Developing Two Novel Lagrange-based Algorithms for Direct Calculation of Interpolating Polynomial Coefficients

Seyed Reza Moasheri, Ali Momeni, Seyed Majid Moasheri

***Abstract*— In engineering studies, normally the behavioral function of a system is, normally replaced by a number of measured values per known inputs – in the form of a table function. Sometimes it is necessary to access a system function in order to be able to study behavior of the system. This is where we normally approximate with a polynomial through a table function which is due to the special features of polynomial expressions. Different methods are used to approximate the behavior of a system with a polynomial expression including Lagrange and Newton interpolating polynomial expressions. Although Lagrange and Newton interpolations are meant for calculation per a given point, neither of them provide interpolating polynomials directly, and the coefficients are produced from combination of multiplication of several expressions. Reconsidering the relations in this study, we offer an algorithm for direct calculation of coefficients of different powers of *x* in an interpolating polynomial.**

***Index Terms*— Neutral Points, coefficients of Interpolating, Lagrange Interpolation**

I. Introduction

THE interpolation formula was discovered and introduced by Waring in 1779 and Lagrange interpolation was presented in 1795 [1]. The main Lagrange form has certain shortcomings, e.g., increasing the degree of the polynomial by adding a new interpolation point requires computations from scratch, and also the computation is numerically unstable [2]. Berrut *et al.* have modified Lagrange form and the barycentric Lagrange were introduced as a fast and stable method [3], overcoming the shortcomings of the original form and makes Lagrange interpolation suitable for practical application [2]. Many articles for efficient Lagrange interpolation algorithms are also published. Werner *et al.,* showed that the Lagrangian form of the interpolating polynomial may be calculated with the same number of arithmetic operations as the Newtonian form [4]. Feng *et al.* studied how to obtain exact interpolation polynomial with rational coefficients by approximate interpolating methods [5]. Solares *et al.* have also explored the interpolation mechanism of the separable functional networks, when the neuron functions are approximated by Lagrange polynomials. The coefficients of the Lagrange interpolation formula were estimated during the learning of the functional network by simply solving a linear system of equations [6]. Some articles may compute the coefficients of interpolation, as Gonnet *et al.* considered methods to compute the coefficients of interpolants relative to a basis of polynomials satisfying a three-term recurrence relation [7].

The interpolation polynomial in most of studies, is produced from combination of multiplication of several expressions. It seems that calculation of coefficients is complex. In this paper, by rewriting the base form of Lagrange interpolation, we suggest the method to compute the direct and simple interpolation coefficients.

According to [3], Lagrange Interpolating Polynomial for the table function f on points x_0, … x_n can be defined by the following equation:

$$p_f(x) = \sum_{i=0}^{n} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)} \tag{1}$$

One may also write this polynomial in the form of powers of x as follows:

$$p_f(x) = a_{n_f} x^n + a_{n-1_f} x^{n-1} + \cdots + a_{1_f} x + a_{0_f} \tag{2}$$

thus the coefficients x^n, x^{n-1}, and x^0 are produced by expanding equation (1) as follows:

$$a_{n_f} = \sum_{i=0}^{n} \frac{f_i}{\prod_{\substack{j=0 \\ j\neq i}}^{n} (x_i-x_j)} \tag{3}$$

$$a_{n-1_f} = -\sum_{i=0}^{n} \frac{f_i \sum_{\substack{j=0 \\ j\neq i}}^{n} x_i}{\prod_{\substack{j=0 \\ j\neq i}}^{n} (x_i-x_j)} \tag{4}$$

$$a_{0_f} = (-1)^n \sum_{i=0}^{n} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x_j}{(x_i-x_j)} \tag{5}$$

It can be seen that calculation of coefficients of a_{k_f} gets more complex as k approaches the 0 to n field.

It seems due to the lack of a fixed relationship between the coefficients it is not possible to get iterative-based flowcharts for calculation of the coefficients for different values of n. Yet given equation (3) it is always easy to find the coefficient of the highest power of x. This study uses

Manuscript received March 03,14, 2015; revised April 04,9, 2015.

S. R. Moasheri is with Education Organization of Razavi Khorasan, Education Ministry, Mashhad, Iran (e-mail: R_Moasheri@yahoo.com).

A. Momeni is with the department of electrical and electronics, Shiraz university of technology, Shiraz 13876-71557, Iran (e-mail: a.momeni@sutech.ac.ir).

S. M. Moasheri is with Research Institute of Food Science and Technology (RIFST), Department of Food Biotechnology, Mashhad, Iran (e-mail: m.moasheri@rifst.ac.ir)

equation (3) to calculate coefficient a_{n_f} , and then applies a set of operations on the table function *f*, based on the proposed methods in such a way that a_{n-1_f} can be recalculated from equation (3) (the same goes for other coefficients). As all coefficients are always calculated from the same equations it is possible to make an algorithm for it. Yet before proceeding to such methods we have first to prove a few theorems:

II. New Theorems

A. Theorem 1 (*Retrieval*)

If $g(x)$ is a polynomial function of power *n*, any given $n+1$ point set of this function may be produced through interpolation.

Proof.

We assume a given $n+1$ point set of function $g(x)$ in which x_is are different two by two. We suppose that the function $p_g(x)$ is the result of interpolating per these same points. Since $p_g(x)$ is produced by interpolating points, maximum power of this function is *n*. Assuming that the power of $p_g(x)$ is *l* and less that *n* we have:

$$g(x) = k_{n_g} x^n + k_{n-1_g} x^{n-1} + \cdots + k_{1_g} x + k_{0_g} \tag{6}$$

$$p_g(x) = a_{l_g} x^l + a_{l-1_g} x^{l-1} + \cdots + a_{1_g} x + a_{0_g} \tag{7}$$

Thus the function $T(x)$ is defined as follows:

$$T(x) = g(x) - p_g(x) \tag{8}$$

Since $p_g(x)$ is produced by interpolation of $n+1$ points, it coincides with $g(x)$ at point $n+1$. Therefore, function $T(x)$ must have at least $n+1$ roots. This is while function $T(x)$ has at most *n* roots (because its power is *n*). Hence the assumption that the power of the interpolating polynomial $p_g(x)$ is lower than $g(x)$ is not true.

But is the power of the interpolating polynomial $p_g(x)$ equals that of function $g(x)$ then we have:

$$p_g(x) = a_{n_g} x^n + a_{n-1_g} x^{n-1} + \cdots + a_{1_g} x + a_{0_g} \tag{9}$$

In case since function $p_g(x)$ is produced from interpolating $n+1$ points of function $g(x)$, it coincides with $g(x)$ at $n+1$ points, thus function $T(x)$ must have at least $n+1$ roots. Since function $p_g(x)$ and $g(x)$ are of the same power the function $T(x)$ may rewritten as follows:

$$T(x) = (k_{n_g} - a_{n_g})x^n + (k_{n-1_g} - a_{n-1_g})x^{n-1} + \cdots + (k_{1_g} - a_{1_g})x + (k_{0_g} - a_{0_g}) \tag{10}$$

Thus the only way function $T(x)$ can have at least $n+1$ roots is that $k_{i_g} = a_{i_g}$, in other words the result of interpolation equals the function itself.

B. Theorem 2 (*Mediating Polynomial*)

It is always possible to add the polynomial function $g(x)$ (with maximum power of *n*) per x_is to the table function *f* to calculate the interpolating function $p_f(x)$, and after interpolation of the table function $f+g$ subtract the function $g(x)$ from the interpolating function $p_{f+g}(x)$ to get to the interpolating function $p_f(x)$.

Proof.

Assuming table function *f* as:

x_i	$x_0 \ldots\ldots\ldots\ldots\ldots x_n$
f_i	$f_0 \ldots\ldots\ldots\ldots\ldots f_n$

$P_f(x)$ would be;

$$p_f(x) = \sum_{i=0}^{n} f_i \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)} \tag{11}$$

If a polynomial function of maximum power *n* such as $g(x)$ is added to function *f* per x_is we will have the table function as:

x_i	$x_0 \quad \ldots\ldots\ldots\ldots\ldots \quad x_n$
$f_i + g_i$	$f_0 + g(x_0) \ldots\ldots\ldots\ldots\ldots f_n + g(x_n)$

Now by interpolation we have:

$$p_{f+g}(x) = \sum_{i=0}^{n} (f_i + g(x_i)) \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)} \tag{12}$$

thus,

$$p_{f+g}(x) = \underbrace{\sum_{i=0}^{n} f_i \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}}_{p_f(x)} + \underbrace{\sum_{i=0}^{n} g(x_i) \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}}_{p_g(x)} \tag{13}$$

$$p_{f+g}(x) = p_f(x) + p_g(x) \tag{14}$$

On the other hand we have:

$$p_g(x) = g(x) \tag{15}$$

Because if we show the values of $g(x_i)$ with g_i

x_i	$x_0 \ldots\ldots\ldots\ldots\ldots x_n$
g_i	$g_0 \ldots\ldots\ldots\ldots\ldots g_n$

There is always a unique polynomial function of maximum power *n* that includes all points of the table function *g*. Also since $p_g(x)$ is produced from interpolation of $n+1$ points of n^{th} power function $g(x)$, according to Theorem 1 (*Retrieval*) the $p_g(x)$ will surely be the same polynomial function $g(x)$. Thus we have:

$$p_f(x) = p_{f+g}(x) - g(x) \tag{16}$$

Based on our discussion so far we can offer two methods for calculation of coefficients in interpolating polynomial expression: we know that the Lagrange interpolating polynomial is calculated by the table function *f* based on equation (1). It is obvious that maximum power of $p_f(x)$ is *n*. As mentioned before, a_n may always be calculated from equation (3). Now if we assume that $p_f(x)$ is of n-1 power, then a_{n_f} produced by equation (3) will surely be zero (in other words for $n+1$ points the $p_f(x)$ function is of n-1 power) and to calculate a_{n-1_f} we must use equation (4). However, there may be other solutions as well.

III. Generated Novel Methods

A. The first method

Assume that $p_f(x)$ is of the power of n-1 and is the result interpolation of $n+1$ point.

If we define:

$$g(x) = x.p_f(x) \tag{17}$$

As function $p_f(x)$ is of power n-1 the $p_g(x)$ (is a result of interpolation of $x_i.p_f(x_i)$) is definitely of power *n*. Thus according to Theorem 1 (*Retrieval*) with any given set of n+1 points of function $x.p_f(x)$ we can produce the function $p_g(x)$ from interpolation. Now if this n+1 point set is taken as x_0, … x_n since $x_i.p_f(x_i)=x_i f_i$ we can take the polynomial function $p_g(x)$ as the interpolated form of the table function $g_i=x_i f_i$. Thus $p_g(x)$ becomes:

$$p_g(x) = a_{n_g} x^n + \cdots + a_{1_g} x^1 + a_{0_g} = a_{n-1_f} x^n + a_{1_f} x^2 + a_{0_f} x \quad (18)$$

When assuming that function $p_f(x)$ is of power n-1 then a_{n_g} will never be zero and equation (3) may be used for calculations.

Assume that the table function f is as follows:

x_i	x_0 …………….x_n
f_i	f_0 …………….f_n

Hence $g_i=x_i f_i$ becomes

x_i	x_0 …………… x_n
$g_i = x_i f_i$	$x_0 f_0$ ……………. $x_n f_n$

Now, from equation (3) we have

$$a_{n_g} = \sum_{i=0}^{n} f_i x_i \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{1}{x_i - x_j} \quad (19)$$

On the other hand we know that $p_f(x)$ is of power n-1 so according to equation (4):

$$a_{n-1_f} = -\sum_{i=0}^{n} \frac{f_i \sum_{\substack{j=0\\ j\neq i}}^{n} x_j}{\prod_{\substack{j=0\\ j\neq i}}^{n} x_i - x_j} \quad (20)$$

We can rewrite the equation (20) as;

$$a_{n-1_f} = -\sum_{i=0}^{n} \frac{\left(f_i \sum_{\substack{j=0\\ j\neq i}}^{n} x_j + f_i x_i \right) - f_i x_i}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)}$$

$$= -\sum_{i=0}^{n} \frac{f_i \sum_{j=0}^{n} x_j - f_i x_i}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)}$$

$$a_{n-1_f} = \sum_{i=0}^{n} \frac{-f_i \sum_{j=0}^{n} x_j}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)} + \sum_{i=0}^{n} \frac{f_i x_i}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)} \quad (21)$$

As $\sum_{j=0}^{n} x_j$ is independent of i it may be taken out of the $\sum_i$.

$$a_{n-1_f} = -(\sum_{j=0}^{n} x_j) \sum_{i=0}^{n} \frac{f_i}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)} + \sum_{i=0}^{n} \frac{f_i x_i}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)} \quad (22)$$

so,

$$a_{n-1_f} = -(\sum_{j=0}^{n} x_j) a_{n_f} + \sum_{i=0}^{n} \frac{f_i x_i}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)} \quad (23)$$

But we know that a_{n_f} is zero because $p_f(x)$ is of power n-1, thus

$$a_{n-1_f} = \sum_{i=0}^{n} \frac{f_i x_i}{\prod_{\substack{j=0\\ j\neq i}}^{n} (x_i - x_j)} = \sum_{i=0}^{n} f_i x_i \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{1}{x_i - x_j} \quad (24)$$

Therefore given equation (19) we have

$$a_{n-1_f} = a_{n_g} \quad (25)$$

Now if function $p_f(x)$ is of power α and α<n-1 then a_{n_g} will definitely be zero. Thus to find a_{n-2_f} we can multiply the g_is in x_i to redefine the table function and find the coefficient of the n[th] power of its interpolating function.

$$k_i = x_i g_i \quad (26)$$

According to what we have mentioned so far

$$p_k(x) = a_{n_k} x^n + a_{n-1_k} x^{n-1} + \ldots + a_{2_k} x^2 + a_{1_k} x^1 + a_{0_k} = a_{n-2_f} x^n + a_{n-3_f} x^{n-1} + \ldots + a_{0_f} x^2 \quad (27)$$

$$\Rightarrow a_{n-2_f} = a_{n_k} \quad (28)$$

This way the power of the function may be found without calculating all coefficients. If after T iterations the a_{n_T} is not zero for the first time, the power of the interpolating function will be n-T. But calculation of the first coefficient of the highest power of x from the interpolating expression suffices to enables us to calculate the other coefficients according to the above theorems. Given Theorem 2 (*Mediating Polynomial*) we can subtract any given polynomial function of maximum power n from the table function and then add it to the table function after interpolating the resulting function table. Thus assuming that after T iterations the a_{n_T} coefficient of the interpolating polynomial $p_f(x)$ function for the following table

x_i	x_0 …………….x_n
T_i	T_0 …………….T_n

… has resulted from equation (3), the interpolating function $p_T(x)$ becomes:

$$p_T(x) = a_{n_T} x^n + q(x) \quad (29)$$

… the power of $q(x)$ will at most be one less than n.

$$\Rightarrow q(x) = p_T(x) - a_{n_T} x^n \quad (30)$$

Thus the table function q for x_0, … x_n becomes:

$$\Rightarrow q_i = p_T(x_i) - a_{n_T} x_i^{\,n} = T_i - a_{n_T} x_i^{\,n} \quad (31)$$

So we can say that $q(x)$ is the interpolating function of the table function q.

We know that coefficient of x_n at $q(x)$ is definitely zero. Thus, to find the coefficient of x^{n-1} in $q(x)$ (or x^{n-T-1} coefficient from the main interpolating polynomial) it suffices to multiply the q_is in x_is and then calculate the x_n coefficient in the new table function through equation (3). Since in this method always $\prod_{\substack{j=0\\ j\neq i}}^{n} \left(x_i - x_j \right)$ remains unchanged for different values of i, so they only have to be

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

calculated once. Now given the above discussions we can offer the following simple algorithm for calculation of the coefficients of interpolating polynomial expressions:

$k = n$

$For\ i = 0 \ldots n\ do\{$

$$L_i = \prod_{\substack{j=0 \\ j\neq i}}^{n} (x_i - x_j)$$

$\}end\ for$

$While(k\rangle 0\ \ and\ \ f_i \neq 0)\,\mathrm{do}\{$

$$a_n = \sum_{i=0}^{n} \frac{f_i}{L_i}$$

Print a_n

$For\ i = 0 \ldots n\ do\{$

$$f_i = x_i . f_i - a_n x_i^{n+1}$$

$\}end\ for$

$k = k - 1$

$\}end\ while$

B. The second method

Since the function $p_f(x)$ is of power n-1 and also given Theorem 1 (*Retrieval*) we can find the $p_f(x)$ function from re-interpolation for n given points. Also as the table function f coincides with $p_f(x)$ at n-1 point we can rule out any given point and through interpolating of the remaining n points the same $p_f(x)$ will be produced again (This is in fact another form of the *Retrieval Theorem*).

Proof.

We assume that point x_n is ruled out, now we want to prove that a_{n-1_f} is produced from interpolation of the n remaining points. According to equation (24) we have:

$$a_{n-1_f} = \sum_{i=0}^{n} f_i x_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{1}{x_i - x_j}$$

$$= f_n x_n \prod_{j=0}^{n-1} \frac{1}{x_n - x_j} + \sum_{i=0}^{n-1} \frac{f_i x_i}{(x_i - x_n)} \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j}$$

$$= f_n x_n \prod_{j=0}^{n-1} \frac{1}{x_n - x_j} + \sum_{i=0}^{n-1} \frac{f_i\left(x_i - x_n + x_n\right)}{(x_i - x_n)} \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j}$$

with more expiation and rewriting;

$$= f_n x_n \prod_{j=0}^{n-1} \frac{1}{x_n - x_j} + \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j} + x_n \sum_{i=0}^{n-1} \frac{f_i}{(x_i - x_n)} \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j}$$

$$= \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j} + f_n x_n \prod_{j=0}^{n-1} \frac{1}{x_n - x_j} + x_n \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{1}{x_i - x_j}$$

$$= \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j} + x_n \left(f_n \prod_{\substack{j=0 \\ j\neq n}}^{n} \frac{1}{x_n - x_j} + \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{1}{x_i - x_j} \right)$$

$$= \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j} + x_n \sum_{i=0}^{n} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{1}{x_i - x_j}$$

therefore,

$$a_{n-1_f} = \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j} + x_n a_{n_f} \tag{32}$$

On the other hand we know that a_{n_f} is zero, so we have:

$$a_{n-1_f} = \sum_{i=0}^{n-1} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n-1} \frac{1}{x_i - x_j} \tag{33}$$

As there is no order among the points, whenever a_{n_f} is produced from equation (3) for n+1 points from the table function and is zero, we will be able to rule out any given point (e.g. x_n) from the table function f and use the equation (3) to calculate a_{n-1_f}, again. Now, fallowing T iterations the a_{n-T_f} coefficient is not zero for the first time, the power of the interpolating function would be $n-T$. Now we may subtract $a_{n-T_f} x_i^{n-T}$ from the table function f for all points of $x_0, \ldots x_{n-T-1}$ then use equation (3) to produce the coefficient of x^{n-T-1} for all points of $x_0, \ldots x_{n-T-1}$. In this way, we may offer the following algorithm for calculation of coefficients in interpolating polynomial expressions:

$For\ i = 0 \ldots n\ do\{$

$$L_i = \prod_{\substack{j=0 \\ j\neq i}}^{n} (x_i - x_j)$$

$\}end\ do$

$While(n \geq 0\ \ and\ \ f_i \neq 0)\,\mathrm{do}\{$

$$a_n = \sum_{i=0}^{n} \frac{f_i}{L_i}$$

Print a_n

$For\ i = 0 \ldots (n-1)\ \ do\{$

$$L_i = \frac{L_i}{(x_i - x_n)}$$

$$f_i = f_i - a_n x_i^n$$

$\}end\ for$

$n = n - 1$

$\}end\ while$

IV. THE OTHER APPLICATION OF THEOREM 2

We may always add any given function of a polynomial $g(x)$ with maximum power of n to f_i expressions in such a way that a number of f_i+g_is are would be zero. Then after calculating the $p_{f+g}(x)$, the $p_f(x)$ will be produced through subtracting $g(x)$ from $p_{f+g}(x)$. On the other hand, when some of f_is are a fixed k number, if we choose g(x)=k (fixed number) and subtract the table function f_i we can reduce the number of calculations to some extent. This type of calculation is suitable for the updatable systems. If the all points are interpolated and new data is received, updating interpolation polynomial, needed calculating for just a new data, and adds the coefficients of new interpolation polynomial to the main polynomial.

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

V. Some examples

(Example 1 for *Mediating Polynomial theorem*) We want to interpolate the following table function based on the Mediating Polynomial Theorem.

x_i	1	2	3
f_i	2	5	10

Assume that point (x_i=1,f_i=2) is a new added point and polynomial for other two points (points 2 and 3) is driven out as $g(x)$=5(x-1). By subtracting $g(x)$ from the new table function f we will have:

x_i	1	2	3
$f_i - g_i$	2	0	0

Now we calculate the Lagrange Polynomial only for *one point*:

$$p_{f-g}(x) = 2\frac{(x-2)(x-3)}{(1-2)(1-3)} = x^2 - 5x + 6$$

$$p_f(x) = p_{f-g}(x) + g(x) = x^2 - 5x + 6 + 5x - 5 = x^2 + 1$$

(Example 2) We need to calculate the interpolating function of the following table function based on the proposed algorithms.

x_i	x_4	x_3	x_2	x_1	x_0
	-2	-1	2	1	0
f_i	-13	0	15	2	1

Solution based on the first method:

$n = 4$

(Phase 1)

x_i	-2	-1	2	1	0
f_i	-13	0	15	2	1
L_i	24	-6	24	-6	4
$\frac{f_i}{L_i}$	$\frac{-13}{24}$	$\frac{0}{-6}$	$\frac{15}{24}$	$\frac{2}{-6}$	$\frac{1}{4}$

$$a_{4_f} = \sum_{i=0}^{4} \frac{f_i}{L_i} = 0$$

(Phase 2)

$f_i = x_i . f_i - (0)x^5$

x_i	-2	-1	2	1	0
f_i	26	0	30	2	0
L_i	24	-6	24	-6	4
$\frac{f_i}{L_i}$	$\frac{26}{24}$	$\frac{0}{-6}$	$\frac{30}{24}$	$\frac{2}{-6}$	$\frac{0}{4}$

$$a_{3_f} = \sum_{i=0}^{4} \frac{f_i}{L_i} = 2$$

(Phase 3)

$f_i = x_i . f_i - (2)x^5$

x_i	-2	-1	2	1	0
f_i	12	2	-4	0	0
L_i	24	-6	24	-6	4
$\frac{f_i}{L_i}$	$\frac{12}{24}$	$\frac{2}{-6}$	$\frac{-4}{24}$	$\frac{0}{-6}$	$\frac{0}{4}$

$$a_{2_f} = \sum_{i=0}^{4} \frac{f_i}{L_i} = 0$$

(Phase 4)

$f_i = x_i . f_i - (0)x^5$

x_i	-2	-1	2	1	0
f_i	-24	-2	-8	0	0
L_i	24	-6	24	-6	4
$\frac{f_i}{L_i}$	$\frac{-24}{24}$	$\frac{-2}{-6}$	$\frac{-8}{24}$	$\frac{0}{-6}$	$\frac{0}{4}$

$$a_{1_f} = \sum_{i=0}^{4} \frac{f_i}{L_i} = -1$$

(Phase 5)

$f_i = x_i . f_i - (-1)x^5$

x_i	-2	-1	2	1	0
f_i	16	1	16	1	0
L_i	24	-6	24	-6	4
$\frac{f_i}{L_i}$	$\frac{16}{24}$	$\frac{1}{-6}$	$\frac{16}{24}$	$\frac{1}{-6}$	$\frac{0}{4}$

$$a_{0_f} = \sum_{i=0}^{4} \frac{f_i}{L_i} = 1$$

Thus the interpolating function becomes:

$$p_f(x) = a_{4_f}x^4 + a_{3_f}x^3 + a_{2_f}x^2 + a_{1_f}x + a_{0_f} = 2x^3 - x + 1$$

Solution based on the second method:

$n = 4$

(Phase 1)

x_i	-2	-1	2	1	0
f_i	-13	0	15	2	1
L_i	24	-6	24	-6	4
$\frac{f_i}{L_i}$	$\frac{-13}{24}$	$\frac{0}{-6}$	$\frac{15}{24}$	$\frac{2}{-6}$	$\frac{1}{4}$

$$a_4 = \sum_{i=0}^{4} \frac{f_i}{L_i} = 0$$

$$L_i = \frac{L_i}{(x_i - x_4)}$$

$$f_i = f_i - (0)x_i^4$$

Delete $x_4 \rightarrow n=3$

(Phase 2)

x_i	-2	-1	2	1	0
f_i		0	15	2	1
L_i		-6	6	-2	2
$\frac{f_i}{L_i}$		$\frac{0}{-6}$	$\frac{15}{6}$	$\frac{2}{-2}$	$\frac{1}{2}$

$$a_3 = \sum_{i=0}^{3} \frac{f_i}{L_i} = 2$$

$$L_i = \frac{L_i}{(x_i - x_3)}$$

$$f_i = f_i - (2)x_i^3$$

Delete $x_3 \rightarrow n=2$

(Phase 3)

x_i	-2	-1	2	1	0
f_i			-1	0	1
L_i			2	-1	2
$\frac{f_i}{L_i}$			$\frac{-1}{2}$	$\frac{0}{1}$	$\frac{1}{2}$

$$a_2 = \sum_{i=0}^{2} \frac{f_i}{L_i} = 0$$

$$L_i = \frac{L_i}{(x_i - x_2)}$$

$$f_i = f_i - (0)x_i^2$$

Delete $x_2 \rightarrow n=1$

(Phase 4)

x_i	-2	-1	2	1	0
f_i				0	1
L_i				1	-1
$\frac{f_i}{L_i}$				$\frac{0}{1}$	$\frac{1}{-1}$

$$a_1 = \sum_{i=0}^{1} \frac{f_i}{L_i} = -1$$

$$L_i = \frac{L_i}{(x_i - x_1)}$$

$$f_i = f_i - (-1)x_i^1$$

Delete $x_1 \rightarrow n=0$

(Phase 5)

x_i	-2	-1	2	1	0
f_i					1
L_i					1
$\frac{f_i}{L_i}$					$\frac{1}{1}$

$$a_1 = \sum_{i=0}^{0} \frac{f_i}{L_i} = 1$$

Thus the interpolating function becomes:

$$p_f(x) = a_{4_f} x^4 + a_{3_f} x^3 + a_{2_f} x^2 + a_{1_f} x + a_{0_f} = 2x^3 - x + 1$$

VI. Conclusion

We've presented here two new algorithms based on Lagrange interpolation formula to calculate coefficients of polynomial. Two theorems are presented here. The main objective of this paper is, however polynomial coefficients calculation, the second theorem (*Mediating Polynomial*) is applicable for updating systems. When a new point is added to the table, it is just needed to calculate the polynomial for the new point, by using the Mediating Polynomial theorem and updating the coefficients.

References

[1] Trefethen, Lloyd N. *Approximation theory and approximation practice* (Book style). Siam, 2013, pp. 3.

[2] Van Beeumen, Roel, Wim Michiels, and Karl Meerbergen, “Linearization of Lagrange and Hermite interpolating matrix polynomials.” *IMA Journal of Numerical Analysis*, 2014, dru019.

[3] Berrut, Jean-Paul, and Lloyd N. Trefethen, “Barycentric Lagrange interpolation,” *Siam Review.* Vol. 46, no. 3, 2004, pp. 501-517.

[4] Werner, Wilhelm, “Polynomial interpolation: Lagrange versus Newton,” *Mathematics of computation,* vol. 43, no. 167, 1984, pp. 205-217.

[5] Feng, Yong, Xiaolin Qin, Jingzhong Zhang, and Xun Yuan, “Obtaining exact interpolation multivariate polynomial by approximation.” *Journal of Systems Science and Complexity.* Vol. 24, no. 4, 2011, pp. 803-815.

[6] Solares, Cristina, Eduardo W. Vieira, and Roberto Mínguez, “Functional networks and the lagrange polynomial interpolation.” In *Intelligent Data Engineering and Automated Learning–IDEAL,* Springer Berlin Heidelberg, *2006*, pp. 394-401.

[7] Gonnet, Pedro. "Efficient construction, update and downdate of the coefficients of interpolants based on polynomials satisfying a three-term recurrence relation." *arXiv preprint arXiv:1003.4628,* 2010.