Results.Precision:

The analytical solution to f(x) = 100\*exp(-10\*x) is known to be u(x) = 1-(1-exp(-10))\*x-exp(-10\*x). This is used to compare the accuracy of the numerical solutions obtained with both the general Gaussian elimination algorithm as well as the LU decomposition version for various n. Plots are generated to give a summarized view of the generated output files.

Results.Precision.GaussGeneral:

Figure x shows the analytical solution and the numerical solution for various n using the general Gaussian elimination solver. As n increases, the result is more tightly fit to the analytical solution.

[ins figs Gauss/\*.png]

Figure x: Plots of the analytical solution and numerical solution for numerical Gaussian elimination

The relative error is computed using:

e\_i=|(v\_i-u\_i)/(u\_i)|

where v\_i is the analytical solution value, and u\_i is the numerical solution value

The relative error is plotted for various n in figure y.

[ins figs Gauss/\*\_error.png]

Figure y: Plots of the relative error between analytical and exact solution for numerical Gaussian elimination

Additionally, to verify that the algorithm was working correctly, the overall error as the solver progresses was captured and plotted since the variations in relative error can seem unintuitive. For each n, the average of the relative error is acquired.

[ins fig Gauss/BigError.png]

Figure \_: Average error with respect to n for Gaussian elimination on a linear scale

As n is increased, the overall relative error decreases, which is to be expected as the approximation is closer to the exact result. It is easier to see the trend when plotted on a log scale.

[ins fig Gauss/BigError\_log.png]

Figure \_: Average error with respect to n for Gaussian elimination on a log scale

Results.Precision.LUDecomp:

Figure a shows the analytical solution and the numerical solution for various n using the LU decomposition function-based solver. As n increases, the result is more tightly fit to the analytical solution which is expected.

[ins figs LU/\*.png]

Figure a: Plots of the analytical solution and numerical solution for solving with LU decomposition

The relative error is plotted for the LU decomposition as well in figure b.

[ins figs LU/\*\_error.png]

Figure b: Plots of the relative error between analytical and exact solution when solving with LU decomposition

The overall error is also plotted, both on linear and log scales

[ins fig LU/BigError.png]

Figure c: Average error for LU decomposition on a linear scale

[ins fig LU/BigError\_log.png]

Figure d: Average error for LU decomposition on a log scale

Interestingly enough, the LU decomposition could only go to n=10^4 before running out of memory instead of n=10^5 for the Gaussian elimination solver. Perhaps the implementation in the Numpy library does not allocate memory well.