E06c 编程作业解答

姓名：任云玮　　学号：516030910586

问题　用不同数值方法计算积分

$$\int_0^1 \sqrt{x} \ln x \mathrm{d}x = -\frac{4}{9}.$$

1　复化求积方法

1.1　图像及作图相关代码

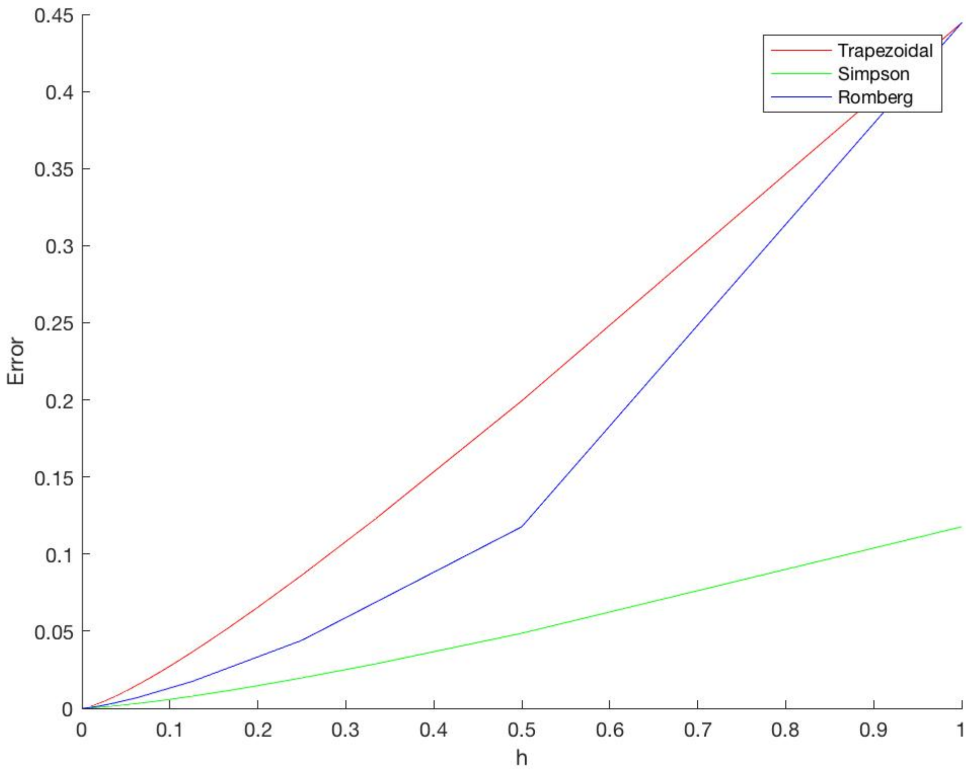

图 1: 步长与误差图

```
foo = @(x) Foo(x);

N = 1:1:100;
```

```
H = 1 ./ N;
errSim = H; errTra = H;

len = length(N);
for i = 1:100
    errSim(i) = abs(-4/9 - Simpson(foo, 0, 1, N(i)));
    errTra(i) = abs(-4/9 - Trapezoidal(foo, 0, 1, N(i)));
end

figure
hold on
xlabel('h');
ylabel('Error');
plot(H, errTra, 'r-');
plot(H, errSim, 'g-');

N = 1:1:20;
H = 1 ./ 2.^(N-1);
errRom = H;
for i = 1:20
    errRom(i) = abs(-4/9 - Romberg(foo, 0, 1, N(i)));
end

plot(H, errRom, 'b-');
legend('Trapezoidal', 'Simpson', 'Romberg');
hold off

function y = Foo(x)
    y = sqrt(x) .* log(x);
    y(x == 0) = 0;
end
```

1.2 编写复化梯形公式的函数文件，命名为 Trapezoidal.m，画出最大模误差与步长 h 的函数图像

```
function result = Trapezoidal(f, a, b, n)
    h = (b - a) / n;
    X = linspace(a, b, n + 1);
    Y = f(X);
    Y(1) = Y(1) / 2;
    Y(end) = Y(end) / 2;
    result = h * sum(Y);
end
```

1.3 编写复化 Simpson 公式的函数文件，命名为 Simpson.m，画出最大模误差与步长 h 的函数图像

```
function result = Simpson(f, a, b, n)
    h = (b - a) ./ n;
    X1 = linspace(a, b, n + 1);
    X2 = X1(1:end-1) + 0.5 * h;
```

```
    Y1 = f(X1);
    Y1(1) = Y1(1) / 2;
    Y1(end) = Y1(end) / 2;
    Y2 = f(X2);
    result = h / 6 * (2 * sum(Y1) + 4 * sum(Y2));
end
```

1.4 比较两个公式的精度，回答问题：是否存在一个最小的 h，使得精度不能再被改善?

复合梯形公式的误差为 $O(h^2)$，复合 Simpson 公式的误差为 $O(h^4)$. 从图中可知当 $h < 1$ 时，复合 Simpson 误差始终比复合梯形公式误差要小. 不存在一个最小的 h，使得精度不能再提高.

2 Romberg 求积方法

2.1 编写 Romberg 公式的函数文件，命名为 Romberg.m，画出最大模误差与步长 h 的函数图像

```
function result = Romberg(f, a, b, n, eps)
    if nargin <= 4
        eps = 0;
    end
    n = n - 1;
    h = b - a;
    R = ones(n+1, n+1) * NaN;
    R(1, 1) = h / 2 * (f(a) + f(b));
    for i = 2:n+1
        h = h / 2;
        R(i, 1) = 0.5 * R(i-1, 1) + h * sum(f(a+h:2*h:b-h));
        for j = 2:i+1
            R(i, j) = (4^(j-1) * R(i, j-1) - R(i-1, j-1)) / (4^(j-1) - 1);
        end
        result = R(i, i);
        if abs(R(i, i) - R(i-1, i-1)) < eps
            return
        end
    end
    result = R(n+1, n+1);
end
```

3 自适应 Simpson 求积方法

3.1 说明如何处理二分过程中树状图扩展的问题

points 表示当前的各个区间的端点. 调用 separateInterval(l, r) 来递归划分区间 $[l, r]$，如果在该区间上满足停机条件，则直接返回. 否则记 $m = (l + r)/2$，递归划分区间 $[l, m]$ 和 $[m, r]$. 同时把新产生的点 m 加入 points 中. 最后将 points 升序排序即可.

3.2 编写自适应的程序，停止准则为精度达到 10^{-4}，程序命名为 AdaptSimpson.m

```matlab
foo = @(x) Foo(x);
[result, points] = adaptSimpson(foo, 0, 1, 10^(-4));
disp(result);
disp(points);

function [result, points] = adaptSimpson(f, a, b, eps)
    points = [a, b];
    function [] = separateInterval(l, r)
        if abs(Simpson(f, l, r, 1) - Simpson(f, l, r, 2)) / 15 ...
                > (r - l) / (b - a) * eps
            m = (l + r) / 2;
            points(end + 1) = m;
            separateInterval(l, m);
            separateInterval(m, r);
        end
    end

    separateInterval(a, b);
    points = sort(points);
    len = length(points);
    result = 0;
    for i = 1:len-1
        result = result + Romberg(f, points(i), points(i+1), 10, ...
            (points(i+1) - points(i)) / (b - a) * eps);
    end
end

function y = Foo(x)
    y = sqrt(x) .* log(x);
    y(x == 0) = 0;
end
```

对应各层划分出的区间为

1. $[0, 1]$
2. $[0, 0.5]$，$[0, 5, 1]$
3. $[0, 0.25]$，$[0.25, 0.5]$
4. $[0, 0.125]$，$[0.125, 0.25]$

5. $[0, 0.0625]$, $[0.0625, 0.125]$
6. $[0, 0.0312]$, $0.0312, 0.0625]$
7. $[0, 0.0156]$, $[0.0156, 0.0312]$
8. $[0, 0.00781]$, $[0.00781, 0.0156]$
9. $[0, 0.00391]$, $[0.00391, 0.00781]$
10. $[0, 0.00195]$, $[0.00195, 0.00391]$
11. $[0, 0.000977]$, $[0.000977, 0.00195]$
12. $[0, 0.000489]$, $[0.000489, 0.000977]$
13. $[0, 0.000244]$, $[0.000244, 0.000489]$
14. $[0, 0.000122]$, $[0.000122, 0.000244]$
15. $[0, 0.0000610]$, $[0.0000610, 0.000122]$
16. $[0, 0.0000305]$, $[0.0000305, 0.0000610]$