as splines and kriging do, by choosing appropriate values of $h_{i,j}$, for better results. The method presented, does not assume that the surface to be estimated, based on a given set of data in the three-dimensional space, has a continuous first or second derivatives, nor does it assume that the data satisfy the assumption of stationary or the intrinsic hypothesis.

The grid formed by the given data does not have to be equidistant. Also, since the method leads to parametric equations for the patches of the surface, the estimating surface does not need to be a function.
1.2 Curve estimation

Let \((x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_n, y_n, z_n)\) be \(n\) given points in the three dimensional space. We would like to find a curve passing through these points. The curve consists of \((n-1)\) cubic patches. The \(i\)th cubic patch connects the points \((x_i, y_i, z_i)\) and \((x_{i+1}, y_{i+1}, z_{i+1})\). The \(i\)th cubic patch has continuous first and second derivatives for \(0 < t < 1\), but the curve is not necessarily \(C^1\), or \(C^2\), i.e. our algorithm allows for patches that have discontinuous first or second derivatives at the knots or joints. The way we calculate the patches is by calculating first a cubic parametric equation passing through the \((i-2), (i-1), i, \) and \((i+1)\) points; second we calculate a parametric cubic equation passing through the points \((i-1), i, (i+1), \) and \((i+2)\); third we calculate a cubic parametric equation passing through \(i, (i+1), (i+2), \) and \((i+3)\); fourth we consider the common part which extends between \(i, \) and \((i+1)\) and we express that as an average of the first the second and the third parts. This kind of approach allows us to introduce parameters to allow for the estimated curve to be \(C^1\) or not to be \(C^2\). Also it enables us to introduce tension. We calculate the cubic equation passing through the \(i, (i+1), (i+2), \) and \((i+3)\) points by introducing two new points \((a_1, b_1, c_1)\) and \((a_2, b_2, c_2)\) and use the \(i\) point, the two new points and the \((i+3)\) point to calculate a parametric cubic curve. This curve passes through the \(i, \) and \((i+3)\) points since they are end points, but not through \((a_1, b_1, c_1)\) and \((a_2, b_2, c_2)\). The points \((a_1, b_1, c_1)\) and \((a_2, b_2, c_2)\) are selected so that for \(t = 1/3\) the parametric equation passes through the \((i+1)\) point and for \(t = 2/3\) the parametric equation passes through the
(i+2) point. Thus if \( X_i(t) \) denotes the parametric equation of the x-coordinate of the cubic Bezier defined by \( x_i, a_1, a_2, x_{i+3} \), then

\[
X_i(t) = (1 - t)^3 x_i + 3t(1 - t)^2 a_1 + 3t^2(1 - t)a_2 + t^3 x_{i+3},
\]

\[0 \leq t \leq 1, 1 \leq i \leq n - 3, a_1, a_2 \in R\]  

(1)

the parameters \( a_1, a_2 \in R \) are determined so that \( X_i(1/3) = x_{i+1} \) and \( X_i(2/3) = x_{i+2} \). Therefore from equation (1) we obtain:

\[
\begin{align*}
27x_{i+1} &= 8x_i + 12a_1 + 6a_2 + x_{i+3}, \\
27x_{i+2} &= x_i + 6a_1 + 12a_2 + 8x_{i+3},
\end{align*}
\]

\[1 \leq i \leq n - 3, a_1, a_2 \in R\]  

(2)

which is a system of two equations in two unknowns, \( a_1 \) and \( a_2 \). After solving system (2) for \( a_1 \) and \( a_2 \), we find

\[
a_1 = \frac{-5x_i + 18x_{i+1} - 9x_{i+2} + 2x_{i+3}}{6},
\]

and

\[
a_2 = \frac{2x_i - 9x_{i+1} + 18x_{i+2} - 5x_{i+3}}{6}
\]

Now, we substitute back to the original equation (1), and we obtain the Lagrange cubic form:

\[
X_i(t) = (1 - t)^3 x_i + \frac{(-5x_i + 18x_{i+1} - 9x_{i+2} + 2x_{i+3})t(1 - t)^2}{2} + \frac{(2x_i - 9x_{i+1} + 18x_{i+2} - 5x_{i+3})t^2(1 - t)}{2} + t^3 x_{i+3},
\]

\[1 \leq i \leq n - 3, 0 \leq t \leq 1\]
or

\[ X_i(t) = \frac{(1 - t)(3t - 1)(3t - 2)}{2} x_i + \frac{9t(1 - t)(2 - 3t)}{2} x_{i+1} + \]
\[ + \frac{9t(1 - t)(3t - 1)}{2} x_{i+2} + \frac{t(3t - 1)(3t - 2)}{2} x_{i+3}, \]
\[ 1 \leq i \leq n - 3, 0 \leq t \leq 1 \]

(3)

From the above equation we have \( X_i(0) = x_i, X_i(1/3) = x_{i+1}, X_i(2/3) = x_{i+2}, X_i(1) = x_{i+3} \). So, we made sure that the Bezier curve, in the ith patch, passes through \( x_i, x_{i+1}, x_{i+2} \) and \( x_{i+3} \) points, for \( 0 \leq t \leq 1 \).

If in equation (3) we make a change of parameters, namely if we let \( u = 3t \), or

\[ t = \frac{u}{3}, 0 \leq t \leq 1, 0 \leq u \leq 3, \]

then (3) becomes:

\[ X_i(u) = \frac{(3 - u)(u - 1)(u - 2)}{6} x_i + \frac{3u(3 - u)(2 - u)}{6} x_{i+1} + \]
\[ + \frac{3u(3 - u)(u - 1)}{6} x_{i+2} + \frac{u(u - 1)(u - 2)}{6} x_{i+3}, \]
\[ 0 \leq u \leq 3, \]

(4)

\[ X_i(0) = x_i, X_i(1) = x_{i+1}, X_i(2) = x_{i+2}, X_i(3) = x_{i+3}. \]

Now, if we let \( v = 3t - 1 \), or

\[ t = \frac{v + 1}{3}. \]
−1 ≤ v ≤ 2, then (3) becomes:

\[ X_i(v) = \frac{v(v - 1)(2 - v)}{6} x_i + \frac{(1 + v)(1 - v)(2 - v)}{2} x_{i+1} + \frac{v(v + 1)(2 - v)}{2} x_{i+2} + \frac{v(v + 1)(v - 1)}{6} x_{i+3}, \]

\(-1 ≤ v ≤ 2,\)

(5)

\[ X_i(-1) = x_i, X_i(0) = x_{i+1}, X_i(1) = x_{i+2}, X_i(2) = x_{i+3}. \]

Finally let \( w = 3t - 2, or \)

\[ t = \frac{w + 2}{3}, \]

\(-2 ≤ w ≤ 1\) then (3) becomes:

\[ X_i(w) = \frac{w(w + 1)(1 - w)}{6} x_i + \frac{w(w + 2)(w - 1)}{2} x_{i+1} + \frac{(w + 2)(w + 1)(1 - w)}{2} x_{i+2} + \frac{(w + 2)(w + 1)w}{6} x_{i+3}, \]

\(-2 ≤ w ≤ 1,\)

(6)

\[ X_i(-2) = x_i, X_i(-1) = x_{i+1}, X_i(0) = x_{i+2}, X_i(1) = x_{i+3}. \]

So for i=1 (the first patch) from equation (3) we have:

\[ X_1(t) = \frac{(1 - t)(3t - 1)(3t - 2)}{2} x_1 + \frac{9(t(1 - t)(2 - 3t)}{2} x_2 + \frac{9t(1 - t)(3t - 1)}{2} x_3 + \frac{t(3t - 1)(3t - 2)}{2} x_4, \]

\(0 ≤ t ≤ 1,\)
or we can use (4) and then we will get:

\[ X_1(u) = \frac{(3-u)(u-1)(u-2)}{6} x_1 + \frac{3u(3-u)(2-u)}{6} x_2 + \]
\[ + h_1(\frac{3u(3-u)(u-1)}{6}) x_3 + \frac{u(u-1)(u-2)}{6} x_4, \]
\[ 0 \leq u \leq 3 \]

(7)

where \( h_1 \) (or \( h_{1,2} \)) is a parameter which can be used to obtain stress and also continuity of the first and the second derivatives, as we will see later, if such an assumption is supported by the data or the physics of the experiment under consideration.

Now, set \( i=2 \), which is the second patch. Then from equation (5) we get:

\[ X_1(v) = \frac{v(v-1)(2-v)}{6} x_1 + \frac{(1+v)(1-v)(2-v)}{2} x_2 + \]
\[ + \frac{v(v+1)(2-v)}{2} x_3 + \frac{v(v+1)(v-1)}{6} x_4, \]
\[ -1 \leq v \leq 2 \]

(8)

Also from (3) and (4) we get that

\[ X_2(t) = \frac{(1-t)(3t-1)(3t-2)}{2} x_2 + \frac{9t(1-t)(2-3t)}{2} x_3 + \]
\[ + \frac{9t(1-t)(3t-1)}{2} x_4 + \frac{t(3t-1)(3t-2)}{2} x_5 \]
\[ 0 \leq t \leq 1 \]
\[ X_2(u) = \frac{(3-u)(u-1)(u-2)}{6} x_2 + \frac{u(3-u)(2-u)}{2} x_3 + \frac{u(3-u)(u-1)}{2} x_4 + \frac{u(u-1)(u-2)}{6} x_5, \quad 0 \leq u \leq 3 \]

(9)

respectively. Now, we can see that both (8) and (9) are passing through \( x_2 \) and \( x_3 \). Of course the parameters \( t, u, v \) used above are dummy variables. So now we are going to replace \( u, v \) by \( t \), we'll average the common parts in the second path and we'll use the stress parameters \( h_{2,1} \) and \( h_{2,2} \) for the noncommon parts, in the following way:

\[
X_2(t) = h_{2,1} \frac{t(t-1)(2-t)}{6} x_1 + \frac{1}{2} \frac{(1+t)(1-t)(2-t)}{2} x_2 + \frac{t(t+1)(2-t)}{2} x_3 + \frac{(3-t)(t-1)(t-2)}{2} x_2 + \frac{t(3-t)(2-t)}{2} x_3 + h_{2,2} \left[ \frac{t(3-t)(t-1)}{2} + \frac{t(t+1)(t-1)}{6} \right] x_4 + h_{2,1} \frac{t(t-1)(t-2)}{6} x_5
\]

or

\[
X_2(t) = \frac{(1-t)(2-t)(6+2t)}{12} x_2 + t(2-t) x_3 + h_{2,1} \left[ \frac{t(t-1)(2-t)}{6} x_1 + \frac{t(t-1)(t-2)}{6} x_5 \right] + h_{2,2} \frac{t(t-1)(5-t)}{3} x_4, \quad 0 \leq t \leq 1.
\]

(10)

Now, we are going to look at the \( j \)th patch, where \( 3 \leq j \leq n-3 \). The way of calculating the equation for the \( j \)th patch, is similar
to the one used for the second patch. So, for $i=j ,3 \leq j \leq n-3$, and from equations (6), (5), and (4) we get:

$$X_{j-2}(w) = \frac{(1 - w)(1 + w)w}{6} x_{j-2} - \frac{w(w + 2)(1 - w)}{2} x_{j-1} + \frac{(w + 2)(w + 1)(1 - w)}{2} x_{j} + \frac{(w + 2)(w + 1)w}{6} x_{j+1}$$

$$-2 \leq w \leq 1$$

$$X_{j-1}(v) = \frac{v(v - 1)(2 - v)}{6} x_{j-1} + \frac{(1 + v)(1 - v)(2 - v)}{2} x_{j} + \frac{v(v + 1)(2 - v)}{2} x_{j+1} + \frac{v(v + 1)(v - 1)}{6} x_{j+2}$$

$$-1 \leq v \leq 2$$

$$X_{j}(u) = \frac{(3 - u)(u - 1)(u - 2)}{6} x_{j} + \frac{3u(3 - u)(2 - u)}{6} x_{j+1} + \frac{3u(3 - u)(u - 1)}{6} x_{j+2} + \frac{u(u - 1)(u - 2)}{6} x_{j+3}$$

$$0 \leq u \leq 3$$

respectively. As we can see, equations (11), (12) and (13) are passing through $x_{j}$ and $x_{j+1}$. So, if we replace the dummy variables $v, w, u$ by $t$, we average the common parts in the $j$-th path and we use the stress parameters $h_{j,1}$ and $h_{j,2}$ for the noncommon parts, as before, we’ll get:

$$X_{j}(t) = \frac{1}{3} \left[ \frac{(t + 2)(t + 1)(1 - t)}{2} x_{j} + \frac{t(t + 1)(t + 2)}{2} x_{j+1} + \frac{(t + 1)(1 - t)(2 - t)}{2} x_{j} + \frac{t(t + 1)(2 - t)}{2} x_{j+1} + \frac{3u(3 - u)(2 - u)}{6} x_{j+1} + \frac{u(u - 1)(u - 2)}{6} x_{j+3} \right]$$
\[
X_j(t) = \frac{1}{18}[(1 - t)(t^2 + 7t + 18)x_j + t(t^2 - 9t + 26)x_{j+1}] + \\
+ \frac{h_{j,1}}{6} [t(1 - t)(1 + t)x_{j-2} + t(t - 1)(t - 2)x_{j+3}] + \\
+ \frac{h_{j,2}}{3} [t(t - 1)(4 + t)x_{j-1} + t(t - 1)(5 - t)x_{j+2}]
\]  

or

\[
X_j(t) = \frac{1}{6} + \frac{2}{3} t + \frac{1}{2} t^2 + \frac{1}{6} t^3
\]  

(14)

So, \(X_j(0) = x_j, X_j(1) = x_{j+1}\), which means that the jth patch, as defined in equation (19), passes through the points \(x_j\) and \(x_{j+1}\). Also, equation (14), is of a third-degree Bezier curve type, with Bezier points, [Farin '90]

\(b_0 = P_j, b_3 = P_{j+1}\) and

\[
b_1 = \frac{184}{243}P_j + \frac{104}{243}P_{j+1} + \frac{4}{81}h_{j,1}P_{j-2} + \frac{5}{81}h_{j,2}P_{j-3} - \\
- \frac{26}{81}h_{j,2}P_{j-1} - \frac{28}{81}h_{j,2}P_{j+2}
\]

\[
b_2 = \frac{104}{243}P_j + \frac{184}{243}P_{j+1} + \frac{5}{81}h_{j,1}P_{j-2} + \frac{4}{81}h_{j,1}P_{j+3} - \\
- \frac{26}{81}h_{j,2}P_{j-1} - \frac{28}{81}h_{j,2}P_{j+2}
\]
where $P_j = (x_j, y_j, z_j), i = 1, \ldots, n$.

Now, since we have totally $(n-1)$ patches, there are two left to calculate, $(n-2)$ and $(n-1)$. So we are going to do similar work for the last two patches, for $i=n-2$ and $i=n-1$. So if $i=n-2$, from equations (5) and (6) we get:

$$X_{n-3}(v) = \frac{(2-v)(v-1)v}{6} x_{n-3} + \frac{(v+1)(1-v)(2-v)}{2} x_{n-2} +$$

$$+ \frac{v(2-v)(v+1)}{2} x_{n-1} + \frac{(v+1)(v-1)v}{6} x_n,$$

$-1 \leq v \leq 2$

and

$$X_{n-4}(w) = \frac{(1-w)(w+1)w}{6} x_{n-4} + \frac{(w+2)(w-1)w}{2} x_{n-3} +$$

$$+ \frac{(w+2)(1-w)(w+1)}{2} x_{n-2} + \frac{(w+2)(w+1)w}{6} x_{n-1},$$

$-2 \leq w \leq 1$

respectively, and both are passing through $x_{n-2}$ and $x_{n-1}$. So by replacing $v, w$ by $t$, averaging the common parts, and using the stress parameters $h_{n-2,1}$ and $h_{n-2,2}$ we’ll get:

$$X_{n-2}(t) = \frac{1}{2} \left[ \frac{(t+2)(1-t)(1+t)}{2} x_{n-2} + \frac{(1+t)(1-t)(2-t)}{2} x_{n-2} + \frac{t(t+1)(t+2)}{6} x_{n-2} + \frac{t(t+1)(2-t)}{2} x_{n-1} \right] + h_{n-2,1} \left[ \frac{(1-t)(1+t)t}{6} x_{n-4} + \frac{t(t+1)(t-1)}{6} x_n \right] + h_{n-2,2} \left[ \frac{(t+2)(t-1)t}{2} + \frac{(2-t)(t-1)t}{6} \right] x_{n-3}$$
or

\[
X_{n-2}(t) = \frac{1}{2}[2(1-t)(1+t)x_{n-2} + \frac{(t+1)(4-t)t}{3}x_{n-1}] + \\
+h_{n-2,1}[\frac{(1-t)(1+t)t}{6}x_{n-4} + \frac{t(t+1)(t-1)}{6}x_n] + \\
+h_{n-2,2}\frac{t(t-1)(t+4)}{3}x_{n-3}
\]

(15)

Last, for \(i=n-1\) (the last path), we use equation (6), starting from \(x_{n-3}\) point, which passes through \(x_{n-1}\) point, and by using the stress parameter \(h_{n-1}\) (or \(h_{n-1,1}\)) we get:

\[
X_{n-1}(t) = h_{n-1}[\frac{(1-t)(t+1)t}{6}x_{n-3} + \frac{(t+2)(t-1)t}{2}x_{n-2}] + \\
\frac{(t+2)(1-t)(t+1)}{2}x_{n-1} + \frac{(t+2)(t+1)t}{6}x_n, \\
0 \leq t \leq 1
\]

(16)

So we have five forms of equations for \(i=1, i=2, 3 \leq i \leq n-3, i=n-2\) and \(i=n-1\), which are equations (7), (10), (14), (15) and (16) respectively.

This is the time to specify these stress parameters \(h_{i,j}\), in order to get second order continuity. Since we have \(X_i(t)\) to be a cubic polynomial of \(t\), the only points that we should require continuity are the end points. Thus, we require \(X'_i(1) = X'_{i+1}(0)\)
and $X''_i(1) = X''_{i+1}(0), 1 \leq i \leq n-2,$ in order to get continuity at the $x_i$ point. So, we need to calculate $X'_i(t)$ and $X''_i(t)$, for $1 \leq i \leq n-1,$ and to estimate the $h_{i,j}, 1 \leq i \leq n-1.$ and $j = 1,2.$ From equations (9),(14),(19),(23) and (24) we get:

$$X'_1(t) = \frac{-3t^2 + 12t - 11}{6} x_1 + \frac{3t^2 - 10t + 6}{2} x_2$$
$$+ \frac{h_1}{6} [3(-3t^2 + 8t - 3)x_3 + (3t^2 - 6t + 2)x_4]$$

$$X''_1(t) = (-t + 2)x_1 + (3t - 5)x_2 + \frac{h_1}{6} [3(-6t + 8)x_3 + (6t - 6)x_4]$$

$$X'_2(t) = \frac{-7 + 3t^2}{6} x_2 + (2 - 2t)x_3 + \frac{h_{2,1}}{6} [(-2t^2 + 5t - 2)x_1 +$$
$$+ (3t^2 - 6t + 2)x_5] + \frac{h_{2,2}}{3} (-3t^2 + 12t - 5)x_4$$

$$X''_2(t) = tx_2 - 2x_3 + \frac{h_{2,1}}{6} [(-4t + 5)x_1 + (6t - 6)x_5] +$$
$$\frac{h_{2,2}}{3} (-6t + 12)x_4$$

$$X'_j(t) = \frac{1}{18} [(-3t^2 - 12t - 11)x_j + (3t^2 - 18t + 26)x_{j+1}]$$
$$+ \frac{h_{j,1}}{6} [(-3t^2 + 1)x_{j-2} + (3t^2 - 6t + 2)x_{j+3}] +$$
$$+ \frac{h_{j,2}}{3} [(3t^2 + 6t - 4)x_{j-1} + (-3t^2 + 12t - 5)x_{j+2}]$$

$$X''_j(t) = \frac{1}{18} [(-6t - 12)x_j + (6t - 18)x_{j+1}] + h_{j,1}[-tx_{j-2}$$
$$+ (t - 1)x_{j+3}] + h_{j,2}[(2t + 2)x_{j-1} + (-2t + 6)x_{j+2}]$$

$$3 \leq j \leq n-3$$
\[ X'_{n-2}(t) = -2tx_{n-2} + \frac{-3t^2 + 6t + 4}{6}x_{n-1} \]
\[ + \frac{h_{n-2,1}}{6}[(1 - 3t^2)x_{n-4} + \]
\[ + (3t^2 - 1)x_n] + \frac{h_{n-2,2}}{3}(3t^2 + 6t - 4)x_{n-3} \]

\[ X''_{n-2}(t) = -2x_{n-2} + (1 - t)x_n - 1 + h_{n-2,1}(-tx_{n-4} + tx_n) + \]
\[ 2h_{n-2,2}(t + 1)x_{n-3} \]
\[ X'_{n-1}(t) = [(1 - 3t^2)x_{n-3} + 3(3t^2 + 2t - 2)x_{n-2}] \frac{h_{n-1,1}}{6} + \]
\[ + \frac{3t^2 - 4t + 1}{2}x_{n-1} + \frac{3t^2 + 6t + 2}{6}x_n \]
\[ X''_{n-1}(t) = [-tx_{n-3} + (3t + 1)x_{n-2}]h_{n-1,1} + (3t - 2)x_{n-1} \]
\[ + (t + 1)x_n \]

respectively. Now we require:

\[ X'_j(1) = X'_{j+1}(0), \]
\[ X''_j(1) = X''_{j+1}(0), \]
\[ 1 \leq j \leq n - 2 \] (17)

So from (17) we can get a \(2(n-2)\times2(n-2)\) system, with \(h_{i,j}\) as unknowns. This system looks as follows:

\[-2x_1 - 3x_2 + h_1(6x_3 - x_4) = -7x_2 + 12x_3 + 2h_{2,1}(-x_1 + x_5) - 10h_{2,2}x_4,\]
\[6x_1 - 12x_2 + h_16x_3 = -2x_3 + h_{2,1}(5x_1 - 6x_5) + 24x_4h_{2,2}
\[-12x_2 + h_{2,13}(x_1 - x_5) + h_{2,2}24x_4 = -11x_3 + 26x_4 + h_{3,13}(x_1 + 2x_6) + h_{3,26}(-4x_2 - 5x_5),\]
Now, by simplifying the previous system, we end up with the following system:
\begin{align*}
h_1(6x_3 - x_4) - 2h_{2,1}(-x_1 + x_5) + 10h_{2,2}x_4 &= 2x_1 - 4x_2 + 12x_3 \\
h_16x_3 - h_{2,1}(5x_1 - 6x_5) - 24x_4 &= -6x_1 + 12x_2 - 12x_3 \\
h_{2,1}3(x_1 - x_5) + 24h_{2,2}x_4 - h_{3,1}(x_1 + 2x_6) - h_{3,2}6(-4x_2 - 5x_5) &= 12x_2 - 13x_3 \\
h_{2,1}x_1 + 12h_{2,2}x_4 + h_{3,1}6x_6 - 2h_{3,2}(6x_2 + 12x_5) &= -6x_2 + 8x_3 - 6x_4 \\
h_{j,1}3(-2x_{j-2} - x_{j+3}) + h_{j,2}6(5x_{j-1} + 4x_{j+2}) - h_{j+1,1}3(x_{j-1} + 2x_{j+4}) - h_{j+1,2}6(-4x_j - 5x_{j+3}) &= 26x_j - 22x_{j+1} + 26x_{j+2} \\
h_{j,1}x_{j-2} + h_{j,2}(4x_{j-1} + 2x_{j+2}) + h_{j+1,1}x_{j+4} - h_{j+1,2}(2x_j + 4x_{j+3}) &= x_j - x_{j+2}, \quad 3 \leq j \leq n - 4 \\
h_{n-3,1}3(-2x_{n-5} - x_n) + h_{n-3,2}(5x_{n-4} + 4x_{n-1}) - h_{n-2,1}3(x_{n-4} - x_n) + 24h_{n-2,2}x_{n-3} &= 26x_{n-3} - 11x_{n-2} + 12x_{n-1} \\
-3h_{n-3,1}x_{n-5} + h_{n-3,2}(2x_{n-4} + x_{n-1}) - 6h_{n-2,2}x_{n-3} &= -3x_{n-3} - 4x_{n-2} + 3x_{n-1} \\
h_{n-2,1}2(-x_{n-4} + x_n) + h_{n-2,2}10x_{n-3} - h_{n-1}(x_{n-3} - 6x_{n-2}) &= 12x_{n-2} - 4x_{n-1} + 2x_n \\
h_{n-2,1}(-x_{n-4} + x_n) + 4h_{n-2,2}x_{n-3} - h_{n-1}x_{n-2} &= 2x_{n-2} - 2x_{n-1} + x_n 
\end{align*}

Note: Similar work we do for \( Y_i, Z_i \) patches and we get alike equations.
1.3 Surface estimation

Let \((x_{ij}, y_{ij}, z_{ij}), i, j \in N\) be data points in the three dimensional space. The \((i,j)\) patch will be denoted by \(X_{ij}(t, u), i, j \in N, 0 \leq t \leq 1, 0 \leq u \leq 1\) and is defined by \(x_{ij}, x_{i+1,j}, x_{i,j+1}, x_{i+1,j+1}\). The equation of the patch \(X_{ij}(t, u)\) can be found if we find the equation of the curve \(X_{ij}(t,0), 0 \leq t \leq 1\), defined by \(x_{ij}, x_{i+1,j}\), and the curve \(x_{ij}(0,u), 0 \leq u \leq 1\), defined by \(x_{ij}, x_{i,j+1}\), and then we take their tensor product. Thus for an interior patch \(X_{ij}(t, u), X_{2,j}(t, u), X_{2,k-2}(t, u), X_{2,2}(t, u), X_{i,2}(t, u), X_{i,k-2}(t, u), X_{n-2,2}(t, u), X_{n-2,j}(t, u), X_{n-2,k-2}(t, u)\) (a patch that has neighboring patches on all sides), we have:

\[
X_{ij}(t, u) = T_1 B_1 U_1, \ 0 \leq t, u \leq 1, 3 \leq i \leq n-3, 3 \leq j \leq k-3
\]

where

\[
T_1 = \left[ \frac{1}{6}h_{i,1}t(1-t)(1+t), \frac{1}{6}h_{i,2}t(1-t)(8+2t), \frac{1}{18}(12(t+1)(1-t) + (3-t)(t-1)(t-2)), \frac{1}{18}(t(t^2 - 9t + 26)), \frac{1}{3}h_{i,1}t(1-t)(5-t), \frac{1}{6}h_{i,1}t(t-1)(t-2) \right]
\]

\[
B_1 = \begin{bmatrix}
x_{i-2,j-2} & x_{i-2,j-1} & x_{i-2,j} & x_{i-2,j+1} & x_{i-2,j+2} & x_{i-2,j+3} \\
x_{i-1,j-2} & x_{i-1,j-1} & x_{i-1,j} & x_{i-1,j+1} & x_{i-1,j+2} & x_{i-1,j+3} \\
x_{i,j-2} & x_{i,j-1} & x_{i,j} & x_{i,j+1} & x_{i,j+2} & x_{i,j+3} \\
x_{i+1,j-2} & x_{i+1,j-1} & x_{i+1,j} & x_{i+1,j+1} & x_{i+1,j+2} & x_{i+1,j+3} \\
x_{i+2,j-2} & x_{i+2,j-1} & x_{i+2,j} & x_{i+2,j+1} & x_{i+2,j+2} & x_{i+2,j+3} \\
x_{i+3,j-2} & x_{i+3,j-1} & x_{i+3,j} & x_{i+3,j+1} & x_{i+3,j+2} & x_{i+3,j+3} \\
\end{bmatrix}
\]
\[ U_1 = \begin{bmatrix} \frac{1}{6}s_{i,1}(1 - u)(1 + u)u \\ \frac{1}{3}s_{i,2}(u(u - 1)(4 + u)) \\ \frac{1}{18}(12(u + 1)(1 - u) + (3 - u)(u - 1)(u - 2)) \\ \frac{1}{18}(u(u^2 - 9u + 26)) \\ \frac{1}{3}s_{i,2}u(u - 1)(5 - u) \\ \frac{1}{6}s_{i,1}u(u - 1)(u - 2) \end{bmatrix} \]

It is easy to see that: \( X_{ij}(0, 0) = x_{ij}, X_{ij}(1, 0) = x_{i+1,j}, \)
\( X_{ij}(0, 1) = x_{i,j+1}, X_{ij}(1, 1) = x_{i+1,j+1}, 1 \leq i \leq n, 1 \leq j \leq k \)

\[ X_{2,j}(t, u) = T_2 B_2 U_1, \quad 0 \leq t, u \leq 1, 3 \leq j \leq k - 3 \quad (21) \]

where
\[ T_2 = [\frac{1}{6}h_{2,1}t(t - 1)(2 - t), \frac{1}{6}(1 - t)(2 - t)(3 + t), t(2 - t), \frac{1}{3}h_{2,2}t(t - 1)(5 - t), \frac{1}{6}h_{2,1}t(t - 1)(t - 2)] \]
$B_2 = \begin{bmatrix}
  x_{1,j-2} & x_{1,j-1} & x_{1,j} & x_{1,j+1} & x_{1,j+2} & x_{1,j+3} \\
  x_{2,j-2} & x_{2,j-1} & x_{2,j} & x_{2,j+1} & x_{2,j+2} & x_{2,j+3} \\
  x_{3,j-2} & x_{3,j-1} & x_{3,j} & x_{3,j+1} & x_{3,j+2} & x_{3,j+3} \\
  x_{4,j-2} & x_{4,j-1} & x_{4,j} & x_{4,j+1} & x_{4,j+2} & x_{4,j+3} \\
  x_{5,j-2} & x_{5,j-1} & x_{5,j} & x_{5,j+1} & x_{5,j+2} & x_{5,j+3}
\end{bmatrix}
$
\[ X_{2,2}(t, u) = T_2 B_4 U_3, \ 0 \leq t, u \leq 1, \]  \tag{23} 

where

\[
B_4 = \begin{bmatrix}
  x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & x_{1,5} \\
  x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & x_{2,5} \\
  x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} & x_{3,5} \\
  x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} & x_{4,5} \\
  x_{5,1} & x_{5,2} & x_{5,3} & x_{5,4} & x_{5,5}
\end{bmatrix}
\]

\[ U_3 = \begin{bmatrix}
  \frac{1}{6} s_{2,1}(u(u - 1)(2 - u)) \\
  \frac{1}{6} (1 - u)(2 - u)(3 + u) \\
  u(2 - u) \\
  \frac{1}{3} s_{2,2}(u(u - 1)(5 - u)) \\
  \frac{1}{6} s_{2,1}(u(u - 1)(u - 2))
\end{bmatrix}
\]

\[ X_{i,2} = T_1 B_5 U_3, \ 0 \leq t, u \leq 1, \ 3 \leq i \leq n - 3 \]  \tag{24} 

where
\[
B_5 = \begin{bmatrix}
  x_{i-2,1} & x_{i-2,2} & x_{i-2,3} & x_{i-2,4} & x_{i-2,5} \\
  x_{i-1,1} & x_{i-1,2} & x_{i-1,3} & x_{i-1,4} & x_{i-1,5} \\
  x_{i,1} & x_{i,2} & x_{i,3} & x_{i,4} & x_{i,5} \\
  x_{i+1,1} & x_{i+1,2} & x_{i+1,3} & x_{i+1,4} & x_{i+1,5} \\
  x_{i+2,1} & x_{i+2,2} & x_{i+2,3} & x_{i+2,4} & x_{i+2,5} \\
  x_{i+3,1} & x_{i+3,2} & x_{i+3,3} & x_{i+3,4} & x_{i+3,5}
\end{bmatrix};
\]

\[
X_{i,k-2} = T_1 B_6 U_2, \quad 0 \leq t, u \leq 1, \quad 3 \leq i \leq n - 3 \quad (25)
\]

where
\[
B_6 = \begin{bmatrix}
  x_{i-2,k-4} & x_{i-2,k-3} & x_{i-2,k-2} & x_{i-2,k-1} & x_{i-2,k} \\
  x_{i-1,k-4} & x_{i-1,k-3} & x_{i-1,k-2} & x_{i-1,k-1} & x_{i-1,k} \\
  x_{i,k-4} & x_{i,k-3} & x_{i,k-2} & x_{i,k-1} & x_{i,k} \\
  x_{i+1,k-4} & x_{i+1,k-3} & x_{i+1,k-2} & x_{i+1,k-1} & x_{i+1,k} \\
  x_{i+2,k-4} & x_{i+2,k-3} & x_{i+2,k-2} & x_{i+2,k-1} & x_{i+2,k} \\
  x_{i+3,k-4} & x_{i+3,k-3} & x_{i+3,k-2} & x_{i+3,k-1} & x_{i+3,k}
\end{bmatrix};
\]

\[
X_{n-2,2} = T_3 B_7 U_3, \quad 0 \leq t, u \leq 1 \quad (26)
\]

where
\[
T_3 = \left[ \frac{1}{6} h_{n-2,1} (1 - t)(1 + t), \frac{1}{3} h_{n-2,2} t(t - 1)(t + 4),
\right.

(1 - t)(1 + t), \frac{1}{6} (t + 1)(4 - t), \frac{1}{6} h_{n-2,1} t(t + 1)(t - 1) \right]
\[ B_7 = \begin{bmatrix}
  x_{n-4,1} & x_{n-4,2} & x_{n-4,3} & x_{n-4,4} \\
  x_{n-3,1} & x_{n-3,2} & x_{n-3,3} & x_{n-3,4} \\
  x_{n-2,1} & x_{n-2,2} & x_{n-2,3} & x_{n-2,4} \\
  x_{n-1,1} & x_{n-1,2} & x_{n-1,3} & x_{n-1,4} \\
  x_{n,1} & x_{n,2} & x_{n,3} & x_{n,4}
\end{bmatrix} ;
\]

\[ X_{n-2,j} = T_3 B_7 U_1, \ 0 \leq t, u \leq 1 \quad (27) \]

where

\[ B_8 = \begin{bmatrix}
  x_{n-4,j-2} & x_{n-4,j-1} & x_{n-4,j} & x_{n-4,j+1} & x_{n-4,j+2} & x_{n-4,j+3} \\
  x_{n-3,j-2} & x_{n-3,j-1} & x_{n-3,j} & x_{n-3,j+1} & x_{n-3,j+2} & x_{n-3,j+3} \\
  x_{n-2,j-2} & x_{n-2,j-1} & x_{n-2,j} & x_{n-2,j+1} & x_{n-2,j+2} & x_{n-2,j+3} \\
  x_{n-1,j-2} & x_{n-1,j-1} & x_{n-1,j} & x_{n-1,j+1} & x_{n-1,j+2} & x_{n-1,j+3} \\
  x_{n,j-2} & x_{n,j-1} & x_{n,j} & x_{n,j+1} & x_{n,j+2} & x_{n,j+3}
\end{bmatrix} ;
\]

\[ X_{n-2,k-2} = T_3 B_8 U_2, \ 0 \leq t, u \leq 1 \quad (28) \]

where
Noninterior patches have one or more sides with no neighbors. Non interior patches with one side having no neighbors are for example: the \( X_{12}(t, u), X_{n-1,1}(t, u), X_{1j}, X_{n-1,j} \), where \( j \) is an index corresponding to an interior edge \((3 \leq j \leq k - 3)\) and \( 1, n \) are indices corresponding to edges with no neighbors, also \( X_{21}(t, u), X_{i1}, X_{2,k-1}, X_{n-2,k-1}(t, u), X_{n-2,1}(t, u), X_{1,k-2}(t, u), X_{1,k-1}(t, u), X_{n-1,2}(t, u), X_{n-1,k-2}(t, u) \) and \( X_{i,k-1}(t, u) \), where \( k \) corresponds to an edge with no neighbors and \( 3 \leq i \leq n - 3 \). The equations of these patches are:

\[
X_{12}(t, u) = T_4 B_{10} U_3, \quad 0 \leq t, u \leq 1
\]  

(29)

where

\[
T_4 = \left[ \frac{1}{6} (3 - t)(t - 1)(t - 2), \frac{1}{2}(t(3 - t)(2 - t), \frac{1}{2} h_1 t(3 - t)(t - 1), \frac{1}{6} h_1(t(t - 1)(t - 2)) \right]
\]
The next one is

\[ X_{1j}(t,u) = T_4B_{11}U_1, \ 0 \leq t,u \leq 1 \]  \hspace{1cm} (30)

where

\[ B_{11} = \begin{bmatrix}
  x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & x_{1,5} \\
  x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & x_{2,5} \\
  x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} & x_{3,5} \\
  x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} & x_{4,5}
\end{bmatrix} \]

Now we take the equation:

\[ X_{21}(t,u) = T_2B_{12}U_4, \ 0 \leq t,u \leq 1 \]  \hspace{1cm} (31)

where

\[ B_{12} = \begin{bmatrix}
  x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} \\
  x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} \\
  x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} \\
  x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} \\
  x_{5,1} & x_{5,2} & x_{5,3} & x_{5,4}
\end{bmatrix} \]

and
\[ U_4 = \begin{bmatrix} \frac{1}{6}(3-u)(u-1)(u-2) \\ \frac{1}{2}u(3-u)(2-u) \\ \frac{1}{2}s_1(u(3-u)(u-1)) \\ \frac{1}{6}s_1(u(u-1)(u-2)) \end{bmatrix} \]

For \( X_{n-1,1} \) patch we have:

\[ X_{n-1,1} = T_5 B_{13} U_4, \quad 0 \leq t, u \leq 1 \quad (32) \]

where

\[ T_5 = \left[ \frac{1}{6}h_{n-1}(1-t)(t+1)t, \frac{1}{2}h_{n-1}(t+2)(t-1)t, \frac{1}{2}(t+2)(t+1)(1-t), \frac{1}{6}(t+2)(t+1)t \right] \]

and

\[ B_{13} = \begin{bmatrix} x_{n-3,1} & x_{n-3,2} & x_{n-3,3} & x_{n-3,4} \\ x_{n-2,1} & x_{n-2,2} & x_{n-2,3} & x_{n-2,4} \\ x_{n-1,1} & x_{n-1,2} & x_{n-1,3} & x_{n-1,4} \\ x_{n,1} & x_{n,2} & x_{n,3} & x_{n,4} \end{bmatrix} \]
For $X_{i1}, 3 \leq i \leq n-3$, patch we have

$$X_{i1} = T_1 B_{14} U_4, \quad 0 \leq t, u \leq 1$$

(33)

where

$$B_{14} = \begin{bmatrix} x_{i-2,1} & x_{i-2,2} & x_{i-2,3} & x_{i-2,4} \\ x_{i-1,1} & x_{i-1,2} & x_{i-1,3} & x_{i-1,4} \\ x_{i,1} & x_{i,2} & x_{i,3} & x_{i,4} \\ x_{i+1,1} & x_{i+1,2} & x_{i+1,3} & x_{i+1,4} \\ x_{i+2,1} & x_{i+2,2} & x_{i+2,3} & x_{i+2,4} \\ x_{i+3,1} & x_{i+3,2} & x_{i+3,3} & x_{i+3,4} \end{bmatrix}$$

Also for the patch $X_{n-1,j}, 3 \leq j \leq n-3$ we have:

$$X_{n-1,j}(t, u) = T_5 B_{15} U_1, \quad 0 \leq t, u \leq 1$$

(34)

where

$$B_{15} = \begin{bmatrix} x_{n-3,j-2} & x_{n-3,j-1} & x_{n-3,j} & x_{n-3,j+1} & x_{n-3,j+2} & x_{n-3,j+3} \\ x_{n-2,j-2} & x_{n-2,j-1} & x_{n-2,j} & x_{n-2,j+1} & x_{n-2,j+2} & x_{n-2,j+3} \\ x_{n-1,j-2} & x_{n-1,j-1} & x_{n-1,j} & x_{n-1,j+1} & x_{n-1,j+2} & x_{n-1,j+3} \\ x_{n,j-2} & x_{n,j-1} & x_{n,j} & x_{n,j+1} & x_{n,j+2} & x_{n,j+3} \end{bmatrix}$$

For $X_{2,k-1}$ we have:

$$X_{2,k-1}(t, u) = T_2 B_{16} U_5, \quad 0 \leq t, u \leq 1$$

(35)
where

\[
B_{16} = \begin{bmatrix}
  x_{1,k-3} & x_{1,k-2} & x_{1,k-1} & x_{1,k} \\
  x_{2,k-3} & x_{2,k-2} & x_{2,k-1} & x_{2,k} \\
  x_{3,k-3} & x_{3,k-2} & x_{3,k-1} & x_{3,k} \\
  x_{4,k-3} & x_{4,k-2} & x_{4,k-1} & x_{4,k} \\
  x_{5,k-3} & x_{5,k-2} & x_{5,k-1} & x_{5,k} 
\end{bmatrix}
\]

and

\[
U_5 = \begin{bmatrix}
  \frac{1}{6} s_{k-1}((1 - u)(u + 1)u) \\
  \frac{1}{2} s_{k-1}((u + 2)(u - 1)u) \\
  \frac{1}{2}(u + 2)(u + 1)(1 - u) \\
  \frac{1}{6}(u + 2)(u + 1)u
\end{bmatrix}
\]

Also for \( X_{i,k-1} \), \( 3 \leq i \leq n - 3 \), \( X_{n-2,k-1} \) and \( X_{n-2,1} \) we have:

\[
X_{i,k-1}(t, u) = T_1 B_{17} U_5, \quad 0 \leq t, u \leq 1 \quad (36)
\]

where
where

\[
X_{n-2,k-1} = T_3 B_{18} U_5, \quad 0 \leq t, u \leq 1
\]  

and

\[
X_{n-2,1} = T_3 B_{19} U_4, \quad 0 \leq t, u \leq 1
\]
\[ X_{1,k-2} = T_4 B_{20} U_2, \ 0 \leq t, u \leq 1 \quad (39) \]

where

\[
B_{20} = \begin{bmatrix}
  x_{1,k-4} & x_{1,k-3} & x_{1,k-2} & x_{1,k-1} & x_{1,k} \\
  x_{2,k-4} & x_{2,k-3} & x_{2,k-2} & x_{2,k-1} & x_{2,k} \\
  x_{3,k-4} & x_{3,k-3} & x_{3,k-2} & x_{3,k-1} & x_{3,k} \\
  x_{4,k-4} & x_{4,k-3} & x_{4,k-2} & x_{4,k-1} & x_{4,k}
\end{bmatrix};
\]

\[
X_{1,k-1} = T_4 B_{21} U_5, \ 0 \leq t, u \leq 1 \quad (40) \]

where

\[
B_{21} = \begin{bmatrix}
  x_{1,k-3} & x_{1,k-2} & x_{1,k-1} & x_{1,k} \\
  x_{2,k-3} & x_{2,k-2} & x_{2,k-1} & x_{2,k} \\
  x_{3,k-3} & x_{3,k-2} & x_{3,k-1} & x_{3,k} \\
  x_{4,k-3} & x_{4,k-2} & x_{4,k-1} & x_{4,k}
\end{bmatrix};
\]

\[
X_{n-1,2} = T_5 B_{22} U_3, \ 0 \leq t, u \leq 1 \quad (41) \]

where

\[
B_{22} = \begin{bmatrix}
  x_{n-3,1} & x_{n-3,2} & x_{n-3,3} & x_{n-3,4} & x_{n-3,5} \\
  x_{n-2,1} & x_{n-2,2} & x_{n-2,3} & x_{n-2,4} & x_{n-2,5} \\
  x_{n-1,1} & x_{n-1,2} & x_{n-1,3} & x_{n-1,4} & x_{n-1,5} \\
  x_{n,1} & x_{n,2} & x_{n,3} & x_{n,4} & x_{n,5}
\end{bmatrix};
\]
and at last,

$$X_{n-1,k-2} = T_5 B_{23} U_2, \ 0 \leq t, u \leq 1 \quad (42)$$

where

$$B_{23} = \begin{bmatrix}
 x_{n-3,k-4} & x_{n-3,k-3} & x_{n-3,k-2} & x_{n-3,k-1} & x_{n-3,k} \\
 x_{n-2,k-4} & x_{n-2,k-3} & x_{n-2,k-2} & x_{n-2,k-1} & x_{n-2,k} \\
 x_{n-1,k-4} & x_{n-1,k-3} & x_{n-1,k-2} & x_{n-1,k-1} & x_{n-1,k} \\
 x_{n,k-4} & x_{n,k-3} & x_{n,k-2} & x_{n,k-1} & x_{n,k}
\end{bmatrix};$$

Finally, we have the patches for the points, with more than one noneighbors. And these are the patches $X_{1,1}(t,u)$ and $X_{n-1,k-1}(t,u)$, where $0 \leq t, u \leq 1$.

The tensor products for these patches are:

$$X_{1,1}(t,u) = T_4 B_{24} U_4, \ 0 \leq t, u \leq 1 \quad (43)$$

where

$$B_{24} = \begin{bmatrix}
 x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} \\
 x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} \\
 x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} \\
 x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4}
\end{bmatrix};$$

and

$$X_{n-1,k-1}(t,u) = T_5 B_{25} U_5, \ 0 \leq t, u \leq 1 \quad (44)$$
where

\[
B_{25} = \begin{bmatrix}
x_{n-3,k-3} & x_{n-3,k-2} & x_{n-3,k-1} & x_{n-3,k} \\
x_{n-2,k-3} & x_{n-2,k-2} & x_{n-2,k-1} & x_{n-2,k} \\
x_{n-1,k-3} & x_{n-1,k-2} & x_{n-1,k-1} & x_{n-1,k} \\
x_{n,k-3} & x_{n,k-2} & x_{n,k-1} & x_{n,k}
\end{bmatrix};
\]

The parametric equations for y and z are similar to those for x. The only difference is that we replace the x’s with y’s or the x’s with z’s.
Chapter 2
Examples and Error discussion

In this section we present some estimated surfaces and we discuss our results. Note that our estimation method does not require systematic sampling (equal intervals between neighboring sampling points), continuity of the first or second derivatives, or the assumption that the underlying process satisfies the intrinsic hypothesis, or is wide sense stationary. Furthermore, it does not require finding the inverse of a matrix as kriging and splines require. The calculations therefore are fast, and it is an algorithm for such real time applications as robotics and robot motion. One such application is to use a robot hand to paid spray arbitrary surfaces. Finally, the generated surface does not have to be a function as in the case of kriging.

2.1 First example

The first surface estimated in the following graphs, is given by the equation

\[ z = 0.5 + 0.00001562(x^3 + y^3 + x^2y + y^2x), -20 \leq x, y \leq 20. \]

The graph of this function is shown in Figure 1. The number of the given points \( x \) and \( y \) is 11, the range between -20 and 20, for both \( x \) and \( y \), and the grid of these given points is \( \delta x = \delta y = 4 \).
In Figure 2 we present an estimation of this function, by using the cubic method with stress parameters only. More precisely, we did not require continuity and the $h_{i,j}$ parameters were used with appropriate values, in order to get only stress. The values of $h_{i,j}$ used in Figure 2 were:

$$h_1 = 0.65, \; h_{n-1} = 0.65, \; h_{i,j} = 0.33$$

where, $n = 11$, $1 \leq i \leq n - 2$, and $j = 1, 2$. The grid used is equal to 4, which means that the data points are equidistant and the range is the same as in Figure 1 for both $x$ and $y$.

We use the same range for $x$ and $y$ as above, but not equidistant data points, with $n = 7$. Moreover by requiring continuity which leads us to solve system (19, Ch.1), we use stress and continuity parameters $h_{i,j}$, this time, and we get Figure 4. In this Figure we can see the smoothness of the estimated surface, compared to the estimated surface by using our method (Figure 2), where the stress parameters $h_{i,j}$ are chosen.

The error in both cases, is presented in Figure 3 and Figure 5. The difference between the true function (Figure 1) and the one estimated in Figure 2, where $h_{i,j}$ parameters are chosen, is shown in Figure 3. In Figure 5 we present the difference between the true function (Figure 1) and the one estimated by using our method, (Figure 4), where continuity is required.
Figure 4.

Figure 5.
In the previous graphs, the z-scales and rotation about the x and z axis, are chosen in a proper way, in order to give general aspects of the results.

The mean error and the mean of the absolute error of the first estimation, shown in Figure 2, where about 5601 points are used to form the estimated surface, is $-2.45 \times 10^{-2}$ and $2.66 \times 10^{-2}$ respectively. In graph shown in Figure 4, although only 1297 points (about 1/4 of the points used in Figure 2) are used to form the estimated surface, the mean error is $-4.91 \times 10^{-2}$ and the mean of the absolute error is $8.37 \times 10^{-2}$, which also gives order of $10^{-2}$. 
2.2 Second Example

In this example the function used for the graphs of the following Figures, is:

$$z = 1 - 0.0008(x^2 + y^2 + xy), -20 \leq x, y \leq 20$$

In Figure 6 the graph of this function is presented, where $-20 \leq x, y \leq 20$ and the distance between the given points is $\delta x = 4$ and $\delta y = 4$.

We start our approximation Figures, by Figure 7, where the estimation of the surface is based on the cubic method, without requiring continuity. The range of $x$ and $y$ is the same as before and the given points are equidistant with $\delta x = 4$ and $\delta y = 4$. The stress parameters $h_{i,j}$ are chosen the same as in (2.1) example: $h_1 = h_{n-1} = 0.65$ and $h_{i,j} = 0.33, 1 \leq i \leq n - 2$ $j = 1, 2$, and $n = 11$.

In Figure 8 we present the error, expressing the difference between the true surface (Figure 6) and the estimated one of Figure 7. In order to see our results in a "closer aspect", we give, in addition, a zoom in the graph of the estimated surface in Figure 9. The stress parameters are chosen equal to the ones of Figure 7. The number of the equidistant given points is $n = 7$, for $x$ and $y$ respectively, and the range for both $x$ and $y$ is between -20 and -10.

In the last of the graphs presented here, Figure 10, our method is applied requiring continuity. Thus the continuity and stress parameters are derived from the solution of the $2(n-2)x2(n-2)$ linear system (19), where $n = 7$. The range of both $x$ and $y$ is also between -20 and -10.
Figure 9.

Figure 10.
As in section 2.1, the z-scales and the rotation about the x and z axis are chosen in a proper way, in order to give more general aspects of the results. The mean error of the above cases is also of order $10^{-2}$ and even when we did not use continuity parameters, i.e. we chose their values, we got a mean error equal to $7.61 \times 10^{-2}$ (Figure 8).

The set of the 27 functions-programs used to get the results in section 2.1 and section 2.2 are given in the Appendix.
APPENDIX

FUNCTIONS PROGRAMMED IN C

The following functions are called from a main C program in the following way and order: The main program gets \((x_{ij}, y_{ij}, z_{ij})\) given points of \(R^3\) and provides these to SOLUTION, which returns \(h_{i,j}\) and \(s_{i,j}\). \(T_i\) gets \(h_{i,j}\) from SOLUTION and \(t, u\) from the main program, and evaluates the matrix \(T\). \(U_i\) gets \(s_{i,j}\) from SOLUTION and \(t, u\) from the main program and evaluates the matrix \(U\). MULT gets \(B\) from main program and \(T, U\) from functions \(T_i, U_i\), and returns the tensor product of \(X_{ij}(t, u), Y_{ij}(t, u), Z_{ij}(t, u)\) patches.
/* FUNCTION FOR THE SOLUTION OF THE 2(n-2)x2(n-2) SYSTEM */

float *solution(n, x, h)

int n;
float x[n], h[2*(n-2)][2];

float a[2][2], b[2][2], c, A[2], arb, *d;
int i, j;

arb = 0.0;

for (i = 0; i <= 2*n-5; i++)
    h[i][0] = 0.0;
for (i = 0; i <= 2*n-5; i++)
    h[i][1] = 0.0;

for (i = 0; i < 2; i++)
    h[0][i] = 1.0;

a[0][0] = 2.0*(x[0]-x[4]);
a[0][1] = 10.0*x[3];
a[1][0] = 6.0*x[4]-5.0*x[0];
a[1][1] = -24.0*x[3];
b[0][0] = 2.0*x[0]-4.0*x[1]+12.0*x[2];
b[0][1] = x[3]-6.0*x[2];
b[1][0] = -6.0*x[0]+12.0*x[1]-12.0*x[2];
b[1][1] = -6.0*x[2];

if (ABS(a[0][1]) <= 0.000005)
{
    h[2][0] = arb;
    h[2][1] = 0.0;
    if (b[1][1] != 0.0)
    {
        c = a[0][0] + b[0][1]*a[1][0]/(6.0*x[2]);
        if (c != 0.0)
        {
            h[1][0] = (b[0][0] - b[0][1]*b[1][0]/b[1][1]) / c;
            h[1][1] = 0.0;
            h[0][0] = (a[1][0]*h[1][0] - b[1][0]) / b[1][1];
            h[0][1] = 0.0;
            goto end1;
        }
    }
    if (b[0][0] - b[0][1]*b[1][0]/b[1][1] != 0.0)
    {
        goto stop;
    }
}

if (ABS(b[0][1]) > 0.000005)
{
    h[2][0] = arb;
    h[2][1] = 0.0;
    printf("h21 is arbitrary \n");
    h[0][0] = (a[0][0]*h[1][0] - b[0][0]) / b[0][1];
    h[0][1] = 0.0;
    goto end1;
}

if (a[0][0] != 0.0)
{
    h[0][0] = arb;
    h[0][1] = 0.0;
    printf("h1 is arbitrary \n");
    h[1][0] = (b[0][0]*b[0][1]*h[0][0]) / a[0][0];
    h[1][1] = 0.0;
    goto end1;
}

if (ABS(b[0][0]) > 0.000005)
{
    goto stop;
}

h[0][0] = arb;
h[0][1] = 0.0;

h[1][0] = arb;
h[1][1] = 0.0;
printf("h1 and h2 are arbitrary \n");

goto end1;)

A[0] = 12.0*x[0]-24.0*x[1]-168.0*x[2];
A[1] = 204.0*x[2]-24.0*x[3];
if (ABS(x[0]-6.0*x[4]) >= 0.000005)
{
    h[1][0] = A[0]/(2.0*x[0]-12.0*x[4]);
    h[1][1] = A[1]/(2.0*x[0]-12.0*x[4]);
    h[2][0] = (b[1][0]-a[1][0]*h[1][0])/a[1][1];
    h[2][1] = (b[1][1]-a[1][0]*h[1][1])/a[1][1];
    goto end1;)
if (ABS(A[1]) > 0.000005)
{
    h[0][0] = -A[0]/A[1];
    h[0][1] = 0.0;
    h[1][0] = arb;
    h[1][1] = 0.0;
    printf("h1 is arbitrary \n");
}
c2:  h[2][0] = (b[1][0]+b[1][1]*h[0][0]-a[1][0]*h[1][0])/24.0;
    h[2][1] = 0.0;
    goto end1;)
if (ABS(A[0]) <= 0.000005)
{
    h[0][0] = arb;
    h[0][1] = 0.0;
    h[1][0] = arb;
    h[1][1] = 0.0;
    printf("h2 and h22 are arbitrary \n");
    goto c2;)
    goto stop;}
if (ABS(a[1][1]) > 0.000005)
{
    if (ABS(a[0][0]) > 0.000005)
    {
        h[3][0] = b[0][0]/a[0][0];
        h[3][1] = b[0][1]/a[0][0];
        c4: h[4][0] = (b[1][0] - 6.0*x[5]*h[3][0])/a[1][1];
        h[4][1] = (b[1][1] - 6.0*x[5]*h[3][1])/a[1][1];
        goto end2;
    }
    if (ABS(b[0][1]) > 0.000005)
    {
        if (ABS(b[0][0]*b[0][1] + b[0][0]) <= 0.000005)
        {
            goto c5;
        }
        goto stop;
    }

    if (ABS(b[1][1]) <= 0.000005)
    {
        if (ABS(b[0][1]) <= 0.000005)
        {
            if (ABS(-b[0][0]/b[0][1]) <= 0.000005)
            {
                goto c6;
            }
            goto stop;
        }
        goto c5;
    }

    if (ABS(b[0][0]) <= 0.000005)
    {
        if (ABS(b[0][0] + b[0][0]) <= 0.000005)
        {
            goto c6;
        }
        goto stop;
    }

    if (ABS(b[0][0]) <= 0.000005)
    {
        goto c5;
    }
    goto stop;
}

if (ABS(b[0][0]) <= 0.000005)
{
    goto c6;
    goto stop;
}

c35: h[3][0] = arb;
    h[3][1] = 0.0;
    h[0][1] = 0.0;
    printf(\"h31 is arbitrary \n\");
    goto end2;
}

if (ABS(a[1][1]) > 0.000005)
{
    if (ABS(a[0][0]) > 0.000005)
    {
        h[3][0] = b[0][0]/a[0][0];
        h[3][1] = b[0][1]/a[0][0];
        c4: h[4][0] = (b[1][0] - 6.0*x[5]*h[3][0])/a[1][1];
        h[4][1] = (b[1][1] - 6.0*x[5]*h[3][1])/a[1][1];
        goto end2;
    }
end2:  for (j = 3; j <= n-4; j++)
{
    a[0][0] = -3.0*(x[j-2]+2.0*x[j+3]);
    a[0][1] = 6.0*(4.0*x[j-1]+5.0*x[j+2]);
    a[1][0] = x[j+3];
    a[1][1] = -2.0*(x[j-1]+2.0*x[j+2]);
    b[0][0] = 6.0*x[j-1]-22.0*x[j]+26*x[j+1]+3.0*(2.0*x[j-3]+x[j+2])*h[2*j-3][0];
    b[0][1] = b[0][0]*6.0*(5.0*x[j-2]+4.0*x[j+1])*h[2*j-2][0];
    b[0][1] = 3.0*(2.0*x[j-3]+x[j+2])*h[2*j-3][1];
    b[0][1] = b[0][1]-6.0*(5.0*x[j-2]+4.0*x[j+1])*h[2*j-2][1];
    b[1][0] = x[j-2]*x[j+3]+x[j-3]*h[2*j-3][0]+4.0*x[j-2]+2.0*x[j+1]*h[2*j-2][0];
    b[1][1] = x[j]-3.0*h[2*j-3][1]-2.0*(2.0*x[j-2]+x[j+1])*h[2*j-2][1];
    c = a[0][0]*a[1][0]-a[1][0]*a[0][0];
    A[0] = -a[1][0]*b[0][0]+a[0][0]*b[1][0];
    A[1] = -a[1][0]*b[0][1]+a[0][0]*b[1][1];
    if (ABS(A[1][0]) > 0.000005)
    { if (ABS(c) > 0.000005)
      { h[2*j][0] = A[0]/c;
        h[2*j][1] = A[1]/c;
      }
      goto c7;
    }
    if (ABS(A[1][1]) > 0.000005)
    { if (ABS(h[0][1]) <= 0.000005)
      { goto stop; }
        h[0][0] = A[0]/A[1];
        h[0][1] = 0.0;
        goto c8;
        goto stop; }
    if (ABS(A[1][1]) > 0.000005)
    { h[2*j][0] = b[1][0]/a[1][1];
      h[2*j][1] = b[1][1]/a[1][1];
      A[0] = h[2*j][0]*a[0][0]-b[1][0];
      A[1] = h[2*j][1]*a[0][1]-b[1][1];
      if (ABS(a[0][0]) > 0.000005)
      { h[2*j-1][0] = -A[0]/a[0][0];
        h[2*j-1][1] = -A[1]/a[0][1];
        goto end3; }
        h[2*j-1][0] = abs;
        h[2*j-1][1] = 0.0;
        if (ABS(h[0][1]) > 0.000005)
        { if (ABS(A[1]) > 0.000005)
          { h[0][0] = -A[0]/A[1];
            h[0][1] = 0.0;
            goto end3; }
            if (ABS(A[0]) <= 0.000005)
            { goto end3; }
            goto stop; }
        if (A[0]*A[1]*h[0][0] == 0.0)
\begin{verbatim}
if (h[2*n-7][0] = arb; 
h[2*n-7][1] = 0.0;  
  goto stop;
end3:

a[0][0] = -1.0*(x[n-5]-x[n-1]);
a[0][1] = 24.0*x[n-4];
a[1][0] = 0.0;
a[1][1] = -6.0*x[n-4];
b[0][0] = 26.0*x[n-6]-12.0*x[n-3]+12.0*x[n-2]+3.0*h[2*n-9][0]*(2.0*x[n-6]+x[n-1])
b[0][1] = b[0][0]-6.0*(5.0*x[n-5]+4.0*x[n-2])*h[2*n-8][0];
b[1][1] = -0.0;
b[1][0] = b[1][0]-1.0*x[n-6]*h[2*n-9][0];

if (ABS(b[0][0]+b[1][0]*h[0][0]) < 0.000005)
  goto end4;

if (ABS(h[0][0]) > 0.000005)
  goto c9;
if (ABS(h[0][1]) > 0.000005)
  goto stop;
c9: if(ABS(a[0][0]) > 0.000005)
  goto stop;

h[2*n-7][0] = b[0][0]/a[0][0];
h[2*n-7][1] = b[0][1]/a[0][0];
goto end4;

if (ABS(b[0][0]+b[0][1]*h[0][0]) <= 0.000005)
  goto end4;

if (ABS(b[1][0]+b[1][0]*h[0][0]) <= 0.000005)
  goto stop;

to and4;)

A[0] = -b[0][0]-4.0*b[1][0];
A[1] = -b[0][1]-4.0*b[1][1];
if (ABS(A[0][0]) > 0.000005)
  goto stop;

h[2*n-7][0] = -A[0][0]/A[0][0];
h[2*n-7][1] = -A[1][0]/A[0][0];
h[2*n-6][0] = b[1][0]/a[1][0];
h[2*n-6][1] = b[1][1]/a[1][1];
goto end4;
h[2*n-7][0] = arb;
h[2*n-7][1] = 0.0;

if (ABS(h[0][1]) > 0.000005)
  goto c9;
if (ABS(A[1][1]) > 0.000005)
  goto stop;

h[0][0] = A[0][0]/A[1][0];
h[0][1] = A[1][0]/A[1][0];
\end{verbatim}
48

c10:  h[2*n-6][0] = b[1][0]/a[1][1];
    h[2*n-6][1] = b[1][1]/a[1][1];
    goto end4;
if (ABS(A[1]) < -0.000005)
(goto c10);
    goto stop;
if (ABS(A[0]*A[1]*h[0][0]) < -0.000005)
(goto c10);
    goto stop;
end4:
a[0][0] = -x[n-4]+6*x[n-3];
a[0][1] = -2.0*h[2*n-7][1]*(x[n-5]-x[n-1])+10.0*x[n-4]*h[2*n-6][1];
a(1)[0] = -x[n-3];
a(1)[1] = 4.0*x[n-4]*h[3*n-6][1]+h[2*n-7][1]*(-x[n-5]+x[n-1]);
b[0][0] = 12.0*x[n-3]-4.0*x[n-2]+2.0*x[n-1]-2.0*h[2*n-7][0]*(x[n-1]-x[n-5]);
b[0][0] = b[0][0]-10.0*x[n-4]*h[2*n-6][0];
b[0][1] = 0.0;
b(1)[0] = 2.0*x[n-3]-x[n-2]+x[n-1]-4.0*x[n-4]*h[2*n-6][0];
b(1)[0] = b(1)[0]*(x[n-1]-x[n-5])*h[2*n-7][0];
b(1)[1] = 0.0;
if (ABS(h[0][1]) > 0.000005)
{
    if (ABS(a[0][0]) < -0.000005)
    {
        if (ABS(a[0][1]) > 0.000005)
        {
            h[0][0] = b[0][0]/a[0][1];
            h[0][1] = 0.0;
            if (ABS(a[1][0]) > 0.000005)
            {
                h[2*n-5][0] = (b[1][0]-a[1][1]*h[0][0])/a[1][1];
                h[2*n-5][1] = 0.0;
                goto end5;
            }
            if (ABS(a[1][1]*h[0][0]-b[1][0]) <= 0.000005)
            {
                h[2*n-5][0] = arb;
                h[2*n-5][1] = 0.0;
                goto end5;
            }
            goto stop;
        }
        if (ABS(a[1][0]) > 0.000005)
        {
            h[0][0] = arb;
            h[0][1] = 0.0;
            printf("hi is arbitrary \n");
            goto c11;
        }
        if (ABS(a[1][1]) > 0.000005)
        {
            h[2*n-5][0] = arb;
            h[2*n-5][1] = 0.0;
            printf("h%d is arbitrary \n", n-1);
            h[0][0] = b[1][0]/a[1][1];
            h[0][1] = 0.0;
            if (ABS(b[1][0]) <= 0.000005)
            {
                h[0][0] = arb;
                h[0][1] = 0.0;
                h[2*n-5][0] = arb;
                h[2*n-5][1] = 0.0;
                printf("hi and h%d are arbitrary \n", n-1);
            }
            goto stop;
        }
    }
    if (ABS(a[0]) <= 0.000005)
    {
        if (ABS(A[0]) < -0.000005)
        {
            h[0][0] = arb;
        }
    }
}

h[0][1] = 0.0;
printf("h1 is arbitrary \n");
c12: h[2*n-5][0] = (h[0][0]*a[0][1]+h[0][0])/a[0][0];
h[2*n-5][1] = 0.0;
goto end5;
goto stop;
h[0][0] = A[0]/c;
h[0][1] = 0.0;
printf("h0 \n");
h[2*n-5][0] = (1.0*C[1]*h[0][0])/a[0][0];
h[2*n-5][1] = 0.0;
goto end5;

if (ABS(a[0][0]) > 0.000005)
    goto c12;
if (a[0][1]*h[0][0] == b[0][0])
    if (ABS(a[1][0]) > 0.000005)
        goto c11;
    if (a[1][1]*h[0][0] == b[1][0])
    h[2*n-5][0] = az:
    h[2*n-5][1] = 0.0;
printf("h2 is arbitrary \n", n+1);
goto end5;
goto stop;
end5: if ((ABS(h[1][1] <= 0.000005) & (ABS(h[2][1] <= 0.000005))
    goto end;
for (i=1;i<= 2*n-6;i++)
    h[1][0] = h[1][0]+h[1][1]*h[0][0];
h[1][1] = 0.0;
goto end;
sh[0][0];
return d;

/* T1 FUNCTION-----------------------------------------------*/
float *T1(h,t, i,n)

int i;
float n[n][2], t;

{ float T[6], d;

T[0] = h[1][0] * c * (1.0-t) * (1.0-t) / 6.0;
T[1] = h[1][1] * c * (t-1.0) * (4.0-t) / 3.0;
T[2] = 1.0/18.0 * (12.0 * t * (1.0-t) + (1.0-t)* (t-1.0) * (t-2.0));
T[3] = 1.0/18.0 * t * (t-9.0+t+26);
T[4] = 1.0/3.0 * h[1][1] * c * (c-t-1) * (c+t-2);
T[5] = 1.0/6.0 * h[1][0] * c * (t-t-1) * (t-t-2);
return d;
}

/* T2 FUNCTION-----------------------------------------------*/
float *T2(h,t, i,n)

float r[n][2], t;

{ float T[5], d;

T[0] = -1.0/6.0 * h[1][1] * c * (t-t-1) * (t-t-2);
T[1] = 1.0/6.0 * c * (1.0-t) * (1.0-t) * (t-t-2);
T[2] = t * (t-0.0);
return d;
}
float *T3(h,t,n)
{
    float h[n][2], t;
    float T[5], *d;
    T[0] = 1.0/6.0*h[4][0]/(t-1.0)*(1.0-t);  
    T[1] = 1.0/6.0*h[4][0]/(t-1.0)*(1.0-t);  
    T[2] = 1.0/6.0*h[4][0]/(t-1.0)*(1.0-t);  
    T[3] = 1.0/6.0*h[4][0]/(t-1.0)*(1.0-t);  
    T[4] = 1.0/6.0*h[4][0]/(t-1.0)*(t+1.0);  
    d = & T[0];
    return d;
}

/*/ T3 FUNCTION

float *T4(h,t,n)
{
    float h[n][2], t;
    float T[4], *d;
    T[0] = 1.0/6.0*t*(3.0-t)*(t-1.0)*(t-2.0);  
    T[1] = 1.0/6.0*t*(3.0-t)*(t-1.0)*(t-2.0);  
    T[2] = 1.0/6.0*t*(3.0-t)*(t-1.0)*(t-2.0);  
    T[3] = 1.0/6.0*t*(3.0-t)*(t-1.0)*(t-2.0);  
    d = & T[0];
    return d;
}

/*/ T4 FUNCTION

float *T5(h,t,n)
{
    float h[n][2], t;
    float T[4], *d;
    T[0] = 1.0/6.0*h[5][1]/(t-1.0)*(t-2.0);  
    T[1] = 1.0/6.0*h[5][1]/(t-1.0)*(t-2.0);  
    T[2] = 1.0/6.0*h[5][1]/(t-1.0)*(t-2.0);  
    T[3] = 1.0/6.0*h[5][1]/(t-1.0)*(t-2.0);  
    d = & T[0];
    return d;
}

/*/ T5 FUNCTION

float *V1(s,t) n'
{
    int T,
    float b h[1][1], u;
}
{
float v[6], *d;

v[0] = 1.0/6.0*a[5][0]*u*(u-1.0)*u;
v[1] = 1.0/3.0*a[5][1]*u*(u-1.0)*(4.0-u);
v[2] = 1.0/18.0*(12.0*(u-1.0)*u*(1.0-u)+(3.0-u)*(u-1.0)*(u-2.0));
v[3] = 1.0/18.0*u*(u*u-9.0*u+26);
v[4] = 1.0/3.0*a[5][1]*u*(u-1.0)*(5.0-u);
v[5] = 1.0/6.0*a[5][0]*u*(u-1.0)*(u-2.0);
d = v[0];
return d;
}

/* U2 FUNCTION-----------------------------------------------*/
float *U2(s, u, n)
float s[n][2], u;
{
float v[5], *d;

v[0] = 1.0/6.0*a[4][0]*u*(u-1.0)*u;
v[1] = 1.0/3.0*a[4][1]*u*(u-1.0)*(4.0-u);
v[2] = (1.0-u)*(u+1.0);
v[3] = 1.0/6.0*(u+1.0)*(4.0-u)*u;
v[4] = 1.0/6.0*a[4][0]*u*(u+1.0)*(u-1.0);
d = v[0];
return d;
}

/* U3 FUNCTION-----------------------------------------------*/
float *U3(s, u, n)
float s[n][2], u;
{
float v[5], *d;

v[0] = 1.0/6.0*a[3][0]*u*(u-1.0)*(2.0-u);
v[1] = 1.0/6.0*(1.0-u)*(u+3.0)*(2.0-u);
v[2] = u*(2.0-u);
v[3] = 1.0/2.0*a[3][1]*u*(u-1.0)*(5.0-u)*u;
v[4] = 1.0/6.0*a[3][0]*u*(u-2.0)*(u-1.0);
d = v[0];
return d;
}

/* U4 FUNCTION-----------------------------------------------*/
float *U4(s, u, n)
float s[n][2], u;
{
float v[4], *d;

v[0] = 1.0/6.0*(3.0-u)*(u-1.0)*(u-2.0);
v[1] = 2/2.0*u*(3.0-u)*(2.0-u);
v[2] = 2/2.0*a[0][1]*u*(3.0-u)*(u-1.0);
v[3] = 2/6.0*a[0][1]*u*u*(u-1.0)*(u-2.0);
\[ d = 4U[0]; \]
\[ \text{return } d; \]

\[
\text{/** U5 FUNCTION---------------------------------------------*/}
\]

\[
\text{float } *05(s,u) \]
\[
\text{float } s[n][2], u; \]
\[
\{ \text{float } U[4], *d; \]
\[
U[0] = 1.0/6.0*s[5][0]*(1.0-u)*(u+1.0)*u; \]
\[
U[1] = 1.0/2.0*s[5][0]*u*(u-1.0)*(2.0+u); \]
\[
U[2] = 1.0/2.0*(u+2.0)*(1.0-u)*(u+1.0); \]
\[
U[3] = 1.0/6.0*(u+1.0)*(u+2.0)*u; \]
\[
d = s[0]; \]
\[
\text{return } d; \}
\]

\[
\text{/** FUNCTION FOR THE TENSOR PRODUCT--------------------------------------------*/}
\]

\[
\text{float } \text{mult}(T,x1),bl,el,be,ec,U,n,k) \]
\[
\text{int } n,k; \]
\[
\text{float } T[n],U[6]; \]
\[
\text{float } x1[n][k]; \]
\[
\text{int } bl,el,be,ec; \]
\[
\{ \text{float } L[6]; \]
\[
\text{float } P; \]
\[
\text{int } i,j; \]
\[
\text{for } (i = 0; i <= ec-be; i++) \}
\[
\{ \text{L}[L] = 0.0; \]
\[
\text{for } (j = 0; j <= el-bl; j++) \}
\[
\{ \text{L}[i] = L[i] + T(j)*x1[j+bl][i+be]; \}
\]
\[
P = 0.0; \]
\[
\text{for } (j = 0; j <= ec-be; j++) \}
\[
\{ \text{P} = P +U[j]*L[j]; \}
\]
\[
\text{return } P; \}? \]
Bibliography


Davis, B. (1988), Uses and Abuses of Cross Validation in Geostatistics, Mathematical Geology. 19 (3) 241-249.

de Boor, C. and Hollig, K. (1987), B-Splines without divided differences, In G. Farin, editor, Geometric Modeling Algorithms and new Trends, 21-27, SIAM.


Farin, G. (1983), Smooth interpolation to scattered 3D data, in: [Barnhill and Boehm '83].


