Matt McDade

ANM Take-Home Final

1 & 2) See attached

3) **Trapezoid rule**

|  |  |
| --- | --- |
| CODE  f\_aa = @(x)1./(1+x.^2);  f\_bb = @(x)x.^3 - 3 \* x.^2 + 2;    trapRuleAns\_aa = trap(f\_aa, 0, 2)  trapRuleAns\_bb = trap(f\_bb, 1, 4)  function result = trap(f, a, b)  result = ((b - a) ./ 2) .\* (f(a) + f(b));  end | OUTPUT  trapRuleAns\_aa =  1.2000  trapRuleAns\_bb =  27 |

**Estimating Error ET**

|  |  |
| --- | --- |
| CODE  syms aa(x) bb(x)    aa(x) = 1./(1+x.^2);  bb(x) = x.^3 - 3 \* x.^2 + 2;    aa\_2 = diff(aa, 2);  bb\_2 = diff(bb, 2);    Et\_aa = -1/12 \* aa\_2(2) \* (2 - 0)^3  Et\_bb = -1/12 \* bb\_2(4) \* (4 - 1)^3 | OUTPUT  Et\_aa =    -44/375      Et\_bb =    -81/2 |

Actual Error:

aa) |1.107148717794090 - 1.2000| = **0.09285128220591 <** **0.11733333 =** 44/375

bb) |6.75 - 27| = **20.25 < 40.5** = 81/2

Both Actual errors are lower than the max estimated error ET, which is expected

**Composite Trapezoid Rule**

|  |  |
| --- | --- |
| CODE  f\_aa = @(x)1./(1+x.^2);  f\_bb = @(x)x.^3 - 3 \* x.^2 + 2;  intervals = 10;    for i = 1:5  ctrapRuleAns\_aa(i) = ctrap(f\_aa, 0, 2, intervals);  ctrapRuleAns\_bb(i) = ctrap(f\_bb, 1, 4, intervals);  intervals = intervals .\* 2;  end    disp(ctrapRuleAns\_aa)  disp(ctrapRuleAns\_bb)  function result = ctrap(f, a, b, n)  delta\_x = (b-a)/n;  x = [a+delta\_x:delta\_x:b-delta\_x];  result = 0;  for i = 1:n-1  result = result + f(x(i));  end  result = delta\_x \* (result + 0.5 \* (f(a)+f(b)));  end | OUTPUT  1.1066 1.1070 1.1071 1.1071 1.1071  6.9525 6.8006 6.7627 6.7532 6.7508 |

**Estimating Error ECT**

|  |  |
| --- | --- |
| CODE  syms aa(x) bb(x)    aa(x) = 1./(1+x.^2);  bb(x) = x.^3 - 3 \* x.^2 + 2;  aa\_2 = diff(aa, 2);  bb\_2 = diff(bb, 2);  intervals = 10;    for i=1:5  aa\_h = (2 - 0) ./ intervals;  bb\_h = (4 - 1) ./ intervals;  Ect\_aa(i) = -1/12 \* aa\_2(2) \* (2 - 0) \* aa\_h.^2;  Ect\_bb(i) = -1/12 \* bb\_2(4) \* (4 - 1) \* bb\_h.^2;  intervals = intervals \* 2;  end    disp(Ect\_aa)  disp(Ect\_bb) | OUTPUT  [ -11/9375, -11/37500, -11/150000, -11/600000, -11/2400000]    [ -81/200, -81/800, -81/3200, -81/12800, -81/51200] |

Actual Error (10 intervals):

aa) |1.107148717794090 – 1.1066|= **0.00054871779409 < 0.0011733333**  = 11 / 9375

bb) |6.75 – 6.9525| = **0.2025 < 0.405** = 81/200

The Actual error is less than the estimated error for every number of intervals. The calculations for all other number of intervals is similar to what is shown.

**Simpson’s Rule**

|  |  |
| --- | --- |
| CODE  f\_aa = @(x)1./(1+x.^2);  f\_bb = @(x)x.^3 - 3 \* x.^2 + 2;    simpRuleAns\_aa = simp(f\_aa, 0, 2)  simpRuleAns\_bb = simp(f\_bb, 1, 4)  function result = simp(f, a, b)  result = ((b - a) ./ 3) .\* (f(a) + f(b));  end | OUTPUT  simpRuleAns\_aa =  0.8000  simpRuleAns\_bb =  18 |

**Estimating Error ES**

|  |  |
| --- | --- |
| CODE  syms aa(x) bb(x)    aa(x) = 1./(1+x.^2);  bb(x) = x.^3 - 3 \* x.^2 + 2;    aa\_2 = diff(aa, 4);  bb\_2 = diff(bb, 4);    Es\_aa = -1/2880 \* aa\_2(0) \* (2 - 0)^5  Es\_bb = -1/2880 \* bb\_2(1) \* (4 - 1)^5 | OUTPUT  Es\_aa =    -4/15      Es\_bb =    0 |

Actual Error:

aa) |1.107148717794090 – 0.8000| = **0.30714871779409 <** **0.26666 =** 4/15

bb) |6.75 - 18| = **11.25 > 0**

Everything worked out nicely for equation aa, but bb became 0 when taking the 4th derivative, so calculating the actual error for this function seems to be meaningless, since you get 0.

**Composite Simpson’s Rule**

|  |  |
| --- | --- |
| CODE  f\_aa = @(x)1./(1+x.^2);  f\_bb = @(x)x.^3 - 3 \* x.^2 + 2;  intervals = 10;    for i = 1:5  csimpRuleAns\_aa(i) = csimp(f\_aa, 0, 2, intervals);  csimpRuleAns\_bb(i) = csimp(f\_bb, 1, 4, intervals);  intervals = intervals .\* 2;  end    disp(csimpRuleAns\_aa)  disp(csimpRuleAns\_bb)  function result = csimp(f,a,b,n)  delta\_x = (b-a)/n;  x = [a:delta\_x:b];  x4 = 0; x2 = 0;  for i=2:(((n+1)/2) - 1)  x2 = x2 + f(x(2\*i));  end  for i=2:((n+1)/2)  x4 = x4 + f(x(2\*i));  end  result = (delta\_x./3).\*(f(a)+ f(b) + 4.\*x4+ 2.\*x2);  end | OUTPUT  0.7721 0.9349 1.0205 1.0638 1.0855  6.3522 6.2228 6.3959 6.5493 6.6436 |

**Estimating Error ECS**

|  |  |
| --- | --- |
| CODE  syms aa(x) bb(x)    aa(x) = 1./(1+x.^2);  bb(x) = x.^3 - 3 \* x.^2 + 2;  aa\_2 = diff(aa, 4);  bb\_2 = diff(bb, 4);  intervals = 10;    for i=1:5  aa\_h = (2 - 0) ./ intervals;  bb\_h = (4 - 1) ./ intervals;  Est\_aa(i) = -1/180 \* aa\_2(0) \* (2 - 0) \* aa\_h.^4;  Est\_bb(i) = -1/180 \* bb\_2(1) \* (4 - 1) \* bb\_h.^4;  intervals = intervals \* 2;  end    disp(Est\_aa)  disp(Est\_bb) | OUTPUT  [ -4 / 9375, -1 / 37500,  -3689348814741911 / 2213609288845146193920, -3689348814741911 / 35417748621522339102720, -3689348814741911 / 566683977944357425643520 ]    [ 0, 0, 0, 0, 0] |

Actual Error (10 intervals):

aa) |1.107148717794090 – 0.7721 |= **0.33504871779409 > 0.000426666666** = 4/9375

bb) |6.75 – 6.3522| = **0.3978 > 0**

Similar to the Simpson’s Rule with only 1 interval, all estimated errors for bb turned out to be 0 since the 4th derivative of equation bb is 0. Interestingly though, the estimated error for aa with 10 intervals was much less than the actual error, which shouldn’t be the case. I suspect there is some kind of error in my error implementation, but I couldn’t find it.

**Table of Estimated & Actual Errors from Above**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Equation ->** | **aa** | | **bb** | |
| **Method** | **# Intervals** | **Estimated** | **Actual** | **Estimated** | **Actual** |
| **Trapezoid** | **1** | 0.11733333 | 0.09285128220591 | 40.5 | 20.25 |
| **10** | 0.0011733333 | 0.00054871779409 | 0.405 | 0.2025 |
| **20** | 0.0002933333 | 0.00014871779409 | 0.10125 | 0.0506 |
| **40** | 0.0000733333 | 0.00004871779409 | 0.0253125 | 0.0127 |
| **80** | 0.0000183333 | 0.00004871779409 | 0.006328125 | 0.0032 |
| **160** | 0.0000048333 | 0.00004871779409 | 0.00158203125 | 0.0008 |
| **Simpsons** | **1** | 0.2666666666 | 0.30714871779409 | 0 | 11.25 |
| **10** | 0.0004266666 | 0.33504871779409 | 0 | 0.3978 |
| **20** | 0.0000266666 | 0.17224871779409 | 0 | 0.5272 |
| **40** | 0.0000016666 | 0.08664871779409 | 0 | 0.3541 |
| **80** | 0.0000001041666 | 0.04334871779409 | 0 | 0.2007 |
| **160** | 0.00000000121646 | 0.02164871779409 | 0 | 0.1064 |