Developing Optimisers in Python
Coursework Assignment (2020-21)

Module:	Optimisation.
Module code:	CSC372-M72.
Module coordinator:	Dr Alma Rahat (a.a.m.rahat@swansea.ac.uk).
Allocated marks:	**20**% of the total module marks.
Submission deadline:	**11:00 on 30 November 2020 (Monday)**.
Submission site:	`canvas.swansea.ac.uk`

Tasks

Consider the following specification for an optimisation problem.

We are to optimise the weight of a tension or compression spring. There are three variables that determine the weight: the diameter of the wire (x_1), the mean of the diameter of coil (x_2) and the number of active coils (x_3). Therefore, any arbitrary decision vector has three components $\mathbf{x} = (x_1, x_2, x_3)^\top \in \mathcal{X}$. The feasible space $\mathcal{X}$ is defined by four constraints and bounds on $\mathbf{x}$. The problem is defined as:

$$\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) = x_1^2 x_2 (2 + x_3), \tag{1}$$

where the feasible space $\mathcal{X}$ is subject to:

$$g_1(\mathbf{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0, \tag{2}$$

$$g_2(\mathbf{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \leq 0, \tag{3}$$

$$g_3(\mathbf{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0, \tag{4}$$

$$g_4(\mathbf{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0, \tag{5}$$

and with bounds

$$x_1 \in [0.05, 2], \tag{6}$$

$$x_2 \in [0.25, 1.3], \tag{7}$$

$$x_3 \in [2, 15]. \tag{8}$$

Now, you have the following tasks.

1. Implement all the functions $f(\mathbf{x})$ and $g_i(\mathbf{x}); \forall i \in [1, 4]$ independently, where each function takes at least a Numpy array $\mathbf{x}$. Each function shouold have an independent counter that represents how many times a respective function has been called (or in other words evaluated).

2. Implement the Random Search (RS) method discussed in the lectures that can use the functions defined above and return an approximation of the optimum.

3. Implement a stochastic global optimisation algorithm of your choice (selected from the algorithms covered in the course) that can use the functions defined above and return an approximation of the optimum.

4. For 21 repetitions of each of the algorithms implemented in 2 and 3, compare and comment on the performances of these optimisers. The number of function evaluations on $f(\mathbf{x})$ that you are allowed at each instance of an optimisation is 3000 at most.

In this assignment, you must use Python 3.x to develop your code in Jupyter notebook. Please note you are allowed to use basic and advanced Python modules, such as Numpy, Scipy, Matplotlib, etc. However, the core of the algorithmic implementations must be your own: for instance, you cannot just use a module that already implements the algorithm you selected. If you are in doubt, please feel free to contact the module coordinator for clarifications.

Submission

You are required to submit the following on Canvas.

Code A single Jupyter notebook file (with the format `XXXXXXX.ipynb`[1]). You must ensure that all the results are pre-generated and repeatable, otherwise you may lose marks.

Documentation Within the notebook, at the top before your code, please provide answers to the following questions. The maximum word limit for your answer for each question is given at the right.

1. Which stochastic optimiser did you choose (for Task 3)? [100 words]
2. Why did you choose this optimiser? [100 words]
3. What constraint handling method(s) are you using and why? [250 words]

Assessed Learning Outcomes (ALOs)

The assessed learning outcomes of this assignment are as follows.

ALO_1 Demonstrate systematic understanding of fundamental concepts of optimisation problems and algorithms.

ALO_2 Propose an appropriate method to solve an optimisation problem.

ALO_3 Develop appropriate software for solving an optimisation problem.

ALO_4 Critically evaluate performance of multiple competing optimisers, and communicate analysis.

[1]Please use your student ID to replace XXXXXXX in the name of the document.

Marking Criteria

Below we provide a table describing the mark allocation. Please note that the following allocation is indicative only, and may change following moderation.

Topic	Expectation	Indicative percentage of marks (%)
Problem implementation (Task 1, ALO_1)	Correctly implement all the functions.	20%
Implementation of RS (Task 2, ALO_3)	A flexible implementation of RS that can be used for constrained problems. It should work with a random seed such that results may be repeated exactly.	15%
Proposing an algorithm, and implementation (Task 3, ALO_2, ALO_3)	A flexible implementation of an optimiser of your choice that can be used for different problems. It should work with a random seed such that results may be repeated exactly.	30%
Using appropriate constraint handling technique (Tasks 2 and 3; ALO_2)	Use contemporary methods for RS and the optimiser of your choice.	10%
Performance evaluation (Task 4,ALO_4)	A boxplot showing performances of the competing optimisers over 21 repetitions. Statistical comparison of final results, and appropriate comments on significance.	20%
Document structure and code quality (ALO_4)	The document follows a sound logical structure, and the code is written following standard practices.	5%

You will be marked in each category based on the level at which you meet the expectations described in the table above. The level descriptions are given below.

Excellent	Very good	Good	Adequate	Unacceptable
$70-100\%$	$60-69\%$	$50-59\%$	$40-49\%$	$0-39\%$

Please note that we may hold vivas as part of the assessment process. If you are invited for a viva, you will be expected to download your notebook from Canvas, demonstrate functionalities, and answer questions about your submission.

This is an individual coursework, and please adhere to the guidelines about academic misconduct which can be found in the following link:

`myuni.swansea.ac.uk/academic-life/academic-misconduct/`

Please remember that failing to adhere to the guidelines may have severe consequences, including withdrawal from the university.