作业二

姓名 郑龙韬 学号 **PB18061352** 日期 **2021/5/31**

第一题 本题考虑对于定义在 $[-1,1]$ 上的一个光滑函数 $f(x)$ 的三次样条插值的使用。以下误差均指绝对误差。

(a) 解: 给定插值点 $\{(x_i, f(x_i)), i = 0, 1, \ldots, n\}$, 记 $S(x)$ 在区间 $[x_i, x_{i+1}]$ 上的表达式为 $S_i(x)$,$S(x)$ 是三次多项式,$S''(x)$ 是线性函数,用插值点 $\{(x_i, S''(x_i)), (x_{i+1}, S''(x_{i+1}))\}$ 作线性插值。记 $\{M_i = S''(x_i), m_i = S'(x_i), i = 0, 1, \ldots, n\}$, 则

$$S_i''(x) = \frac{x - x_{i+1}}{x_i - x_{i+1}} M_i + \frac{x - x_i}{x_{i+1} - x_i} M_{i+1},\ x_i \le x \le x_{i+1}$$

积分两次 $S''(x)$, 记 $h_i = x_{i+1} - x_i$,

$$\begin{aligned} S(x) = S_i(x) &= \frac{(x_{i+1} - x)^3}{6h_i} M_i + \frac{(x - x_i)^3}{6h_i} M_{i+1} + cx + d \\ &= \frac{(x_{i+1} - x)^3}{6h_i} M_i + \frac{(x - x_i)^3}{6h_i} M_{i+1} + C(x_{i+1} - x) + D(x - x_i) \end{aligned}$$

将 $S(x_i) = y_i$, $S(x_{i+1}) = y_{i+1}$ 代入得

$$\begin{aligned} C &= \frac{y_i}{h_i} - \frac{h_i M_i}{6} \\ D &= \frac{y_{i+1}}{h_i} - \frac{h_i M_{i+1}}{6} \end{aligned}$$

$$\begin{aligned} S(x) =& \frac{(x_{i+1} - x)^3}{6h_i} M_i + \frac{(x - x_i)^3}{6h_i} M_{i+1} + \frac{y_i}{h_i}(x_{i+1} - x) + \frac{y_{i+1}}{h_i}(x - x_i) \\ &- \frac{h_i}{6}[(x_{i+1} - x) M_i + (x - x_i) M_{i+1}],\ x \in [x_i, x_{i+1}] \end{aligned}$$

在内结点 x_i，由 $S_i'(x_i) = S_{i-1}'(x_i)$ 可得

$$f(x_i, x_{i+1}) - \frac{h_i}{3}M_i - \frac{h_i}{6}M_{i+1} = f(x_{i-1}, x_i) + \frac{h_{i-1}}{6}M_{i-1} + \frac{h_{i-1}}{3}M_i$$

记

$$\begin{aligned}
&\lambda_i = \frac{h_i}{h_i + h_{i-1}} \\
&\mu_i = 1 - \lambda_i \\
&d_i = \frac{6}{h_i + h_{i-1}}(\frac{y_{i+1} - y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}}) = 6f(x_{i-1}, x_i, x_{i+1})
\end{aligned}$$

那么得到样条插值的 M 关系方程组

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i,\ i = 1, 2, \ldots, n-1 \tag{1}$$

解方程组得到 $\{M_i, i = 1, 2, \ldots, n-1\}$。若已知边界处 $S'(-1) = m_0$ 和 $S'(1) = m_n$ 的值，分别代入 $S'(x)$ 在 $[x_0, x_1]$ 和 $[x_{n-1}, x_n]$ 中的表达式，得到另外两个方程

$$\begin{aligned}
2M_0 + M_1 &= \frac{6}{h_0}[f[x_o, x_1] - m_0] = d_0 \\
M_{n-1} + 2M_n &= \frac{6}{h_{n-1}}[m_n - f[x_{n-1}, x_n]] = d_n
\end{aligned}$$

则得到 $n+1$ 个未知量，$n+1$ 个方程组如下

$$\begin{bmatrix}
2 & 1 & & & & \\
\mu_1 & 2 & \lambda_1 & & & \\
& \mu_2 & 2 & \lambda_2 & & \\
& & \ddots & \ddots & \ddots & \\
& & & \mu_{n-1} & 2 & \lambda_{n-1} \\
& & & & 1 & 2
\end{bmatrix}
\begin{bmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n \end{bmatrix}
=
\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

由此便推导出了关于额外给定边界点处（-1 和 1）三次样条插值多项式的一次导数值时，其在各插值点上的二次导数值应该满足的线性方程组。

(b) 使用 $n = 2^4$ 个子区间插值一个定义在 $[-1, 1]$ 上的函数 $f(x) = \sin(4x^2) + \sin^2(4x)$ 并使用 semilogy 图通过在 2000 个等距点上取真实值画出构造的三次样条插值的逐点误差，如图 1。

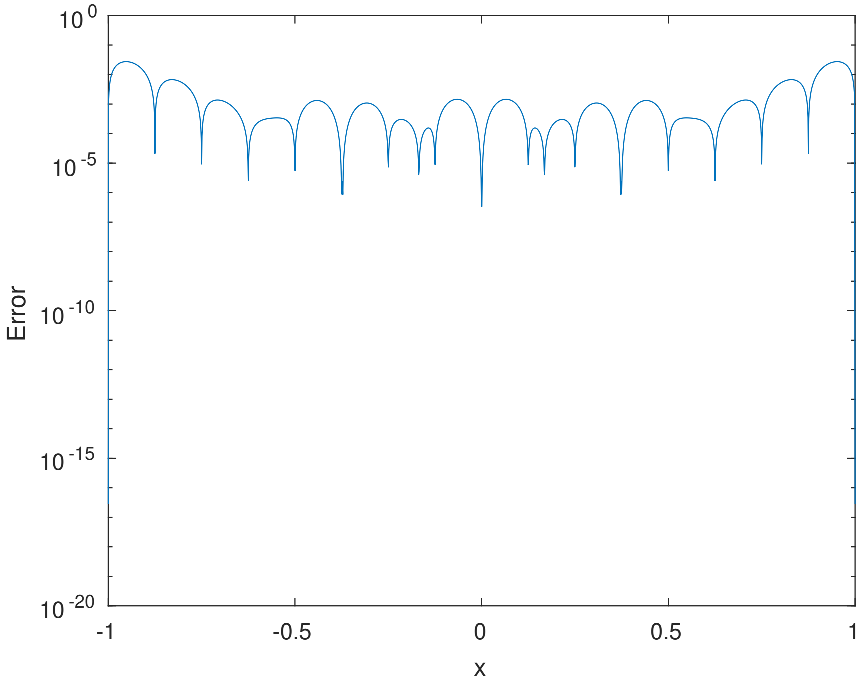

图 1: 三次样条插值的逐点误差

本题 Matlab 代码如下：

```
clear, clc
format long
n = 2^4;  % num of points
% define function
f = @(x) sin(4*(x.^2)) + sin(4*x).^2;
% boundary conditions
m_0 = 0;
m_n = 0;
% cubic spline
all_x = linspace(-1,1,n+1);
y = f(all_x);
h = all_x(2:n+1)-all_x(1:n);
delta = (y(2:n+1)-y(1:n))./h;
d = 6./(h(1:n-1) + h(2:n)).*(delta(2:n)-delta(1:n-1));
lambda = h(2:n)./(h(1:n-1) + h(2:n));
mu = 1.-lambda;
T = diag([mu 1], -1)+diag(2*ones(n+1, 1))...
```

```
    +diag([1 lambda], 1);
d_0 = 6/h(1)*(delta(1)-m_0);
d_n = 6/h(n)*(m_n-delta(n));
M = T\[d_0 d d_n]';
% compute error
test_points = linspace(-1,1,2000);
S = [];
for i = 1:2000
    point = test_points(i);
    idx = 0;
    for k = 1:n  % find x_i and x_{i+1}
        if point < all_x(k+1) && point >= all_x(k)
            idx = k;
        end
    end
    if point == 1  % right range value
        idx = n;
    end
    item1 = ((all_x(idx+1)-point)^3*M(idx)...
        +(point-all_x(idx))^3*M(idx+1))/(6*h(idx));
    item2 = ((all_x(idx+1)-point)*y(idx)...
        +(point-all_x(idx))*y(idx+1))/h(idx);
    item3 = h(idx)/6*((all_x(idx+1)-point)*M(idx)...
        +(point-all_x(idx))*M(idx+1));
    S(i) = item1+item2-item3;
end
error = abs(S - f(test_points));
% draw semilogy
semilogy(test_points, error)
xlabel('x')
ylabel('Error')
```

(c) 令 $n = 2^4, 2^5, \ldots, 2^{10}$，描述插值区间上最大误差值随 n 变化情况的 loglog 图，如图 2。

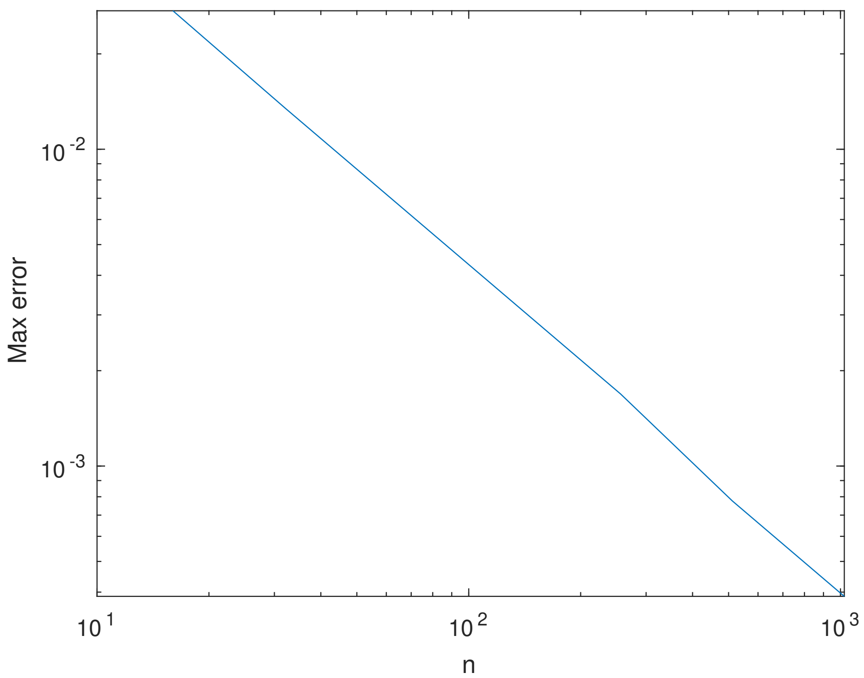

图 2: 插值区间上最大误差值随 n 变化的情况

本题 Matlab 代码如下：

```
clear, clc
format long
n_s = [2^4, 2^5, 2^6, 2^7, 2^8, 2^9, 2^10];% num of points
errors = zeros(1, size(n_s, 2));
for i = 1:size(n_s, 2)
    errors(i) = max(cubic(n_s(i)));
end
% draw loglog
loglog(n_s, errors)
xlabel('n')
ylabel('Max error')

function error = cubic(n)
    % define function
    f = @(x) sin(4*(x.^2)) + sin(4*x).^2;
    % boundary conditions
    m_0 = 0;
```

```
m_n = 0;
% cubic spline
all_x = linspace(-1,1,n+1);
y = f(all_x);
h = all_x(2:n+1)-all_x(1:n);
delta = (y(2:n+1)-y(1:n))./h;
d = 6./(h(1:n-1) + h(2:n)).*(delta(2:n)-delta(1:n-1));
lambda = h(2:n)./(h(1:n-1) + h(2:n));
mu = 1.-lambda;
T = diag([mu 1], -1)+diag(2*ones(n+1, 1))...
    +diag([1 lambda], 1);
d_0 = 6/h(1)*(delta(1)-m_0);
d_n = 6/h(n)*(m_n-delta(n));
M = T\[d_0 d d_n]';
% compute error
test_points = linspace(-1,1,2000);
S = [];
for i = 1:2000
    point = test_points(i);
    idx = 0;
    for k = 1:n  % find x_i and x_{i+1}
        if point < all_x(k+1) && point >= all_x(k)
            idx = k;
        end
    end
    if point == 1  % right range value
        idx = n;
    end
    item1 = ((all_x(idx+1)-point)^3*M(idx)...
        +(point-all_x(idx))^3*M(idx+1))/(6*h(idx));
    item2 = ((all_x(idx+1)-point)*y(idx)...
        +(point-all_x(idx))*y(idx+1))/h(idx);
    item3 = h(idx)/6*((all_x(idx+1)-point)*M(idx)...
        +(point-all_x(idx))*M(idx+1));
    S(i) = item1+item2-item3;
```

```
    end
    error = abs(S - f(test_points));
end
```

(d) 假设三次样条函数满足 $S'(-1) = S'(1)$ 和 $S''(-1) = S''(1)$。

(d.a) 推导过程:

若已知边界处 $S'(-1) = S'(1)$ 和 $S''(-1) = S''(1)$，得到的两个方程为

$$M_0 - M_n = 0 = d_0$$

$$\frac{h_0}{h_0 + h_{n-1}} M_1 + \frac{h_{n-1}}{h_0 + h_{n-1}} M_{n-1} + 2M_n = \frac{6(f[x_0, x_1] - f[x_{n-1}, x_n])}{h_0 + h_{n-1}} = d_n$$

则得到 $n+1$ 个未知量，$n+1$ 个方程组如下

$$\begin{bmatrix} 1 & & & & & -1 \\ & 2 & \lambda_1 & & & \mu_1 \\ & \mu_2 & 2 & \lambda_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \mu_{n-1} & 2 & \lambda_{n-1} \\ & \frac{h_0}{h_0+h_{n-1}} & & & \frac{h_{n-1}}{h_0+h_{n-1}} & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

其中 λ, μ 等符号的定义与其他推导过程均同题 (a)。由此便推导出了针对周期边界，三次样条插值多项式在各插值点上的二次导数值应该满足的线性方程组。

(d.b) 使用 $n = 2^4$ 个子区间插值一个定义在 $[-1, 1]$ 上的函数 $f(x) = \sin(4x^2) + \sin^2(4x)$ 并使用 semilogy 图通过在 2000 个等距点上取真实值画出构造的三次样条插值的逐点误差，如图 3。

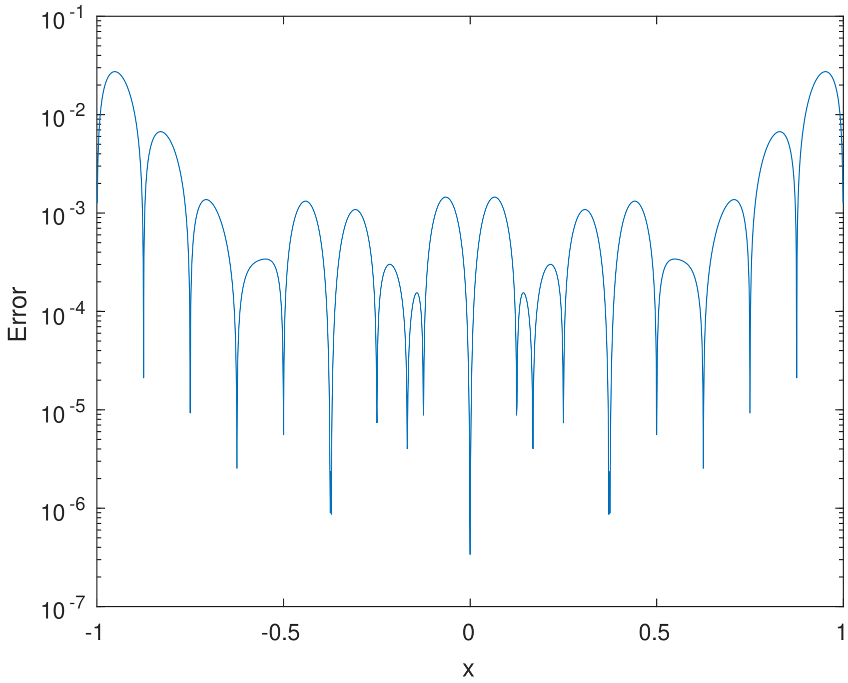

图 3: 三次样条插值的逐点误差

本题 Matlab 代码如下:

```
clear, clc
format long
n = 2^4;  % num of points
% define function
f = @(x) sin(4*(x.^2)) + sin(4*x).^2;
% cubic spline
all_x = linspace(-1,1,n+1);
y = f(all_x);
h = all_x(2:n+1)-all_x(1:n);
delta = (y(2:n+1)-y(1:n))./h;
d = 6./(h(1:n-1) + h(2:n)).*(delta(2:n)-delta(1:n-1));
lambda = h(2:n)./(h(1:n-1) + h(2:n));
mu = 1.-lambda;
T = diag([mu 0], -1)+diag(2*ones(n+1, 1))...
    +diag([0 lambda], 1);
T(1, 1) = 1;
T(1, n+1) = -1;
```

```
T(2, 1) = 0;
T(2, n+1) = mu(1);
T(n+1, 2) = h(1)/(h(1)+h(n));
T(n+1, n) = 1-T(n+1, 2);
% boundary conditions
d_0 = 0;
d_n = 6*(delta(1)-delta(n))/(h(1)+h(n));
M = T\[d_0 d d_n]';
% compute error
test_points = linspace(-1,1,2000);
S = [];
for i = 1:2000
    point = test_points(i);
    idx = 0;
    for k = 1:n  % find x_i and x_{i+1}
        if point < all_x(k+1) && point >= all_x(k)
            idx = k;
        end
    end
    if point == 1  % right range value
        idx = n;
    end
    item1 = ((all_x(idx+1)-point)^3*M(idx)...
        +(point-all_x(idx))^3*M(idx+1))/(6*h(idx));
    item2 = ((all_x(idx+1)-point)*y(idx)...
        +(point-all_x(idx))*y(idx+1))/h(idx);
    item3 = h(idx)/6*((all_x(idx+1)-point)*M(idx)...
        +(point-all_x(idx))*M(idx+1));
    S(i) = item1+item2-item3;
end
error = abs(S - f(test_points));
% draw semilogy
semilogy(test_points, error)
xlabel('x')
ylabel('Error')
```

(d.c) 令 $n = 2^4, 2^5, \ldots, 2^{10}$，描述插值区间上最大误差值随 n 变化情况的 loglog 图，如图 4。

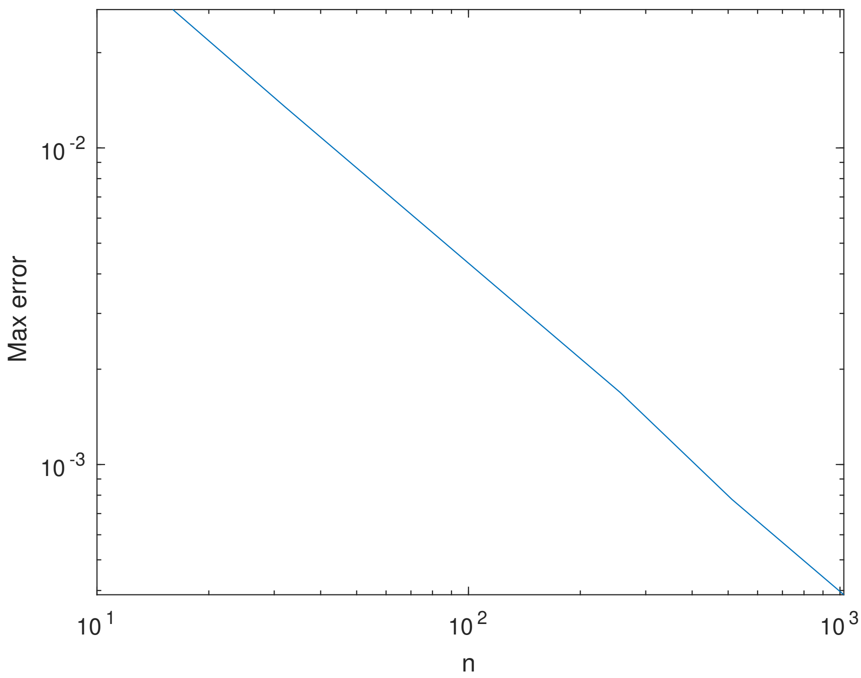

图 4: 插值区间上最大误差值随 n 变化的情况

本题 Matlab 代码如下：

```
clear, clc
format long
n_s = [2^4, 2^5, 2^6, 2^7, 2^8, 2^9, 2^10];% num of points
errors = zeros(1, size(n_s, 2));
for i = 1:size(n_s, 2)
    errors(i) = max(cubic(n_s(i)));
end
% draw loglog
loglog(n_s, errors)
xlabel('n')
ylabel('Max error')

function error = cubic(n)
    % define function
    f = @(x) sin(4*(x.^2)) + sin(4*x).^2;
```

```
% cubic spline
all_x = linspace(-1,1,n+1);
y = f(all_x);
h = all_x(2:n+1)-all_x(1:n);
delta = (y(2:n+1)-y(1:n))./h;
d = 6./(h(1:n-1) + h(2:n)).*(delta(2:n)-delta(1:n-1));
lambda = h(2:n)./(h(1:n-1) + h(2:n));
mu = 1.-lambda;
T = diag([mu 0], -1)+diag(2*ones(n+1, 1))...
+diag([0 lambda], 1);
T(1, 1) = 1;
T(1, n+1) = -1;
T(2, 1) = 0;
T(2, n+1) = mu(1);
T(n+1, 2) = h(1)/(h(1)+h(n));
T(n+1, n) = 1-T(n+1, 2);
% boundary conditions
d_0 = 0;
d_n = 6*(delta(1)-delta(n))/(h(1)+h(n));
M = T\[d_0 d d_n]';
% compute error
test_points = linspace(-1,1,2000);
S = [];
for i = 1:2000
    point = test_points(i);
    idx = 0;
    for k = 1:n  % find x_i and x_{i+1}
        if point < all_x(k+1) && point >= all_x(k)
            idx = k;
        end
    end
    if point == 1  % right range value
        idx = n;
    end
    item1 = ((all_x(idx+1)-point)^3*M(idx)...
```

```
            +(point-all_x(idx))^3*M(idx+1))/(6*h(idx));
        item2 = ((all_x(idx+1)-point)*y(idx)...
            +(point-all_x(idx))*y(idx+1))/h(idx);
        item3 = h(idx)/6*((all_x(idx+1)-point)*M(idx)...
            +(point-all_x(idx))*M(idx+1));
        S(i) = item1+item2-item3;
    end
    error = abs(S - f(test_points));
end
```

第二题 本题深入讨论 Newton 插值公式的性质。

(a) 证明:

首先由归纳法证明，差商可以表示为函数值的线性组合，即

$$f[x_0, x_1, \ldots, x_k] = \sum_{j=0}^{k} \frac{f(x_j)}{\Pi_{i=0,i\neq j}^{k}(x_j - x_i)}$$

证明如下，当 $k = 1$ 时，一阶差商的定义为

$$f[x_0, x_1] = \frac{f(x_0) - f(x_1)}{x_0 - x_1}$$

显然，一阶差商的定义符合要证的等式。

当 $k > 1$ 时，假设 $k - 1$ 时满足上述线性组合的等式，展开得

$$f[x_0, \ldots, x_{k-1}] = \sum_{j=0}^{k-1} \frac{f(x_j)}{\Pi_{i=0,i\neq j}^{k-1}(x_j - x_i)}$$

$$f[x_1, \ldots, x_k] = \sum_{j=1}^{k} \frac{f(x_j)}{\Pi_{i=1,i\neq j}^{k}(x_j - x_i)}$$

由定义，由两个 $k - 1$ 阶差商的值计算 k 阶差商

$$\begin{aligned} f[x_0, x_1, \ldots, x_k] &= \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0} \\ &= \sum_{j=0}^{k} \frac{f(x_j)}{\Pi_{i=0,i\neq j}^{k}(x_j - x_i)} \end{aligned}$$

由此得证，上述等式对 $\forall k, k > 0$ 成立。可见差商与节点的排序无关。对一个光滑函数 $f(x)$，若 $i_0, i_1, ..., i_k$ 是 $0, 1, ..., k$ 的任意一个排列，则 $f[x_0, x_1, ..., x_k] = f[x_{i_0}, x_{i_1}, ..., x_{i_k}]$。证毕。

(b) 令 $n = 2^2, 2^3, \ldots, 2^7$，描述插值区间上最大误差值随 n 变化情况的 semilogy 图，如图 5。

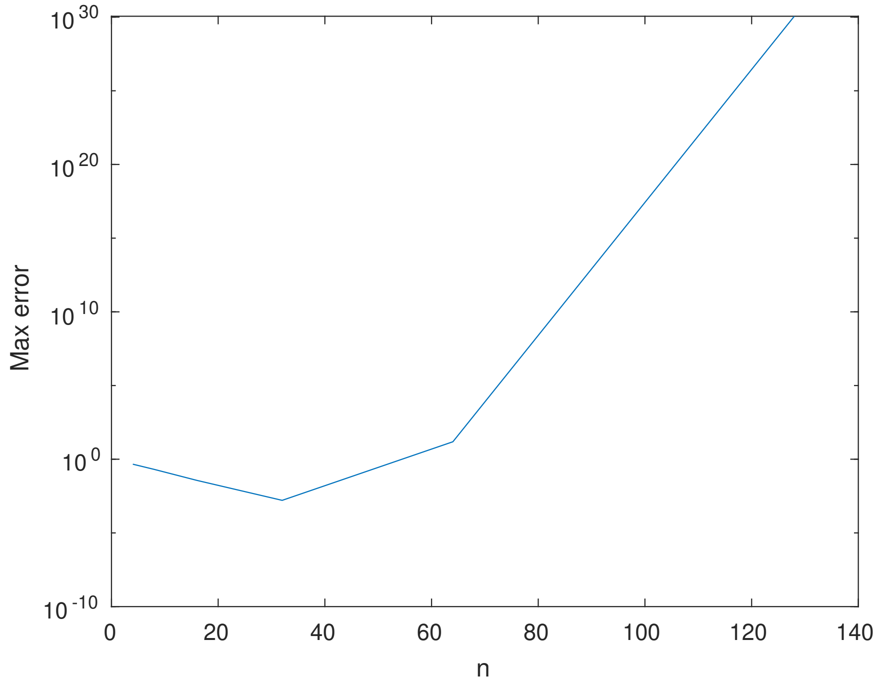

图 5: 插值区间上最大误差值随 n 变化的情况

本题 Matlab 代码如下：

```
clear, clc
format long
n_s = [2^2, 2^3, 2^4, 2^5, 2^6, 2^7];% num of points
errors = zeros(1, size(n_s, 2));
for i = 1:size(n_s, 2)
    errors(i) = max(newton(n_s(i)));
end
% draw semilogy
semilogy(n_s, errors)
xlabel('n')
ylabel('Max error')

function error = newton(n)
    % define function
    f = @(x) 1./(1+25*x.^2);
    all_x = zeros(1, n+1);
    for i = 1:n+1  % chebyshev
```

```
        all_x(i) = cos((i-1)*pi/n);
    end
    % table of difference quotient
    g = f(all_x);
    for k = 1:n  % compute k-order diff
        for j = n+1:-1:k+1
            g(j) = (g(j)-g(j-1))/(all_x(j)-all_x(j-k));
        end
    end
    % error of newton interpolation
    test_points = linspace(-1,1,2000);
    error = zeros(1, 2000);
    for i = 1:2000
        u = test_points(i);
        t = 1;
        val = g(1);
        for k = 2:n+1
            t = t*(u-all_x(k-1));
            val = val+t*g(k);
        end
        error(i) = abs(val - f(u));
    end
end
```

(c) 使用随机数种子 rng(22) 和 randperm 函数来随机计算差商时插值点的使用顺序，得到 semilogy 图，如图 6。

本题 Matlab 代码如下：

```
clear, clc
format long
rng(22);
n_s = [2^2, 2^3, 2^4, 2^5, 2^6, 2^7];% num of points
errors = zeros(1, size(n_s, 2));
for i = 1:size(n_s, 2)
    errors(i) = max(newton(n_s(i)));
end
```

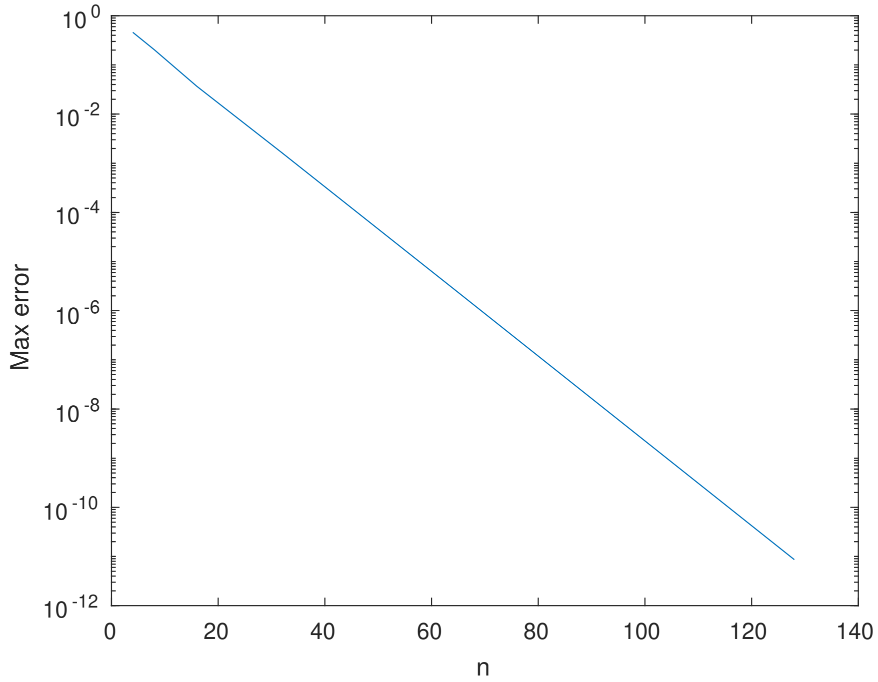

图 6: 插值区间上最大误差值随 n 变化的情况

```
% draw semilogy
semilogy(n_s, errors)
xlabel('n')
ylabel('Max error')

function error = newton(n)
    % define function
    f = @(x) 1./(1+25*x.^2);
    p = randperm(n+1);
    for i = 1:n+1   % chebyshev
        all_x(i) = cos((p(i)-1)*pi/n);
    end
    % table of difference quotient
    g = f(all_x);
    for k = 1:n  % compute k-order diff
        for j = n+1:-1:k+1
            g(j) = (g(j)-g(j-1))/(all_x(j)-all_x(j-k));
        end
```

```
    end
    % error of newton interpolation
    test_points = linspace(-1,1,2000);
    error = zeros(1, 2000);
    for i = 1:2000
        u = test_points(i);
        t = 1;
        val = g(1);
        for k = 2:n+1
            t = t*(u-all_x(k-1));
            val = val+t*g(k);
        end
        error(i) = abs(val - f(u));
    end
end
```

(d) 上面两小问中观察到的不同现象产生的原因：使用 Chebyshev 点可以有效地克服 Runge 现象，但这仅在理论上保证收敛。实际操作中，由于计算机有限的精度，基于 Chebyshev 点的插值多项式不会收敛到 $f(x)$。理论上存在无限位数的浮点数，但实际上计算机只能用有限位数的浮点数来表示实数，这就导致了舍入误差；由此还引入了 overflow 的问题。这些因素会导致当位数增加时，由于表示能力限制，在计算差商和误差时产生数值不稳定性，从而在上两问产生了不同现象。

第三题 本题讨论周期函数的 Lagrange 插值方法。对于周期函数而言，多项式不再是最有效的基函数，而等距插值点也不再会出现 Runge 现象。逼近周期函数的基函数通常选用三角函数或者复指数。注意，对于周期函数而言，插值点数量和子区间个数相等。

(a) 证明：

在 $[0,1]$ 上关于周期函数的基于等间距插值点 $x_j = \frac{j}{n}, j = 0, 1, \ldots, n-1$ 的 Lagrange 插值基函数为

$$\ell_k(x) = \begin{cases} \frac{(-1)^k}{n}\sin(n\pi x)\csc(\pi(x - x_k)) & n\text{为奇数} \\ \frac{(-1)^k}{n}\sin(n\pi x)\cot(\pi(x - x_k)) & n\text{为偶数} \end{cases}$$

Case1：当 $k \neq j$ 时，$\sin(n\pi x_j) = \sin(j\pi) = 0$，所以 $\ell_k(x_j) = 0$。

Case2：当 $k = j$ 时，

i) 若 n 为奇数

$$
\begin{aligned}
\ell_k(x_j) &= \frac{(-1)^k}{n}\sin(n\pi x_j)\csc(\pi(x_j-x_k)) \\
&= \frac{(-1)^k\sin(n\pi x_j)}{n\sin(\pi(x_j-x_k))} \\
&= \frac{(-1)^k\sin(j\pi)}{n\sin(\frac{j-k}{n}\pi)}
\end{aligned}
$$

$\lim_{j\to k}\sin(j\pi)=\lim_{j\to k}\sin(\frac{j-k}{n}\pi)=0$，由 L'Hôpital's rule

$$
\begin{aligned}
\lim_{j\to k}\ell_k(x_j) &= \lim_{j\to k}\frac{(-1)^k\sin(j\pi)}{n\sin(\frac{j-k}{n}\pi)} \\
&= \lim_{j\to k}\frac{(-1)^k\pi\cos(j\pi)}{\pi\cos(\frac{j-k}{n}\pi)} \\
&= (-1)^{2k} \\
&= 1
\end{aligned}
$$

ii) 若 n 为偶数

$$
\begin{aligned}
\ell_k(x_j) &= \frac{(-1)^k}{n}\sin(n\pi x)\cot(\pi(x-x_k)) \\
&= \frac{(-1)^k\sin(n\pi x_j)\cos(\pi(x_j-x_k))}{n\sin(\pi(x_j-x_k))} \\
&= \frac{(-1)^k\sin(j\pi)\cos(\frac{j-k}{n}\pi)}{n\sin(\frac{j-k}{n}\pi)} \\
&= \frac{(-1)^k\sin(j\pi)}{n\sin(\frac{j-k}{n}\pi)}
\end{aligned}
$$

同理有 $\lim_{j\to k}\ell_k(x_j)=1$。

综上，

$$
\ell_k(x_j)=\begin{cases}1 & k=j\\ 0 & k\neq j\end{cases}
$$

(b) 用上述对应于 n 为偶数的 Lagrange 基函数构造 Lagrange 插值多项式，并用 $n=2^6$ 个点对周期函数 $f(x)=\sin(2\pi x)e^{\cos(2\pi x)}$ 在 $[0,1]$ 上进行插值。取 1000 个等距点上的误差，用 semilogy 图描述插值区间上误差值随 x 变化的情况，如图 7。

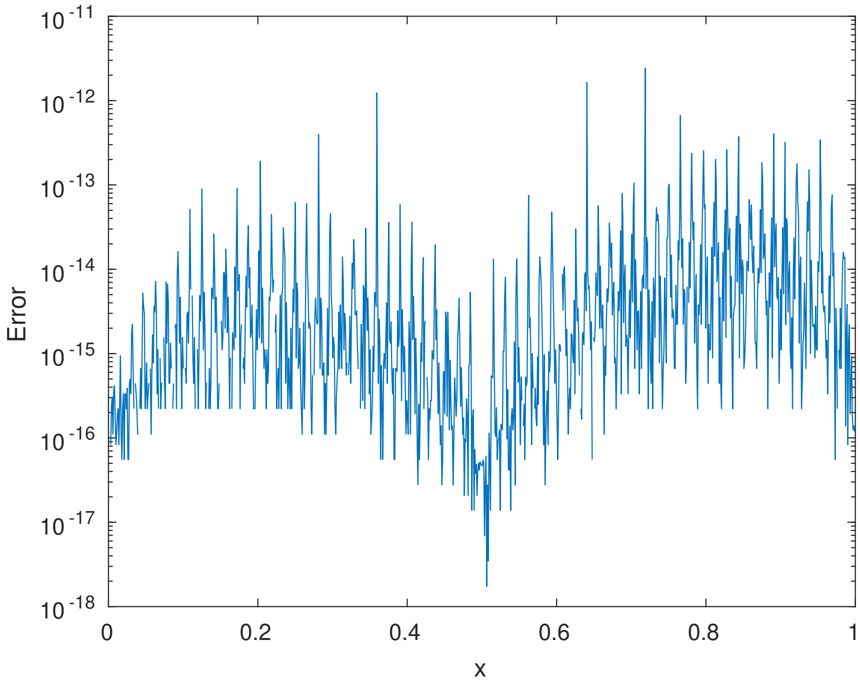

图 7: 插值区间上误差值随 x 变化的情况

本题 Matlab 代码如下：

```
clear, clc
format long
n = 2^6;   % num of points
% define function
f = @(x) sin(2*pi.*x).*exp(1).^(cos(2*pi.*x));
all_x = linspace(0,1,n+1);
all_x = all_x(1:end-1);
y = f(all_x);
% error of lagrange interpolation
test_points = linspace(0,1,1000);
error = zeros(1, 1000);
for i = 1:1000
    u = test_points(i);
    l = zeros(1, n);
    for k = 1:n
        l(k) = l(k)+(-1)^(k-1)/n*sin(n*pi*u)...
            *cot(pi*(u-all_x(k)));
```

```
    end
    L = l*y';
    error(i) = abs(L - f(u));
end
% draw semilogy
semilogy(test_points, error)
xlabel('x')
ylabel('Error')
```

第四题 对表 1 中数据作形如 $f(x)=\frac{x}{a+bx}$ 的拟合函数。

根据最小二乘法的思想，为方便对 a 和 b 求偏导，定义 $Q=\sum|\frac{1}{f(x_i)}-\frac{1}{y_i}|^2$ 为衡量 y_i 与 $f(x_i)$ 接近程度的标准，求 a，b 使得 Q 最小。

$$
\begin{aligned}
Q &= \sum_{i=0}^{3}|\frac{1}{f(x_i)}-\frac{1}{y_i}|^2 \\
&= |\frac{a+2.1b}{2.1}-\frac{1}{0.6087}|^2+|\frac{a+2.5b}{2.5}-\frac{1}{0.6849}|^2+|\frac{a+2.8b}{2.8}-\frac{1}{0.7368}|^2+|\frac{a+3.2b}{3.2}-\frac{1}{0.8111}|^2
\end{aligned}
$$

令偏导为 0，整理得到线性方程组（以下系数由 Matlab 计算得到）

$$
\begin{cases}
\frac{\partial Q}{\partial a}=\frac{1727209}{1411200}a+\frac{371}{120}b-\frac{51590245850500}{11534552109939}=0 \\
\frac{\partial Q}{\partial b}=\frac{371}{120}a+8b-\frac{393999070862500}{34603656329817}=0
\end{cases}
$$

由 Matlab 计算得到拟合函数

$$
f(x)=\frac{x}{a+bx}
$$

其中

$$
\begin{cases}
a=\frac{4716255408526900000}{1896591799780939953} \\
b=\frac{2630102038827661250}{5689775399342819859}
\end{cases}
$$

拟合函数对比所给数据点的误差的 2-范数为 0.005978241829150。

代码输出如下：

x_i	2.1	2.5	2.8	3.2
y_i	0.6087	0.6849	0.7368	0.8111

表 1: 题目数据

```
coef =
[ 1727209/1411200, 371/120]
[         371/120,       8]
y =
  51590245850500/11534552109939
 393999070862500/34603656329817
a =
4716255408526900000/1896591799780939953
b =
2630102038827661250/5689775399342819859
error =
   0.005978241829150
```

本题 Matlab 代码如下：

```
clear, clc
format long
syms a b
x_i = [2.1,2.5,2.8,3.2];
y_i = [0.6087,0.6849,0.7368,0.8111];
Q = sum(((a+b*x_i)./x_i-1./y_i).^2);
dQa = diff(Q, a);
dQb = diff(Q, b);
dQa_coef = coeffs(dQa);
dQb_coef = coeffs(dQb);
coef=[dQa_coef(3),dQa_coef(2);dQb_coef(3),dQb_coef(2)]
y=-[dQa_coef(1);dQb_coef(1)]
ab=coef\y;
a=ab(1)
b=ab(2)
f = @(x) x./(a+b*x);
pred = f(x_i);
error = double(norm(pred-y_i, 2))
```