Rules: Projects can be completed at any time during the semester. All students are required to do at least one project, either from the list, or approved by me (the second option may be especially appealing for students already doing computing research on their own). Students must work individually. Project options will be posted here throughout the semester. One project is required. A second project may be turned in to replace two quiz grades.

Formatting: Turn in a written explanation of what you have done, including mathematical justification of important steps. If you do a significant amount of coding, more than say 8 lines, turn in your code as well. Include figures where appropriate.

Options:

1. **Newton's Method :** Write a code to find the roots of the function

$$\begin{pmatrix} f_1(x,y) \\ f_2(x,y) \end{pmatrix} = \begin{pmatrix} x^2 - y^3 \\ \sin(y)(x^2-1) \end{pmatrix}$$

   using Newton's Method. Find three distinct roots. Choose two of these and compute their *basins of attraction*. The basin of attraction is the set of initial guesses that converge to this root. Show your solution in the form of a graph. If you are using matlab I recommend the plotting command pcolor to display your results.

2. **Rounding Error :** Consider approximations of the derivative of $f(x) = x^3$ at $x = 1$ using a secant method

$$f'(x) \equiv \frac{f(x+h) - f(x)}{h}$$

   and a complex step derivative

$$f'(x) \equiv \text{Imag}\left(\frac{f(x+ih)}{h}\right)$$

   Plot the absolute error of these approximations as the step size h goes to zero (machine precision). Find which step size gives the minimum error for each method. Comment on reason for this difference with respect to rounding error. Which method performs better? Find a limitation of this method. You may use the literature to find such a limitation, but cite your source.

3. **Matrix Inversion:** Write programs to invert a matrix using Gaussian elimination and the Jacobi iteration. Apply your codes to matrices of varying size. Use the residual from your Gaussian elimination program to set the tolerance for your Jacobi iteration. Plot the number of iterations necessary for your Jacobi iteration to converge as a function of matrix size. Be sure your matrix size varies over a few orders of magnitude. What conclusions can you make about the cost of each scheme?

4. **Interpolation:** Derive the tri-diagonal matrix associated with fitting a cubic spline to n data points, with "natural" boundary conditions. Write a program that finds the cubic spline for your favorite function on the interval $[-1, 1]$. Compare the error of your interpolation with $n = 10, 20$, and 100.