The class content consists of five main topics. These are, in order of presentation in class: numerical solution of initial value problems, Monte-Carlo methods, matrix factorizations, optimization, and numerical methods for solving nonlinear systems. At this writing, we have covered the first three and are about to start the fourth. The aim of the project is for you to explore and test at least one of these topics in depth, by identifying a benchmark problem whose solution requires at least one of the methods we have studied, demonstrating the performance of the method(s) you are using, identifying advantages or disadvantages of such methods, and, if possible, showing that the methods behave the way analysis of them predicts.

Examples of possible topics are:

1. **Optimization.** Identify a nontrivial optimization problem and explore the performance of various solution strategies to solve it. These could include Newton’s method, alternatives that avoid the computation of Hessians, and (possibly) simple methods that avoid the computation of gradients.

2. **Nonlinear solvers.** Identify a nontrivial nonlinear system of equations and explore Newton’s method and variants of it.

3. **Image processing.** The text discusses the use of matrix methods to display images on a screen, and to recover corrupted images. Explore this problem and compare several methods for solving it.

4. **Stability analysis of dynamical systems.** If you have some knowledge of this topic: stability analysis requires the computation of the rightmost eigenvalues of Jacobian matrices associated with a dynamical system. Such systems usually depend on a parameter, and there is a critical value where the system becomes unstable. One way to study this problem is to compute all the eigenvalues of the matrices involved. Another way is to use iterative methods (related to the Lanczos method studied in HW3), which are built into the MATLAB program `eigs`. Compare these approaches for performing stability analysis.

5. **Impact of computer architecture.** Write your own version of a matrix factorization routine (for example, LU decomposition) and explore its performance as a function of how you structure the code. If you have access, this could be done on a high performance computer or a graphics processing unit. It is not necessary to use MATLAB for this, and in fact it might be easier without it.

6. **Probabilistic methods.** Learn about and explore the use of a quasi-Monte-Carlo method for numerical quadrature (see the web for definitions), and compare its performance with the "standard" Monte-Carlo method.

For benchmark problems, you could look at the text for examples, or the web. If you are already involved in a research project, you could take something that complements what you are doing for your research. It is important that you choose something that exercises at least one of numerical techniques studied in the class. You should begin by writing up a short (half a page should suffice) description of what you want to do.