Project: Comparing Direct and Indirect Methods for Solving Linear Systems of Equations

Solutions to Linear Systems of Equations

Due: 11:59PM, 2025-05-05

1 Project Overview

In this project, you will implement and analyze two numerical methods for solving linear systems of equations: Gaussian Elimination with partial pivoting and the Gauss-Seidel method. You will then compare the performance of these methods in terms of accuracy and computational efficiency.

2 Background: Linear Systems

Linear systems of equations arise in many engineering and scientific applications. Consider the following system:

$$A\mathbf{x} = \mathbf{b}$$

Where $A \in \mathrm{R}^{n\times n}$ is an $n \times n$ coefficient matrix, $\mathbf{x} \in \mathrm{R}^n$ is the vector of unknowns, and $\mathbf{b} \in \mathrm{R}^n$ is a vector of constants.

3 Methods to Implement

3.1 Gaussian Elimination with Partial Pivoting

Gaussian Elimination is a direct method for solving linear systems. The basic procedure transforms the original system into an upper triangular system using elementary row operations. Partial pivoting is a technique used to improve the numerical stability of Gaussian elimination by selecting the element with the largest absolute value in the current column as the pivot element.

3.2 Gauss-Seidel Method

The Gauss-Seidel method is an iterative method for solving linear systems. It starts with an initial guess for the solution and then iteratively refines the solution until convergence is achieved.

1. **Initialization:** Start with an initial guess for the solution vector $\mathbf{x}^{(0)}$.
2. **Compute $\mathbf{x}^{(k)}$:** For $k = 1, 2, \ldots, N$, compute the components of x¡sup¿(k)¡/sup¿ using the following formula:
$$x_i^{(k)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right)$$
3. **Check for Convergence:** If $||\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}||_\infty < \varepsilon$, stop iterations and return $\mathbf{x}^{(k)}$ as the solution.

4 Project Assignment

1. **Implement Gaussian Elimination with Partial Pivoting:** Write a function that implements Gaussian Elimination with Partial Pivoting. This function should have the following inputs: the matrix A and the vector $\mathbf{b}$. It should return the solution vector $\mathbf{x}$ if it was able to solve the equation and throw an error if it is not able to solve it.
2. **Implement the Gauss-Seidel Method:** Write a function that implements the Gauss-Seidel iterative method. This function should have the following inputs: the matrix A, the vector $\mathbf{b}$, an initial guess $\mathbf{x}^0$, a tolerance ε, and a maximum number of iterations N. It should return the solution vector $\mathbf{x}$ if it was able to solve the equation and throw an error if it is not able to solve it.
3. **Testing and Analysis:**
    - Use your two algorithms to solve the small and large linear systems described in the "Supplement" section.
    - Compare the solutions obtained by both methods. Was Jacobi faster than Gaussian Elimination? Which method was more accurate?
    - GRADUATE STUDENTS: Compare the computational costs each method.
    - GRADUATE STUDENTS: Discuss how the properties of each method impact the performance and accuracy of the method/solution.

5 Supplement

To help you develop your methods, I have provided a few supplementary files.

Unpack the `supplements.zip` file into your project directory.

This folder contains two `.py` files: `utils.py` and `main.py`. `utils.py` has utility functions for you to set up two test systems to help you format and solve a linear system using the methods details above. You can call `get_small_system()` to load the following matrices into variables A and $\mathbf{b}$:

$$A = \begin{bmatrix} 10 & 2 & 2 & 2 & 1 \\ 1 & 11 & 2 & 1 & 1 \\ 1 & 1 & 12 & 1 & 1 \\ 2 & 1 & 2 & 13 & 2 \\ 2 & 2 & 2 & 2 & 14 \end{bmatrix},$$

$$\mathbf{b} = \begin{bmatrix} 119 \\ 96 \\ 106 \\ 107 \\ 94 \end{bmatrix}$$

And you can call `get_large_system()` to load a 100×100 matrix A, a 100 dimensional vector $\mathbf{b}$, and solution vector x. This is the system that I would like you to report your results for. You can use the small system to test your algorithms in a less computationally intensive setting.

The file `main.py` has examples of how to call these functions and can be used as the basis for your submission.

`utils.py` also has helper functions for computing $||\cdot||_\infty$.

6 Submission

You will need to submit the following:

- **Source Code:** Submit all code files implementing Gaussian Elimination with partial pivoting and the Gauss-Seidel method. Code should be well-commented and easy to understand. Include a README file explaining how to run your code.
- **Report:** A written report (in PDF format) that includes:
  - GRADUATE STUDENTS ONLY: A clear description of the implemented methods.
  - Results of your testing and analysis described in the previous sections.
  - Discussion of your findings. Include the following:
    - Compare the accuracy of the two methods using the residuals $\mathbf{r} = A\mathbf{x}^{(k)} - \mathbf{b}$ and the norm: $||\mathbf{x} - \mathbf{x}^{(k)}||_\infty$. How do they compare?
    - Discuss any challenges you encountered when implementing the methods.
  - GRADUATE STUDENTS: Comparison of the computational costs each method.
  - GRADUATE STUDENTS: Discussion of how the properties of each method impact the performance and accuracy of the method/solution.

7 Grading

50 points Complete, functioning, and correct code that meets the specifications outlined above.

50 points Report which includes the following:

- Clear descriptions of the numerical methods implemented.
- Presentation of test results with appropriate visualizations.
- Insightful discussion of the method's performance and analysis.

Any work you submit for grading, must be completely your own work. You should not base your code in any way on anyone else's code (including mine). Any outside sources used for general background should be cited.
