习题三

第一题

已知

$$\int_0^1 \frac{4}{1+x^2}\mathrm{d}x = \pi \tag{1}$$

因此可以通过数值积分来计算π的近似值。

第一问

分别用四点、六点Newton-Cotes公式计算近似值。

解：Cotes系数为

$$C_k^{(n)} = \frac{(-1)^{n-k}}{nk!(n-k)!}\int_0^n \prod_{\substack{0\le i\le n\\ i\ne k}}(x-i)\mathrm{d}x \tag{2}$$

定义Cotes系数函数

```
function result = CotesCoefficient(n, k)

    % 名称：Cotes系数
    % 输入：
    %       n
    %       k
    % 输出：
    %      result: Cotes系数C_k^n

    %% 函数
    syms x;

    result = (- 1) ^ (n - k) / (n * factorial(k) * factorial(n - k));

    % 定义被积函数
    integrand = 1;
    for i = 0: n
        if i ~= k
            integrand = integrand * (x - i);
        end
    end

    % 计算积分
    result = result * int(integrand, 0, n);

end

```

对于等距节点$x_k = a + \frac{b-a}{n}k$，Newton-Cotes公式为

$$\int_a^b f(x)\mathrm{d}x \approx (b-a)\sum_{k=0}^{n} C_k^{(n)} f(x_k) \tag{3}$$

定义Newton-Cotes公式函数

```
function result = NewtonCotesFormula(fun, n, a, b)

    % 名称：Newton-Cotes公式
    % 输入：
    %       fun:    积分函数
    %       n:      积分节点数
    %       a:      积分左边界
    %       b:      积分右边界
    % 输出：
    %      result: Newton-Cotes公式积分值

    %% 函数

    result = 0;
    for k = 0: n
        result = result + CotesCoefficient(n, k) * f(a + (b - a) * k / n);
    end
    result = (b - a) * result;

end

```

主函数

```
clear; clc

% 定义积分函数
fun = @(x) 4 ./ (1 + x .^ 2);

% 计算积分值
int4 = double(NewtonCotesFormula(fun, 3, 0, 1)); % 四点Newton-Cotes公式近似值
int6 = double(NewtonCotesFormula(fun, 5, 0, 1)); % 六点Newton-Cotes公式近似值

% 输出结果
fprintf('四点Newton-Cotes公式近似值为：%.4f\n', int4)
fprintf('六点Newton-Cotes公式近似值为：%.4f\n', int6)

```

输出结果

```
四点Newton-Cotes公式近似值为：3.1385
六点Newton-Cotes公式近似值为：3.1419
```

第二问

分别取$h=0.1$和$h=0.2$，利用复合梯形公式和复合Simpson公式计算π的近似值。

解：等距节点$x_k=a+\frac{b-a}{n}k$的复合梯形公式为

$$\int_a^b f(x)\mathrm{d}x \approx \frac{b-a}{2n}\left(f(a)+2\sum_{k=1}^{n-1}f(x_k)+f(b)\right) \tag{4}$$

等距节点$x_k=a+\frac{b-a}{n}k$的复合Simpson公式为

$$\int_a^b f(x)\mathrm{d}x \approx \frac{b-a}{6n}\left(f(a)+2\sum_{k=1}^{n-1}f(x_k)+4\sum_{k=1}^{n}f\left(\frac{x_{k-1}+x_k}{2}\right)+f(b)\right) \tag{5}$$

定义复合梯形公式函数

```
function result = compoundTrapezoidalFormula(fun, n, a, b)

    % 名称：复合梯形公式
    % 输入：
    %       fun:    积分函数
    %       n:      积分节点数
    %       a:      积分左边界
    %       b:      积分右边界
    % 输出：
    %      result：复合梯形公式积分值

    %% 函数

    result = fun(a) + fun(b);
    for k = 1: n-1
        result = result + 2 * fun(a + (b - a) * k / n);
    end
    result = (b - a) / (2 * n) * result;

end

```

定义复合Simpson公式函数

```
function result = compoundSimpsonFormula(fun, n, a, b)

    % 名称：复合Simpson公式
    % 输入：
    %       fun:    积分函数
    %       n:      积分节点数
    %       a:      积分左边界
    %       b:      积分右边界
    % 输出：
    %      result：复合Simpson公式积分值

    %% 函数

    result = fun(a) + fun(b);
    for k = 1: n-1
```

```
        result = result + 2 * fun(a + (b - a) * k / n);
    end
    for k = 1: n
        result = result + 4 * fun(a + (b - a) * (k - 1 / 2) / n);
    end
    result = (b - a) / (6 * n) * result;

end

```

主函数

```
clear; clc

% 定义积分函数
fun = @(x) 4 ./ (1 + x .^ 2);

% 计算积分值
trapezoidal1 = compoundTrapezoidalFormula(fun, 10, 0, 1); % 间距为0.1的复合梯形公式近似值
trapezoidal2 = compoundTrapezoidalFormula(fun, 5, 0, 1); % 间距为0.2的复合梯形公式近似值
Simpson1 = compoundSimpsonFormula(fun, 10, 0, 1); % 间距为0.1的复合Simpson公式近似值
Simpson2 = compoundSimpsonFormula(fun, 5, 0, 1); % 间距为0.2的复合Simpson公式近似值

% 输出结果
fprintf('间距为0.1的复合梯形公式近似值为：%.5f\n', trapezoidal1)
fprintf('间距为0.2的复合梯形公式近似值为：%.5f\n', trapezoidal2)
fprintf('间距为0.1的复合Simpson公式近似值为：%.10f\n', Simpson1)
fprintf('间距为0.2的复合Simpson公式近似值为：%.10f\n', Simpson2)

```

输出结果

```
间距为0.1的复合梯形公式近似值为：3.13993
间距为0.2的复合梯形公式近似值为：3.13493
间距为0.1的复合Simpson公式近似值为：3.1415926530
间距为0.2的复合Simpson公式近似值为：3.1415926139
```

第三问

把区间$[0,1]$进行n等分，利用复合梯形公式和复合Simpson公式计算π的近似值。若要求误差不超过0.5×10^{-6}，问需要把区间$[0,1]$划分成多少等份。

解：复合梯形公式函数和复合Simpson公式函数见上。

主函数

```
clear; clc

% 定义积分函数
fun = @(x) 4 ./ (1 + x .^ 2);
```

```

trapezoidalNumber = 2;
while abs(compoundTrapezoidalFormula(fun, trapezoidalNumber, 0, 1) - pi) > 0.5 * 10 ^ (-6)
    trapezoidalNumber = trapezoidalNumber + 1;
end

SimpsonNumber = 2;
while abs(compoundSimpsonFormula(fun, SimpsonNumber, 0, 1) - pi) > 0.5 * 10 ^ (-6)
    SimpsonNumber = SimpsonNumber + 1;
end

% 输出结果
fprintf('复合梯形公式需要把区间[0,1]划分成等份%.0f等份\n', trapezoidalNumber)
fprintf('复合Simpson公式需要把区间[0,1]划分成等份%.0f等份\n', SimpsonNumber)

```

输出结果

```
复合梯形公式需要把区间[0,1]划分成等份578等份
复合Simpson公式需要把区间[0,1]划分成等份4等份
```

第四问

选择不同的h，对两种复合求积公式，试将误差描述为h的函数，输出函数表达式。

解：复合梯形公式的积分余项的绝对值为

$$R[f] = \frac{1}{12n^2} f''(\xi), \qquad \xi \in (0,1) \tag{6}$$

使用拟合求出$f''(\xi)$的拟合值

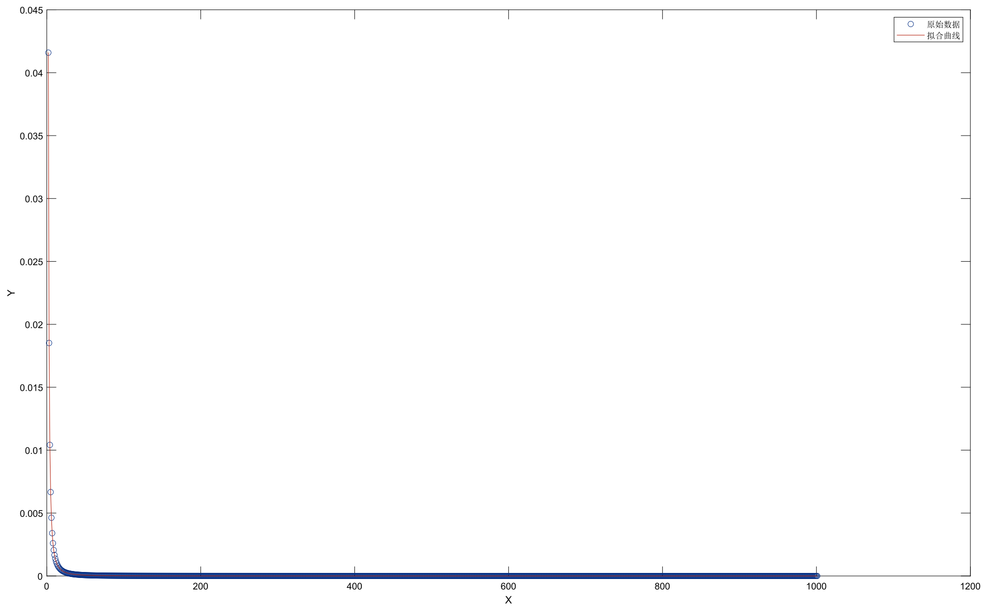
```
clear; clc

% 定义积分函数
fun = @(x) 4 ./ (1 + x .^ 2);

compoundTrapezoidalFormulaError = zeros(1, 1000);
for n = 2: 1001
    compoundTrapezoidalFormulaError(n - 1) = abs(compoundTrapezoidalFormula(fun, n, 0, 1) - pi);
end

X = 2: 1001;
Y = compoundTrapezoidalFormulaError;

% 定义函数模型
model = fittype(@(a, x) a./(12 * x.^2), 'independent', 'x', 'dependent', 'y');

% 初始参数猜测
initialGuess = 1;

```

```
% 进行非线性拟合
fitResult = fit(X', Y', model, 'StartPoint', initialGuess);

% 获取拟合后的参数
a_fit = fitResult.a;

% 计算拟合后的Y
Y_fit = a_fit./(12 * X.^2);

% 计算R方
R_squared = 1 - sum((Y - Y_fit).^2) / sum((Y - mean(Y)).^2);

% 计算RMSE
RMSE = sqrt(sum((Y - Y_fit).^2) / n);

% 计算SSE
SSE = sum((Y - Y_fit).^2);

% 输出结果
fprintf('拟合值: %f\n', a_fit);
fprintf('R方: %f\n', R_squared);
fprintf('RMSE: %f\n', RMSE);
fprintf('SSE: %f\n', SSE);

% 绘制拟合曲线
figure;
plot(X, Y, 'o', X, Y_fit, '-')
legend('原始数据', '拟合曲线');
xlabel('X');
ylabel('Y');
```

输出结果

```
拟合值: 1.997252
R方: 0.999999
RMSE: 0.000001
SSE: 0.000000
```

输出图像



因此复合梯形公式的积分余项的绝对值为

$$R[f]=\frac{1}{6n^2}=\frac{h^2}{6} \tag{7}$$

复合Simpson公式的积分余项的绝对值为

$$R[f]=\frac{1}{2880n^4}f^{(4)}(\xi), \qquad \xi\in(0,1) \tag{8}$$

使用拟合求出$f^{(4)}(\xi)$的拟合值

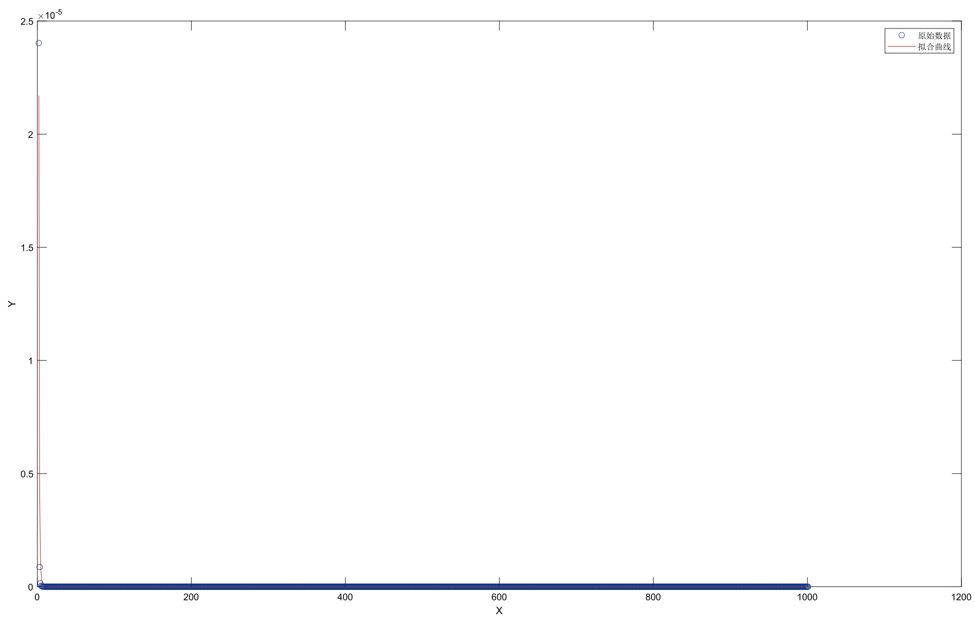
```
clear; clc

% 定义积分函数
fun = @(x) 4 ./ (1 + x .^ 2);

compoundSimpsonFormulaError = zeros(1, 1000);
for n = 2: 1001
    compoundSimpsonFormulaError(n - 1) = abs(compoundSimpsonFormula(fun, n, 0, 1) - pi);
end

X = 2: 1001;
Y = compoundSimpsonFormulaError;

% 定义函数模型
model = fittype(@(a, x) a./(2880 * x.^4), 'independent', 'x', 'dependent', 'y');

% 初始参数猜测
initialGuess = 1;

% 进行非线性拟合
fitResult = fit(X', Y', model, 'StartPoint', initialGuess);

% 获取拟合后的参数
```

```
a_fit = fitResult.a;

% 计算拟合后的Y
Y_fit = a_fit./(2880 * X.^4);

% 计算R方
R_squared = 1 - sum((Y - Y_fit).^2) / sum((Y - mean(Y)).^2);

% 计算RMSE
RMSE = sqrt(sum((Y - Y_fit).^2) / n);

% 计算SSE
SSE = sum((Y - Y_fit).^2);

% 输出结果
fprintf('拟合值: %f\n', a_fit);
fprintf('R方: %f\n', R_squared);
fprintf('RMSE: %f\n', RMSE);
fprintf('SSE: %f\n', SSE);

% 绘制拟合曲线
figure;
plot(X, Y, 'o', X, Y_fit, '-')
legend('原始数据', '拟合曲线');
xlabel('X');
ylabel('Y');
```

输出结果

```
拟合值: 1.000000
R方: 0.967311
RMSE: 0.000000
SSE: 0.000000
```

输出图像



因此复合Simpson公式的积分余项的绝对值为

$$R[f]=\frac{1}{2280n^4}=\frac{h^4}{2280} \tag{9}$$

第二题

分别用三点和五点Gauss-Legendre公式计算积分

$$\int_0^1 \frac{xe^x}{(1+x)^2}\mathrm{d}x=\frac{\mathrm{e}}{2}-1\approx 0.3591409142295 \tag{10}$$

解：区间$[-1,1]$上关于权$\rho=1$的Gauss型求积公式为Gauss-Legendre求积公式

$$\int_{-1}^1 f(x)\mathrm{d}x\approx\sum_{k=1}^n A_k f(x_k) \tag{11}$$

其中求积节点$\{x_k\}_{k=1}^n$为n次Legendre多项式$L_n(x)$的零点，且

$$L_n(x)=\frac{1}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}(x^2-1)^n \tag{12}$$

同时

$$\int_{-1}^1 x^m\mathrm{d}x=\sum_{k=1}^n A_k x_k^m,\qquad 0\le m\le 2n-1 \tag{13}$$

一般的

$$\int_a^b f(x)\mathrm{d}x=\frac{b-a}{2}\int_{-1}^1 f\left(\frac{b-a}{2}x+\frac{b+a}{2}\right)\mathrm{d}x \tag{14}$$

定义Gauss-Legendre求积公式函数

```
function result = GaussLegendreIntegralFormula(fun, n, a, b)

    % 名称：Gauss-Legendre求积公式
    % 输入：
    %       fun:    积分函数
    %       n:      积分节点数
    %       a:      积分左边界
    %       b:      积分右边界
    % 输出：
    %       result: 积分值

    %% 函数

    % 求解Legendre多项式的零点
    syms x
    L = diff((x^2-1)^n, x, n) / (2^n * factorial(n)); % Legendre多项式
    root = solve(L);                                  % Legendre多项式的根

    % 求解权重
    A = zeros(2 * n, n);
    B = zeros(2 * n, 1);
    for k = 0: 2 * n - 1
        A(k + 1, :) = transpose(root .^ k);
```

```
        B(k + 1) = int(x .^ k, -1, 1);
    end
    w = A \ B;

    % 求解积分值
    f = @(x) fun((b - a) / 2 .* x + (b + a) / 2);
    result = (b - a) / 2 * sum(w .* f(root));

end

```

主函数

```
clear; clc

% 定义函数
fun = @(x) x .* exp(x) ./ (1 + x) .^ 2;
% 计算积分值
int3 = GaussLegendreIntegralFormula(fun, 3, 0, 1);
int5 = GaussLegendreIntegralFormula(fun, 5, 0, 1);
% 输出结果
fprintf('三点Gauss-Legendre公式积分值为: %.10f\n', int3)
fprintf('五点Gauss-Legendre公式积分值为: %.10f\n', int5)

```

输出结果

```
三点Gauss-Legendre公式积分值为: 0.3591871703
五点Gauss-Legendre公式积分值为: 0.3591409792
```

第三题

分别用三点和四点Gauss-Lagurre公式计算积分

$$\int_0^{\infty} \mathrm{e}^{-10x} \sin x \mathrm{d}x = \frac{1}{101} \approx 0.00990099 \tag{15}$$

解:

$$\int_0^{\infty} \mathrm{e}^{-10x} \sin x \mathrm{d}x = \int_0^{\infty} \mathrm{e}^{-x} \frac{1}{10} \sin \frac{x}{10} \mathrm{d}x \tag{16}$$

区间$[0, \infty)$上关于权$\rho = \mathrm{e}^{-x}$的Gauss型求积公式为Gauss-Laguerre求积公式

$$\int_0^{\infty} \mathrm{e}^{-x} f(x) \mathrm{d}x \approx \sum_{k=1}^{n} A_k f(x_k) \tag{17}$$

其中求积节点$\{x_k\}_{k=1}^n$为n次Laguerre多项式$L_n(x)$的零点，且

$$L_n(x) = \mathrm{e}^x \frac{\mathrm{d}}{\mathrm{d}x^n} x^n \mathrm{e}^{-x} \tag{18}$$

同时

$$A_k = \frac{((n+1)!)^2}{x_k (L'_{n+1}(x_k))^2} \tag{19}$$

定义Gauss-Laguerre求积公式函数

```matlab
function result = GaussLaguerreIntegralFormula(fun, n)

    % 名称：Gauss-Laguerre求积公式
    % 输入：
    %       fun:    积分函数
    %       n:      积分节点数
    % 输出：
    %       result：积分值

    %% 函数
    syms x
    L = exp(x) * diff(x^n * exp(-x), x, n); % Laguerre多项式
    root = solve(L);                        % Laguerre多项式的根
    DL = matlabFunction(diff(L, x));
    result = 0;
    for k = 1: n
        result = result + (factorial(n))^2 / root(k) / (DL(root(k)))^2 * fun(root(k));
    end

end

```

主函数

```matlab
clear; clc

% 定义函数
fun = @(x) sin(x / 10) / 10;
% 计算积分值
int3 = GaussLaguerreIntegralFormula(fun, 3);
int4 = GaussLaguerreIntegralFormula(fun, 4);
% 输出结果
fprintf('三点Gauss-Lagurre公式积分值为：%.10f\n', int3)
fprintf('四点Gauss-Lagurre公式积分值为：%.10f\n', int4)

```

输出结果

```
三点Gauss-Lagurre公式积分值为：0.0099009918
四点Gauss-Lagurre公式积分值为：0.0099009901
```

第四题

设$f(x)=\ln x$，分别取$h=10^{-n}$，其中$n=1,2,3,4$，用以下三个公式计算$f'(0.7)$的近似值。

$$f'(x)=\frac{f(x+h)-f(x)}{h} \tag{20}$$

$$f'(x)=\frac{f(x+h)-f(x-h)}{2h} \tag{21}$$

$$f'(x)=\frac{f(x-2h)-8f(x-h)+8f(x+h)-f(x+2h)}{12h} \tag{22}$$

列表比较三个公式的计算误差，从误差可以得出什么结论?

解：主函数

```matlab
clear; clc

% 定义函数 f(x) = ln(x)
f = @(x) log(x);

% 待计算的点
x = 0.7;

% 求导数的准确值
exactDerivative = 1 / x;

% 不同的步长
H = transpose(10 .^ (-1: -1: -4));

% 初始化误差矩阵
errors = zeros(numel(H), 3);

% 计算误差
for k = 1: numel(H)
    h = H(k);

    % 使用第一个公式计算近似值
    derivative1 = (f(x + h) - f(x)) / h;
    errors(k, 1) = abs(exactDerivative - derivative1);

    % 使用第二个公式计算近似值
    derivative2 = (f(x + h) - f(x - h)) / (2 * h);
    errors(k, 2) = abs(exactDerivative - derivative2);

    % 使用第三个公式计算近似值
    derivative3 = (f(x - 2 * h) - 8 * f(x - h) + 8 * f(x + h) - f(x + 2 * h)) / (12 * h);
    errors(k, 3) = abs(exactDerivative - derivative3);
end

% 创建表格
variable_names = {'步长', '公式1', '公式2', '公式3'};
T = table(H, errors(:, 1), errors(:, 2), errors(:, 3), 'VariableNames', variable_names);
% 显示表格
format short e
disp(T);

```

输出结果

```
         步长          公式1          公式2          公式3
    __________    __________    __________    __________

    1.0000e-01    9.3258e-02    9.8389e-03    5.1317e-04
    1.0000e-02    1.0108e-02    9.7194e-05    4.7634e-08
    1.0000e-03    1.0194e-03    9.7182e-07    4.7569e-12
    1.0000e-04    1.0203e-04    9.7180e-09    3.1619e-13
```

横向比较：同一步长，公式1误差>公式2误差>公式3误差。

纵向比较：同一公式，步长越小误差越小。

第五题

对于积分

$$\int_0^1 \frac{4}{1+x^2}\mathrm{d}x = \pi \tag{23}$$

，取$h=0.1$和$h=0.2$，分别用复合两点Gauss-Legendre公式和复合三点Gauss-Legendre公式计算π的近似值。

解：定义Gauss-Legendre求积公式函数

```
function result = GaussLegendreIntegralFormula(fun, n, a, b)

    % 名称：Gauss-Legendre求积公式
    % 输入：
    %       fun:     积分函数
    %       n:       积分节点数
    %       a:       积分左边界
    %       b:       积分右边界
    % 输出：
    %       result：积分值

    %% 函数

    % 求解Legendre多项式的零点
    syms x
    L = diff((x^2-1)^n, x, n) / (2^n * factorial(n)); % Legendre多项式
    root = solve(L);                                  % Legendre多项式的根

    % 求解权重
    A = zeros(2 * n, n);
    B = zeros(2 * n, 1);
    for k = 0: 2 * n - 1
        A(k + 1, :) = transpose(root .^ k);
        B(k + 1) = int(x .^ k, -1, 1);
    end
    w = A \ B;

    % 求解积分值
    f = @(x) fun((b - a) / 2 .* x + (b + a) / 2);
    result = (b - a) / 2 * sum(w .* f(root));

```

```
end

```

定义复合Gauss-Legendre求积公式函数

```
function result = CompoundGaussLegendreIntegralFormula(fun, n, k, a, b)

    % 名称：复合Gauss-Legendre求积公式
    % 输入：
    %       fun:     积分函数
    %       n:       积分区间数
    %       k:       区间积分节点数
    %       a:       积分左边界
    %       b:       积分右边界
    % 输出：
    %       result: 积分值

    %% 函数
    result = 0;
    x = @(i) a + (b - a) / n * i;
    for i = 1: n
        result = result + GaussLegendreIntegralFormula(fun, k, x(i - 1), x(i));
    end

end

```

主函数

```
clear; clc

% 定义函数
fun = @(x) 4 ./ (1 + x .^ 2);

% 计算积分值
a = 0;
b = 1;
h = [0.1; 0.2];
int12 = CompoundGaussLegendreIntegralFormula(fun, (b - a) / h(1), 2, a, b);
int13 = CompoundGaussLegendreIntegralFormula(fun, (b - a) / h(1), 3, a, b);
int22 = CompoundGaussLegendreIntegralFormula(fun, (b - a) / h(2), 2, a, b);
int23 = CompoundGaussLegendreIntegralFormula(fun, (b - a) / h(2), 3, a, b);
int = [int12, int13; int22, int23];

% 创建表格
variable_names = {'步长', '两点', '三点'};
precision = 15;  % 设置精度
T = table(vpa(h, 2), vpa(int(:, 1), precision), vpa(int(:, 2), precision), 'VariableNames', variable_names);
% 显示表格
disp(T);

```

输出结果

```
步长          两点                三点
____    ________________    ________________

0.1     3.14159265403069    3.14159265356003
0.2     3.14159268178543    3.14159265168714
```