corrected Solutions to Assignment #7

1. I wrote a code that solves parts **(a)** and **(b)** of this problem, and also does problem **5**:

easyintegrals.m

```
% EASYINTEGRALS  Do some integrals for Assignment # 7.

% problem 1a
format long g
fa = @(x) cos(x).^2;
trap1a = (0.5/2) * (fa(0) + fa(0.5))                  % Trapezoid
simp1a = (0.25/3) * (fa(0) + 4 * fa(0.25) + fa(0.5))  % Simpson's
mid1a = (0.5) * fa(0.25)                             % Midpoint
exact1a = 0.460367746

% problem 1b; same order of rules
fb = @(x) x .* log(1+x);
trap1b = (0.5/2) * (fb(-0.5) + fb(0))
simp1b = (0.25/3) * (fb(-0.5) + 4 * fb(-0.25) + fb(0))
mid1b = (0.5) * fb(-0.25)

% problem 2 actual errors
trapactualerr2 = abs(trap1a - exact1a)
simpactualerr2 = abs(simp1a - exact1a)
midactualerr2  = abs(mid1a - exact1a)

% problem 5; composite Trapezoid and Simpson's, implemented using loops
n = 10;  h = 0.5 / n;
sum5T = fa(0);
sum5S = sum5T;
for j = 1:n-1
  x = 0 + j * h;
  sum5T = sum5T + 2 * fa(x);
  if abs(mod(j,2)) < 0.5,  sum5S = sum5S + 2 * fa(x);
  else,                    sum5S = sum5S + 4 * fa(x);   end
end
sum5T = sum5T + fa(0.5);
sum5S = sum5S + fa(0.5);
comptrap5 = (h/2) * sum5T
compsimp5 = (h/3) * sum5S
trapactualerr5 = abs(comptrap5 - exact1a)
simpactualerr5 = abs(compsimp5 - exact1a)
```

Running it gives:

```
>> easyintegrals
trap1a =    0.442537788233517
simp1a =    0.460443023059568
mid1a =     0.469395640472593
exact1a =   0.460367746
trap1b =   0.0866433975699932
simp1b =   0.0528546385609795
```

```
mid1b =    0.0359602590564726
trapactualerr2 = 0.0178299577664826
simpactualerr2 = 7.52770595679464e-05
midactualerr2 =  0.00902789447259311
comptrap5 =  0.460192410522105
compsimp5 =  0.460367863212223
trapactualerr5 = 0.000175335477895111
simpactualerr5 = 1.1721222253902e-07
```

Thus for the integral in part **(a)** I get $0.44254, 0.46044, 0.46939$, for Trapezoid, Simpson's, and Midpoint rules, respectively. For part **(b)** I get $0.086643, 0.052855, 0.035960$ respectively.

2. For part **(a)** the function is $f(x) = \cos^2 x = 0.5(1 + \cos 2x)$ so $f'(x) = -\sin 2x$, $f''(x) = -2\cos 2x$, $f'''(x) = +4\sin 2x$, and $f^{(4)}(x) = +8\cos 2x$. The error formula for Trapezoid rule is $E_T(h) = -(1/12)h^3 f''(\xi)$ and $h = 0.5$ so

$$|E_T(h)| = (0.5^3/12)\, 2|\cos(2\xi)| \le 1/48 = 0.0208 = 2.08 \times 10^{-2}.$$

For Simpson's $h = 0.25$ and $E_S(h) = -(1/90)h^5 f^{(4)}(\xi)$ so

$$|E_S(h)| = (0.25^5/90)\, 8|\cos(2\xi)| \le 8/(4^5 90) = 8.68 \times 10^{-5}.$$

For Midpoint $h = 0.5$ and $E_M(h) = +(1/3)h^3 f''(\xi)$ so

$$|E_M(h)| = (0.5^3/3)\, 2|\cos(2\xi)| \le 1/12 = 0.0833 = 8.33 \times 10^{-2}.$$

In each case I have simply used $|\cos\theta| \le 1$.

How do these compare to the actual errors? For Trapezoid we have (actual) $= 1.78 \times 10^{-2} = |E_T(h)| \le 2.08 \times 10^{-2}$. The estimate exceeds the actual error, as it must, but the estimate is very good. For Simpson's we have (actual) $= 7.53 \times 10^{-5} = |E_S(h)| \le 8.68 \times 10^{-5}$, again a good estimate. Finally for Midpoint we have (actual) $9.02 \times 10^{-3} = |E_M(h)| \le 8.33 \times 10^{-2}$, a rather poor estimate. This shows that sometimes the methods can be quite good because, like in the case of Midpoint here, the integrand happens to be evaluated at a "just right" location for estimating its area.

3. Here $h = 2$ for both Trapezoid and Midpoint rules. Thus, as noted in class, we know that $(2/2)(f(0) + f(2)) = 5$ and that $2(f(1)) = 4$. It follows that $f(0) + f(2) = 5$ and $f(1) = 2$. This allows us to evaluate Simpson's rule, in which $h = 1$:

$$\frac{1}{3}(f(0) + 4f(1) + f(2)) = \frac{1}{3}([f(0) + f(2)] + 4[f(1)]) = \frac{1}{3}([5] + 4[2]) = \frac{13}{3} = 4.33333.$$

4. We try the rule on successive powers of x:

$$2 = \int_{-1}^{1} x^0\,dx \overset{\checkmark}{=} \left(\frac{-\sqrt{3}}{3}\right)^0 + \left(\frac{\sqrt{3}}{3}\right)^0 = 2$$

$$0 = \int_{-1}^{1} x^1\,dx \overset{\checkmark}{=} \left(\frac{-\sqrt{3}}{3}\right)^1 + \left(\frac{\sqrt{3}}{3}\right)^1 = 0$$

$$\frac{2}{3} = \int_{-1}^{1} x^2\,dx \overset{\checkmark}{=} \left(\frac{-\sqrt{3}}{3}\right)^2 + \left(\frac{\sqrt{3}}{3}\right)^2 = \frac{3}{9} + \frac{3}{9} = \frac{2}{3}$$

$$0 = \int_{-1}^{1} x^3\,dx \overset{\checkmark}{=} \left(\frac{-\sqrt{3}}{3}\right)^3 + \left(\frac{\sqrt{3}}{3}\right)^3 = 0$$

$$\frac{2}{5} = \int_{-1}^{1} x^4\,dx \ne \left(\frac{-\sqrt{3}}{3}\right)^4 + \left(\frac{\sqrt{3}}{3}\right)^4 = \frac{9}{81} + \frac{9}{81} = \frac{2}{9}$$

Thus the rule has degree of precision *three*, which is very impressive because it evaluates the integrand the same number of times as the Trapezoid rule, namely twice, but the Trapezoid rule has degree of precision *one*!

So: *Is this rule really better than Simpson's? (*  *? I mean Simpson's rule!) Where did this rule come from? (Google it?) Can we do better with two evaluations? Are there rules which evaluate the integrand 3 times like Simpson's but have much higher precision? These are all good questions . . .*

5. Done in the code in problem 1. Note that the two methods evaluate the integrand the same number of times, but the Simpson's error 1.17×10^{-7} is about three orders of magnitude smaller than the Trapezoid error 1.75×10^{-4}.

6. CORRECTED SOLUTION. For composite Trapezoid rule the error formula says $E_{cT}(h) = -((b-a)/12)h^2 f''(\xi)$ so with $b - a = 0.5$ and $h = 0.5/n$ we have

$$|E_{cT}(h)| = \frac{0.5(0.5^2)}{12n^2}(2|\cos 2\xi|) \le \frac{1}{48n^2}.$$

We seek n so that the error is less than 10^{-13} so we solve the inequality $1/(48n^2) \le 10^{-13}$. This gives $n^2 \ge 2.0833 \times 10^{11}$ or $n \ge 4.56435 \times 10^5$. Thus $n = 456436$ would suffice. This means about half a million function evaluations.

For composite Simpson's rule, $E_{cS}(h) = -((b-a)/180)h^4 f^{(4)}(\xi)$. Thus with $h = 0.5/n$ (again!) we seek n so that

$$|E_{cS}(h)| = \frac{0.5(0.5^4)}{180n^4} 8|\cos 2\xi| \le \frac{1}{720n^4} \le 10^{-13}.$$

This gives $n \ge 343.29$. Thus $n = 344$ would suffice, noting n does need to be even for Simpson's rule. This represents far fewer function evaluations than for Trapezoid, by a factor of more than 1000.

We see that Simpson's is a very good idea for the right kind of integral.

A. I wrote this code, which produced Figure 1:

`snoopy.m`

```
% SNOOPY Plot the upper side of the noble beast.

x1 = [   1   2   5   6   7   8  10  13  17];
y1 = [ 3.0 3.7 3.9 4.2 5.7 6.6 7.1 6.7 4.5];
x2 = [  17  20  23  24  25  27 27.7];
y2 = [ 4.5 7.0 6.1 5.6 5.8 5.2  4.1];
x3 = [27.7  28  29  30];
y3 = [ 4.1 4.3 4.1 3.0];

xx1 = 1:.01:17;    % dense points for plotting
xx2 = 17:.01:27.7; % ditto
xx3 = 27.7:.01:30; % ditto

subplot(411)  % make plot aspect ratio close to original
set(0,'defaultlinelinewidth',1.5,'defaultlinemarkersize',4)
plot(xx1, ncspline(x1,y1,xx1), 'b', x1, y1, 'bo', ...
     xx2, ncspline(x2,y2,xx2), 'g', x2, y2, 'go', ...
     xx3, ncspline(x3,y3,xx3), 'r', x3, y3, 'ro' )
title('noble beast'),  grid on,  xlabel x,  ylabel('f(x)')
```

B. The full program I wrote is at `http://www.dms.uaf.edu/~bueler/hand.m`. I followed Moler's advice from *Numerical Computing with MATLAB*, on his exercise 3.4. I got the points by clicking the mouse. I ended up with 39 clicks, captured this way:

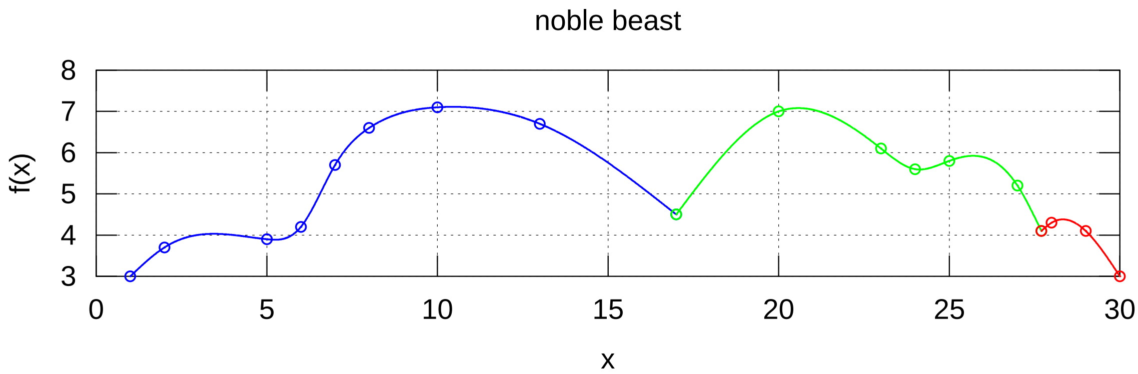

FIGURE 1. Plot from given data (circles) interpolated with three natural cubic splines.

```
>> figure('position',get(0,'screensize'))      % figure fills whole screen
>> axes('position',[0 0 1 1])                  % treat axes as  0<x<1,  0<y<1
>> [x,y]=ginput;                               % record mouse click location until
                                               %   enter is pressed
```

Then I used a "fake" t-axis with $1 \le t \le 39$. And ran this code:

```
t = 1:39;        % "fake" t-axis for parameterized curve is just the point index
tt = 1:.01:39;   % fill-in for smooth plot
xx = ncspline(t,x',tt);
yy = ncspline(t,y',tt);
plot(xx,yy,'g-',x,y,'bo')
axis off   % show no axes labels, tick marks, grid, ...
```

The result is in Figure 2.

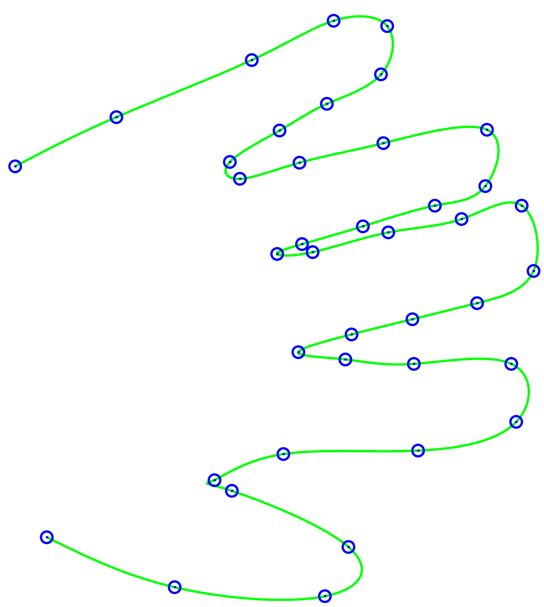

FIGURE 2. My hand, drawn smoothly from 39 marked points, using natural cubic splines.