Solutions to Assignment #9

Problems 7.2, exercise 21: *NOT GRADED!*

The first rule is Simpson's rule, so

$$a = \frac{1}{6}, \qquad b = \frac{4}{6}, \qquad c = \frac{1}{6}.$$

The second rule is not familiar, but since $x_0 = 1/4$, $x_1 = 1/2$, and $x_2 = 3/4$ we have

$$\alpha = \int_0^1 \ell_0(x)\,dx = \frac{2}{3}, \qquad \beta = \int_0^1 \ell_1(x)\,dx = -\frac{1}{3}, \qquad \gamma = \int_0^1 \ell_2(x)\,dx = \frac{2}{3}.$$

I guess I am fundamentally lazy because I did not do these integrals by hand. The integrands are quadratic polynomials, so Simpson's rule does these integrals exactly. So I wrote and ran a little code:

othersimp.m

```
% OTHERSIMP  Compute coefficients in second rule in 7.2 exercise #21.

% build the Lagrange polynomials as functions:
x = [0.25 0.5 0.75];
l0 = @(y) (y-x(2)) .* (y-x(3)) / ((x(1)-x(2)) * (x(1)-x(3)));
l1 = @(y) (y-x(1)) .* (y-x(3)) / ((x(2)-x(1)) * (x(2)-x(3)));
l2 = @(y) (y-x(1)) .* (y-x(2)) / ((x(3)-x(1)) * (x(3)-x(2)));

% plot them to check they are the right functions:
xf=0:0.001:1;
plot(x,ones(size(x)),'o',x,zeros(size(x)),'o'), hold on
plot(xf,l0(xf),xf,l1(xf),xf,l2(xf)), hold off, grid on
legend('l_0(x)','l_1(x)','l_2(x)')

% now apply Simpson's rule to *exactly* do the integrals which
%   determine the Newton-Cotes coefficients:
format rat
alpha = (1/6) * (l0(0) + 4*l0(0.5) + l0(1))
beta  = (1/6) * (l1(0) + 4*l1(0.5) + l1(1))
gamma = (1/6) * (l2(0) + 4*l2(0.5) + l2(1))
```

```
>> othersimp
alpha = 2/3
beta = -1/3
gamma = 2/3
```

Now, which rule is better?

It is appropriate to look at the integrals which appear in the error analysis, as done in class, namely the integrals of the known-part of the remainder term from polynomial interpolation,

$$\int_0^1 |x-0||x-\frac{1}{2}||x-1|\,dx, \qquad \int_0^1 |x-\frac{1}{4}||x-\frac{1}{2}||x-\frac{3}{4}|\,dx$$

But the result is ambiguous to me because we only know inequalities about these results. (For instance, if I tell you that all I know about X and Y are that $|X| < 10$ and $|Y| < 20$, you still cannot answer: Which is bigger, X or Y?)

And it turns out that both rules exactly integrate all cubic polynomials, and neither rule exactly integrates quartics. (I thought *this* was the difference ...)

So I think the answer the authors had in mind is this: Integration has this property

$$M(b-a) \le \int_a^b f(x)\,dx \le m(b-a),$$

where $M = \max f(x)$ and $m = \min f(x)$. Simpson's rule also has this property. But the second rule does not, *because* $\beta = -1/3$ *is negative*, even though $a+b+c=1$ and $\alpha+\beta+\gamma = 1$. In an informal sense, the second rule is not an average like integration and Simpson's rule are. Concretely, suppose we have a continuous function for which $M = 1$, $m = 0$, and $f(1/4) = 0$ and $f(1/2) = 1$ and $f(3/4) = 0$. Then the second rule wrongly says that the integral is $-1/3$, which is outside of the range of the function.

Problems 7.2, exercise 22: If this formula is true for all quadratics then it is true for the monomials $1, x, x^2$. That means, immediately, three equations in the three unknowns α, x_0, x_1:

$$\begin{aligned} 1 &= \int_0^1 1\,dx = \alpha\,[1+1] \\ \frac{1}{2} &= \int_0^1 x\,dx = \alpha\,[x_0 + x_1] \\ \frac{1}{3} &= \int_0^1 x^2\,dx = \alpha\left[x_0^2 + x_1^2\right]. \end{aligned}$$

These are not linear equations, so there is no particular guarantee we can solve them at all. But one tries ... and in this case it works out fine.

The first equation implies $\alpha = 1/2$. The second and third equations above simplify to

$$1 = x_0 + x_1 \qquad \text{and} \qquad \frac{2}{3} = x_0^2 + x_1^2.$$

The first of these can be solved for x_1, for example, and that result substituted into the second gives

$$\frac{2}{3} = x_0^2 + (1-x_0)^2 = 2x_0^2 - 2x_0 + 1$$

or $2x_0^2 - 2x_0 + \frac{1}{3} = 0$. This quadratic has solutions

$$x_0 = \frac{2 \pm \sqrt{4 - (8/3)}}{4} = \frac{1}{2} \pm \frac{1}{2\sqrt{3}}.$$

Which solution do we choose? *Both.* Note that the problem is symmetric in x_0 and x_1, so one of these must be x_0 and the other x_1. Fortunately they are both in the interval $[0,1]$. (*Being in the interval of integration is not completely obligatory; one can imagine numerical rules from polynomial* extrapolation*, but they are not widely used.*)

In summary, *yes* there is such an integration rule, and

$$\alpha = \frac{1}{2}, \qquad x_0 = 0.21132486540519, \qquad x_1 = 0.78867513459481$$

This turns out that we have found one of the simplest "Gaussian integration" rules. It turns out to be the best two-point numerical integration rule in lots of ways. Translated to the interval $[-1, 1]$, it is equation (8) in section 7.3, which we skipped.

Problems 7.4, exercise 6a: This is an application of the online program `romberg.m`, *if* you know how to alter that code the right way, or display its intermediate results. Note that the exact answer is

$$\int_1^3 \frac{dx}{x} = \ln 3 - \ln 1 = \ln 3$$

Here is my much-more-direct method using equations (4) and (5) on pages 503–504:

```
>> f = @(x) 1./x;
>> R00 = trap(f,1,3,1), R10 = trap(f,1,3,2), R20 = trap(f,1,3,4)
R00 =  1.33333333333333
R10 =  1.16666666666667
R20 =  1.11666666666667
>> R11 = R10 + (1/(4-1)) * (R10 - R00)
R11 =  1.11111111111111
>> R21 = R20 + (1/(4-1)) * (R20 - R10)
R21 =  1.10000000000000
>> R22 = R21 + (1/(4^2-1)) * (R21 - R11)
R22 =  1.09925925925926
>> log(3)
ans =  1.09861228866811
```

We see that Romberg's value $R(2, 2) =$ `R22` takes three not-so-good trapezoid rule results, and produces a pretty good estimate.

Problem 4 in the same section motivates the following comment. Note that `R11` and `R21` are actually Simpson's rule results:

```
>> (1/3) * (f(1) + 4*f(2) + f(3))
ans =  1.11111111111111
>> (1/6) * (f(1) + 4*f(1.5) + 2*f(2) + 4*f(2.5) + f(3))
ans =  1.10000000000000
```

Thus, just by building the second column of the Romberg "array", we are making identifiable progress over raw trapezoid rule. Filling in the rest of the array is going "beyond" Simpson's rule.

Problems 7.4, exercise 8: The points of this exercise are that $R(i, j)$ is computable with exactly the same number of function evaluates as is $R(i, 0)$, which is $2^i + 1$, and that $R(i, 0)$ is the trapezoid rule estimate *efficiently computed with no more than the minimal number of function evaluations.* Specifically, my answer to this question is

$$\text{(number of function evaluations of } f(x) \text{ to compute } R(i, j)) = 2^i + 1.$$

Problems 8.1, exercise 4: These solutions are just integration, and in both cases we can write

$$x(t) = x(0) + \int_0^t f(s)\, ds = \int_0^t f(s)\, ds$$

because $f(t, x) = f(t)$ does not depend on the unknown solution $x(t)$.

In this and **exercise 5** *you should check your answer. That is, substitute into the differential equation and check also that* $x(0) = 0$.

(a) $x(t) = t^4/4$

(b) Since I don't have my "arc" integrals well-memorized, here is the re-derivation using $s = \sin\theta$:

$$x(t) = \int_0^t \frac{ds}{\sqrt{1-s^2}} = \int_0^{\arcsin(t)} \frac{\cos\theta\, d\theta}{\sqrt{1-\sin^2\theta}} = \int_0^{\arcsin(t)} \frac{\cos\theta\, d\theta}{\cos\theta} = \int_0^{\arcsin(t)} d\theta = \arcsin(t)$$

Problems 8.1, exercise 5: The problem hints that you can think of this case, with $f(t,x) = f(x)$ not a function of t, by the manipulation

$$\frac{dx}{dt} = f(x) \quad \Longleftrightarrow \quad \frac{dt}{dx} = \frac{1}{f(x)}$$

and then you can again solve by integration with respect to x, because the roles of t,x are reversed relative to **exercise 4** previously. By contrast, most students use this technique,

$$\frac{dx}{dt} = f(x) \quad \Longleftrightarrow \quad \frac{dx}{f(x)} = dt \quad \Longleftrightarrow \quad \int \frac{dx}{f(x)} = \int dt,$$

so these problems look like this:

(a)

$$\int x^2\, dx = \int dt$$

$$\frac{1}{3}x^3 = t + C \qquad (\text{and } x(0) = 0 \implies C = 0)$$

$$x(t) = (3t)^{1/3}$$

(b) I did this one in class: $x(t) = \tan t$.

The point of **exercises 4 and 5** *is to get just good enough at solving ODEs so that we can generate exact solutions in enough cases to know that our numerical methods are doing the right thing, or are making expected-size errors. That process of checking against exact solutions is called* "verification".

Exercise 1: **(a)** The "exact" answer is known in this case just as well as the values of the exponential and the sine function are known. This integral can be computed from a very-well-understood built-in function just like those functions. I typed "`help erf`" to remind myself how the built-in error function is defined:

$$\operatorname{erf} x = \frac{2}{\sqrt{\pi}} \int_0^x e^{-s^2}\, ds.$$

Also I wrote an un-exciting Simpson's rule method which has the same calling-arguments as `trap`:

simp.m

```
function z = simp(f,a,b,n)
% SIMP  Approximate the integral
%    /b
%    |   f(x) dx
%    /a
% using composite Simpson's rule with n subintervals.
% The function f(x) must be vectorized and n must be even.
%
% Example:  >> simp(@(x) exp(-x),0,2,100)
%           >> exact = 1-exp(-2)

if mod(n,2)~=0, error('n must be even for Simpsons rule'), end
h = (b - a) / n;
x = a + (0:n) * h;
z = (h/3) * sum([1 repmat([4 2],1,(n/2)-1) 4 1] .* feval(f,x));
```

So now the results:

```
>> exact = (sqrt(pi)/2) * erf(2)
exact =  0.882081390762422
>> f = @(x) exp(-x.^2);
>> trap(f,0,2,32)
ans =  0.882057557801211
>> simp(f,0,2,32)
ans =  0.882081328646356
>> romberg(f,0,2)
ans =  0.882081390708990
>> relacctrap = abs(trap(f,0,2,32) - exact) / exact
relacctrap =  2.70190046630777e-05
>> relaccsimp = abs(simp(f,0,2,32) - exact) / exact
relaccsimp =  7.04198792989501e-08
>> relaccromb = abs(romberg(f,0,2) - exact) / exact
relaccromb =  6.05739435772925e-11
```

I suppose I would only conclude that all of these methods do a reasonable job, and that this problem benefits both from switching from linear to quadratic interpolation (i.e. from trapezoid to Simpson's) and from Richardson extrapolation (i.e. Romberg).

(b) This is no harder. Note that when we hand a function to a code we hand a "function handle" to the code, which explains the "`@sin`" argument:

```
>> exact = -cos(7) + cos(2)
exact = -1.17004909089045
>> relacctrap = abs(trap(@sin,2,7,32) - exact) / abs(exact)
relacctrap =  0.00203533353212976
>> relaccsimp = abs(simp(@sin,2,7,32) - exact) / abs(exact)
relaccsimp =  3.32101817011846e-06
>> relaccromb = abs(romberg(@sin,2,7) - exact) / abs(exact)
relaccromb =  4.10526169284464e-10
```

Here we see essentially the same pattern as in part **(a)**.