MA348 Numerical Analysis, Systems of Equations and Matrices

David Jefts

February 18, 2019

Introduction

The goal of this lab is to use the Gaussian Matrix Elimination algorithm to solve for the values in a System of Equations. For the purposes of this lab, the System of Equations was:

$$\begin{cases} x - y + 3z = -3 \\ -x - 2z = 1 \\ 2x + 2y + 4z = 0 \end{cases}$$

This was translated into the matrix below to be used by the Gaussian Elimination software. The dark grey line separates the left-hand-side of the equations with the right-hand-side. Much of the Gaussian Elimination Method is based upon what is called the 'main diagonal', encircled in red in the matrix below.

$$\left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ -1 & 0 & -2 & 1 \\ 2 & 2 & 4 & 0 \end{array}\right]$$

Theory-Analysis

There were no special equations used in this lab, nor were there any assumptions made. The methods used were Gaussian Elimination, explained in Section Numerical Solution, and the Matrix Row Operations, row switching, row multiplication, and row addition. Row switching is swapping the position of one row with another row. Row multiplication is multiplying a row by a constant. Row addition is replacing a row by the sum of that row and some multiple of another row.

Numerical Solution

The entirety of the Gaussian Elimination was coded in Python, with the matrices displayed created in LaTeX.

Gaussian Elimination has two major steps, with two optional steps that are discussed at the end of this section. Forward Elimination, which solves the bottom half of the matrix, followed by Backwards Substitution, which solves for the variable values.

By the end of the Forward Elimination step, the bottom left-hand corner of the matrix, as highlighted below, should be all equivalent to 0.

$$\left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ 0 & -1 & 1 & -2 \\ 0 & 0 & 2 & -2 \end{array}\right]$$

Expected matrix post-Forward Elimination Step

This is accomplished through a multi-step algorithm. Below is a walk-through of the method using the matrix mentioned in the introduction. Matrices can be modified by adding and subtracting rows of the matrix from other rows.

Step 1: The first step begins with the first column and the second row. The goal is to make each spot in the bottom left corner equivalent to 0. For the first step the first row of the matrix is subtracted from each of the other rows a number of times until each spot in that column, below the main diagonal, is equivalent to 0. For the second row, adding the first row just once is sufficient to make a zero ($R_2 + R_1 \rightarrow R_2$, where R_1 and R_2 refer to rows 1 and 2, respectively), in this case the multiplier for the row is -1. To add two rows together, sum the values in each of the matching columns independently. The result is the left matrix. Continuing this process on the third row, achieving a 0 in the first column can be done with: $R_2 - (2 \times R_1) \rightarrow R_2$ and the result is the right matrix, in this case the multiplier would be 2.

$$\left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ 0 & -1 & 1 & -2 \\ 2 & 2 & 4 & 0 \end{array}\right] \qquad \left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ 0 & -1 & 1 & -2 \\ 0 & 4 & -2 & 6 \end{array}\right]$$

Step 2: The second step is the same process as the first step, but using the second row to make the second column, below the main diagonal, equivalent

to 0. For the third row, achieving a 0 in the second column can be done with: $R_2 + (4 \times R_1) \rightarrow R_2$ and the result is the matrix below.

$$\left[\begin{array}{ccc|c} 1 & \text{-1} & 3 & \text{-3} \\ 0 & \text{-1} & 1 & \text{-2} \\ 0 & 0 & 2 & \text{-2} \end{array}\right]$$

The next major step of the Gaussian Elimination Method is Backwards Substitution. Backwards substitution finds the final solution, by converting the matrix back into a system of equations and solving 'up' the matrix to find all of the x values.

$$\left[\begin{array}{ccc|c} 1 & \text{-1} & 3 & \text{-3} \\ 0 & \text{-1} & 1 & \text{-2} \\ 0 & 0 & 2 & \text{-2} \end{array}\right] \longrightarrow \begin{cases} x - y + 3z = -3 \\ \quad -y + \ \ z = -2 \\ \qquad\qquad 2z = -2 \end{cases}$$

z can now easily be solved for, resulting in -1. Substitute z into the second equation to find y, then substitute y and z into the first equation to calculate x. Then the matrix is solved.

Two minor methods that are not always necessary for the Gaussian Elimination Method are Scaling and Partial Pivoting. Scaling is dividing each element by the highest value in a given row. This scales down the matrix so that no number is greater than 1. This can reduce the complexity of the matrix and help it be more readable, but it also increases the chance of round-off errors in computers since some fractions cannot be well-represented digitally.

Partial Pivoting is required if, at any point in the Forward Elimination Step, a number in the main diagonal is equivalent to 0. This causes errors when finding the multiplier since numbers cannot be divided by 0. Partial Pivoting is the swapping of two rows so that the number on the main diagonal is not 0. The current row is swapped with the row that has the highest absolute value for the column currently being worked on. So if the current row has a 0 in the 3rd column and this 0 is on the main diagonal, then the Partial Pivot would swap the current row with whichever row has the greatest absolute value in the 3rd column.

Results and Discussion

The solved System of Equations used as an example in this lab to demonstrate the Gaussian Elimination Method is shown below.

$$\begin{cases} x - y + 3z = -3 \\ -x - 2z = 1 \\ 2x + 2y + 4z = 0 \end{cases} \longrightarrow \begin{aligned} x &= 1 \\ y &= 1 \\ z &= -1 \end{aligned}$$

Conclusions

The Gaussian Elimination Method is very effective at solving matrices, however it has a high operation cost, at $O(n) = O(\frac{2}{3}n^3) + O(n^2)$. Additionally, the method may require the entire matrix to be scaled beforehand which can increase the chance of rounding error when solving with a computer, and certain matrices require pivoting which slightly increases the overall complexity. In conclusion, the Gaussian Elimination Method is very effective at solving matrices and guarantees a solution, but it sacrifices operational power to do so.

Appendix A

```
def forward_elimination(self):
    for k in range(self.n - 1):  # loop through steps
        for i in range(k + 1, len(self.matrix)):  # loop through rows in bottom left corner
            if self.matrix[k][k] == 0.0:
                print("pivoting")
                self.partial_pivoting(k, k)
            m = self.matrix[i][k] / self.matrix[k][k]
            for j in range(len(self.matrix) + 1):  # loop through columns
                self.matrix[i][j] = self.matrix[i][j] - m * self.matrix[k][j]
        print(self)

def backward_substitution(self):
    sol = [0] * self.n
    for row in range(self.n - 1, -1, -1):  # loop through rows backwards
        sum = self.matrix[row][self.n]  # value in right-most column
        for col in range(row + 1, self.n):  # loop through columns
            sum -= self.matrix[row][col] * sol[col]  # subtract each known value
        sol[row] = sum / self.matrix[row][row]  # divide by the diag value
    self.sol = sol

def partial_pivoting(self, r, c):
    row = 0
    largest = 0
    # find the row with the largest value for the input column
    for i in range(self.n):
        if abs(self.matrix[i][c]) > largest:
            largest = self.matrix[i][c]
            row = i
    # swap the row found above with the input row
    temp = self.matrix[r]
    self.matrix[r] = self.matrix[row]
    self.matrix[row] = temp

def scale_matrix(self):
    for i in range(len(self.matrix) - 1):
        largest = 0
        for j in range(len(self.matrix[i])):
            if abs(self.matrix[i][j]) > largest:
                largest = self.matrix[i][j]
        for j in range(len(self.matrix[i])):
            self.matrix[i][j] = self.matrix[i][j] / largest
```