ASEN2012 - Coding Challenge 3 (25 pts) - Due Wednesday 10/18 by 11:59 pm
Experimental and Computational Methods in Aerospace Engineering Sciences

---

Key Deliverables:

- 5 "Check your answer" questions in Canvas (**10 pts**)
  - **You only get 3 attempts** (correct answers carry over to next attempt) at the quiz.
  - **There is a 5 minute "cool-down" between attempts.** Use this time to debug and reevaluate your code. Once you compute a new answer, make sure to critically think about it before submitting again.
  - **No additional attempts will be provided for any reason.**
- 1 .zip file containing the following (naming convention lastname_firstname_CC1.zip) (**15pts**):
  - Functioning Matlab code (.m file(s) - if using multiple files, please name driver code main.m)
  - Published code (.pdf file)
  - Flowchart sketch (.pdf file)

Background

Supercomputers have *exponentially* increased their computational speed at which floating-point operations (FLOP) take place. One FLOP is equivalent to one addition, subtraction, multiplication, or division of two decimal numbers. Today, we will be attempting to use *linear* least squares fitting to model the computational capacity of the fastest supercomputers over the last 30 years. All the data presented is from the TOP500 database, which tracks the top 500 supercomputers every 2 years (`https://www.top500.org/statistics/perfdevel/` and `https://ourworldindata.org/grapher/supercomputer-power-flops`).

The goal of this coding challenge is to use the built-in Matlab functions `polyfit` and `polyval` to fit a **line** through the data provided. The data provided is in gigaFLOPS, or 10^9 floating-point operations per second (GFLOPS). The rate of growth is expected to be proportional to an exponential function (i.e., of the form $f(x) = a_1 p^{a_0 x}$, where a_1 (pre-exponential factor) and a_0 are the fitting constants and p is the base value). You will most commonly find base values of 2, 10, or e ($e \approx 2.71828$, this is the same e as in the natural logarithm).

Step 1: Read in the data set

Download the data set from Coding Challenge 3 Assignment in Canvas, named **supercomputer-power-flops.csv**. Use any Matlab function of your choosing to read in the data set (you may **not** use the Import Data Tool). **Subtract off 1990 from the time column. This allows us to consider "time since 1990" as the measured quantity of time.** Use the `scatter` function to plot GFLOPS versus years since 1990. Notice that it's very difficult to see variations in the early years.

You can use a logarithmic axis in Matlab by the line `set(gca, 'YScale', 'log')`. This should allow you to see a linear trend in the data.

In Canvas quiz, enter what the base value of the exponential term (p) would be for the line shown in the scatter plot after YScale is set to log.

Step 2: Repose the exponential growth problem

Repose the exponential growth problem ($f(x) = a_1 p^{a_0 x}$) using logarithms (with a convenient base of your choosing) to write a linear equation of the form $g(x) = A + Bx$.

HINT: Recall properties of logarithms, i.e., $\log_a x^n = n \cdot \log_a x$, $\log_a b = \frac{\log_c b}{\log_c a}$, etc.

Carry through your conversion of the provided data by taking your chosen logarithm base of the GLOPS data. Create a new scatter plot of these data with the standard, linear scale. Does this plot qualitatively agree with the previous plot with a logarithmic axis?

In Canvas quiz, select the equation that represents $g(x)$ using a generic p value, i.e., do not choose a particular value. Note, there are infinitely many solutions as you can choose p to be anything.

Step 3: Find the line of best fit

Using the `polyfit` function in Matlab, determine the two fitting parameters of the best fit line through the data (note this should be performed on $g(x)$, the logarithmic version from Step 2).

These values will change based on your chosen logarithmic base, p. Add text to your plot using the `text` command in Matlab. The syntax is `text(x,y,string)` where x and y are the actual coordinates on the Matlab figure. Remember that your y-axis is in log scale. There are many ways to create the string in Matlab, but here are two. The example code below assumes a_0 is saved in a variable a0, and a_1 is saved in a variable a1:

- Create a "character array" as you would a vector:
  `string = [ 'a_0 = ' num2str(a0) ', a_1 = ' num2str(a1)];` , note the function `num2str()` which turns a floating point number into a string. The syntax of forming a character array necessitates spaces after the apostrophes.
- Use `sprintf` command:
  `string = sprintf('a_0 = %2.4f, a_1 = %2.4f', a0, a1);`, note "%2.4f" are formatted outputs, more information can be found at `https://www.mathworks.com/help/matlab/ref/sprintf.html` under "Input arguments" and `formatSpec`.

In Canvas quiz, enter in the projected GFLOPS of the year 1990 and 2025 (2 separate questions).

Step 4: Extrapolate the fit and determine the 95% confidence interval

Use the `polyval` command to extrapolate the line of best fit for GFLOPS for the years 1985 until 2030. Compute the error associated with the fit using the `polyval` command's output options, i.e., `[extrap_fit,delta] = polyval(polycoeff,extrap_years,S)`, where `delta` is the error, `polycoeff` is the coefficient vector output from `polyfit`, and `S` is the error structure output from `polyfit`.

Plot the extrapolated line of best fit and plot lines of 95% confidence intervals ($\pm 2\sigma$). Note that with the log-scale axes, the error bars are *actually* exponential curves as well.

In Canvas quiz, enter in the difference between the upper and lower error bounds of GFLOPS for the years 1990 and 2025 (2 separate questions). Note: do **not** assume that $\pm 2\sigma$ in "log-space" is the same in "linear-space".

Reflection Question

Of the two predicted GFLOP values, specifically 1990 and 2025, which would you be more likely to trust? **Why is that?**

Please write out the answers to these questions in the comments of your Matlab script. This should be about 1 paragraph in length.

Coding Rubric

	Excellent (100%)	Above Average (80%)	Average (70%)	Below Average (50%)
Requirements and Delivery (2pts)	• Completed 90-100% of the requirements • Delivered on time, and in correct format.	• Completed 80-90% of the requirements • Delivered on time, and in correct format.	• Completed 70-80% of the requirements • Delivered on time, and in correct format.	• Completed <70% of the requirements • Delivered on time, but not in correct format.
Coding Standards (2pts)	• Includes full header* • Excellent use of variables (no global or unambiguous variable naming)	• Includes full header* • Good use of variables (1-2 global or ambiguous variable naming)	• Includes incomp. header* • Fine use of variables (3-5 global or ambiguous variable naming)	• No header • Poor use of variables (many global or ambiguous variable naming)
Documentation (2pts) Comment your code	• Clearly documented • Specific purpose noted for each function and/or section	• Well documented • Specific purpose noted for each function and/or section	• Some documentation • Purpose noted for each function and/or section	• Limited to no documentation
Runtime (1pt)	• Executes quickly, without errors	• Executes without errors, over 1 min runtime	• Executes with warnings/errors	• Does not execute
Efficiency (2pts)	• Easy to understand, and maintain	• Logical, without sacrificing readability and understanding	• Difficult to follow	• Difficult to follow, huge and appears patched together
Figure Quality (2pts)	• Easy to understand, labels and legend present	• Easy to understand, lacks labels and legend	• Difficult to understand, labels and legend present	• Difficult to understand, lacks labels and legend
Reflection Questions (4pts)	• Easy to understand, fully thought out	• Easy to understand, mostly thought out	• Hard to understand, effort made	• Hard to understand, lacking effort

* Header includes author name(s), assignment title, purpose, creation date, revisions (applicable to **group** projects only)