AM213A Homework 2 Part 1 (Numerical Problems)

Owen Morehead

February 11, 2022

1 General Guidelines

All the files for this assignment include: driver file, *Driver_LinAl.f90*, the *LinAl.f90* program containing all the subroutines, a *Makefile*, and the *Amat.dat* and *Bmat.dat* data files containing a 4×4 matrix A and a matrix B containing n rhs vectors.
I am coding this assignment in Fortran 90 and have used the templates provided.
All code for all 5 problems in this numerical assignment are contained within one git directory titled *code* within the *Part1* subdirectory.
The makefile is setup to compile the program such that the command *make LinAl* will make the executable that can then be run with the command, *./ LinAl* which will print the results for assignment labeled problems 2, 3, 4, and 5 in order and with labels.
There is an additional matlab file, *am213a_hw2_problem5.m*, that is used in problem 5 to make the 3d plot of the given points and the plane which passes through all of these points, solved numerically using Gaussian elimination.

2 Warming Up

All the specified routines have been written. A comparison of the analytical and numerical results follows. We see matrix A = $\begin{pmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{pmatrix}$ and has dimensions 4×4.
The trace of this matrix is the sum of the diagonals, Trace$(A) = 2 + 3 + 9 + 8 = 22$. The two norm or Euclidean norm of each column vector of A are:
$||A_{i,1}||_2 = \sqrt{120} = 10.9544511501$, $||A_{i,2}||_2 = \sqrt{108} = 10.3923048454$,
$||A_{i,3}||_2 = \sqrt{108} = 13.1148770486$, $||A_{i,4}||_2 = \sqrt{90} = 9.48683298051$.
All these values match the values found numerically.

3 Gaussian Elimination with Partial Pivoting

The matrices used for both the Gaussian elimination and LU decomposition routines are the 4×4 nonsingular matrix A and the n rhs column vectors formed into the 4×6 matrix B:

$$A = \begin{pmatrix} 2.0 & 1.0 & 1.0 & 0 \\ 4.0 & 3.0 & 3.0 & 1.0 \\ 8.0 & 7.0 & 9.0 & 5.0 \\ 6.0 & 7.0 & 9.0 & 8.0 \end{pmatrix}, \; b = \begin{pmatrix} 3.0 & 0.0 & 1.0 & 0.9 & 2.1 & 3.14159 \\ 6.0 & -2.0 & 1.0 & 10.4 & -491.2 & -4.71239 \\ 10.0 & 2.0 & 0.0 & -20.2 & 0.12 & 2.24399 \\ 1.0 & 10.0 & -5.0 & -5.12 & -51.3 & 2.35619 \end{pmatrix}$$

This is a standard algorithm for the solution of linear systems given as $\mathbf{Ax} = \mathbf{b}$.
The steps of the process includes sweeping the matrix from left to right and successively zeroing out all the coefficients below the diagonal using linear operations.
However, this Gaussian elimination method breaks down when the divisor (pivot) is zero or very small ($\epsilon_{mach} \approx 10^{-16}$ for double precision). Hence, the general principle behind partial pivoting is to make sure we always work with the largest absolute value pivot by exchanging rows of the matrix and the rhs vector(s).
The elimination algorithm creates a product of $m-1$ lower triangular matrices that form the non-singular lower trianngular matrix M. For the 4×4 matrix at hand:
$M_3M_2M_1A = MA \rightarrow MAx = Mb \rightarrow Ux = y$ which is an upper triangular system that can be solved by a computationally cheap backsubstitution method to yield the solution to the original linear system, $Ax = b$.
In taking the following steps, this numerical algorithm produces solution vectors that are within machine accuracy.

In summary, this routine does the following: correctly reads and allocates the matrices in `Amat.dat` and matrix of n rhs solution vectors `Bmat.dat`, copies A and B into saved arrays A_s and B_s, prints A and B before Gaussian elimination (GE), performs GE and then prints A and B, performs the backsubstitution and then prints the solution matrix of n column vectors, X, prints the error matrix, $E = A_sX - B_s$, and prints the norm of each of these column vectors of E, values which are within machine accuracy (most values $\leq |10^{-15}|$, largest error is $\approx |10^{-13}|$). Although the largest errors are not *too large*, they make sense given the size of some of the numbers in these calculations. For example, the 5th column of b and hence the same column of the solution matrix both contain the largest values. This is why these solution values have the largest errors and largest error norm as seen below ($\approx 7.0 \times 10^{-13}$).

$$x = \begin{pmatrix} 6.661 \times 10^{-16} & 3.500 & 0.250 & -2.005 & 360.269 & 10.6309 \\ 0.999 & -6.000 & -0.500 & 33.399 & -1234.359 & -22.327 \\ 2.000 & -1.000 & 0.999 & -28.489 & 515.919 & 4.207 \\ -2.999 & 5.000 & -1.499 & 3.689 & 223.040 & 7.1246 \end{pmatrix}$$

$$||\mathbf{E}||_2^T = \begin{pmatrix} 1.831 \times 10^{-15} \\ 9.155 \times 10^{-16} \\ 9.155 \times 10^{-16} \\ 2.513 \times 10^{-14} \\ 6.9966 \times 10^{-13} \\ 5.4025 \times 10^{-15} \end{pmatrix} \tag{1}$$

4 LU Decomposition with Partial Pivoting

This LU factorization method is another popular and efficient way to solve linear systems. It factors a matrix ($A = LU$) into a product of a lower triangular (L) and upper triangular (U) matrix. This

is a key step when inverting a matrix or computing a determinant efficiently. Again, we use row partial pivoting, equivalent to a permutation matrix, $P = P_3P_2P_1$, such that $PA = LU$, where $L = (M')^{-1}$ and $M' = M_3'M_2'M_1'$ is a product of lower triangular matrices created from the partial pivoting.
Solving this problem $Ax = b$ is therefore equivalent to solving $Ux = y$ where $y = L^{-1}Pb$, and note we only need to record the integer permutation vector s which saves space.
The LU backsubstitution is slightly different that GE backsubstitution as we first use foward substitution to generate the solution matrix $y = L^{-1}Pb$ This works easily because the permutation vector was constructed such that $(Pb)_i = b_{s_i}$.
So in total this backsubstitution routine entails creating $y = L^{-1}Pb$ by first letting $y = Pb$ and then successively applying the algorithm $y = M_jy = y - 1_je_j^*y = y - y_j1_j$, which since $1_j = (0, 0, ..., l_{j+1}, l_{j+2}, ..., l_{mj})^T$, y can be easily expressed in component form.

In summary, this routine does the following: reads and allocates the same matrices `Amat.dat` and `Bmat.dat`, copies A and B into saved arrays A_s and B_s (they are already copied when routine starts with GE), prints A before LU decomposition, performs LU decomposition and prints A which is now split into its upper and lower triangular matrices LU, performs the backsubstitution and then prints the solution matrix of n column vectors, X, prints the error matrix, $E = A_sX - B_s$, and prints the norm of each of these column vectors of E. Again, these error values are within machine accuracy (most values $\leq |10^{-15}|$, largest error is $\approx |10^{-13}|$). As shown below, the solution matrix is the exact same as found from GE, therefore the error norms are also the same values. This is good confirmation that both algorithms are reporting the same solution.

$$
\begin{aligned}
\mathrm{L} &= \begin{pmatrix} 1.0 & 0 & 0 & 0 \\ 0.75 & 1.0 & 0 & 0 \\ 0.50 & -0.2857 & 1.0 & 0 \\ 0.25 & -0.42857 & 0.333 & 1.0 \end{pmatrix}, \mathrm{U} = \begin{pmatrix} 8.0 & 7.0 & 9.0 & 5.0 \\ 0 & 1.75 & 2.25 & 4.25 \\ 0 & 0 & -0.8571 & -0.2857 \\ 0 & 0 & 0 & 0.666 \end{pmatrix} \\
\mathrm{x} &= \begin{pmatrix} 6.661 \times 10^{-16} & 3.500 & 0.250 & -2.005 & 360.269 & 10.6309 \\ 0.999 & -6.000 & -0.500 & 33.399 & -1234.359 & -22.327 \\ 2.000 & -1.000 & 0.999 & -28.489 & 515.919 & 4.207 \\ -2.999 & 5.000 & -1.499 & 3.689 & 223.040 & 7.1246 \end{pmatrix} \\
||\mathbf{E}||_2^T &= \begin{pmatrix} 1.831 \times 10^{-15} \\ 9.155 \times 10^{-16} \\ 9.155 \times 10^{-16} \\ 2.513 \times 10^{-14} \\ 6.9966 \times 10^{-13} \\ 5.4025 \times 10^{-15} \end{pmatrix}
\end{aligned}
\tag{2}
$$

5 Equation of Plane from 3 Points

We consider the following three points with corresponding (x, y, z) coordinates, $A = (1, 2, 3)$, $B = (-3, 2, 5)$, $C = (\pi, e, -\sqrt{2})$. In order to find the equation of the plane passing through all three points, we can solve the system of equations:

$$
\begin{aligned}
ax_1 + by_1 + cz_1 &= d \\
ax_2 + by_2 + cz_2 &= d \\
ax_3 + by_3 + cz_3 &= d
\end{aligned}
$$

where we are solving for the constants a, b, c. We can solve this system of equations as $Ax = d$, where $A = \begin{pmatrix} 1 & 2 & 3 \\ -3 & 2 & 5 \\ \pi & e & -\sqrt{2} \end{pmatrix}$ $X = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$ $d = \begin{pmatrix} d \\ d \\ d \end{pmatrix}$
since these equations are parametric in d, we can set d to be any nonzero constant. I chose d = 1 for my numerical solution. I chose to solve this system of equations using Gaussian elimination method. The solution of the plane is of the form:

$$a(x - x_0) + b(y - y_0) + c(z - z_0) = 0$$

where we are solving for the constants a, b, and c, and x_0, y_0, and z_0 can be the coordinates of any point. We find through Gaussian elimination that:

$$a = 3.903357 \times 10^{-2}$$

$$b = 0.363383$$

$$c = 7.806715 \times 10^{-2}$$

where I am rounding at the 6th decimal place and not including all the decimals the fortran output includes. We can normalize these values with respect to a:

$$a = 1.0000$$

$$b = 9.3095$$

$$c = 1.9999$$

Using the first point A, we find that that one equation of this plane is:

$$(x - 1) + 9.3095(y - 2) + (1.9999)(z - 3) = 0$$

This solution can also be write in the form of the system of equations mentioned above, where $d = 1$, but we normalized by dividing by a $\rightarrow d = 1/(3.903357 \times 10^{-2}) = 25.619$, as:

$$(1)(x) + 9.3095(y) + (1.9999)(z) - 25.619 = 0.$$

Plots of this plane and the three points is shown below. We can clearly see that this plane passes through all three points.

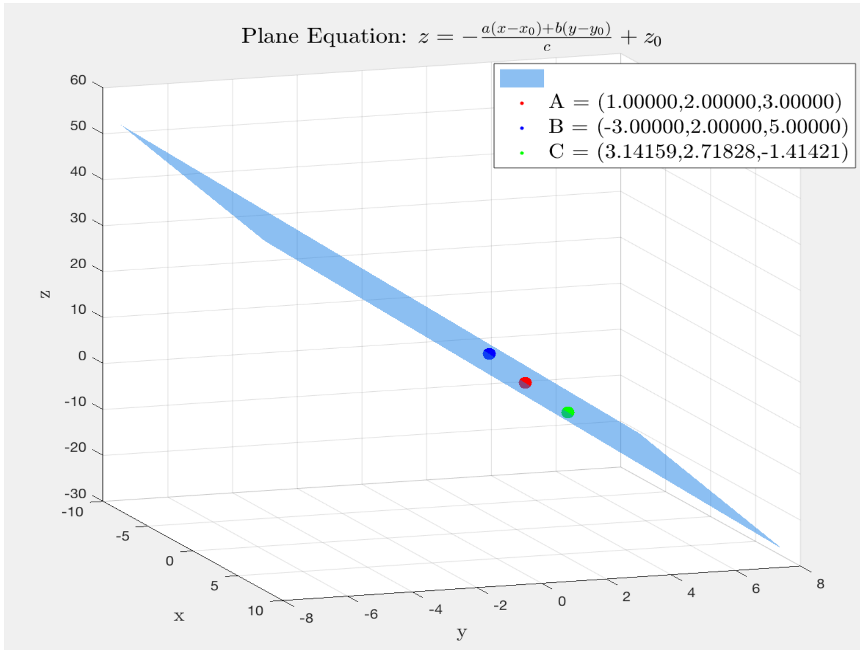

Figure 1: A figure of the plane that connects the three labeled points. The equation of the plane was found using Gaussian elimination with partial pivoting method to solve the linear system of equations defined above.

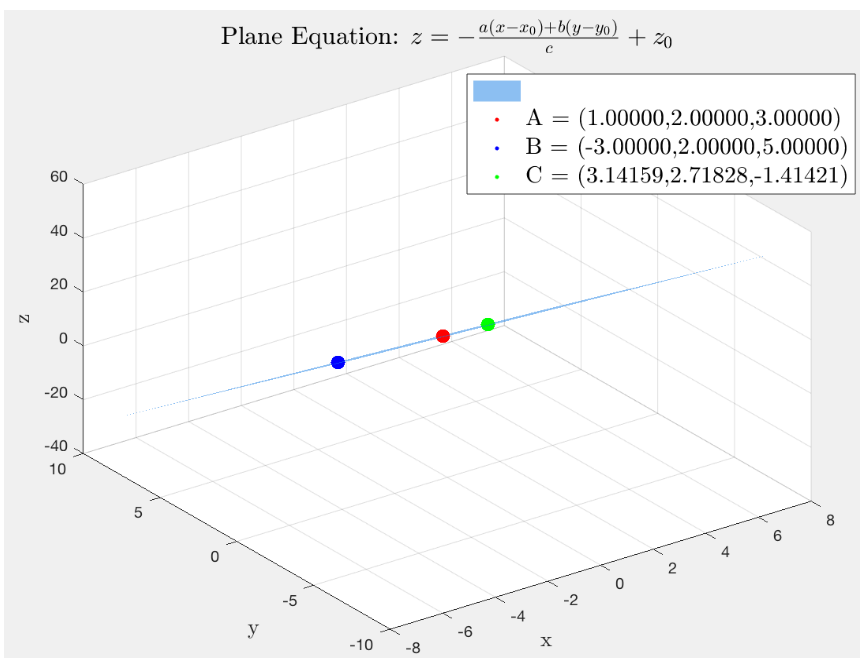

Figure 2: A figure of the plane that connects the three labeled points from a different angle. The equation of the plane was found using Gaussian elimination with partial pivoting method to solve the linear system of equations defined above.