ECE-210-B HW2

Instructor: Jonathan Lam

Spring 2021

This homework will review numerical integration and differentiation using vectorized operations, some plotting operations. Remember the style guidelines from the previous assignment. For each question, either save the result to a variable or print out the result to the screen (don't suppress the result).

1. Now it's your turn to try out numerical integration and differentiation.

   (a) Create two vectors of length 100 and 1000, each containing evenly spaced samples of the function $g(t) = (1+t)^{-1}$ for $0 \leq t \leq 2\pi$.

   (b) Approximate the derivatives of the vectors using the difference quotient method (i.e., use `diff` to take the difference between adjacent verbs and divide elementwise by Δt). (This will give you vectors of length 99 and 999). Call these numerical estimates $\hat{g}'(t)$.

   (c) Find the analytical derivative $g'(t)$ by hand, and evaluate g' over the same intervals of t. (This will give you vectors of length 100 and 1000; truncate them to their first 99 and 999 elements, respectively).

   (d) Say we define the error of the estimate of the derivative to be a normalized mean-square error:

   $$\epsilon(g', \hat{g}') = \frac{1}{N} \sum_t (g'(t) - \hat{g}'(t))^2$$

   where N is the number of samples (99 or 999). Calculate the error for both estimates for the derivative – which is smaller?

   (e) Approximate the integrals of your original vectors using `cumtrapz` and `cumsum`. This will give you four approximations of the integral of g. These will be length 100 and 1000, so you do not have to perform truncation. Repeat the steps in parts (c) and (d): find the analytical antiderivative, evaluate it at the same t points, and calculate the error estimates for each of the four estimates of the integral.

   (f) Plot the best estimate for the integral. Title your plot.

   (g) *(Optional)* Explore the plotting API: Give the horizontal and vertical axes a label. Turn the grid on/off. Change the axis ticks. Subplots! This is useful for writing reports for your classes, and these plotting functions are closely mirrored by Python's matplotlib library.

   (h) *(Optional)* Integrate using Simpson's rule and compare results.

2. Perform the following matrix operations (without `for` loops). Save each result to a separate variable (i.e., don't alter A after creating it).

   (a) Generate the matrix:

   $$A = \begin{bmatrix} 10^0 & 10^1 & \dots & 10^4 \\ 10^5 & 10^6 & \dots & 10^9 \\ \vdots & \vdots & \ddots & \vdots \\ 10^{45} & 10^{46} & \dots & 10^{49} \end{bmatrix} \in M_{10\times 5}(\mathbb{R})$$

   (Hint: use `logspace` and `reshape`.)

   (b) Flip the third row of A left to right.

   (c) Create a column vector of the geometric means of each row. (Recall that the geometric mean of $x_1, x_2, \dots, x_n$ is $\sqrt[n]{x_1 x_2 \dots x_n}$. The `prod` function will probably be helpful.)

   (d) Create the submatrix $B \in M_{3\times 3}(\mathbb{R})$ such that $b_{ij} = a_{(i+5)(j+1)}$.

   (e) Delete rows 5-10 of A. (Do this in at most five operations.)

3. Create a matrix $C \in M_{1000\times 1000}(\mathbb{R})$ such that $c_{ij} = 2i^4/(3j+2)$, using each of the following methods. Time each method using `tic`/`toc` and write a comment on whether you think the times make sense.

   (a) Using `for` loops and no pre-allocation.

   (b) Using `for` loops and pre-allocation.

   (c) Using (`meshgrid` or `repmat`) and elementwise matrix operations.

   (d) Using broadcasting.

   (*Caution!* Make sure you clear the variables before running this, especially the matrices in (a) and (b) to show the effect of allocation. E.g., if the four matrices created in this section are called `C1`, `C2`, `C3`, and `C4`, make sure to do something like `clear C*` before running your code for this question. If you don't clear your variables and you re-run your code, then you will have pre-allocated the matrix for part (a).)

4. *(Optional)* Try these problems using Python and numpy.

5. *(Optional)* How does vectorization work (in MATLAB or in general)? Do a little research and write a few sentences explaining in your own words how it achieves speedup over `for` loops.