APPM 5600 - Homework 12

Eappen Nelluvelil

November 19, 2021

1. (a) Write a code to approximate $\int_{-5}^{5} \frac{1}{1+s^2} \, \mathrm{d}s$ using a composite trapezoidal rule. To do this, partition the interval $[-5, 5]$ into equally spaced points $t_0, t_1, \ldots, t_n$.

   Write another code to approximate $\int_{-5}^{5} \frac{1}{1+s^2} \, \mathrm{d}s$ using a composite Simpson's rule. To do this, partition the interval $[-5, 5]$ into equally spaced points $t_0, t_1, \ldots, t_n$, where $n = 2k$ is even. The even indexed points should be the endpoints of your subintervals.

   You may combine the two into one code that selects the desired method if you wish.

   Turn in a listing of your code(s).

   See attached code.

   (b) Use the error estimates derived in class to choose n so that

$$\left| \int_{-5}^{5} \frac{1}{1+s^2} \, \mathrm{d}s - T_n \right| < 10^{-4}$$

   and

$$\left| \int_{-5}^{5} \frac{1}{1+s^2} \, \mathrm{d}s - S_n \right| < 10^{-4},$$

   where T_n is the result of the composite trapezoidal rule and where S_n is the result of the composite Simpson's rule. Be sure to explain your reasoning for choosing n in both cases (these n values will be different in the two cases).

   Recall that the error for the composite trapezoidal rule is given by

$$E_n(f) = -\frac{1}{12} h^2 (b-a) f''(\nu_n),$$

   where $f \in C^2([a, b])$, $h = \frac{(b-a)}{n}$, and ν_n is a value in (a, b).
   We see that

$$\begin{aligned}
|E_n(f)| &\leq \max_{\nu_n \in (-5,5)} \left| -\frac{1}{12} h^2 (b-a) f''(\nu_n) \right| \\
&= \frac{1}{12} h^2 (b-a) \max_{\nu_n \in (-5,5)} |f''(\nu_n)| \\
&= \frac{1}{6} h^2 (b-a)
\end{aligned}$$

   We can rearrange the above to obtain the n value required to have the error associated with the composite trapezoidal rule be less than 10^{-4}:

$$\begin{aligned}
\frac{1}{6} \frac{(b-a)^3}{n^2} < 10^{-4} &\implies \frac{10^7}{6} < n^2 \\
&\implies \sqrt{\frac{10^7}{6}} < n,
\end{aligned}$$

which means we must have $n \geq 1291$ to get the absolute error below 10^{-4}.

Also recall that the error for the composite Simpson's rule is given by

$$E_n(f) = -\frac{1}{180} h^4 (b-a) f''''(\nu_n),$$

where $f \in C^4([a,b])$, $h = \frac{(b-a)}{n}$, and ν_n is a value in (a,b). We see that

$$\begin{aligned} |E_n(f)| &\leq \max_{\nu_n \in (-5,5)} \left| -\frac{1}{180} h^4 (b-a) f''''(\nu_n) \right| \\ &= \frac{1}{180} h^4 (b-a) \max_{\nu_n \in (-5,5)} |f''''(\nu_n)| \\ &= \frac{2}{15} \frac{(b-a)^5}{n^4}. \end{aligned}$$

We can rearrange the above to obtain the n value required to have the error associated with the composite Simpson's rule be less than 10^{-4}:

$$\begin{aligned} \frac{2}{15} \frac{(b-a)^5}{n^4} \leq 10^{-4} &\implies \frac{20000}{15} 10^5 \leq n^4 \\ &\implies \sqrt[4]{\frac{20000}{15} 10^5} \leq n, \end{aligned}$$

which means we must have $n \geq 108$ to get the absolute error below 10^{-4}.

(c) Run your code with the predicted values of n and compare your computed values S_n and T_n with that of the MATLAB function `quad` on the same problem.

Run `quad` twice, once with the default tolerance of 10^{-6} and another time with the set tolerance of 10^{-4}. Report the number of function evaluations required in both cases and compare to the number of function values your codes (both S_n and T_n) required to meet the tolerance.

Turn in your codes and the results of this test.

When `quad` is run with a tolerance of 10^{-6}, it requires 81 function evaluations. When `quad` is run with a tolerance of 10^{-4}, it requires 41 function evaluations.

We see that with adaptive quadrature, we do not require as many function evaluations as with the composite trapezoidal rule (1292 function evaluations) or the composite Simpson's rule (109 function evaluations) to get the absolute error below the specified tolerances.

2. Apply the composite midpoint rule, composite trapezoidal rule, and composite Simpson's rule to approximate the integral

$$-4 \int_0^1 x \ln(x) \, \mathrm{d}x = 1.$$

Use $n = 2, 4, 8, 16, \ldots, 512$. Plot the absolute value of the error versus the stepsize h on a single log-log plot. Discuss the relationship of your results with the error formulas for these quadratures.

Recall that the corresponding error estimates for the rules are given by

$$\begin{aligned} |E_n(f)| &\leq \frac{1}{12} (b-a) h^2 \|f''\|_\infty \quad \text{(composite trapezoidal rule)} \\ |E_n(f)| &\leq \frac{1}{24} (b-a) h^2 \|f''\|_\infty \quad \text{(composite midpoint rule)} \\ |E_n(f)| &\leq \frac{1}{180} (b-a) h^4 \left\| f^{(4)} \right\|_\infty \quad \text{(composite Simpson's rule).} \end{aligned}$$

From the above error estimates, we see that the composite trapezoidal rule should perform the worst theoretically, the composite midpoint rule should perform the next best theoretically, and the composite Simpson's rule should perform the best.

In the `loglog` plot, we see that the composite Simpson's rule performs the best, the composite midpoint rule performs the next best, and the composite trapezoidal rules performs the worst, as expected.

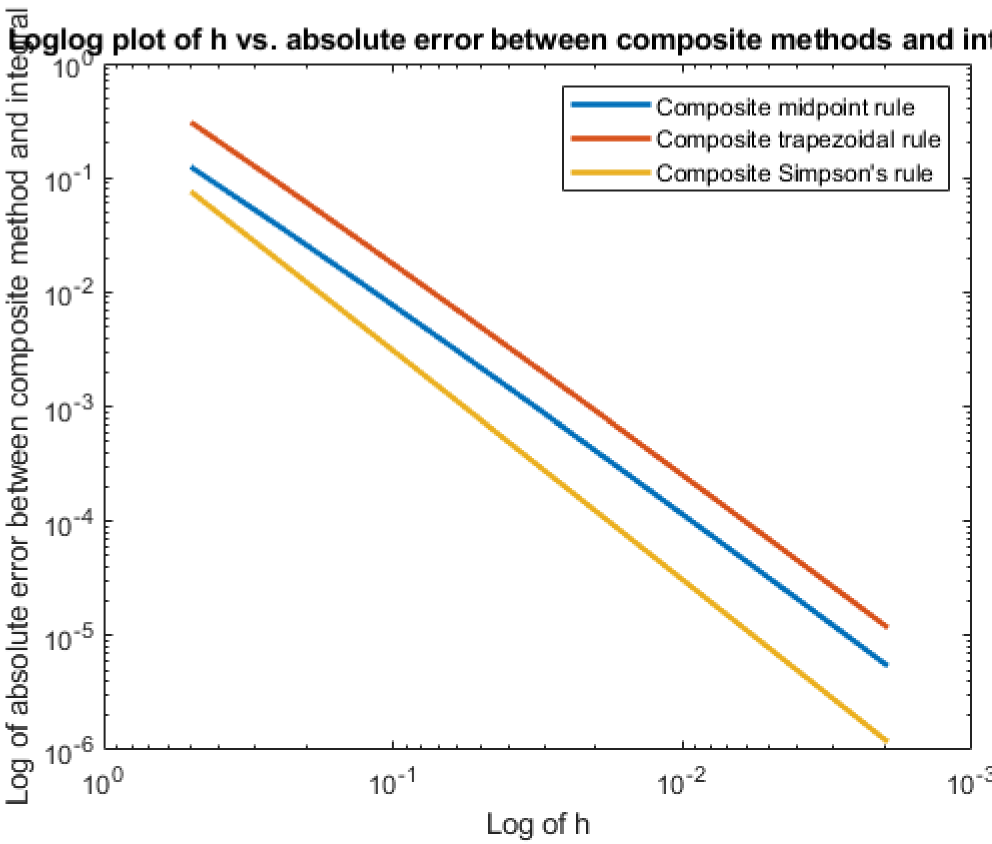
```
clc;
clear;

a = -5;
b = 5;
k = 25;
n = 2*k;
f = @(x) 1./(1+x.^(2));


% Problem 1c
tol1 = 1e-6;
tol2 = 1e-4;

n_trap = 1291;
n_simp = 108;

val_comp_trap = comp_trap(a, b, f, n_trap);
val_comp_simp = comp_simp(a, b, f, n_simp);

% Run quad with a tolerance of 1e-6
[val_integral1, fcnt1]  = quad(f, a, b, tol1);
fprintf("Absolute error between integral (tol = %e) and composite trapezoidal
 rule (n = %d): %e\n", ...
        tol1, n_trap, abs(val_integral1-val_comp_trap));
fprintf("Absolute error between integral (tol = %e) and composite Simpson's
 rule (n = %d): %e\n", ...
        tol1, n_simp, abs(val_integral1-val_comp_simp));
fprintf("Number of function evaluations needed for integral (tol = %e): %d
\n", ...
        tol1, fcnt1)

% Run quad with a tolerance of 1e-4
[val_integral2, fcnt2] = quad(f, a, b, tol2);
fprintf("\n");
fprintf("Absolute error between integral (tol = %e) and composite trapezoidal
 rule (n = %d): %e\n", ...
        tol2, n_trap, abs(val_integral2-val_comp_trap));
fprintf("Absolute error between integral (tol = %e) and composite Simpson's
 rule (n = %d): %e\n", ...
        tol2, n_simp, abs(val_integral2-val_comp_simp));
fprintf("Number of function evaluations needed for integral (tol = %e): %d
\n", ...
        tol2, fcnt2)
```

Problem 2

```
clear;
g = @(x) x.*log(x);
a_g = 1e-16;
b_g = 1;
```

```
n_vals = 2.^(1:9)';
h = (b_g-a_g)./(n_vals);

% Actual value of integral
val_integral_g = 1;

% Arrays to store the absolute value of the errors
abs_errs = zeros(length(n_vals), 3);

for i = 1:length(n_vals)
    val_comp_mid_g = -4*comp_mid(a_g, b_g, g, n_vals(i));
    val_comp_trap_g = -4*comp_trap(a_g, b_g, g, n_vals(i));
    val_comp_simp_g = -4*comp_simp(a_g, b_g, g, n_vals(i));

    abs_errs(i, 1) = abs(val_comp_mid_g - val_integral_g);
    abs_errs(i, 2) = abs(val_comp_trap_g - val_integral_g);
    abs_errs(i, 3) = abs(val_comp_simp_g - val_integral_g);
end

figure;
loglog(h, abs_errs, "LineWidth", 2);
legend("Composite midpoint rule", "Composite trapezoidal rule", "Composite
 Simpson's rule");
xlabel("Log of h");
ylabel("Log of absolute error between composite method and integral");
title("Loglog plot of h vs. absolute error between composite methods and
 integral")
set(gca, "xdir", "reverse");

function val = comp_mid(a, b, f, n)
    nodes = linspace(a, b, n+1)';
    h = (b-a)/n;
    nodes = 0.5*(nodes(1:end-1) + nodes(2:end));
    val = h*sum(f(nodes));
end

function val = comp_trap(a, b, f, n)
    nodes = linspace(a, b, n+1)';
    h = (b-a)/n;
    f_nodes = f(nodes);
    f_nodes(1) = 0.5*f_nodes(1);
    f_nodes(end) = 0.5*f_nodes(end);
    val = h*sum(f_nodes);
end

function val = comp_simp(a, b, f, n)
    nodes = linspace(a, b, n+1)';
    h = (b-a)/n;
    f_nodes = f(nodes);
    f_nodes(2:2:n) = 4*f_nodes(2:2:n);
    f_nodes(3:2:n-1) = 2*f_nodes(3:2:n-1);
    val = (h/3)*sum(f_nodes);
end
```

```
Absolute error between integral (tol = 1.000000e-06) and composite trapezoidal
 rule (n = 1291): 1.077025e-07
Absolute error between integral (tol = 1.000000e-06) and composite Simpson's
 rule (n = 108): 2.504883e-07
Number of function evaluations needed for integral (tol = 1.000000e-06): 81

Absolute error between integral (tol = 1.000000e-04) and composite trapezoidal
 rule (n = 1291): 6.295931e-06
Absolute error between integral (tol = 1.000000e-04) and composite Simpson's
 rule (n = 108): 6.438717e-06
Number of function evaluations needed for integral (tol = 1.000000e-04): 41
```



Published with MATLAB® R2021b