Available online at www.isr-publications.com/jmcs
J. Math. Computer Sci. 16 (2016), 419–434
Research Article

Online: ISSN 2008-949x
Journal of Mathematics and Computer Science
Journal Homepage: www.tjmcs.com - www.isr-publications.com/jmcs

Linear interpolation for spatial data grid by newton polynomials

Yiming Jiang[a,*], Xiaodong Hu[a], Sen Wu[a], Ancai Zhang[b], Fei Yuan[c], Lu Liu[a], Ridong Zha[a]

[a]*School of Precision Instrument and Opto-electronics Engineering, Tianjin University, Tianjin 300072, China.*

[b]*School of Automation and Electrical Engineering , Linyi University, Linyi Shandong 276000, China.*

[c]*China Electric Power Equipment and Technology co., Ltd. Beijing 100052, China.*

Abstract

This article concerns with an improved interpolation method based on Newton scheme with high dimensional data. It obtains the interpolation function when the nodes are arranged in a spatial grid. The spatial interpolation method is systematic and very useful in engineering field. As an example of application, we use the method to simulate the 3-D temperature field nearby a motor by 27 nodal temperature values. The results show that the interpolation function is quickly obtained within a limited cube space which realized data visualization. This demonstrates the effectiveness of the proposed method. ©2016 All rights reserved.

Keywords: Newton polynomials, spatial interpolation, numerical estimation, data visualization, simulation, temperature field.
2010 MSC: 97N50, 41A63, 65D05.

1. Introduction

Because of the restriction by measuring tools and working environment and other factors in engineering, it is difficult to get sampling information completely in the process of measurement. Only the countable sampling values is not enough to depict the spatial data field. Interpolation method is an effective method for data reconstruction. By interpolation calculation, inaccessible data is estimated, which improves the data density. At present, the research of interpolation method based

*Corresponding author
Email address: `tjujym@163.com` (Yiming Jiang)

Received 2016-05-26

on two-dimensional space has been intensive. With the quick development of computer technology, and appearance of 3-D data processing tools, interpolation-based data visualization of space becomes faster and easier to realize [6, 11, 12].

Spatial data interpolation is described as: given a set of known space discrete point data, trying to find a function relationship from these data to calculate the value of any other point in the region [6]. It has been widely used in meteorology, geological exploration, petroleum industry, hydrology and other fields. However, most of the spatial interpolation methods require a lot of point data to obtain an accurate solution. In contrast, the polynomial based interpolation method performs well with a few points. We also studied and referred some related interpolation methods. Refs. [1, 3, 4, 7, 13, 15] mainly researched the polynomial interpolation method of two variables, in which [13] deducted the polynomial interpolation with nodes in triangular arrangement and has a similar idea with this article. Refs. [2, 8] and [10] provided some algorithms for multivariate polynomial interpolation. Refs. [5, 9] and [14] studied the multivariate Birkhoff interpolation method which can be used in three-dimensional interpolation.

Newton interpolation is an important part of polynomial interpolation. It inherited the Lagrange interpolation, and inspired the Hermite interpolation. So the Newton interpolation method is usually considered to make a simple uniform to polynomial interpolation. The advantage lies in that while each basis function of Lagrange interpolation polynomial changes along with the increase or decrease of nodes, the difference quotients (also known as Newton's quotients) of Newton interpolation do not. By calculating difference quotient of each order as Newton's method, we can obtain the interpolation polynomial with systematic and orderly form step by step. Therefore, it has been widely used in scientific computing and engineering [16]. However, as polynomial interpolation methods, their common drawback is that they cannot keep stable in the infinite domain. That is the problem which should be solved.

Normally the Newton interpolation method is only used in one-dimensional interpolation of a single variable. However, by extending the form of difference quotient, the interpolation function in two-dimensional and three-dimensional cases is derived by relative properties. To solve this problem, we study the extension of Newton interpolation method in spatial space. First, we introduce the basic Newton interpolation method and its principle in Section 2. After that, in Section 3 we put forward an improved method to make sure the interpolation function converges to constant value when $x \to \infty$. In Section 4, we define the 2-D (two-dimensional) and 3-D (three-dimensional) difference quotients, respectively, and we obtain the corresponding interpolation formula using Newton's method. Finally, in Section 5, we use the improved spatial Newton interpolation method to simulate the temperature field nearby a motor by 27 nodal temperature values and drew the contour plot. The proposed spatial interpolation method can be widely used in engineering simulation and data fitting. The focus of this article is to introduce the Newton interpolation as a spatial interpolation method. Systemic, scientific, convenient and quick, this kind of method can compute the approximate function accurately with few nodes.

2. Newton interpolation [16] and its improvement

2.1. Difference quotient

Suppose the values of function $f(x)$ at different nodes $x_0, x_1, \dots, x_n$ (not have to be put in order of size) on x-axis are $f(x_0), f(x_1), \dots, f(x_n)$. Define the mathematical form as (2.1).

$$\frac{Df(x)}{Dx}\Big|_{x=x_i,x_{i+1}} = f[x_i, x_{i+1}] = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}, \tag{2.1}$$

where $\{i,(i+1)\}\in\{0,1,\ldots,n\}$. $\frac{D^n f(x)}{Dx^n}$ $(n=1)$ denotes the difference function as a discrete function of x_i, x_{i+1}. $f\left[x_i,x_j\right]$ is called the 1st order difference quotient of $f(x)$ on nodes x_i, x_{i+1}. Similarly, the 2nd order difference quotient is seen as the "difference quotient" of $f\left[x_i,x_{i+1}\right]$ as (2.2).

$$\frac{D^2 f(x)}{Dx^2}\Big|_{x=x_i,x_{i+1},x_{i+2}}=f\left[x_i,x_{i+1},x_{i+2}\right]=\frac{f\left[x_{i+1},x_{i+2}\right]-f\left[x_i,x_{i+1}\right]}{x_{i+2}-x_i}, \tag{2.2}$$

where $\{i,(i+1),(i+2)\}\in\{0,1,\ldots,n\}$.$f\left[x_i,x_{i+1},x_{i+2}\right]$ is defined as the 2nd order difference quotient of $f(x)$ on nodes x_i, x_{i+1}, x_{i+2}. By continuing this process, we can deduce the rest by such iterative format as (2.3). In general, we call such mathematical form (2.3) the r-th order difference quotient of $f(x)$ on nodes x_i, x_{i+1}, ... , x_{i+r}.

$$\frac{D^r f(x)}{Dx^r}\Big|_{x=x_i,\ldots,x_{i+r}}=f\left[x_i,x_{i+1},\ldots,x_{i+r}\right]=\frac{f\left[x_{i+1},\ldots,x_{i+r}\right]-f\left[x_i,\ldots,x_{i+r-1}\right]}{x_{i+r}-x_i}. \tag{2.3}$$

In particular, the 0th order difference quotient of $f(x)$ at nodes x_i is defined as $f(x_i)$.

$$\frac{D^0 f(x)}{Dx^0}\Big|_{x=x_i}=f\left[x_i\right]=f(x_i).$$

2.2. Properties of difference quotient

(1) The r-th order difference quotient of $f(x)$ as $f\left[x_0,x_1,\ldots,x_r\right]$ is expressed as the linear combination of $f(x_0)$, $f(x_1)$, ... , $f(x_r)$,

$$f\left[x_0,x_1,\ldots,x_r\right]=\sum_{i=0}^{r}\frac{f(x_i)}{\frac{d\omega_{r+1}(x)}{dx}\big|_{x=x_i}},$$

where

$$\omega_{r+1}(x)=\prod_{i=0}^{r}(x-x_i).$$

(2) The difference quotient is independent of the order of nodes. For any x_{i_1}, x_{i_2},

$$f\left[x_0,\ldots,x_{i_1},\ldots,x_{i_2},\ldots,x_r\right]=f\left[x_0,\ldots,x_{i_2},\ldots,x_{i_1},\ldots,x_r\right].$$

(3) A new difference quotient can be constructed with a new node x added.

$$f\left[x_0,\ldots,x_r,x\right]=\frac{f\left[x_0,\ldots,x_{r-1},x\right]-f\left[x_0,\ldots,x_{r-1},x_r\right]}{x-x_r}.$$

(4) If $f(x)$ is polynomial of degree n, $f\left[x_0,x_1,\ldots,x_{r-1},x\right]$ is polynomial of degree $(n-r)$ when $r\le n$, and $f\left[x_0,x_1,\ldots,x_{r-1},x\right]\equiv 0$ when $r>n$.

(5) The order of the difference quotient is no more than the number of nodes minus 1.

2.3. Newton's interpolation formula

By property (3) in Subsection 2.2,

$$f\left[x_0,\ldots,x_{r-1},x\right]=f\left[x_0,\ldots,x_{r-1},x_r\right]+f\left[x_0,\ldots,x_{r-1},x_r,x\right](x-x_r),$$

such relation is obtained as

$$f(x)=\sum_{r=0}^{n}\left(f\left[x_0,\ldots,x_r\right]\omega_r(x)\right)+f\left[x_0,\ldots,x_n,x\right]\omega_{n+1}(x),$$

where $\omega_r(x)=\frac{\omega_{r+1}(x)}{x-x_r}=\frac{\prod_{i=0}^{r}(x-x_i)}{x-x_r}$, the same below. Define

$$N_n(x) = \sum_{r=0}^{n} \left(f[x_0, \ldots, x_r] \, \omega_r(x) \right). \tag{2.4}$$

In (2.4), $N_n(x)$ denotes the Newton interpolation polynomial of degree n, and $N_n(x_i) = f(x_i)$ $(i = 0, 1, 2, \ldots, n)$. Specially,

$$N_0(x) = f(x_0).$$

It can be concluded that the Newton interpolation method uses the difference quotient as a tool to approach the actual function. The result is a linear function of a single variable of degree n. With new nodal values added to the original data, we are able to obtain the new interpolation function by extending the original function with basis functions.

3. Improvement on Newton's interpolation

3.1. Construct the base-function to improve the convergence

Apparently the Newton interpolation with $(n+1)$ nodes is to get the n-th degree function of variable x. When $x \to \infty$, $N_n(x)$ diverges to infinity, and that is contrary to normal conditions. Here we introduce a method for improving Newton's method, it will make the curve converge.

Construct function $M_n(x)$,

$$M_n(x) = O(x^n), \tag{3.1}$$

$M_n(x)$ is described by the "Big O Notation", if

$$f(x) = O(g(x)) \text{ as } x \to \infty,$$

there exists a positive real number C and x_c.

$$|f(x)| \le |C \cdot (g(x))| \text{ for all } x \ge x_c$$

and

$$\begin{cases} |M_n(x)| \le L, \\ M_n(x) \ne 0, \end{cases} \text{for all } x \in (-\infty, \infty), \tag{3.2}$$

where L is a positive real number. On each node, the values of $M_n(x)$ are $M_n(x_0)$, $M_n(x_1)$,..., $M_n(x_n)$. Then change the values of each node into

$$f^*(x_i) = f(x_i) M_n(x_i).$$

Reuse the Newton's method to construct interpolating function, by (2.4), such as

$$N_n^*(x) = \sum_{r=0}^{n} \left(f^*[x_0, \ldots, x_r] \, \omega_r(x) \right),$$

where

$$f^*[x_i, x_{i+1}] = \frac{f^*(x_{i+1}) - f^*(x_i)}{x_{i+1} - x_i},$$

$$f^*[x_i, x_{i+1}, \ldots, x_{i+r}] = \frac{f^*[x_{i+1}, \ldots, x_{i+r}] - f^*[x_i, \ldots, x_{i+r-1}]}{x_{i+r} - x_i}.$$

Such the final interpolating function $F(x)$ for $f(x)$ should be

$$F_n(x) = \frac{N_n^*(x)}{M_n(x)}.$$

$F_n(x)$ converges when $x \to \infty$. And for $i = 0, 1, \ldots, n$, $F(x_i) = f(x_i)$.

Example 3.1. Suppose a function $f(x)$ in real field with unknown expression. We define three nodes as $x_0 = -1$, $x_1 = 0$, $x_2 = 1$, with $f(x_0) = f(-1) = 1$, $f(x_1) = f(0) = 0.5$, $f(x_2) = f(1) = 2$. Firstly, we use traditional Newton interpolation method.

$$\begin{aligned} f[x_0, x_1] = f[-1, 0] &= \frac{0.5 - 1}{0 - (-1)} = -0.5, \\ f[x_0, x_1, x_2] = f[-1, 0, 1] &= \frac{1.5 - (-0.5)}{1 - (-1)} = 1. \end{aligned}$$

Then

$$\begin{aligned} N_2(x) &= f(-1) + f[-1, 0](x + 1) + f[-1, 0, 1]\, x(x + 1) \\ &= 1 - 0.5(x + 1) + x(x + 1) \\ &= x^2 + 0.5x + 0.5. \end{aligned}$$

After that, we use the improved new method to fit $f(x)$. Construct $M_2(x)$ which satisfies (3.1) and (3.2). Set

$$M_2(x) = 1 + x^2,\ x \in (-\infty, \infty).$$

Such $M_2(x_0) = M_2(-1) = 2$, $M_2(x_1) = M_2(0) = 1$, $M_2(x_2) = M_2(1) = 2$, and $f^*(x_0) = f(-1)M_2(-1) = 2$, $f^*(x_1) = f(0)M_2(0) = 0.5$, $f^*(x_2) = f(1)M_2(1) = 4$. Then

$$\begin{aligned} N_2^*(x) &= f^*(-1) + f^*[-1, 0](x + 1) + f^*[-1, 0, 1]\, x(x + 1) \\ &= 2 - 1.5(x + 1) + 2.5x(x + 1) \\ &= 2.5x^2 + x + 0.5. \end{aligned}$$

Finally, we get

$$F_2(x) = \frac{N_2^*(x)}{M_2(x)} = \frac{2.5x^2 + x + 0.5}{1 + x^2}.$$

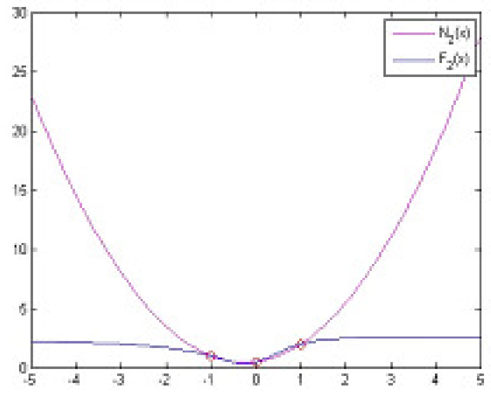

Figure 1: Comparison of curves $N_2(x)$ and $F_2(x)$ in the example.

Figure 1 shows that when $|x|$ increases, $F_2(x)$ tends to constant value while $N_2(x)$ tends to infinity. $M_2(x)$ is to reduce the degree of $N_2(x)$.

4. Extension in the high dimensions

4.1. Definition of 2-D difference quotient

Suppose $f(x,y)$ is a 2-D function in the x-y coordinate system. Take (n_1+1) nodes $x_0, x_1, ..., x_{n_1}$ on the x axis and extend them in the direction of y axis to form (n_1+1) parallel lines. Take (n_2+1) nodes $y_0, y_1, ..., y_{n_2}$ on the y axis and extend them in the direction of x axis to form (n_2+1) parallel lines. The two groups of parallel lines structure an $(n_1+1)\times(n_2+1)$ grid with $(n_1+1)(n_2+1)$ nodes, and the corresponding value of $f(x,y)$ should be $f(x_0,y_0)$, $f(x_1,y_0)$, ... , $f(x_{n_1},y_{n_2})$. Define the mathematical form

$$\left.\frac{Df(x,y)}{Dx}\right|_{\substack{x=x_i,x_{i+1}\\ y=y_j}} = f([x_i,x_{i+1}],y_j) = \frac{f(x_{i+1},y_j)-f(x_i,y_j)}{x_{i+1}-x_i}, \tag{4.1}$$

where $\{i,(i+1)\}\in\{0,1,\ldots,n_1\}$, $j\in\{0,1,\ldots,n_2\}$.

In the formula (4.1), $f([x_i,x_{i+1}],y_j)$ is defined as difference quotient of 1st order of x at $x = x_i, x_{i+1}$ and $y = y_j$. In the same way,

$$\left.\frac{Df(x,y)}{Dy}\right|_{\substack{x=x_i\\ y=y_j,y_{j+1}}} = f(x_i,[y_j,y_{j+1}]) = \frac{f(x_i,y_{j+1})-f(x_i,y_j)}{y_{j+1}-y_j},$$

where $i\in\{0,1,\ldots,n_1\}$, $\{j,j+1\}\in\{0,1,\ldots,n_2\}$. And $f(x_i,[y_j,y_{j+1}])$ is defined as the difference quotient of 1st order of y at $x = x_i$ and $y = y_j, y_{j+1}$.

$$\begin{aligned}\left.\frac{D^2f(x,y)}{DxDy}\right|_{\substack{x=x_i,x_{i+1}\\ y=y_j,y_{j+1}}} &= f([x_i,x_{i+1}],[y_j,y_{j+1}])\\ &= \frac{f(x_{i+1},[y_j,y_{j+1}])-f(x_i,[y_j,y_{j+1}])}{x_{i+1}-x_i}\\ &= \frac{f([x_i,x_{i+1}],y_{j+1})-f([x_i,x_{i+1}],y_j)}{y_{j+1}-y_j},\end{aligned}$$

where $\{i,i+1\}\in\{0,1,\ldots,n_1\}$, $\{j,j+1\}\in\{0,1,\ldots,n_2\}$. And $f([x_i,x_{i+1}],[y_j,y_{j+1}])$ is defined as the difference quotient of 1st order of x and 1st order of y (later expressed as the x-y difference quotient) at $x = x_i, x_{i+1}$ and $y = y_j, y_{j+1}$. Generally,

$$\begin{aligned}&\left.\frac{D^{r+s}f(x,y)}{Dx^rDy^s}\right|_{\substack{x=x_i,\ldots,x_{i+r}\\ y=y_j,\ldots,y_{j+s}}}\\ &= f([x_i,x_{i+1},\ldots,x_{i+r}],[y_j,y_{j+1},\ldots,y_{j+s}])\\ &= \frac{f([x_{i+1},\ldots,x_{i+r}],[y_j,\ldots,y_{j+s}])-f([x_i,\ldots,x_{i+r-1}],[y_j,\ldots,y_{j+s}])}{x_{i+r}-x_i}\\ &= \frac{f([x_i,\ldots,x_{i+r}],[y_{j+1},\ldots,y_{j+s}])-f([x_i,\ldots,x_{i+r}],[y_j,\ldots,y_{j+s-1}])}{y_{j+s}-y_j},\end{aligned} \tag{4.2}$$

where $\{i, \ldots, i+r\} \in \{0, 1, \ldots, n_1\}$, $\{j, j+s\} \in \{0, 1, \ldots, n_2\}$.

In (4.2), $f\left(\left[x_i, x_{i+1}, \ldots, x_{i+r}\right], \left[y_j, y_{j+1}, \ldots, y_{j+s}\right]\right)$ denotes the 2-D x^r-y^s difference quotient of r-th order of x and s-th order of y at $x = x_i, x_{i+1}, \ldots, x_{i+r}$, $y = y_j, y_{j+1}, \ldots, y_{j+s}$.

Specially, when r or s are equal to zero, it turns into the normal difference quotient as

$$\begin{cases} \left.\frac{D^r f(x,y)}{Dx^r}\right|_{\substack{x = x_i, \ldots, x_{i+r} \\ y = y_j}} = f\left(\left[x_i, x_{i+1}, \ldots, x_{i+r}\right], y_j\right), \\ \left.\frac{D^s f(x,y)}{Dy^s}\right|_{\substack{x = x_i \\ y = y_j, \ldots, y_{j+s}}} = f\left(x_i, \left[y_j, y_{j+1}, \ldots, y_{j+s}\right]\right). \end{cases}$$

The x^r-y^s difference quotient is expressed by the 'violet' vector in Fig. 2, and the vectors in 'red' and 'blue' express the r-th and s-th order difference quotient of x, y, respectively.

4.2. Properties of 2-D difference quotient

Think of the 2-D difference quotient as the difference quotient of 1-D traditional difference quotient. It has the following properties.

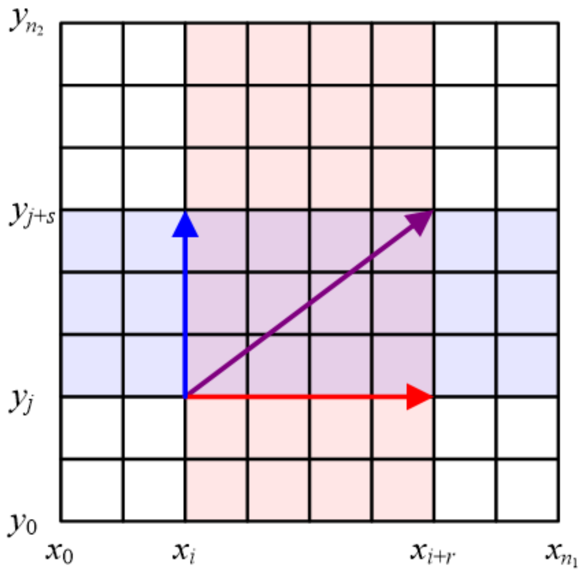

Figure 2: The x^r-y^s difference quotient diagram in vector form.

(1) The difference quotient of each variable is independently calculated, and that of each variable meets the properties (1)-(5) in Subsection 2.2.

(2) If the nodes construct a grid (here defined as the matrix of interpolation nodes) as Fig. 2 shows, the x^r-y^s difference quotient exists as shown in (4.2). The x^r-y^s difference quotient with r-th order of x and s-th order of y can be obtained by calculating the difference quotient along

the x direction for order r and along the y direction for order s. The sequence of derivation from x or y does not affect the result.

$$\frac{D^{r+s} f(x,y)}{Dx^r Dy^s} = \frac{D^r\left(\frac{D^s f(x,y)}{Dy^s}\right)}{Dx^r} = \frac{D^s\left(\frac{D^r f(x,y)}{Dx^r}\right)}{Dy^s}.$$

(3) If the nodes $(x_i, y_i)\,(i = 0, 1, 2, \ldots, n)$ construct an $(n+1)\times(n+1)$ matrix in the x-y plane, $g(x)$ is a function of real domain and each node on y axis satisfies $y_i = g(x_i)$ ($\{x_i\}$ belongs to the domain of function $g(x)$, $\{x_i\} \to \{g(x_i)\}$ is a bijection). The relation (4.3) can transform 2-D difference quotient into traditional difference quotient.

$$\begin{aligned}&\frac{D^r f(x, g(x))}{Dx^r}\Big|_{x=x_0,\ldots,x_r}\\ &= \sum_{i=0}^{n}\sum_{j=0}^{n}\left(\frac{D^{i+j} f(x,y)}{Dx^i Dy^j}\Bigg|_{\substack{x=x_0,\ldots,x_i\\ y=y_0,\ldots,y_j}} \cdot \frac{D^{r-i}\omega_j(g(x))}{Dx^{r-i}}\Big|_{x=x_i,\ldots,x_r}\right),\end{aligned} \tag{4.3}$$

where, $r \geq n$, $\omega_j(g(x)) = \prod_{i=0}^{j-1}(g(x) - g(x_i))$.

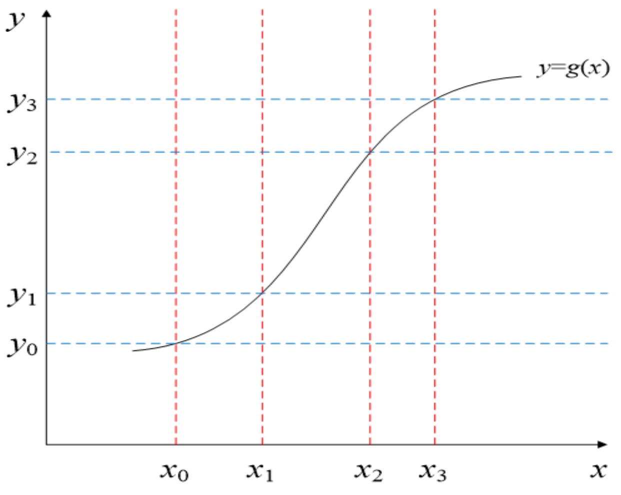

Figure 3: Distribution of nodes $(x_i, y_i)\,(i = 0, 1, 2, 3)$ through $y = g(x)$.

4.3. Newton interpolation formula in 2-D space

According to property (1) of the above and property (3) of Subsection 2.2,

$$\begin{aligned}&f([x_0,\ldots,x_{r-1},x],[y_0,\ldots,y_{s-1},y])\\ &\quad= f([x_0,\ldots,x_r],[y_0,\ldots,y_{s-1},y]) + f([x_0,\ldots,x_r,x],[y_0,\ldots,y_{s-1},y])(x-x_r)\\ &\quad= f([x_0,\ldots,x_{r-1},x],[y_0,\ldots,y_s]) + f([x_0,\ldots,x_{r-1},x],[y_0,\ldots,y_s,y])(y-y_s).\end{aligned} \tag{4.4}$$

Furthermore, by (4.4),

$$\begin{aligned}&f([x_0,\ldots,x_{r-1},x],[y_0,\ldots,y_{s-1},y])\\ &\quad= f([x_0,\ldots,x_r],[y_0,\ldots,y_s]) + f([x_0,\ldots,x_r,x],[y_0,\ldots,y_s])(x-x_r)\\ &\qquad+ f([x_0,\ldots,x_r],[y_0,\ldots,y_s,y])(y-y_s)\\ &\qquad+ f([x_0,\ldots,x_r,x],[y_0,\ldots,y_s,y])(x-x_r)(y-y_s).\end{aligned}$$

Similar to the derivation process of traditional interpolation formula, we can obtain the general 2-D Newton interpolation function as

$$N_{n_1,n_2}(x,y)=\sum_{r=0}^{n_1}\sum_{s=0}^{n_2}\left(f\left([x_0,\dots,x_r],[y_0,\dots,y_s]\right)\omega_r(x)\,\omega_s(y)\right)$$

and the remainder

$$\begin{aligned}E_{n_1,n_2}(x,y)&=\sum_{s=0}^{n_2}\left(f\left([x_0,\dots,x_{n_1},x],[y_0,\dots,y_s]\right)\omega_s(y)\right)\omega_{n_1+1}(x)\\&\quad+\sum_{r=0}^{n_1}\left(f\left([x_0,\dots,x_r],[y_0,\dots,y_{n_2},y]\right)\omega_r(x)\right)\omega_{n_2+1}(y)\\&\quad+f\left([x_0,\dots,x_{n_1},x],[y_0,\dots,y_{n_2},y]\right)\omega_{n_1+1}(x)\,\omega_{n_2+1}(y).\end{aligned}\tag{4.5}$$

Also we noticed that in (4.5),

$$\sum_{s=0}^{n_2}\left(f\left([x_0,\dots,x_{n_1},x],[y_0,\dots,y_s]\right)\omega_s(y)\right)=\frac{D^{n_1+1}N_{0,n_2}(x,y)}{Dx^{n_1+1}}\Big|_{x=x_0,\dots,x_{n_1},x},$$

$$\sum_{r=0}^{n_1}\left(f\left([x_0,\dots,x_r],[y_0,\dots,y_{n_2},y]\right)\omega_r(x)\right)=\frac{D^{n_2+1}N_{n_1,0}(x,y)}{Dy^{n_2+1}}\Big|_{y=y_0,\dots,y_{n_2},y},$$

where, $N_{0,n_2}(x,y)\,(x=x_0,x_1,\dots,x_{n_1})$ is the y Newton interpolation function, which is an x discrete function and $N_{n_1,0}(x,y)\,(y=y_0,y_1,\dots,y_{n_2})$ is the x Newton interpolation function, which is an y discrete function. Equal to (4.5),

$$\begin{aligned}E_{n_1,n_2}(x,y)&=\omega_{n_1+1}(x)\frac{D^{n_1+1}N_{0,n_2}(x,y)}{Dx^{n_1+1}}\Big|_{x=x_0,\dots,x_{n_1},x}\\&\quad+\omega_{n_2+1}(y)\frac{D^{n_2+1}N_{n_1,0}(x,y)}{Dy^{n_2+1}}\Big|_{y=y_0,\dots,y_{n_2},y}\\&\quad+\omega_{n_1+1}(x)\,\omega_{n_2+1}(y)\frac{D^{n_1+n_2+2}f(x,y)}{Dx^{n_1+1}Dy^{n_2+1}}\Bigg|_{\substack{x=x_0,\dots,x_{n_1},x\\y=y_0,\dots,y_{n_2},y}}.\end{aligned}$$

Thus$f(x,y)=N_{n_1,n_2}(x,y)+E_{n_1,n_2}(x,y)$, where $N_{n_1,n_2}(x,y)$ is the 2-D Newton interpolation polynomial of degree (n_1+n_2) (degree n_1 of x and degree n_2 of y), and $N_{n_1,n_2}(x_i,y_j)=f(x_i,y_j)$ $(i=0,1,2,\dots,n_1;j=0,1,2,\dots,n_2)$; $E_{n_1,n_2}(x,y)$ is the remainder of interpolation. The basis function of $N_{n_1,n_2}(x,y)$ is

$$N_{(r,s)}(x,y)=f\left([x_0,x_1,\dots,x_r],[y_0,y_1,\dots,y_s]\right)\omega_r(x)\,\omega_s(y),$$

where $r=0,1,\dots,n_1$ and $s=0,1,\dots,n_2$. Thus,

$$N_{n_1,n_2}(x,y)=\sum_{r=0}^{n_1}\sum_{s=0}^{n_2}N_{(r,s)}(x,y).$$

Specially, when $n_1=n_2=0$,

$$N_{0,0}(x,y)=N_{(0,0)}(x,y)=f(x_0,y_0).$$

The 2-D Newton interpolation function is a polynomial of x-degree n_1 and y-degree n_2 with $n_1 \times n_2$ series. It can be regarded as the interpolation for interpolation functions of x on the y direction. Also, we are able to extend the 2-D function by adding new basis functions in the same with normal Newton interpolation.

For example, take 4 nodes $(0,0)$, $(2,0)$, $(0,1)$, $(2,1)$, with $f(0,0) = 0.5$, $f(2,0) = 1$, $f(0,1) = 0.75$, $f(2,1) = 0.25$. By (4.2), we get

$$\begin{aligned} f(0,[0,1]) &= \frac{f(0,1) - f(0,0)}{1-0} = 0.25, \\ f([0,2],0) &= \frac{f(2,0) - f(0,0)}{2-0} = 0.25, \\ f([0,2],1) &= \frac{f(2,1) - f(0,1)}{2-0} = -0.25, \\ f([0,2],[0,1]) &= \frac{f([0,2],1) - f([0,2],0)}{1-0} = -0.5, \\ N_{1,1}(x,y) &= f(0,0) + f([0,2],0)\,x + f(0,[0,1])\,y + f([0,2],[0,1])\,xy \\ &= 0.5 + 0.25x + 0.25y - 0.5xy. \end{aligned}$$

4.4. Newton interpolation formula in 3-D space

Suppose $f(x,y,z)$ is a 3-D function in the x-y-z coordinate system. The nodes $\{x_i, y_j, z_k\}$ (where $i = 0,1,2,\ldots,n_1; j = 0,1,2,\ldots,n_2; k = 0,1,2,\ldots,n_3$) construct a 3-D $(n_1+1)\times(n_2+1)\times(n_3+1)$ matrix. The derivation to 3-D x^r-y^s-z^t difference quotient is similar to (2.3) and (4.2).

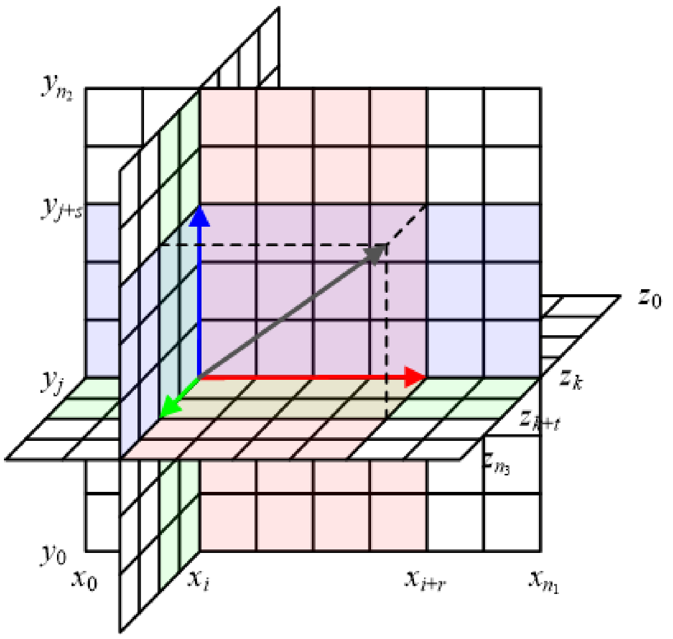

Figure 4: The x^r-y^s-z^t difference quotient diagram in vector form.

The x^r-y^s-z^t difference quotient is expressed by the 'dark gray' vector in Fig. 4, and the vectors in 'red', 'blue' and 'green' express the r-th, s-th and t-th order difference quotient of x, y, z, respectively.

We used this kind of diagram to point out that 3-D difference quotient can be regarded as the combination of difference quotients in each direction.

Specially, when r, s, or t are equal to zero, it turns into 2-D difference quotient.

$$\begin{cases} \left.\frac{D^{r+s}f(x,y,z)}{Dx^r Dy^s}\right|_{\substack{x=x_i,\dots,x_{i+r}\\ y=y_j,\dots,y_{j+s}\\ z=z_k}} = f\left(\left[x_i,\dots,x_{i+r}\right],\left[y_j,\dots,y_{j+s}\right],z_k\right), \\ \left.\frac{D^{r+t}f(x,y,z)}{Dx^r Dz^t}\right|_{\substack{x=x_i,\dots,x_{i+r}\\ y=y_j\\ z=z_k,\dots,z_{k+t}}} = f\left(\left[x_i,\dots,x_{i+r}\right],y_j,\left[z_k,\dots,z_{k+t}\right]\right), \\ \left.\frac{D^{s+t}f(x,y,z)}{Dy^s Dz^t}\right|_{\substack{x=x_i\\ y=y_j,\dots,y_{j+s}\\ z=z_k,\dots,z_{k+t}}} = f\left(x_i,\left[y_j,\dots,y_{j+s}\right],\left[z_k,\dots,z_{k+t}\right]\right). \end{cases}$$

Also, the general 3-D Newton interpolation function is

$$N_{n_1,n_2,n_3}(x,y,z)=\sum_{r=0}^{n_1}\sum_{s=0}^{n_2}\sum_{t=0}^{n_3}\left(f\left(\left[x_0,\dots,x_r\right],\left[y_0,\dots,y_s\right],\left[z_0,\dots,z_t\right]\right)\omega_r(x)\,\omega_s(y)\,\omega_t(z)\right).$$

Define the basis function

$$N_{(r,s,t)}(x,y,z)=f\left(\left[x_0,\dots,x_r\right],\left[y_0,\dots,y_s\right],\left[z_0,\dots,z_t\right]\right)\omega_r(x)\,\omega_s(y)\,\omega_t(z).$$

Thus

$$N_{n_1,n_2,n_3}(x,y,z)=\sum_{r=0}^{n_1}\sum_{s=0}^{n_2}\sum_{t=0}^{n_3}N_{(r,s,t)}(x,y,z). \tag{4.6}$$

Specially, when $n_1=n_2=n_3=0$,

$$N_{0,0,0}(x,y,z)=N_{(0,0,0)}(x,y,z)=f\left(x_0,y_0,z_0\right). \tag{4.7}$$

In (4.6), n_1,n_2,n_3 are the orders of interpolation in all directions. If only one of n_1,n_2,n_3 is zero, the interpolation function is equal to the 2-D form; if two of n_1,n_2,n_3 are zero, the interpolation function is equal to the basic form (1-D interpolation). They can be regarded as the special cases of 3-D interpolation.

5. Application for temperature field

Heat source and temperature field around always has a subtle and inevitable effect on the mechanical structure. Usually because of the material coefficient of thermal expansion is limited, a small deformation caused by temperature changes is usually neglected. But for scanning probe microscopy (SPM), deformation due to the temperature can influence measurement results to a relative great extent, which has become a major source of measurement error. Therefore, the corresponding compensation based on real-time observation of the spatial distribution of temperature is the key to eliminate errors. In this chapter, we used the Newton interpolation method to deal with 27 nodes data and the purpose was to simulate the temperature distribution and realize data visualization.

As an obvious character of Newton interpolation method, the expanded interpolation function can be obtained on the original basis with node expansion. The process is similar to basic Newton interpolation, but the form is a bit more complicated. In this chapter, we discussed the interpolation of 27 nodes by the method mentioned above.

5.1. Measurement on given nodes

The SPM system used in experiments mainly includes SPM, CCD monitor system, PSD, data acquisition card, computer, step motor, step motor driver and micro displacement platform, shown as Fig. 5 (a). The whole set of instruments is placed in a thermostat, in which the constant temperature was set to be 27°. The temperature value on each node was measured by thermistor sensors distributed equidistantly, and the data was collected when it was stabilized. The axes and the coordinate of each node have been defined, shown as Fig. 5 (b), in which the x and z axes were set to be parallel to the plane of backboard, and the temperature on each node was measured at the same time. The relative position of the $3 \times 3 \times 3$ matrix constituted by nodes is shown in Fig. 5 (c).

5.2. Computing process

First of all, we need to investigate the properties of the temperature field. For spacial temperature function $T(x, y, z)$ created by constant heat source, when $x, y, z \to \infty$, the difference between $T(x, y, z)$ and ambient temperature T_0 varies inversely with distance.

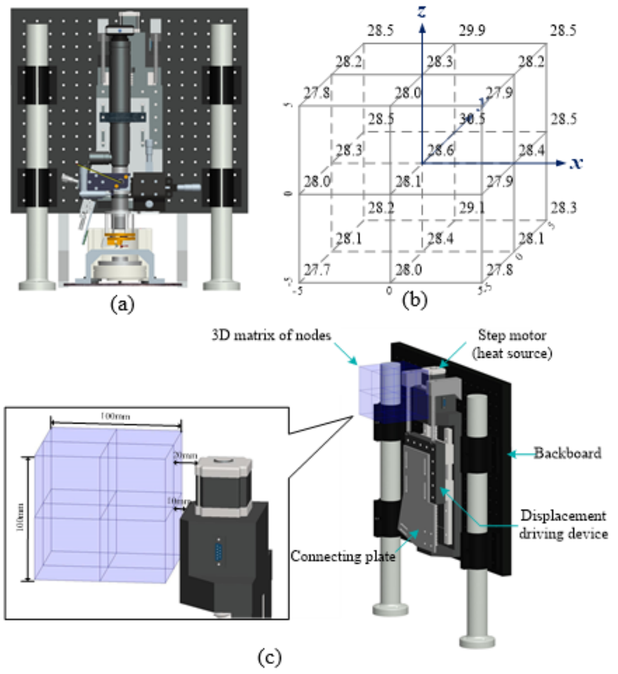

Figure 5: Temperature data collected and site environmental conditions.

$$T(x,y,z) - T_0 = O\left(\frac{1}{\sqrt{x^2+y^2+z^2}}\right) \text{ as } x,y,z \to \infty.$$

By (4.7), the Newton interpolation function with 27 nodes has the highest order term $x^2y^2z^2$. The $M_{3,3,3}(x,y,z)$ that satisfies the conditions is

$$M(x,y,z) = O\left(x^2y^2z^2\sqrt{x^2+y^2+z^2}\right) \text{ as } x,y,z \to \infty.$$

According to actual situation, the heat source focuses nearby point $(0,9,2)$, so we define

$$M(x,y,z) = \left(1+\alpha x^2(y-9)^2(z-2)^2\right)\left(1+\beta\sqrt{x^2+(y-9)^2+(z-2)^2}\right),$$

where we set α=10^{-6}, β=0.01 to make sure the $M_{3,3,3}(x,y,z)$ does not vary too much during the given nodes. The values of $M_{3,3,3}(x_i,y_j,z_k)$ on each node are listed as Table 1. By conversion formula (3.1),

$$\Delta T^*(x_i,y_j,z_k) = \Delta T(x_i,y_j,z_k)\, M_{3,3,3}(x_i,y_j,z_k),$$

where $\Delta T(x_i,y_j,z_k)$=$T(x_i,y_j,z_k) - T_0$, $T_0 = 27$.

Table 1: $M(x_i,y_j,z_k)$ on each node.

M	x_i	y_j	z_k	M	x_i	y_j	z_k
1.092	0	0	0	1.205	0	-5	5
1.114	5	0	0	1.236	5	0	-5
1.045	0	5	0	1.081	0	5	-5
1.095	0	0	5	1.075	-5	5	5
1.069	5	5	0	1.202	5	-5	5
1.127	5	0	5	1.116	5	5	-5
1.05	0	5	5	1.173	-5	-5	0
1.075	5	5	5	1.236	-5	0	-5
1.114	-5	0	0	1.157	0	-5	-5
1.141	0	-5	0	1.202	-5	-5	5
1.114	0	0	-5	1.116	-5	5	-5
1.069	-5	5	0	1.444	5	-5	-5
1.127	-5	0	5	1.444	-5	-5	-5
1.173	5	-5	0				

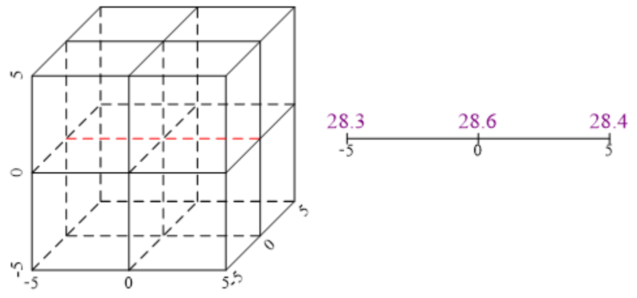

Figure 6: Interpolation on the line.

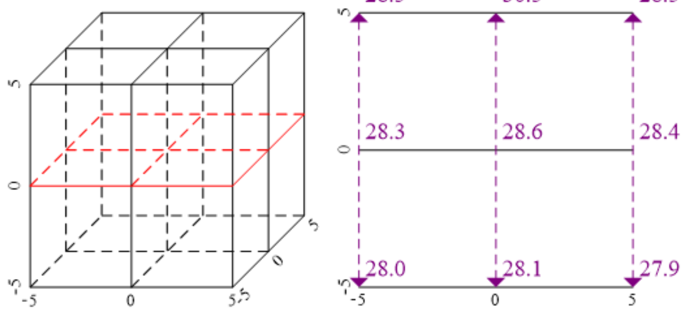

Figure 7: Interpolation on the plane.

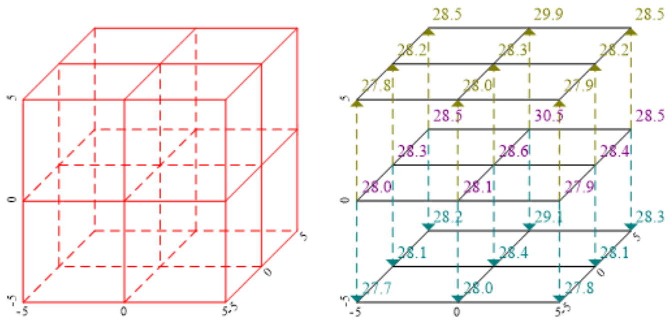

Figure 8: Interpolation on the cube.

Using the Newton interpolation in (4.6) for $\Delta T^{*}\left(x_i, y_j, z_k\right)$, finally we get

$$\hat{T}\left(x,y,z\right) = T_0 + \frac{N^{*}_{2,2,2}\left(x,y,z\right)}{M\left(x,y,z\right)},$$

where $N^{*}_{2,2,2}\left(x,y,z\right)$ is the linear Newton interpolation function for the temperature distribution. Fig. 9 is the image of $\hat{T}\left(x,y,z\right)$ plotted by MATLAB. $\hat{T}\left(x,y,z\right)$ is the simulation to temperature distribution function.

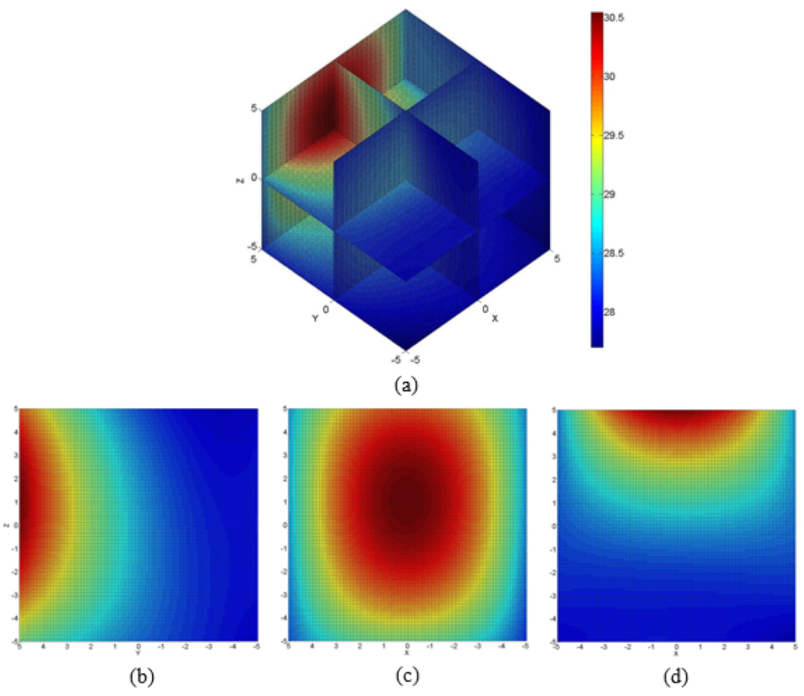

Figure 9: Contour plot of $\hat{T}\,(x, y, z)$ by slices.

Fig. 9 (a) shows the 3-D image of $\hat{T}\,(x, y, z)$ ($\hat{T}\,(x, y, z)$ is the simulation on $T\,(x, y, z)$, where $x, y, z \in [-5, 5]$) with slices ($x = 0, 5$), ($y = 0, 5$), ($z = -5, 0$). Fig. 9 (b) is the image on slice ($x = 0$). Fig. 9 (c) is the image on slice ($y = 5$). Fig. 9 (d) is the image on slice ($z = 0$). From the contour picture we saw that simulated temperature on the side near the motor (where $y = 5$) was obviously higher than the further side, which was consistent with the empirical facts.

Acknowledgment

This work was supported by the National Nature Science Foundation of China (No.61204117) and the National High Technology Research and Development Program of China (No.2012AA041204).

References

[1] B. Bojanov, Y. Xu, *On polynomial interpolation of two variables*, J. Approx. Theory, **120** (2003), 267–282. 1

[2] C. de Boor, A. Ron, *On multivariate polynomial interpolation*, Constr. Approx., **6** (1990), 287–302. 1

[3] L. Bos, M. Caliari, S. De Marchi, Y. Xu, *Bivariate Lagrange interpolation at the Padua points: the generating curve approach*, J. Approx. Theory, **143** (2006), 15–25. 1

[4] M. Caliari, S. De Marchi, M. Vianello, *Bivariate polynomial interpolation on the square at new nodal sets*, Appl. Math. Comput., **165** (2005), 261–274. 1

[5] M. Crainic, N. Crainic, *Birkhoff interpolation with rectangular sets of nodes and with few derivatives*, East J. Approx., **14** (2008), 423–437. 1

[6] L. Du, *Comparison of interpolation methods used in thermal recovery of heavy oil and the visualization of temperature field*, Dalian University of Technology, (2008). 1

[7] R. Franke, *Scattered data interpolation: tests of some methods*, Math. Comput., **38** (1982), 181–200. 1

[8] M. Gasca, T. Sauer, *Polynomial interpolation in several variables*, Adv. Comput. Math., **12** (2000), 377–410. 1

[9] R. A. Lorentz, *Multivariate Birkhoff interpolation*, Lecture Notes in Mathematics, Springer-Verlag, Berlin, (1992). 1

[10] T. Sauer, Y. Xu, *On multivariate Lagrange interpolation*, Math. Comput., **64** (1995), 1147–1170. 1

[11] J. Su, *Research on the modeling theory, method and realization technology of the visualization-oriented temperature field FEA*, Zhejiang University, (2002). 1

[12] Z. Tang, *3-D data field visualization*, Tsinghua University Press, Beijing, (1999). 1

[13] D. N. Varsamis, N. P. Karampetakis, *On the Newton bivariate polynomial interpolation with applications*, Multidimens. Syst. Signal Process., **25** (2014), 179–209. 1

[14] X. Wang, S. Zhang, T. Dong, *Newton basis for multivariate Birkhoff interpolation*, J. Comput. Appl. Math., **228** (2009), 466–479. 1

[15] R. Wu, T. Wu, H. Li, *A family of multivariate multiquadric quasi-interpolation operators with higher degree polynomial reproduction*, J. Comput. Appl. Math., **274** (2015), 88–108. 1

[16] H. Xiong, S. Zeng, Y. Mao, *Applied mathematics foundation*, Tianjin University Press, Tianjin, (1993). 1, 2