Numerical Analysis

Final task

Submission date: 17/2/2023 13:00

This task is individual. No collaboration is allowed. Plagiarism will be checked and will not be tolerated.

The programming language for this task is Python 3.7. You can use standard libraries coming with Anaconda distribution. In particular limited use of numpy and pytorch is allowed and highly encouraged.

Comments within the Python templates of the assignment code are an integral part of the assignment instructions.

You should not use those parts of the libraries that implement numerical methods taught in this course. This includes, for example, finding roots and intersections of functions, interpolation, integration, matrix decomposition, eigenvectors, solving linear systems, etc.

The use of the following methods in the submitted code must be clearly announced in the beginning of the explanation of each assignment where it is used and will result in reduction of points:

numpy.linalg.solve (15% of the assignment score)

(not studied in class) numpy.linalg.cholesky, torch.cholesky, linalg.qr, torch.qr (1% of the assignment score)

numpy.*.polyfit, numpy.*.*fit (40% of the assignment score)

numpy.*.interpolate, torch.*.interpolate (60% of the assignment score)

numpy.*.roots (30% of the assignment 2 score and 15% of the assignment 3 score)

All numeric differentiation functions are allowed (including gradients, and the gradient descent algorithm).

Additional functions and penalties may be allowed according to the task forum.

You must not use reflection (self-modifying code).

Attached are mockups of for 4 assignments where you need to add your code implementing the relevant functions. You can add classes and auxiliary methods as needed. Unittests found within the assignment files must pass before submission. You can add any number of additional unittests to ensure correctness of your implementation.

In addition, attached are two supplementary python modules. You can use them but you cannot change them.

Upon the completion of the final task, you should submit the four assignment files and this document with answers to the theoretical questions archived together in a file named <your ID>.zip

All assignments will be graded according to **accuracy** of the numerical solutions and **running time**.

Expect that the assignment will be tested on various combinations of the arguments including function, ranges, target errors, and target time. We advise to use the functions listed below as test cases and benchmarks. At least half of the test functions will be polynomials. Functions 3,8,10,11 will account for at most 4% of the test cases. All test functions are continuous in the given range. If no range is given the function is continuous in $[-\infty, +\infty]$.

1. $f_1(x) = 5$
2. $f_2(x) = x^2 - 3x + 5$
3. $f_3(x) = \sin(x^2)$
4. $f_4(x) = e^{-2x^2}$
5. $f_5(x) = \arctan(x)$
6. $f_6(x) = \frac{\sin(x)}{x}$
7. $f_7(x) = \frac{1}{\ln(x)}$
8. $f_8(x) = e^{ex}$
9. $f_9(x) = \ln(\ln(x))$
10. $f_{10}(x) = \sin(\ln(x))$
11. $f_{11}(x) = 2^{\frac{1}{x^2}} * \sin\left(\frac{1}{x}\right)$
12. For Assignment 4 see sampleFunction.*

Assignment 1 (14pt):

Check comments in Assignment1.py.

Implement the function **Assignment1.interpolate(..)**.

The function will receive a function f, a range, and a number of points to use.

The function will return another "interpolated" function g. During testing, g will be called with various floats x to test for the interpolation errors.

<u>Grading policy (10pt):</u>

Running time complexity > O(n^2): 0-20%

Running time complexity = O(n^2): 20-80%

Running time complexity = O(n): 50-100%

The grade within the above ranges is a function of the average absolute error of the interpolation function at random test points. Correctly implemented linear splines will give you 50% of the assignment value.

Solutions will be tested with $n \in \{1,10,20,50,100\}$ on variety of functions at least half of which are polynomials of various degrees with coefficients ranging in $[-1,1]$.

Question 1.1: Explain the key points in your implementation (4pt).

שאלה זו עסקה באינטרפולציה. את השאלה פתרתי באמצעות שיטת בזייה כפי שנלמדה בכיתה. דגמתי נקודות במרווחים שווים מהפונקציה אשר קיבלתי, לאחר מכן יצרתי וקטור של נקודות שאותו אכפיל בהמשך במטריצת המקדמים אשר יצרתי גם כן. באמצעות פעולה זו מצאתי את נקודות הקונטרול בכל טווח של זוג נקודות מהדגימה. לאחר מכן אני מחשב אץ עקומות בזייה עם נקודות אלה. לבסוף אני מציב את ערכי האיקס ובכך מקבל את הערכי הוואי, ובכך אחזיר את הפונקציה הקרובה ביותר לפונקציה המקורית.

Assignment 2 (14pt):

Check comments in Assignment2.py.

Implement the function **Assignment2.intersections(..)**.

The function will receive 2 functions- f_1, f_2, and a float maxerr.

The function will return an iterable of approximate intersection Xs, such that:

$$\forall x \in X, |f_1(x) - f_2(x)| < maxerr$$

<u>Grading policy (10pt):</u> The grade will be affected by the number of correct/incorrect intersection points found and the running time of **Assignment2.intersections(..)**.

Question 2.1: Explain the key points in your implementation (4pt).

על מנת לפתור את שאלה זו השתמשתי בשיטה אשר משלבת בין bisection לsecant. החיסרון של שיטה זו הוא שהיא לא מוצאת נקודות חיתוך שהן גם קיצון ובשביל כך מצאתי את נקודות הקיצון (נקודות חיתוך בפונקציית הנגזרת), במהלך הבדיקה אני אבדוק האם ערך הy של נקודת הקיצון שמצאתי הוא אפס עד כדי טווח טעות.

Assignment 3 (36pt):

Check comments in Assignment3.py.

Implement a function **Assignment3.integrate(…)** and **Assignment3.areabetween(..)** and answer two theoretical questions.

Assignment3.integrate(…) receives a function f, a range, and several points to use.

It must return approximation to the integral of the function f in the given range.

You may call f at most n times.

<u>Grading policy (10pt):</u> The grade is affected by the integration error only, provided reasonable running time e.g., no more than 5 minutes for n=100.

Question 3.1: Explain the key points in your implementation of Assignment3.integrate(…). (4pt)

בשאלה זו השתמשתי באלגוריתם Gaussian quadrature על מנת למצוא אינטרגרציה של הפונקציה בהינתן תחום מסויים.

Assignment3.areabetween(..) receives two functions f_1, f_2 .

It must return the area between f_1, f_2 .

In order to correctly solve this assignment you will have to find all intersection points between the two functions. You may ignore all intersection points outside the range $x \in [1,100]$.

Note: there is no such thing as negative "area".

<u>Grading policy (10pt):</u> The assignment will be graded according to the integration error and running time.

Question 3.2: Explain the key points in your implementation of Assignment3. areabetween (…) (4pt).

על מנת לפתור את השאלה הזאת מצאתי את נקודות החיתוך בין שתי הפונקציות באמצעות הפתרון לשאלה 2. לאחר מכן רצתי על מערך הנקודות ומצאתי את השטח בין כל שתי נקודות חיתוך (נשים לב כי אם f(x) נמצאת מעל g(x) וחישבנו ש g(x) מעל f(x) אזי נקבל את אותו השטח רק בסימן שלילי ולכן ניקח ערך מוחלט של השטח).

Question 3.3: Explain why is the function $2^{\frac{1}{x^2}} * \sin\left(\frac{1}{x}\right)$ is difficult for numeric integration with equally spaced points? (4pt)

קשה לחשב את הפונקציה הזו בגלל היותה מחזורית שעולה ויורדת באופן צפוף בקטע [0.159, -0.159] ולכן אנו נאבד קטע אחד (ואפילו מספר רב של קטעים) וכתוצאה מכך יהיה קשה לדייק ברמת שגיאה שהיא סבירה וככל הנראה תהיה שגיאה משמעותית מאוד.

Question 3.4: What is the maximal integration error of the $2^{\frac{1}{x^2}} * \sin\left(\frac{1}{x}\right)$ in the range [0.1, 10]? Explain. (4pt)

הטעות המקסימלית תהיה כאשר לא נחלק את הקטע למקטעים ונחשב אותו פשוט באמצעות סימפסון עבור מקטע אחד בקטע הנתון. כאשר נחשב את השטח בשיטה הזו הטעות המקסימלית תהיה $-\frac{h}{90} \cdot f^{(4)}(\epsilon)$ כאשר $\epsilon = 0.1$. נשים לב כי לא משנה כמה פעמים נבצע גזירה לפונקציה לבסוף נשאר עם רכיב מהצורה $2^{\frac{1}{x^2}+c}$ כאשר בנוסף אנו יודעים כי $\left(\sin\left(\frac{1}{x^2}\right)\right)' = -\frac{\cos\left(\frac{1}{x^2}\right)}{x^2}$ מכיוון שsin וcos חסומים בין הערכים [1,-1] נתעלם מהם במהלך חישוב הנגזרת. מכאן נובע כי הטעות המקסימלית תהיה בסדר גודל (כאשר מדובר על קירוב אסימפטוטי של $\epsilon = 0.1$):

$$\left(2^{\frac{1}{x^2}} \cdot \sin\left(\frac{1}{x}\right)\right)^{(4)} = \left(\frac{2^{\frac{1}{x^2}}}{x^3}\right)^{(3)} = \left(-\frac{2^{\frac{1}{x^2}}}{x^4}\right)^{(2)} \left(\frac{2^{\frac{1}{x^2}}}{x^5}\right)^{(1)} = -\frac{2^{\frac{1}{x^2}}}{x^6}$$

כאשר נציב $\epsilon = 0.1$ נקבל שגיאה מסדר גודל של $-\frac{(0.1)^2}{90} \cdot -\frac{2^{100}}{0.1^6} = \frac{2^{100}}{90 \cdot 0.1^4} = 1.4 \cdot 10^{32}$
כלומר השגיאה שקיבלנו שואפת לאינסוף (מספר מאוד מאוד גדול!)

Assignment 4 (14pt)

Check comments in Assignment4.py.

Implement the function **Assignment4.fit(…)**

The function will receive an input function that returns noisy results. The noise is normally distributed.

Assignment4A.fit should return a function g fitting the data sampled from the noisy function. Use least squares fitting such that g will exactly match the clean (not noisy) version of the given function.

To aid in the fitting process the arguments a and b signify the range of the sampling. The argument d is the expected degree of a polynomial that would match the clean (not noisy) version of the given function.

You have no constrains on the number of invocation of the noisy function but the maximal running time is limited. Additional parameter to **Assignment4.fit** is maxtime representing the maximum allowed runtime of the function, if the function will execute more than the given amount of time, the grade will be significantly reduced.

<u>Grading policy (10pt):</u> the grade is affected by the error between g (that you return) and the clean (not noisy) version of the given function, much like in Assignment1. 65% of the test cases for grading will be polynomials with degree up to 3, with the correct degree specified by d. 30% will be polynomials of degrees 4-12, with the correct degree specified by d. 5% will be non-polynomials

Question 4.1: Explain the key points in your implementation. (4pt)

דגמתי מערכי הפונקציה שקיבלתי ערכי איקס בכמות אשר שווה ל4 פעמים maxtime. לאחר מכן חישבתי לכל ערך של איקס את ערך הy שלו. לבסוף פתרתי מערכת משוואות ליניאריות בעזרת least squares והחזרתי פונקציה לאחר שמצאתי את ערכי הy בהתאם לפולינום המקורי שקיבלתי.

Assignment 5 (27pt).

Check comments in Assignment5.py.

Implement the function **Assignment5.area(…)**

The function will receive a shape contour and should return the approximate area of the shape. Contour can be sampled by calling with the desired number of points on the contour as an argument. The points are roughly equally spaced.

Naturally, the more points you request from the contour the more accurately you can compute the area. Your error will converge to zero for large n. You can assume that 10,000 points are sufficient to precisely compute the shape area. Your challenge is stopping earlier than according to the desired error in order to save running time.

<u>Grading policy (9pt):</u> the grade is affected by your running time.

Question 4B.1: Explain the key points in your implementation. (4pt)

בשאלה זו השתמשתי בשיטת shoelace על מנת לחשב את השטח, שלחתי לשיטה שאני רוצה שהיא תדגום לי 10000 נקודות מהcontour אשר היא מקבלת ותשמש בנוסחה של shoelace על מנת לסכום את השטח שמתחת לפוליגון.

Implement the function **Assignment4.fit_shape(…)** and the class **MyShape**

The function will receive a generator (a function that when called), will return a point (tuple) (x,y), a that is close to the shape contour.

Assume the sampling method might be noisy- meaning there might be errors in the sampling.

The function will return an object which extends **AbstractShape**
When calling the function **AbstractShape.contour(n)**, the return value should be array of n equally spaced points (tuples of x,y).

Additional parameter to **Assignment4.fit_shape** is maxtime representing the maximum allowed runtime of the function, if the function will execute more than the given amount of time, the grade will be significantly reduced.

In this assignment only, you may use any numeric optimization libraries and tools. Reflection is not allowed.

<u>Grading policy (10pt):</u> the grade is affected by the error of the area function of the shape returned by Assignment4.fit_shape.

Question 4B.2: Explain the key points in your implementation. (4pt)

אני יוצר מערך בגודל X אשר מכיל את הנקודות (x,y). יצרית המערך מתבסס על המשתנה maxtime.על מנת להפחית מן הרעש שיש על הצורה אני משתמש בעלגוריתם K-means עם Z מרכזים ולאחר מכן אני ממיין את

המרכזים אשר אני מוצא לפי מרחק ממרכז אחד לקרוב אליו על מנת לקבל צורה משוערת. לאחר הפחתה של הרעש במידה הרבה ביותר אני מחשב את שטח הצורה כמו שחישבתי בשאלה הקודמת.