**Numerical Analysis**

**Final task**

Submission date: 22/2/22 23:59 (strict).

This task is individual. No collaboration is allowed. Plagiarism 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.

**You should not use those parts of the libraries that implement numerical methods taught in this course** (unless explicitly stated otherwise in the instructions of the particular assignment)**.** This restriction 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 deduction of points. Failure to announce the use of any restricted functions will result in disqualification of the assignment.

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

(not studied in class) numpy.linalg.cholesky, torch.cholesky, linalg.qr, torch.qr (10% 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)

numeric differentiation functions are allowed!

numpy.linalg.inv, scipy.linalg.inv, torch.inverse, and all other external libraries for matrix inversion (20% of the assignment score)

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

**You must not use reflection (self-modifying or self-inspecting 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. BUT! existing unit tests are provided for demonstration and to encourage you to write additional tests as you go. You can add any number of additional unittests to ensure correctness of your implementation. Passing only the existing unittests does not ensure that your code will not fail in all cases. It is your responsibility to test your code and ensure that it is stable. You should add 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 five assignment files and this document with answers to the theoretical questions. The archive should not contain folders, but only the submission files!

Assignments will be graded according to **error** of the numerical solutions and **running time**. Some assignments have required specific error bounds – they will be graded according to running time. Some assignments limit the running time – they will be graded according to error. For all executions there is 2 minutes running time cap after which the execution will be halted.

Every assignment will be AUTOMATICALLY tested on a number of different functions and different parameters. It may be executed multiple times on the same function with the same parameters. Every execution will start with a clean memory. Any exception throwed during an execution will render the execution invalid and nullify its contribution to the grade. **Test your code!!!**

Any disqualification of an assignment (e.g. due to unannounced use of restricted functions) or an execution (e.g. due to exception) will not contribute to the grade regardless the effort put in the development.

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 – add additional unittests with implementations of these functions. At least half of the test functions will be polynomials. Functions 3,8,10,11 will account for at most 5% of the test cases. All test functions are continuous in the given range. If no range is given the function is continuous in .

1. For Assignment 4 see sampleFunction.\*

**Assignment 1 (14pt):**

**(10pt)** Implement the function **Assignment1.interpolate(..)** following the pydoc instructions.

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.

Grading policy:

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

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

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

Running time complexity will be measured empirically as a function of n.

The grade within the above ranges is a function of the average relative 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 on variety of functions at least half of which are polynomials of various degrees with coefficients ranging in .

**Restricted functions I used:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

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

|  |
| --- |
| דגמתי n נקודות במרחקים שווים בטווח הנתון ( (a,bמהפונקציה, לאחר מכן חישבתי את ה-control points עבור כל שתי נקודות סמוכות והכנסתי את ה-4 נקודות שנוצרו לפונקציית בזייר, שהחזירה לי פונקציה שמקבלת t בין 0 ל1 ומחזירה נקודה (x,y). יצרתי מערך של פונקציות בזייר עבור כל 2 נקודות סמוכות והcontrol points- המתאימים, בסופו של דבר יצרתי פונקציה חדשה שמקבלת ערך x כלשהו, מוצאת את 2 הנקודות שביניהם אותו ה-x נמצא ולפי כך מוצאת את הפונקציית בזייר המתאימה. בפונקציית הבזייר שחזרה נציב t המהווה יחס של מיקום האיקס ביחס לקטע ונקבל כך נקודה, נחלץ מהנקודה את ערך ה-y המתאים ונחזיר אותו. |

**Assignment 2 (14pt):**

**(10pt)** Implement the function **Assignment2.intersections(..)** following the pydoc instructions.

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

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

Grading policy: The grade will be affected by the number of correct and incorrect intersection points found, the running time of **itr =** **Assignment2.intersections(..)** followed by **list(itr).**

**Restricted functions I used:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

**(4pt) Question 2.1:** Explain the key points in your implementation in particular explain how did you address the problem of finding multiple roots.

|  |
| --- |
| חילקתי את הטווח הנתון (a,b) ל-n נקודות במרחקים שווים, קבעתי שלכל מקטע באורך 1 יהיו 50 נקודות כלומר, הקפיצות בין נקודה לנקודה שאני דוגם יהיו במרחק של 0.02 אחד משני. לאחר החלוקה, השתמשתי באלגוריתם של Regula falsi שלמדנו בכיתה, על פונקציית ההפרש של f1 וf2 ,מאחר שנקודות חיתוך של פונקציית ההפרש מהווה נקודת חיתוך (intersection) של הפונקציות. עברתי על כל שתי נקודות סמוכות ובדקתי האם הערך שלהם בפונקציית ההפרש הוא 0 והוספתי אותו למערך. בנוסף, בדקתי אם הסימנים מנוגדים, אם כן הפעלתי Regula falsi והוספתי את הנקודה למערך, אם לא אז המשכתי לבדוק במקטע הבא. לבסוף עברתי על כל מערך הנקודות ויצרתי מערך חדש שאליו הכנסתי נקודות שהמרחקים ביניהם חייבים להיות גדולים מ-maxerr והחזרתי מערך זה. |

**Assignment 3 (31pt):**

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

**(5pt) Assignment3.integrate(…)** receives a function f, a range, and a number of points n.

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

You may call f at most n times.

Grading policy: The grade is affected by the integration error only, provided reasonable running time e.g., no more than 2 minutes for n=100.

**Restricted functions I used:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

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

|  |
| --- |
| ניסיתי לקרוא ולבדוק באינטרנט על קירובים נומריים של אינטגרלים שונים, קראתי על השיטות השונות של Newton closed cotes ושמתי לב שהאלגוריתם של Composite Simpson הוא האלגוריתם המומלץ ביותר לפי אתרים שונים באינטרנט, ולכן החלטתי לממש אותו. האלגוריתם של הקירוב מתואר בנוסחה הבאה:  כאשר h מהווה גודל כל אינטרוול /(n-1)(b-a) ו-n מהווה את כמות הנקודות שמשתמשים בהם לקירוב של האינטגרל. |

**(10pt) Assignment3.areabetween(..)** receives two functions .

It must return the area between .

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 .

Note: there is no such thing as negative “area”.

Grading policy: The assignment will be graded according to the integration error and running time.

**Restricted functions I used:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

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

|  |
| --- |
| על מנת לחשב שטח בתחום נתון שכלוא בין 2 פונקציות, עלינו תחילה למצוא את כל נקודות החיתוך של הפונקציות בעזרת שאלה 2 שפתרנו ובין כל 2 נקודות חיתוך כאלה נחשב את השטח באופן הבא:  ראשית, נשים לב שאין דבר כזה שטח שלילי והשטח של כל מקטע כזה (השטח שכלוא בין 2 הפונקציות בין 2 נקודות חיתוך סמוכות) הוא האינטגרל של הפונקציה הבאה: g(x)=|f1(x)-f2(x)|. נקרא לקירוב שעשינו בסעיף הקודם של אינטגרל ונסכום את כל השטחים של כל המקטעים הנ"ל בתחום הנתון שבין 1 ל100. |

**(4pt) Question 3.3:** Explain why is the function is difficult for numeric integration with equally spaced points?

|  |
| --- |
| נשים לב שהפונקציה הזאת בעלת שיפועים מאוד תלולים קרוב ל0 דבר אשר גורם לפונקציה לגדול בצורה דרסטית ובמהירות, אם נשתמש באינטגרל בעל נקודות במרחקים שווים אנחנו עלולים לפספס את הנקודות הקריטיות של הפונקציה בעלי הערכים הקיצוניים ולא להתחשב בהם. למשל אם נרצה לחשב את השטח של האינטגרל הנ"ל בין 0.1 ל-10 ונחלק את הקטע לנקודות שוות אנחנו עלולים לדגום יותר נקודות מהערכים הקטנים של הפונקציה, ערכים שכבר מתחילים להתכנס ל0 למשל האינטרוול [0.5,10] מאשר הערכים הקריטייםשנמצאים בטווח [0.1,0.5] שמשפיעים בצורה דרסטית יותר מאשר האינטרוול האחר שבו גם ידגמו יותר נקודות עקב הגודל שלו. |

**(4pt) Question 3.4:** What is the maximal integration error of the in the range [0.1, 10]? Explain.

|  |
| --- |
| השתמשתי באלגוריתם של Composite Simpsonעל מנת לחשב את הקירוב של השטח, לכן אני אחשב את החסם העליון של השגיאה בעזרת הנוסחה הנתונה של השגיאה של Composite Simpson:  כאשר M הוא הערך המקסימלי של הנגזרת הרביעית בקטע הנתון [0.1,10].  השתמשתי ב-Desmos על מנת לחשב ערך זה וקיבלתי כי :  נשים לב שאנחנו משתמשים באלגוריתם של Composite Simpson לכן, נקבל את השגיאה המקסימלית עבור n=2.  התחום הנתון הוא [0.1,10] אז אנחנו יודעים מכך ש-a=0.1 ו-b=10 מהצבה של ערכים אלו במשוואה שלמעלה נקבל:  לסיכום מהנוסחה של השגיאה המקסימאלית של האלגוריתם קיבלנו כי, השגיאה המקסימלית בתחום הנתון של האלגוריתם הנ"ל היא : |

**Assignment 4 (14pt)**

**(10pt)** Implement the function **Assignment4.fit(…)** following the pydoc instructions.

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

Assignment4.fit should return a function fitting the data sampled from the noisy function. Use least squares fitting such that will exactly match the clean (not noisy) version of the given function.

To aid in the fitting process the arguments and signify the range of the sampling. The argument 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 invocations of the noisy function but the maximal running time is limited. Invocation of f may take some time but will never take longer than 0.5 sec.

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 execution will not contribute to the grade causing significant deduction. You should consider the risk of failure vs gains in accuracy when you get close to the time limit.

Grading policy: the grade is affected by the error between (that you return) and the clean (not noisy) version of the given function, much like in Assignment1. 60% of the test cases for grading will be polynomials with degree up to 3, with the correct degree specified by . 30% will be polynomials of degrees 4-12, with the correct degree specified by . 10% will be non-polynomials with random .

**Restricted functions I used:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

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

|  |
| --- |
| בשאלה זו השתמשתי בנוסחה של Least Squares. על מנת לדאוג לכך שהפונקציה המותאמת תהיה בעלת שגיאה מינימלית, גזרתי את הנוסחה והשוואתי אותה ל-0 על מנת למצוא קיצון מסוג מינימום. כמו שעשינו בתרגול במקרה פרטי יותר, לאחר הכללה קיבלתי את המשוואה הליניארית הבאה:    מצאתי את הווקטור a0,a1,...,ak בהסתמך על דגימת n נקודות כלשהם מהפונקציה הנתונה. הוקטור שמצאתי הוא וקטור שמייצג את הפולינום של קירוב הפונקציה והחזרתי פונקציה שמייצגת את הפולינום הנ"ל. k מייצג במטריצה זו את הדרגה המשוערת של הפונקציה הנתונה. פתרתי מטריצה זו על פי שיטת LU-DECOMPOSITION. |

**Assignment 5 (27pt).**

**(9pt)** Implement the function **Assignment5.area(…)** following the pydoc instructions.

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 . You can assume that 10,000 points are sufficient to precisely compute the shape area. Your challenge is stopping earlier than that according to the desired error in order to save running time.

Grading policy: the grade is affected by your running time.

**Restricted functions I used:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

**(4pt) Question 5.1:** Explain the key points in your implementation.

|  |
| --- |
| בשאלה זו חישבתי את השטח באמצעות יצירת טרפזים עם ציר ה-x, בהתחלה אני מגדיר שיש 200 נקודות שמהם אני צריך לחשב את השטח. אני דוגם באמצעות contour את הנקודות הנ"ל ומחשב את השטח שנוצר מהנקודות הנ"ל. לאחר מכן אני קורא בקריאה רקורסיבית לחישוב מחדש עם כמות נקודות כפולה, אני בודק אם השגיאה היחסית קטנה מ-maxerr ואם כן אני מחזיר את הסכום האחרון שדגמתי אחרת, אני מעדכן את הסכום הישן להיות הסכום העכשווי ומריץ את האלגוריתם פעם נוספת באופן רקורסיבי. |

**(10pt)** Implement the function **Assignment5.fit\_shape(…)** and the class **MyShape** following the pydoc instructions.

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 should 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). When calling the function **AbstractShape.area()**, the return value should be the area of the shape. You may use your solution to **Assignment5.area** to implement the area function.

Additional parameter to **Assignment5.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 execution will be halted.

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

Grading policy: the grade is affected by the error of the area function of the shape returned by Assignment4.fit\_shape.

**There are no restricted functions. The use of any library is allowed.**

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

|  |
| --- |
| תחילה, בדקתי כמה זמן לוקח לדגום נקודה ואז בדקתי כמה נקודות אני יכול לדגום ב-40 אחוז מהזמן המותר (maxtime). אם ראיתי שאפשר לדגום יותר מ-10,000 נקודות הסתפקתי ב-10,000 כדי לא להגזים עם כמות הנקודות בדגימה. לאחר הדגימה, מיינתי את הנקודות לפי אלגוריתם מיון שנקרא clockwise, אלגוריתם זה ממיין את הנקודות עם כיוון השעון ונותן חשיבות גם למרחק מהמרכז של הצורה. מצאתי את הנקודה המרכזית של הצורה ונרמלתי את הנקודות במיון לנקודה הזו וע"י כך הפחתתי את ה-"רעש".  החזקתי באחד השדות של הצורה MyShape, את ערכי נקודות הדגימה האלו לאחר המיון.  פונקציית ה-area של MyShape מעבירה את הקריאה לפונקציית ה-area של הסעיף הקודם עם פונקצית ה-contour של הצורה החדשה. את פונקציית ה-contour מימשתי באופן הבא:  הפונקציה מקבלת כפרמטר את מספר הדגימות שרוצים לבצע וע"פ מספר זה ידעתי כמה נקודות עליי להחזיר מהנקודות שיש לצורה בשדה samples וזה מה שעשיתי, כמובן תוך כדי בדיקת מקרי קצה של חריגה ממערך וכולי. |