Laboratory Assignment 5
Exploring Penalty Functions

CSC372-M72: Optimisation

2022-23

1 Objectives.

- To code penalty functions.
- To investigate how penalties transform original problems.

2 Background

In this assignment, you will deal with the optimisation problem below.

$$\min_{x\in[0,1]} f(x) = (6x-2)^2 \sin(12x-4), \tag{1}$$

subject to:

$$g(x) = \sin(10x) \leq 0. \tag{2}$$

Our goal would be identify the feasible solution with the minimum value for $f(x)$ within the bounds of $x \in [0, 1]$.

We will primarily deal with two ways of penalising constraint violations:

- Death penalty
- Static penalty

In death penalty, we transform the function and and its constraints in the following way:

$$\phi_{death}(x) = \begin{cases} f(x) & ;\text{if } g(x) \leq 0 \\ \infty & ;\text{otherwise.} \end{cases} \tag{3}$$

Practically, we set the value of ∞ to a large value.

For static penalty, we have the following transformation:

$$\phi_{static}(x) = f(x) + s(x), \tag{4}$$

where,

$$s(x) = r \times \max[0, g(x)]^{\beta}. \tag{5}$$

With the transformed function $\phi.(x)$, the original optimisation problem can be recast as:

$$\min_{x \in [0,1]} \phi.(x), \tag{6}$$

where $\phi.(x)$ represents either $\phi_{death}(x)$ or $\phi_{static}(x)$.

3 Tasks

1. Implement the functions in equations (1), (2), (3) and (4).

2. Plot these function responses for 1000 equally spaced points in horizontal axis (representing x), and the associated function responses in the vertical axis for all function implemented in the first task, using `matplotlib.pyplot`. Use $r = 20$, $\beta = 1$ and $\infty = 20$.
   HINT: You may find `x = np.linspace(0, 1, 1000)` and `matplotlib.pyplot.plot` useful for this. You can also plot the horizontal line at 0 using `matplotlib.pyplot.axhline` to see where $g(x)$ crosses the 0 level.

3. What is the location of the minimum $x^* = \arg\min_{x \in [0,1]} f(x)$ disregarding the constraint in (2)?
   HINT: What you have done in the previous task is essentially a form of grid search with 1000 points in one dimension.The minimum function value and its location across these 1000 points may help you answer this question. In particular, you may find the following functions useful: `numpy.min` and `numpy.argmin`.

4. Now, consider the constraint function $g(x)$, and locate the best $x_g^* = \arg\min_{x \in [0,1] \wedge g(x) \leq 0} f(x)$. Is this different from the solution you found in the previous task?

5. Where are the minima $x_d^* = \arg\min_{x \in [0,1]} \phi_{death}(x)$ and $x_s^* = \arg\min_{x \in [0,1]} \phi_{static}(x)$? Are x_d^* and x_s^* the same? Do they have comparable values against x^* and x_g^*?

6. If we change the equation (4) such that the penalty term is subtracted instead, i.e. $\phi'_{static}(x) = f(x) - s(x)$, what would the function response $\phi'_{static}(x)$ look like? Plot the function response $\phi'_{static}(x)$. Now, answer the following questions:

   (a) Did the location of the minimum change?

   (b) How can this new transformation be useful?

7. If equation 2 was changed as follows: $g(x) = \sin(x) \geq 0$, how should you deal with that when designing penalty functions? What would be the optimal solution in this case?