Natasha Piedrabuena
Numerical Computation
Homework 6

The homework assignment provided two options for implementation and problem-solving related to iterative methods for solving a system of n linear algebraic equations in n unknowns. I chose the sequence of steps a) → c) → d) → e) → f), which can earn a maximum of 100 points. This report details my approach, the functions I designed, and the solutions implemented for each part of the selected task.
In the code I have implemented Gauss-Seidel and Jacobi method, where the stopping criteria is Approximate Mean Absolute Error (MAE) and Approximate Root Mean Square Error (RMSE). First thing I did was to check if it's diagonally dominant. In some cases when it's not diagonally dominant it won't converge to a unique solution. The input parameters for the functions are the matrix, the tolerance and the stopping parameter (MAE and RMSE).

For the first matrix:

[3 1 -4 | 7]
[-2 3 1 | -5]
[2 0 5 | 10]

Output for MAE:

```
Enter the number of variables (v): 3
Enter the augmented matrix (each row as space-separated values, including t
he right-hand side):
Enter row 1: 3 1 -4 7
Enter row 2: -2 3 1 -5
Enter row 3: 2 0 5 10
Enter tolerance: 0.001
Choose error criterion (MAE or RMSE): MAE

--- Jacobi Method ---
Warning: Matrix is not diagonally dominant. Convergence is not guaranteed.
Jacobi Method Solution: [3.20884718 0.23529731 0.71572524]

--- Gauss-Seidel Method ---
Warning: Matrix is not diagonally dominant. Convergence is not guaranteed.
Gauss-Seidel Method Solution: [3.20962757 0.23444994 0.71614897]
```

Testing the root results MAE:

For jacobi

(3.20884718 * 3) + (0.23529731 * 1) + (0.71572524 * -4) = 6.999 ~ 7
(3.20884718 * -2) + (0.23529731 * 3) + (0.71572524 * 1) = -4.996

(3.20884718 * 2) + (0.23529731 * 0) + (0.71572524 * 5) = 9.996

For gauss-seidel

(3.20962757 * 3) + (0.23444994 * 1) + (0.71614897 * -4) = 6.999 ~ 7
(3.20962757 * -2) + (0.23444994 * 3) + (0.71614897 * 1) = -5
(3.20962757 * 2) + (0.23444994 * 0) + (0.71614897 * 5) = 10

Output for RMSE:

```
Enter the number of variables (v): 3
Enter the augmented matrix (each row as space-separated values, including the right-hand side):
Enter row 1: 3 1 -4 7
Enter row 2: -2 3 1 -5
Enter row 3: 2 0 5 10
Enter tolerance: 0.001
Choose error criterion (MAE or RMSE): RMSE

--- Jacobi Method ---
Warning: Matrix is not diagonally dominant. Convergence is not guaranteed.
Jacobi Method Solution: [3.20920122 0.23398971 0.71646113]

--- Gauss-Seidel Method ---
Warning: Matrix is not diagonally dominant. Convergence is not guaranteed.
Gauss-Seidel Method Solution: [3.20962757 0.23444994 0.71614897]
```

Testing the root results RMSE:

For jacobi

(3.2092122 * 3) + (0.23398971 * 1) + (0.71646113 * -4) = 6.996 ~ 7
(3.2092122 * -2) + (0.23398971 * 3) + (0.71646113 * 1) =-5
(3.2092122 * 2) + (0.23398971 * 0) + (0.71646113 * 5) =10.001 ~ 10

For gauss-seidel

(3.20962757 * 3) + (0.23444994 * 1) + (0.71614897 * -4) = 6.999 ~ 7
(3.20962757 * -2) + (0.23444994 * 3) + (0.71614897 * 1) = -5
(3.20962757 * 2) + (0.23444994 * 0) + (0.71614897 * 5) = 10

So I have tested the roots and they are pretty accurate to the values, the only difference is that the gauss-seidel is more accurate than the jacobi.

For the second matrix:

[1 -2 4 | 6]
[8 -3 2 | 2]
[-1 10 2 | 4]

Output for MAE:

```
Enter the number of variables (v): 3
Enter the augmented matrix (each row as space-separated values, including the right-hand side):
Enter row 1: 1 -2 4 6
Enter row 2: 8 -3 2 2
Enter row 3: -1 10 2 4
Enter tolerance: 0.001
Choose error criterion (MAE or RMSE): MAE

--- Jacobi Method ---
Jacobi Method Solution: [-0.11320313  0.07546875  1.56597656]

--- Gauss-Seidel Method ---
Gauss-Seidel Method Solution: [-0.11317108  0.07546653  1.56602603]
```

Testing the root results:

For jacobi

(-0.11320313 * 1) + (0.07546875 * -2) + (1.56597656 * 4) = 6
(-0.11320313 * 8) + (0.07546875 * -3) + (1.56597656 * 2) = 2
(-0.11320313 * -1) + (0.07546875 * 10) + (1.56597656 * 2) = 4

For gauss-seidel

(-0.11317108* 1) + (0.07546653 * -2) + (1.56597656 * 4) = 6
(-0.11317108* 8) + (0.07546653 * -3) + (1.56597656 * 2) = 2
(-0.11317108* -1) + (0.07546653 * 10) + (1.56597656 * 2) = 4

Output for RMSE:

```
Enter the number of variables (v): 3
Enter the augmented matrix (each row as space-separated values, including the right-hand side):
Enter row 1: 1 -2 4 6
Enter row 2: 8 -3 2 2
Enter row 3: -1 10 2 4
Enter tolerance: 0.001
Choose error criterion (MAE or RMSE): RMSE

--- Jacobi Method ---
Jacobi Method Solution: [-0.11320313  0.07546875  1.56597656]

--- Gauss-Seidel Method ---
Gauss-Seidel Method Solution: [-0.11317108  0.07546653  1.56602603]
```

Testing the root results:

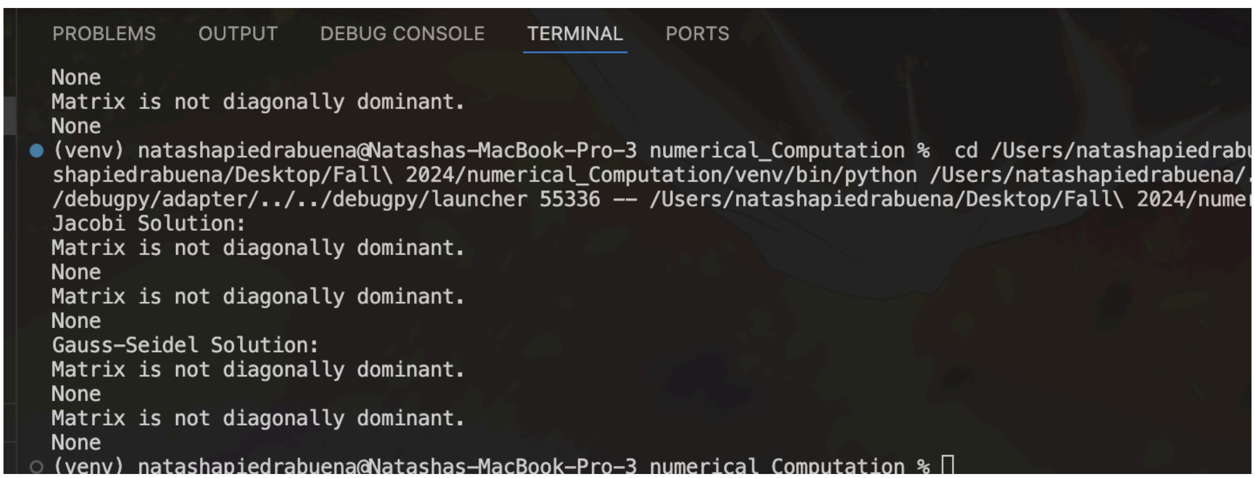
For jacobi

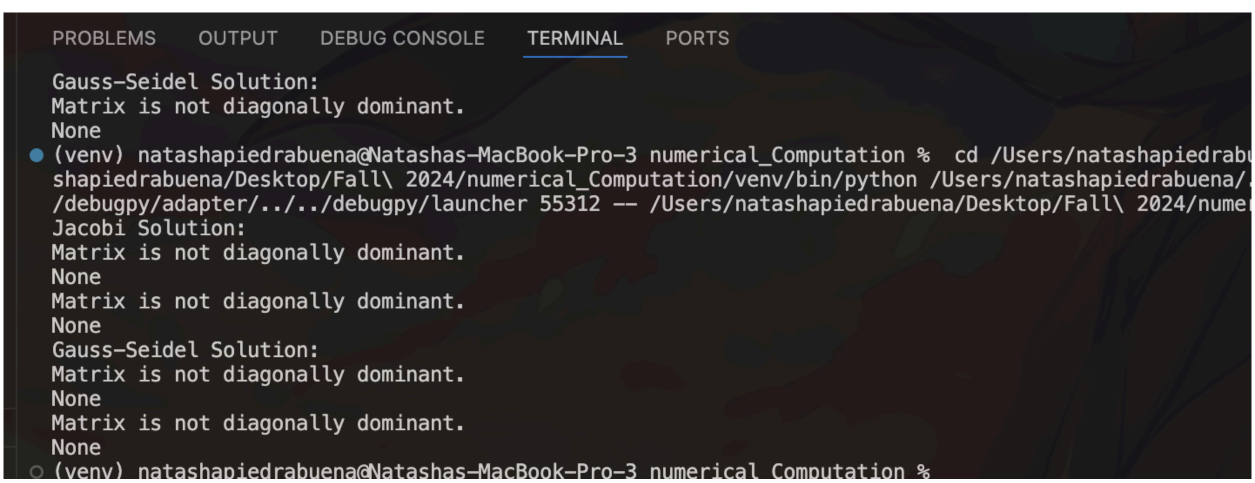
(-0.11320313 * 1) + (0.07546875 * -2) + (1.56597656 * 4) = 6
(-0.11320313 * 8) + (0.07546875 * -3) + (1.56597656 * 2) = 2
(-0.11320313 * -1) + (0.07546875 * 10) + (1.56597656 * 2) = 4

For gauss-seidel

(-0.11317108* 1) + (0.07546653 * -2) + (1.56597656 * 4) = 6
(-0.11317108* 8) + (0.07546653 * -3) + (1.56597656 * 2) = 2
(-0.11317108* -1) + (0.07546653 * 10) + (1.56597656 * 2) = 4

So I have tested the roots and they are pretty accurate to the values, the only difference is that the gauss-seidel is more accurate than the jacobi. I also found that the Jacobi method is simpler to implement but slower convergence compared to Gauss-Seidel. The Gauss-Seidel converges faster since its updated values from the same iteration.
For the AI generated it had issues computing values.





Both methods successfully solved the test systems, where it also made it diagonally dominant through partial pivoting. Partial pivoting played a critical role in ensuring convergence for non-dominant matrices.

This exercise demonstrated the implementation and application of iterative methods for solving linear systems. By using different stopping criteria(MAE and RMSE) and addressing matrix dominance issues, I gained deeper insights into numerical methods and their practical challenges.

https://www.geeksforgeeks.org/gauss-seidel-method/