1 Samrat Mondal Roll : 20101105088

Numerical Method Lab

# LAB - 1

**NAME OF EXP.** : Newton's Forward and Backward Interpolation.

**THEORY** : The interpolation is the technique of estimating the value of a function for any intermediate value of the independent value.

- Newton's Forward Interpolation

If $y = f(x)$ takes the value of $y_0, y_1 \dots y_n$ corresponding to $x = x_0, x_1 \dots x_n$, then

$$f(x) = y_0 + u \Delta y_0 + \frac{u(u-1)\Delta^2 y_0}{2!} + \dots + \frac{u(u-1)\dots(u-n+1)}{n!} \times \Delta^n y_0$$

where, $u = \dfrac{x - x_0}{h}$

- Newton's Backward Interpolation

$$f(x) = y_n + u \Delta y_n + \frac{u(u+1)\Delta^2 y_n}{2!} + \dots \dots + \frac{u(u+1)\dots(u+(n-1))}{n!} \Delta^n y_n$$

We use these two methods only if the values under $x$ have equal intervals.

# CODING :

Question -

Find the number of men getting wages between Rs 10 and Rs 15 from following data.

| Wages | Freq. |
| --- | --- |
| 0 - 10 | 9 |
| 10 - 20 | 30 |
| 20 - 30 | 35 |
| 30 - 40 | 42 |

- By Newton's Forward interpolation
- Python

```
# calculating u mentioned in the formula
 def u_cal (u, n) ;
   temp = u
   for i in range (1, n) :
        temp = temp * (u - i) ;
   return temp ;
# calculating factorial given number n
 def fact (n) :
   f = 1 ;
   for  i in range (2, n+1) :
      f * = i ;
   return f ;
```

```
n = 4
x = [ 10, 20, 30, 40 ];

# y[ ][ ] is used for difference table
y = [[0 for i in range(n)] for j in range(n)];

y [0] [0] = 9 ;
y [1] [0] = 39;
y [2] [0] = 74;
y [3] [0] = 116;

# Calculating the forward difference table
for i in range (1, n) :
    for j in range (n-i) :
        y [j] [i] = y[j+1] [i-1] - y[j] [i-1];

print ("______________________________________");
print (" x(i)  y(i)  y1 (i) y2(i) y3(i)");
print ("______________________________________");

# Displaying forward difference table
for i in range (n) :
    print (x [i], end = "\t\t\t");
    for j in range (n-i) :
        print (y [i] [j], end = "\t\t\t");
    print (" ");
```

```
value = 15;
# initialising u and sum
sum = y[0][0];
u = (value - x[0]) / (x[1] - x[0]);
print("u = ", u);
for i in range(1, n):
    sum = sum + (u_cal(u, i) * y[0][i]) / fact(i);

print("\n Interpolated Value at ", value, " is ", round(sum))
result = round(sum) - (y[0][0]);
print("Number of men getting wages b/w $10 and $15 :",
                                        result);
```

**RESULT :** The screenshot of the output is attached

File Edit Selection View Go Run Terminal Help    forward interpolation.py - Visual Studio Code

PROBLEMS    OUTPUT    TERMINAL    DEBUG CONSOLE    Code

```
[Running] python -u "c:\Python\forward interpolation.py"

x(i)        y(i)        y1(i)       y2(i)       y3(i)

10          9           30          5           2
20          39          35          7
30          74          42
40          116
u = 0.5

Interpolated Value at 15 is 24
Number of men getting wages b/w $10 and $15: 15

[Done] exited with code=0 in 0.163 seconds
```

Select Python Interpreter ⊗ 0 ⚠ 0    Ln 62, Col 67  Spaces: 4  UTF-8  CRLF  Python

Type here to search    27°C  ENG  21:35  23-07-2021

**Question :** Find the lowest degree polynomial of the data and then calculate $y(5)$

| $x$ | $y$ |
|---|---|
| 0 | 5 |
| 2 | 9 |
| 4 | 61 |
| 6 | 209 |
| 8 | 501 |

$$\therefore \quad u = \frac{x-8}{2}$$

$$f(x) = 501 + \left(\frac{x-8}{2}\right) \times 292 + \left(\frac{x-8}{2}\right)\left(\frac{x-6}{2}\right) \times \frac{144}{2} + \frac{(x-8)}{2}\frac{(x-6)}{2}\frac{(x-4)}{2} \times \frac{48}{6}$$

$$\Rightarrow x^3 - 2x + 5$$

- By Newton's Backward Interpolation
- Python

```
# calculating u mentioned in the formula
def u_cal (u, n):
    temp = u ;
    for i in range ( 1, n) :
        temp = temp * (u + i) ;
    return temp ;

# calculating factorial of given number n
def fact (n) :
    f = 1 ;
    for i in range (2, n+ 1) :
        f *= i ;
    return f ;
```

⑥

```
n = 5;
x = [0, 2, 4, 6, 8];
# y[][] is used for difference table
y = [[0 for i in range(n)] for j in range(n)];

y[0][0] = 5;
y[1][0] = 9;
y[2][0] = 61;
y[3][0] = 209;
y[4][0] = 501;
# Calculating the backward difference table
for i in range(1, n):
    for j in range(n-1, i-1, -1):
        y[j][i] = y[j][i-1] - y[j-1][i-1];
print("Newton's Backward Interpolation");
print("________________________________________");
print("x(i)  y(i)  y1(i)  y2(i)  y3(i)  y4(i)");
print("________________________________________");
# Displaying the table
for i in range(0, n):
    print(x[i], end="\t\t\t");
    for j in range(0, i+1):
```

```
        print (y[i][j], end = "\t\t\t");
        print (" ");

Value = 5;
sum = y[4][0];
u = (Value - x[4])/(x[1] - x[0]);
print ("u =", u);
for i in range (1, n):
        sum = sum + (u_cal (u, i) * y[4][i])/fact(i);

print ("\n Interpolated Value at", value, "is", sum);
```

**RESULT :** The screenshot of the output is attached below.

**CONCLUSION :** In this lab we learnt how to estimate the value of a function for any intermediate value of the independent variable using Newton's Forward and Backward Interpolation.

File Edit Selection View Go Run Terminal Help backward interpolation.py - Visual Studio Code

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE Code

```
[Running] python -u "c:\Python\backward interpolation.py"
Newton's Backward Interpolation

______________________________________________________________________
x(i)        y(i)        y1(i)        y2(i)        y3(i)        y4(i)
______________________________________________________________________
0           5
2           9           4
4           61          52           48
6           209         148          96           48
8           501         292          144          48           0
u = -1.5

Interpolated Value at 5 is 120.0

[Done] exited with code=0 in 0.151 seconds
```

Select Python Interpreter ⊗ 0 ⚠ 0 Ln 61, Col 18 Spaces: 4 UTF-8 CRLF Python

Type here to search 99+ 27°C ENG 23:17 23-07-2021