Linear System Solver

By Justin Pretlow

My program is intended to solve a system of linear equations. The goal of it is to be able to handle systems having any number of variables and equations. The program reads input from a file and prints out the solution. In cases where there are infinitely many solutions it prints [NaN, NaN], and in cases where there is no solution it prints [Infinity, Infinity]. Otherwise, it prints the solution for the system. To run it you type main while in ghci and then type the path to your file and the program will then print one of three possible solutions.

My program first has a function and its helpers at the top of the file that had not been implemented. This function, swap, was to handle the edge case of when the pivot is zero when reducing the system into row echelon form. This edge case does not happen every time thus I did not get to it before the deadline. Next, you will see the sources that I used for research. One of the sources is a document explaining the process of solving systems of linear equations. The second source was a helpful implementation of solving a system of linear equations. I included this since I could not get my implementation of a substitution method to work correctly in time. This function is vital to the program to work and compile so I included it for demonstration purposes and a temporary workaround until I can figure out how to get my implementation working. Next, are the types that I used which simply state that a Matrix type is a list of Rows which is a list of Doubles.

The functions that I used are ref, gaussElim, substitute, solve, parseInput, and my main function. Ref takes in a matrix and an integer representing the row you want to start at. This function performs row reduction on a matrix to aid in getting the matrix into row echelon form. GaussElim is a function that applies Gaussian elimination to transform a given matrix into row echelon form by recursively eliminating rows starting from the first row until all rows have been processed. The substitute function is from Luckytoilet[1] and is vital for the program to work. The author states, “folding from the right, each step it substitutes in the corresponding solution and multiplies and subtracts to get the next solution, adding that to the solution list”[1]. Next, the function, solve, takes a matrix as input and uses gaussElim to get the matrix into row echelon form and then back substitutes to solve for the variables. The parseInput method parses the input file, takes it from String, and outputs a matrix. Lastly, the main outputs, “enter the path to the input file”, reads the file from the path, parses the input, finds the solution by calling solve on the matrix from the file, and prints the solution. The solution is either a list of the values of the variables, [NaN, NaN] in the case of infinitely many solutions, and [Infinity, Infinity] in the case that there is no solution.

Research wasn’t a major part of the project. Reviewing how to solve a system of linear equations was very necessary since I had not done this in a while and a lot of practice by hand was necessary. Bruce Ikenaga’s[2] document was very helpful in aiding in reviewing this subject. Luckytoilets’ implementation[1] gave great insight into how to approach the problem when I came to the point of not knowing where to go next. But I came across this a little too late. I do think that if I had done more research then parts of this program would have been implemented better and or finished such as in the cases of the swap and substitution function.

The most difficult parts of the project were the swap and substitution functions. I did not finish these functions and used Luckytoilet's[1] substitution function since it was vital for the program to work. I did not fully understand how to tackle those pieces and in the end, was not able to finish them. If I had given myself more time then I would have most likely been able to figure out these pieces to this puzzle. I am proud of the ref function and that alone took a while to implement. It required me to go step by step by hand and think about how to turn that into a function that I could use for my program.

In the future, I would like to finish the swap function and implement my version of the substitute function. The swap function is vital in handling the edge case when the pivot is zero when executing Gaussian elimination. The substitute function is vital for back substitution when solving the matrix. Making these changes would vastly improve the quality of the project. Additionally, I would like to add functionality to properly handle the cases of infinitely many solutions and no solutions. After the following changes and improvements then my program would properly handle the goals of the proposal.

In conclusion, my program's goal was to solve and handle systems of linear equations having any number of variables and equations. I met some of the requirements but fell short in the end. If I had started the project sooner and given myself more time I would have been able to figure out these helper methods. This project turned out to be more difficult than I anticipated. What I have learned is to start things sooner rather than later. Doing so will allow you to ask those difficult questions you come across when tackling a project such as this.

References

[1] luckytoilet. (2010). Solving systems of linear equations in Haskell. https://luckytoilet.wordpress.com/2010/02/21/solving-systems-of-linear-equations-in-haskell/

[2] Bruce Ikenaga. (2021). Solving Systems of Linear Equations. https://sites.millersville.edu/bikenaga/linear-algebra/solving-systems-of-linear-equations/solving-systems-of-linear-equations.pdf