Instructions: (Please read carefully and follow them!)

Try to solve all problems on your own. If you have difficulties, ask the instructor or TAs.

Please follow the instructions given below to prepare your solution notebooks:

- Please use different notebooks for solving different Exercise problems.
- The notebook name for Exercise 1 should be `ROLLNUMBER-labLL-ex1.ipynb`. `ROLLNUMBER-labLL-ex2.ipynb`, etc for others. 'LL' is the two digit lab number (lab-3 is 03, etc).
- Please ask your doubts to TAs or instructors or post in Moodle Discussion Forum channel.
- You should upload on the .ipynb files on Moodle (one per exercise).

Only the questions marked [**R**] need to be answered on paper. Write legible and to-the-point explanations. The work-sheet on which you write needs to be submitted before leaving the session.

Some other questions require plotting graphs (histograms, trajectories, level-sets etc) or tables. Please make sure that the plots are present in the submitted ipython notebooks.

Submission Time: Please check the submission deadline as show on the assignment web-page in Moodle. Late submissions will be accepted upto 24 hours from the deadline. All late submissions will have a penalty of 3 marks. Submissions later than 24 hours after the deadline will not be accepted.

The third laboratory exercise aims to help you understand line-search and BFGS (a Quasi-Newton method).

Exercise 1[5 marks] Recall that we implemented the gradient descent algorithm to solve $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$. The key components in the gradient descent iterations include the descent direction $\mathbf{p}_k$, which is set to $-\nabla f(\mathbf{x}_k)$, and the step length η_k. Also recall that the step length η_k was constant (η) in the previous lab. In general, it is difficult to guess a good value of η that will result in fast convergence. A more practical solution is to change η in every iteration. This is done through line-search. Algorithm-1 describes gradient descent with inexact line-search.

Algorithm 1 Gradient Descent Procedure with Line Search to compute step length

```
Require: Starting point $x_0$, Stopping tolerance $\tau$
1: Initialize $k = 0$, $p_k = -\nabla f(x_k)$
2: while $\|p_k\|_2 > \tau$ do
3:     Select step-length $\eta_k$ through line-search.
4:     $x_{k+1} \leftarrow x_k + \eta_k p_k$
5:     $p_k \leftarrow -\nabla f(x_k)$
6:     $k \leftarrow k + 1$
7: Output: $x_k$
```

To determine the step length η_k in iteration-k, we will use the following property: Suppose a non-zero $\mathbf{p} \in \mathbb{R}^n$ is a descent direction at point $\mathbf{x}$, and let $\gamma \in (0, 1)$. Then there exists $\varepsilon > 0$ such that

$$f(\mathbf{x} + \alpha\mathbf{p}) \le f(\mathbf{x}) + \gamma\alpha\nabla f(\mathbf{x})^\top \mathbf{p}, \quad \forall\alpha \in (0, \varepsilon].$$

This condition is known as a sufficient decrease condition.

Utilizing the concept of sufficient decrease, the step length η_k can be determined using a backtracking procedure illustrated below to find an appropriate value of ε.

Algorithm 2 Backtracking (Inexact) Line Search

```
Require: $x_k, p_k, \alpha_0, \rho \in (0,1), \gamma \in (0,1)$
1: Initialize $\alpha = \alpha_0, p_k = -\nabla f(x_k)$
2: while $f(x_k + \alpha p_k) > f(x_k) + \gamma\alpha\nabla f(x_k)^\top p_k$ do
3:     $\alpha = \rho\alpha$
4: Output: $\alpha$
```

This is known as approximate (inexact) line search method to find the step length at each iteration.

1. Let $g(x) = g(x_1, x_2) = (x_1 + 49)^2 + (x_2 - 36)^2$. Select the starting point $\mathbf{x}_0 = (100, 100)$ and $\tau = 10^{-10}$, we will investigate the behavior of the backtracking line search algorithm for different choices of α_0. Set $\gamma = \rho = 0.5$ and try $\alpha_0 \in \{1, 0.9, 0.75, 0.6, 0.5, 0.4, 0.25, 0.1, 0.01\}$. For each α_0, record the final minimizer, final objective function value, and the number of iterations taken by the gradient descent algorithm with backtracking line search to terminate. Generate a plot where the number of iterations is plotted against α_0 values. Provide observations on the results, and comment on the minimizers and objective function values obtained for different choices of α_0.
2. **[R]** Check and comment if, for any α_0 value, gradient descent with backtracking line search takes a lesser number of iterations compared to the gradient descent procedure with fixed step-length of previous lab.
3. Plot the level sets of the function $g(\mathbf{x})$ and also plot the trajectory of the optimization on the same plot for both inexact line search method and the fixed step length method of gradient descent algorithm and report your observations.
4. Redo (1) above using the function $f(x) = 256(x_2 - x_1^2)^2 + (2 - x_1)^2$ from Exercise-1 and also keep in mind the answer of the part (2) from Exercise-1. It is not required to report anything on paper for this exercise.

5. **[R]** What do you conclude from (1) and (2) regarding these two line search approaches ?

Exercise 2[5 marks]

Recall that to solve problems of the form $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, the update rule involved in Newton's method is of the form:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \eta^k(\nabla^2 f(\mathbf{x}^k))^{-1}\nabla f(\mathbf{x}^k). \tag{1}$$

Now we will discuss a method which avoids explicit computation of the inverse of Hessian matrix at each iteration, but is nearly efficient as the Newton's method. This method will be called BFGS named after the famous applied Mathematicians Broyden, Fletcher, Goldfarb and Shanno. A brief description can be found at BFGS Theory. **Note:** Please go through the above link as some steps may be needed to implement the algorithm which are not mentioned here in this sheet explicitly.

Algorithm 3 BFGS Algorithm

```
Require: Starting point $x_0$, Stopping tolerance $\tau$
1: Initialize $k = 0, B_0 = ??$                              ▷ Initialize Hessian approximation
2: while $\|\nabla f(x_k)\|_2 > \tau$ do
3:     Compute descent direction: $p_k = -B_k\nabla f(x_k)$
4:     Choose step size: $\alpha_k$ through line-search
5:     Update iterate: $x_{k+1} = x_k + \alpha_k p_k$
6:     Compute new gradient: $s_k = x_{k+1} - x_k$,   $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$
7:     Update Hessian approximation: $B_{k+1} = ??$      ▷ Replace ?? by the update mentioned in BFGS Theory
8:     $k = k + 1$
9: Output: $x_k$
```

Consider the functions $f(\mathbf{x}) = f(x_1, x_2, x_3, ..., x_n) = \sum_{i=1}^{n-1}[4(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$, $g(\mathbf{x}) = g(x_1, x_2, x_3, ..., x_n) = \sum_{i=1}^{n}[(x_1 - x_i^2)^2 + (x_i - 1)^2]$.

1. **[R]** What is the minimizer and minimum function value of $f(\mathbf{x})$ and $g(\mathbf{x})$? Are both the functions convex? What is a suitable initial choice of B (denoted by B_0, i.e. Replacement of first ?? in the **Algorithm 3**)? Justify with proper reasons.

2. Implement **Algorithm 3** for solving $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, Use backtracking line search with $\alpha_0 = 0.9, \rho = 0.5, \gamma = 0.5$. Take the starting point to be $\mathbf{x_0} = (0, 0, ..., 0)$. Take $n \in \{1000, 2500, 5000, 7500, 10000\}$, find minimizer of the objective function in each case and compute the time taken by the BFGS method with backtracking line search. Tabulate the time taken by BFGS method for each n.

3. Implement **Algorithm 1** for solving $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, Use backtracking line search with $\alpha_0 = 0.9, \rho = 0.5, \gamma = 0.5$. Take the starting point to be $\mathbf{x_0} = (0, 0, ..., 0)$. Take $n \in \{1000, 2500, 5000, 7500, 10000\}$, find minimizer of the objective function in each case and compute the time taken by the Newton's method with backtracking line search. Tabulate the time taken by steepest descent with inexact line-search for each n.

4. Compare the time taken by **Algorithm 1 - Steepest Descent** with backtracking line search against the time taken by **Algorithm 3 - BFGS Method** with backtracking line search for each value of n. Plot the time taken by both methods vs n using different colors. Comment on your observations.

Bibliography

1. Y. Nesterov, *Introductory lectures on convex optimization: A basic course.* Springer Science & Business Media, 2003, vol. 87.

2. Jiantao Jiao, Course Materials, https://people.eecs.berkeley.edu/~jiantao/227c2022spring/material.html

3. Matthias L. R. Haußer, Lecture Slides, https://people.maths.ox.ac.uk/hauser/hauser_lecture2.pdf

4. Jhon A. Jhonorio, CS 52000: Optimization: Quasi-Newton Methods, https://www.cs.purdue.edu/homes/jhonorio/16spring-cs52000-quasinewton.pdf