Module 20 Numerical Optimisation

Module title	Numerical Optimisation
Module NFQ level (only if an NFQ level can be demonstrated)	7
Module number/reference	BSCH-NO
Parent programme(s)	Bachelor of Science (Honours) in Computing Science
Stage of parent programme	Award stage
Semester (semester1/semester2 if applicable)	Semester 1
Module credit units (FET/HET/ECTS)	ECTS
Module credit number of units	5
List the teaching and learning modes	Direct, Blended
Entry requirements (statement of knowledge, skill and competence)	Learners must have achieved programme entry requirements.
Pre-requisite module titles	BSCH-FC, BSCH-PAS, BSCH-LA
Co-requisite module titles	None
Is this a capstone module? (Yes or No)	No
Specification of the qualifications (academic, pedagogical and professional/occupational) and experience required of staff (staff includes workplace personnel who are responsible for learners such as apprentices, trainees and learners in clinical placements)	Qualified to as least a Bachelor of Science (Honours) level in Computer Science or equivalent and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent.
Maximum number of learners per centre (or instance of the module)	60
Duration of the module	One Academic Semester, 12 weeks teaching
Average (over the duration of the module) of the contact hours per week	3
Module-specific physical resources and support required per centre (or instance of the module)	One class room with capacity for 60 learners and a class room with capacity for 25 learners for tutorials

Analysis of required learning effort		
	Minimum ratio teacher / learner	**Hours**
Effort while in contact with staff		
Classroom and demonstrations	1:60	24
Monitoring and small-group teaching	1:25	12
Other (specify)		
Independent Learning		
Directed e-learning		
Independent Learning		53
Other hours (worksheets and assignments)		36
Work-based learning – learning effort		
Total Effort		125

Allocation of marks (within the module)					
	Continuous assessment	**Supervised project**	**Proctored practical examination**	**Proctored written examination**	**Total**
Percentage contribution	50%			50%	100%

Module aims and objectives

The main aim of the module is to introduce learners to the concepts, notations and operations of mathematics that provide basis for the foundational knowledge required for working and developing competencies in various emerging fields. The material covered extends the knowledge of learners who have completed mathematical modules in stage 1 and stage 2. The module provides conceptual understanding of optimisation problems and their real-world applications. First order and second orders methods to solve optimisation problems are covered in detail. Further the module explores probabilistic techniques such as simulated annealing, and evolutionary algorithms to find solutions to optimisation problems. The learners will have hands-on experience of implementing solution methods and applying algorithms and techniques covered in this module. Finally the challenge of dimensionality is discussed.

Minimum intended module learning outcomes

On successful completion of this module, the learner will be able to:

1. Explain and exemplify clearly various optimisation problems and their real-world applications
2. Solve optimisation problems using first order and second order methods
3. Use a probabilistic method to approximate an optimal solution
4. Explain and use approaches such as simulated annealing to solve optimisation problems

5. Explain and use evolutionary (meta heuristic, fitness based) algorithms to solve optimisation problems
6. Apply various computation approaches to optimisation problems in an informed manner.
7. Apply mathematical techniques for reducing dimensionality

Rationale for inclusion of the module in the programme and its contribution to the overall MIPLOs

Concepts conserved in the module are foundational to a variety of emerging areas in the field of computing science. Appendix 1 of the programme document maps MIPLOs to the modules through which they are delivered.

Information provided to learners about the module

Learners receive a programme handbook to include module descriptor, module learning outcomes (MIMLO), class plan, assignment briefs, assessment strategy and reading materials.

Module content, organisation and structure

Introduction to optimisation

- Introduction to basics of optimisation
- Real world applications
- Mathematical basics of optimisation problems
- Convergence: Global Aspects
- Convergence: Local aspects
- Computing the Gradient
- Modelling an optimisation problem

Basic Methods

- Optimality Conditions
- First Order Methods
- Gradient Decent Method
- Concepts, the method, implementation & application
- Second Order Methods
- Newton's Method
- Concepts, the method, implementation & application

Probabilistic Methods

- Probabilistic approach to optimisation problem
- Introduction to simulated Annealing
- Basic Iteration
- The Neighbours of a sate
- Selecting parameters
- Acceptance probabilities

- Appling Simulated Annealing

Evolutionary Algorithms

- Introduction to evolutionary algorithms
- Using biological knowledge to inspire the development of natural computation approaches
- Underlying theory: how such algorithms work
- Genetic Algorithms to solve optimisation problems
- Applying Genetic Algorithms

Dimensionality

- Basics of analysing and organising high-dimensional spaces
- Mathematical approached to address dimensionality challenge (e.g. Principal Component Analysis)

Module teaching and learning (including formative assessment) strategy

The module is taught as a combination of lectures and lab sessions. The lecture sessions discuss and explain to learners the mathematical and theoretical concepts and underpinnings. The practical lab sessions give learners an opportunity to implement methods covered in this module to solve optimisation problems and to have a hands-on experience of applying techniques covered to find optimal solutions of various problems.

Assessment is divided into three elements. First is a class test that focusses on the mathematical concepts covered in this module. Second is a computer programming assignment where the learners are asked to develop a programming solution to selected optimisation problems. Finally, there is an end of semester exam that tests the learners understanding of the theoretical and mathematical material.

Timetabling, learner effort and credit

The module is timetabled as one 2-hour lecture and one 1-hour lab per week.

Continuous assessment spreads the learner effort to focus on small steps before integrating all steps into the overall process of computer programming assignment.

There are 36 contact hours made up of 12 lectures delivered over 12 weeks with classes taking place in a classroom. There are also 12 lab sessions delivered over 12 weeks taking place in a fully equipped computer lab. The learner will need 53 hours of independent effort to further develop the skills and knowledge gained through the contact hours. An additional 36 hours are set aside for learners to work on worksheets and assignments that must be completed for the module.

The team believes that 125 hours of learner effort are required by learners to achieve the MIMLOs and justify the award of 5 ECTS credits at this stage of the programme.

Work-based learning and practice-placement
There is no work based learning or practice placement involved in the module.

E-learning
The college VLE is used to disseminate notes, advice, and online resources to support the learners. The learners are also given access to Lynda.com as a resource for reference.

Module physical resource requirements
Requirements are for a classroom for 60 learners equipped with a projector, and a 25 seater computer lab for practical sessions with access to Java and Python (this may change should better techonologies arise).

Reading lists and other information resources
Recommended Text
Bonnans, J. F. (2006) *Numerical Optimization*. Berlin: Springer.

Sheppard Clinton (2017) *Genetic Algorithms with Python*. Los Gatos: Smashwords.

Secondary Reading
Simon, D. (2013) *Evolutionary Optimization Algorithms:* Hoboken: Wiley.

Nocedal, J. and Wright, S. J. (2006) *Numerical Optimization*. New York: Springer.

Brinkhuis, J. and Tikhomirov, V. M. (2005) *Optimization: Insights and Applications*. Princeton: Princeton University Press.

Specifications for module staffing requirements
For each instance of the module, one lecturer qualified to at least Bachelor of Science (Honours) in Computer Science or equivalent, and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent.. Industry experience would be a benefit but is not a requirement.

Learners also benefit from the support of the programme director, programme administrator, learner representative and the Student Union and Counselling Service.

Module Assessment Strategy
The assignments constitute the overall grade achieved, and are based on each individual learner's work. The continuous assessments provide for ongoing feedback to the learner and relates to the module curriculum.

No.	Description	MIMLOs	Weighting
1	Closed book class test	2,3,6,7	20%
2	Computer Programming Assignment	1,3,4,5,6	30%
3	Written exam that tests the theoretical and mathematical aspects of the module	1-7	50%

All repeat work is capped at 40%.

Sample assessment materials

Note: All assignment briefs are subject to change in order to maintain current content.

Class Test

This test consists of three questions; you are to complete all questions.

The test will last 1.5 hour.

Question 1

By using an example, diagrams and correct expressions, explain in detail, step by step, the working of Newton's Method for optimisation.

Question 2

Explain in detail Jacobian Matrix and Hessian Matrix in the context of a function. How do these two relate to first order and second order optimisations?

Question 3

Using the function below and by selecting an appropriate starting point, please show the first five iteration of the Newton method.

$$Y = 0.5X^4 - (X+1)^3 - 3X^2$$

Take Home Assignment

For this assignment you are required to solve the travelling salesman problem either using a Genetic Algorithm or by Simulated Annealing. The travelling salesman problem (TSP) asks the following question: "Given a list of cities and the distance between each pair of cities, what is the shortest possible route that visit each city and returns to the origin city". The first aspect of the assignment that you clearly state which technique you are going to use, and why you have decided to use this technique. This should be an original statement written in a reflective manner.

Please consider a 9 x 9 grid that has 81 points on it. On this grid, randomly select 18 points as 18 cities; from these 18 points (cities), select one where the salesperson is located. This is the point / origin where the journey should start. Distance between two cities (points on the grid) is equal to the Euclidean distance where length / weight of the line connecting two adjacent points on the grid is 5. You are required to provide the code for randomly generating your travelling salesman problem and a description of the generated problem (please provide a diagram), with the relevant dataset.

At this point, using your travelling salesman problem above, give a brief description that how the selected technique will solve this problem. How would you describe this as a optimisation problem?

Solve the problem with your selected technique and by using appropriate libraries; provide the code and present the results in a brief report.

Assessment Criteria:

- Technique selection and a justification for that (5%)
- Generating the problem as per the assignment brief (10%)
- Description of the optimisation problem (10%)
- Solving the problem using the selected technique (70%)
- A brief description of the solution using the diagram of your travelling salesman problem (5%)

This assignment assesses module learning outcomes 1, 3, 4, 5, 6.

GRIFFITH COLLEGE DUBLIN

QUALITY AND QUALIFICATIONS IRELAND
EXAMINATION

SAMPLE PAPER

Numerical Optimisation

Lecturer(s):
External Examiner(s):

Date: **XXXXXXXX** **Time:** **XXXXXXX**

THIS PAPER CONSISTS OF FIVE QUESTIONS
FOUR QUESTIONS TO BE ATTEMPTED
ALL QUESTIONS CARRY EQUAL MARKS

QUESTION 1

(a) "Various forms of optimisation play critical roles in developing intelligent systems". Discuss this statement in detail by using real world examples. Give at least four examples.

(12 marks)

(b) Differentiate between first order optimisation methods and second order optimisation methods. Clearly explain the relevant mathematical concepts. Discuss their strengths and weaknesses.

(9 marks)

(c) What does the "curse of dimensionality" refer to in the context of optimisation problems?

(4 marks)

Total (25 marks)

QUESTION 2

(a) Explain the principle of the gradient descent algorithm. Accompany your explanation with diagram. Explain the use of all the terms and constants that you introduce and comment on the range of values that they can take.

(9 marks)

(b) How can the gradient descent method be generalised to deal with high dimensional optimisation problems?

(4 marks)

(c) Using the function below and by selecting an appropriate starting point, please show first five steps of the gradient descent method.

Y = 0.25X^4 – (X+1)^3 - 3x^2

(12 marks)

Total (25 marks)

`

QUESTION 3

(a) Describe the idea behind the simulated annealing algorithm making reference to its origins as an optimisation methodology?

(11 marks)

(b) Show, in pseudo code, simulated annealing algorithm with description of each step.

(14 marks)

Total (25 marks)

QUESTION 4

(a) Show, in pseudo code, a simple genetic algorithm with a brief description of each of the main elements?

(14 marks)

(b) Explain in detail, by using an example, why it is important to have a mutation operator in a genetic algorithm.

(11 marks)

Total (25 marks)

QUESTION 5

(a) You are required to write a programme in python that, by using appropriate libraries, applies genetic algorithm to find highest point in a region for which the relevant topographic data is provided in a file. The programme should first read this data from the file. You are required to define necessary variables and functions first. The programme should then, by using appropriate libraries, and by applying genetic algorithm looks for the highest point and then displays the results. Clearly explain each step of the solution. Provide comments in the code so that the solution is clear and understandable.

Total (25 marks)