IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ⁙ Impact Factor 7.12 ⁙ Vol. 10, Issue 11, November 2022

DOI: 10.17148/IJIREEICE.2022.101101

Review Paper on Newton's forward, backward interpolation formulae

Pranav Suryawanshi[1], Shrajal Dwivedi[1], Vishal V. Mehtre[2]

Student, Department of Electrical Engineering, Bharati Vidyapeeth (Deemed to be University) College of Engineering Pune, Maharashtra[1]

Assistant Professor, Department of Electrical Engineering, Bharati Vidyapeeth (Deemed to be University) College of Engineering Pune, Maharashtra[2]

Abstract: In this paper, we have given provided all the information "Newton's forward and backward interpolation method", as this method is used for equal intervals, and both forward and backward is applied. Which includes derivation, numerical solutions for both methods, differentiation between forward and backward methods, applications used in real life, and advantages and disadvantages, given with its result that which one is more efficient. We have also given the references we have used for this review paper.

Keywords: Newton's Forward Method, Newton's Backward Method, Mathematical Solution, Derivation, MATLAB Program, C++ Program, Differentiate between Forward and Backward.

INTRODUCTION

Interpolation is a method that determines a function's value for any interim value of an independent variable, whereas extrapolation is the method used to calculate the value of the function outside of the specified range.[5]

Forward Method - Newton's forward interpolation is a polynomial interpolation that is based on the initial value and degrees of Newton's forward operator. The polynomial degree has one less data point than there are. This approach can only interpolate data points that are uniformly spaced apart. [1]

Backward Method - Newton's backward interpolation is a polynomial interpolation that is based on the final value and degrees of Newton's backward operator. This formula is used when the value of f(x) is required at the end of the table. h is known as the common difference and u = (x – an) / h, Here an is the last term in a table.[2]

DERIVATION

Forward Method –

Let y = f(x). Let the values of x be x0, x1, x2, …
Let the values of x be equally spaced. So all the values of x are spaced away from each other by 'h'.
Since they are all equally spaced, we have the values x0, x0+h, x0+2h, …
So now the y values are y(x1), y(x0+h), y(x0 + 2h), …Let us represent the values of y as y0, y1, y2, …
Let us consider the first two values of y (namely y0 and y1)
△y0 = y1 - y0 (eq 1)
△y1 = y2 - y1 (eq 2)
In general,
△Yn = Yn+1 - Yn.
These are the first divided differences.
The difference between the first divided differences is the second divided difference.
△^2Yo = △(△Yo) = △(Y1 - Yo) = △Y1 - △Yo
From eq (1) and eq (2)
△^2Yo = △(△Yo) = △(Y1 - Yo) = △Y1 - △Yo = y2 - y1 -y1 + y0 = y2 - 2y1 + y0
△y0 = y1 - y0. Thus y1 = y0 + △y0 = (1 + △) y0
△y1 = y2 - y1. Thus y2 = y1 + △y1 = (1 + △) y1. But see y1 from the previous line
Thus △y1 = (1 + △)^2 y0

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
ISO 3297:2007 Certified ⋇ Impact Factor 7.12 ⋇ Vol. 10, Issue 11, November 2022
DOI: 10.17148/IJIREEICE.2022.101101

Similarly
△y2 = (1 + △)^3 y0
Thus any difference between the divided difference can be represented in terms of the first divided difference.
Thus, yn = (1 + △)^n yo
Since the above expression is of the form a + b and there are two terms in the equation and since it is raised to the power 'n', we can use the binomial theorem to evaluate the expression.
By applying the binomial theorem to yn = (1 + △)^n yo, we have-

$$y(x) = y_0 + py_0 + \frac{p(p-1)}{2!}.\Delta^2 y_0 + \frac{p(p-1)(p-2)}{3!}.\Delta^3 y_0 + \frac{p(p-1)(p-2)\,(p-3)}{4!}\,.\Delta^4 y_0$$

where p can also be written as 'n' (the difference factor)[8]

Backward Method –

Let the function y f(x) take the values y0, y1, y2, corresponding to the values x0, x0 h, x0 2h, of x. Suppose it is required to evaluate f(x) for x in ph, where p is any real number. Then we have

$$y_{p=} f(x_n + ph) = Epf(x_n) = (1-\nabla)^{-p} y_n$$

$$= [+p\nabla + \frac{p(p+1)\nabla^2}{2!} + p(p+1)(p+2)y^3 y_0 + \cdots]y_n$$

$$y_{p=} y_n + p\nabla y_n + \frac{p(p+1)\nabla^2_{yn}}{2!} + \frac{p(p+1)(p+2)\nabla^2_{yn}}{3!} + \cdots \ldots (1)$$

It is called Newton's backward interpolation formula as (1) contains yn and backward differences of yn[8].

MATLAB Program –

Forward Method -

```
%Newton's Forward Difference Formula MATLAB Program
x=[0 2 4 7 10 12]; % inputting values of x
fx=[20 20 12 7 6 6]; % inputting values of y
dt=zeros(6,10); % function
for i=1:6 dt(i,1)=x(i);% for loop
dt(i,2)=fx(i); % calling function
end
n=5; % number of iterations
for j=3:10
for i=1:n
dt(i,j)=dt(i+1,j-1)-dt(i,j-1)
end
n=n-1;
end
h=x(2)-x(1) % finding the value of h
xp=1.5; % defining the value of xp
for i=1:5
q=(xp-x(i))/h; % calculating number of intervals
if (q>0&&q<1)
p=q;
 end
 end
 p
l=xp-(p*h)
 for i=1:5
 if(l==x(i))
```

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
ISO 3297:2007 Certified ⋇ Impact Factor 7.12 ⋇ Vol. 10, Issue 11, November 2022
DOI: 10.17148/IJIREEICE.2022.101101

```
 r=i;
 end
 end % calculating the different values of y
 f0=fx(r);
 f01=dt(r,3);
 f02=dt(r,(3+1));
 f03=dt((r),(3+2));
 f04=dt((r),(3+3));
% using the forward interpolation formula
 fp=(f0)+((p*f01)+(p*(p-1)*f02)/(2)) + ((p*(p-1)*(p-2)*f03)/(6))+((p*(p-1)*(p-2)*(p-3)*f04)/(24)). [9]
```

Output –

dt=

0	20	0	-8	11	-10	6	0	0	0
2	20	-8	3	1	-4	0	0	0	0
4	12	-5	4	-3	0	0	0	0	0
7	7	-1	1	0	0	0	0	0	0
10	6	6	0	0	0	0	0	0	0
12	6	6	0	0	0	0	0	0	0

h= 2

p= 0.7500

l= 0

fp = 21.3996

Backward Method –

```
%Newton's Backward Difference Formula MATLAB Program
x=[0 8 16 24 32 40]; % inputting the values of x
fx=[14.621 11.843 9.870 8.418 7.305 6.413]; % inputting the value of y
dt=zeros(6,7); % declaring function
for i=1:6 % stating loop
dt(i,1)=x(i);
dt(i,2)=fx(i);
end
n=5;
for j=3:7
 for i=1:n % using for loop
 dt(i,j)=dt(i+1,j-1)-dt(i,j-1) % defining dt
 end n=n-1;
 end
h=x(2)-x(1) % finding the value of h
xp=27; % defining xp
for i=1:6
q=(xp-x(i))/h;
if (q>0&&q<1)
 p=q;
 end
 end
 p
 l=xp-(p*h)
 for i=1:6
```

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
ISO 3297:2007 Certified ⁑ Impact Factor 7.12 ⁑ Vol. 10, Issue 11, November 2022
DOI: 10.17148/IJIREEICE.2022.101101

```
 if(l==x(i))
 r=i;
 end
 end
% finding different values of y
 f0=fx(r);
 f01=dt((r-1),3);
 f02=dt((r-2),(3+1));
f03=dt((r-3),(3+2));
 f04=dt((r-4),(3+3));
%using backward difference formula
 fp=(f0)+((p*f01)+(p*(p+1)*f02)/(2)) + ((p*(p+1)*(p+2)*f03)/(6))[9]
```

Output-

dt=

0	14.6210	-2.54	0.8050	-0.284	0.1020	-0.0308
8	11.845	-1.974	0.521	-0.182	0.0640	0
16	9.8547	-1.468	0.339	-0.1182	0	0
24	8.4180	-1.100	0.220	0	0	0
32	7.3045	-0.854	0	0	0	0
40	6.4123	0	0	0	0	0

h= 8

p= 0.3720

l= 24

C++ Program –

Forward Method –

```
// CPP Program to interpolate using
// newton forward interpolation
#include <bits/stdc++.h>
using namespace std;
// calculating u mentioned in the formula
float u_cal(float u, int n)
{
	float temp = u;
	for (int i = 1; i < n; i++)
		temp = temp * (u - i);
	return temp;
}
// calculating factorial of given number n
int fact(int n)
{
	int f = 1;
	for (int i = 2; i <= n; i++)
		f *= i;
	return f;
}

int main()
```

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
ISO 3297:2007 Certified ⁜ Impact Factor 7.12 ⁜ Vol. 10, Issue 11, November 2022
DOI: 10.17148/IJIREEICE.2022.101101

```
{
        // Number of values given
        int n = 4;
        float x[] = { 45, 50, 55, 60 };

        // y[][] is used for difference table
        // with y[][0] used for input
        float y[n][n];
        y[0][0] = 0.7071;
        y[1][0] = 0.7660;
        y[2][0] = 0.8192;
        y[3][0] = 0.8660;

        // Calculating the forward difference
        // table
        for (int i = 1; i < n; i++) {
                for (int j = 0; j < n - i; j++)
                        y[j][i] = y[j + 1][i - 1] - y[j][i - 1];
        }
        // Displaying the forward difference table
        for (int i = 0; i < n; i++) {
                cout << set(4) << x[i]
                        << "\t";
                for (int j = 0; j < n - i; j++)
                        cout << set(4) << y[i][j]
                                << "\t";
                cout << endl;
        }
        // Value to interpolate at
        float value = 52;
        // initializing u and sum
        float sum = y[0][0];
        float u = (value - x[0]) / (x[1] - x[0]);
        for (int i = 1; i < n; i++) {
                sum = sum + (u_cal(u, i) * y[0][i]) /
        fact(i);
        }
        cout << "\n Value at " << value << " is "
                << sum << endl;
        return 0;
}
```

Output –

```
 45    0.7071    0.0589    -0.00569999    -0.000699997
 50    0.766    0.0532    -0.00639999
 55    0.8192    0.0468
 60    0.866

Value at 52 is 0.788003
```

Backward Method –

```
// CPP Program to interpolate using
// newton backward interpolation
#include <bits/stdc++.h>
using namespace std;
// Calculation of u mentioned in the formula
float u_cal(float u, int n)
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ⋇ Impact Factor 7.12 ⋇ Vol. 10, Issue 11, November 2022

DOI: 10.17148/IJIREEICE.2022.101101

```
{
        float temp = u;
        for (int i = 1; i < n; i++)
                temp = temp * (u + i);
        return temp;
}
// Calculating factorial of given n
int fact(int n)
{
        int f = 1;
        for (int i = 2; i <= n; i++)
                f *= i;
        return f;
}

int main()
{
        // number of values given
        int n = 5;
        float x[] = { 1891, 1901, 1911,
                                        1921, 1931 };
        // y[][] is used for difference
        // table and y[][0] used for input
        float y[n][n];
        y[0][0] = 46;
        y[1][0] = 66;
        y[2][0] = 81;
        y[3][0] = 93;
        y[4][0] = 101;
        // Calculating the backward difference table
        for (int i = 1; i < n; i++) {
                for (int j = n - 1; j >= i; j--)
                        y[j][i] = y[j][i - 1] - y[j - 1][i - 1];
        }
        // Displaying the backward difference table
        for (int i = 0; i < n; i++) {
                for (int j = 0; j <= i; j++)
                        cout << set(4) << y[i][j]
                                << "\t";
                cout << endl;
        }
        // Value to interpolate at
        float value = 1925;
        // Initializing u and sum
        float sum = y[n - 1][0];
        float u = (value - x[n - 1]) / (x[1] - x[0]);
        for (int i = 1; i < n; i++) {
                sum = sum + (u_cal(u, i) * y[n - 1][i]) /
fact(i);
        }

        cout << "\n Value at " << value << " is "
                << sum << endl;
        return 0;
}
```

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
ISO 3297:2007 Certified ⁜ Impact Factor 7.12 ⁜ Vol. 10, Issue 11, November 2022
DOI: 10.17148/IJIREEICE.2022.101101

Output –

46				
66	20			
81	15	-5		
93	12	-3	2	
101	8	-4	-1	-3

Value at 1925 is 96.8368

Mathematical solution of Newton's forward, backward

interpolation formulae –

Example:

Forward Method –

Q. Find Solution using Newton's Forward Difference Formula:

x	f(x)
1891	46
1901	66
1911	81
1921	93
1931	101

x = 1895

Finding option 1. Value f(2)

Solution:

The value of the table for x and y

x	f(x)
1891	46
1901	66
1911	81
1921	93
1931	101

Newton's forward difference interpolation method to find the solution

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ※ Impact Factor 7.12 ※ Vol. 10, Issue 11, November 2022

DOI: 10.17148/IJIREEICE.2022.101101

Newton's forward difference table is

x	y	Δy	Δ2y	Δ3y	Δ4y
1891	46				
66-46=20					
1901	66		15-20=-5		
81-66=15		-3--5=2			
1911	81		12-15=-3		-1-2=-3
93-81=12		-4--3=-1			
1921	93		8-12=-4		
101-93=8					
1931	101				

The value of x at you want to find the f(x):x=1895

$$\mathbf{h = x_1 - x_0 = 1901 - 1811 = 10}$$

$$\boldsymbol{p = x - x_0/h = \ 1895 - 1891/10 = \ 0.4}$$

Newton's forward difference interpolation formula is:

$$\boldsymbol{y(x) = y_0 + py_0 + \frac{p(p-1)}{2!}.\Delta^2 y_0 + \frac{p(p-1)(p-2)}{3!}.\Delta^3 y_0 + \frac{p(p-1)(p-2)\,(p-3)}{4!}\,.\Delta^4 y_0}$$

$$\boldsymbol{parensparensy(1895) = 46 + 0.4 \times 20 + \frac{0.4(.4-1)}{2!} \times 5 + \frac{0.4(0.4-1)(0.4-2)}{3!} \times 2 + \frac{0.4(0.4-1)(0.4-2)(0.4-3)}{4!} \times (-3)\ y(1895) = 46 + 8 + 0.6 + 0.128 + +0.1248}$$

$$\boldsymbol{y(1895) = 54.8528}$$

Solution of newton's forward interpolation method y(1895)=54.8528

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ⋇ Impact Factor 7.12 ⋇ Vol. 10, Issue 11, November 2022

DOI: 10.17148/IJIREEICE.2022.101101

Backward Method –

Q. Find a Solution using Newton's Backward Difference formula.

x	f(x)
1891	46
1901	66
1911	81
1921	93
1931	101

x = 192

Solution:

The value of the table for x and y

x	f(x)
1891	46
1901	66
1911	81
1921	93
1931	101

Newton's backward difference interpolation method to find the solution

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ⁘ Impact Factor 7.12 ⁘ Vol. 10, Issue 11, November 2022

DOI: 10.17148/IJIREEICE.2022.101101

Newton's backward difference table is

X	Y	∇y	∇2y	∇3y	∇4y
1891	46				
		20			
1901	66		-5		
		15		2	
1911	81		-3		-3
		-12		-1	
1921	93		-4		
		8			
1931	101				

The value of x at you want to find the f(x):x=1925

h=x1-x0=1901-1891=10

$$p = x - x_0/h = \ 1925 - 1931/10 = \ 0.6$$

$$y(x) = y_0 + p\nabla y_0 + \frac{p(p-1)}{2!}.\nabla^2 y_0 + \frac{p(p-1)(p-2)}{3!}.\nabla^3 y_0 + \frac{p(p-1)(p-2)\,(p-3)}{4!}\,.\nabla^4 y_0$$

$$y(x) = 101 + (-0.6)8 + \frac{(-0.6)(-0.6+1)}{2} \times -4 + \frac{(-0.6)(0.6+1)(0.6+2)(0.6+2)}{6} \times (-1) + \frac{(-0.6)(-0.6+1)(0.6+2)(-0.6+3)}{24} \times (-3)$$

y(1925)=101-4.8+0.48+0.056+0.1008

y(1925)=96.8368

Solution of newton's backward interpolation method y(1925)=96.8368.[10]

Differentiate between Newton’s forward and backward

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified ⋇ Impact Factor 7.12 ⋇ Vol. 10, Issue 11, November 2022

DOI: 10.17148/IJIREEICE.2022.101101

Interpolation –

Newton's Forward Interpolation	Newton's Backward Interpolation
For locations that are nearer to x0, using Newton's forward difference approach is preferable.	For locations that are nearer to xn, the Newton backward difference method works better.
Newton's forward interpolation technique is to be used when the x - data point is near the beginning.	Newton's backward difference technique is to be used when the x - data points is near the finish.
The differences y1 – y0, y2 – y1, y3 – y2, ……, yn – yn–1 when denoted by dy0, dy1, dy2, ……, dyn–1 are respectively, called the first forward differences. Thus, the first forward differences are: $\Delta \mathbf{Y_r} = \mathbf{Y_{r+1}} + \mathbf{Y_r}$[6]	The differences y1 – y0, y2 – y1, ……, yn – yn–1 when denoted by dy1, dy2, ……, dyn, respectively, are called first backward difference. Thus, the first backward differences are: $\nabla \mathbf{Y_r} = \mathbf{Y_{r-1}} + \mathbf{Y_r}$.[6]

RESULT AND DISCUSSION –

The polynomial has a degree that is one lower than the total number of observational pairs. You can utilise the polynomial that symbolises the provided collection of numerical data.
for interpolation at any point when the independent variable's two extreme values fall inside the range. The method of interpolation described here may also be successfully used for inverse interpolation.
Under the following two circumstances [2], Newton's forward interpolation method is appropriate for calculating the value of the dependant variable: 1. The independent variable's values are supplied within an equal range. 2. The independent variable's value that corresponds to which the first half of the series of the dependent variable, whose value needs to be estimated, based on the independent variable's values. Moreover, there is also necessity of searching for some formula for representing a set of numerical data on a pair ofvariables by a polynomial if the given values of the independent variable are not at equal interval.[2]
This formula can only be used in these two circumstances to describe a collection of numerical data on a pair of variables by a polynomial. Therefore, if the value of the independent variable corresponding to which the value of the dependent variable is to be estimated lies in the last half of the series of the given values, which are at equal intervals, of the independent variable, then some formula for representing a set of numerical data on a pair of variables by a polynomial must be found.

APPLICATION

Interpolation is primarily used to assist users, whether they are scientists, photographers, engineers, or mathematicians, in evaluating possible data sources outside of their acquired data. Interpolation is commonly used beyond mathematics to scale pictures and change the sample rate of digital signals.[8]

One of the most fundamental and practical numerical techniques is an interpolation. When working with tabular or graphical functions, it is an essential tool.[5]

Another of the most significant numerical methods with broad applications in mathematics, computer science, and technical research is Newton's backward interpolation.[4]

Instead, new pixels need to be made. The programme that enlarges the image use interpolation to "guess" what these pixels should appear like.[3]

Advantages – The use of nested multiplication and the relatively simple addition of new data points for higher-order interpolating polynomials are the benefits of Newton interpolation. This technique is relatively easy to use and has excellent local convergence.[2]

DOI: 10.17148/IJIREEICE.2022.101101

They always over the data because of their stiffness (caused by smoothness).[3]

Disadvantages – The computational cost of using this approach is high. Numerous indicators must be taken into account when solving, and if even one sign is overlooked, the proper solution cannot be obtained.[10]

Another drawback is that when working just with provided data, it is not always feasible to have a functional representation of our function's derivative.[4]

CONCLUSION

The examination of Newton's Forward and Backward interpolation equations was done to show how effective the method had been. It was also stated how the step affected the approaches' accuracy.[5]

Additionally, we have included the programs for Newton's Forward and Backward interpolation in MATLAB and C++.[4]

In this work, the results of various comparison questions are examined in light of the theory.[8]

REFERENCES

[1] Biswajit Das and Dhritikesh Chakrabarty, "A Study on Newton Forward and Backward Interpolation", Published By: - International Journal of Statistics and Applied Mathematics 2016; 1(2): 36-41, issued on 12, June, 2016.
[2] Prof. Danny Newton Forward and Backward Interpolation, sixth edition.
[3] "Polymorphism Measures for early risk prediction", "Saida Benlarbi"2016.
[4] Study of Measurements using Newton Forward and Backward Interpolation Method "Shivam" issue 2nd July 2013.
[5] S.P. Venkatesh "Computational Methods in Engineering" Prof. Emertius, Mechanical Engineering, Indian Institute of Technology Madras 2015.
[6] Prasanna Swaminath "Computational Methods in Engineering" Published By- British Library Cataloguing, Postdoctoral researcher, EM2C Laboratory, Ecole Centrale Paris, France 2012.
[7] "Ehiwario J.C &Aghamie S.O", Comparative Study of Bisection, Newton-Raphson and Secant Methods of Root. Finding Problems, Volume no 04, Issue no 04, April 2014.
[8] "Tan Tingting", "Li Ying" & " Jiang Tong", The Analysis of the Convergence of Newton-RaphsonMethod Based on the Current Injection in Distribution Network Case",Volume no 5, Issue no 03, June 2013.
[9] "Saba Akram" International Journal of Scientific & Engineering Research, Volume 6, Issue 7, July- 2015 1748 ISSN2229-5518
[10] Robert J Schilling, Sandra L Harries. Applied Numerical Methods for Engineers, Brooks /Cole, Pacific Grove, CA, 2010.