UCK 337
Introduction to Optimization
Spring 2019-2020
Problem Set II

Asst. Prof. N. Kemal Ure
Department of Aeronautics
Istanbul Technical University

Due date is **April 28th.**

Homework Policy:

- All homework should be submitted through Ninova. No hard copy submission are needed/allowed.
- Late homeworks can be submitted by email with a -30 points penalty per day. (Hence after 4 late days you will not gain any scores by submitting your homework)
- No hand written solutions are allowed. If you type the solution in LaTeX, you will get $+10$ points. There are no extra points for solutions written in MS Word or any other word processing tool.
- There is a bonus problem at the end, which is worth extra $+10$ points. The bonus problem is significantly more difficult than the rest of the PSet. Only attempt at doing it after you have mastered the subject and solved all the other problems.
- The full score for this PSet is 100 points. However, you can get a total of 120 points if you type the solutions in LaTeXand solve all the problems correctly along with the bonus problem.
- You can discuss the solution strategies with your classmates, however you must completely type your solutions on your own. If any cheating is detected, you will get a -100 (yes that is a negative score) from the PSet.

Gradient Descent Methods

- **Problem 1 (70 points):**

  1. Implement the steepest descent and fixed-step size gradient descent algorithm in a programming language of your choice. For the steepest descent, use the Secant method for computing α_k. Submit all your code. (10 points)
  2. Verify that your steepest descent code works by comparing your results with the example 8.1 from Chong's book (15 points).
  3. Consider the Rosenbrock function:

  $$\text{minimize} \quad \boldsymbol{f}(\boldsymbol{x}) = \sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$$

  Use the steepest descent algorithm to minimize the Rosenbrock function for the dimensions $n = 2, 5, 10, 20, 100, 200, 300$. Terminate the optimization when the norm of the gradient is less than $\epsilon = 10^{-2}$. For initialization, Pick 100 random initial points from the intervals $x_i \in [-2, 2], i = 1, \ldots, n$ and average your computation time (in seconds) and number of iterations over those random points. Plot the average computation time and number of iterations over n. (25 points)
  4. Repeat Part 3 for the fixed step-size gradient descent algorithm using the same initial conditions and the tolerance. Tune the step-size α to get a good convergence rate. What values of α works for this problem? How does the dimension of the problem affect the selection of α? Support your arguments with plots. You are free to use any number/kind of plots. (20 points)

Newton's Method

- **Problem 2 (30 points):** Consider Newton's algorithm.

  1. Write a computer software that implements this algorithm. Submit all your code. Once again confirm that your implementation is correct by comparing results with Chong's book. (10 points)
  2. Find a continuous function $f : \mathbb{R}^n \to \mathbb{R}$ such that the original Newton's algorithm does not converge, but the Levenberg modification works. Verify this using your code. (10 points)
  3. Find a function $f : \mathbb{R}^n \to \mathbb{R}$ such that the neither Newton's algorithm nor Levenberg's modification converges, regardless of how μ_k is chosen in Levenberg's modification . Verify using your code. (10 points)

Bonus Problem

- **Problem 4 (10 points):** Consider the neural network example we did for the MNIST dataset. Improve this example by:
  - Replacing the numerical derivative with the analytical derivative (Hint: Look up the concept of *backpropagation*)
  - Train it on more data points and test the model in digits that were not used in training.