Assignment 5

Due Date: April 5, 8:59pm

Please see the guidelines at https://www.cs.toronto.edu/~lczhang/338/homework.html

What to Hand In

Please hand in 2 files:

- Python file containing all your code, named `csc338_a5.py`. If you are using Jupyter Notebook to complete the work, your notebook can be exported as a .py file (File -> Download As -> Python). Your code will be auto-graded using Python 3.6, so please make sure that your code runs. There will be a 20% penalty if you need a remark due to small issues that render your code untestable.
- PDF file named `csc338_a5_written.pdf` containing your solutions to the written parts of the assignment. Your solutions can be hand-written, but must be legible. Graders may deduct marks for illegible or poorly presented solutions.

Submit the assignment on **MarkUs** by 9pm on the due date. See the syllabus for the course policy regarding late assignments. All assignments must be done individually.

Q1. Golden Section Search [12 pt]

Part (a) [8 pt]

Write a function `golden_section_search` that uses the Golden Section search to find a minima of a function `f` on the interval `[a, b]`. You may assume that the function `f` is unimodal on `[a, b]`.

The function should evaluate `f` only as many times as necessary. There will be a penalty for calling `f` more times than necessary.

Refer to algo 6.1 in the textbook, or slide 19 of the posted slides accompanying the textbook.

Part (b) [4 pt]

Consider the following functions, both of which are unimodal on `[0, 2]`

$f_1(x) = 2x^2 - 2x + 3$

$f_2(x) = -xe^{-x^2}$

Run the golden section search on the two functions for $n = 30$ iterations.

Save the returned lists in the variables `golden_f1` and `golden_f2`

Q2. Newton's Method in One Dimension [20 pt]

Part (a) [8 pt]

Implement Newton's Method to find a minima of a real function in one dimension.

Part (b) [4 pt]

Consider `f1` and `f2` from Question 1 Part (b).

Complete the functions `df1`, `df2` which compute the derivatives of `f1` and `f2`.

Complete the functions `ddf1`, `ddf2` which compute the second derivatives of `f1` and `f2`.

Part (c) [4 pt]

Run Newton's Method on the two functions f_1 and f_2 for $n = 30$ iterations, starting at $x = 1$.

Save the returned lists in the variables `newton_f1` and `newton_f2`.

Part (d) [4 pt]

Newton's method should have a quadratic convergence rate. Do you observe a quadratic convergence rate in `newton_f1` and `newton_f2`?

Explain your observation and justification in your PDF writeup.

Q3. Matrix Types [16 pt]

Part (a) [6 pt]

Write a function to check if a symmetric matrix is positive definite, negative definite, or indefinite using the Cholesky factorization. Use the Cholesky Factorization algorithm we discussed in class. You might find your Assignment 3 code to be helpful.

Part (b) [4 pt]

Write a function `matrix_type` to check whether a matrix is positive definite, negative definite, or indefinite. You can use the `is_positive_definite` function you wrote in part (a) as a helper function.

Part (b) [3 pt]

Use the function you wrote in part (b) to determine whether each of the following matrices `M1`, `M2`, and `M3` is positive definite, negative definite, or indefinite.

Store the values corresponding to constants `POS_DEF`, `NEG_DEF`, or `INDEF` in the variables `q3b_M1`, `q3b_M2`, `q3b_M3`.

Part (c) [3 pt]

Determine, **by hand**, whether each of the following matrices `M4`, `M5`, `M6`, is positive definite, negative definite, or indefinite.

Store the values corresponding to constants `POS_DEF`, `NEG_DEF`, or `INDEF` in the variables `q3b_M4`, `q3b_M5`, `q3b_M6`.

Q4. Optimization in Multiple Dimensions [32 pt]

Consider the function

$$f(x_0, x_1) = 2x_0^4 + 3x_1^4 - x_0^2 x_1^2 - x_0^2 - 4x_1^2 + 7$$

Part (a) [4 pt]

Derive the gradient. Include your solution in your PDF writeup.

Part (b) [4 pt]

Derive the Hessian. Include your solution in your PDF writeup.

Part (c) [8 pt]

Write a function `steepest_descent_f` that uses a variation of the steepest descent method to solve for a local minimum of $f(x_0, x_1)$ from part (a). Instead of performing a line search as in Algorithm 6.3, the parameter α will be provided to you as a parameter. Likewise, the initial guess (x_0, x_1) will be provided.

Use `(1, 1)` as the initial value and perform 10 iterations of the steepest descent variant. Save the result in the variable `q4c_steepest`. (The result `q4c_steepest` should be a list of length 11)

Part (d) [8 pt]

Write a function `newton_f` that uses Newton's method to find a local minimum of the function $f(x_0, x_1)$ given an initial value. You may use matrix inversion from `np.linalg`.

Use `(1, 1)` as the initial value and perform 10 iterations of Newton's Method. Save the result in the variable `q4d_newton`. (The result `q4d_newton` should be a list of length 11)

Part (e) [4 pt]

Compare the two algorithms, and the observed convergence rates of e_k of the sequences in parts (c) and (d). Which one converged faster?

Explain your observation and your finding in your PDF writeup.

Part (d) [4 pt]

Without running your programs, find three other local minima of $f(x)$. Hint: notice that the value of $f(x)$ does not depend on the signs of x_0 and x_1.

Include your solution in your PDF writeup.