Assignment #1
Due Friday, 29 January 2010.

MOP = MATLAB, OCTAVE, or Pylab

1. (There is nothing to turn in on this problem.) Find textbooks on calculus and on ordinary differential equations (ODEs). Generally, you will need these references to recall ideas needed for Math 615. Specifically, you should review these two topics:
   i) Taylor’s theorem with remainder formula, and
   ii) the solution of linear homogeneous constant-coefficient ODEs.

2. Calculate \( \sqrt{16.1} \) to five digits without any computing machinery except a pen(cil). Prove that your answer is correct to five digits, again without any computing machinery.

3. Assume \( f' \) and \( f'' \) are continuous. Derive the midpoint-rule-with-remainder formula
\[
\int_{-a}^{a} f(x) \, dx = 2af(0) + \frac{1}{3}a^3f''(\nu)
\]
for some (unknown) \(-a \leq \nu \leq a\). [Hint: \( f(x) = f(0) + f'(0)x + (1/2)f''(\xi)x^2 \) where \( \xi = \xi(x) \) is between 0 and \( x \).] Use two sentences to explain the meaning of this formula to the layperson.

How accurate is the midpoint rule on the integral \( \int_{-0.1}^{0.1} e^x \, dx \)?

4. Solve
\[
y'''' + 5y''' - 5y'' - y = 0, \quad y(1) = 0, \quad y'(1) = 0, \quad y''(1) = 2
\]
to find \( y(3) \). (Do not use a computer ODE solver. You will probably not be able to solve by-hand the polynomial equation which appears. You may, and probably should, find and use the root-finder built-in in MOP.)

5. (A write-your-first-MOP-program kind of problem.) Download and/or install and/or find MOP. Now work at the command line to compute a finite sum approximation to
\[
\sum_{n=0}^{\infty} \frac{1}{(4n+1)^2}.
\]
Compute at least a couple of finite sums with different numbers of terms; are you getting close to the infinite sum? Turn your command line work into a saved program (script or function).

6. Use MATLAB’s ode45 or OCTAVE’s lsode or Pylab’s odeint (with slightly different semantics; see examples on back page) to solve initial value problem (2) and find \( y(3) \). Equation (2) can and must be written as a first order system before using the built-in solver.

7. Using Euler’s method for approximately solving ODEs, write your own MOP program (either script or function) to solve initial value problem (2) to find \( y(3) \). Use a few step sizes, decreasing as needed, so that any reasonable observer would agree that you have four digit accuracy. (Even if the observer has not already done problem 4 or 6 above, that is.)

8. (There is nothing to turn in on this problem.) Read lightly the introduction of the textbook MORTON & MAYERs. Read seriously subsections 2.1, 2.2, 2.3, and 2.4 of MORTON & MAYERs.
The following example codes can all be found at
http://www.dms.uaf.edu/~bueler/Math615S10.htm

odeMatlab.m

% solve the ODE initial value problem
% \( y' = 3 \cdot y, \ y(0) = 0.2 \)
% on the interval \([0,1]\), and plot soln
f = @(t,y) 3 * y; % define function
t = 0:.01:1;
[t,y] = ode45(f,t,0.2);
y(end) % print \( y(1) \)
plot(t,y)

odeOctave.m

% solve the ODE initial value problem
% \( y' = 3 \cdot y, \ y(0) = 0.2 \)
% on the interval \([0,1]\), and plot soln
f = @(y,t) 3 * y; % define function
t = 0:.01:1;
y = lsode(f,0.2,t);
y(end) % print \( y(1) \)
plot(t,y)

odePylab.py

# solve the ODE initial value problem
# \( y' = 3 \cdot y, \ y(0) = 0.2 \)
# on the interval \([0,1]\), and plot soln
from pylab import linspace,plot,show
from scipy.integrate import odeint

def f(y,t): # define function
    return 3 * y
t = linspace(0,1,101)
y = odeint(f,0.2,t)
print y[-1] # print \( y(1) \)
plot(t,y) # print \( y(1) \)
show()