Solutions to Assignment #9

1. I wrote these:

gaussian2.m

```
function z = gaussian2(f,a,b)
% GAUSSIAN2  Use n=2 Gauss-Legendre rule to approximate integral.  Note
% that c_1 = c_2 = 1.
%
% Example:
%   >> gaussian2(@(x) sin(x),0,pi/4)
%   >> exact = 1 - sqrt(2)/2          % = 0.2928932188

sh = @(t) 0.5 * ((b-a) * t + a+b);  % does shift and scale: x = sh(t)

t1 = - sqrt(3)/3;   t2 = - t1;

z = 0.5 * (b-a) * ( feval(f,sh(t1)) + feval(f,sh(t2)) );
```

gaussian3.m

```
function z = gaussian3(f,a,b)
% GAUSSIAN3  Use n=3 Gauss-Legendre rule to approximate integral.  Uses
% x1,x2,x3 and c1,c2,c3 from textbook.
%
% Example:
%   >> gaussian3(@(x) sin(x),0,pi/4)
%   >> exact = 1 - sqrt(2)/2          % = 0.2928932188

sh = @(t) 0.5 * ((b-a) * t + a+b);  % does shift and scale: x = sh(t)

t1 = 0.7745966692;  t2 = 0.0;     t3 = - t1;
c1 = 5/9;           c2 = 8/9;     c3 = 5/9;

z = c1 * feval(f,sh(t1)) + c2 * feval(f,sh(t2)) + c3 * feval(f,sh(t3));
z = 0.5 * (b-a) * z;
```

Here is the integration-by-parts calculation for the exact value of the test integral:

$$\int_1^2 xe^{-x}\,dx = -xe^{-x}\Big]_1^2 + \int_1^2 e^{-x}\,dx = e^{-1} - 2e^{-2} - e^{-x}\Big]_1^2 = 2e^{-1} - 3e^{-2}.$$

And comparison to results of $n = 2, 3$ Gaussian quadrature:

```
>> exact = 2 * exp(-1) - 3 * exp(-2);
>> f = @(x) x .* exp(-x);
>> [gaussian2(f,1,2); gaussian3(f,1,2)]
ans =
      0.329884478631073
```

```
      0.329753536211964
>> err = abs(ans - exact)
err =
    0.000131445998026392
    5.03578917843139e-07
```

Thus the $n = 2$ rule makes error about 1.3×10^{-4} while the $n = 3$ rule makes error about 5.0×10^{-7}.

2. I did this problem by-hand using long polynomial division, yielding:

$$Q(x) = x^2 + x - 2/5, \qquad R(x) = -(2/5)x^2 - (31/25)x + 17.$$

Thus $P(x) = Q(x)P_3(x) + R(x)$. The degrees of Q and R are both 2, and this is expected because (degree $P(x)$) = (degree $Q(x)$) + (degree $P_3(x)$) and because (degree $R(x)$) < (degree $P_3(x)$).

But I also checked my by-hand computation this way:

```
>> P = [1 1 -1 -1 -1 17];
>> P3 = [1 0 -3/5 0];
>> [Q,R] = deconv(P,P3)
Q =
        1           1        -0.4
R =
     -0.4       -1.24          17
```

Can you figure out what "`deconv`*" does? Do "*`help conv`*" to start.*

3. First I generated a graph which actually showed the zeros; this required brief experimentation using the `axis` command to get a good view:

```
>> f = @(x) (1/63) * (63 * x.^5 - 70 * x.^3 + 15 * x);
>> x = -1:.001:1;  plot(x,f(x)),  axis([-1 1 -0.2 0.2]),  grid on
```

The result is shown in Figure 1.

Clearly $x = 0$ is a root. Symmetry is clear as well, and this is all explained by factoring:

$$P_5(x) = \frac{1}{63} x \left(63x^4 - 70x^2 + 15\right).$$

We see that if x is a root then so is $-x$ because the quartic factor is an even function. Thus we only need to find the *two* positive roots by Newton's method. It suffices to find the roots of the quartic factor $G(x) = 63x^4 - 70x^2 + 15$.

From the figure, the first guesses $p_0 = 0.6$ and $p_0 = 0.9$ should lead in a few steps to highly-accurate roots by Newton's. That is what happens, as follows:

```
>> format long g
>> G = @(x) 63 * x.^4 - 70 * x.^2 + 15;
>> dG = @(x) 252 * x.^3 - 140 * x;
>> p = 0.6,  for k=1:5, p = p - G(p)/dG(p), end
p =                  0.6
p =    0.531168831168831
```

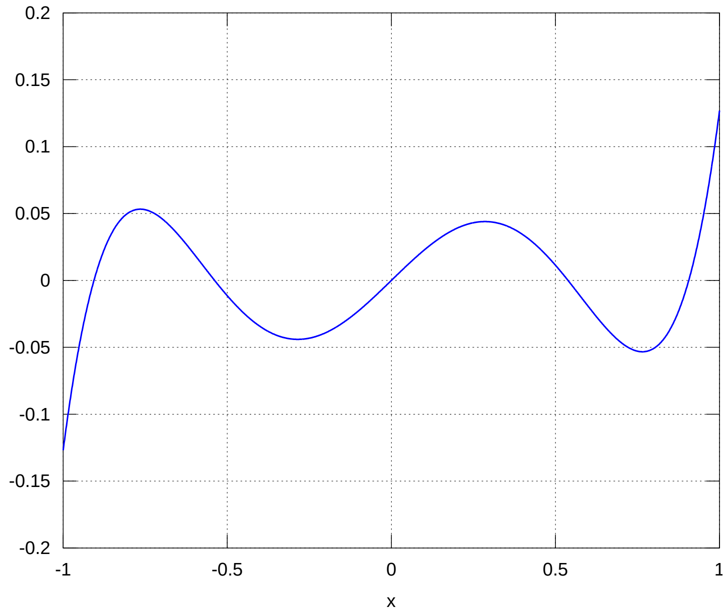

FIGURE 1. Plot of $P_5(x)$ to help make first guesses about roots.

```
p =    0.538414510614559
p =    0.538469306807752
p =    0.538469310105683
p =    0.538469310105683
>> p = 0.9,  for k=1:5, p = p - G(p)/dG(p), end
p =                  0.9
p =    0.906337076315242
p =    0.906179943829304
p =    0.906179845938702
p =    0.906179845938664
p =    0.906179845938664
```

A quick comparison to the textbook suggests Burden&Faires *also* believe these are roots of $P_5(x)$.

In fact it is quite possible to check this by hand. That is because $G(x) = 63x^4 - 70x^2 + 15$ is a *quadratic* function of the variable $z = x^2$. Thus

$$x^2 = \frac{70 \pm \sqrt{70^2 - 4(63)(15)}}{2(63)} = \{0.28994919792569, 0.821161913185421\}.$$

We take the square roots and get $x = \{0.538469310105683, 0.906179845938664\}$. So Newton's method works ... as expected.

4. The Gaussian elimination by hand is easy here. The plot in Figure 2 is from this code fragment; note the use of "`axis off`" and "`text`", which you may not have seen:

```
x1 = -5:.1:5;  x2a = -x1 / 2;  x2b = x1 + 3;
plot(x1,x2a,x1,x2b),  axis off,  hold on
plot([-5 5],[0 0],'k',[0 0],[-3 8],'k')
text(5.2,0,'x_1','fontsize',14), text(0,8.2,'x_2','fontsize',14)
```

```
text(-4,3,' intersect at (-2,1)','fontsize',14)
plot(-2,1,'ro','markersize',14),  hold off
print -dpdf linesfigure.pdf
```

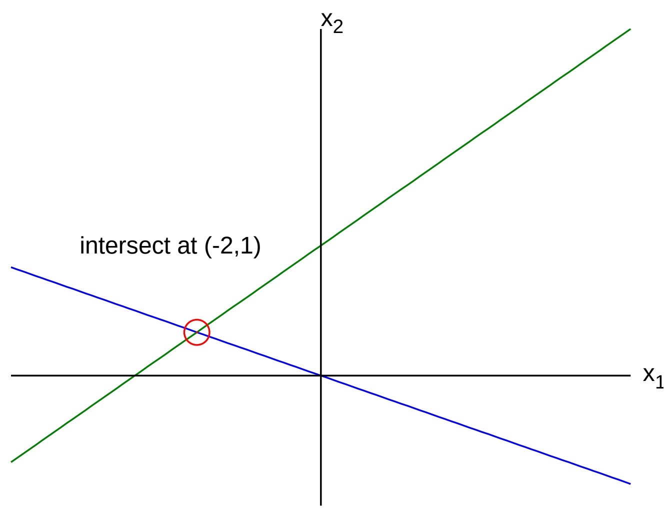

FIGURE 2. Intersecting lines in problem 4.

5. First I performed these row operations (Gaussian elimination):

$$E_2 \longleftarrow E_2 + E_1$$
$$E_3 \longleftarrow E_3 - \alpha E_1$$
$$E_3 \longleftarrow E_3 - (1+\alpha)E_2$$

The result was the system

$$\begin{aligned} x_1 \quad - x_2 \quad + \quad \alpha\, x_3 &= -2 \\ x_2 &= 1 \\ +(1-\alpha^2)\, x_3 &= 1+\alpha \end{aligned}$$

a. The system cannot be solved for a single (unique) solution if

$$1 - \alpha^2 = 0.$$

If $\alpha = 1$ then the last equation says "$0x_3 = 2$", which is impossible. The only value of α for which the system has no solutions is $\alpha = 1$.

b. If $\alpha = -1$ then the last equation says "$0x_3 = 0$", which very possible because it says nothing. When $\alpha = -1$ the system after Gaussian elimination is just these two equations:

$$\begin{aligned} x_1 \quad - x_2 \quad - x_3 &= -2 \\ x_2 &= 1 \end{aligned}$$

The set of all solutions (*which was not asked for!*) can be described by letting $x_3 = t$, to parameterize the solutions, and then:

$$\left\{ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \right\} = \left\{ \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} + t \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \;\middle|\; -\infty < t < \infty \right\}$$

In any case, the only value of α for which the system has ∞ly-many solutions is $\alpha = -1$.

c. Now we proceed to solve the system by backward substitution, in the (generic) cases in which $1 - \alpha^2 \neq 0$:

$$\begin{aligned} x_3 &= \frac{1+\alpha}{1-\alpha^2} = \frac{1}{1-\alpha}, \\ x_2 &= 1, \\ x_1 &= +x_2 - \alpha x_3 - 2 = -1 - \frac{\alpha}{1-\alpha} = -\frac{1}{1-\alpha}. \end{aligned}$$

By the way, I checked my work in MATLAB/OCTAVE *with one-liners which compared the original system, for a specific value of α, with the system after Gaussian elimination and with my final answer in part* **(c)**. *I did several values of α. One example looked like this:*

```
>> alpha=5;  A = [1 -1 alpha; -1 2 -alpha; alpha 1 1];  b = [-2 3 2]';  (A \ b)'
ans =
                  0.25                      1                -0.25
>> alpha=5;  U = [1 -1 alpha; 0 1 0; 0 0 (1-alpha^2)];  c = [-2 1 (1+alpha)]';  (U \ c)'
ans =
                  0.25                      1                -0.25
>> alpha=5;  x = [-1/(1-alpha) 1 1/(1-alpha)]
x =
                  0.25                      1                -0.25
```

6. Very easy, and easy to check.

7. **a.** I wrote the following working code:

forwardbueler.m

```
function x = forwardbueler(A,b)
% FORWARDBUELER  Solve lower triangular system by forward substitution.
% Check the size of the inputs, and does checks before division by zero.
% Also checks that the input matrix A is lower triangular.
%
% Example:
%   >> A = tril(randn(3,3))     % lower triangular 3x3 matrix
%   >> b = randn(3,1)
%   >> x = forwardbueler(A,b)
%   >> A * x - b                % should be nearly zero

[n,m] = size(A);       % for any size of matrix, but must be square (mxm)
if m ~= n,  error('A must be square (n x n)'), end
if max(max(abs(triu(A,1))))>0, error('A must be lower triangular'), end

[p,q] = size(b);
if q ~= 1, error('b must be a column vector'), end
if p ~= n, error('b must be have same number of rows as A'), end

x = zeros(size(b));    % create  x  as a column vector like  b

if A(1,1) == 0.0, error('zero in A(1,1) position'), end
x(1) = b(1) / A(1,1);
```

```
for i = 2:n
  if A(i,i) == 0.0, error('zero in A(%d,%d) position',i,i), end
  % next line does dot product of i-1 values:
  x(i) = (b(i) - A(i,1:i-1) * x(1:i-1)) / A(i,i);
end
```

In writing the code we see we must assume that *each diagonal entry* a_{ii} *is nonzero*, if we are to have a unique solution.

b. With the way I wrote it, here are the counts:

additions:	$(1/2)n^2 - (3/2)n + 1$
subtractions:	$n - 1$
multiplications:	$(1/2)n^2 - (1/2)n$
divisions:	n

I computed the number of multiplications done in the dot products by doing the sum

$$\sum_{i=2}^{n} i - 1 = \sum_{j=1}^{n-1} j = \frac{(n-1)n}{2} = (1/2)(n^2 - n).$$

The number of additions done in the dot products is one less per dot product:

$$\sum_{i=2}^{n} i - 2 = \sum_{j=1}^{n-2} j = \frac{(n-2)(n-1)}{2} = (1/2)(n^2 - 3n + 2).$$

The total number of arithmetic operations is $n^2 + 1$.

(*Other answers may be correct, because one may do subtraction instead of addition, but the number of divisions and multiplications should be these as stated.*)