Journal of
Computer Science and Software Development

Research Article Open Access

A Recursive Relationship in Lagrange Interpolating

Edmund A. Chadwick[1*] **and Ali Hatam**[2*]

[1]School of Computing, Science and Engineering, University of Salford, Salford, M5 4WT, United Kingdom

[2]Department of Applied Mathematics, Faculty of Mathematics and Computer Science, Amirkabir University of Technology, No. 350, Hafez Ave, Valiasr Square, Tehran, Iran

[*]**Corresponding Author:** Edmund A. Chadwick, School of Computing, Science and Engineering, University of Salford, Salford, M5 4WT, United Kingdom, Tel: +44 (0)161 295 3259, E-mail: e.a.chadwick@salford.ac.uk

Received Date: May 04, 2024 **Accepted Date:** June 04, 2024 **Published Date:** June 07, 2024

Citation: Edmund A. Chadwick, Ali Hatam (2024) A Recursive Relationship in Lagrange Interpolating. J Comput Sci Software Dev 3: 1-4

Abstract

Lagrange interpolation, as described in many numerical analysis books, has a deficiency that by adding a new data, all computations must be recalculated from the beginning. A very simple method that comes here shows that there is no needed to do the calculations from the beginning, but as well as the interpolating by Newton's divided differences, all former computations without any changes, can be extended by adding new data.

Keywords: Numerical Analysis; Polynomial Approximation; Lagrange Interpolation; Newton's Divided Differences

Introduction

Theorem 1.1 (Theorem 3.2 in [2]) If $x_0, x_1, \ldots, x_n$ are $n+1$ distinct numbers and f is a function whose values are given at these numbers, then a unique polynomial $Pn(x)$ of degree at most n exists with $f(x_k) = P_n(x_k)$, for each $k = 0, 1, \ldots, n$. This polynomial is given by

$$P_n(x) = \sum_{k=0}^{n} f(x_k) L_{n,k}(x), \quad (1)$$

where for each k = 0, 1, ..., n,

$$L_{n,k}(x) = \frac{(x - x_0)(x - x_1)\ldots(x - x_{k-1})(x - x_{k+1})\ldots(x - x_n)}{(x_k - x_0)(x_k - x_1)\ldots(x_k - x_{k-1})(x_k - x_{k+1})\ldots(x_k - x_n)}. \quad (2)$$

As it's seen in both formulas (1) and (2), all Lagrange polynomials $L_{n,k}$ are polynomials of degree n and if a point $(x_{n+1}, f(x_{n+1}))$ (provided x_{n+1} differs from x_i , $i = 0, 1, \ldots, n$) to be added to the previous data, for producing $L_{n+1,k}$ and therefore $P_{n+1}(x)$, previous $L_{n,k}$ are useless and all former computations must be recalculated. This is because the numerator and denominator of all $L_{n,k}$'s are involving with all fixed data and if any new dada to be added, for producing new $L_{n+1,k}$, in all factors in the denominator of $L_{n,k}$, the number x_k must be replaced with the new data x_{k+1} and this means previous results in denominator can not be extended and all must be completely recalculated, and this is the lack of traditional Lagrange interpolating. Fortunately, interpolating by Newton's method has no this deficiency because of Newton's interpolating is based on the following recursive formula (3),

$$f[x_j, x_{j+1}, \ldots, x_{k-1}, x_k] = \frac{f[x_{j+1}, \ldots, x_k] - f[x_j, \ldots, x_{k-1}]}{x_k - x_j}, \quad (3)$$

for more analysis see [1,3].

We now introduce the following formula (4),

$$P_{n+1}(x) = \frac{(x - x_0)(x - x_1)\ldots(x - x_n)}{(x_{n+1} - x_0)(x_{n+1} - x_1)\ldots(x_{n+1} - x_n)} (f(x_{n+1}) - P_n(x_{n+1})) + P_n(x), \quad (4)$$

to modify traditional Lagrange's interpolating and equipped this method with a recursive formula, such that by adding new data, for extending to higher degree interpolation polynomial, all old computations can be used.

New Lagrange Interpolating (Chadwick's method)

Again we suppose $x_0, x_1, \ldots, x_n$ *are* $n+1$ distinct numbers, we present the following repeated method to find the polynomial $P_n(x)$ of degree n passing the points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$.

$$P_1(x) = \frac{x - x_1}{x_0 - x_1} y_0 + \frac{x - x_0}{x_1 - x_0} y_1, \quad \text{defining } P_1,$$

$$P_{k+1}(x) = \frac{(x - x_0)(x - x_1)\ldots(x - x_k)}{(x_{k+1} - x_0)(x_{k+1} - x_1)\ldots(x_{k+1} - x_k)} (y_{k+1} - P_k(x_{k+1})) + P_k(x), \; k = 1, 2, \ldots, n-1. \quad (5)$$

Equ. (5) shows that P_k has been used to produce P_{k+1}.

Computer Algorithm

```
Read (x0,y0), (x1,y1);  % E1
     |   If x0=x1 output "Error in Data" ;  % E2
     |      stop.
     |      otherwise;
     |      P(1)(x)=((x-x1)/(x0-x1))*y0 + ((x-x0)/(x1-x0))*y1;  % E3
     |   end If;
n=1;
x(0)=x0;
x(1)=x1;
L(0)(x)=1;
     |   1 Read (x,y);
     |     n=n+1;
     |     x(n)=x;
     |     y(n)=y;
     |          | Do I=0 to n-1;
     |          |    If x(n)=x(I) output "Error in Data";  % E4
     |          |       stop.
     |          |       otherwise;
     |          |       L(I+1)(x)=((x-x(I))/(x(n)-x(I)))*L(I)(x);  % E5
     |          | end Do;
     |     P(n)(x)=L(n)(x) * (y(n)-P(n-1)(x(n))) + P(n-1)(x);  % E6
     |   Continue 1;
output P(n)(x);  % E7
End Program.
Explanations:
E1: at least two points are necessary for interpolating
E2: checking distinctness of data
E3: making P_1(x)
E4: checking distinctness of data
E5: coefficient of interpolating polynomial
E6: making desirable interpolating polynomial (applying recursive formula)
E7: desired interpolating polynomial of all data
```

Conclusion

The heart of this article is defining a recurrence formula for Lagrange interpolating polynomials rather than calculating individual formula for each polynomial. This method can also be extended for interpolating polynomials of two variable or multivariate as well.

Conflict of Interest

The authors declare that we have no conflict of interest. Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

This is a joint-authored original work that has not been supported financially by any organizations.

References

1. JP Berrut, LN Trefethen (2004) Barycentric lagrange interpolation, SIAM Review, 46: 501-17.

2. RL Burden, JD Faires, AM Burden (2016) Numerical Analysis, Cengage Learning, Boston, MA 02210 USA.

3. MS Floater, K Hormann (2007) Barycentric rational interpolation with no poles and high rates of approximation, Numerische Mathematik, 107: 315-31.