Computational Physics; January–May 2020
Assignment 1
Due: Thursday, 12 March 2020

Instructions

- When you are asked to write a code, submit your code by posting it to Github and sending in the Github link.
- When you are asked to solve something manually, or when you are asked for a number as an answer, submit your response in writing.

Direct methods for solving linear systems

▶ Solve the system of equations

$$\begin{bmatrix} 1 & 0.67 & 0.33 \\ 0.45 & 1 & 0.55 \\ 0.67 & 0.33 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$$

manually using Gaussian Elimination assuming that the calculation is being done on a decimal computer capable of carrying only two floating-point digits.

▶ Solve the above equations using `numpy.linalg.solve`. How do the two answers compare?

▶ Write a C code and using ready-made GSL functions compute the LU decomposition of the above matrix. In your code, demonstrate that the LU decomposition is correct.

▶ Show that Gaussian Elimination is an $\mathcal{O}(n^3)$ method for solving a system of linear equations $\mathbf{Ax} = \mathbf{b}$.

▶ Reduce the problem of inverting a complex matrix to that of inverting real matrices.

Iterative methods for solving linear systems

▶ Starting with the zero vector as your initial guess, manually compute the first two iterations of the Jacobi Method for the equation

$$\begin{bmatrix} 3 & -1 & 1 \\ 3 & 6 & 2 \\ 3 & 3 & 7 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 4 \end{bmatrix}$$

▶ Repeat above for the Gauss-Seidel Method.

▶ Starting with the zero vector as you initial guess, repeat the above two exercises for the equation

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 \\ -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \\ 0 \\ 6 \\ -2 \\ 6 \end{bmatrix}$$

▶ The vector

$$\mathbf{x} = \begin{bmatrix} 7.859713071 \\ 0.422926408 \\ -0.073592239 \\ -0.540643016 \\ 0.010626163 \end{bmatrix}$$

solves the equation

$$\begin{bmatrix} 0.2 & 0.1 & 1 & 1 & 0 \\ 0.1 & 4 & -1 & 1 & -1 \\ 1 & -1 & 60 & 0 & -2 \\ 1 & 1 & 0 & 8 & 4 \\ 0 & -1 & -2 & 4 & 700 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}.$$

Write Python code to solve this equation using Jacobi, Gauss-Seidel, Relaxation, and Conjugate Gradient methods. (Use $\omega = 1.25$ for the relaxation method.) In each case, stop when the difference between the approximate solution vector and the true solution written above is less than 0.01. (In other words, have a tolerance of 0.01.) Mention how many iterations you need in each method to reach this accuracy.

Eigenproblems

▶ Write a Python code that produces the QR decomposition of

$$\mathbf{A} = \begin{bmatrix} 5 & -2 \\ -2 & 8 \end{bmatrix}$$

using `numpy.linalg.qr`. Use the decomposition to calculate the eigenvalues of the matrix. Compare your result with that produced by `numpy.linalg.eigh`.

▶ Write a Python code to apply the Power Method to find the dominant eigenvalue and eigenvector of the matrix

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}.$$

Your eigenvalue should be accurate to within a percent.

▶ Write a Python code to determine the Singular Value Decomposition of an $m \times n$ matrix. Use this code to determine the singular values of

$$\begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

Compare your result with that obtained using `numpy.linalg.svd`. Compare the time taken by your code with that taken by `numpy.linalg.svd`.

Libraries

▶ If available, list the names of LAPACK, GSL, and Numpy functions to do the following computations:

1. Solving a system of linear equations using Gaussian Elimination with Pivoting
2. Solving a system of linear equations using the Jacobi method
3. Solving a system of linear equations using the Gauss-Seidel method
4. Solving a system of linear equations using the Relaxation method
5. Solving a system of linear equations using the Conjugate Gradient method
6. Obtaining the LU decomposition of a matrix
7. Obtaining the QR decomposition of a matrix
8. Obtaining the singular value decomposition of a matrix
9. Obtaining the eigenvalues of a real symmetric matrix
10. Obtaining the eigenvalues of a complex Hermitian matrix
11. Obtaining the eigenvalues of a general real or complex $n \times n$ matrix