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

In this lab, we will consider the ordinary least squares regression (OLSR) problem. We will discuss a few optimization algorithms to solve it. Please use **only** Python as your programming language.

Preparation Exercise (PREP):

1. Import the required Python packages using the following commands
   `import numpy as np`
   `import matplotlib.pyplot as plt`
   (Recall the functionality of these packages.)
2. For replication purposes, initialize the random number generator using `np.random.seed(1000)`
3. Use the `np.random.randn` function to create A as a random numpy array of 100 rows and 2 columns. Comment on the purpose of `np.random.randn` function. In the nomenclature of data science, we shall call A to be a data set of 100 data points, each of dimension 2.
4. Create $\bar{x}$ as a random vector of size 2×1 such that $\bar{x} \in [1, 5] \times [-5, -8] \subset \mathbb{R}^2$.
5. Create ε as a random vector of size 100×1.
6. Compute $y = A\bar{x} + \varepsilon$. Use an appropriate numpy package to do the matrix multiplication $A\bar{x}$ efficiently.

Exercise 1: Direct least squares loss minimization

Note that y is a noisy version of $A\bar{x}$. We will now try to estimate $\bar{x}$ assuming that we are given y and A. One possible approach is to solve the following problem:

$$\min_x f(x) = \frac{1}{2}\|Ax - y\|_2^2. \tag{1}$$

The loss term $\|Ax - y\|_2^2$ is called the ordinary least squares (OLS) loss and the problem (1) is called the OLS Regression problem.

1. Write Python functions using appropriate `numpy` routines to compute the objective function value, the gradient value and the Hessian of f.
2. With a starting point of $x = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, solve problem (1) using the Newton's method implemented in the previous lab. Comment on difficulties (if any) you face when computing the inverse of Hessian (recall that you need to use an appropriate Python library to compute the inverse of the Hessian). Plot the iterates (use `plt.scatter`) against the iterations. Similarly, plot objective function values obtained from Newton's method against the iterations. Comment on the convergence rates of the iterates and the objective function values, by recalling the definitions given in previous lab.
3. With a starting point of $x = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, solve problem (1) using the BFGS method implemented in the previous lab. Plot the iterates (use `plt.scatter`) against the iterations. Similarly, plot objective function values obtained from BFGS method against the iterations. Comment on the convergence rates of the iterates and the objective function values.

4. Compare and contrast the results obtained by Newton's method and BFGS method.

5. In a single 3D plot, depict the regressors obtained by both the methods as lines, along with the data points. Use different colors for the two regressors and a different color for the data points. For plotting, try to use the following code snippet:

```
 from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure() # Note:  plt is already created
ax = fig.add_subplot(111, projection='3d') # to display the axes
# ax.scatter(A[:,0],A[:,1]) # display the data points
# describe the functionality of the following two lines
x1 = np.linspace(-10, 10, 10000)
x2 = np.linspace(-10, 10, 10000)
# Write code to compute the regressor for the space spanned by x1 and x2
# regressor = ????
# Plot the regressor
# ax.scatter(x1,x2,regressor)
plt.show()
```

6. Prepare similar plots for Newton and BFGS methods for the initial values $x = \begin{pmatrix}1\\1\end{pmatrix}$, $x = \begin{pmatrix}-1\\-1\end{pmatrix}$, $x = \begin{pmatrix}5\\5\end{pmatrix}$ and $x = \begin{pmatrix}-5\\-5\end{pmatrix}$. Comment on the convergence in each case. Also, compare and contrast the results obtained by Newton's method and BFGS method in each case. Explain if the convergence is dependent upon the starting points.

Exercise 2: Regularized least squares loss minimization

1. Let us now introduce the following regularized problem (with $\lambda > 0$):

$$\min_x f_\lambda(x) = \frac{\lambda}{2}x^\top x + \frac{1}{2}\|Ax - y\|_2^2. \tag{2}$$

   Comment on the significance of the newly added regularizer term $\frac{\lambda}{2}x^\top x$, when compared to problem (1).

2. Write Python functions to compute the function value, gradient and Hessian of f_λ.

3. Fixing $\lambda = 1$, solve the problem (2) using Newton and BFGS methods with five different starting point values $x = \begin{pmatrix}0\\0\end{pmatrix}$, $x = \begin{pmatrix}1\\1\end{pmatrix}$, $x = \begin{pmatrix}-1\\-1\end{pmatrix}$, $x = \begin{pmatrix}5\\5\end{pmatrix}$ and $x = \begin{pmatrix}-5\\-5\end{pmatrix}$. Prepare two plots: in the first plot, depict the objective function values against iterations for each starting point; in the other plot, depict the iterates against iterations for each starting point (use different colors where necessary). Comment on the convergence rates of the iterates and the objective function values. Compare and contrast the results obtained by Newton's method and BFGS method. Depict the regressors obtained by both the methods as lines, along with the data points in a single 3D plot.

4. Repeat the above experiment with $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100, 1000\}$ for the starting point $x = \begin{pmatrix}0\\0\end{pmatrix}$. Plot the final objective function values obtained for diferent values of λ. Find the λ value which achieves the best objective function value. Explain the behavior of objective function value $f_\lambda(x)$ relative to λ. Comment on the running times required for different λ values.

5. Now consider A to be of size 500×2. Appropriately, change the size of ε vector. Compute $y = A\bar{x} + \varepsilon$. For a starting point $x = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, solve (2) with Newton and BFGS methods for different values of λ similar to that in the previous problem. Prepare two plots: in the first plot, depict the objective function values against iterations for each value of λ; in the other plot, depict the iterates against iterations for each value of λ (use different colors where necessary). Comment on the convergence. Repeat the experiments with A to be of size 1000×2. Prepare a table where you clearly write the size of the data matrix A, λ value used and the CPU times taken to solve each of these problems.