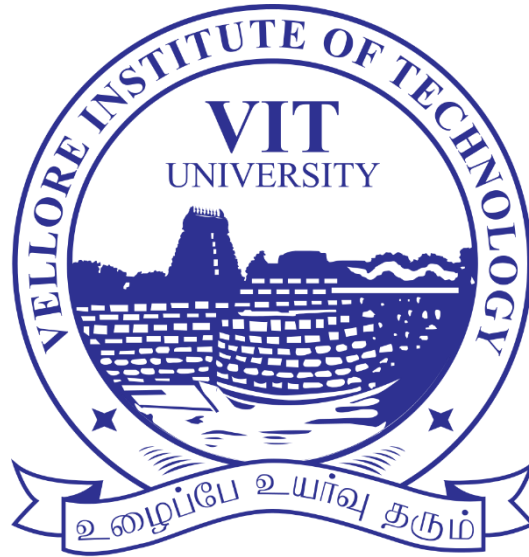


MAT 203: Numerical Analysis Project Report



Topic: Analysis of Interpolation Formulae

Team members:

C.Geetha Charan Reddy -14BCE1064

Siddanth Varyambat- 14BCE1028

Shubham Majmudar -14BCE1069

Under the guidance of:

Prof.David Maxim Gururaj

School of Computer Science and Engineering

Vellore Institute of Technology, Chennai
Campus, Chennai

Analysis of Interpolation formulae

1 ABSTRACT

Interpolation formulas have been used for long time to approximate values using computers. These were designed to be implemented in computers.

Reason being that the number of operations which can be done were limited and iterations could be done more easily. So the best way to understand these interpolation methods is to analyze and implement them on a computer.

A Computer offers many tools to analyze the power of an interpolation formula like

- Compilation time
- Accuracy
- Memory

Thereby providing with information to judge which Method is appropriate for a given situation to evaluate the best possible solution.

In the project we would like to implement two types of interpolation Methods.

- Newton's Interpolation Method.
- Bessel's interpolation Method.

Our work plan involves preparing codes for These Methods and concluding which Method is useful for getting the most accurate and quick results. Go ahead and get started.

2 UTILITIES USED

- DEV C++
- Core temp

3 BRIEFING

Newton forward interpolation formula:

- An interpolation formula to determine the value of x chosen at the edge of the table.
- Follows a diagonal addition of the values in a difference table.
- Can result in loss of accuracy of the result when computed at center of the table.

$$f(x) \cong P_n(x) = f_0 + r\Delta f_0 + \frac{r(r-1)}{2!} \Delta^2 f_0 + \dots + \frac{r(r-1) \dots (r-n+1)}{n!} \Delta^n f_0$$

Bessel's Interpolation formula:

- Similar to newton but is computed using central difference in a difference table.
- Follows a specific pattern for addition of values in the difference table.
- Can give very accurate result when the value has to be found at center of the difference table.

$$y = y(t_n) + u \cdot [y(t_{n+1}) - y(t_n)]$$

4 CODE FOR NEWTON FORWARD INTERPOLATION FORMULA

```
#include<stdio.h>

#include<time.h>

#define MAXN 100

#define ORDER 4

main()

{clock_t begin, end;

double time_spent;

begin = clock();

/* here, do your time-consuming job */

end = clock();

time_spent = (double)(end - begin) / CLOCKS_PER_SEC;

float ax[MAXN+1], ay [MAXN+1], diff[MAXN+1][ORDER+1], nr=1.0, dr=1.0,x,p,h,yp;

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);
}

```

5 CODE FOR BESSEL'S INTERPOLATION FORMULA

```

#include<stdio.h>

#include<time.h>

#define MAXN 100

#define ORDER 4

main()

{clock_t begin, end;

double time_spent;

begin = clock();

/* here, do your time-consuming job */

end = clock();

time_spent = (double)(end - begin) / CLOCKS_PER_SEC;

float ax[MAXN+1], ay [MAXN+1], diff[MAXN+1][ORDER+1], nr=1.0, dr=1.0,x,p,h,yp;

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
```

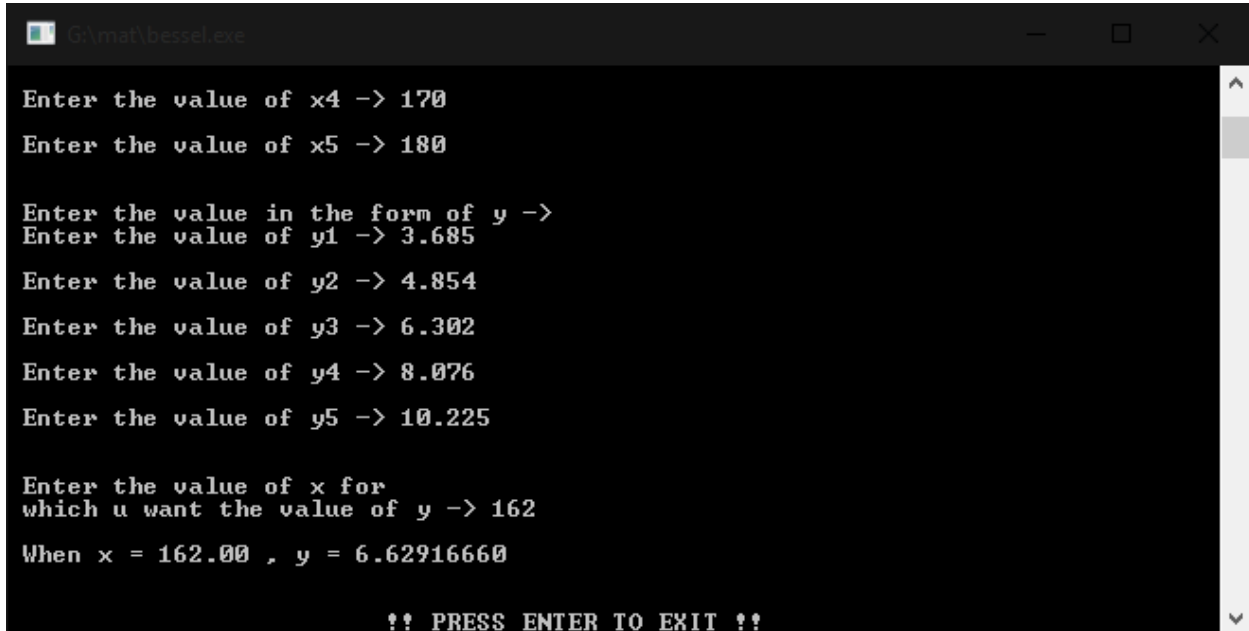
```
i--;  
p = (x-ax[i])/h;  
yp = ay[i];
```

```
//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); }
```

6 OUTPUT FOR BESSEL'S INTERPOLATION PROGRAM



```
G:\mat\bessel.exe

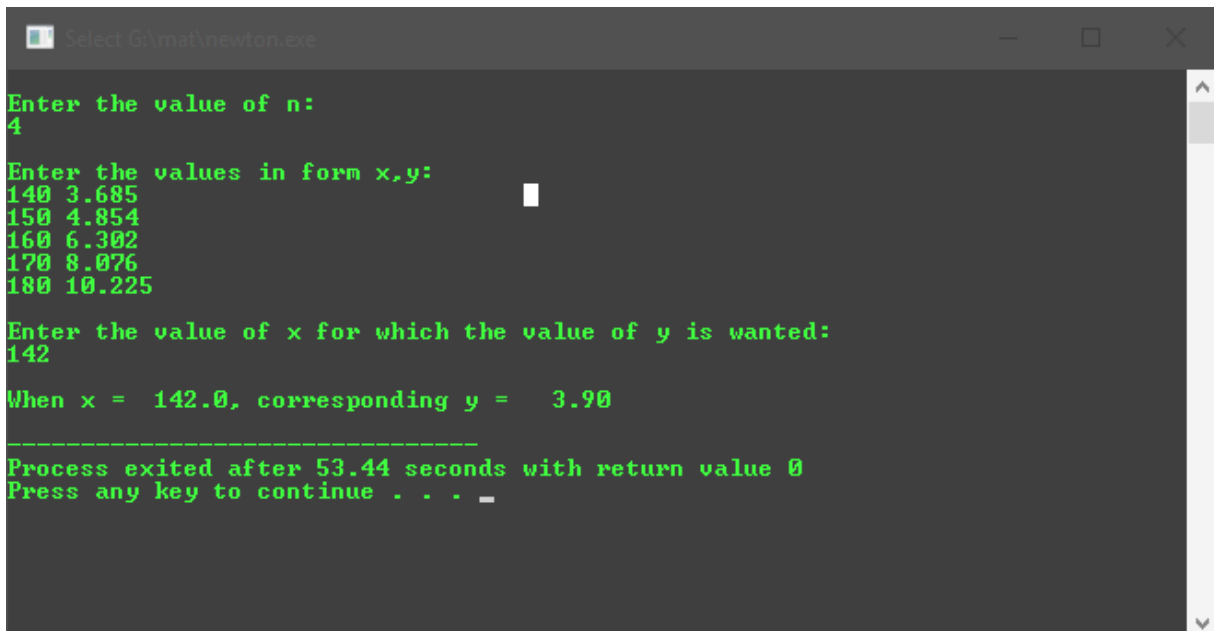
Enter the value of x4 -> 170
Enter the value of x5 -> 180

Enter the value in the form of y ->
Enter the value of y1 -> 3.685
Enter the value of y2 -> 4.854
Enter the value of y3 -> 6.302
Enter the value of y4 -> 8.076
Enter the value of y5 -> 10.225

Enter the value of x for
which u want the value of y -> 162
When x = 162.00 , y = 6.62916660

!! PRESS ENTER TO EXIT !!
```

7 OUTPUT FOR NEWTON FORWARD INTERPOLATION PROGRAM



```
Select G:\mat\newton.exe

Enter the value of n:
4

Enter the values in form x,y:
140 3.685
150 4.854
160 6.302
170 8.076
180 10.225

Enter the value of x for which the value of y is wanted:
142

When x = 142.0, corresponding y = 3.90

-----
Process exited after 53.44 seconds with return value 0
Press any key to continue . . . _
```

8 ANALYSIS

The Bessel formula gives a better approximation than newton interpolation formula

Bessel formula causes temperature rises of 3 degrees when compared to newton forward interpolation formula.

Both have the same worst case time complexity $O(n)$.

Both have same space complexity.

Bessel is more accurate by 8 – order decimals than newton forward interpolation formula.

9 CONCLUSION

Both interpolation formulas are good for the purpose

Bessel's Interpolation formula is suitable for finding central values of the table.

Newton forward difference interpolation formula is suitable for edge values of the table.

*****THE END*****