# Numerical Computation - Assignment 7

**薛劭杰　　1930026143**

**Q1.**

There are 4 points (-1, 6), (0, 3), (1, 2), (2, 3), by Newton divided difference:

$f(x_0) = 6, f(x_1) = 3, f(x_2) = 2, f(x_3) = 2$

$f(x_0, x_1) = \frac{f(x_1)-f(x_0)}{x_1-x_0} = \frac{3-6}{0-(-1)} =- 3$

$f(x_1, x_2) = \frac{f(x_2)-f(x_1)}{x_2-x_1} = \frac{2-3}{1-0} =- 1$

$f(x_2, x_3) = \frac{f(x_3)-f(x_2)}{x_3-x_2} = \frac{3-2}{2-1} = 1$

$\left.\begin{matrix} f(x_0, x_1) \\ f(x_1, x_2) \end{matrix}\right\} \Rightarrow f(x_0, x_1, x_2) = \frac{f(x_1,x_2)-f(x_0,x_1)}{x_2-x_0} = 1$

$\left.\begin{matrix} f(x_1, x_2) \\ f(x_2, x_3) \end{matrix}\right\} \Rightarrow f(x_1, x_2, x_3) = \frac{f(x_2,x_3)-f(x_1,x_2)}{x_3-x_1} = 1$

$\left.\begin{matrix} f(x_0, x_1, x_2) \\ f(x_1, x_2, x_3) \end{matrix}\right\} \Rightarrow f(x_1, x_2, x_3, x_4) = \frac{f(x_1, x_2,x_3)-f(x_0,x_1,x_2)}{x_3-x_0} = 0$

Divide difference table.

| | $f[]$ | $f[,]$ | $f[,,]$ | $f[,,,]$ |
|---|---|---|---|---|
| $x_0 =- 1$ | 6 | -3 | 1 | 0 |
| $x_1 = 0$ | 3 | -1 | 1 | |
| $x_2 = 1$ | 2 | 1 | | |
| $x_3 = 2$ | 3 | | | |

$P_1(x) = f[x_0] = 6$

$P_2(x) = f[x_0] + f[x_0, x_1](x - x_0) = 6 - 3(x + 1) =- 3x + 3$

$P_3(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) = 6 - 3(x + 1) + 1(x + 1)(x - 0) = x^2 + x - 3x + 3 = x^2 - 2x + 3$

$P_4(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + f[x_0, x_1, x_2, x_3](x - x_0)(x - x_1)(x - 1) = 6 - 3(x + 1) + 1(x + 1)(x - 0) + 0 = x^2 - 2x + 3$ .

So, the degree of the polynomial function is 2.

2.

(a).

There are 4 points (0, 0), (1, 1), (2, 2), (3, 7), by Lagrange Polynomials:

$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x-x_j}{x_i-x_j}, \ P_n(x) = \sum_{i=0}^{n} [y_i * L_i(x)].$

$$P(x) = y_0 * \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + y_1 * \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + y_2 * \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)}$$
$$+ y_3 * \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} = 0 + 1 * \frac{(x-0)(x-2)(x-3)}{1*(-1)*(-2)} + 2 * \frac{(x-0)(x-1)(x-3)}{2*1*(-1)} + 7 *$$
$$\frac{(x-0)(x-1)(x-2)}{3*2*1} = x + \frac{2}{3}x(x - 1)(x - 2) = \frac{2}{3}x^3 - 2x^2 + \frac{7}{3}x$$

(b).

We can just add the degree of the polynomial function but not add some point by degree elevation:

$$P(x)_1 = y_0 * \frac{(x-x_1)^2(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + y_1 * \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + y_2 * \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)}$$

$$+ y_3 * \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)}$$

Or

$$P(x) = y_0 * \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + y_1 * \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + y_2 * \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)}$$

$$+ y_3 * \frac{(x-x_0)(x-x_1)(x-x_2)^2}{(x_3-x_0)(x_3-x_1)(x_3-x_2)}$$

(c).

In the question (a), we get a Lagrange Polynomials to fit the four points. We can assume that the degree of polynomial function is 3, which means the point (4, 2) is in the polynomial function. Put the point into the function:

$$P(4) = 0 + 1 * \frac{(4-0)(4-2)(4-3)}{1*(-1)*(-2)} + 2 * \frac{(4-0)(4-1)(4-3)}{2*1*(-1)} + 7 * \frac{(4-0)(4-1)(4-2)}{3*2*1}$$

$$= 1 + 4 - 12 + 28 = 21$$

Since $p(4) = 21 \neq 2$, the point is not in the polynomial curve. It means that if the function across the point (4, 2), the degree of polynomial function must not be 3 because for any values at the nodes, there must is exactly one polynomial (Uniqueness).

Q3.

```
function polyvals = polyinterp(x,y,u)
    n = length(x);
    y_l = length(y);
    if y_l ~= n:
        error('x and y must be same size');
    end

    m = length(u);
    polyvals = zeros(size(u));

    for i=1:m
        v = 0;  %Initialize the term and store
        for k=1:n
            w = 1; %Initialize the term and store
            for j = [1:k-1 k+1:n]  %j cannot be equal to k
                w = w*(u(i)-x(j))/(x(k)-x(j));
            end
            v = v+w*y(k);
        end
        polyvals(i) = v;
    end
    plot(x, y, 'o', u, polyvals, '-');
end
```

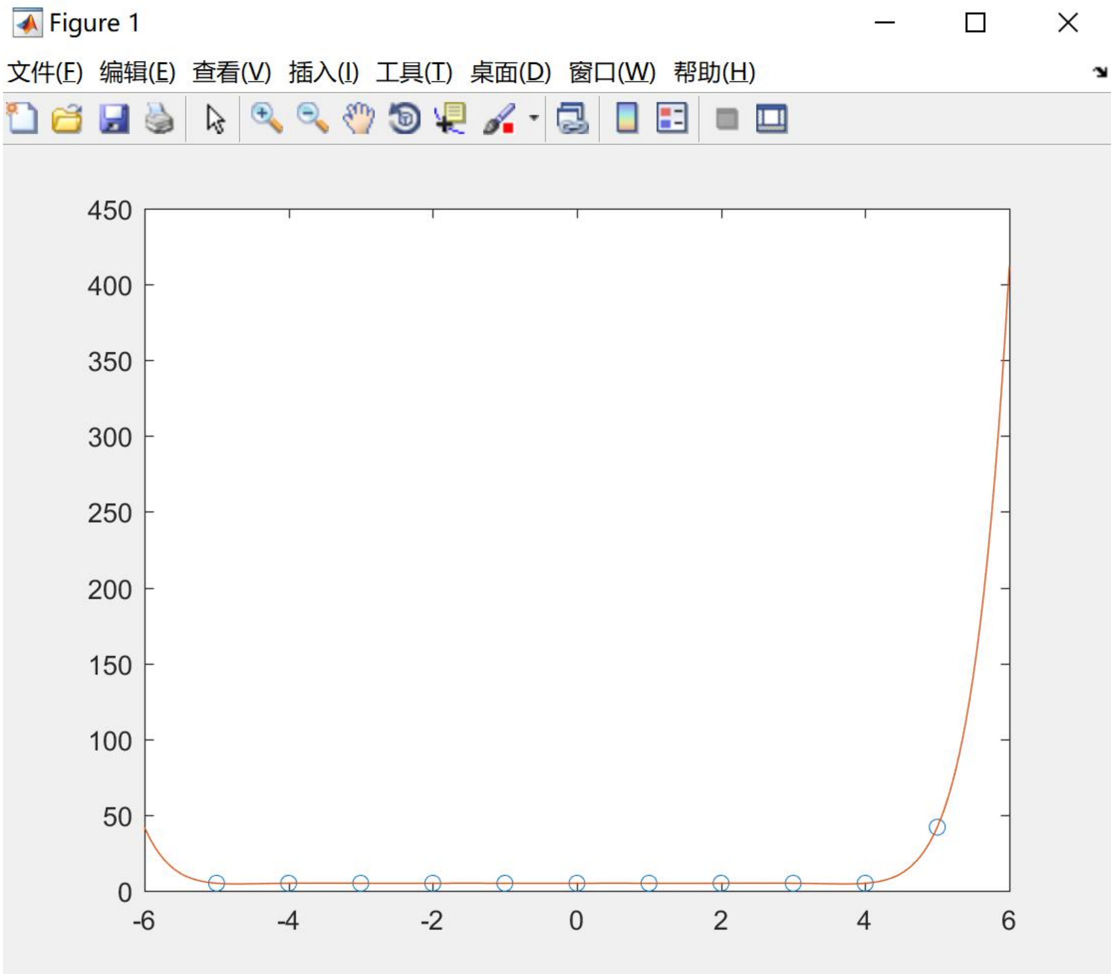
Q4.

```
function polyvals = diff_newtoninterp(x, y, u)
    n = length(x);
    y_l = length(y);
    m = length(u);
    polyvals = zeros(size(u));
    if n~=y_l
        error('x and y must be same size');
    else
        F = zeros(n, n);  % initalize
        for i=1:n
            F(i,1)=y(i);
        end
        for k=i:n-1
            for j=1:n-k
                F(j,k+1)=(F(j+1,k)-F(j,k))/(x(j+k)-x(j));
            end
        end
        for a=1:m
            v = F(1,n);
            for b=n-1:-1:1
                v=F(1,b)+(u(a)-x(b))*v;
            end
            polyvals(a) = v;
        end
        plot(x, y, 'o', u, polyvals, '-')
    end
end
```

Q5.

There are 11 points (−5; 5); (−4; 5); (−3; 5); (−2; 5); (−1; 5); (0; 5); (1; 5); (2; 5); (3; 5); (4; 5); (5; 42). Put the x, y, u into the two function and we can see the same result:

```
>> x=[-5,-4,-3,-2,-1,0,1,2,3,4,5]

x =

  1 至 10 列

    -5    -4    -3    -2    -1     0     1     2     3     4

  11 列

     5

>> y=[5,5,5,5,5,5,5,5,5,5,42]

y =

  1 至 10 列

     5     5     5     5     5     5     5     5     5     5

  11 列

    42

>> u=-6:0.05:6
```




And part of the return result is:

```
>> polyinterp(x, y, u)

ans =

  1 至 12 列

   42.0000    36.8960    32.3808    28.3977    24.8941    21.8222    19.1378    16.8006    14.7737    13.0235    11.5193    10.2332

  13 至 24 列

    9.1398     8.2163     7.4418     6.7977     6.2671     5.8348     5.4874     5.2127     5.0000     4.8397     4.7234     4.6437

  25 至 36 列

    4.5940     4.5687     4.5627     4.5720     4.5928     4.6219     4.6569     4.6954     4.7358     4.7764     4.8162     4.8542
```

(b).

The spend of the $div\_diff\_newton$ function is faster than the $polyinterp$ function. Since in the $div\_diff\_newton$, for each element in a vector not need to create the divided difference table again, it just takes one time to generate and change the input elements just okay. So, the sum of the flops is $(n-1)*3$ for each element. In the $polyinterp$ function, each element should experience two nested layers of loop, every time it takes $\theta\big(n*(n-1)\big)=\theta(n^2)$ flops. In conclusion, $div\_diff\_newton$ function is faster than the $polyinterp$ function.

(c).

We know that n degree polynomial has maximum n roots. In this question, the degree of polynomial function is 10, so the function must have 10 roots at most. Then let $Q(x) = P(x) - 5$. Then the root of $Q(x)$ is (−5; 0), (−4; 0), (−3; 0), (−2; 0), (−1; 0), (0; 0), (1; 0), (2; 0), (3; 0), (4, 0).

by Lagrange Polynomials:

$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x-x_j}{x_i-x_j}$, $P_n(x) = \sum_{i=0}^{n} [y_i * L_i(x)]$.

$$Q(6) = \frac{(\mathrm{x}-\mathrm{x}_0)(\mathrm{x}-\mathrm{x}_1)(\mathrm{x}-\mathrm{x}_2)(\mathrm{x}-\mathrm{x}_3)(\mathrm{x}-\mathrm{x}_4)(\mathrm{x}-\mathrm{x}_5)(\mathrm{x}-\mathrm{x}_6)(\mathrm{x}-\mathrm{x}_7)(\mathrm{x}-\mathrm{x}_8)(\mathrm{x}-\mathrm{x}_9) * y_{10}}{(\mathrm{x}_{11}-\mathrm{x}_0)(\mathrm{x}_{11}-\mathrm{x}_1)(\mathrm{x}_{11}-\mathrm{x}_2)(\mathrm{x}_{11}-\mathrm{x}_3)(\mathrm{x}_{11}-\mathrm{x}_4)(\mathrm{x}_{11}-\mathrm{x}_5)(\mathrm{x}_{11}-\mathrm{x}_6)(\mathrm{x}_{11}-\mathrm{x}_7)(\mathrm{x}_{11}-\mathrm{x}_8)(\mathrm{x}_{11}-\mathrm{x}_9)}$$

$+0 + \ldots + 0 = 37 \times \frac{11*10*9*8*7*6*5*4*3*2}{10*9*8*7*6*5*4*3*2*1} = 407$ .

So, $P(6) = Q(6) + 5 = 412$.