1 Course description

Mathematical optimisation is one of the cornerstones of fields such as Machine Learning, Artificial Intelligence, and Operations Research. Most decision support methods have, at some level, a mathematical optimisation technique at its core, and it is precisely these methods that we will learn in this course.

Linear optimisation is a powerful framework in which one seeks to represent systems by means of linear objective functions and constraints. Using the analogy that variables represent decisions or parameters to be defined, constraints represent rules that a valid configuration or a plan of action for the system, and the function is a measure of performance, one can use that framework to support decision-making in a wide range of applications, from planning industrial chemical plants to training models that learn from data.

In this course, the students will learn the basics of linear optimisation theory as well as advanced algorithms available and how they can be applied to solve challenging real-world inspired optimisation problems. Throughout the course, we will also look into practical and research applications of linear optimisation.

2 Required background knowledge

To be able to follow this course, students need to:

- have a solid foundation in basic linear algebra (minimally equivalent to MS-A0001 - Matrix Algebra; ideally equivalent to MS-C1342 - Linear Algebra);
- have a solid command of programming (equivalent to CS-A1110 - Programming I and CS-A1120 - Programming II).

Prospective students coming from other majors than Mathematics and Systems Sciences (Matematiikan ja systeemitieteiden pääaine) are *strongly* advised to check whether they have sufficient knowledge on the topics covered by these courses before enrolling.

On the other hand, although Julia is used for the computational exercises in the course, we *do not* expect the students to have any knowledge about Julia. All necessary language-specific knowledge will be covered in the exercise sessions.

3 Learning outcomes

Upon completing this course, the student should be able to

- understand how several important problems arising from diverse fields can be cast and solved as linear optimisation problems;
- acquire basic modelling skills for translating real-world problems into linear mathematical optimisation models
- know the main techniques for solving linear optimisation problems and how to apply them in practice;
- know how to use optimisation software for implementing and solving linear optimisation problems.

4 Teaching methods

This course uses a combination of lectures and exercise sessions that introduce the material. The assessment is continuous throughout the course using homework assignments. This means that the student's learning depends on their initiative in attending the lectures and exercise sessions, formulating questions and discussing with colleagues.

The course will be taught by a composition of the following methods:

- lectures;
- exercise sessions;
- guided self-study;
- theoretical and practical exercises;
- homework assignments and feedback.

All lectures will be delivered in person. Voice-over/video recordings will be provided after the lectures as well, subject to the availability of recording equipment in the assigned lecture room.

The exercise sessions will take place as in-person sessions. No recordings will be made. These sessions will consist of practical exercises that will be done as tutorials. These will focus on illustrating and developing concepts using computational tools and provide support for doing the homework assignments. The homework material will also be covered and its solutions will be discussed in these sessions. The teaching assistants will also be available to address questions related to content and homework assignments.

5 Assessment

The final grade of the course is composed of two components:

H: 5 homework assignments, each worth 20 points;
P: Participation component, worth 10 *extra* points;

The conversion scale to the 1-5 scale is as follows.

1-5	0-100
Fail	0-50
1	51-60
2	61-70
3	71-80
4	81-90
5	91-100

Table 1: Conversion from 0-100 to 1-5 scale

5.1 Homework assignments

A total of 5 homework assignments will be handed out. Each homework is worth 20 points, adding to a total of 100 points. The homework for each fortnight (fortnight = two weeks) will be available on Monday from MyCourses and will have as the deadline the Wednesday 23.59h two or three weeks after the release. The submission of the solutions must be made through the course MyCourses website. Homework submissions after the deadlines will be penalised by 5 points per day after the deadline.

The homework will be composed of theoretical and computational exercises. The computational skills required to solve the exercises will be introduced in the exercise sessions, but it is expected that the students have some basic knowledge of programming, and learn and practise the programming language used (Julia) on their own. Supporting material for that will be provided. The programming language that will be used in this course is Julia (julialang.org).

5.2 Participation bonus

Students with attendance at lectures and exercise sessions greater or equal to 20 (out of 22 sessions each, being 11 lectures and 11 exercise sessions) earn 10 extra points. Otherwise, points will be given proportionally to attendance.

6 Course material

Main study material: lecture notes, exercises, homework assignments.

The lecture notes are available at github.com/gamma-opt/optimisation-notes. These notes are open source and may be updated as needed. Please make sure to use an up-to-date version.

7 Course schedule

A tentative schedule for the course is given. The content of each class may be adapted according to the pace of the classes.

Week	Lecture	Content
2	1	Introduction - modelling linear problems
3	2	Linear algebra, polyhedral sets
4	3	Basis, extreme points and optimality of LP
5	4	The simplex method
6	5	Duality I - the dual simplex
7	6	Duality II - other applications
8	–	*Break between Periods III and IV*
9	7	Barrier methods for LP
10	8	Mixed-integer programming (MIP) formulations
11	9	Branch-and-bound
12	10	Cutting planes method
13	–	*Easter holiday*
14	11	Branch-and-cut solvers
15	–	Placeholder session

Table 2: Schedule of classes