Computational Physics
Spring 2014, Shoemaker
Homework 1 (Due date February 7)

Email homework as an attachment to (`deirdre@gatech.edu` and `kjani3@gatech.edu`). Include a `README` file that lists the files included with a short description (e.g. programs, plots, etc).

1. (20 points) Modify the Matlab program we discussed in class `findroot.m` to find the roots of a continuous function to handle a system of 2 equations. The programs are posted on piazza. Test your program with the system of equations presented in Chapter_01.pdf, namely

   $$f(x,y) = e^x - 3y - 1$$

   and

   $$g(x,y) = x^2 + y^2 - 4\,.$$

   You can use the material available in Chapter_01.pdf to test your code. Your code should mimic `findroot.m` by using all the root finding methods.

2. (20 points) Write a Matlab function that solves the linear system of equations $\mathbf{A}\cdot\mathbf{x} = \mathbf{b}$ using the Gauss-Seidel method. Specifically, write the function

   `function x = gs_solve ( A, b, xold, TOL, Nmax )` where *inputs*: `A` coefficient matrix for linear system - must be a square matrix, `b` right-hand side vector for linear system, `xold` vector containing initial guess for solution of linear system, `TOL` convergence tolerance - applied to maximum norm of difference between successive approximations, `NMax` maximum number of iterations to be performed, and *output*: `x` approximate solution of linear system.

   NOTE: You are not allowed to use Matlab's implicit matrix-vector multiplications. The implementation has to use explicit `for` loops to carry out matrix-vector operations.

   Provide a program with an example of a system that uses the function `gs_solve` for a system of at least 20 equations.

3. (20 points) Write a Matlab function that performs a Lagrange polynomial interpolation as discussed in class. That is, write a function `function y=interpol_lagrange(x,px,py)`, where `px` and `py` are the tabulated values of the function $y(x)$, `x` is the value or values where the interpolation is calculated. `y` is the output of the interpolation. Provide a program that uses `interpol_lagrange(x,px,py)` with a set of at least 20 tabulated points. The program must display a figure in which you plot the Lagrange polynomial interpolating function as well as the tabulated points used in the contraction of the interpolating function.

4. (20 points) Repeat the previous problem but for `cubic spline interpolation`. Notice that in this case you will also need to write a tri-diagonal solver. You can use Matlab solver for this problem. Writing your own tri-diagonal solver will be extra credit. **CHANGE MADE TO THIS ASSIGNMENT***

5. (20 points) Write a function `I = simpson38(f,a,b,n)` that computes and approximation the integral $I = \int_a^b f(x)\,dx$ of a function $f(x)$ in the interval $[a,b]$ with $n+1$ equally spaced points using the Simpson 3/8 rule. Provide an example that demonstrates that the global error of integration is given as discussed in class.