C Program for Newton Forward Interpolation

September 29, 2014

Interpolation is the process of finding the values of y corresponding to the any value of x between x0 and xn for the given values of y=f(x) for a set of values of x. Out of the many techniques of interpolation, Newton's Forward and Backward Interpolation are two very widely used formulas. In this tutorial, we're going to discuss a C program for Newton Forward Interpolation along with its sample output.

Both of Newton's formulas are based on finite difference calculus. These formulas are very often used in engineering and related science fields. Before going through the source code for Newton Forward Interpolation, let's go through the forward interpolation formula and the variables used in the C program.

Newton's forward interpolation formula contains y0 and the forward differences of y0. This formula is used for interpolating the values of y near the beginning of a set of tabulated values and extrapolation the values of y a little backward (i.e. to the left) of y0. The formula is given below:

$$P(x) = y_0 + q\,\Delta y_0 + \frac{q(q-1)}{2!}\Delta^2 y_{-1} + \frac{(q+1)q(q-1)}{3!}\Delta^3 y_{-1} +$$
$$+ \frac{(q+1)q(q-1)(q-2)}{4!}\Delta^4 y_{-2} + \frac{(q+2)(q+1)q(q-1)(q-2)}{5!}\Delta^5 y_{-2} + \dots$$
$$\dots + \frac{(q+n-1)\dots(q-n+1)}{(2n-1)!}\Delta^{2n-1} y_{-(n-1)} + \frac{(q+n-1)\dots(q-n)}{(2n)!}\Delta^{2n} y_{-n},$$

Compared to forward interpolation, the backward interpolation formula contains yn and the backward differences of yn. This formula is used for interpolating the values of y near the end of a set of tabulated values and also for extrapolating the values of y a little ahead (i.e. to the right) of yn.

Variables Used:

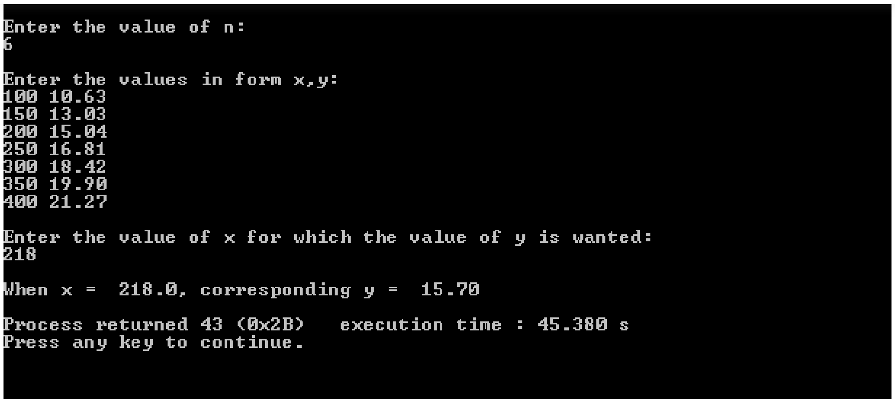
- MAXN – the minimum value of N
- ORDER – the maximum order in the difference table
- ax – an array containing values of x
- ay – an array containing values of y
- diff – a 2D array containing the difference table
- h – spacing between values of x
- x – value of x at which the value of y is wanted
- yp – calculated value of y
- nr – numerator of the terms in expansion of y.p
- dr – denominator of the terms in expansion of y.p

Newton Forward Interpolation in C:

```
#include<stdio.h>
```

```
    int n,i,j,k;
    printf("\nEnter the value of n:\n");
    scanf("%d",&n);

    printf("\nEnter the values in form x,y:\n");
    for (i=0;i<=n;i++)
        scanf("%f %f",&ax[i],&ay[i]);
    printf("\nEnter the value of x for which the value of y is wanted: \n");
    scanf("%f",&x);
    h=ax[1]-ax[0];

    //now making the difference table
    //calculating the 1st order of differences
    for (i=0;i<=n-1;i++)
        diff[i][1] = ay[i+1]-ay[i];

    //now calculating the second and higher order differences
    for (j=2;j<=ORDER;j++)
        for(i=0;i<=n-j;i++)
        diff[i][j] = diff[i+1][j-1] - diff[i][j-1];

    //now finding x0
    i=0;
    while (!(ax[i]>x))
        i++;

    //now ax[i] is x0 and ay[i] is y0
    i--;
    p = (x-ax[i])/h;
    yp = ay[i];

    //now carrying out interpolation
    for (k=1;k<=ORDER;k++)
    {
        nr *=p-k+1;
        dr *=k;
        yp +=(nr/dr)*diff[i][k];
    }
    printf("\nWhen x = %6.1f, corresponding y = %6.2f\n",x,yp);
}
```

Input/Output:



```
Enter the value of n:
6

Enter the values in form x,y:
100 10.63
150 13.03
200 15.04
250 16.81
300 18.42
350 19.90
400 21.27

Enter the value of x for which the value of y is wanted:
218

When x =  218.0, corresponding y =  15.70

Process returned 43 (0x2B)   execution time : 45.380 s
Press any key to continue.
```

Also see,

Lagrange Interpolation in C

Newton Divided Difference in C

Numerical Methods Tutorial Compilation

You are requested to write a program of your own for backward interpolation based on the source code above. If you have any queries regarding Newton's forward interpolation, or its C source code, bring them up to me from the comments section.

You can find more Numerical methods tutorial using C language here.