Description of Material Presented in Lectures:

The following description relates the algorithms presented in Math 4610 to a variety of computational skills that will be discussed during lectures. The course will attempt to illustrate the use of High Performance Computation (HPC) on specific algorithms. The idea is to introduce students to computational techniques that can be used to increase the performance of computer codes that implement numerical and computational algorithms.

1. **Lecture: First Day**
   - **Linear Systems of Equations with OpenMP:** OpenMP provides inline directives, a library of intrinsic functions, and global variables that can be used by compilers to optimize executable performance.
   - **The Bisection Method with Recursion:** Every student of computational mathematics should know how to use recursion effectively. Some programming languages allow the use of recursive constructs.
   - **Newton's Method and Interval Analysis:** Interval analysis provides a means to put bounds on the accuracy of some approximation methods. Note that there are a number of people trying to use these ideas to provide computer aided proofs in mathematics.
   - **Bracketing Roots with Parallel Methods:**
   - **Monte Carlo Integration:** Using GPUs to ...
   - **Numerical Integration:** Use properties of the integral to divide and conquer.
   - **Composite Quadrature Rules: The Trapezoid and Simpson's Rules** Rewriting formulas to determine equivalent approximations to increase efficiency of the algorithm.
   - **Power Method for Estimating Largest Eigenvalue:** Using parallel algorithms to increase code performance.
   - **Linear Regression for Overdetermined Systems:** Normal equations approach.
   - **Fast Fourier Transform (FFT):** Matrix multiplication versus the FFT.
   - **Cholesky Factorization:** for symmetric, positive definite (spd) linear systems.
   - **QR-factorization:** matrices for the solution of linear systems of equations.
   - **Estimating the Condition Number of a Matrix**
   - **Euler's method for solving Initial Value Problems.**