Homework Project #4
Math 365, Fall 2017

This homework assignment has two parts. In the first part, you will create timing data and in the second part, you will plot your results. Turn in scripts for both parts of the assignment.

Part I
In this part, you will run timing loops and write results to files that you will later use. Turn in the MATLAB script you create for this part, as well as the two data files to your Dropbox folder. I will not run this part of your code; the timing results should be from your computer, not mine.

1. (Matrix-matrix multiply.) Write a timing loop to obtain timing results for a matrix-matrix multiply for a sequence of $N \times N$ matrices $A$. Use `logspace` to create a range of 25 $N$ values in $[2000, \ldots, 5000]$. Write your results to the file `matmat.dat`.

2. (Matrix factorization.) Write a timing loop to obtain timing results for the LU factorization of sequence of $N \times N$ matrices $A$. Use `logspace` to create a range of 25 $N$ values in $[1000, \ldots, N_{\text{max}}]$. Store your timing results in an array and write your results to the file `lufactor.dat`.

Here are some details on this problem:

- To determine a matrix size $N_{\text{max}}$ that you can comfortably fit into your available RAM, first determine how much available RAM you have. Typical values are 8 Gb (GB=Gigabytes), 16GB or 32 GB. For example, if you have 16 GB of RAM available, you should be able to comfortably store a matrix that occupies about 500 MB (MB = Megabytes) of memory. Use the following information to compute $N_{\text{max}}$.
  - An $N \times N$ matrix contains $N^2$ floating point numbers.
  - A floating point number is 8 bytes.
  - A kilobyte is 1024 bytes
  - A megabyte has 1024 kilobytes

- When you solve a linear system using $x = A\backslash b$, MATLAB first factors the matrix $A$ as $A = LU$, and then does forward and backward substitution to solve for $x$. An LU factorization produces a lower triangular matrix $L$ and an upper triangular matrix $U$ such that $LU = A$.

For this problem, we only want to time the factorization step, not the forward and back substitutions needed for a full solve. To call the factorization directly, use the `lu(A)` command (as opposed to $A\backslash b$).

The approximate operation count for the LU factorization is $\frac{2}{3}N^3$.

Part II
In this part, you will post-process and plot your timing results. An important part of this assignment is to make use of more advanced plotting features such as `legend` and to customize tick marks and fontsizes using in labeling.

1. (Matrix-matrix multiply results.) Create a plot showing your timing results for the matrix-matrix multiply. Your final plot should look as close to the left plot in Figure 1 as possible. Your actual data will be different.

   Here are the details you should include in your plot.
2. (Factorization results.) Create a plot showing your timing results for the LU factorization. Your final plot should look as close to the right plot in Figure 1 as possible. Your actual data and $N_{\text{max}}$ will be different.

Here are the details you should include in your plot.

- Load your timing results for the matrix factorization to get the range of $N$ values you used to produce the timing results, and the timing results data.
- Using the `loglog` command, plot your timing results verses $N$.
- Plot the theoretical operation count on your plot. Be sure to scale the value you get so that your timing data and the theoretical count are close.

2. (Factorization results.) Create a plot showing your timing results for the LU factorization. Your final plot should look as close to the right plot in Figure 1 as possible. Your actual data and $N_{\text{max}}$ will be different.

Here are the details you should include in your plot.

- Load your timing results for the matrix factorization to get the range of $N$ values you used to produce the timing results, and the timing results data.
- Using the `loglog` command, plot your timing results verses $N$.
- Plot the theoretical operation count on your plot. Be sure to scale the value you get so that your timing data and the theoretical count are close.
3. (Curve fitting) The temperature dependence of the reaction rate coefficient of a chemical reaction is often modeled by the Arrhenius equation

\[ k = A \exp\left(-\frac{E_a}{RT}\right) \]  

where \( k \) is the reaction rate, \( A \) is the pre-exponential factor, \( E_a \) is the activation energy, \( R \) is the universal gas constant, and \( T \) is the absolute temperature (\( K^\circ \)). Experimental data for a particular reaction yield the following results.

\[
\begin{array}{c|ccccccccc}
T & 773 & 786 & 797 & 810 & 810 & 820 & 834 \\
\hline
k & 1.63 & 2.95 & 4.19 & 8.13 & 8.19 & 14.9 & 22.2 \\
\end{array}
\]

(a) Use polyfit to get a best-fit curve of this data to obtain values for \( A \) and \( E_a \) for the reaction. Take \( R = 8314\, \text{J/kmol/K}^\circ \). Write out the two values \( A \) and \( E_a \) that you find to the file reactioncoeffs.out.

(b) Plot the data points \( (T_i, k_i) \). On the same plot, plot the best-fit curve of your data. Add a legend, axes labels and a title to your plot.

(c) Compute the reaction rate of this process at \( T = 813K^\circ \). Add this value to your plot, and use sprintf and the MATLAB command text to add a label to your plot. Use floating point notation and show 2 significant figures after the decimal place.

Your plot should look like the plot shown in Figure 2.