CMPS251 - Assignment 8

Kinan Dak Al Bab

October 31, 2014

1 Trapezoidal Rule

1.1 Question

Write a Matlab function to implement the trapezoidal rule using n intervals
function result = trapezoidal(f, a, b, n)

Use your function to evaluate the following integral

$$\int_0^1 \frac{4}{1+x^2}$$

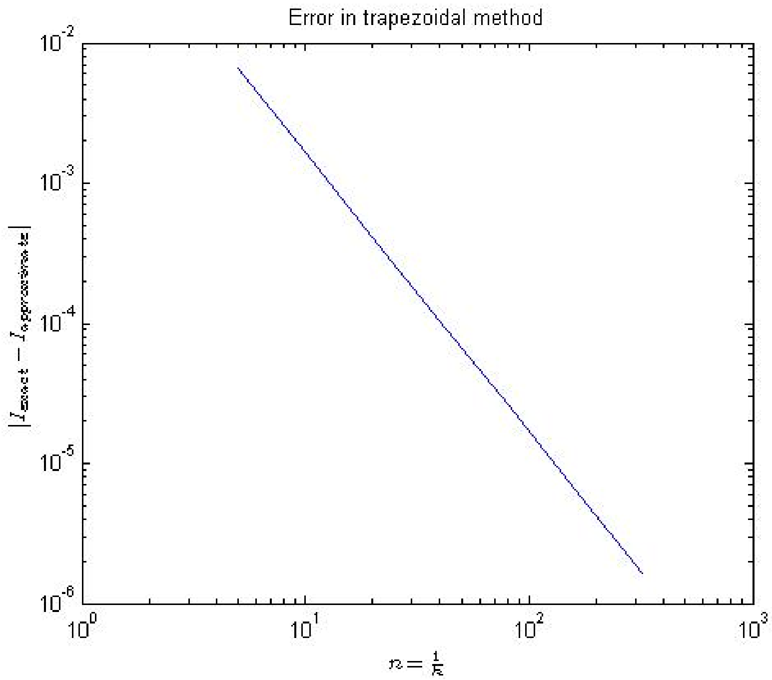
using $n = 5, 10, 20, 40, 80, 160,$ and 320 The exact answer is π. Plot the error vs n on a log-log plot. What is the slope of the resulting line? Was this expected?

1.2 Code

```matlab
%% Function
function result = trapezoidal(f, a, b, n)
h = (b - a) / n;

result = (f(a) + f(b)) / 2;

for i = 1:n-1
    result = result + f(a + h*i);
end

result = result * h;
end

%% Plot Script
f = @(x) 4/(1 + x^2);
N = [5 10 20 40 80 160 320];
I = zeros(1, length(N));
for i = 1:length(N)
    I(i) = trapezoidal(f, 0, 1, N(i));
end

E = abs(I - pi);
loglog(N , E);

title('Error in trapezoidal method');
xlabel('$n = \frac{1}{h}$', 'Interpreter', 'Latex');
ylabel('$| I_{exact} - I_{approximate}|$', 'Interpreter', 'Latex');
```

```
h = 1./N;

E = log(E);
N = log(N);
h = log(h);

slope = (E(length(E)) - E(1))/(N(length(E)) - N(1))
slope_against_h = (E(length(E)) - E(1))/(h(length(h)) - h(1))
```

1.3 Output




```
slope =

  -1.999995423263959
slope_against_h =

   1.999995423263959
```

1.4 Comment

At the first glance it looked like everything is wrong, I expected a line with slope 2 but got a line of slope -2, But then I noticed that the plot is against n not h, and we know that trapezoidal rule has an error in order of $O(h^2)$, Notice in our example that $h = \frac{b-a}{n} = \frac{1}{n}$, We can calculate the slope of Error versus h now (on loglog plot), and we get the expected slope of nearly 2.

2 A-priori error estimates

2.1 Question

Derive the following error estimate of the mid-point quadrature rule for computing $\int_a^b f(x)dx$

$$\frac{b-a}{24}||f''||_\infty h^2$$

(Note: these error estimates are called a-priori error estimates because they tell us about the error behavior before we obtain the approximations.)

2.2 Solution

Midpoint method is based on dividing the integral into smaller intervals of equal size. Let us First derive an estimate for the error in one interval:

We Know from taylor theorem:

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(c)}{2}(x - x_0)^2$$

But in the Case of midpoint rule, $x_0 = \frac{b'-a'}{2}$ Where b', a' are the end and start points of the interval studied.

$$f(x) = f(\frac{b'-a'}{2}) + f'(\frac{b'-a'}{2})(x - \frac{b'-a'}{2}) + \frac{f''(c)}{2}(x - \frac{b'-a'}{2})^2$$

$$f(x) - f(\frac{b'-a'}{2}) = f'(\frac{b'-a'}{2})(x - \frac{b'-a'}{2}) + \frac{f''(c)}{2}(x - \frac{b'-a'}{2})^2$$

$$Error = \int_{a'}^{b'} f'(\frac{b'-a'}{2})(x - \frac{b'-a'}{2})dx + \int_{a'}^{b'} \frac{f''(c)}{2}(x - \frac{b'-a'}{2})^2 dx$$

But:

$$\int_{a'}^{b'} (x - \frac{b'-a'}{2})dx = 0(x - \frac{b'-a'}{2} \text{ is an odd function (asymmetric)).}$$

$$\int_{a'}^{b'} (x - \frac{b'-a'}{2})^2 dx = \frac{1}{3}(x - \frac{b'-a'}{2})^3|_{b'}^{a'} = \frac{1}{3}(\frac{(b+a)^3}{8} - \frac{(3a-b)^3}{8}) = \frac{(2b'-2a')^3}{24}$$

So :

$$Error = \frac{f''(c)}{2}\frac{(2b'-2a')^3}{24} = \frac{(b'-a')^3}{24}f''(c)$$

Notice that $b' - a' = h$, Now let us take an upper bound for the error:

$$Error \leq \frac{h^3}{24}||f''||_\infty$$

Remember that this error is only for one interval, To get an estimate of the total maierror we multiply by the number of intervals $n = \frac{b-a}{h}$

$$Error \leq \frac{b-a}{24} ||f''||_\infty h^2$$

$\square$

3 A-posteriori error estimate in trapezoidal rule

3.1 Solution

We know that the error of the trapezoidal method can be written as:

$$E = K_1 h^2 + K_2 h^4 + \cdots + K_s h^{2s} + O(h^{2s+2})$$

Where the Ks are constants that depend on the derivatives of f. (The odd powers cancel out due to asymmetric behavior in the integration phase in the error derivation).

Now Let us start with the first row:

$$h_{1,1} = b - a(oneinterval)$$

$$h_{2,1} = \frac{h_{1,1}}{2}$$

$$R_{1,1} = \frac{h1}{2}(f(a) + f(b))$$

$$R_{2,1} = \frac{h1}{2}(f(a) + 2f(a + h_2) + f(b))$$

$$E_{1,1} = K_1 h_1^2 + K_2 h_1^4 + \dots$$

$$E_{2,1} = K_1(\frac{h_1}{2})^2 + K_2(\frac{h_1}{2})^4 + \dots$$

by taking a linear combination of $E_{1,1} - 4E_{2,1}$ We can get an error in order $O(h^4)$ (Extrapolation).

$$R_{2,2} = \frac{4R2,1 - R1,1}{3}$$

$$E_{2,2} = \frac{4E2,1 - E1,1}{3} = \frac{1}{4}K_2 h^4 + \frac{5}{16}K_3 h^6 + \dots$$

This can be generalized for a bigger number of extrapolation to form a triangular matrix, the derivation shown above is for such a matrix with 2 columns and 2 rows, with each extra column we remove another leading term of the error, but we need an extra row (to use for extrapolation). So for $s = 4$ we reach $O(h^{2s}) = O(h^8)$.

4 Romberg integration

4.1 Question

In this problem you are to use Romberg's method to compute the integral in problem 1 above.

- Generate the 1st column of the Romberg table using a trapezoidal rule with n = 2, 4, 8, 16, and 32.
- Generate the remaining columns one at a time to obtain the best estimate $R(s, s)$.
- Compute error estimates for the values in the Romberg table.

4.2 Code

```
f = @(x) 4/(1 + x^2);
a = 0;
b = 1;

s = 5;
R = zeros(s);
E = zeros(s);
for i = 1:s
    n = 2^i;
    R(i, 1) = trapezoidal(f, a, b, n);
    E(i, 1) = abs(pi - R(i, 1));
end

for k = 2:s
    for j = k:s
        R(j, k) = (4^(k-1) * R(j, k - 1) - R(j - 1, k - 1)) / ...
            (4^(k-1) - 1);
        E(j, k) = abs(pi - R(j, k));
    end
end

R
E
```

4.3 Output

```
R =

   3.100000000000000                   0                   0
                   0                   0
   3.131176470588235   3.141568627450981                   0
                   0                   0
   3.138988494491089   3.141592502458707   3.141594094125888
                   0                   0
   3.140941612041389   3.141592651224822   3.141592661142563   3.141592638396796
                   0
```

```
   3.141429893174975   3.141592653552837   3.141592653708038   3.141592653590029
  3.141592653649611


E =

   0.041592653589793                   0                   0
           0                   0
   0.010416183001558   0.000024026138812                   0
           0                   0
   0.002604159098704   0.000000151131086   0.000001440536095
           0                   0
   0.000651041548404   0.000000002364971   0.000000007552770   0.000000015192997
                   0
   0.000162760414818   0.000000000036957   0.000000000118245   0.000000000000236
  0.000000000059817
```

4.4 Comment

For Some Reason we are getting better estimates than that in R_5^5 at some entries. The Asymptotic behavior of Romberg method indicates that we should get better and better approximation ($O(h^2s)$). My Guess would that Inside the double nested loop, when we are trying to calculate the next entry using previous entries we are facing some floating point and rounding errors, because starting from the third column we start to get very small numbers. The example given in the book (Example 15.17) has the same anomaly.

5 Built-in: quad()

5.1 Question

Compute the integral above, to ten digits of accuracy, using the built-in Matlab function *quad*().

5.2 Code

```
f = @(x) 4./(1 + x.^2);
value = quad(f, 0, 1, 0.5 * 10^(-10))
value - pi
```

5.3 Output

```
value =

   3.141592653589911


ans =

    1.181277298201167e-013
```

5.4 Comment

As expected, the error was after the tenth digit.