1 Abstract

For the final project of AMS 209, I chose to build and execute the numerical linear algebra solver. The portion of the code used for the numerical operations to solve a linear system \( \mathbf{A} \mathbf{x} = \mathbf{b} \) was built using various FORTRAN subroutines, with a Python program used to compile and execute the FORTRAN portion. Python was also used to build the different linear systems that were to be solved, as well as output solutions in both .dat files and visual plots.

2

2.1 Methods

From the PyRun folder in the project directory, the entire program can be executed by the command

\[
\text{python pyRun_linalg.py}
\]

This initiates the program to query the user as to whether the program should run and solve the linear systems using partial pivoting or not. Once the user enters the required ‘1’ or ‘2’, the Python program then proceeds to check whether a FORTRAN-created executable exists already. If so, the directory is directed to make clean in order to clear any old executables, and old .dat, .o, or .mod files. The program then proceeds to initiate a make command and compile the FORTRAN program.

The FORTRAN implementation utilizes a central program entitled linear_solve.f90 to call three different modules. The setup module contains a read data subroutine that reads in which the attributes of each linear system, including its size, and a write to screen subroutine that prints the input values so the user can verify accurate input.

Next is a solve module that breaks up an LU decomposition process into three separate steps. The first subroutine LU_decomp includes procedures to solve using partial pivoting as well as not using it. The other two modules conduct a forward and backward solver.

Finally, the write module outputs the solution for \( \mathbf{x} \) into a .dat file.

The Python program creates the .dat files for all linear systems as well as compares the solutions to a Python linear solver and plots visual representations of the matrices and vectors using matplotlib.

2.2 Results

Below are the plots obtained using the program. Without partial pivoting, we have below...
With partial pivoting, we obtain
2.3 Findings

A big issue I could not resolve was obtaining an accurate answer for the third system using partial pivoting. I was able to obtain the correct answer not using partial pivoting, but I believe my code for the forward substitution did not accurately reflect the process. This remains an area that needs improvement.

3 Conclusion

The process of coding relatively simple mathematical procedures in two different computing languages proved a worthy and challenging endeavor. With any luck further studies will utilized the skills I’ve honed in 209. Thanks for a great quarter!