ME 701 – Development of Computer Applications In Mechanical Engineering
Homework 9 –Due 11/27/2017

Instructions: All source code should be placed in a single TAR file of the form `lastname_firstname_hw8.tar`. All code should be compiled with a Makefile, as specified in Problem 3.

Problem 1 – Just One!

Consider the equation

$$\frac{d}{dx} a(x) \frac{dy}{dx} + b(x) y(x) = c(x) \tag{1}$$

subject to $y(L) = y_L$ and $y(R) = y_R$. Here, a, b, and c are arbitrary functions of x provided in *tabulated* format. For simplicity, assume that a, b, and c are defined for the same values of x.

Write a function or subroutine in C++ or Fortran with the following signature

```
void solve(int n, double L, double R, double y_L, double y_R,
           int m, double *a, double *b, double *c,
           int option=0, double tol = 1e-8, int maxit = 100);
// similar for fortran
```

where `n` is the number of equally spaced x points at which $y(x)$ is to be evaluated, L and R are the left and right boundaries, y_L and y_R are the left and right boundary conditions, and `a`, `b`, and `c` are double arrays evaluated at `m` equally spaced points from `L` to `R`. The argument `option` is an integer for which `0` means use tridiagonal elimination, `1` means use Jacobi, `2` means use Gauss-Seidel, and `3` means use any other iterative method of your choice (+1/2 point). Finally, `tol` and `maxit` define the tolerance and maximum number of iterations for options 1, 2, and 3.

You may wish to produce a `main.cc` for testing, or to couple this with Python via `f2py` or `swig`. All I want is your function or subroutine code (include a header if doing C++).

Note!!! I strongly encourage you to work out the finite difference (or any other approximation you want to use) before lecture on 11/17 so that I can tell you if you're on the right track.

Hint!!! Have a reference solution handy for debugging and method verification! (Solve, e.g., $y'' = 1$ with $y(0) = y(1) = 0$, etc.).