Int. j. adv. multidisc. res. stud. 2022; 2(4):431-434

Received: 09-06-2022
Accepted: 19-07-2022

International Journal of Advanced Multidisciplinary Research and Studies

ISSN: 2583-049X

The Analysis of the Advantages and Disadvantages of Construction Method of Lagrange Interpolated Polynomial and Newton Interpolated Polynomial

[1] Nguyen Thi Xuan Mai, [2] Nguyen Trung Thanh
[1, 2] Thai Nguyen University of Technology-Thai Nguyen University, Vietnam

Corresponding Author: **Nguyen Trung Thanh**

Abstract
To build a relationship between a function and a variable from a given table of $n+1$ interpolation nodes $(x_k, y_k), k = 0..n$, in Mathematics, if using numerical methods then people often build interpolated polynomials such as Lagrange interpolated polynomial, Newton interpolated polynomial, Hermit interpolated polynomial... or least squares method. However, each method has certain advantages and disadvantages. This paper will show some advantages and disadvantages of the method of constructing Lagrange interpolated polynomials and Newton interpolated polynomials.

Keywords: Interpolated Polynomial, Lagrange Interpolated Polynomial, Newton Interpolated Polynomial, Basic Largrange Polynomials

1. Introduction

Currently, in real life, production and in mathematical models, we often encounter situations: already know some of the input data $x_k(k = 0..n)$ and corresponding output data $y_k(k = 0..n)$ without determining the functional relationship between these data. Therefore, for the convenience of programming, for determining the output data corresponding to any input data..., we seek to establish a functional relationship, in there variable is input, function is output. It means: find function $f(x)$ satisfies $y_k = f(x_k), k = 0..n$. In which pairs $(x_k, y_k), k = 0..n$ are called interpolation nodes. The establishment of the function $f(x)$ from $n+1$ these interpolation nodes can be through constructing the interpolated polynomial or using the least squares method.

If we use the method of least squares, we can construct any function $f(x)$ $f(x)$ ($f(x)$ is either linear or non-linear). And if we use interpolation polynomials, we build a function $f(x)$ in the form of a polynomial function - This is the most convenient form of function for programming on computers as well as for calculating and illustrating graphs... In some cases, even though the analytic expression of the function $f(x)$ is known explicitly, but because the calculation with that function is complicated, the corresponding interpolation polynomial $P_n(x)$ is built. Then all computations with $f(x)$ are approximated by the results of calculations over $P_n(x)$.

There are three types of interpolated polynomials commonly used: Lagrange interpolated polynomial, Newton interpolated polynomial, and Hermit interpolated polynomial. The process of identifying each type of polynomial will have its own advantages and disadvantages. The article will focus on analyzing the advantages and disadvantages of the method of building Lagrange interpolated polynomial and Newton interpolated polynomial.

2. Lagrange interpolated polynomial

We have $n+1$ interpolation nodes$(x_k, y_k), k = 0..n$ on [a; b]. At each $x_k(k = 0..n)$, construct the polynomial

$$L_k(x) = \frac{(x-x_0)(x-x_1)(x-x_2)..(x-x_{k-1})(x-x_{k+1})...(x-x_n)}{(x_k-x_0)(x_k-x_1)(x_k-x_2)..(x_k-x_{k-1})(x_k-x_{k+1})...(x_k-x_n)}$$

$L_k(x), k = 0..n$ are called basic Lagrange polynomials. Then the Lagrange interpolation polynomial is determined by the formula:

$$L(x) = \sum_{k=0}^{n}(y_k.L_k(x))$$

Notice that $L_k(x), k = 0..n$ are polynomials of degree not more than degree *n*, so $L(x)$ should also be polynomials of degree no more than degree *n* and $L(x_k) = y_k, k = 0..n$.

Example 1: Construct a Lagrange interpolated polynomial for the following table of interpolation nodes and use the interpolated polynomial to approximate $y(4)$?

Table 1: Four interpolation nodes (Case 1)

x	1	3	5	7
y	1	17	49	97

(Table 1) Realizing that table 1 has four interpolation nodes, the Lagrange interpolated polynomial has degree no more than 3 degree.

At each node, we define the basic Lagrange polynomials:

$$L_0(x) = \frac{(x-3)(x-5)(x-7)}{(1-3)(1-5)(1-7)}$$

$$L_1(x) = \frac{(x-1)(x-5)(x-7)}{(3-1)(3-5)(3-7)}$$

$$L_2(x) = \frac{(x-1)(x-3)(x-7)}{(5-1)(5-3)(5-7)}$$

$$L_3(x) = \frac{(x-1)(x-3)(x-5)}{(7-1)(7-3)(7-5)}$$

Then the Lagrange interpolated polynomial to be found is:

$$L(x) = 1.L_0(x) + 17.L_1(x) + 49.L_2(x) + 97.L_3(x)$$

$$= 1.\frac{(x-3)(x-5)(x-7)}{(1-3)(1-5)(1-7)} + 17.\frac{(x-1)(x-4)(x-7)}{(3-1)(3-5)(3-7)}$$

$$+49.\frac{(x-1)(x-3)(x-7)}{(5-1)(5-3)(5-7)} + 97.\frac{(x-1)(x-3)(x-5)}{(7-1)(7-3)(7-5)}$$

$$= -\frac{1}{48}(x^3 - 15x^2 + 71x - 105)$$
$$+\frac{17}{16}(x^3 - 13x^2 + 47x - 35)$$
$$-\frac{49}{16}(x^3 - 11x^2 + 31x - 21)$$
$$+\frac{97}{48}(x^3 - 9x^2 + 23x - 15)$$

$$= 2x^2 - 1$$

And then: $y(4) \approx L(4) = 31$

Through example 1 above, it can be seen that the Lagrange interpolated polynomial construction process is very clear, simple, easy to understand, easy to implement with the above data set (the values are all integers). Changing the data slightly from Table 1, suppose we need to construct the Lagrange interpolated polynomial from the following table of interpolation nodes:

Table 2: Five interpolation nodes (Case 2)

x	1.1	3.2	4.05	5.5	7.02
y	1.42	19.48	31.805	59.5	97.5608

With the data in Table 2, we need to construct five basic Lagrange polynomials. These basic polynomials cannot be re-used from Example 1, but must be recalculated from scratch. Each basic Lagrange polynomial has degree no more than 4 degree and basic Lagrange polynomial calculation involves decimal numbers so it is certainly easy to get confused, the calculation operation will be more complicated than with integers. If we do the full process as in example 1, we can also determine the Lagrange interpolation polynomial we are looking for $L(x) = 2x^2 - 1$ but giving this result is no longer as simple as example 1.

Thus, from example 1 combined with the above hypothetical situation, we can see some advantages and disadvantages of the Lagrange interpolation polynomial construction method as follows:

Some advantages

- The polynomial construction method is easy to understand, easy to perform calculations, the process of building polynomials is very clear step by step.
- According to the method content, programming on the computer is simple and easy to perform. Once the method has been programmed on the computer, determining the interpolated polynomial for a table with many nodes, or nodes with real numbers will become easier, faster and more accurate.

Some disadvantages

- If the number of interpolated nodes is large, the calculation operations, although easy, are quite cumbersome and take a lot of time.
- Sometimes to further increase the accuracy of the polynomial, we can define extra interpolation nodes. However, if this method is used, when adding one or more interpolation nodes, the construction basic Lagrange polynomial must be performed from the beginning without reusing the basic Lagrange polynomials of the number table original material.

3. Newton interpolated polynomial

Assume on [a, b], we have $y_k = f(x_k), k = 0..n, x_k < x_{k+1} (k = 0..n-1)$

Put $\Delta x_k = x_{k+1} - x_k (k = 0..n-1)$ and $\Delta y_k = y_{k+1} - y_k (k = 0..n-1)$.

We call the first- ratio of difference of the function $f(x)$ at x_k is:

$$f[x_k, x_{k+1}] = \frac{f(x_{k+1}) - f(x_k)}{x_{k+1} - x_k} = \frac{y_{k+1} - y_k}{x_{k+1} - x_k} = \frac{\Delta y_k}{\Delta x_k} \quad (k = 0..n-1)$$

Similarly, the second ratio of difference of the function $f(x)$ at x_k is:

$$f[x_k, x_{k+1}, x_{k+2}] = \frac{f[x_{k+1}, x_{k+2}] - f[x_k, x_{k+1}]}{x_{k+2} - x_k} \quad (k = 0..n-2)$$

In general, the m-th ratio of difference of the function $f(x)$ at x_k is:

$$f[x_k, x_{k+1}, ..., x_{k+m}] = \frac{f[x_{k+1}, ..., x_{k+m}] - f[x_k, ..., x_{k+m-1}]}{x_{k+m} - x_k} \quad (k = 0..n-m)$$

Then we can determine:

$$P_n(x) = y_0 + (x - x_0).f[x_0, x_1] + (x - x_0)(x - x_1).f[x_0, x_1, x_2]$$
$$+ ... + (x - x_0)(x - x_1)...(x - x_{n-1}) f[x_0, x_1, ..., x_n]$$

$P_n(x)$ is the polynomial to be found, called the forward Newton interpolated polynomial, starting from x_0 . $P_n(x)$ has a degree not exceeding n. Due to the uniqueness of the interpolated polynomial,$P_n(x)$ is also the Lagrange interpolated polynomial, but differing only in construction. In a similar way, we can construct the backward Newton interpolated polynomial, starting at x_n. The backward Newton interpolated polynomial is obtained from the forward Newton interpolated polynomial by interchanging the k and $n - k$ indices. Specifically:

$$P_n(x) = y_n + (x - x_n).f[x_n, x_{n-1}] + (x - x_n)(x - x_{n-1}).f[x_n, x_{n-1}, x_{n-2}]$$
$$+ ... + (x - x_n)(x - x_{n-1})...(x - x_1) f[x_n, x_{n-1}, ..., x_0]$$

Example 2: Construct the Newton interpolated polynomial for the table of interpolated nodes in example 1 (table 1) and use the interpolated polynomial to approximate y(4)?

Notice that table 1 has four interpolation nodes, so the Newton interpolated polynomial has degree no more than cubic.

First, we make a spreadsheet of the ratios from level one to level three. For the convenience of tabulation, we denote TH1, TH2, TH3 as level 1, level 2, and level 3 ratios respectively at $x_k, k = 0,1,2$. Then the table of ratios is determined as follows:

Table 3: The ratio of the finite difference (E.g.,2)

x	y	TH1	TH2	TH3
1	1			
		8		
3	17		2	
		16		0
5	49		2	
		24		
7	97			

From the spreadsheet of ratio of differences, we get the forward Newton interpolated polynomial from $x_0 = 1$ as

P(x) = 1 + (x – 1). 8 + (x – 1)(x – 3). 2 + (x – 1)(x – 3)(x – 4). 0 = $2x^2 - 1$

Similarly: The backward Newton interpolated polynomial from $x_3 = 7$ is

P(x) = 97 + (x – 7). 24 + (x – 7)(x – 5). 2 + (x – 7)(x – 5)(x – 3). 0 = $2x^2 - 1$

It is easy to see that the Newton interpolated polynomial is also the Lagrange interpolated polynomial obtained in Example 1 (due to the unique property of the interpolated polynomial). And then we also get y(4) ≈ $P(4) = 31$

With the data in Table 1, we see that the nodes $x_k, k = 0,1,2,3$ are equally spaced h = 2 (h is called a jump). In this case, we can also construct Newton interpolated polynomials with equidistant nodes. The construction method is as follows: Suppose that on $[a, b]$, we know $y_k = f(x_k)$ with $x_k = x_0 + k.h$ $(k = 0..n)$, h is a constant number and $h > 0$. Then we call:

$\Delta y_k = y_{k+1} - y_k$ $(k = 0...n-1)$ is the first-order progressive finite difference at x_k.

$\Delta^2 y_k = \Delta(\Delta y_k) = \Delta y_{k+1} - \Delta y_k$ $(k = 0...n-2)$ is the second-order progressive finite difference at x_k.

$\Delta^n y_k = \Delta(\Delta^{n-1} y_k) = \Delta^{n-1} y_{k+1} - \Delta^{n-1} y_k$ $(k = 0)$ is the n-order progressive finite difference at x_k.

$\nabla y_k = y_k - y_{k-1}$ $(k = 1...n)$ is the first-order backward finite difference at x_k.

$\nabla^2 y_k = \nabla(\nabla y_k) = \nabla y_k - \nabla y_{k-1}$ $(k = 2...n)$ is the second-order backward finite difference at x_k.

$\nabla^n y_k = \nabla(\nabla^{n-1} y_k) = \nabla^{n-1} y_k - \nabla^{n-1} y_{k-1}$ $(k = n)$ is the n-order backward finite difference at x_k.

Then: set $x = x_0 + t.h$, we have a forward Newton interpolated polynomial starting from x_0 with equidistant nodes is

$$P_n(t) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + ... + \frac{t(t-1)...(t-(n-1))}{n!}\Delta^n y_0.$$

Similarly, set $x = x_n + t.h$, we have a backward Newton interpolation polynomial from x_n with equidistant nodes is

$$P_n(t) = y_n + t\nabla y_n + \frac{t(t+1)}{2!}\nabla^2 y_n + ... + \frac{t(t+1)...(t+(n-1))}{n!}\nabla^n y_n.$$

Example 3: Construct the Newton interpolated polynomial with equidistant nodes for the table of interpolated nodes in example 1 (table 1) and use the interpolated polynomial to approximate $y(4)$?

First, make a spreadsheet of finite differences from the first to the third order. For the convenience of tabulation, we denote HH1, HH2, HH3 as finite differences of first, second, and third order respectively at $x_k, k = 0,1,2$. Then the table of finite differences is defined as follows:

Table 4: The ratio of the finite difference (E.g.,3)

x	y	HH1	HH2	HH3
1	1			
		16		
3	17		16	
		32		0
5	49		16	
		48		
7	97			

From the spreadsheet of finite differences, we have:

The forward Newton interpolated polynomial starting from $x_0 = 1$:

With each $x = x_0 + t.h = 1 + 2t \Rightarrow t = \frac{x-1}{2}$

$$P_n(t) = 1 + t.16 + \frac{t(t-1)}{2!}.16 + \frac{t(t-1)(t-2)}{3!}.0 = 8t^2 + 8t + 1$$

Notice that the Newton interpolated polynomial in this case is a polynomial whose variable is t.

So, with x = 4, we have $t = \frac{4-1}{2} = 1{,}5$. Therefore y(4) $\approx P(1{,}5) = 31$

Similarly: The backward Newton interpolation polynomial starting from $x_3 = 7$:

With each $x = x_3 + t.h = 7 + 2t \Rightarrow t = \frac{x-7}{2}$

$$P_n(t) = 97 + t.48 + \frac{t(t+1)}{2!}.16 + \frac{t(t+1)(t+2)}{3!}.0 = 8t^2 + 56t + 97$$

And with x = 4, we have $t = \frac{4-7}{2} = -1{,}5$. Therefore y(4) $\approx P(-1{,}5) = 31$

Through example 2 and example 3 above, it can be seen that the process of building Newton interpolated polynomials is also very clear, simple, easy to understand, and easy to implement. Assuming we need to build Newton interpolated polynomials from a table of interpolated nodes that are decimal numbers like table 2, the calculation is quite simple with the help of a calculator.

Thus, from example 2 and example 3, we can see some advantages - some disadvantages of construction method of Newton interpolated polynomials as follows:

Some advantages

Newton's interpolation polynomial construction method is easy to understand, easy to perform calculations, the polynomial construction process is very clear step by step

From the original data table, in order to increase the accuracy of the polynomial, if we add interpolation nodes, the tabulation of the ratios of difference or the finite difference does not have to be done from scratch as for the Lagrange interpolated polynomial, we just need to define other finite differences and differences, we can establish a polynomial.

Some disadvantages

If the number of interpolated nodes is large, the calculation operations, although easy, are still quite cumbersome and time consuming.

In the case of Newton interpolated polynomials with equidistant nodes, special attention should be paid to the polynomial when it is a polynomial with variable t. Therefore, if we use interpolation polynomial to approximate the function value at $x = a$ (calculate $f(a)$), there is a need for more operations to convert the value from $x = a$ to $t = t_0$respectively. Then $f(x = a) \approx P(t = t_0)$. This conversion is often forgotten by people, leading to erroneous results.

For Newton interpolated polynomials, people often confuse the calculation of the differences with the table of finite signs. Therefore, teachers need to re-emphasize this when teaching those contents.

4. Conclusions

The construction of each type of interpolation polynomial will have certain advantages and disadvantages as described above. Depending on the requirements of the problem and the purpose of the operator, we can consider which type of polynomial to use. However, because the construction of interpolation polynomials here is using numerical methods, without the support of software, the polynomials all have the common disadvantage that when the number of interpolation nodes is large, the operations Although easy to calculate, it is still quite cumbersome, time consuming and confusing.

Therefore, the method of using polynomial interpolation is only oriented, developing mathematical thinking for workers, but in fact to solve the problem of finding the functional relationship between input - output, people will use the support of software on the computer, specifically using MATLAB to produce results quickly and accurately.

5. Acknowledgement

The authors thank the Thai Nguyen University of Technology for supporting this work.

6. References

1. Dương Thủy Vỹ. Giáo trình phương pháp tính, NXB Khoa học và Kỹ thuật, 2006. [Vietnamese]
2. James G, Dyke PP, Searl J, Craven M, Wei Y. Modern engineering mathematics. Pearson UK, 2020.
3. Ngô Như Khoa, Ôn Ngũ Minh, Phạm Thị Thu Hằng. Giáo trình Toán ứng dụng trong kỹ thuật, NXB Đại học Thái Nguyên, 2019. [Vietnamese]
4. Zill DG. Advanced engineering mathematics. Jones & Bartlett Publishers, 2020.
5. James G. Solutions Manual to Advanced Modern Engineering Mathematics, 2011.
6. Higham NJ. The numerical stability of barycentric Lagrange interpolation. IMA Journal of Numerical Analysis. 2004; 24(4):547-556.
7. Weisstein EW. Lagrange interpolating polynomial, 2004. https://mathworld. wolfram. com/.
8. Mastroianni G, Occorsio D. Optimal systems of nodes for Lagrange interpolation on bounded intervals. A survey. Journal of Computational and Applied Mathematics. 2001; 134(1-2):325-341.
9. Chan WCC, Chyan CJ, Srivastava HM. The Lagrange polynomials in several variables. Integral Transforms and Special Functions. 2001; 12(2):139-148.
10. Chen KY, Liu SJ, Srivastava HM. Some new results for the Lagrange polynomials in several variables. The ANZIAM Journal. 2007; 49(2):243-258.