# Math 273 Midterm I Solutions

**1.** The code fragment can be revised to

(1) Pre-allocate arrays

(2) Compute series terms recursively

(3) Vectorize

```
k = linspace(.05,.5,10)';
nTerms = 50;
term = ones(10,1);  % (1)
F    = term;        % (1)
for n = 1:nTerms
  term = term .* ( (2*n-1)/(2*n) )^2 .* k.^2; % (2), (3)
  F    = F + term;
end
F = pi/2*F;
```

**2.** Write the polynomial in the form

$$p(x) = c_1 + c_2(x+1) + c_3(x+1)x + c_4(x+1)x(x-1) = y.$$

Substitute $x = -1$, $y = 12$ to get $c_1 = 12$. Subtract $c_1$, divide by $(x+1)$:

$$c_2 + c_3 x + c_4 x(x-1) = \frac{y-12}{x+1}$$

Substitute $x = 0$, $y = 3$ to find $c_2 = -9$. Subtract $c_2$ and divide by $x$:

$$c_3 + c_4(x-1) = \frac{\frac{y-12}{x+1} + 9}{x}$$

Substitute $x = 1$, $y = 0$ to get $c_3 = 3$. Subtract $c_3$, divide by $x - 1$:

$$c_4 = \frac{\frac{\frac{y-12}{x+1}+9}{x} - 3}{x-1}$$

Evaluating at $x = 2$, $y = -9$ gives $c_4 = -2$. Hence the Newton form of the interpolating polynomial is

$$p(x) = 12 - 9(x+1) + 3(x+1)x - 2(x+1)x(x-1).$$

**3.** The fragment at the beginning of section 2.1.4 evaluates the power form of the polynomial:

```
n = length(a); zpower = 1; pval = a(1);
for i = 2:n                        % n=4, so 3 repetitions
   zpower = z*zpower               % 1 flop
   pval   = pval + a(i)*zpower % 2 flops
end
```

Three repetitions, times 3 flops per repetition, give a total of 9 flops.

The fragment at the bottom of the page evaluates the polynomial by nested multiplication:

```
pval = a(4);
pval = z*pval + a(3);
pval = z*pval + a(2);
pval = z*pval + a(1);
```

The last three lines require 2 flops each, for a total of 6 flops.

**4.** The given fragment approximates the derivative $f_1'(x)$ at 100 equally-spaced points from 1.00 to 1.99 by the forward difference method, and plots the result.

The given fragment invokes the function `f1` 200 times, with nearly half of those being duplicates. Assuming the function `f1` accepts vector arguments, we can make a vectorized version that is easy on the memory manager and only evaluates `f1` on a single vector of length 100, as follows:

```
x   = linspace(1,2,101)';
h   = 1/100;
y   = f1(x);
df1 = (y(2:101)-y(1:100))/h;
plot(x(1:100), df1)
```

If $|f''|$ and $|f'''|$ are not large, then $h = 0.01$ is far larger than the optimum $h$ for either the forward or the centered difference approximation. Therefore the error in both methods is mainly truncation error, and the centered difference approximation is more accurate. To use the centered approximation, replace the last two lines above by

```
df1 = (y(3:101)-y(1:99))/(2*h);
plot( x(2:100), df1)
```