Optimization

Evangelos Markakis, Assistant Professor, AUEB, markakis@aueb.gr, 210 8203149

Overview

The course provides an overview of optimization tools and techniques, motivated by large-scale problems that arise in Data Science. Optimization is at the heart of various critical tasks related to handling big data (e.g. classification problems, machine learning, discrete optimization, etc), and a great number of methodologies have been developed over the years. The course aims at first to illustrate how we can model problems as optimization questions. We will then examine a variety of techniques that are currently used for solving such problems in practice (including among others, linear programming and convex programming techniques, combinatorial algorithms, local search methods and genetic algorithms).

Key Outcomes

By completing the course the students will be able to:

- Model relevant problems arising in practice as appropriate optimization questions.
- Argue about developing an algorithmic solution to various optimization problems.
- Employ optimization methodologies and models, along with problem-specific fine tuning.
- Carry out a fully-fledged implementation of optimization tasks .

Requirements and Prerequisites

There are no formal prerequisites. However, it is expected that the students should have familiarity and basic knowledge of algorithmic principles. All algorithms during the course will be presented in the form of pseudocode, independent of specific programming languages. It is also expected that the students will spend some amount of time working with relevant libraries and tools, which will be introduced during the lectures. Finally, mathematical maturity and a basic background on calculus, algebra and discrete math is encouraged.

Required Course Materials

There is no required textbook. All course material will be provided in class and available for downloading.

Books

There are many books on the subject, and a lot of free resources on the Internet; the following selection provides a good foundation for those students who wish to delve deeper on the topics discussed in class.

For the lectures focused on Linear Programming, Integer Programming, Combinatorial Optimization and Local Search methods, the relevant chapters in the following books provide a thorough exposition:

- F. Hillier, G. Lieberman. Introduction to Operations Research, 7th edition, McGraw Hill, 2000.

- C. Papadimitriou, K. Steiglitz. Combinatorial Optimization: Algorithms and Complexity, Dover Publications, 1998.
- W. Michiels, E. Aarts, J. Korst. Theoretical Aspects of Local Search, Springer, Monographs in Theoretical Computer Science, 2007.
- L. Wolsey. Integer Programming, John Wiley and Sons Inc, 1998

For the lectures that concern convex optimization and its interplay with machine learning, the relevant chapters in the following books are appropriate.

- S. Boyd, L. Vandenberghe. Convex Optimization, Cambridge University Press, 2004.
- M. Mohri, A. Rostamizadeh, A. Talwalkar. Foundations of Machine Learning, MIT Press, 2012.
- T. Mitchell, Machine Learning, McGraw Hill, 1997.
- N. Cristianini, J. Shawe-Taylor. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, 2000.

Software/Computing requirements

There are no such requirements

Grading

Students will be graded based on the final exam, as well as on course assignments that will be determined during the course.

Specifically, the grading is decomposed as follows:

1. The final exam will count 60% towards the final grade.
2. One homework assignment will be announced after the first three units and will count 15% towards the final grade.
3. A course project (with the option to work in teams of 2 people) will be announced after the sixth unit, and will count 25% towards the final grade.

Late assignments will either not be accepted or will incur a grade penalty unless due to documented serious illness or family emergency.

Participation & Attendance Requirements

Class attendance is essential to succeed in this course and is part of your grade. In-class contribution is also a significant part of our shared learning experience. When possible, please notify the instructor in advance of an excused absence.

Students are responsible for keeping up with the course material. It is the student's obligation to bring oneself up to date on any missed coursework.

Please arrive to class on time and stay till the end of the class period. Chronically arriving late or leaving class early is unprofessional and disruptive to the entire class. Please also *turn off all electronic devices prior to the start of class. Cell phones tablets and other electronic devices are a distraction to everyone.*

Code of Ethics

Students may not work together on assignments unless the instructor gives permission to do so.

Exercise integrity in all aspects of one's academic work and do not engage in any method or means that provides an unfair advantage. In any case of doubt, students must be able to prove that they are the sole authors of their work by demonstrating their knowledge to the instructor.

Clearly acknowledge and provide citations for the work and efforts of others when submitting written work. Ideas, data, direct quotations, paraphrasing, or any other incorporation of the work of others should be fully referenced. No plagiarism of any sort will be tolerated. This includes any material found on the internet. Reuse of material found in question and answer forums, code repositories, other lecture sites, etc., is unacceptable. You may use online material to deepen your understanding of a concept, not for finding answers.

Any violations will have an impact on your final grade.

Course Syllabus

The course comprises ten units of three hours each. Roughly, the topics of linear and convex optimization will form the first half of the course (5 units), whereas the second half will contain further techniques centered around branch and bound and local search methods.

Units 1-3: Linear and Integer Programming

We will start with an introduction to optimization and discuss simple examples. We will then focus on one of the fundamental topics in optimization, namely optimizing linear objectives under linear constraints (linear programming). We will overview known methods, such as simplex and interior point methods along with software tools for solving linear programs. We will briefly also discuss a class of more difficult problems that are modelled using integer linear programming. The latter class will be revisited in Unit 6.

Units 4-5: Convex Optimization and Applications to Machine Learning

The next form of optimization we will discuss concerns convex objective functions. We will study the gradient descent method as well as applications of convex optimization to machine learning and other problems concerning big data. These include, among others, linear regression, the least square method, and support vector machines.

Units 6-7: Combinatorial Optimization

The next section of the course involves discrete optimization, i.e., problems with a finite solution space defined over combinatorial domains. Classic examples include the SAT problem as well as problems defined on graphs, such as the Vertex Cover, or the Traveling Salesman problem. We will revisit Integer Linear Programming from Units 1-3 as a way of formulating and solving these problems. We will discuss branch and bound and other heuristic methods for obtaining fast near optimal solutions and also briefly overview techniques for establishing provable approximation guarantees.

Units 8-9: Local Search Methods

By local search, we mean a class of heuristic methods where one starts from a feasible solution and tries to make small local changes that would lead to an improvement of the objective function. Although such a methodology cannot always succeed, there are many classes of problems where we can obtain close to optimal solutions with a small number of iterations. Among the available local search methods, we will pay attention to tabu search and simulated annealing.

Unit 10: Genetic Algorithms

The last part of the course is dedicated to genetic algorithms. These are algorithms inspired by the process of natural selection, where in each iteration a pool of candidate solutions is maintained and evaluated according to their fitness. The “fittest” solutions are stochastically selected and form the new generation of solutions after some “mutation” takes place. We will overview some applications of these methods and their relations to the broader class of evolutionary algorithms.