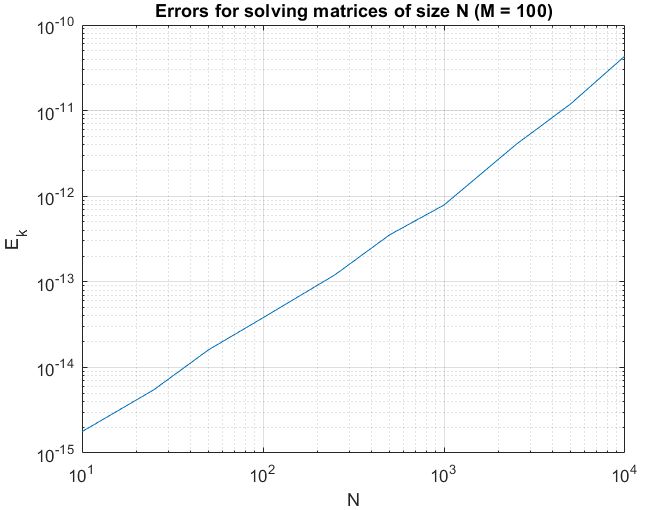
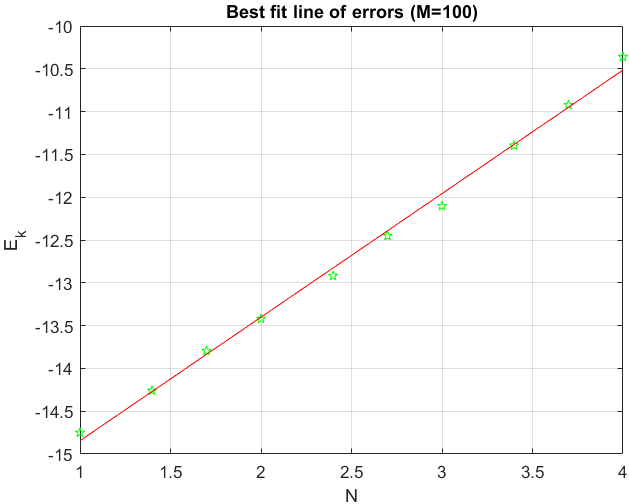
**MACM 316 – A4**

The largest value of N I could choose which didn’t appear to take an absurdly long time to compute was N=10,000. After that I chose 9 more N values which decreased in size by roughly 2.25x down to N = 10. I did this manually. With such values of N, I believed an accurate extrapolation could be obtained. There would be no need to go lower than N = 10 because the error difference would be indifferentiable. Furthermore, allowing N to decrease by roughly 2.25x per step shouldn’t allow for overly large error differences per step. I used M = 100 because any larger would result in lengthy computation times, forcing me to use smaller N, increasing extrapolation error. A smaller M would yield less accurate error estimates due to a smaller number of matrix-solving samples, so I found M = 100 to be ideal.



The extrapolation can be done through the equation E = mN + b <-> E = (N-b)/m for the best fit line. MATLAB provides us m and b through the polyfit function, and we want E to be 1. Since the best fit line is operating on a log graph, we want to extrapolate it for values log(E) and log(N), giving N = 10^((log(1)-b)/m). Plugging in m=1.443296372106925 and b=-16.287938608951094, we get N\*=1.928567809250406e11.

