Please submit your answers to Mycourses before the given deadline as a separate files (.pdf file with the report and Jupyter notebook .ipynb files if there are any). **Do not compress them into a single (zip or equivalent) archive**.

Introduction

In this project, you will learn how to implement the *Alternating Direction Method of Multipliers* (ADMM) algorithm to solve large-scale optimisation problems with decomposable structures. The lectures and exercises presented in the course have all the information needed to successfully complete this project. You are expected to implement ADMM using a skeleton code provided as a Jupyter notebook to solve two given problem instances. In addition, you must write a report where you (i) discuss the technical details regarding how ADMM can be adapted to the problem at hand and (ii) describe implementation details and assess the a performance of your ADMM implementation.

Learning objectives

1. Learn how the ADMM and its variants operate in greater depth and understand its potential to solve optimisation problems with decomposable structures.
2. Learn how to implement ADMM and apply it to solve mathematical programming problems.
3. Learn how to employ the knowledge acquired in the course to understand technical literature.

General requirements

1. Students can work individually or in pairs to complete this project. Groups of three or more individuals will not be accepted.
2. Students are expected to submit a report (see details in the "Report format" section) and **the completed Jupyter notebook skeleton** developed for the project. The code will be tested for correctness which is part of the assessment. The notebook can be downloaded here. **For detailed information about the implementation, carefully read the comments in the skeleton notebook**.
3. **Rename your notebook according to your student number before submission. For example, if your student number is 112233, rename the notebook as 112233.ipynb**
4. **Deadline** for submitting the project report and the code files is **Friday 17/12/2021 23:55**. Late submissions will not be accepted.

Specific project requirements

You must implement an ADMM algorithm to solve large-scale linear optimisation problems exploiting its decomposable structure. In particular, you will solve three randomly generated instances of the *stochastic capacity expansion problem*, which is a two-stage stochastic programming problem. Details of the problem and its formulation are described below.

Carefully follow the instructions in the provided skeleton code when implementing the algorithm. The values you select for the penalty parameter ρ and their impact on the convergence of the algorithm must be discussed in the report. All other algorithmic parameters, such as initial values for primal and dual variables and tolerances, will be provided in the skeleton code.

The skeleton code first presents solutions to the two problem instances using the formulation presented below. After implementing the ADMM variant of this formulation in the notebook, compare and discuss in your report the solution times of these two formulations on the two problem instances.

Stochastic capacity expansion problem

Let I be a set of suppliers and J a set of clients with unknown demands. The possible demand values are represented by a finite set of scenarios S with corresponding demand realisations D_{js} (notice the capital letters for input parameters and lower-case letters fro decision variables) and associated scenario probabilities P_s, for all $j \in J$ and $s \in S$. The objective is to minimise the costs associated with reserving capacity amounts x_i from each supplier $i \in I$ in advance so that realised demands D_{js} of all customers $j \in J$ in each scenario $s \in S$ can be optimally fulfilled (on average) when these demands are observed in the future.

Let y_{ijs} be the amount of capacity reserved from supplier $i \in I$ that is used to fulfil the demand of client $j \in J$ in scenario $s \in S$. Let u_{js} be the amount of demand of client $j \in J$ in scenario $s \in S$ that is not fulfilled in case not enough capacity was reserved. We assume that unfulfilled amounts u_{js} are penalised at a unit cost Q_j for each client $j \in J$ in all scenarios $s \in S$. Each unit of capacity x_i reserved from supplier $i \in I$ costs C_i, and f_{ij} is the unit cost to fulfil the demand of client $j \in J$ using supplier $i \in I$. A maximum capacity B_i can be reserved from each supplier $i \in I$, and a total budget of B is available for capacity acquisition. This problem can be formulated as follows:

$$
\begin{aligned}
\text{min.} \quad & \sum_{i \in I} C_i x_i + \sum_{s \in S} P_s \left(\sum_{i \in I} \sum_{j \in J} F_{ij} y_{ijs} + \sum_{j \in J} Q_j u_{js} \right) & \\
\text{subject to:} \quad & \sum_{i \in I} C_i x_i \leq B & \\
& \sum_{j \in J} y_{ijs} \leq x_i, & \forall i \in I, \ \forall s \in S \\
& \sum_{i \in I} y_{ijs} = D_{js} - u_{js}, & \forall j \in J, \ \forall s \in S \\
& x_i \leq B_i, & \forall i \in I \\
& x_i \geq 0, & \forall i \in I \\
& y_{ijs} \geq 0, & \forall i \in I, \ \forall j \in J, \ \forall s \in S \\
& u_{js} \geq 0, & \forall j \in J, \ \forall s \in S.
\end{aligned}
$$

To accurately model the random demands D_{js} for all $j \in J$, a large number of scenarios $|S|$ might be needed, which makes this class of problems challenging from a computational perspective. In this assignment, you are required to implement an ADMM-based algorithm that decomposes the problem into several subproblems, one for each scenario $s \in S$, that can be solved independently (and potentially in parallel).

Information concerning ADMM and how it can be applied to these problems can be found in this technical paper which is strongly recommended as a supporting reference for the technical developments required in this assignment.

Report format

A Latex template for the report is provided and can be downloaded here. Font sizes, margins, and other layout settings will be set for the template and are not to be tampered with.

The report has a strict page limit of 10 pages, which includes figures. References are not necessary, but if used, they are not included in the page limit.

The report must consist of the following sections:

1. **Background**

Describe the ADMM and its relevant technical details for the application at hand. Provide a general pseudocode of ADMM and provide a technical description of its steps based on the knowledge acquired in the course. A discussion on stopping conditions and convergence would also be of value in this section.

2. **Applications**

   Describe how ADMM can be specialised for the stochastic capacity expansion problem. Provide a detailed description of each of the ADMM steps, relating to the pseudocode presented in the previous section. Provide all details necessary for implementing this variant of ADMM for the problem you are solving.

3. **Discussion and conclusions**

   Describe the results obtained in terms of number of iterations. Assess the ADMM implementation with respect to values used for the penalty term ρ and solution obtained when compared to that you obtain by solving the full-scale model (i.e., the model without decomposition). Present a structured performance comparison with respect to the total computational time and the number of iterations before converging to a solution for different values of the penalty term ρ within the interval [0.5, 100] .

Assessment

The assessment is based on the report and the correctness of the ADMM implementation. The report and the Julia code will be assessed by the lecturer according to the following criteria:

(1) Correctness of the algorithms - if the implementations are all correct.

(2) Technical quality of analysis - adequate statements and precise analyses, supported by experimental evidence and theoretical knowledge.

(3) Format, clarity, and presentation - tables and plots that are pristine, easily readable, and support the arguments made; use of precise technical language.

Each criterion (1), (2), and (3) will be graded by the lecturer and the final grade will be the rounded average of the criteria grades.