Diss. ETH No. 18188

Finding Approximate Solutions for Large Scale Linear Programs

A dissertation submitted to
ETH Zurich

for the degree of
Doctor of Sciences

presented by
Vânia Lúcia Dos Santos Eleutério
Dipl. Math. ETH

born June 20, 1978
citizen of Setúbal, Portugal
and Genève (GE)

accepted on the recommendation of
Prof. Dr. Hans-Jakob Lüthi, examiner
Prof. Dr. Fabián A. Chudak, co-examiner
Prof. Dr. Yurii Nesterov, co-examiner
Prof. Dr. Kay W. Axhausen, co-examiner

2009

Acknowledgments

I wish to express my gratitude to Prof. Hans-Jakob Lüthi for giving me the opportunity to work on this project and his constant patience and generosity.

My special thanks go to Prof. Fabian A. Chudak, whose deep understanding and enthusiasm for the subject had a big influence on this thesis reaching this point. Working together has been a constant source of creativity.

Having had Prof. Yurii Nesterov as co-examiner of this thesis was the best basis to work on algorithms founding on his research. I am definitely indebted to his support.

Further, I would like to thank Prof. Kay W. Axhausen for enriching the application part of the thesis with his knowledge on traffic problems and providing meaningful critique. I am also thankful for him and Mr. Arendt from the Federal Office for Territorial Development for providing me with real traffic data.

My very special thanks go to Michel Baes who arrived just in time at the Institute for Operations Research to conduct a thorough mathematical check of my thesis. Alexander Rudyk contributed during and after his Bachelor thesis to this work for which I am grateful. Also thanks to Rico Zenklusen and Michael Guarisco for helping me out when I had problems with Linux or C++ (I will try to never again destroy hardware).

I would like to thank Hilda Fritze-Vomvoris for proofreading this thesis despite having to digest so many mathematical formulas.

I am grateful to the Swiss National Science Foundation for partially funding this project.

Here is the place to deliver a global bunch of thanks to all my former and present colleagues at the Department of Mathematics and especially at the Institute for Operations Research. The multi-cultural environment led to many enjoyable moments (and improved my skills in Europanto).

Finally, I wish to say thank you to my mother, my sisters, my boyfriend Dan, and my close friend Benoît for their devotion, support, encouragement, and patience.

Many thanks go to all my friends who have always been there for me.

Abstract

Linear Programming is one of the most frequently applied tools for modeling and solving real world optimization problems. Nonetheless, most commercially available solvers are often incapable of dealing with large problem sizes, e.g. millions of variables or hundreds of thousands of constraints, arising in modern applications. To cope, researchers have applied decomposition methods, in particular Lagrange relaxation. In this thesis, we investigate new methods for solving Lagrange relaxations and consequently the Linear Program approximately both from a theoretical as well as a numerical point of view.

In the theoretical part, we consider two recently developed primal-dual optimization methods by Nesterov for approximately solving non-smooth convex optimization problems. The first method, called Primal-Dual Subgradient method, is a variation of the standard subgradient method, where the computed subgradients are used not only to create at each step a primal but also a dual solution. This method has a convergence rate of $O(\frac{1}{\epsilon^2})$ to reach an absolute accuracy of $\epsilon > 0$, which is the best possible rate for techniques based on subgradients. The second method, called Excessive Gap method, consists of a smoothing of the objective functions and the usage of optimal gradient schemes. It has a convergence rate of $O(\frac{1}{\epsilon})$. Using this method, we design a polynomial time approximation scheme for the linear programming relaxation of the Uncapacitated Facility Location problem, which improves the running time dependence on ϵ from the previously known $O(\frac{1}{\epsilon^2})$ to $O(\frac{1}{\epsilon})$.

Both methods depend on oracles for solving subproblems in each iteration. However, exact oracles are often unavailable. We examine the influence of approximate oracles on the overall convergence of the methods. We show that in order to obtain an overall absolute accuracy of ϵ, the Primal-Dual Subgradient method requires oracles with a theoretical accuracy of ϵ^2. In contrast, the Excessive Gap method needs an oracle accuracy of ϵ^5 suggesting that it is much less stable. However, we only assumed to know the absolute accuracy of the solution delivered by the oracles. Introducing other requirements may lead to an improvement of these results.

In the practical part of the thesis, we examine the application of the methods to the linear programming relaxation of the Uncapacitated Facility Location problem and the Static Traffic Assignment Problem, for which we had access to real world data.

As expected by theory, the Excessive Gap method outperformed the Primal-Dual Subgradient method in solving the linear programming relaxation of the Uncapacitated Facility Location problem, for which exact oracles are available. Surprisingly, the Primal-Dual Subgradient method was much better than the Excessive Gap method in solving the Static Traffic Assignment Problem. This can be explained by the subproblems arising in the methods. In the Excessive Gap method, we have to solve Minimum Quadratic Cost Flow problems, which are much harder than the Shortest Paths computations arising in the Primal-Dual Subgradient method. For solving the Minimum Quadratic Cost Flow problem, we considered approximate oracles and we noticed that the Excessive Gap method demonstrated more stability than predicted by theory.

As we used a novel formulation of the Static Traffic Assignment problem developed by Nesterov and de Palma, we also compared the characteristics of the obtained traffic assignments to assignments obtained using the traditional Beckmann model. Results showed that the Nesterov and de Palma model better concentrates travel flows leading to more predictable congestions.

Zusammenfassung

Lineare Programme gehören zu den meist verwendeten Werkzeugen zum Modellieren und Lösen von Optimierungsproblemen aus der Praxis. Die meisten kommerziellen Programme sind jedoch überfordert, wenn die Probleminstanzen riesig werden (Millionen von Variablen oder Hunderttausende von Restriktionen). Für solche Fälle werden oft Dekompositionsmethoden, wie etwa die Lagrange Relaxation, verwendet. In dieser Dissertation untersuchen wir neue Methoden zum approximativen Lösen von Lagrange Relaxationen und damit Linearer Programme sowohl aus einer theoretischen als auch einer praktischen Perspektive.

Im theoretischen Teil betrachten wir zwei kürzlich entwickelte Primal-Duale Optimierungsmethoden von Nesterov, welche nicht glatte konvexe Optimierungsprobleme approximativ lösen. Die erste Methode, Primal-Duale Subgradientenmethode genannt, ist eine Variation der normalen Subgradientenmethode, welche die in jeder Iteration berechneten Subgradienten zur Berechnung nicht nur primaler sondern auch dualer Lösungen verwendet. Die Methode hat eine Konvergenzrate von $O(\frac{1}{\epsilon^2})$, um eine absolute Präzision von $\epsilon > 0$ zu erreichen. Dies entspricht dem bestmöglichen Wert, welcher mit Methoden basierend auf Subgradienten erreicht werden kann. Die zweite Methode, Methode der exzessiven Lücke genannt, verwendet eine Glättung der Zielfunktionen und optimale Gradientenschemen. Sie hat eine Konvergenzrate von $O(\frac{1}{\epsilon})$. Mit Hilfe dieser Methode entwickeln wir ein Approximationsschema in polynomieller Zeit für die Relaxation als Lineares Programm des Platzierungsproblem von Anlagen ohne Kapazitätseinschränkungen, welche die Rechenzeitabhängigkeit vom bisher besten bekannten Wert $O(\frac{1}{\epsilon^2})$ auf $O(\frac{1}{\epsilon})$ verbessert.

Beide Methoden benötigen Orakel zum Lösen von Subproblemen. Exakte Orakel sind aber nicht immer vorhanden. Wir untersuchen den Einfluss approximativer Orakel auf die Konvergenz der Methoden und zeigen, dass, um eine absolute Präzision von ϵ zu erreichen, die Primal-Duale Subgradientenmethode ein Orakel mit einer theoretischen Präzision von ϵ^2 benötigt. Demgegenüber bedarf die Methode der exzessiven Lücke eines Orakels mit einer Präzision von ϵ^5, was auf eine viel kleinere Stabilität hindeutet. Wir haben jedoch ausser der Präzision keine weiteren Voraussetzungen an die Orakel gestellt. Das Einführen weiterer Vorgaben könnte zu einer Verbesserung dieser Resultate führen.

Im Anwendungsteil der Dissertation haben wir die Methoden zum Lösen der Re-

laxation als Lineares Progamm des Platzierungsproblems von Anlagen ohne Kapazitätseinschränkungen und des statischen Verkehrsumlegungsproblems verwendet. Für letzteres hatten wir Zugang zu Daten aus der realen Welt. Wie aufgrund der Theorie erwartet, schlägt die Methode der exzessiven Lücke die Primal-Duale Subgradientenmethode beim Lösen der Relaxation als Lineares Progamm des Anlagenplatzierungsproblems, für welches exakte Orakel vorhanden sind. Überraschenderweise ist die Primal-Duale Subgradientenmethode viel besser im Lösen des Verkehrsumlegungsproblems. Dies kann man durch die zu lösenden Subprobleme erklären. Die Methode der exzessiven Lücke muss Flussprobleme mit quadratischen Kosten lösen, welche viel schwieriger als die kürzeste Wege Probleme in der Subgradientenmethode sind. Für das Flussproblem mit quadratischen Kosten mussten approximative Orakel verwendet werden und es zeigte sich, dass die Methode der exzessiven Lücke stabiler war, als die Theorie erwarten liess.

Da wir eine neue Formulierung von Nesterov und de Palma des statischen Verkehrsumlegungsproblems verwendet haben, haben wir auch die Charakteristiken der erhaltenen Verkehrsumlegungen mit den Umlegungen verglichen, welche man mit dem traditionellen Beckmannmodell erhält. Die Resultate zeigten, dass das Modell von Nesterov und de Palma die Verkehrsflüsse stärker konzentriert, was zu einer Verkehrsprognose mit mehr Staus führt.

Résumé

La Programmation Linéaire fait partie des outils les plus utilisés pour modéliser et résoudre des problèmes d'optimisation survenant dans l'industrie. Ces problèmes ont besoin en général de plusieurs millions de variables et de quelques centaines de milliers de contraintes pour être décrits, ce qui les rend intraitables pour la plupart des logiciels commerciaux. Afin de traiter ce type de problèmes, des méthodes de décomposition sont utilisées, en particulier les relaxations de Lagrange. Dans cette thèse, nous étudions à la fois d'un point de vue théorique et pratique des nouvelles méthodes pour résoudre les relaxations de Lagrange de manière approximative et ainsi les programmes linéaires correspondants.

Dans la partie théorique de cette thèse, nous considérons deux méthodes développées récemment par Nesterov pour résoudre de manière approximative des problèmes d'optimisation convexe mais non differentiables. La première méthode, nommée en anglais Primal-Dual Subgradient method (Méthode de Sous-gradients primale-duale), est une variante de la méthode standard des sous-gradients. A chaque itération, les sous-gradients calculés sont non seulement utilisés pour créer une solution primale mais aussi une solution duale. Pour obtenir une erreur absolue de $\epsilon > 0$, $O(\frac{1}{\epsilon^2})$ itérations sont suffisantes. Cette vitesse de convergence est la meilleure que l'on puisse espérer pour des méthodes basées uniquement sur des sous-gradients. La deuxième méthode envisagée se nomme en anglais Excessive Gap method (Méthode du Seuil excessif). Elle consiste à lisser préalablement la fonction objectif et ensuite à utiliser des méthodes de gradient optimales pour résoudre le problème rendu différentiable. Sa vitesse de convergence est de l'ordre de $O(\frac{1}{\epsilon})$. Grâce à cette méthode, nous avons pu créer un schéma d'approximation dont le temps d'exécution est polynomial en la donnée de l'instance pour résoudre la relaxation en variables continues du problème de localisation de dépôts à capacité illimitée. Ceci améliore la dépendance du temps d'exécution en ϵ de $O(\frac{1}{\epsilon^2})$ à $O(\frac{1}{\epsilon})$ par rapport à la meilleure méthode connue.

Les deux méthodes de Nesterov supposent l'existence d'oracles résolvant de manière exacte des sous-problèmes d'optimisation à chaque itération. Il se peut cependant que de tels oracles ne soit pas disponibles. Nous étudions l'impact de solutions approchées de ces sous-problèmes sur la convergence des deux méthodes. De notre étude, nous obtenons que pour maintenir une erreur absolue de ϵ, les sous-problèmes dans la méthode des Sous-gradients primale-duale doivent être résolus avec une

précision absolue de ϵ^2, et pour la méthode du Seuil excessif avec une précision absolue de ϵ^5. Ce dernier résultat indique que la méthode du Seuil excessif est moins stable que la méthode des Sous-gradients primale-duale. Toutefois nous n'exigeons des oracles approchés qu'une garantie sur l'erreur absolue. D'autres conditions peuvent peut-être améliorer ce résultat.

Cette thèse contient une partie expérimentale, où nous considérons la relaxation en variables continues du problème de localisation de dépôts a capacité illimitée et le problème d'affectation statique du trafic routier. Pour ce dernier problème, nous disposons de données provenant d'un réseau routier réel. La méthode du Seuil excessif surpasse la méthode des Sous-gradients primale-duale pour la relaxation en variables continues du problème de localisation de dépôts à capacité illimitée, ce qui est en accord avec nos résultats théoriques. Pour ce problème nous disposons d'oracles exacts. Par contre, dans le cas du problème d'affectation statique du trafic routier la méthode des Sous-gradients primale-duale est beaucoup plus efficace que la méthode du Seuil excessif. Ceci peut s'expliquer par le type de sous-problèmes qui doivent être résolus dans chacun des algorithmes. Dans le cas de la méthode des Sous-gradients primale-duale, nous devons simplement effectuer un calcul des plus courts chemins alors que dans le cas de la méthode du Seuil excessif nous devons déterminer des flots de coût quadratique minimal. Pour ce dernier problème, nous ne disposons que d'oracles approximatifs. Nous avons constaté numériquement que la méthode du Seuil excessif est plus stable que notre théorie le prévoyait.

Nous avons envisagé dans cette thèse une nouvelle formulation du problème d'affectation statique du trafic routier développée par Nesterov et de Palma. Nous comparons également certaines caractéristiques des affectations du trafic routier obtenues par ce nouveau modèle avec celles obtenues par le modèle classique de Beckmann. Les résultats montrent que le modèle de Nesterov et de Palma distribue le trafic de manière plus concentrée et pronostique plus de congestion.

Contents

1. Introduction

1.1 Motivation

Linear Programming (LP) is the most frequently used tool for modeling and solving optimization problems arising in diverse areas such as telecommunication network management, supply chain management, statistics, finance, and biology. The main reason for its popularity is the widespread belief by practitioners that commercially available solvers are capable of solving any instance of an LP problem in a reasonable amount of time.

Commercially available solvers are based on simplex or interior point methods. Both methods terminate with an optimal solution or conclude that the problem instance is unbounded or infeasible. An algorithm is considered efficient in theory if the time to find an optimal solution is bounded by a polynomial in the size of the input data. In this case the method or the algorithm is said to have a polynomial running time or to be polynomial. Simplex like algorithms are not known to be polynomial. Interior point methods on the other hand are polynomial (see [Kha79, Kar84]).

From a practical point of view, these methods are limited with respect to the problem size. As the problem size grows, simplex-like algorithms require a prohibitive number of iterations and interior point methods need a large amount of time and memory to compute a single iteration. Thus, commercially available solvers are frequently inadequate for the large instances of LP problems arising in practice today, in spite of many improvements introduced in the last twenty years to scale and speed up these solvers (see [Bix02]). For example, problem instances in telecommunication networks exist with several million variables and hundreds of thousands of constraints all needing to be solved in a few minutes (see [Bie02]). However, currently available solvers running on modern computers need hours or even days to solve such instances if they are able to solve them at all (see [Bie02]).

In order to cope with these problems many researchers have investigated decomposition methods, such as Benders' and Dantzig-Wolfe decomposition, and Lagrange relaxations (see [Van96, BT97, Mar99]). The main idea of these methods is to divide the initial problem into subproblems that are simpler and easier to solve. Researchers also exploit the structure of special LP for dealing with even larger problems, for example by using problem sparsity. The efficiency analysis of these methods is mostly based on empirical results for specific problem instances.

In this thesis, the aim is to develop algorithms that given a target accuracy of $\epsilon > 0$ deliver a solution within a factor of $(1 + \epsilon)$ of the optimum of the LP problem and have a running time, which is polynomial in the size of the input data. Such an algorithm is called an approximation algorithm. We can safely relax the optimality requirement since it usually has little value in real world problems, where the model itself and the data may not be accurate. In this case a solution can only be as accurate as the input.

In order to design such an approximation algorithm, we use decomposition methods, in particular Lagrange relaxation. The main difficulty with this approach is that it leads to non-smooth optimization. Usually, researchers have applied subgradient techniques to solve this problem. We apply two recently proposed methods by Nesterov. The first is a primal-dual subgradient technique [Nes09, Nes05b]. The second consists of two steps. First, the Lagrange relaxation is smoothed and then solved by optimal gradient-like optimization schemes [Nes05c, Nes05a]. In this fashion, we obtain algorithms with an approximation guarantee for special LP.

Bienstock and Iyengar were to our knowledge the first to apply these methods. They obtain approximation algorithms for Fractional Packing problems with a running time of $O(\frac{1}{\epsilon})$ [BI04]. Simultaneously to our work, Iyengar, Phillips, and Stein develop approximation algorithms for Semidefinite Packing problems [IPS05]. Chudak and Nagano exploit both methods of Nesterov to solve relaxations of combinatorial problems with submodular penalties [CN07]. Finally, Gilpin et al. develop an algorithm for finding Nash equilibria in sequential games [GHPS07] using the method described in [Nes05a]. That paper presents also numerical results.

Note that many other methods related to Lagrange relaxation exist. Just to mention a few, there are the so-called bundle methods, see e.g. [LNN95], the penalty methods, see e.g. [CD94] or [FG99], the volume method, see e.g. [BA00], and the mirror descent method, see e.g. [NY83] or [BTMN01]. However, these methods are not further investigated in this thesis.

1.2 Goals

This thesis aims at contributing to the understanding of recently developed approximation algorithms both theoretically and numerically.

Theory

From a theoretical point of view, we design polynomial time approximation schemes for special LPs, i.e., schemes that for a given $\epsilon > 0$ deliver a solution within a factor of $(1+\epsilon)$ of the optimum and have an execution time that may depend on

ϵ and must be bounded by a polynomial in the length of the input. We focus on the dependence of the running time of the designed method on ϵ. These schemes are based on optimization methods for non-smooth convex problems developed by Nesterov in [Nes05c], [Nes05a], [Nes09], and [Nes05b]. Binary search and knowledge on the optimal solution are subsequently used to restrict the feasible region as in [You01], [Bie02], [GK02], and [BI04].

Using this approach, we design approximation algorithms for the linear programming relaxation of the Uncapacitated Facility Location (UFL) problem. [CE05] presents an approximation algorithm that improves the running time dependence on ϵ from $O(1/\epsilon^2)$ to $O(1/\epsilon)$. To our knowledge, $O(1/\epsilon^2)$ was previously the best known complexity, see for example the work of [GK02]. The previous result was obtained applying the method presented in [Nes05a]. In this thesis, we extend this approach to other methods by Nesterov.

At each iteration, Nesterov's algorithms require an optimal solution of subproblems that may be non-linear and even more difficult than the original problem. Based on the work of Bienstock and Iyengard in [BI04], we extend the results of Nesterov to accommodate approximate solutions. For the method presented in [Nes05a], [CE05] gives a theoretical bound on the required accuracy of the solutions to the subproblems, however without proof. This proof as well as theoretical bounds for the other methods are presented in this thesis.

The results for the linear programming relaxation of the UFL problem can be extended to other problems such as the Packing problem, Scheduling problem, the Set Covering problem, the Maximum Concurrent Multicommodity Flow problem, and the Survival Network Design problem. Results based on [Nes05a] are given in [CE05]. However, we will not further investigate these problems in this thesis.

Application

An important part of this thesis is the implementation of the designed approximation schemes, since up until now only little empirical work on the performance of approximation algorithms for LP has been done. In the second part of the thesis, the focus is on the practical applicability of the developed methods.

In this work, numerical performance study concerns two problem classes, the linear programming relaxation of the Uncapacitated Facility Location problem and the Static Traffic Assignment (STA) problem. Both problem classes are derived from the class of linear multicommodity flow problems. Since both applied approximation schemes deliver a primal and dual solution at each step, the relative errors can easily be bounded using the duality gap.

The algorithms have been tested on randomly generated instances for the linear programming relaxation of the Uncapacitated Facility Location problem. Both

algorithms, primal-dual subgradient methods and smoothing techniques with gradient schemes were implemented with subproblems solved to optimality. Numerical results showed that performance was good for algorithms based on the smoothing techniques with gradient schemes and poor for the algorithms based on primal-dual subgradient schemes.

For the Static Traffic Assignment problem, a new model developed by Nesterov and de Palma in 2000 [NdP00, NdP03] is considered, which is compared with Beckmann's classical model developed in 1956 [BMW56]. We test both models in small benchmark instances using commercial solvers. Particularly of interest is an understanding of how the two models compare regarding e.g. congestion, Braess paradox, and Price of Anarchy. Then, based on Nesterov's optimization various algorithms are designed for solving the Static Traffic Assignment problem. Using benchmark and real world instances of the problem we compare the algorithms. As opposed to the Uncapacited Facility Location problem, the algorithms based on the primal-dual subgradients schemes show better performance than those based on smoothing techniques with optimal gradient schemes. Furthermore, a comparison between the quality of the Traffic Assignment delivered by our algorithms and the quality of Traffic Assignment computed by VISUM, ([VIS06]), a commercial software based on the Beckmann model, is made.

The Static Traffic Assignment problem as formulated by Nesterov and de Palma corresponds to a Minimum Linear Cost Multicommodity Flow problem and its dual. Lagrange relaxation remains one of the most often used method for solving large Minimum Cost Multicommodity Flow problems. As an example for a different solution approach (without an approximation guarantee), we refer to the recent paper of Babonneau, du Merle, and Vial [BdMV06], where the Lagrange relaxation is solved with a variant of the analytic center cutting-plane method. An overview on approximation algorithms for solving the Minimum Cost Multicommodity Flow problem is given in Chapter 9.

1.3 Structure of the Thesis

The thesis is divided into two main parts. The first part (Chapters 2 to 6) is concerned with the theoretical basis of large scale linear optimization, while the second part (Chapters 7 to 10) deals with practical applications of the algorithms from the first part. Chapter 2 briefly introduces linear programming and frequently used methods applied to solve them exactly or approximately. Chapter 3 describes strongly convex functions, which are fundamental for the methods in the following two chapters. In Chapter 4, we present primal-dual subgradient methods for approximately solving large scale linear programs in detail. Chapter 5 then examines smoothing techniques and gradient schemes. All methods from this thesis

assume that oracles exist, which can provide exact solutions to "easy" subproblems. Chapter 6 provides answers as to whether these methods still converge when only approximate oracles are available.

The second part of the thesis begins with Chapter 7, in which the linear programming relaxation of the Uncapacitated Facility Location problem is examined. The algorithms from Chapters 4 and 5 are implemented and test results are given. Chapter 8 deals with the static Traffic Assignment Problem for which two models are given. Chapter 9 shows numerical results for the Traffic Assignment Problem. We compare on the one hand the practical performance of the implemented algorithms and on the other hand the traffic predictions made by the two models. Chapter 10 concludes the application part with tests of two algorithms on their practical performance for solving the Minimum Quadratic Cost Flow problem, which is the only subproblem that must be solved approximately. Finally, Chapter 11 gives a summary and an outlook.

Part I

Large Scale Linear Programs and Optimization Methods

2. Large Scale Linear Programs (LP)

We assume that the reader is familiar with the theory of Linear Programming (LP). Nevertheless, the following provides a brief overview of the concepts used. Details and proofs in particular are omitted however and the interested reader is referred to the following books [Van96, BT97, BV04].

2.1 Linear Programs

Simply stated, a linear program is a mathematical optimization problem in which both the objective function and the constraints are linear functions. Many problems arising in practice can be formulated in this fashion. Examples are network problems such as telecommunication, portfolio optimization, or resource allocation. This widely applicable use is one of the main reasons for the enormous interest in linear programming.

A compact formulation of an LP is as follows

$$\begin{array}{lll} \text{(LP)} & \text{minimize} & c^T x \\ & \text{subject to} & Ax \leq b \end{array}$$

where $c \in \mathbb{R}^n, b \in \mathbb{R}^m$, and $A \in \mathbb{R}^{m \times n}$. Each LP has an associated dual problem, which is formulated as follows

$$\begin{array}{lll} \text{(Dual LP)} & \text{maximize} & -b^T z \\ & \text{subject to} & A^T z + c = 0. \\ & & z \geq 0 \end{array}$$

The relation of the optimal solutions of the two problems is given by the weak and strong duality theorems.

Theorem 2.1 (Weak duality — Theorem 4.3 in [BT97])
If x is primal feasible and z dual feasible, then $c^T x \geq -b^T z$.

Thus, a duality gap can be defined as $c^T x + b^T z$. By defining the optimal value of the LP by $p^\star$ and the optimal value of the dual LP by $d^\star$, we can state the strong duality theorem.

Theorem 2.2 (Strong duality — Theorem 4.4 in [BT97])
If either the LP or its dual has a bounded feasible solution, then the other also has a bounded feasible solution and $p^\star = d^\star$.

One can prove the optimality of a pair of primal and dual solutions $(x^\star, z^\star)$ using the so-called Karush-Kuhn-Tucker (KKT) conditions.

Theorem 2.3 (KKT Conditions — Section 5.5.3 in [BV04])
$x^\star$ *and* $z^\star$ *are primal respectively dual optimal solutions if and only if*

$$\begin{array}{lll} 1) & Ax^\star \leq b & \textit{(feasibility of } x^\star\textit{)} \\ 2) & A^T z^\star + c = 0, z^\star \geq 0 & \textit{(feasibility of } z^\star\textit{)} \\ 3) & (z^\star)^T(Ax^\star - b) = 0 & \textit{(complementarity)} \end{array}$$

The feasible region of an LP is given by a polyhedron since each constraint that the decision variable is subject to, defines a hyperplane. The following theorem states an interesting fact about the location of optimal solutions. Note that an extreme point of a polyhedron is by definition a point, which cannot be expressed as a convex combination of other points in the polyhedron, i.e. a vertex.

Theorem 2.4 (Theorem 2.8 in [BT97])
If the feasible region of an LP contains at least one extreme point, the optimal objective value is either unbounded or is attained at an extreme point.

Both exact and approximation algorithms for solving linear programs exist. An overview is given in the next sections.

2.2 Exact Optimization Methods

Several algorithms exist which can solve linear programs exactly. Here, two of the most widely used methods shall be briefly described, namely the Simplex and the Interior Point methods.

The Simplex method was introduced in 1947 by Dantzig [Dan63]. It starts with an extreme point of the polyhedron that defines the feasible region of the LP and then moves in every iteration of the algorithm to an adjacent extreme point with improved objective value. Combining Theorem 2.4 with the fact that the number of extreme points is finite guarantees termination at an optimal solution (though precautions against cycling are needed if the polyhedron is degenerate).

A major drawback of the Simplex method is given by the fact that the number of extreme points of polyhedrons grows exponentially with the number of constraints. Klee and Minty showed an example of an LP, for which the Simplex method passed

all extreme points before reaching the optimal solution, thus proving that the worst case performance of the algorithm is exponential [KM72]. However, the Simplex method proves to be efficient in most real life applications of modest size.

Whether the linear programming problem could be solved in polynomial time remained an open question until 1979, when a method was presented that had polynomial worst case running time (Khachian's Ellipsoid method [Kha79]. However, it compared poorly to the Simplex method in real life applications.

The first algorithm having a polynomial worst case running time and showing good performance in practical applications was the Interior Point method presented by Karmakar in 1984 [Kar84]. The method starts from an interior point of the feasible region and then follows a "central" path given by adding a logarithmic barrier function to the objective function towards an optimal point. Newton's method is applied to follow the central path. Finally, when a point sufficiently near an optimal extreme point is found, it can be rounded in a polynomial number of steps to an exact solution. The theoretical worst case running time is given by $O(Ln\ln(n))$, where n is the dimension of the problem space and L is the length of the input data, i.e., A, b, and c.

2.3 Decomposition Methods

We are interested in large scale linear programs, i.e. problems containing millions of decision variables. In this case it is usually no longer possible to solve the problem exactly because both running time and memory usage become prohibitively large. Most frequently used approaches are then decomposing the problem into subproblems or relaxing some constraints. An overview is given in the following subsections.

2.3.1 Dantzig-Wolfe Decomposition and Bender's Decomposition

In order to improve problem tractability, decomposition methods reduce either the number of variables or the number of constraints to be considered in an algorithmic step. The basic idea of both Dantzig-Wolfe and Bender's decomposition is to reformulate the linear program into a suitable master-problem and then iteratively solve subproblems to improve the current solution of the master-problem.

The Dantzig-Wolfe decomposition is applied if the linear program has an excessive number of variables, i.e. columns in the tableau of the LP. The method starts from a basic solution of the master-problem and solves linear subproblems in order to find variables, which must enter the basic solution of the master-problem in order to

improve the objective function value. Thus, variables enter the solution iteratively, which is called "delayed column generation".

Bender's decomposition is applied if the LP has a large number of constraints. This method starts with a solution of a relaxed master-problem, in which only a part of the constraints are included. Subproblems are then applied to find violated constraints, which must enter the relaxed master-problem. This approach is called "delayed constraint generation".

Note, that both methods are very similar in the sense that Bender's decomposition is essentially the same as Dantzig-Wolfe decomposition applied to the dual problem.

2.3.2 Lagrange Relaxation

The Lagrange relaxation is a technique that enables us to transfer some constraints into the objective function. Assume that the constraints of a given LP can be divided into "easy" and "hard" ones. We then rewrite the problem in the following fashion.

$$\begin{array}{lll} \text{(LP)} & \text{minimize} & c^T x \\ & \text{subject to} & Ax \leq b \\ & & x \in Q \end{array}$$

where $c \in \mathbb{R}^n, b \in \mathbb{R}^m, A \in \mathbb{R}^{m \times n}$, $Q \subseteq \mathbb{R}^n$ is a sufficiently simple non-empty polyhedron, e.g., a simplex or box. Thus, the "easy" constraints define Q. The Lagrange relaxation of the inequality constraints of this problem is as follows.

$$\text{(LR)} \quad \max_{u \geq 0} \ \psi(u) \qquad \psi(u) := \min_{x \in Q} \{c^T x + u^T (Ax - b)\}.$$

The problem LR is called the *Lagrange dual* problem. Note that the dual LP is obtained if $Q = \mathbb{R}^n$. If we suppose that the linear problem, LP, has a finite optimal solution, solving the LP problem is equivalent to solving its Lagrange relaxation (LR) as the following theorem shows.

Theorem 2.5
Consider the linear minimization problem LP and its Lagrange relaxation LR,

$$LP := \begin{array}{ll} \min & c^T x \\ & Ax \leq b \\ & x \in Q \end{array} \qquad LR := \max_{u \geq 0} \left\{ \min_{x \in Q} \{c^T x + u^T (Ax - b)\} \right\}$$

where $Q := \{x \mid \tilde{A}x \leq \tilde{b}\}$ is a polyhedron. If the linear minimization problem has a finite optimal solution, then $LP = LR$.

A proof is given in Appendix A, Theorem A.1.

3. Theoretical Foundations

In this chapter we give a definition of the problem, which will be investigated in subsequent chapters, and introduce strongly convex functions. These functions play an important part in the optimization methods and thus some of their basic properties are also investigated.

3.1 Problem Description

We consider specially structured large scale linear programs, which can be formulated as follows

$$\begin{array}{lll} \text{(LP)} & \text{minimize} & c^T x \\ & \text{subject to} & Ax \leq b \\ & & x \in Q \end{array}$$

where $c \in \mathbb{R}^n, b \in \mathbb{R}^m, A \in \mathbb{R}^{m \times n}$, $Q \subset \mathbb{R}^n$ is a sufficiently simple non-empty polyhedron, e.g., a simplex, and n and m are in the order of millions. Then, we consider the Lagrange relaxation of the inequality constraints,

$$\text{(LR)} \quad \max_{u \geq 0} \ \psi(u) \qquad \psi(u) := \min_{x \in Q} \{c^T x + u^T (Ax - b)\}.$$

and we assume in addition that Q is bounded and that a non-empty, convex and compact polytope $P \subset \mathbb{R}^m$ containing an optimal solution of the Lagrange relaxation LR exists. Thus,

$$\max_{u \in P} \ \psi(u) = \min_{x \in Q} \ f(x) \tag{3.1}$$

where $f(x) := \max_{u \in P} \{c^T x + u^T (Ax - b)\}$, see Theorem A.3 or Corollary 37.3.2 in [Roc70]. The function $\psi(u)$ is a concave piecewise linear function and the function $f(x)$ is a convex piecewise linear function. Hence, both functions are non-differentiable. Our objective is to solve (3.1) approximately, i.e., to find $\bar{x} \in Q$ and $\bar{u} \in P$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon, \ \epsilon > 0$.

Note, however, that bounding the polyhedron Q and finding a non-empty, convex and compact polytope P containing an optimal solution of LR is not trivial. The choice of Q and P influences the tractability of the implemented methods.

Let us present the main assumptions that we are making as well as the problem we are considering.

Assumption 1

- $Q \subset \mathbb{R}^n$ is a non-empty, convex and compact polytope,
- $P \subset \mathbb{R}^m$ is a non-empty, convex and compact polytope.

As a consequence

- the function $f(x) := \max_{u \in P} \{c^T x + u^T(Ax - b)\}$ is well defined for all $x \in Q$,
- the function $\psi(u) := \min_{x \in Q} \{c^T x + u^T(Ax - b)\}$ is well defined for all $u \in P$,
- the function $f(x)$ has a finite minimum f^* over Q attained at $x^* \in Q$,
- the function $\psi(u)$ has a finite maximum ψ^* over P attained at $u^* \in P$.

We note that under Assumption 1 there is no duality gap.

Problem 1 Suppose Assumption 1 holds. Given $\epsilon > 0$, find a feasible primal solution $\bar{x} \in Q$ and a feasible dual solution $\bar{u} \in P$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

The methods presented in Chapters 4 and 5 deal with convex non-smooth optimization problems. They were developed by Yurii Nesterov between 2001 and 2007, see [Nes05c], [Nes05a], [Nes09], and [Nes05b]. We use them here in order to solve simultaneously the maximization and minimization problems in Equation (3.1). First, we consider optimization methods based on subgradient techniques (Chapter 4). Next, we consider methods based on gradient optimization (Chapter 5), which consist mainly of two steps. First, a smooth approximation of the objective function is derived, and then gradient-based optimization methods are applied. In the following chapters, we will follow the lines of Nesterov in [Nes05c], [Nes05a], [Nes09], and [Nes05b] to introduce these methods.

3.2 Strongly Convex Functions

In the following we introduce some basic definitions that are used throughout the text.

Definition 3.1 (Convex and Closed Convex Function)
Let $S \subseteq \mathbb{R}^n$ be a convex set and $g : S \longrightarrow \mathbb{R}$. The set

$$\text{epi } g := \{(x,t) \mid x \in S, t \in \mathbb{R}, g(x) \leq t\} \subset \mathbb{R}^{n+1} \tag{3.2}$$

is called the epigraph *of the function g. The function g is* convex *if its epigraph,* epi g*, is a convex set. Moreover, if the epigraph is a closed convex set, the function g is called* closed convex.

Note that the convexity of a function $g : S \longrightarrow \mathbb{R}$, over its convex domain $S \subseteq \mathbb{R}^n$ is equivalent to satisfy

$$g((1-\theta)x + \theta y) \leq (1-\theta)g(x) + \theta g(y) \qquad 0 < \theta < 1 \tag{3.3}$$

for every $x, y \in S$, Theorem 4.1 in [Roc70].

Definition 3.2 (Smooth Function and Lipschitz Continuous Gradient)
Let $S \subseteq \mathbb{R}^n$, with $\mathbb{R}^n$ endowed with a norm $\|.\|_S$. The function $g : S \longrightarrow \mathbb{R}$ is called smooth on S *if it is finite and differentiable throughout S.*

The gradient of $\nabla g(.)$ is Lipschitz continuous *on S, if*

$$\exists\ L_{g,S} > 0 \ \textit{s.t.}\ \|\nabla g(x) - \nabla g(y)\|_S^* \leq L_{g,S}\|x-y\|_S \quad \forall\ x, y \in S, \tag{3.4}$$

where the dual norm is defined by $\|\zeta\|_S^* := \max_{\|x\|_S \leq 1}\{\langle \zeta, x\rangle\}$ for $\zeta \in \mathbb{R}^n$. The constant $L_{g,S}$ is called Lipschitz constant *of ∇g with respect to $\|.\|_S$.*

For the sake of clarity we use the following notation. Our main space is $\mathbb{R}^n$ and we consider a set $S \subseteq \mathbb{R}^n$. Any parameter whose value depends on the subset S will have S as index, e.g., the Lipschitz constant $L_{g,S}$ in the previous definition. The same notation will be used for the norms.

Note that for any smooth convex function $g : S \longrightarrow \mathbb{R}$, with a convex domain $S \subseteq \mathbb{R}^n$,

$$g(y) \geq g(x) + \langle \nabla g(x), y - x\rangle \tag{3.5}$$

holds, see e.g. Definition 2.1.1 and Theorem 2.1.2 in [Nes03]. If the gradient of g is also Lipschitz continuous, then g has the following nice property.

Theorem 3.3
Let $S \subseteq \mathbb{R}^n$ be convex and let $\mathbb{R}^n$ be endowed with a norm $\|.\|_S$. Let $g : S \longrightarrow R$ be a smooth convex function, such that its gradient is Lipschitz continuous with Lipschitz constant $L_{g,S}$ with respect to $\|.\|_S$. Then, for all $x, y \in S$

$$g(y) \leq g(x) + \langle \nabla g(x), y - x\rangle + \frac{L_{g,S}}{2}\|y-x\|_S^2 \tag{3.6}$$

holds.

For a proof see e.g. Theorem 2.1.5 in [Nes03].

Definition 3.4 (Smooth Strongly Convex Function)
Let S be a convex subset of $\mathbb{R}^n$ and let $\mathbb{R}^n$ be endowed with a norm $\|.\|_S$. A smooth function $g : S \longrightarrow \mathbb{R}$ is called strongly convex *with* convexity parameter $\sigma_S > 0$ *for $\|.\|_S$ if*

$$g(y) \geq g(x) + \langle \nabla g(x), y - x \rangle + \frac{1}{2}\sigma_S \|y - x\|_S^2 \quad \forall\, y, x \in S. \tag{3.7}$$

A typical smooth strongly convex function over $\mathbb{R}^n$ related to the Euclidean norm is the squared Euclidean norm,

$$g(x) := \frac{1}{2}\|x\|_2^2 = \frac{1}{2}\left(\sum_{i=1}^{n} x_i^2\right).$$

In this case, the convexity parameter σ is equal to 1 with respect to $\|.\|_2$.

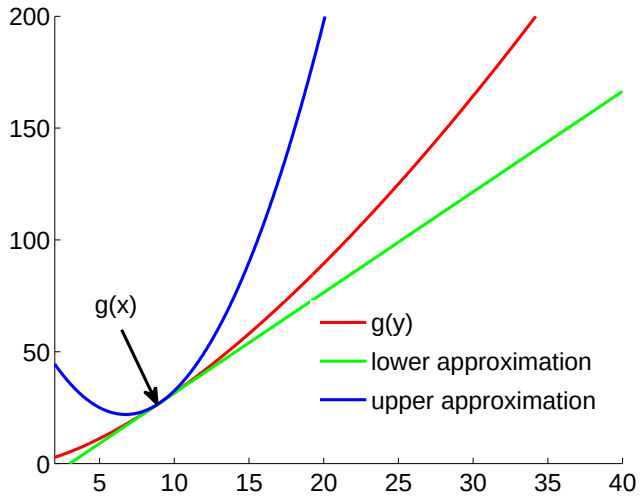


Fig. 3.1: Smooth convex function $g(x)$ with lower approximation, $g(x) + \langle \nabla g(x), y - x \rangle$, and upper approximation, $g(x) + \langle \nabla g(x), y - x \rangle + \frac{L_{g,S}}{2}\|y - x\|_S$, at x

A crucial property of a strongly convex function is the uniqueness of its minimizer.

Theorem 3.5 (First Order Condition)
Let $S \subset \mathbb{R}^n$ be a closed convex set and let $g : S \longrightarrow \mathbb{R}$ be a smooth convex function. Consider the minimization problem

$$\min_{x \in S} \; g(x). \tag{3.8}$$

a) $x^* \in S$ *is an optimal solution of (3.8) if and only if*

$$\langle \nabla g(x^*), x - x^* \rangle \geq 0 \quad \forall\ x \in S. \tag{3.9}$$

b) If g *is strongly convex then the optimal solution* $x^* \in S$ *of (3.8) exists and is unique.*

The reader is referred to Theorem 2.2.5 and 2.2.6 in [Nes03] for a proof.

The methods that we present in this chapter are oracle based, i.e., they assume that specific subproblems can be solved to optimality, in the sense that an optimal solution can be cheaply computed. The optimal solution of these subproblems are needed in order to evaluate special function values and their gradients. These special functions are of the following type

$$g(x) := \hat{g}(x) + \max_{u \in T} \{\langle Bx, u \rangle - \hat{\varphi}(u) - \beta d_T(u)\} \qquad \forall\ x \in S \tag{3.10}$$

where $S \subset \mathbb{R}^n$ and $T \subset \mathbb{R}^m$ are convex and compact sets and $\beta > 0$. Both spaces, R^n and R^m, are endowed with a norm, respectively $\|.\|_S$ and $\|.\|_T$. The function $\hat{g} : S \to \mathbb{R}$ is smooth and convex with a Lipschitz continuous gradient with Lipschitz constant $L_{\hat{g},S}$ with respect to $\|.\|_S$. The function $\hat{\varphi} : T \to \mathbb{R}$ is also smooth and convex with a Lipschitz continuous gradient with Lipschitz constant $L_{\hat{\varphi},T}$ with respect to $\|.\|_T$. $B \in \mathbb{R}^{m \times n}$ is a linear operator and its norm is defined as

$$\|B\|_{S,T} := \max_{\|x\|_S \leq 1} \max_{\|u\|_T \leq 1} \langle Bx, u \rangle. \tag{3.11}$$

Finally, $d_T : T \to \mathbb{R}_+$ is a prox-function of T, i.e., a strongly convex function with special properties, see Definition 3.6. Its convexity parameter is denoted by σ_T and its minimizer over T by u^o.

Definition 3.6 (Prox-Function)
Let S *be a convex subset of* $\mathbb{R}^n$ *and let us endow* R^n *with a norm* $\|.\|_S$*. The function* $d_S : S \longrightarrow \mathbb{R}_+$ *is a* prox-function *associated to* S *if it is a smooth strongly convex function over* S *with convexity parameter* $\sigma_S > 0$ *with respect to norm* $\|.\|_S$*. Moreover an* $x^o \in ri\ S$ *exists, such that* $d_S(x^o) = 0$*.*

For clarity we introduce the auxiliary function Γ_β,

$$\begin{aligned} \Gamma_\beta : S \times T &\to \mathbb{R} \\ (x, u) &\mapsto \langle Bx, u \rangle - \hat{\varphi}(u) - \beta d_T(u). \end{aligned} \tag{3.12}$$

Hence for all $x \in S$, $g(x) = \hat{g}(x) + \max_{u \in T} \Gamma_\beta(x, u)$.

For all fixed $u \in T$, the function $\Gamma_\beta(., u)$ is convex and smooth. Its gradient at $x \in S$ is given by

$$\nabla_x \Gamma_\beta(x, u) = B^T u.$$

For all fixed $x \in S$, the function $\Gamma_\beta(x, .)$ is smooth and its gradient at $u \in T$ is

$$\nabla_u \Gamma_\beta(x, u) = Bx - \nabla\hat{\varphi}(u) - \beta\nabla d_T(u).$$

$-\Gamma_\beta(x, .)$ is strongly convex with convexity parameter $\beta\sigma_T$ with respect to the norm $\|.\|_T$. Therefore, the maximum u_x,

$$u_x := \arg \max_{u \in T} \Gamma_\beta(x, u), \tag{3.13}$$

is unique.

Theorem 3.7 (Properties of $g(x)$ — Theorem 1 in [Nes05c])
Consider the function $g(x)$ defined in (3.10). Then, $g(x)$ has the following properties.
$g(x)$ is a convex and smooth function and its gradient $\nabla g(x)$ is Lipschitz continuous with Lipschitz constant $L_{g,S}$, i.e.,

$$\nabla g(x) = \nabla\hat{g}(x) + B^T u_x \qquad L_{g,S} = L_{\hat{g},S} + \frac{\|B\|^2_{S,T}}{\beta\sigma_T} \tag{3.14}$$

where $u_x := \arg \max_{u \in T} \{\langle Bx, u\rangle - \hat{\varphi}(u) - \beta d_T(u)\}$.

Proof. The convexity of g follows from the convexity of $\hat{g}$ and of $\Gamma_\beta(., u)$ for all $u \in T$ (see (3.12)). Its differentiability is due to the uniqueness of the maximizer of $\Gamma_\beta(x, .)$ over T. Namely, for fixed x, the function $\Gamma_\beta(x, .)$ is strongly concave. Thus, the gradient of g at x is then defined by the sum of the gradient of $\hat{g}$ at x and the gradient of $\Gamma_\beta(., u_x)$ at x, where u_x is the unique maximizer of $\Gamma_\beta(x, .)$. For a detailed proof see Theorem A.6 in Appendix A.

The value of the Lipschitz constant, $L_{g,S}$, remains to be evaluated. For $x, y \in S$ we have

$$\begin{aligned} \|\nabla g(x) - \nabla g(y)\|^*_S &= \|\nabla\hat{g}(x) - \nabla\hat{g}(y) + B^T u_x - B^T u_y\|^*_S \\ &\leq \|\nabla\hat{g}(x) - \nabla\hat{g}(y)\|^*_S + \|B^T(u_x - u_y)\|^*_S. \end{aligned}$$

Thus we need to bound $\|B^T(u_x - u_y)\|^*_S$. Using the first order condition for $\Gamma_\beta(x, .)$ at u_x and for $\Gamma_\beta(y, .)$ at u_y, i.e.,

$$\begin{aligned} \langle Bx - \nabla\hat{\varphi}(u_x) - \beta\nabla d_T(u_x), u_y - u_x\rangle &\leq 0 \\ \langle By - \nabla\hat{\varphi}(u_y) - \beta\nabla d_T(u_y), u_x - u_y\rangle &\leq 0 \end{aligned}$$

we get

$$\begin{aligned} \langle B(x - y), u_x - u_y\rangle &\geq \langle\nabla\hat{\varphi}(u_x) - \nabla\hat{\varphi}(u_y), u_x - u_y\rangle \\ &\quad + \beta\langle\nabla d_T(u_x) - \nabla d_T(u_y), u_x - u_y\rangle \\ &\geq \beta\langle\nabla d_T(u_x) - \nabla d_T(u_y), u_x - u_y\rangle \\ &\geq \beta\sigma_T\|u_x - u_y\|^2_T. \end{aligned}$$

We used the convexity of $\hat{\varphi}(u)$ to prove the second inequality and the strong convexity of $d_T(u)$ to prove the third inequality. Hence,

$$\begin{aligned}\left(\|B^T(u_x-u_y)\|_S^*\right)^2 &\leq \|B\|_{S,T}^2\|u_x-u_y\|_T^2\\ &\leq \|B\|_{S,T}^2\frac{1}{\beta\sigma_T}\langle B(x-y),u_x-u_y\rangle\\ &\leq \frac{\|B\|_{S,T}^2}{\beta\sigma_T}\|B^T(u_x-u_y)\|_S^*\|x-y\|_S.\end{aligned}$$

Finally, $\|\nabla g(x)-\nabla g(y)\|_S^* \leq \left(L_{\hat{g},S}+\frac{\|B\|_{S,T}^2}{\beta\sigma_T}\right)\|x-y\|_S$. □

3.3 From an Absolute to a Relative Error

The methods in Chapters 4, 5, and 6 are meant for computing ϵ-approximate solutions, $\epsilon > 0$, i.e., solutions that guarantee an absolute optimality gap of at most ϵ. Here we present a procedure to generate solutions that ensure a *relative* error of at most $\epsilon' > 0$, if an algorithm delivering solutions with a guaranteed absolute error of ϵ is available for specially structured large scale linear problems. This approach has been presented in the primary paper [CE05]. However, note that it has been followed before, see [You01], [Bie02], [GK02] and [BI04]. In a paper of the same flavor as [CE05], Chudak and Nagano, [CN07], use the same approach, albeit proofs are harder than in [CE05].

Under Assumption 1, we define the following problem.

Problem 2 Given $\epsilon' > 0$, find a feasible primal solution $\bar{x} \in Q$ such that $f(\bar{x}) \leq (1+\epsilon') \cdot f^*$. We call $\bar{x}$ an ϵ'-*relative-approximate* solution.

The procedure for solving Problem 2 consists of three steps.

1. Modeling.
2. Developing a method for solving the LP within an absolute error of ϵ, $\epsilon > 0$.
3. Using binary search to obtain a relative error of $(1+\epsilon')$, $\epsilon' > 0$, for the LP.

Step 1 is the most important and difficult since it has to take into account the tractability of the following steps. Recall that solving our original structured linear

problem is equivalent to solving its Lagrange relaxation $\max_{u \geq 0} \psi(u)$, with $\psi(u) := \min_{x \in Q} \{c^T x + u^T(Ax - b)\}$ and $\min_{x \in Q} f(x)$, with $f(x) := \max_{u \geq 0} \{c^T x + u^T(Ax - b)\}$.

In Step 2 we design a decision procedure, denoted by A-FEAS, that given an error $\epsilon > 0$ and a target $R > 0$, either finds a feasible solution x such that $f(x) < (1+\epsilon)R$ or concludes that $f^* > R$. In order to achieve that, methods with a running time that is a polynomial of the input size times a polynomial of $\frac{1}{\epsilon}$ times R, have to be available. Note that the choice of R requires a good knowledge of the problem.

Step 3 follows from a result of Young ([You01]) where the important problem is finding a good enough initial feasible solution. Suppose that we have a decision procedure A-FEAS, whose running time depends on the size of the input and a polynomial on $\frac{1}{\epsilon}$ (but *not* on R). The following lemma provides us with a method for getting a relative approximate solution using A-FEAS.

Lemma 3.8 ([You01])
Suppose that we know a feasible solution x and a lower bound LB such that $\text{LB} \leq f^* \leq f(x) \leq l \text{ LB}$*, for some $l > 0$, then we can find a feasible solution $\bar{x}$ such that $f(\bar{x}) \leq (1 + \epsilon')f^*$, $\epsilon' > 0$, by running A-FEAS $O(\log \log l)$ times with $\epsilon = \frac{1}{2}$ and an additional number of A-FEAS runs whose overall running time is $O(1)$ times the time necessary to run once A-FEAS with $\epsilon = \epsilon'$.*

In the second part of this thesis, we apply this procedure to generate polynomial time approximation schemes for the linear programming relaxation of the Uncapacitated Facility Location Problem. We will study them not only from a theoretical point of view but also from a computational point of view.

4. Primal-Dual Subgradient Method

The Primal-Dual Subgradient method developed by Nesterov in [Nes09] is a simple but ingenious variation of the standard subgradient method. The main advantage of the method is that it is a primal-dual method. Namely, at each step an approximate solution for the considered non-smooth minimization problem as well as an approximate solution for a dual problem are computed. Thus, a dual gap can be evaluated at each step and used as stopping criteria.

We first introduce the definition of a subgradient of a convex function as well as the type of problems we are interested in and a few necessary assumptions.

Definition 4.1 (Subgradient)
Let g be a convex function. A vector ξ is called a subgradient of g at $\tilde{x} \in dom\ g$ if

$$g(x) \geq g(\tilde{x}) + \langle \xi, x - \tilde{x} \rangle \quad \forall\ x \in dom\ g. \tag{4.1}$$

The subset of all subgradients at $\tilde{x}$ is denoted by $\partial g(\tilde{x})$ and is called the subdifferential of g at $\tilde{x}$.

Assumption 2

- $S \subseteq \mathbb{R}^n$ is a non-empty, convex and compact set,
- $g : S \longrightarrow \mathbb{R}$ is a closed, finite and convex function over S, its minimum, g^*, is attained at $x^* \in S$,
- the subgradients of g are bounded over S, i.e.,

$$\exists\ L > 0 \text{ such that } \forall\ x \in S,\ \forall\ \xi \in \partial g(x) \quad \|\xi\|_S^* \leq L \tag{4.2}$$

  where $\|.\|_S$ is a norm defined over $\mathbb{R}^n$ and $\|.\|_S^*$ is its dual norm.

Problem 3 Suppose Assumption 2 holds. Given $\epsilon > 0$, find a feasible primal solution $\bar{x} \in S$ such that $g(\bar{x}) - g^* \leq \epsilon$. We call $\bar{x}$ an ϵ-approximate solution.

Here, we focus on the presentation of the Primal-Dual Subgradient methods introduced in [Nes09] and [Nes05b] to solve Problem 3. We will also present the important definitions and results of standard subgradient techniques. A comprehensive survey of the standard subgradient techniques applied to Problem 3 can be found in [Pol87, Ber95, Nes03].

4.1 Standard Subgradient Method

The main objects used in the standard subgradient method are the subgradient and the Euclidean projection. The calculation of both objects is assumed to be tractable and numerically cheap.

Definition 4.2 (Euclidean Projection)
Let $S \subseteq \mathbb{R}^n$ be a closed convex set with non-empty interior. For $y \in \mathbb{R}^n$, the Euclidean projection of y onto S is denoted by $\pi_S(y)$ and defined as follows

$$\pi_S(y) = \arg \min_{x \in S} \|y - x\|_2. \tag{4.3}$$

Lemma 4.3 (Euclidean Projection Property)
Let $S \subseteq \mathbb{R}^n$ be a closed convex set with non-empty interior. For any $y \in \mathbb{R}^n$ and for any $x \in S$

$$\|\pi_S(y) - x\|_2 \leq \|y - x\|_2 \tag{4.4}$$

holds.

For a proof see e.g. Lemma 3.1.4 and 3.1.5 in [Nes03].

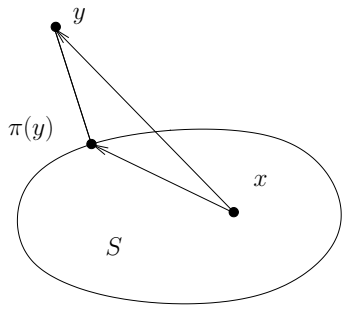


Fig. 4.1: Euclidean Projection Property

In the following, we introduce the standard subgradient methods. In addition to Assumption 2, we assume that we know a point $x_0 \in S$ and a Euclidean ball of radius R around x_0 containing the set S. Consider Algorithm 1, which corresponds to the standard subgradient method. We note that the difficult operations in the

algorithm are the computation of the subgradient, which depends on the complexity of $g(x)$, and the computation of the Euclidean projection, which depends on the complexity of the set S.

Algorithm 1 Standard Subgradient Algorithm

Requires: - An initial point $x_0 \in S$
- A constant $R > 0$ such that $\|x - x_0\|_2 \leq R \quad \forall x \in S$
- A constant $L > 0$ such that $\|\xi\|_2 \leq L \quad \forall \xi \in \partial g(x) \quad \forall x \in S$
- A maximal number of iterations N

Ensures: An approximate solution $\bar{x} \in S$ such that $g(\bar{x}) - g^* \leq \frac{RL}{\sqrt{N+1}}$.

set step size $h = \frac{1}{\sqrt{N+1}} \frac{R}{L}$
while $k \leq N$ **do**
  compute $\xi_k \in \partial g(x_k)$ `Subgradient computation`
  set $y_k = x_k - h\xi_k$
  compute $x_{k+1} = \arg \min_{x \in S} \|y_k - x\|_2$ `Euclidean projection`
end while
$\bar{x} = \frac{1}{N+1} \sum_{i=0}^{N} x_k$

The convergence result for the standard subgradient algorithm is as follows.

Theorem 4.4 (Convergence of Standard Subgradient Algorithm—Theorem 3.2.2 in [Nes03])
Suppose that Assumption 2 holds for the Euclidean norm, $\|.\|_S = \|.\|_2$. Assume also that a constant $R > 0$ and an initial point x_0 exist such that $\|x - x_0\|_2 \leq R$ for all $x \in S$ and consider Problem 3. Finally let $\{x_k\}_{k=0}^{N}$ be the sequence of points in S generated by Algorithm 1 and define $\bar{x} := \frac{1}{N+1} \sum_{k=0}^{N} x_k$.

Then,

$$g(\bar{x}) - g^* \leq \frac{RL}{\sqrt{N+1}}. \tag{4.5}$$

Proof. Consider the following lower linear approximation of $g(x)$,

$$l_N(x) = \frac{1}{N+1} \left(\sum_{k=0}^{N} g(x_k) + \langle \xi_k, x - x_k \rangle \right)$$

and define $l_N^* := \min_{x \in S} l_N(x)$. We observe that $l_N^* \leq g^*$. Then,

$$\begin{aligned} g(\bar{x}) - g^* &\leq g(\bar{x}) - l_N^* \\ &\leq \frac{1}{N+1}\left(\sum_{k=0}^{N} g(x_k) - \min_{x \in S}\Big\{\sum_{k=0}^{N} g(x_k) + \langle \xi_k, x - x_k\rangle\Big\}\right) \\ &= \frac{1}{N+1}\max_{x \in S}\Big\{\sum_{k=0}^{N}\langle \xi_k, x_k - x\rangle\Big\}. \end{aligned} \tag{4.6}$$

In order to bound (4.6), we define $r_k = \|x_k - x\|_2$ for all $x \in S$ and evaluate

$$\begin{aligned} r_{k+1}^2 - r_k^2 &= \|x_{k+1} - x\|_2^2 - \|x_k - x\|_2^2 \\ &\leq \|x_k - h\xi_k - x\|_2^2 - \|x_k - x\|_2^2 \qquad \text{(Lemma 4.3)} \\ &\leq h^2\|\xi_k\|_2^2 - 2h\langle x_k - x, \xi_k\rangle. \end{aligned}$$

Then

$$r_{N+1}^2 - r_0^2 = \sum_{k=0}^{N}(r_{k+1}^2 - r_k^2) \leq h^2\sum_{k=0}^{N}\|\xi_k\|_2^2 - 2h\sum_{k=0}^{N}\langle x_k - x, \xi_k\rangle,$$

and

$$0 \leq r_{N+1}^2 \leq R^2 + \frac{R^2}{(N+1)L^2}(N+1)L^2 - 2\frac{R}{\sqrt{N+1}L}\sum_{k=0}^{N}\langle x_k - x, \xi_k\rangle,$$

since $r_0 \leq R$. We deduce

$$\sum_{k=0}^{N}\langle x_k - x, \xi_k\rangle \leq RL\sqrt{N+1},$$

and, from (4.6)

$$g(\bar{x}) - g^* \leq \frac{RL}{\sqrt{N+1}}.$$

□

We note that the convergence rate of the standard subgradient method does not depend directly on n, the dimension of the problem. However for an absolute error of ϵ, we need to compute $O(\frac{1}{\epsilon^2})$ iterations, i.e., $O(\frac{1}{\epsilon^2})$ subgradients and Euclidean projections. A natural question is whether we can find a method with a better convergence-rate dependency on ϵ using the same tools, i.e subgradients and projections. The answer is no. Namely, Nesterov shows in [Nes03] (Theorem 3.2.1) that for $n \geq \frac{1}{\epsilon^2}$, instances of Problem 3 that require at least $\Omega(\frac{1}{\epsilon^2})$ iterations to achieve an ϵ absolute accuracy, exist. Thus, the standard subgradient technique is optimal for Problem 3. Note that the latter result was first published by Nesterov and Yudin in [NY83]

4.2 Primal-Dual Subgradient Method

In spite of the fact that the convergence rate dependency on ϵ cannot be improved, Nesterov provides a refinement of the standard subgradient method with the primal-dual subgradient method. Namely, the latter does not require the number of iterations to be fixed in advance. Instead, it uses an absolute gap calculation as a stopping criterion.

Recall that $g : S \longrightarrow \mathbb{R}$ is a closed, finite, and convex function over S, its minimum g^* is finite and attained at $x^* \in S$. Let us extend the domain of g to $\mathbb{R}^n$ by setting $g(x) = \infty$ for $x \notin S$ and consider its conjugate function,

$$g_*(\zeta) := \sup_{x \in \mathbb{R}^n} \{\langle \zeta, x \rangle - g(x)\}. \tag{4.7}$$

In general the following inequality holds between a convex function and its conjugate,

$$g_*(\xi) + g(x) \geq \langle \xi, x \rangle \qquad \forall\ x \in S,\ \forall\ \xi \in \text{dom } g_*.$$

This inequality is called *Fenchel's Inequality*. For every fixed $x \in \text{dom } g$, the equality

$$g_*(\xi) + g(x) = \langle \xi, x \rangle \tag{4.8}$$

holds, if and only if $\xi \in \partial g(x)$ (Theorem 23.5 in [Roc70]).

Now, recall that the function g is assumed to be convex and closed. Thus, we have $(g_*)_* = g$, i.e,

$$g(x) = \sup_{\zeta \in \text{dom } g_*} \{\langle x, \zeta \rangle - g_*(\zeta)\} = \max_{\zeta \in \text{dom } g_*} \{\langle x, \zeta \rangle - g_*(\zeta)\}.$$

The last equality holds since for any $x \in S$, the previous supremum is attained at $\tilde{\zeta} \in \partial g(x)$. Namely, using Equality (4.8), we have

$$g(x) = \sup_{\zeta \in \text{dom } g_*} \{\langle x, \zeta \rangle - g_*(\zeta)\} = \langle x, \tilde{\zeta} \rangle - g_*(\tilde{\zeta}).$$

Thus we can write our optimization problem as follows

$$\begin{aligned} g^* = \min_{x \in S} g(x) &= \min_{x \in S} \max_{\zeta \in \text{dom } g_*} \{\langle x, \zeta \rangle - g_*(\zeta)\} \\ &= \max_{\zeta \in \text{dom } g_*} \min_{x \in S} \{\langle x, \zeta \rangle - g_*(\zeta)\} \\ &\quad \text{(Theorem A.3 or Corollary 37.3.2 in [Roc70])} \\ &= \max_{\zeta \in \text{dom } g_*} \{-g_*(\zeta) + \min_{x \in S} \langle \zeta, x \rangle\} \\ &= \max_{\zeta \in \text{dom } g_*} \varphi(\zeta). \end{aligned}$$

where

$$\varphi(\zeta) := -g_*(\zeta) + \min_{x \in S} \langle \zeta, x \rangle \tag{4.9}$$

and define the general dual problem as

$$\max_{\zeta \in \text{dom } g_*} \varphi(\zeta). \tag{4.10}$$

The main idea of the method is to use the computed subgradients in order to determine an approximate solution for the dual problem as well as an approximate solution for the primal problem. Moreover, the method retains all computed gradients during the process in order to create a smooth convex function whose minimizer converges to a minimizer of the objective function.

Algorithm 2 Primal-Dual Subgradient Algorithm - Dual Averaging Algorithm (DA)

Requires: - A prox-function $d_S(x)$ over S with convexity parameter $\sigma_S > 0$ with respect to norm $\|.\|_S$ and minimizer x^o over S
- An absolute error $\epsilon > 0$

Ensures: An approximate primal solution $\bar{x} \in S$ and an approximate dual solution $\bar{\zeta}$ such that $g(\bar{x}) - \varphi(\bar{\zeta}) \leq \epsilon$.

choose $\beta_0 > 0$
choose $\lambda_0 > 0$ and set $\Lambda_0 = \lambda_0$
set $x_0 = x^o$
compute $\xi_0 \in \partial g(x_0)$ and set $\zeta_0 = \lambda_0 \xi_0$ `Subgradient computation`
set $\bar{x} = \frac{\lambda_0 x_0}{\Lambda_0}$ and $\bar{\zeta} = \frac{\lambda_0 \xi_0}{\Lambda_0}$
set $k = 0$
while $g(\bar{x}) - \varphi(\bar{\zeta}) > \epsilon$ **do**
  set $k = k + 1$
  choose $\beta_k \geq \beta_{k-1}$ and compute $x_k = \arg \min_{x \in S} \{\langle \zeta_{k-1}, x \rangle + \beta_k d_S(x)\}$
  `Strongly convex projection`
  choose $\lambda_k > 0$ and set $\Lambda_k = \Lambda_{k-1} + \lambda_k$
  compute $\xi_k \in \partial g(x_k)$ and set $\zeta_k = \zeta_{k-1} + \lambda_k \xi_k$ `Subgradient computation`
  set $\bar{x} = \frac{1}{\Lambda_k} \sum_{i=0}^{k} \lambda_i x_i$ and $\bar{\zeta} = \frac{1}{\Lambda_k} \zeta_k = \frac{1}{\Lambda_k} \sum_{i=0}^{k} \lambda_i \xi_i$
  `Objective functions evaluation`
end while

Consider Algorithm 2, which corresponds to the general Primal-Dual Subgradient algorithm, and Figure 4.2, which illustrates the main ideas of the algorithm. Namely, at each step a new subgradient, ξ_2, is computed and accumulated $\zeta_2 := \zeta_1 + \lambda_2 \xi_2$. With this accumulation and a prox-function, a strongly convex approximation of $g(x)$ is created, $\nu_{\beta_3}(x)$. The minimizer of $\nu_{\beta_3}(x)$ over S will be

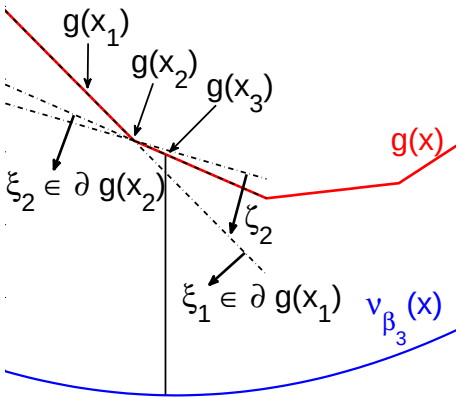


Fig. 4.2: Primal-Dual Subgradient Method

the point where the next subgradient will be evaluated. The primal solution is then given by a weighted sum of the computed evaluation points. The dual solution is similarly given by a weighted sum of the computed subgradients.

We note that the difficult operations in Algorithm 2 are the evaluation of the primal and dual functions $g(x)$ and $\varphi(\zeta)$, the computation of the subgradients of the primal function, and finally the computation of the minimizer of the strongly convex approximation over S, i.e., for fixed ζ and $\beta > 0$, $\arg \min_{x \in S} \{\langle \zeta, x \rangle + \beta d_S(x)\}$.

Theorem 4.5 (Convergence of Primal-Dual Subgradient Algorithm — Theorem 1 in [Nes09])
Suppose that Assumption 2 holds and consider Problem 3. Let $\bar{x}$ and $\bar{\zeta}$ be the solutions generated by Algorithm 2 after k iterations. Then,

$$g(\bar{x}) - g^* \leq g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \right) \tag{4.11}$$

holds, where $D_S := \max_{x \in S} d_S(x)$ and the dual function φ is defined in (4.9).

From the statement of Theorem 4.5, we notice that we can use the last part of Inequality (4.11) as stopping criterion instead of the dual gap, which requires the evaluation of both objective functions, $g(\bar{x})$ and $\varphi(\bar{\zeta})$. For a same desired accuracy, this choice of this stopping criterion leads to an increase of the number of completed iterations with respect to the dual gap. However, each iteration is less expensive, since we do not need to evaluate both objective functions anymore.

Before we prove Theorem 4.5, we investigate the sequences $\{\lambda_i\}_{i=0}^k$ and $\{\beta_i\}_{i=0}^k$ since the quality of the convergence of the primal-dual method depends on them as we note from Equation (4.11).

In particular, let us consider the following two kinds of sequences:

Simple averages sequences:

$$\lambda_i = 1,\ \beta_i = \frac{L}{\sqrt{2\sigma_S D_S}}\hat{\beta}_i \quad \forall\ i \geq 0 \tag{4.12}$$

Weighted averages sequences:

$$\lambda_i = \frac{1}{\|\xi_i\|_S^*},\ \beta_i = \frac{1}{\sqrt{2\sigma_S D_S}}\hat{\beta}_i \quad \forall\ i \geq 0 \tag{4.13}$$

where the sequence $\{\hat{\beta}_i\}_{i=0}^k$ is defined as follows

$$\hat{\beta}_0 = \hat{\beta}_1 = 1 \text{ and } \hat{\beta}_{i+1} = \hat{\beta}_i + \frac{1}{\hat{\beta}_i}\ \forall\ i \geq 1. \tag{4.14}$$

The sequence $\{\lambda_i\}_{i=0}^k$ defines how the evaluation points $\{x_i\}_{i=0}^k$ and the subgradients $\{\xi_i\}_{i=0}^k$ are accumulated. In the simple averages sequences every x_i and ξ_i for $i = 0, \cdots, m$ have the same weight. In the weighted averages sequences, the weight of x_i and ξ_i depends on the norm of ξ_i for $i = 0, \cdots, m$. For the sequence $\{\beta_i\}_{i=0}^k$, the constants multiplying β_i for $i = 0, \cdots, m$, have been chosen in order to minimize the right hand side of Equation (4.11).

The choice of $\{\hat{\beta}_i\}_{i=0}^k$ is motivated by their advantageous properties, see Lemma 4.6.

Lemma 4.6 (Lemma 3 in [Nes09])
The sequence $\{\hat{\beta}_i\}_{i=0}^k$ defined in (4.14) has the following properties,

a) $\hat{\beta}_{k+1} = \sum_{i=0}^k \frac{1}{\hat{\beta}_i}$ *for* $k \geq 0$,

b) $\sqrt{2k-1} \leq \hat{\beta}_k \leq \frac{1}{1+\sqrt{3}} + \sqrt{2k-1}$ *for* $k \geq 1$.

Given the properties of sequence $\{\hat{\beta}_i\}_{i\geq 0}$, Theorem 4.5 results in Theorem 4.7 when the simple averages sequences or the weighted averages sequences are used.

Theorem 4.7 (Theorem 2 and 3 in [Nes09])
Suppose that Assumption 2 holds and consider Problem 3. Let $\bar{x}$ and $\bar{\zeta}$ be the solutions generated by Algorithm 2 after k iterations using either simple averages sequences (4.12) or weighted averages sequences (4.13). Then,

$$g(\bar{x}) - g^* \leq g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{2L}{\sqrt{k+1}}\sqrt{\frac{2D_S}{\sigma_S}} \tag{4.15}$$

holds, where $D_S := \max_{x\in S}\ d_S(x)$ and the dual function φ is defined in (4.9).

Proof. First note that the first inequality in (4.15) is trivially satisfied since $g^* = \max_{\{\zeta \in \text{dom } g_*\}} \varphi(\zeta)$, see Equation (4.10).

Then, we evaluate the convergence result of Theorem 4.5,

$$g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \right),$$

when special $\{\lambda_i\}_{i=0}^k$ and $\{\beta_i\}_{i=0}^k$ sequences are used.

We first consider the simple averages sequences. Since $\lambda_i = 1$ and $\|\xi_i\|_S^* \leq L$ for $i = 1, \ldots, k$,

$$\begin{aligned}
\beta_{k+1} D_S &+ \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \\
&\leq \beta_{k+1} D_S + \frac{L^2}{2\sigma_S} \sum_{i=0}^{k} \frac{1}{\beta_i} \\
&= \frac{L D_S}{\sqrt{2\sigma_S D_S}} \hat{\beta}_{k+1} + \frac{L^2}{2\sigma_S} \sum_{i=0}^{k} \frac{\sqrt{2\sigma_S D_S}}{L \hat{\beta}_i} \qquad \left(\beta_i = \frac{L\hat{\beta}_i}{\sqrt{2\sigma_S D_S}}\right) \\
&= L\sqrt{\frac{D_S}{2\sigma_S}} \left(\hat{\beta}_{k+1} + \sum_{i=0}^{k} \frac{1}{\hat{\beta}_i} \right) \\
&= L\sqrt{\frac{2D_S}{\sigma_S}} \hat{\beta}_{k+1} \qquad \text{(Lemma 4.6 statement a))} \\
&\leq L\sqrt{\frac{2D_S}{\sigma_S}} \left(\frac{1}{1+\sqrt{3}} + \sqrt{2k-1} \right) \qquad \text{(Lemma 4.6 statement b))} \\
&\leq 2L\sqrt{\frac{2D_S}{\sigma_S}} \sqrt{k+1}.
\end{aligned}$$

Finally, we obtain $\frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \right) \leq \frac{2L}{\sqrt{k+1}} \sqrt{\frac{2D_S}{\sigma_S}}$.

Now we consider the weighted average sequences. Since for $i = 1, \ldots, k$, $\lambda_i = 1/\|\xi_i\|_S^*$, $\|\xi_i\|_S^* \leq L$, and $\beta_i = \frac{1}{\sqrt{2\sigma_S D_S}} \hat{\beta}_i$, we get the following results, using similar arguments as previous for the average sequences.

$$\begin{aligned}
\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} &= \beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{1}{\beta_i} \\
&= \sqrt{\frac{D_S}{2\sigma_S}} \left(\hat{\beta}_{k+1} + \sum_{i=0}^{k} \frac{1}{\hat{\beta}_i} \right) \\
&\leq 2\sqrt{\frac{2D_S}{\sigma_S}} \sqrt{k+1}.
\end{aligned}$$

Again we obtain $\frac{1}{\sum_{i=0}^{k}\lambda_i}\left(\beta_{k+1}D_S + \frac{1}{2\sigma_S}\sum_{i=0}^{k}\frac{\lambda_i^2}{\beta_i}\|\xi_i\|_S^{*\,2}\right) \leq \frac{2L}{\sqrt{k+1}}\sqrt{\frac{2D_S}{\sigma_S}}$. □

As expected, the Primal-Dual Subgradient method—same as the standard subgradient method—is optimal. Namely, for an approximate solution with an absolute error of $\epsilon > 0$, the primal-dual subgradient method requires at most $\mathrm{O}(\frac{1}{\epsilon^2}\sqrt{\frac{D_S}{\sigma_S}}L)$ iterations. The values of the constants L, D_S, and σ_S depend on the choice of the norm defined in the feasible set S and on the choice of the strongly convex function d_S. The numerical results presented in the second part of this thesis illustrate the importance of the constants L, D_S, and σ_S for speeding-up the Primal-Dual Subgradient algorithm. Let us turn to the proof of Theorem 4.5. This proof can be found in [Nes09]. However, we present it here since we extensively use it in Chapter 6, when we deal with approximate oracles.

Proof of Theorem 4.5. The solutions $\bar{x}$ and $\bar{\zeta}$ are primal and dual feasible since they are a convex combination of primal, respectively dual, feasible solutions. Moreover, the first inequality in (4.11) is trivially satisfied since $g^* = \max_{\{\zeta\in\mathrm{dom}\ g_*\}}\ \varphi(\zeta)$, see Equation (4.10).

Let us evaluate $g(\bar{x}) - \varphi(\bar{\zeta})$. By definition of $\varphi(\bar{\zeta})$, (4.9), we have

$$\begin{aligned}
g(\bar{x}) - \varphi(\bar{\zeta}) &= g(\bar{x}) + g_*(\bar{\zeta}) - \min_{x\in S}\langle\bar{\zeta}, x\rangle \\
&\leq \sum_{i=0}^{k}\frac{\lambda_i}{\Lambda_k}\left(g(x_i) + g_*(\xi_i)\right) - \min_{x\in S}\langle\sum_{i=0}^{k}\frac{\lambda_i\xi_i}{\Lambda_k}, x\rangle \\
&\quad (\text{ by convexity of } g \text{ and } g_*) \\
&= \sum_{i=0}^{k}\frac{\lambda_i}{\Lambda_k}\langle\xi_i, x_i\rangle - \min_{x\in S}\langle\sum_{i=0}^{k}\frac{\lambda_i\xi_i}{\Lambda_k}, x\rangle \\
&\quad (\xi_i \in \partial g(x_i)) \\
&= \frac{1}{\Lambda_k}\max_{x\in S}\Big\{\sum_{i=0}^{k}\lambda_i\langle\xi_i, x_i - x\rangle\Big\}.
\end{aligned}$$

We define $\theta_{k+1} := \max_{x\in S}\left\{\sum_{i=0}^{k}\lambda_i\langle\xi_i, x_i - x\rangle\right\}$ and proceed in two steps to find an upper bound for it. In the first step we show that

$$\theta_{k+1} \leq \beta_{k+1}D_S + \sum_{i=0}^{k}\lambda_i\langle\xi_i, x_i - x_0\rangle + \max_{x\in S}\left\{\langle\zeta_k, x_0 - x\rangle - \beta_{k+1}d_S(x)\right\} \quad (4.16)$$

and in the second step we show that,

$$(4.16) \leq \beta_{k+1}D_S + \frac{1}{2\sigma_S}\sum_{i=0}^{k}\frac{\lambda_i^2}{\beta_i}\|\xi_i\|_S^{*\,2}, \quad (4.17)$$

where $D_S = \max_{x \in S} d_S(x)$.

We start with the right-hand side of Inequality (4.16),

$$\begin{aligned}
\beta_{k+1} D_S \quad &+ \quad \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i - x_0 \rangle + \max_{x \in S} \{\langle \zeta_k, x_0 - x \rangle - \beta_{k+1} d_S(x)\} \\
&= \quad \max_{x \in S} \{\beta_{k+1}(D_S - d_S(x)) + \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i - x_0 + x_0 - x \rangle\} \\
&\qquad (\text{because } \zeta_k = \sum_{i=0}^{k} \lambda_i \xi_i) \\
&\geq \quad \max_{x \in S} \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i - x \rangle = \theta_{k+1}. \quad (D_S = \max_{x \in S} d_S(x) \text{ and } \beta_{k+1} > 0)
\end{aligned}$$

In order to prove inequality (4.17) we define the following functions

$$\nu_{\beta_i}(\zeta) := \max_{x \in S} \{\langle \zeta, x_0 - x \rangle - \beta_i d_S(x)\}, \quad i = 0, \ldots k. \tag{4.18}$$

In view of Theorem 3.7, the functions $\nu_{\beta_i}(\zeta)$ are convex and their gradient are Lipschitz continuous. The gradients and their Lipschitz constant can be expressed as follows,

$$\nabla \nu_{\beta_i}(\zeta) := x_0 - x_\zeta \qquad L_{\nu_{\beta_i}, S} := \frac{1}{\beta_i \sigma_S} \tag{4.19}$$

where $x_\zeta := \arg\max_{x \in S} \{\langle \zeta, x_0 - x \rangle - \beta d_S(x)\}$ and σ_S is the convexity parameter of d_S. In particular, $\nabla \nu_\beta(0) = 0$. Moreover, since $\beta_{i+1} \geq \beta_i$ for $i \geq 0$, we have $\nu_{\beta_{i+1}}(\zeta) \leq \nu_{\beta_i}(\zeta)$.

For $i = 0, \ldots, k$, note that the points x_i computed in Algorithm 2 correspond to $x_{\zeta_{i-1}} = \arg\max_{x \in S} \{\langle \zeta_{i-1}, x_0 - x \rangle - \beta_i d_S(x)\}$.

For $i \geq 1$ we have

$$\begin{aligned}
\nu_{\beta_{i+1}}(\zeta_i) \quad &\leq \quad \nu_{\beta_i}(\zeta_i) \qquad (\beta_{i+1} \geq \beta_i) \\
&\leq \quad \nu_{\beta_i}(\zeta_{i-1}) + \langle \nabla \nu_{\beta_i}(\zeta_{i-1}), \zeta_i - \zeta_{i-1} \rangle + \frac{L_{\nu_{\beta_i}}}{2} \|\zeta_i - \zeta_{i-1}\|_S^{*\,2} \\
&\qquad (\text{Theorem 3.3}) \\
&= \quad \nu_{\beta_i}(\zeta_{i-1}) + \langle x_0 - x_i, \zeta_i - \zeta_{i-1} \rangle + \frac{1}{2\beta_i \sigma_S} \|\zeta_i - \zeta_{i-1}\|_S^{*\,2} \\
&\qquad \big(\text{Equation (4.19) and } x_{\zeta_{i-1}} = x_i\big) \\
&= \quad \nu_{\beta_i}(\zeta_{i-1}) + \langle x_0 - x_i, \lambda_i \xi_i \rangle + \frac{\lambda_i^2}{2\sigma_S \beta_i} \|\xi_i\|_S^{*\,2}.
\end{aligned}$$

Last equality holds since $\zeta_i - \zeta_{i-1} = \lambda_i \xi_i$ for $i \geq 1$. Hence, the inequality

$$\nu_{\beta_{i+1}}(\zeta_i) - \nu_{\beta_i}(\zeta_{i-1}) \leq \lambda_i \langle \xi_i, x_0 - x_i \rangle + \frac{\lambda_i^2}{2\sigma_S \beta_i} \|\xi_i\|_S^{*\,2}$$

holds for $i = 1, \ldots, k$. Summing up over all $i \geq 1$, we get

$$\nu_{\beta_{k+1}}(\zeta_k) - \nu_{\beta_1}(\zeta_0) \leq \sum_{i=1}^{k} \lambda_i \langle \xi_i, x_0 - x_i \rangle + \frac{1}{2\sigma_S} \sum_{i=1}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2}.$$

Using that $\zeta_0 = \lambda_0 \xi_0$, we have

$$\begin{aligned} \nu_{\beta_1}(\zeta_0) &\leq \nu_{\beta_1}(0) + \langle \nabla \nu_{\beta_1}(0), \zeta_0 \rangle + \frac{\lambda_0^2}{2\sigma_S \beta_1} \|\xi_0\|_S^{*\,2} \\ &\leq -\beta_1 d_S(x_0) + \langle x_0 - x_0, \zeta_0 \rangle + \frac{\lambda_0^2}{2\sigma_S \beta_0} \|\xi_0\|_S^{*\,2} \qquad (\beta_1 \geq \beta_0) \\ &= \frac{\lambda_0^2}{2\sigma_S \beta_0} \|\xi_0\|_S^{*\,2} \qquad (d_S(x_0) = 0) \end{aligned}$$

and $\nu_{\beta_{k+1}}(\zeta_k) + \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i - x_0 \rangle \leq \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2}$.

Thus,

$$\begin{aligned} \theta_{k+1} &\leq \beta_{k+1} D_S + \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i - x_0 \rangle + \nu_{\beta_{k+1}}(\zeta_k) \\ &\leq \beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \end{aligned}$$

and we finally get

$$g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{1}{\Lambda_k} \theta_{k+1} \leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \right).$$

□

4.3 Primal-Dual Subgradient Algorithm for Functions with Bounded Variation of Subgradients

For the analysis of the Primal-Dual Subgradient method we assumed that the subgradients of $g(x)$ were bounded with respect to the chosen norm $\|.\|_S^*$ for any

$x \in S$, i.e, we assumed that a constant $L > 0$ exists so that $\|\xi_x\|_S^* \leq L$ for all $\xi_x \in \partial g(x),\ x \in S$, see Assumption 2. Now, we replace this assumption by supposing that the function g has subgradients with bounded variations, which gives us Assumption 3.

Assumption 3

- $S \subseteq \mathbb{R}^n$ is a non-empty, convex and compact set,
- $g : S \longrightarrow \mathbb{R}$ is a proper, closed, finite and convex function over S, its minimum, g^*, is attained at $x^* \in S$,
- the function g has subgradients with bounded variations over S, i.e.,

$$\exists\ M > 0 \text{ such that } \|\xi_x - \xi_y\|_S^* \leq M \quad \forall\ \xi_x \in \partial g(x),\ \xi_y \in \partial g(y),\ x, y \in S \tag{4.20}$$

where $\|.\|_S$ is a norm defined over $\mathbb{R}^n$ and $\|.\|_S^*$ is its dual.

Then, Problem 3 becomes Problem 4

Problem 4 Suppose Assumption 3 holds. Given $\epsilon > 0$, find a feasible primal solution $\bar{x} \in S$ such that $g(\bar{x}) - g^* \leq \epsilon$. We call $\bar{x}$ an ϵ- approximate solution.

In [Nes05b], Nesterov slightly modified the primal dual subgradient method to cope with the minimization of functions having bounded variations of subgradients. The modification consists of removing the starting primal solution x_0 and the corresponding subgradient $\xi_0 \in \partial g(x_0)$ from the primal and dual solution given by the algorithm, i.e.,

$$\bar{x} = \frac{1}{\Lambda_k} \sum_{i=1}^{k} \lambda_i x_i \quad \text{and} \quad \bar{\zeta} = \frac{1}{\Lambda_k} \sum_{i=1}^{k} \lambda_i \xi_i, \quad \text{where } \Lambda_k = \sum_{i=1}^{k} \lambda_i.$$

As in Algorithm 2, the difficult operations in Algorithm 3 are the evaluation of the primal and dual functions, $g(x)$ and $\varphi(\zeta)$, the computation of subgradients of the primal function, and finally the computation of the minimizer of the strongly convex approximation over S, i.e., for fixed ζ and $\beta > 0$, $\arg \min_{x \in S} \{\langle \zeta, x \rangle + \beta d_S(x)\}$. In order to avoid the evaluation of both objective functions, we may choose as stopping criterion the last part of the Equation (4.21) in Theorem 4.8, which states

Algorithm 3 Truncated Dual Averaging Algorithm (TDA)

Requires: - A prox-function $d_S(x)$ over S with convexity parameter $\sigma_S > 0$ with respect to norm $\|.\|_S$ and minimizer x^o over S
- An absolute error $\epsilon > 0$

Ensures: An approximate primal solution $\bar{x} \in S$ and an approximate dual solution $\bar{\zeta}$ such that $g(\bar{x}) - \varphi(\bar{\zeta}) \leq \epsilon$.

choose $\lambda_0 = \lambda_1 > 0$ and $\beta_0 = \beta_1 > 0$
compute $\xi_0 \in \partial g(x_0)$ and set $\zeta_0 = \lambda_0 \xi_0$ `Subgradient computation`
compute $x_1 = \arg \min_{x \in S} \{\langle \zeta_0, x \rangle + \beta_1 d_S(x)\}$ `Strongly convex projection`
compute $\xi_1 \in \partial g(x_1)$ and set $\zeta_1 = \lambda_1 \xi_1$ `Subgradient computation`
set $\bar{x} = x_1$, $\bar{\zeta} = \xi_1$, and $\Lambda_1 = \lambda_1$
set $k = 1$
while $g(\bar{x}) - \varphi(\bar{\zeta}) > \epsilon$ **do**
  set $k = k + 1$
  compute $x_k = \arg \min_{x \in S} \{\langle \zeta_{k-1}, x \rangle + \beta_{k-1} d_S(x)\}$
  `Strongly convex projection`
  choose $\lambda_k > 0$ and set $\Lambda_k = \Lambda_{k-1} + \lambda_k$
  choose β_k such that $\frac{\beta_k}{\Lambda_k} \leq \frac{\beta_{k-1}}{\Lambda_{k-1}}$
  compute $\xi_k \in \partial g(x_k)$ and set $\zeta_k = \zeta_{k-1} + \lambda_k \xi_k$ `Subgradient computation`
  set $\bar{x} = \frac{1}{\Lambda_k} \sum_{i=1}^{k} \lambda_i x_i$, and $\bar{\zeta} = \frac{1}{\Lambda_k} \zeta_k = \frac{1}{\Lambda_k} \sum_{i=1}^{k} \lambda_i \xi_i$
  `Objective functions evaluation`
end while

the convergence of Algorithm 3. Again, as for the Dual Averaging Algorithm, this may lead to an increase of the number of iterations with respect to the number of iterations done using the dual gap as stopping criterion.

Theorem 4.8 (Truncated Dual Averaging Algorithm)
Suppose that Assumption 3 holds and consider Problem 4. Let $\bar{x}$ and $\bar{\zeta}$ be the solutions generated by Algorithm 3 after k iterations. Then,

$$g(\bar{x}) - g^* \leq g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{1}{\sum_{i=1}^{k} \lambda_i} \left(D_S\big(\beta_0 + \sum_{i=2}^{k} \frac{\lambda_i \beta_{i-1}}{\Lambda_{i-1}}\big) + \frac{M^2}{2\sigma_S} \sum_{i=1}^{k} \frac{\lambda_i^2}{\beta_{i-1}} \right), \tag{4.21}$$

holds, where $D_S := \max_{x \in S} d_S(x)$.

In particular, using the sequences $\{\lambda_i\}_{i=0}^k$ and $\{\beta_i\}_{i=0}^k$ defined as follows

$$\lambda_i = 1, \quad \beta_i = \frac{M}{\sqrt{2\sigma_S D_S}} \sqrt{i} \quad \forall\ i \geq 1, \text{ and } \lambda_0 = \lambda_1,\ \beta_0 = \beta_1,$$

we have after k iterations that $g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{2M}{\sqrt{k}} \sqrt{\frac{2D_S}{\sigma_S}}$.

The proof of convergence of Algorithm 3 is similar to the proof of convergence of Algorithm 2. As for Algorithm 2, the convergence rate of Algorithm 3 depends on the choice of the sequences $\{\lambda_i\}_{i=0}^k$ and $\{\beta_i\}_{i=0}^k$. The main difference between the convergence of Algorithm 2 and Algorithm 3 is then the value of the constants L and M. We know that $M \leq 2L$, however M could be small and L large.

4.4 Applying the Primal-Dual Subgradient Method to LPs

We would like to apply the Primal-Dual Subgradient method to solve the following problem:

$$\min_{x \in Q} f(x), \qquad f(x) := c^T x + \max_{u \in P} \{u^T A x - b^T u\}, \tag{4.22}$$

where $c \in \mathbb{R}^n, b \in \mathbb{R}^m, A \in \mathbb{R}^{n \times m}$, $Q \subset \mathbb{R}^n$ and $P \subset \mathbb{R}^m$ are non-empty, convex, and compact polytopes, see Assumption 1. We assume that $\mathbb{R}^n$ and $\mathbb{R}^m$ are induced with norms, which we denote by $\|.\|_Q$ and $\|.\|_P$ respectively.

First, we consider the subdifferential of f at $x \in \mathbb{R}^n$, $\partial f(x)$. Note that the domain of definition of f is $\mathbb{R}^n$ since P is compact.

Lemma 4.9
For $x \in \mathbb{R}^n$, define $U(x) := \left\{ u \in P \mid u := \arg \max_{v \in P} \{(Ax - b)^T v\} \right\}$. Then,

$$\partial f(x) = c + \operatorname{conv}\{A^T u_x \mid\ u_x \in U(x)\}. \tag{4.23}$$

Proof. "$\supseteq$": Let $x, y \in \mathbb{R}^n$. Take $u_x \in U(x)$ and define $\xi := c + A^T u_x$. Then,

$$\begin{aligned} f(x) + \xi^T(y-x) &= c^T x + \max_{u \in P} \{(Ax-b)^T u\} + (c + A^T u_x)^T (y-x) \\ &= c^T x + (Ax-b)^T u_x + c^T(y-x) + u_x^T A(y-x) \\ &= c^T y + (Ay-b)^T u_x \\ &\leq c^T y + \max_{u \in P} \{(Ay-b)^T u\} = f(y) \end{aligned}$$

holds. Thus, $\xi \in \partial f(x)$.

"$\subseteq$": The function $f(x)$ is closed and convex over its domain, $\mathbb{R}^n$. Therefore, we have (Theorem 23.5 in [Roc70])

$$f(x) + f_*(\xi) = x^T \xi \quad \forall\, x \in \mathbb{R}^n, \forall\, \xi \in \partial f(x). \tag{4.24}$$

Let us consider $f_*(\xi)$ for $\xi \in \partial f(x)$.

$$\begin{aligned} f_*(\xi) &= \sup_{x \in \mathbb{R}^n} \{\xi^T x - f(x)\} \\ &= \sup_{x \in \mathbb{R}^n} \left\{ \xi^T x - c^T x + \min_{u \in P} \{-(Ax)^T u + b^T u\} \right\} \\ &= \min_{u \in P} \left\{ \sup_{x \in \mathbb{R}^n} \{ \left(\xi - (c + A^T u)\right)^T x \} + b^T u \right\}. \end{aligned}$$

We note that $f_*(\xi)$ is finite if and only if an $\tilde{u} \in P$ exists such that $\xi = c + A^T \tilde{u}$, i.e., $\xi - c \in \mathrm{Im} A^T$.

Let us observe that the function $x \mapsto \max_{u \in P} \{(Ax-b)^T u\}$ corresponds to the support function of P evaluated at $Ax - b$ and that its conjugate is the indicator function of P.

Now suppose that a $\tilde{u}$ exists with $\xi = c + A^T \tilde{u}$. Then

$$\begin{aligned} f_*(c + A^T \tilde{u}) &= \sup_{x \in \mathbb{R}^n} \left\{ (c + A^T \tilde{u})^T x - c^T x - \max_{u \in P} \{(Ax-b)^T u\} \right\} \\ &= \sup_{x \in \mathbb{R}^n} \left\{ (Ax-b)^T \tilde{u} + b^T \tilde{u} - \max_{u \in P} \{(Ax-b)^T u\} \right\} \\ &= b^T \tilde{u} + \sup_{x \in \mathbb{R}^n} \left\{ (Ax-b)^T \tilde{u} - \max_{u \in P} \{(Ax-b)^T u\} \right\} \\ &= b^T \tilde{u} \end{aligned}$$

Last equality holds since the last supremum corresponds to the indicator function of P evaluated at $\tilde{u} \in P$.

Then, using Equality (4.24), last result and that $f(x)$ is finite we have

$$\begin{aligned}(c + A^T\tilde{u})^T x - f_*(c + A^T\tilde{u}) &= f(x) \\ c^T x + (Ax)^T\tilde{u} - b^T\tilde{u} &= c^T x + \max_{u\in P}\{(Ax-b)^T u\} \\ (Ax-b)^T\tilde{u} &= \max_{u\in P}\{(Ax-b)^T u\}.\end{aligned}$$

Thus, $\tilde{u} \in U(x)$ and $\xi = (c + A^T\tilde{u}) \in c + \text{conv}\{A^T u_x \mid \ u_x \in U(x)\}$. □

Note that computing a subgradient of f at $x \in \mathbb{R}^n$ is equivalent to solving the maximization problem $\max_{u\in P}\{(Ax-b)^T u\}$ and thus, to evaluating $f(x)$. Since the function $u \mapsto (Ax-b)^T u$ is linear in u, its maximum is attained on the boundary of P and is prone to be non-unique.

We previously presented two Primal-Dual Subgradient methods, one for minimizing functions with bounded subgradients (Algorithm 2) and the other for minimizing functions with bounded variations of subgradients (Algorithm 3). The convergence rates of both methods mainly differ on the value of these bounds, i.e., the bound of the subgradients' norm L and the bound of the norm of subgradients variations M. Which method should we employ for our minimization problem (4.22)?

Let $x, y \in \mathbb{R}^n$ and $\xi_x \in \partial f(x)$, $\xi_y \in \partial f(y)$ be defined as follows

$$\xi_x := c + A^T v_x \qquad \xi_y := c + A^T v_y,$$

where $v_x \in U(x)$ and $v_y \in U(y)$, with $U(x)$ and $U(y)$ defined as in Lemma 4.9. Then, for all $x \in Q$ we have

$$\begin{aligned}\|\xi_x\|_Q^* &= \|c + A^T v_x\|^* \\ &\leq \|c\|_Q^* + \|A\|_{Q,P}\|v_x\|_P \\ &\leq \|c\|_Q^* + \|A\|_{Q,P} \max_{v\in P}\|v\|_P \\ &= \|c\|_Q^* + \|A\|_{Q,P} R_P\end{aligned}$$

where $R_P := \max_{v\in P}\|v\|_P$. For all $x, y \in Q$, we also have

$$\begin{aligned}\|\xi_x - \xi_y\|_Q^* &= \|c + A^T v_x - (c + A^T v_y)\|_Q^* \\ &= \|A^T(v_x - v_y)\|_Q^* \\ &\leq \|A\|_{Q,P}\|v_x - v_y\|_P \\ &\leq 2\|A\|_{Q,P} \max_{v\in P}\|v\|_P \\ &= 2\|A\|_{Q,P} R_P.\end{aligned}$$

Thus, we take $L := \|c\|_Q^* + \|A\|_{Q,P} R_P$ and $M := 2\|A\|_{Q,P} R_P$. Depending on the value of $\|c\|_Q^*$, the constant L may be larger than M. Thus, the choice of the method depends on the considered instance of our LPs.

Algorithms 2 and 3 are primal-dual algorithms. Namely, at each step of the algorithms, primal and dual feasible solutions are computed and a dual gap can therefore be evaluated. The general dual problem considered by the methods is however not the standard one, which is established using the conjugate function of the function to be minimized,

$$\max_{\zeta \in \text{dom } f_*} \varphi(\zeta),$$

where $\varphi(\zeta) := -f_*(\zeta) + \min_{x \in Q} \zeta^T x$, (see Section 4.2), but they are equivalent. Let us consider the conjugate function of f,

$$\begin{aligned}
f_*(\zeta) &= \sup_{x \in \mathbb{R}^n} \{\zeta^T x - f(x)\} \\
&= \sup_{x \in \mathbb{R}^n} \left\{ \zeta^T x - c^T x - \max_{u \in P} \{(Ax)^T u - b^T u\} \right\} \\
&= \sup_{x \in \mathbb{R}^n} \left\{ \zeta^T x - c^T x + \min_{u \in P} \{b^T u - (Ax)^T u\} \right\} \\
&= \min_{u \in P} \left\{ \sup_{x \in \mathbb{R}^n} \{ \left(\zeta - (c + A^T u)\right)^T x\} + b^T u \right\} \\
&= \begin{cases} +\infty & \text{if } \nexists\ \tilde{u} \in P \text{ s.t. } \zeta = c + A^T \tilde{u} \\ b^T \tilde{u} & \text{if } \exists\ \tilde{u} \in P \text{ s.t. } \zeta = c + A^T \tilde{u} \text{ and } \tilde{u} = \arg \min_{u \in P} \{b^T u\} \end{cases}
\end{aligned}$$

and the general dual objective function $\varphi(\zeta)$ is defined as follows,

$$\begin{aligned}
\varphi(\zeta) &:= -f_*(\zeta) + \min_{x \in Q} \{\zeta^T x\} \\
&= \begin{cases} -\infty & \text{if } \nexists\ \tilde{u} \in P \text{ s.t. } \zeta = c + A^T \tilde{u} \\ -b^T \tilde{u} + \min_{x \in Q} \{(c + A^T \tilde{u})^T x\} & \text{if } \exists\ \tilde{u} \in P \text{ s.t. } \zeta = c + A^T \tilde{u} \text{ and } \\ & \tilde{u} = \arg \min_{u \in P} \{b^T u\} \end{cases}
\end{aligned}$$

For $\zeta \in \text{dom } f_*$, a $\tilde{u} \in P$ exists so that $\zeta = c + A^T \tilde{u}$. Thus,

$$\begin{aligned}
\max_{\zeta \in \text{dom } f_*} \varphi(\zeta) &= \max_{u \in P} \varphi(c + A^T u) \\
&= \max_{u \in P} \left\{ -b^T u + \min_{x \in Q} \{(c + A^T u)^T x\} \right\} = \max_{u \in P} \psi(u)
\end{aligned}$$

We show in the following that we can use the information provided by the solution $\bar{\zeta}$ delivered by Algorithm 2 to create a feasible solution for our dual problem having the same dual gap upper bound as $f(\bar{x}) - \varphi(\bar{\zeta})$.

For each computed x_i and ξ_i we choose $v_i \in U(x_i)$ and define $\bar{v} := \frac{1}{\Lambda_k} \sum_{i=0}^{k} \lambda_i v_i$. Since each v_i belongs to P and P is convex, we have $\bar{v} \in P$. Thus, $\bar{v}$ is a dual

feasible solution. Let us now evaluate $f(\bar{x}) - \psi(\bar{v})$.

$$\begin{aligned}
f(\bar{x}) - \psi(\bar{v}) &= f\Big(\frac{1}{\Lambda_k}\sum_{i=0}^{k}\lambda_i x_i\Big) - \psi\Big(\frac{1}{\Lambda_k}\sum_{i=0}^{k}\lambda_i v_i\Big) \\
&\leq \frac{1}{\Lambda_k}\sum_{i=0}^{k}\lambda_i\left(f(x_i) - \psi(v_i)\right) \\
&= \frac{1}{\Lambda_k}\sum_{i=0}^{k}\lambda_i\Big(c^T x_i + \max_{v\in P}\{(Ax_i - b)^T v\} + \\
&\qquad\qquad b^T v_i - \min_{x\in Q}\{(A^T v_i + c)^T x\}\Big) \\
&= \frac{1}{\Lambda_k}\sum_{i=0}^{k}\lambda_i\Big(c^T x_i + v_i^T A x_i - b^T v_i + \\
&\qquad\qquad b^T v_i + \max_{x\in Q}\{-(A^T v_i + c)^T x\}\Big) \\
&= \frac{1}{\Lambda_k}\max_{x\in Q}\Big\{\sum_{i=0}^{k}\lambda_i\langle c + A^T v_i, x_i - x\rangle\Big\} \\
&= \frac{1}{\Lambda_k}\max_{x\in Q}\Big\{\sum_{i=0}^{k}\lambda_i\langle \xi_i, x_i - x\rangle\Big\}
\end{aligned}$$

Recall that we defined in the proof of Theorem 4.5 on the convergence of Algorithm 2 the quantity

$$\theta_{k+1} := \max_{x\in Q}\Big\{\sum_{i=0}^{k}\lambda_i\langle \xi_i, x_i - x\rangle\Big\}$$

and we showed that

$$\theta_{k+1} \leq \beta_{k+1} D_S \; + \; \frac{1}{2\sigma_S}\sum_{i=0}^{k}\frac{\lambda_i^2}{\beta_i}\|\xi_i\|_S^{*\,2}.$$

Thus, we have the same dual gap upper bound as the general dual problem,

$$f(\bar{x}) - \psi(\bar{v}) \leq \frac{1}{\sum_{i=0}^{k}\lambda_i}\left(\beta_{k+1} D_S + \frac{L^2}{2\sigma_S}\sum_{i=0}^{k}\frac{\lambda_i^2}{\beta_i}\right).$$

A similar reasoning can be done for a dual approximation solution $\bar{v}$ obtained by Algorithm 3.

5. Smoothing Techniques and Gradient Mapping

The optimization techniques in [Nes05c] and in [Nes05a] were developed for minimizing non-smooth functions with a special structure and are therefore, less general than the Primal-Dual Subgradient techniques described in the previous chapter. However, they have a better theoretical running time of $O(1/\epsilon)$ instead of $O(1/\epsilon^2)$ for an absolute accuracy of ϵ for this special structure, which is

$$g(x) := \hat{g}(x) + \max_{u \in T} \{\langle Bx, u\rangle - \hat{\varphi}(u)\} \quad \forall\, x \in S \tag{5.1}$$

where $S \subset \mathbb{R}^n$ and $T \subset \mathbb{R}^m$ are non empty, convex, and compact sets. $\hat{g} : S \to \mathbb{R}$ and $\hat{\varphi} : T \to \mathbb{R}$ are differentiable convex functions with Lipschitz continuous gradients with Lipschitz constants $L_{\hat{g},S}$ and $L_{\hat{\varphi},T}$. $B \in \mathbb{R}^{m \times n}$ is a linear operator.

Our objective is to minimize g over S. Given the properties of function g and of the sets S and T we have

$$\min_{x \in S} g(x) = \max_{u \in T} \varphi(u),$$

where

$$\varphi(u) := -\hat{\varphi}(u) + \min_{x \in S} \{\langle Bx, u\rangle + \hat{g}(x)\} \quad \forall\, u \in T, \tag{5.2}$$

see Theorem A.3 or Corollary 37.3.2 in [Roc70].

We use S and T as indices to distinguish between the norms defined on $\mathbb{R}^n$ and $\mathbb{R}^m$ as well as the different parameters defined with respect to the sets S and T. We denote the norm of the linear operator B by

$$\|B\|_{S,T} := \max_{\|x\|_S \leq 1} \max_{\|u\|_T \leq 1} \langle Bx, u\rangle.$$

Moreover, g shall be called the *primal objective function* and finding $g^* = \min_{x \in S} g(x)$ is the corresponding *primal optimization problem*. Analogously φ is the *dual objective function* and finding $\varphi^* = \max_{u \in T} \varphi(u)$ the *dual optimization problem*.

We next describe the characteristics of the problem under consideration in this chapter.

Assumption 4

- $S \subset \mathbb{R}^n$ is a non empty, convex and compact set.
- $T \subset \mathbb{R}^n$ is a non empty, convex and compact set.
- $\hat{g} : S \longrightarrow \mathbb{R}$ is a closed, smooth, and convex function over S
- $\hat{\varphi} : T \longrightarrow \mathbb{R}$ is a closed, smooth, and convex function over T
- $B \in \mathbb{R}^{m \times n}$ is a linear operator.
- $g : S \longrightarrow \mathbb{R}$ is a closed and convex function over S defined as follows

$$g(x) := \hat{g}(x) + \max_{u \in T} \{\langle Bx, u\rangle - \hat{\varphi}(u)\} \quad \forall\, x \in S$$

  Its minimum, g^*, is finite and attained at $x^* \in S$.

- $\varphi : T \longrightarrow \mathbb{R}$ is a closed and concave function over T defined as follows

$$\varphi(u) := -\hat{\varphi}(u) + \min_{x \in S} \{\langle Bx, u\rangle + \hat{g}(x)\} \quad \forall\, u \in T$$

  Its maximum, φ^*, is finite and attained at $u^* \in T$.

Problem 5 Suppose Assumption 4 holds. Given $\epsilon > 0$, find a feasible primal solution $\bar{x} \in S$ and a feasible dual solution $\bar{u} \in T$ such that $g(\bar{x}) - \varphi(\bar{u}) \leq \epsilon$.

The first step in the methods from [Nes05c] and [Nes05a] is to find smooth approximation functions for both the primal and the dual objective function. The functions g and φ are not differentiable in general, since the optimization problems $\max_{\{u \in T\}}\{\langle Bx, u\rangle - \hat{\varphi}(u)\}$ and $\min_{\{x \in S\}}\{\langle Bx, u\rangle + \hat{g}(x)\}$ may have a non unique solution.

In order to create smooth convex, respectively concave, approximations, the methods use prox-functions, see Definition 3.6. Namely, $d_T : T \longrightarrow \mathbb{R}_+$ is a strongly convex function with convexity parameter $\sigma_T > 0$ with respect to norm $\|.\|_T$ and $d_S : S \longrightarrow \mathbb{R}_+$ is a strongly convex function with convexity parameter $\sigma_S > 0$ with respect to norm $\|.\|_S$. Without loss of generality we assume that the minimum of d_S over S and of d_T over T is zero. Then, the smooth approximations of the objective functions are defined as follows,

$$g_{\mu_T}(x) := \hat{g}(x) + \max_{u \in T} \{\langle Bx, u\rangle - \hat{\varphi}(u) - \mu_T d_T(u)\} \quad \forall\, x \in S \tag{5.3}$$

$$\varphi_{\mu_S}(u) := -\hat{\varphi}(u) + \min_{x \in S} \{\langle Bx, u\rangle + \hat{g}(x) + \mu_S d_S(x)\} \quad \forall\, u \in T \tag{5.4}$$

where $\mu_T > 0$ and $\mu_S > 0$ are called the *smoothing factors*.

Both functions g_{μ_T} and φ_{μ_S} are of the same type as the function defined in (3.10) and therefore they have the properties described in Theorem 3.7. In particular, g_{μ_T} is convex and differentiable with Lipschitz continuous gradient ∇g_{μ_T} with Lipschitz constant $L_{g_{\mu_T},S}$,

$$\nabla g_{\mu_T}(x) = \nabla \hat{g}(x) + B^T u_{\mu_T,x} \qquad L_{g_{\mu_T},S} = L_{\hat{g},S} + \frac{\|B\|^2_{S,T}}{\mu_T \sigma_T} \tag{5.5}$$

where

$$u_{\mu_T,x} := \arg \max_{u \in T} \{\langle Bx, u\rangle_T - \hat{\varphi}(u) - \mu_T d_T(u)\}, \tag{5.6}$$

and φ_{μ_S} is concave and differentiable with Lipschitz continuous gradient $\nabla \varphi_{\mu_S}$ with Lipschitz constant $L_{\varphi_{\mu_S},T}$,

$$\nabla \varphi_{\mu_S}(u) = -\nabla \hat{\varphi}(u) + B x_{\mu_S,u} \qquad L_{\varphi_{\mu_S},T} = L_{\hat{\varphi},T} + \frac{\|B\|^2_{S,T}}{\mu_S \sigma_S} \tag{5.7}$$

where

$$x_{\mu_S,u} := \arg \min_{x \in S} \{\langle Bx, u\rangle + \hat{g}(x) + \mu_S d_S(x)\}. \tag{5.8}$$

In [Nes05c], Nesterov defines an optimal scheme for smooth optimization where the smooth objective function also has a Lipschitz continuous gradient. Denoting by L the Lipschitz constant of its gradient, the scheme has a convergence rate of $O(\sqrt{L/\epsilon})$, which is the best we can hope for, see [Nes03], Chapter 2. This result was first published in [NY83]. Putting together the optimal scheme and the approximation of the primal function (5.3), with a smoothing factor μ_T of order $O(\epsilon)$ and thus $L \approx O(\frac{1}{\epsilon})$, we achieve a convergence rate of $O(1/\epsilon)$.

The method presented in [Nes05c] is a primal-dual method, yet it requires a number of iterations fixed in advance and the dual is only computed once, during the last iteration. Here, we concentrate on a variation of the method described in [Nes05a]. It has the same convergence rate as the method described in [Nes05c], yet it computes a primal and a dual solution at each step and the number of iterations needed to proceed does not have to be known in advance.

Before entering into the details of the method described in [Nes05a], let us recall the relation between both objective functions and their approximations. For all $x \in S$ and $u \in T$ we have $\varphi(u) \leq g(x)$ and there is no duality gap, i.e., $\varphi^* = g^*$. Let us denote by D_T the maximum of the prox-function d_T over T and by D_S the maximum of the prox-function d_S over S, i.e.,

$$D_T := \max_{u \in T} d_T(u) \qquad \text{and} \qquad D_S := \max_{x \in S} d_S(x).$$

Then, using Equations (5.1), (5.2), (5.3), and (5.4), we have for all $x \in S$ and $u \in T$

$$g_{\mu_T}(x) \leq g(x) \leq g_{\mu_T}(x) + \mu_T D_T \tag{5.9}$$

and

$$\varphi_{\mu_S}(u) - \mu_S D_S \leq \varphi(u) \leq \varphi_{\mu_S}(u). \tag{5.10}$$

Hence, we note that for all $x \in S$ and $u \in T$ the corresponding gap can be bounded as follows,

$$0 \leq g(x) - \varphi(u) \leq g_{\mu_T}(x) - \varphi_{\mu_S}(u) + \mu_T D_T + \mu_S D_S. \tag{5.11}$$

Thus, if we can ensure for an $x \in S$, and a $u \in T$ that $g_{\mu_T}(x) \leq \varphi_{\mu_S}(u)$, we can bound the duality gap using the smoothing factors and the maximum of the prox-functions,

$$0 \leq g(x) - \varphi(u) \leq \mu_T D_T + \mu_S D_S.$$

The idea of the method consists in finding a way to generate sequences $\{x_k, \mu_T^k\}_{k\geq 0}$ and $\{u_k, \mu_S^k\}_{k\geq 0}$ satisfying $g_{\mu_T^k}(x_k) \leq \varphi_{\mu_S^k}(u_k)$ for each $k \geq 0$ and $\mu_T^k, \mu_S^k \longrightarrow 0$ with $k \longrightarrow \infty$, so that the duality gap $g(x_k) - \varphi(u_k)$ remains bounded by $\mu_T^k D_T + \mu_S^k D_S$ at every step k of the algorithm. Nesterov called the condition

$$g_{\mu_T^k}(x_k) \leq \varphi_{\mu_S^k}(u_k), \tag{5.12}$$

the *Excessive Gap condition* and the method is then called *Excessive Gap method*. Figure 5.1 illustrates this idea for the special case where $S = T$ and $\varphi(u) := -f(x)$. This picture is based on the illustration of the Excessive Gap method in [Chu05].

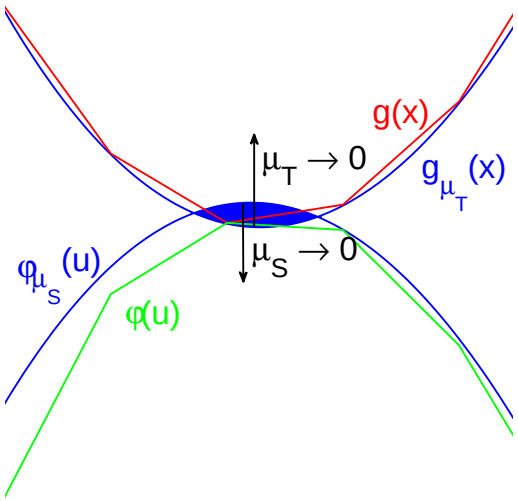


Fig. 5.1: Excessive Gap Method

5.1 Gradient Mapping

The main object used in the methods presented by Nesterov ([Nes05c],[Nes05a]) is called *gradient mapping*. We begin by focusing on the primal approximation g_{μ_T}. The latter is a smooth function with Lipschitz continuous gradient with Lipschitz constant $L_{g_{\mu_T},S}$, see Equations (5.3) and (5.5). Thus, for $x, y \in S$, the inequality

$$g_{\mu_T}(y) \leq g_{\mu_T}(x) + \langle \nabla g_{\mu_T}(x), y - x \rangle + \frac{L_{g_{\mu_T},S}}{2} \|y - x\|_S^2$$

holds (Theorem 3.3). Hence, for a fixed $x \in S$ the function $Z_{\mu_T,x} : S \longrightarrow \mathbb{R}$ defined as

$$Z_{\mu_T,x}(y) := g_{\mu_T}(x) + \langle \nabla g_{\mu_T}(x), y - x \rangle + \frac{L_{g_{\mu_T},S}}{2} \|y - x\|_S^2 \tag{5.13}$$

is smooth and strongly convex and may be interpreted as an upper approximation of $g_{\mu_T}(y)$. The *gradient mapping*, denoted by $GM_{g_{\mu_T}}(x)$, is then the unique minimizer of this upper approximation,

$$\begin{aligned} GM_{g_{\mu_T}}(x) &:= \arg \min_{y \in S} Z_{\mu_T,x}(y) \\ &= \arg \min_{y \in S} \left\{ \langle \nabla g_{\mu_T}(x), y - x \rangle + \frac{L_{g_{\mu_T},S}}{2} \|y - x\|_S^2 \right\}. \end{aligned} \tag{5.14}$$

See Figure 5.2 for an illustration of the gradient mapping.

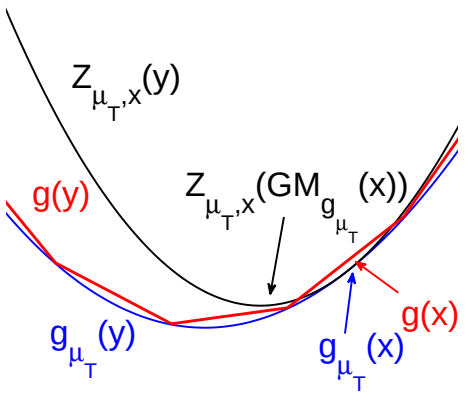


Fig. 5.2: Gradient Mapping

Similarly, for a fixed $u \in T$, we define $W_{\mu_S,u} : T \longrightarrow \mathbb{R}$, a smooth strongly concave lower approximation of $\varphi_{\mu_S}(v)$, as

$$W_{\mu_S,u}(v) := \varphi_{\mu_S}(u) + \langle \nabla \varphi_{\mu_S}(u), v - u \rangle - \frac{L_{\varphi_{\mu_S},T}}{2} \|v - u\|_T^2 \tag{5.15}$$

and its unique maximizer $GM_{\varphi_{\mu_S}}(u)$,

$$\begin{aligned} GM_{\varphi_{\mu_S}}(u) &:= \arg \max_{v \in T} W_{\mu_S,u}(v) \\ &= \arg \max_{v \in T} \left\{ \langle \nabla \varphi_{\mu_S}(u), v-u \rangle - \frac{L_{\varphi_{\mu_S},T}}{2} \|v-u\|_T^2 \right\}. \end{aligned} \tag{5.16}$$

In the following we show how using the gradient mapping $GM_{g_{\mu_T}}(x)$ and its symmetric $GM_{\varphi_{\mu_S}}(u)$, we can generate from the pairs (x, μ_T) and (u, μ_S) satisfying the Excessive Gap condition, $g_{\mu_T}(x) \leq \varphi_{\mu_S}(u)$, the pairs $(\bar{x}, \bar{\mu}_T)$ and $(\bar{u}, \bar{\mu}_S)$ also satisfying the Excessive Gap condition.

Theorem 5.1 ([Nes05a], Theorem 4.2)
Let $x \in S$ and $u \in T$ satisfying the Excessive Gap Condition for some $\mu_T > 0$ and $\mu_S > 0$. For $\tau \in (0,1)$ compute

$$\begin{aligned} \hat{x} &:= (1-\tau)x + \tau x_{\mu_S,u} \\ \bar{u} &:= (1-\tau)u + \tau u_{\mu_T,\hat{x}} \\ \bar{x} &:= GM_{g_{\mu_T}}(\hat{x}) \end{aligned}$$

and set $\bar{\mu}_S := (1-\tau)\mu_S$ and $\bar{\mu}_T := \mu_T$. Then, the pairs $(\bar{x}, \bar{\mu}_T)$ and $(\bar{u}, \bar{\mu}_S)$ satisfy the Excessive Gap condition provided that τ is chosen in accordance to

$$\frac{\tau^2}{1-\tau} \leq \frac{\mu_S \sigma_S}{L_{g_{\mu_T},S}}. \tag{5.17}$$

Theorem 5.2 gives the symmetric result of Theorem 5.1 for the dual step.

Theorem 5.2
Let $x \in S$ and $u \in T$ satisfying the Excessive Gap condition for some $\mu_T > 0$ and $\mu_S > 0$. For $\tau \in (0,1)$ compute

$$\begin{aligned} \hat{u} &:= (1-\tau)u + \tau u_{\mu_T,x} \\ \bar{x} &:= (1-\tau)x + \tau x_{\mu_S,\hat{u}} \\ \bar{u} &:= GM_{\varphi_{\mu_S}}(\hat{u}) \end{aligned}$$

and set $\bar{\mu}_S := \mu_S$ and $\bar{\mu}_T := (1-\tau)\mu_T$. Then, the pairs $(\bar{x}, \bar{\mu}_T)$ and $(\bar{u}, \bar{\mu}_S)$ satisfy the Excessive Gap condition provided that τ is chosen in accordance to

$$\frac{\tau^2}{1-\tau} \leq \frac{\mu_T \sigma_T}{L_{\varphi_{\mu_S},T}}. \tag{5.18}$$

Before proving Theorem 5.1 we consider the following important inequality for the proof of Theorem 5.1.

Lemma 5.3 ([Nes05a], Lemma 3.2)
For any $x, y \in S$ and $\mu_T > 0$ we have

$$g_{\mu_T}(y) + \langle \nabla g_{\mu_T}(y), x - y \rangle \leq \hat{g}(x) + \langle Bx, u_{\mu_T,y} \rangle - \hat{\varphi}(u_{\mu_T,y}). \tag{5.19}$$

Proof. By definition of g_{μ_T} and convexity of $\hat{g}$ we have for any $x, y \in S$ and $\mu_T > 0$,

$$\begin{aligned} g_{\mu_T}(y) &+ \langle \nabla g_{\mu_T}(y), x - y \rangle \\ &= \hat{g}(y) + \langle By, u_{\mu_T,y} \rangle - \hat{\varphi}(u_{\mu_T,y}) - \mu_T d_T(u_{\mu_T,y}) + \\ &\quad \langle \nabla \hat{g}(y) + B^T u_{\mu_T,y}, x - y \rangle \\ &\leq \hat{g}(x) + \langle Bx, u_{\mu_T,y} \rangle - \hat{\varphi}(u_{\mu_T,y}) - \mu_T d_T(u_{\mu_T,y}) \\ &\leq \hat{g}(x) + \langle Bx, u_{\mu_T,y} \rangle - \hat{\varphi}(u_{\mu_T,y}). \end{aligned}$$

The last inequality holds since we assume that d_T is non-negative over T. □

Next, we present the proof of Theorem 5.1. This proof can be found in [Nes05a]. However, we also present it here since we use it in Chapter 6, when we consider approximate oracles.

Proof of Theorem 5.1. We have to show that $\varphi_{\bar{\mu}_S}(\bar{u}) \geq g_{\bar{\mu}_T}(\bar{x})$. Therefore we evaluate $\varphi_{\bar{\mu}_S}(\bar{u})$.

$$\begin{aligned} \varphi_{\bar{\mu}_S}(\bar{u}) &= -\hat{\varphi}(\bar{u}) + \min_{y \in S} \{\langle By, \bar{u} \rangle + \hat{g}(y) + \bar{\mu}_S d_S(y)\} \\ &= -\hat{\varphi}(\bar{u}) + \min_{y \in S} \{\langle By, \bar{u} \rangle + \hat{g}(y) + (1-\tau)\mu_S d_S(y)\} \\ &\quad (\ \bar{\mu}_S = (1-\tau)\mu_S\ ) \\ &\geq -(1-\tau)\hat{\varphi}(u) - \tau\hat{\varphi}(u_{\mu_T,\hat{x}}) + \\ &\quad \min_{y \in S} \left\{\langle By, (1-\tau)u + \tau u_{\mu_T,\hat{x}} \rangle + \hat{g}(y) + (1-\tau)\mu_S d_S(y)\right\} \\ &\quad (\text{by convexity of } \hat{\varphi}) \\ &= \min_{y \in S} \Big\{(1-\tau)\underbrace{[-\hat{\varphi}(u) + \langle By, u \rangle + \hat{g}(y) + \mu_S d_S(y)]}_{A_1} + \\ &\quad \tau\underbrace{[-\hat{\varphi}(u_{\mu_S,\hat{x}}) + \langle By, u_{\mu_T,\hat{x}} \rangle + \hat{g}(y)]}_{A_2}\Big\} \end{aligned}$$

In order to simplify the calculations we first evaluate Expression A_1 and then Expression A_2. We start by considering Expression A_1 as a function of y, $H(y) := -\hat{\varphi}(u) + \langle By, u \rangle + \hat{g}(y) + \mu_S d_S(y)$. Note that $\varphi_{\mu_S}(u) = \min_{y \in S} H(y)$ and therefore $\arg\min_{y \in S} H(y) = x_{\mu_S,u}$ by definition. As H is differentiable and strongly convex with convexity parameter $\mu_S \sigma_S$, we have

$$\begin{aligned} H(y) &\geq H(x_{\mu_S,u}) + \frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2 \\ &= \varphi_{\mu_S}(u) + \frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2. \end{aligned}$$

Then, using the Excessive Gap condition, we have

$$\begin{aligned}
A_1 &\geq \varphi_{\mu_S}(u) + \frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2 \\
&\geq g_{\mu_T}(x) + \frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2 \qquad \text{(Excessive Gap condition)} \\
&\geq g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), x - \hat{x}\rangle + \frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2 \\
&= g_{\mu_T}(\hat{x}) + \tau\langle \nabla g_{\mu_T}(\hat{x}), x - x_{\mu_S,u}\rangle + \frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2.
\end{aligned}$$

Last equality holds, since $x - \hat{x} = \tau(x - x_{\mu_S,u})$ Now we consider Expression A_2. Using Lemma 5.3 we get

$$\begin{aligned}
A_2 &= -\hat{\varphi}(u_{\mu_T,\hat{x}}) + \langle By, u_{\mu_T,\hat{x}}\rangle + \hat{g}(y) \\
&\geq g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), y - \hat{x}\rangle \\
&= g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), y - (1-\tau)x - \tau x_{\mu_S,u}\rangle.
\end{aligned}$$

Finally putting together both expressions and using Inequality (5.17) we get

$$\begin{aligned}
\varphi_{\bar{\mu}_S}(\bar{u}) &\geq \min_{y\in S}\{(1-\tau)A_1 + \tau A_2\} \\
&\geq \min_{y\in S}\Big\{(1-\tau)\big[g_{\mu_T}(\hat{x}) + \tau\langle \nabla g_{\mu_T}(\hat{x}), x - x_{\mu_S,u}\rangle + \frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2\big] + \\
&\quad \tau\big[g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), y - (1-\tau)x - \tau x_{\mu_S,u}\rangle\big]\Big\} \\
&= \min_{y\in S}\{g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), \tau(y - x_{\mu_S,u})\rangle + (1-\tau)\frac{1}{2}\mu_S\sigma_S\|y - x_{\mu_S,u}\|_S^2\} \\
&\geq \min_{y\in S}\{g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), \tau(y - x_{\mu_S,u})\rangle + \frac{1}{2}L_{g_{\mu_T},S}\|\tau(y - x_{\mu_S,u})\|_S^2\} \\
&\quad \text{(Inequality (5.17) : } (1-\tau)\mu_S\sigma_S \geq L_{g_{\mu_T},S}\cdot\tau^2\text{)} \\
&\geq \min_{z\in S}\{g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), z - \hat{x}\rangle + \frac{1}{2}L_{g_{\mu_T},S}\|z - \hat{x}\|_S^2\} \\
&\quad \text{(Define } z := \hat{x} + \tau(y - x_{\mu_S,u}),\ \text{since } \hat{x} = (1-\tau)x + \tau x_{\mu_S,u} \in S, \\
&\quad \text{we have } z \in (1-\tau)x + \tau S \subseteq S\text{)} \\
&= g_{\mu_T}(\hat{x}) + \langle \nabla g_{\mu_T}(\hat{x}), \bar{x} - \hat{x}\rangle + \frac{1}{2}L_{g_{\mu_T},S}\|\bar{x} - \hat{x}\|_S^2 \\
&\quad (\bar{x} = GM_{g_{\mu_T}}(\hat{x})) \\
&\geq g_{\mu_T}(\bar{x}) = g_{\bar{\mu}_T}(\bar{x}).
\end{aligned}$$

□

Theorem 5.1 and Theorem 5.2 lie at the heart of Algorithm 4. Note that the difficult operations in the algorithm are first the evaluation of the primal and dual functions,

$g(x)$ and $\varphi(u)$, the different strongly convex projections onto S, $x_{\mu_S,u}$ and $x_{\mu_S,\hat{u}}$, respectively the strongly concave projections onto T, $u_{\mu_T,x}$ and $u_{\mu_T,\hat{x}}$, and finally both gradient mappings, $GM_{g_{\mu_T}}(\hat{x})$ and $GM_{\varphi_{\mu_S}}(\hat{u})$.

From this point on, we restrain ourselves to a special instance of the problem, where the Lipschitz constants of $\hat{g}$ and $\hat{\varphi}$ are equal to zero, $L_{\hat{g},S} = L_{\hat{\varphi},T} = 0$. Note that the LP instances that we are interested in, have this special structure.

Theorem 5.4 (Convergence of Excessive Gap Algorithm, [Nes05a] Lemma 4.1 and Theorem 6.3)
Assume Assumption 4 holds and consider Problem 5. Let the pairs of sequences $(\{x_k\}_{k\geq 0}, \{\mu_T^k\}_{k\geq 0})$ *and* $(\{u_k\}_{k\geq 0}, \{\mu_S^k\}_{k\geq 0})$ *be generated by Algorithm 4. Then, for each* $k \geq 0$ *the pairs* (x_k, μ_T^k) *and* (u_k, μ_S^k) *satisfy the excessive gap condition,*

$$g_{\mu_T^k}(x_k) \leq \varphi_{\mu_S^k}(u_k)$$

and

$$g(x_k) - \varphi(u_k) \leq \frac{4}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}. \tag{5.20}$$

Thus, the theorem shows that if we run the algorithm for $\frac{4}{\epsilon}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}$ iterations, we obtain an absolute accuracy of at least ϵ. In other words, the convergence rate of the Excessive Gap method is $O(\frac{1}{\epsilon})$.

Proof. A recursive argument exhibits the following behavior of the sequences $\{\mu_T^k\}_{k\geq 0}$ and $\{\mu_S^k\}_{k\geq 0}$. For k even, we have $\mu_S^k = \frac{1}{k+1}\mu_S^0$ and $\mu_T^k = \frac{2}{k+2}\mu_T^0$. For k odd, we have $\mu_S^k = \frac{1}{k+2}\mu_S^0$ and $\mu_T^k = \frac{2}{k+1}\mu_T^0$.

Now we show that at each iteration k, either Inequality (5.17) or (5.18) holds. For $k \geq 0$, we have $\frac{\tau_k^2}{1-\tau_k} = \frac{4}{(k+3)(k+1)}$. If k is even,

$$\frac{\mu_S^k \sigma_S}{L_{g_{\mu_T^k},S}} = \frac{\sigma_S \sigma_T}{\|B\|_{S,T}^2}\mu_S^k \mu_T^k = \frac{\sigma_S \sigma_T}{\|B\|_{S,T}^2}\mu_S^0 \mu_T^0 \frac{2}{(k+1)(k+2)} = \frac{4}{(k+1)(k+2)}$$

and if k is odd

$$\frac{\mu_T^k \sigma_T}{L_{\varphi_{\mu_S^k},T}} = \frac{\sigma_S \sigma_T}{\|B\|_{S,T}^2}\mu_S^k \mu_T^k = \frac{\sigma_S \sigma_T}{\|B\|_{S,T}^2}\mu_S^0 \mu_T^0 \frac{2}{(k+2)(k+1)} = \frac{4}{(k+2)(k+1)}.$$

Thus, inequalities (5.17) and (5.18) hold alternatively. Consequently, the Excessive Gap condition propagates from iteration to iteration provided that it is satisfied for $k = 0$, in view of Theorem 5.1 and Theorem 5.2. Hence, it remains to be shown

Algorithm 4 Excessive Gap Algorithm

Requires: - $x^o = \arg\min_{x \in S} d_s(x)$ $(d_S(x^o) = 0)$

- An initial smoothing factor $\mu_S^0 := 2\|B\|_{S,T}\sqrt{\frac{D_T}{\sigma_S \sigma_T D_S}}$
- An initial smoothing factor $\mu_T^0 := \|B\|_{S,T}\sqrt{\frac{D_S}{\sigma_S \sigma_T D_T}}$
- An absolute error $\epsilon > 0$

Ensures: An approximate primal solution $\bar{x} \in S$ and an approximate dual solution $\bar{u}$ such that $g(\bar{x}) - \varphi(\bar{u}) \leq \epsilon$

compute $x_0 = GM_{g_{\mu_T^0}}(x^o)$ `Quadratic projection onto` S
compute $u_0 = u_{\mu_S^0, x^o}$ `Strongly convex projection onto` T
set $k = 0$
while $g(x_k) - \varphi(u_k) > \epsilon$ **do**
  $\tau_k = \frac{2}{k+3}$
  if k is even **then**
    compute $x_{\mu_S^k, u_k} = \arg\min_{y \in S} \{\langle By, u_k\rangle + \hat{g}(y) + \mu_S^k d_S(y)\}$
    `Strongly convex projection onto` S
    set $\hat{x} = (1 - \tau_k)x_k + \tau_k x_{\mu_S^k, u_k}$
    compute $u_{\mu_T^k, \hat{x}} = \arg\max_{v \in T} \{\langle B\hat{x}, v\rangle - \hat{\varphi}(v) - \mu_T^k d_T(v)\}$
    `Strongly convex projection onto` T
    set $u_{k+1} = (1 - \tau_k)u_k + \tau_k u_{\mu_T^k, \hat{x}}$
    compute $x_{k+1} = GM_{g_{\mu_T^k}}(\hat{x})$ `Quadratic projection onto` S
    set $\mu_S^{k+1} = (1 - \tau_k)\mu_S^k$ and $\mu_T^{k+1} = \mu_T^k$
  end if
  if k is odd **then**
    compute $u_{\mu_T^k, x_k} = \arg\max_{v \in T} \{\langle Bx_k, v\rangle - \hat{\varphi}(v) - \mu_T^k d_T(v)\}$
    `Strongly convex projection onto` T
    set $\hat{u} = (1 - \tau_k)u_k + \tau_k u_{\mu_T^k, x_k}$
    compute $x_{\mu_S^k, \hat{u}} = \arg\min_{y \in S} \{\langle By, \hat{u}\rangle + \hat{g}(y) + \mu_S^k d_S(y)\}$
    `Strongly convex projection onto` S
    set $x_{k+1} = (1 - \tau_k)x_k + \tau_k x_{\mu_S^k, \hat{u}}$
    compute $u_{k+1} = GM_{\varphi_{\mu_S^k}}(\hat{u})$ `Quadratic projection onto` T
    $\mu_T^{k+1} = (1 - \tau_k)\mu_T^k$ and $\mu_S^{k+1} = \mu_S^k$
  end if
  $k = k + 1$ `Objective functions evaluation`
end while
set $\bar{x} = x_k$ and $\bar{u} = u_k$

that the Excessive Gap condition holds for $k = 0$.

$$\begin{aligned}
g_{\mu_T^0}(x_0) &\leq g_{\mu_T^0}(x^o) + \langle \nabla g_{\mu_T^0}(x^o), x_0 - x^o \rangle + \frac{L_{g_{\mu_T^0},S}}{2}\|x_0 - x^o\|_S^2 \\
&= \min_{x \in S} \{ g_{\mu_T^0}(x^o) + \langle \nabla g_{\mu_T^0}(x^o), x - x^o \rangle + \frac{L_{g_{\mu_T^0},S}}{2}\|x - x^o\|_S^2 \} \\
&\leq \min_{x \in S} \{ \hat{g}(x) + \langle Bx, u_0 \rangle - \hat{\varphi}(u_0) + \frac{L_{g_{\mu_T^0},S}}{2}\|x - x^o\|_S^2 \} \\
&\quad \text{(Lemma 5.3)} \\
&\leq \min_{x \in S} \{ \hat{g}(x) + \langle Bx, u_0 \rangle - \hat{\varphi}(u_0) + \frac{1}{2}\mu_S^0 \sigma_S \|x - x^o\|_S^2 \} \\
&\quad (\; L_{g_{\mu_T^0},S} \leq \mu_S^0 \sigma_S \;) \\
&\leq -\hat{\varphi}(u_0) + \min_{x \in S} \{ \hat{g}(x) + \langle Bx, u_0 \rangle + \mu_S^0 d_S(x) \} \\
&\quad (\; \frac{1}{2}\sigma_S \|x - x^o\|_S^2 \leq d_S(x) \quad \forall \; x \in S \;) \\
&= \varphi_{\mu_S^0}(u_0)
\end{aligned}$$

Now, Inequality (5.20) follows immediately from Inequality (5.11) and the particular form of μ_S^k and μ_T^k. □

5.2 Applying Excessive Gap Method to LPs

Recall that our objective is to solve large scale linear problems and for this purpose we consider their Langrange relaxations (see Chapter 3.1). This approach leads us to the following optimization problems, the primal problem

$$\min_{x \in Q} f(x), \qquad f(x) := c^T x + \max_{u \in P} \{ \langle Ax, u \rangle - b^T u \}$$

and the dual problem

$$\max_{u \in P} \psi(u), \qquad \psi(u) := -b^T u + \min_{x \in Q} \langle Ax, u \rangle + c^T x$$

with

$$\min_{x \in Q} f(x) = \max_{u \in P} \psi(u).$$

since $Q \subset \mathbb{R}^m, P \subset \mathbb{R}^n$ are assumed to be convex and compact.

We note that both problems fit perfectly the structure of the optimization problems considered in this chapter, see (5.1) and (5.2). Choosing appropriately the prox-functions $d_Q(x)$ and $d_P(u)$ with convexity parameters $\sigma_Q > 0$ and $\sigma_P > 0$ with respect to norm $\|.\|_Q$ and $\|.\|_P$, we define the following smooth approximations,

$$f_{\mu_P}(x) := c^T x + \max_{u \in P} \{ \langle Ax, u \rangle - b^T u - \mu_P d_P(u) \} \tag{5.21}$$

with $\nabla f_{\mu_P}(x) = c + A^T u_{\mu_P,x}$ and $L_{f_{\mu_P},Q} = \frac{\|A\|^2_{Q,P}}{\mu_P \sigma_P}$, see Equation (5.5), and

$$\psi_{\mu_Q}(u) := -b^T u + \min_{x \in Q} \{\langle Ax, u\rangle + c^T x + \mu_Q d_Q(x)\} \tag{5.22}$$

with $\nabla \psi_{\mu_Q}(u) = -b + A x_{\mu_Q,u}$ and $L_{\psi_{\mu_Q},P} = \frac{\|A\|^2_{Q,P}}{\mu_Q \sigma_Q}$, see Equation (5.7). Note that the corresponding $L_{\hat{g},S} = 0$ and $L_{\hat{\varphi},T} = 0$.

Therefore for an approximate primal and dual solution with an absolute gap of ϵ we need to compute at most $\frac{4}{\epsilon}\|A\|^2_{Q,P}\sqrt{\frac{D_Q D_P}{\sigma_Q \sigma_P}}$ iterations of Excessive Gap method.

As for the Primal-Dual Subgradient algorithms, the choice of the prox-functions as well as the norms is crucial for the performance of the Excessive Gap algorithm.

6. Approximate Oracles and the Optimization Methods

The Primal-Dual Subgradient method as well as the Excessive Gap method, are oracle based methods. As we mentioned in the previous chapter, the methods assume that the minimum of a specific class of strongly convex functions can be exactly determined. We suppose now that it is difficult or numerically expensive to compute this minimum, yet possible to get an approximation with a given guarantee without unacceptable numerical effort. Working with such approximate solutions is attractive but the question remains whether the algorithm will still converge or not? In Theorem 7 in [CE05], the result concerning the Excessive Gap method was presented without proof. The previous question arises when applying the Excessive Gap method to the Survivable Network Design problem since we face Minimum Quadratic Cost Flow problems as subproblems. In the following, we present the proof in details and extend the result of the paper to the Primal-Dual Subgradient methods.

6.1 Approximate Oracles

The main idea of the proof of Theorem 7 in [CE05] consists of using an oracle providing a solution with the guarantee of an absolute accuracy and the properties of strongly convex functions.

Definition 6.1
Let h be a a strongly convex and differentiable function defined over a convex and compact set $S \in \mathbb{R}^n$, and let $\sigma_S > 0$ be its convexity parameter with respect to the norm $\|.\|_S$. A δ-oracle *provides a* δ-approximation *x^δ of the minimum x^* of the function $h(x)$ over S, such that,*

$$h(x^\delta) - \delta \leq h(x^*) \leq h(x^\delta), \tag{6.1}$$

holds for $\delta > 0$.

In the next lemma the properties of such δ-approximations are described.

Lemma 6.2
Let $h(x)$ be a strongly convex and differentiable function defined over the convex and compact set $S \in \mathbb{R}^n$, and let $\sigma_S > 0$ denote its convexity parameter with respect to the norm $\|.\|_S$. For a given $\delta > 0$, let x^δ be a δ-approximation of the minimum x^* of function $h(x)$ over S with

$$h(x^\delta) - \delta \leq h(x^*) \leq h(x^\delta).$$

Then,

1. $\|x^\delta - x^*\|_S \leq \sqrt{2\delta/\sigma_S}$.
2. *For all $y \in S$,*

$$h(y) + \theta(\delta, C_S, \sigma_S) \geq h(x^\delta) + \frac{1}{2}\sigma_S \|y - x^\delta\|_S^2,$$

where $\theta(\delta, C_S, \sigma_S) := (2\delta + \sqrt{2\delta\sigma_S} C_S)$ and $C_S := \max\limits_{x,y \in S} \|x - y\|_S$.

Proof. The function h is strongly convex, thus

$$h(x^\delta) \geq h(x^*) + \langle \nabla h(x^*), x^\delta - x^* \rangle + \frac{1}{2}\sigma_S \|x^\delta - x^*\|_S^2$$

holds. Since x^* is the minimum of h over S, $\langle \nabla h(x^*), x^\delta - x^* \rangle \geq 0$ holds. Thus, we get $\frac{1}{2}\sigma_S \|x^\delta - x^*\|_S^2 \leq h(x^\delta) - h(x^*) \leq \delta$ and $\|x^\delta - x^*\|_S \leq \sqrt{2\delta/\sigma_S}$ holds.

The second property follows from the definition of a δ-oracle and the first statement of this lemma. Namely,

$$\begin{aligned} h(x^\delta) + \frac{1}{2}\sigma_S \|y - x^\delta\|_S^2 &\leq h(x^*) + \delta + \frac{1}{2}\sigma_S \|y - x^*\|_S^2 \\ &\quad + \sigma_S \|y - x^*\|_S \|x^* - x^\delta\|_S + \frac{1}{2}\sigma_S \|x^* - x^\delta\|_S^2 \\ &\leq h(y) + \delta + \sqrt{2\delta\sigma_S} C_S + \delta \\ &= h(y) + (2\delta + \sqrt{2\delta\sigma_S} C_S). \end{aligned}$$

□

6.2 Primal-Dual Subgradient Methods

In the Primal-Dual Subgradient algorithms we suppose that at each step k we can exactly compute x_k, the minimum of a strongly convex function, i.e.,

$$x_k := \arg \min_{x \in S} \{\langle \zeta_{k-1}, x \rangle + \beta_k d_S(x)\}$$

where $\beta_k > 0$ and $d_S(x)$ is a prox-function over S with convexity parameter $\sigma_S > 0$ with respect to a norm $\|.\|_S$ and with minimum attained at x^o, see Algorithm 2. Using the notation introduced in the previous chapter we define

$$\Gamma_{\beta_k}(x, \zeta) := \langle \zeta, x \rangle + \beta_k d_S(x) \tag{6.2}$$

and thus $x_k = \arg \min_{x \in S} \Gamma_{\beta_k}(x, \zeta_{k-1})$. Note that for fixed ζ, $\Gamma_{\beta_k}(x, \zeta)$ is also strongly convex with convexity parameter $\beta_k \sigma_S$ with respect to $\|.\|_S$.

Now, we assume that for $\delta > 0$ and fixed $\zeta_{k-1} \in \mathbb{R}^n$ a $\beta_k\delta$-oracle exists that delivers a $\beta_k\delta$-approximate solution $x_k^{\beta_k\delta}$ of x_k, i.e.,

$$\Gamma_{\beta_k}(x_k^{\beta_k\delta}, \zeta_{k-1}) - \beta_k\delta \leq \Gamma_{\beta_k}(x_k, \zeta_{k-1}) \leq \Gamma_{\beta_k}(x_k^{\beta_k\delta}, \zeta_{k-1}). \tag{6.3}$$

The only difference between the primal-dual subgradients algorithms presented in Chapter 4 and those we present next, is the use of $\beta_k\delta$-oracles instead of exact oracles at each step k. We generate $\beta_k\delta$ approximate solutions instead of exact solutions, see Algorithm 5. Recall that the objective of Algorithm 5 is to minimize approximately a convex function $g(x)$ defined over the set $S \subseteq \mathbb{R}^n$. It generates a approximation solution $\bar{x} \in S$ and a dual approximation solution $\bar{\zeta} \in \partial g(\bar{x})$. The dual objective function is $\varphi(\zeta) = -g_*(\zeta) + \min_{x \in S} \langle \zeta, x \rangle$.

Algorithm 5 Dual Averaging Algorithm with δ-Oracles

Requires: - A δ-oracle
Requires: - A prox-function $d_S(x)$ over S with convexity parameter $\sigma_S > 0$ with respect to norm $\|.\|_S$ and minimizer x^o over S
- A constant $\delta > 0$
- An absolute error $\epsilon > 0$
Ensures: An approximate primal solution $\bar{x} \in S$ and an approximate dual solution $\bar{\zeta}$ such that $g(\bar{x}) - \varphi(\bar{\zeta}) \leq \epsilon$.

set $k = 0$ and $\beta_0 = 1$
choose $\lambda_0 > 0$ and set $\Lambda_0 = \lambda_0$
compute $x^{\beta_0\delta}$, a $\beta_0\delta$-approximation of $x_0 = \arg\min_{x \in S} d_S(x)$ $\beta_0\delta$-`Oracle`
compute $\xi_0 \in \partial g(x_0^{\beta_0\delta})$ and set $\zeta_0 = \lambda_0\xi_0$ `Subgradient computation`
set $\bar{x} = \frac{\lambda_0 x_0^{\beta_0\delta}}{\Lambda_0}$ and $\bar{\zeta} = \frac{\lambda_0\xi_0}{\Lambda_0}$
while $g(\bar{x}) - \varphi(\bar{\zeta}) > \epsilon$ **do**
  set $k = k + 1$
  choose $\beta_k \geq \beta_{k-1}$ and compute $x_k^{\beta_k\delta}$, a $\beta_k\delta$-approximation of $x_k = \arg\min_{x \in S} \{\langle \zeta_{k-1}, x\rangle + \beta_k d_S(x)\}$ $\beta_k\delta$-`Oracle`
  choose $\lambda_k > 0$ and set $\Lambda_k = \Lambda_{k-1} + \lambda_k$
  compute $\xi_k \in \partial g(x_k^{\beta_k\delta})$ and $\zeta_k = \zeta_{k-1} + \lambda_k\xi_k$ `Subgradient computation`
  set $\bar{x} = \frac{1}{\Lambda_k}\sum_{i=0}^{k} \lambda_i x_i^{\beta_i\delta}$ and $\bar{\zeta} = \frac{1}{\Lambda_k}\zeta_k = \frac{1}{\Lambda_k}\sum_{i=0}^{k} \lambda_i\xi_i$ `Objective functions evaluation`
end while

Theorem 6.3 (Convergence of Primal-Dual Subgradient Algorithm with δ-Oracles)
Let $\bar{x}$ and $\bar{\zeta}$ be the solutions generated by Algorithm 5 after k iterations. Then

$$g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \right) + \sqrt{\frac{2\delta}{\sigma_S}} L$$

where $D_S := \max_{x \in S} d_S(x)$ and $\|\xi_i\|_S^* \leq L$ for $i = 1, \ldots, k$.

Before proving Theorem 6.3, we study the influence of the $\beta_k \delta$-approximation on the convergence of Algorithm 5 if it uses the simple averages sequences (4.12) or the weighted averages sequences (4.13). Given the properties of these sequences (Lemma 4.6), the convergence of Algorithm 5 results in Theorem 6.4. It turns out that to ensure a theoretical absolute error $0 < \epsilon \leq 1$ in $O(\frac{1}{\epsilon^2})$ iterations, δ has to be of the same order as ϵ^2.

Theorem 6.4
Let $\bar{x}$ and $\bar{\zeta}$ be the solutions generated by Algorithm 5 after k iterations using either simple averages sequences (4.12) or weighted averages sequences (4.13).

$$\text{If} \quad \delta = \frac{D_S}{k+1}, \quad \text{then} \quad g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{2\sqrt{2}L}{\sqrt{k+1}} \sqrt{\frac{D}{\sigma_S}}, \tag{6.4}$$

where $D_S := \max_{x \in S} d_S(x)$ and $\|\xi_i\|_S^* \leq L$ for $i = 1, \ldots, k$.

Proof. Putting together the results of Theorem 6.3 and Theorem 4.7, we get

$$g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{L}{\sqrt{k+1}} \sqrt{\frac{2D_S}{\sigma_S}} + \sqrt{\frac{2\delta}{\sigma_S}} L.$$

Then, using $\delta = \frac{D_S}{k+1}$, we obtain the desired result,

$$g(\bar{x}) - \varphi(\bar{\zeta}) \leq \frac{L}{\sqrt{k+1}} \sqrt{\frac{2D_S}{\sigma_S}} + \sqrt{\frac{2D_S}{\sigma_S(k+1)}} L \leq \frac{2\sqrt{2}L}{\sqrt{k+1}} \sqrt{\frac{D_S}{\sigma_S}}.$$

□

Let us now prove Theorem 6.3.

Proof of Theorem 6.3. This proof is similar to the proof of Theorem 4.5, except that the absolute error at each step of the algorithm must be carried along the calculations.

As in Algorithm 2, the solutions $\bar{x}$ and $\bar{\zeta}$ are primal and dual feasible since they are a convex combination of primal, respectively dual, feasible solutions. We evaluate

$$\begin{aligned} g(\bar{x}) - \varphi(\bar{\zeta}) &= g(\bar{x}) + g_*(\bar{\zeta}) - \min_{x\in S} \langle \bar{\zeta}, x\rangle \\ &\leq \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} \Big(g(x_i^{\beta_i\delta}) + g_*(\xi_i)\Big) - \min_{x\in S} \langle \sum_{i=0}^{k} \frac{\lambda_i \xi_i}{\Lambda_k}, x\rangle \\ &\quad \text{(by convexity of } g \text{ and } g_*) \\ &= \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} \langle \xi_i, x_i^{\beta_i\delta}\rangle - \min_{x\in S} \langle \sum_{i=0}^{k} \frac{\lambda_i \xi_i}{\Lambda_k}, x\rangle (\xi_i \in \partial g(x_i^{\beta_i\delta})) \\ &= \frac{1}{\Lambda_k} \max_{x\in S} \Big\{ \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i^{\beta_i\delta} - x\rangle \Big\}. \end{aligned}$$

We define $\Theta_{k+1} := \max_{x\in S} \Big\{ \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i^{\beta_i\delta} - x\rangle \Big\}$ and procede in two steps to find an upper bound,

$$\begin{aligned} \Theta_{k+1} &\leq \beta_{k+1} D_S + \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i^{\beta_i\delta} - x_0^{\beta_0\delta}\rangle + \nu^{\delta}_{\beta_{k+1}}(\zeta_k), \qquad (6.5) \\ &\leq \beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}} \sum_{i=0}^{k} \lambda_i \|\xi_i\|_S^*, \qquad (6.6) \end{aligned}$$

where

$$\begin{aligned} \nu^{\delta}_{\beta_{k+1}}(\zeta_k) &:= \langle \zeta_k, x_0^{\beta_0\delta}\rangle - \min_{x\in S} \Gamma_{\beta_{k+1}}(x, \zeta_k) \qquad (6.7) \\ &= \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_0^{\beta_0\delta}\rangle - \min_{x\in S} \{ \sum_{i=0}^{k} \lambda_i \langle \xi_i, x\rangle + \beta_{k+1} d_S(x)\}. \quad (\zeta_k = \sum_{i=0}^{k} \lambda_i \xi_i) \end{aligned}$$

First, recall that $\beta_0 = 1$. Then, note that as $\nu_{\beta_{k+1}}(\zeta_k)$ in proof of Theorem 4.5, (4.18), the function $\nu^{\delta}_{\beta_{k+1}}(\zeta_k)$ is convex and differentiable. Its gradient is given by $\nabla \nu^{\delta}_{\beta_{k+1}}(\zeta) = x^{\beta_0\delta} - x_k$ where $x_k = \arg\min_{x\in S} \Gamma_{\beta_{k+1}}(x, \zeta)$ and it is Lipschitz continuous over S with Lipschitz constant $L_{\nu^{\delta}_{\beta_{k+1}}, S} = \frac{1}{\beta_{k+1}\sigma_S}$ (see Theorem 3.7).

Let us first show Inequality (6.5),

$$\begin{aligned}
\beta_{k+1}D_S &+ \sum_{i=0}^{k}\lambda_i\langle\xi_i, x_i^{\beta_i\delta} - x_0^{\beta_0\delta}\rangle + \nu^{\delta}_{\beta_{k+1}}(\zeta_k) \\
&= \beta_{k+1}D_S + \sum_{i=0}^{k}\lambda_i\langle\xi_i, x_i^{\beta_i\delta} - x_0^{\beta_0\delta}\rangle + \sum_{i=0}^{k}\lambda_i\langle\xi_i, x_0^{\beta_0\delta}\rangle - \\
&\quad \min_{x\in S}\{\sum_{i=0}^{k}\lambda_i\langle\xi_i, x\rangle + \beta_{k+1}d_S(x)\} \\
&= \max_{x\in S}\{\beta_{k+1}(D_S - d_S(x)) + \sum_{i=0}^{k}\lambda_i\langle\xi_i, x_i^{\beta_i\delta} - x\rangle\} \\
&\geq \max_{x\in S}\sum_{i=0}^{k}\lambda_i\langle\xi_i, x_i^{\beta_i\delta} - x\rangle.
\end{aligned}$$

To show Inequality (6.6), we first note that $\nu^{\delta}_{\beta_{i+1}}(\zeta_i) \leq \nu^{\delta}_{\beta_i}(\zeta_i)$ holds for $i \geq 1$ because $\{\beta_i\}_{i\geq 1}$ is a increasing sequence and then

$$\begin{aligned}
\nu^{\delta}_{\beta_{i+1}}(\zeta_i) &\leq \nu^{\delta}_{\beta_i}(\zeta_i) \\
&\leq \nu^{\delta}_{\beta_i}(\zeta_{i-1}) + \langle\nabla\nu^{\delta}_{\beta_i}(\zeta_{i-1}), \zeta_i - \zeta_{i-1}\rangle + \frac{L_{\nu^{\delta}_{\beta_i},S}}{2}\|\zeta_i - \zeta_{i-1}\|_S^{*\,2} \\
&= \nu^{\delta}_{\beta_i}(\zeta_{i-1}) + \langle x_0^{\beta_0\delta} - x_i^{\beta_i\delta} + x_i^{\beta_i\delta} - x_i, \lambda_i\xi_i\rangle + \frac{1}{2\beta_i\sigma_S}\lambda_i^2\|\xi_i\|_S^{*\,2} \\
&= \nu^{\delta}_{\beta_i}(\zeta_{i-1}) + \langle x_0^{\beta_0\delta} - x_i^{\beta_i\delta}, \lambda_i\xi_i\rangle + \frac{1}{2\beta_i\sigma_S}\lambda_i^2\|\xi_i\|_S^{*\,2} + \langle x_i^{\beta_i\delta} - x_i, \lambda_i\xi_i\rangle \\
&\leq \nu^{\delta}_{\beta_i}(\zeta_{i-1}) + \langle x_0^{\beta_0\delta} - x_i^{\beta_i\delta}, \lambda_i\xi_i\rangle + \frac{1}{2\beta_i\sigma_S}\lambda_i^2\|\xi_i\|_S^{*\,2} + \|x_i^{\beta_i\delta} - x_i\|_S\lambda_i\|\xi_i\|_S^* \\
&\leq \nu^{\delta}_{\beta_i}(\zeta_{i-1}) + \langle x_0^{\beta_0\delta} - x_i^{\beta_i\delta}, \lambda_i\xi_i\rangle + \frac{1}{2\beta_i\sigma_S}\lambda_i^2\|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}}\lambda_i\|\xi_i\|_S^*. \\
&\qquad \text{(Lemma 6.2 — convexity parameter of } \Gamma_{\beta_i}(x, \zeta_{i-1}) : \ \beta_i\sigma_S)
\end{aligned}$$

Hence, for $i \geq 1$,

$$\nu^{\delta}_{\beta_{i+1}}(\zeta_i) - \nu^{\delta}_{\beta_i}(\zeta_{i-1}) \leq \lambda_i\langle\xi, x_0^{\beta_0\delta} - x_i^{\beta_i\delta}\rangle + \frac{\lambda_i^2}{2\beta_i\sigma_S}\|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}}\lambda_i\|\xi_i\|_S^*.$$

Summing up over all $i \geq 1$ we get

$$\nu^{\delta}_{\beta_{k+1}}(\zeta_k) - \nu^{\delta}_{\beta_1}(\zeta_0) \leq \sum_{i=1}^{k}\lambda_i\langle\xi_i, x_0^{\beta_0\delta} - x_i^{\beta_i\delta}\rangle + \frac{1}{2\sigma_S}\sum_{i=1}^{k}\frac{\lambda_i^2}{\beta_i}\|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}}\sum_{i=1}^{k}\lambda_i\|\xi_i\|_S^*.$$

Given that

$$\begin{aligned}
\nu^{\delta}_{\beta_1}(\zeta_0) &\leq \nu^{\delta}_{\beta_1}(0) + \langle \nabla \nu^{\delta}_{\beta_1}(0), \zeta_0 \rangle + \frac{L_{\nu^{\delta}_{\beta_1}, S}}{2} \|\zeta_0\|_S^{*\,2} \\
&= \nu^{\delta}_{\beta_1}(0) + \langle x_0^{\beta_0 \delta} - x_0, \lambda_0 \xi_0 \rangle + \frac{\lambda_0^2}{2\beta_1 \sigma_S} \|\xi_0\|_S^{*\,2} \\
&\quad (\nu^{\delta}_{\beta_1}(0) \leq 0 \text{ since } d_S(x) \geq 0 \ \forall \ x \in S) \\
&\leq \frac{1}{2\beta_1 \sigma_S} \lambda_0^2 \|\xi_0\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}} \lambda_0 \|\xi_0\|_S^*
\end{aligned}$$

we have

$$\nu^{\delta}_{\beta_{k+1}}(\zeta_k) \leq \sum_{i=0}^{k} \lambda_i \langle \xi_i, x_0^{\beta_0 \delta} - x_i^{\beta_i \delta} \rangle + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}} \sum_{i=0}^{k} \lambda_i \|\xi_i\|_S^*$$

and thus

$$\sum_{i=0}^{k} \lambda_i \langle \xi_i, x_i^{\beta_i \delta} - x_0^{\beta_0 \delta} \rangle + \nu^{\delta}_{\beta_{k+1}}(\zeta_k) \leq \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}} \sum_{i=0}^{k} \lambda_i \|\xi_i\|_S^*$$

and Inequality (6.6) is proved. Finally, we get

$$\begin{aligned}
g(\bar{x}) - \varphi(\bar{\zeta}) &\leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \Theta_{k+1} \\
&\leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}} \sum_{i=0}^{k} \lambda_i \|\xi_i\|_S^* \right) \\
&\leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} + \sqrt{\frac{2\delta}{\sigma_S}} \sum_{i=0}^{k} \lambda_i L \right) \\
&\quad (\|\xi_i\|_S^* \leq L \ \forall \ i = 1, \ldots, k) \\
&\leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma_S} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i\|_S^{*\,2} \right) + \sqrt{\frac{2\delta}{\sigma_S}} L
\end{aligned}$$

□

Approximate Subgradients.

Each iteration of the primal-dual subgradient method requires not only the computation of a minimizer of a smooth strongly convex function but also the computation of a subgradient of the objective function. The influence of the approximate minimizer in the convergence of the methods was studied in the previous section. Here,

we investigate the influence of approximate subgradients. The definition as well as results presented in this subsection are standard (see, e.g., a work by Chudak and Guarisco, [CG06], concerning the use of subgradient methods for stochastic optimization problems).

Definition 6.5 (abc-subgradients)
Let $a, b > 0$ and $c \geq 0$. For a convex function g, ξ_x^{abc} is an abc-subgradient *of g at $x \in dom\ g$ if*

$$ag(x) + \langle \xi_x^{abc}, y - x \rangle \leq bg(y) + c \qquad \forall\ y \in dom\ g. \tag{6.8}$$

The 110-subgradients correspond to the usual subgradients. The hyperplane defined by a $11c$-subgradient, ξ_x^{11c}, defines a supporting hyperplane after a translation of at most c in the opposite direction of ξ_x^{11c}. An abc-subgradient will perturb the

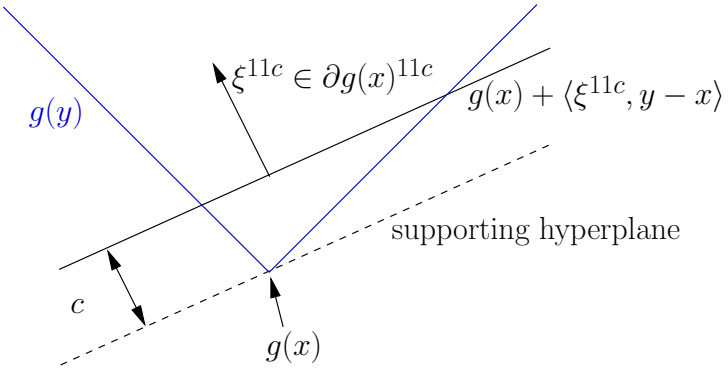


Fig. 6.1: $11c$-subgradient

convergence of the primal-dual subgradient less than an approximate minimizer. In particular, we show in the following theorem that the error due to the approximate subgradients will not propagate through the iterations. In [CG06], Theorem A.3, Chudak and Guarisco show a similar result for the Standard Subgradient Method.

Theorem 6.6 (Convergence of Primal-Dual Subgradient Algorithm with Approximate Subgradient)
Let $a, b > 0$ and $c \geq 0$. Moreover, suppose that at each iteration i of Algorithm 2, an abc-subgradient ξ_i^{abc} of x_i is computed instead of an exact subgradient ξ_i. After k iterations, denote the average primal solution by $\bar{x} := \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} x_i$ and the average dual solution by $\bar{\zeta}^{abc} := \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} \xi_i^{abc}$. Then,

$$ag(\bar{x}) - b\varphi(\frac{\bar{\zeta}^{abc}}{b}) \leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i^{abc}\|_S^{*\,2} \right) + c, \tag{6.9}$$

where $D_S := \max_{x \in S} d_S(x)$.

Proof.

$$\begin{aligned} ag(\bar{x}) - b\varphi(\frac{\bar{\zeta}^{abc}}{b}) &\leq ag(\bar{x}) + bg_*(\frac{\bar{\zeta}^{abc}}{b}) - \min_{x\in S} \langle \bar{\zeta}^{abc}, x\rangle \\ &\leq \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} \left(ag(x_i) + bg_*(\frac{\xi_i^{abc}}{b}) \right) - \min_{x\in S} \langle \bar{\zeta}^{abc}, x\rangle \\ &\leq \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} \left(\langle \xi_i^{abc}, x_i\rangle + c \right) - \frac{1}{\Lambda_k} \min_{x\in S} \sum_{i=0}^{k} \lambda_i \langle \xi_i^{abc}, x\rangle \\ &\leq \frac{1}{\Lambda_k} \max_{x\in S} \left\{ \sum_{i=0}^{k} \lambda_i \langle \xi_i^{abc}, x_i - x\rangle \right\} + c \end{aligned}$$

The fourth inequality holds due to the definition of abc-subgradients (Definition 6.5). Namely,

$$\begin{aligned} ag(x_i) + bg_*(\frac{\xi_i^{abc}}{b}) &= ag(x_i) + b \sup_{y\in \mathrm{dom}\ g} \{\langle \frac{\xi_i^{abc}}{b}, y\rangle - g(y)\} \\ &= ag(x_i) + \sup_{y\in \mathrm{dom}\ g} \{\langle \xi_i^{abc}, y\rangle - bg(y)\} \\ &\leq ag(x_i) + \langle \xi_i^{abc}, x_i\rangle - ag(x_i) + c \\ &= \langle \xi_i^{abc}, x_i\rangle + c. \end{aligned}$$

The remaining part of the proof can be deduced directly from the proof of Theorem 4.5, since from here on, the fact that ξ_i is a subgradient of g at x_i for $i \geq 0$ is not used anymore. □

Note that if we suppose that $a = b = 1$, i.e., that the computed abc-subgradients are exact subgradients up to a constant c, the claim of the previous theorem results in the following corollary.

Corollary 6.7
Let $a = b = 1$ and $c \geq 0$. Moreover, suppose that at each iteration i of Algorithm 2, an abc-subgradient ξ_i^{11c} of x_i is computed instead of an exact subgradient ξ_i. After k iterations, denote the average primal solution by $\bar{x} := \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} x_i$ and the average dual solution by $\bar{\zeta}^{11c} := \sum_{i=0}^{k} \frac{\lambda_i}{\Lambda_k} \xi_i^{11c}$. Then,

$$g(\bar{x}) - \varphi(\bar{\zeta}^{abc}) \leq \frac{1}{\sum_{i=0}^{k} \lambda_i} \left(\beta_{k+1} D_S + \frac{1}{2\sigma} \sum_{i=0}^{k} \frac{\lambda_i^2}{\beta_i} \|\xi_i^{abc}\|_S^{*\,2} \right) + c, \tag{6.10}$$

where $D_S := \max_{x\in S} d_S(x)$.

6.3 Excessive Gap with Approximate Gradients

Compared to the primal-dual subgradient methods, working with approximate oracles within the methods using smoothing techniques and gradient mappings needs additional care. Namely, the errors induced by the approximate oracles influence not only the computed approximate solutions but also the definition of the gradient mapping.

Next, let us recall the kind of functions that can be minimized using the Excessive Gap method (see Chapter 5). For $x \in S \subset \mathbb{R}^n$ and $u \in T \subset \mathbb{R}^m$,

$$g(x) := \hat{g}(x) + \max_{v \in T} \{\langle Bx, v\rangle - \hat{\varphi}(v)\} \tag{6.11}$$

is the primal function and

$$\varphi(x) := -\hat{\varphi}(u) + \min_{y \in S} \{\langle By, u\rangle + \hat{g}(y)\} \tag{6.12}$$

is the dual function. The sets S and T are assumed to be convex and compact. The function $\hat{g}(x)$ and $\hat{\varphi}(u)$ are convex and differentiable. At each step k of the Excessive Gap method, approximate smooth functions of $g(x)$ and $\varphi(u)$ are defined. Namely, for $\mu_T^k > 0$ and $\mu_S^k > 0$, we have

$$g_{\mu_T^k}(x) := \hat{g}(x) + \max_{v \in T} \{\langle Bx, v\rangle - \hat{\varphi}(v) - \mu_T^k d_T(v)\} \tag{6.13}$$

where $d_T(v)$ is a prox-function over T with convexity parameter $\sigma_T > 0$ with respect to a norm $\|.\|_T$ and

$$\varphi_{\mu_S^k}(u) := -\hat{\varphi}(u) + \min_{y \in S} \{\langle By, u\rangle + \hat{g}(y) + \mu_S^k d_S(y)\} \tag{6.14}$$

where $d_S(y)$ is a prox-function over S with convexity parameter $\sigma_S > 0$ with respect to a norm $\|.\|_S$. To simplify the notation we introduce as in Chapter 5 the following auxiliary functions

$$\Gamma_{\mu_T^k}(x, u) := \langle Bx, u\rangle - \hat{\varphi}(u) - \mu_T^k d_T(u) \tag{6.15}$$

and

$$\Phi_{\mu_S^k}(x, u) := \langle Bx, u\rangle + \hat{g}(x) + \mu_S^k d_S(x). \tag{6.16}$$

Then, the approximate smooth function can be rewritten as $g_{\mu_T^k}(x) := \hat{g}(x) + \max_{v \in T} \Gamma_{\mu_T^k}(x, v)$ and $\varphi(u)_{\mu_S} := -\hat{\varphi}(u) + \min_{y \in S} \Phi_{\mu_S^k}(x, y)$. For fixed $x \in S$, we note that $-\Gamma_{\mu_T^k}(x, .)$ is smooth and strongly convex with convexity parameter $\mu_T^k \sigma_T$ and for fixed $u \in T$, $\Phi_{\mu_S^k}(., u)$ is smooth and strongly convex with convexity parameter $\mu_S^k \sigma_S$. Thus, the approximate functions $g_{\mu_T^k}(x)$ and $\varphi(u)_{\mu_S^k}$ are convex respectively

concave, differentiable, and their gradients are Lipschitz continuous, see Theorem 3.7. The gradients and their Lipschitz constants are given as follows

$$\nabla g_{\mu_T^k}(x) = \nabla \hat{g}(x) + B^T u_{\mu_T^k,x} \qquad \nabla \varphi_{\mu_S^k}(u) = -\nabla \hat{\varphi}(u) + B x_{\mu_S^k,u} \tag{6.17}$$

$$L_{g_{\mu_T^k},S} = L_{\hat{g},S} + \frac{\|B\|_{S,T}^2}{\mu_T^k \sigma_T} \qquad L_{\varphi_{\mu_S^k},T} = L_{\hat{\varphi},T} + \frac{\|B\|_{S,T}^2}{\mu_S^k \sigma_S} \tag{6.18}$$

where $L_{\hat{g},S}$ is the Lipschitz constant of $\nabla \hat{g}(x)$ over S, $L_{\hat{\varphi},T}$ is the Lipschitz constant of $\nabla \hat{\varphi}(x)$ over T, $x_{\mu_S^k,u}$ is the minimizer of $\Phi_{\mu_S^k}(., u)$ over S, and $u_{\mu_T^k,x}$ is the maximizer of $\Gamma_{\mu_T^k}(x, .)$ over T.

At each step of the Excessive Gap method, $u_{\mu_T^k,x}$ and $x_{\mu_S^k,u}$ must be computed for a given $x \in S$ and for a given $u \in T$. In Chapter 5 we assumed that both optima are delivered by exact oracles. Here we work with δ-oracles (see Definition 6.1). We consider a δ-approximation of $u_{\mu_T^k,x}$ and a δ-approximation of $x_{\mu_S^k,u}$ at each step k. Both solutions, $u_{\mu_T^k,x}$ and $x_{\mu_S^k,u}$, are needed for evaluating the primal and dual approximate functions as well as for computing their gradients and thus, for defining the gradient mapping. In the following, we investigate how δ-approximations perturb the evaluation of the approximate functions and their gradients.

Lemma 6.8
For $x \in S$, $\mu_T > 0$, and $\delta > 0$, define

$$\begin{aligned} g_{\mu_T}^{\delta}(x) &:= \hat{g}(x) + \Gamma_{\mu_T}(x, u_{\mu_T,x}^{\delta}) \\ &= \hat{g}(x) + \langle Bx, u_{\mu_T,x}^{\delta} \rangle - \hat{\varphi}(u_{\mu_T,x}^{\delta}) - \mu_T d_T(u_{\mu_T,x}^{\delta}) \end{aligned} \tag{6.19}$$

$$\nabla g_{\mu_T}^{\delta}(x) := \nabla \hat{g}(x) + B^T u_{\mu_T,x}^{\delta}, \tag{6.20}$$

where $u_{\mu_T,x}^{\delta}$ is a δ-approximation of $u_{\mu_T,x}$ with guaranteed accuracy δ, i.e.,

$$\Gamma_{\mu_T}(x, u_{\mu_T,x}^{\delta}) \leq \Gamma_{\mu_T}(x, u_{\mu_T,x}) \leq \Gamma_{\mu_T}(x, u_{\mu_T,x}^{\delta}) + \delta.$$

Then,

1. $g_{\mu_T}^{\delta}(x) \leq g_{\mu_T}(x) \leq g_{\mu_T}^{\delta}(x) + \delta$,
2. $\|\nabla g_{\mu_T}(x) - \nabla g_{\mu_T}^{\delta}(x)\|_S^* \leq \varepsilon(\delta, \mu_T \sigma_T)$,

where $\varepsilon(\delta, \mu_T \sigma_T) := \sqrt{\frac{2\delta}{\mu_T \sigma_T}} \|B\|_{S,T}$.

Proof. These results follow directly from the definition of $u_{\mu_T,x}^{\delta}$.

1. Since $\Gamma_{\mu_T}(x, u^\delta_{\mu_T,x}) \le \Gamma_{\mu_T}(x, u_{\mu_T,x}) \le \Gamma_{\mu_T}(x, u^\delta_{\mu_T,x}) + \delta$, we have

$$\underbrace{\hat{g}(x) + \Gamma_{\mu_T}(x, u^\delta_{\mu_T,x})}_{g^\delta_{\mu_T}(x)} \le \underbrace{\hat{g}(x) + \Gamma_{\mu_T}(x, u_{\mu_T,x})}_{g_{\mu_T}(x)} \le \underbrace{\hat{g}(x) + \Gamma_{\mu_T}(x, u^\delta_{\mu_T,x}) + \delta}_{g^\delta_{\mu_T}(x)+\delta}.$$

2. Recall that $\nabla g_{\mu_T}(x) = \nabla \hat{g}(x) + B^T u_{\mu_T,x}$ where $u_{\mu_T,x} = \arg\max_{u\in T} \Gamma_{\mu_T}(x,u)$, then

$$\begin{aligned} \|\nabla g_{\mu_T}(x) - \nabla g^\delta_{\mu_T}(x)\|^*_S &\le \|B^T u_{\mu_T,x} - B^T u^\delta_{\mu_T,x}\|^*_S \le \|B\|_{S,T}\|u_{\mu_T,x} - u^\delta_{\mu_T,x}\|_T \\ &\le \|B\|_{S,T}\sqrt{\frac{2\delta}{\mu_T\sigma_T}} = \varepsilon(\delta, \mu_T\sigma_T) \end{aligned}$$

The last inequality follows from Lemma 6.2 applied to $\Gamma_{\mu_T}(x,u)$. Since $u^\delta_{\mu_T,x}$ is a δ-approximation of u_x and $-\Gamma_{\mu_T}(x,u)$ is smooth and strongly convex with convexity parameter $\mu_T\sigma_T$, we have $\|u_x - u^\delta_{\mu_T,x}\|_T \le \sqrt{\frac{2\delta}{\mu_T\sigma_T}}$. □

The previous lemma can be written in a dual form as follows.

Lemma 6.9
For $u \in T$, $\mu_S > 0$, and $\delta > 0$, define

$$\begin{aligned} \varphi^\delta_{\mu_S}(u) &:= -\hat{\varphi}(u) + \Phi_{\mu_S}(x^\delta_{\mu_S,u}, u) \\ &= -\hat{\varphi}(u) + \langle Bx^\delta_{\mu_S,u}, u\rangle + \hat{g}(x^\delta_{\mu_S,u}) + \mu_S d_S(x^\delta_{\mu_S,u}) \end{aligned} \tag{6.21}$$

$$\nabla\varphi^\delta_{\mu_S}(u) := -\nabla\hat{\varphi}(u) + Bx^\delta_{\mu_S,u}, \tag{6.22}$$

where $x^\delta_{\mu_S,u}$ is a δ-approximation of $x_{\mu_S,u}$ with guaranteed accuracy δ, i.e.,

$$\Phi_{\mu_S}(x^\delta_{\mu_S,u}, u) - \delta \le \Phi_{\mu_S}(x_{\mu_S,u}, u) \le \Phi_{\mu_T}(x^\delta_{\mu_S,u}, u).$$

Then,

1. $\varphi^\delta_{\mu_S}(u) - \delta \le \varphi_{\mu_S}(u) \le \varphi^\delta_{\mu_S}(u)$,
2. $\|\nabla\varphi_{\mu_S}(u) - \nabla\varphi^\delta_{\mu_S}(u)\|^*_T \le \varepsilon(\delta, \mu_S\sigma_S)$,

where $\varepsilon(\delta, \mu_S\sigma_S) := \sqrt{\frac{2\delta}{\mu_S\sigma_S}}\|B\|_{S,T}$.

The approximation functions $g^\delta_{\mu_T}$ and $\varphi^\delta_{\mu_S}$ are respectively convex and concave with Lipschitz continuous gradient. In the following lemma and its corollary we investigate similar properties for the functions $g^\delta_{\mu_T}(x)$ and $\varphi^\delta_{\mu_S}(u)$.

Lemma 6.10 (Properties of $g^{\delta}_{\mu_T}(x)$)
For all $x, y \in S$, we have

1. $\|\nabla g^{\delta}_{\mu_T}(x) - \nabla g^{\delta}_{\mu_T}(y)\|^*_S \leq L_{g_{\mu_T},S}\|y - x\|_S + 2\varepsilon(\delta, \mu_T\sigma_T)$,
2. $g^{\delta}_{\mu_T}(y) + \varepsilon(\delta, \mu_T\sigma_T)\|y - x\|_S + \delta \geq g^{\delta}_{\mu_T}(x) + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle$,
3. $g^{\delta}_{\mu_T}(y) \leq g^{\delta}_{\mu_T}(x) + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle + \frac{L_{g_{\mu_T},S}}{2}\|y - x\|^2_S + \varepsilon(\delta, \mu_T\sigma_T)\|y - x\|_S + \delta$,

with $L_{g_{\mu_T},S} := L_{\hat{g},S} + \frac{\|B\|^2_{S,T}}{\mu_T\sigma_T}$ *and* $\varepsilon(\delta, \mu_T\sigma_T) := \sqrt{\frac{2\delta}{\mu_T\sigma_T}}\|B\|_{S,T}$.

Proof. To prove the properties of $g^{\delta}_{\mu_T}(x)$ we use Lemma 6.8 and the properties of $g_{\mu_T}(x)$ recalled above and shown in Theorem 3.7.

1. $$\begin{aligned}\|\nabla g^{\delta}_{\mu_T}(x) - \nabla g^{\delta}_{\mu_T}(y)\|^*_S &\leq \|\nabla g^{\delta}_{\mu_T}(x) - \nabla g_{\mu_T}(x)\|^*_S + \|\nabla g_{\mu_T}(x) - \nabla g_{\mu_T}(y)\|^*_S \\ &\quad + \|\nabla g_{\mu_T}(y) - \nabla g^{\delta}_{\mu_T}(y)\|^*_S \\ &\leq \varepsilon(\delta, \mu_T\sigma_T) + \|\nabla g_{\mu_T}(x) - \nabla g_{\mu_T}(y)\|^*_S + \varepsilon(\delta, \mu_T\sigma_T) \\ &\quad \text{(Lemma 6.10, statement 1)} \\ &\leq 2\varepsilon(\delta, \mu_T\sigma_T) + L_{g_{\mu_T},S}\|x - y\|_S. \\ &\quad (\nabla g_{\mu_T}\text{is Lipschitz continuous with constant } L_{g_{\mu_T},S})\end{aligned}$$

2. $$\begin{aligned} g^{\delta}_{\mu_T}(y) &\geq g_{\mu_T}(y) - \delta \qquad \text{(Lemma 6.10, statement 1)} \\ &\geq g_{\mu_T}(x) + \langle \nabla g_{\mu_T}(x), y - x\rangle - \delta \qquad (g_{\mu_T}(x) \text{ is smooth and convex}) \\ &\geq g^{\delta}_{\mu_T}(x) + \langle \nabla g_{\mu_T}(x), y - x\rangle - \delta \qquad \text{(Lemma 6.10, statement 1)} \\ &= g^{\delta}_{\mu_T}(x) + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle - \langle \nabla g^{\delta}_{\mu_T}(x) - \nabla g_{\mu_T}(x), y - x\rangle - \delta \\ &\geq g^{\delta}_{\mu_T}(x) + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle - \|\nabla g^{\delta}_{\mu_T}(x) - \nabla g_{\mu_T}(x)\|^*_S\|y - x\|_S - \delta \\ &\geq g^{\delta}_{\mu_T}(x) + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle - \varepsilon(\delta, \mu_T\sigma_T)\|y - x\|_S - \delta \\ &\quad \text{(Lemma 6.10, statement 2)}\end{aligned}$$

3. $$\begin{aligned} g^{\delta}_{\mu_T}(y) &\leq g_{\mu_T}(y) \qquad \text{(Lemma 6.10, statement 1)} \\ &\leq g_{\mu_T}(x) + \langle \nabla g_{\mu_T}(x), y - x\rangle + \frac{L_{g_{\mu_T},S}}{2}\|y - x\|^2_S \\ &\quad (g_{\mu_T}(x) \text{ is convex and smooth with Lipschitz continuous gradient}) \\ &\leq g^{\delta}_{\mu_T}(x) + \delta + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle + \|\nabla g_{\mu_T}(x) - \nabla g^{\delta}_{\mu_T}(x)\|^*_S\|y - x\|_S \\ &\quad + \frac{L_{g_{\mu_T},S}}{2}\|y - x\|^2_S \qquad \text{(Lemma 6.10, statement 1)} \\ &\leq g^{\delta}_{\mu_T}(x) + \delta + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle + \varepsilon(\delta, \mu_T\sigma_T)\|y - x\|_S \\ &\quad + \frac{L_{g_{\mu_T},S}}{2}\|y - x\|^2_S\end{aligned}$$

□

Again, we can formulate a dual version of the previous result.

Lemma 6.11 (Properties of $\varphi^{\delta}_{\mu_S}(u)$)
For all $u, v \in T$, we have

1. $\|\nabla\varphi^{\delta}_{\mu_S}(v) - \nabla\varphi^{\delta}_{\mu_S}(u)\|^*_T \leq L_{\varphi_{\mu_S},T}\|u - v\|_T + 2\varepsilon(\delta, \mu_S\sigma_S)$,
2. $\varphi^{\delta}_{\mu_S}(v) \leq \varphi^{\delta}_{\mu_S}(u) + \langle\nabla\varphi^{\delta}_{\mu_S}(u), v - u\rangle + \varepsilon(\delta, \mu_S\sigma_S)\|v - u\|_T + \delta$,
3. $\varphi^{\delta}_{\mu_S}(v)+\varepsilon(\delta, \mu_S\sigma_S)\|v-u\|_T+\delta \geq \varphi^{\delta}_{\mu_S}(u)+\langle\nabla\varphi^{\delta}_{\mu_S}(u), v - u\rangle - \frac{L_{\varphi_{\mu_S},T}}{2}\|v-u\|^2_S$,

with $L_{\varphi_{\mu_S},T} := L_{\hat{\varphi},T} + \frac{\|B\|^2_{S,T}}{\mu_S\sigma_S}$ *and* $\varepsilon(\delta, \mu_S\sigma_S) := \sqrt{\frac{2\delta}{\mu_S\sigma_S}}\|B\|_{S,T}$.

The influence of the δ-oracles on the main objects of the Excessive Gap method—the Gradient Mappings defined below—remains to be studied.

$$\begin{aligned} GM_{g_{\mu_T}}(x) &:= \arg\min_{y\in S} Z_{\mu_T,x}(y) \\ &= \arg\min_{y\in S}\{g_{\mu_T}(x) + \langle\nabla g_{\mu_T}(x), y - x\rangle + \frac{1}{2}L_{g_{\mu_T},S}\|y - x\|^2_S\} \end{aligned} \tag{6.23}$$

$$\begin{aligned} GM_{\varphi_{\mu_S}}(u) &:= \arg\max_{v\in T} W_{\mu_S,u}(v) \\ &= \arg\max_{v\in T}\{\varphi_{\mu_S}(u) + \langle\nabla\varphi_{\mu_S}(u), v - u\rangle - \frac{1}{2}L_{\varphi_{\mu_S},T}\|v - u\|^2_T\} \end{aligned} \tag{6.24}$$

Our first remark concerns the gradient of both approximation functions, i.e., $\nabla g_{\mu_T}(x)$ and $\nabla\varphi_{\mu_S}(u)$. Their evaluation requires computing the maximum of the function $\Gamma_{\mu_T}(x, .)$ over T and respectively the minimum of the function $\Phi_{\mu_S}(., u)$ over S, see (6.17). Since we assume that we can only compute δ-approximations of these optima, we now aim at evaluating the optima of the following functions

$$Z^{\delta}_{\mu_T,x}(y) := g^{\delta}_{\mu_T}(x) + \langle\nabla g^{\delta}_{\mu_T}(x), y - x\rangle + \frac{1}{2}L_{g_{\mu_T},S}\|y - x\|^2_S \tag{6.25}$$

and

$$W^{\delta}_{\mu_S,u} := \varphi^{\delta}_{\mu_S}(u) + \langle\nabla\varphi^{\delta}_{\mu_S}(u), v - u\rangle - \frac{1}{2}L_{\varphi_{\mu_S},T}\|v - u\|^2_T, \tag{6.26}$$

where $g^{\delta}_{\mu_T}(x)$ and $\nabla g^{\delta}_{\mu_T}(x)$ are defined as in Lemma 6.8 and $\varphi^{\delta}_{\mu_S}(u)$ and $\nabla\varphi^{\delta}_{\mu_S}(u)$ are defined as in Lemma 6.9. Then, we denote by $GM^{\delta}_{g_{\mu_T}}(x)$ the δ-approximation of the minimum y^* of $Z^{\delta}_{\mu_T,x}(y)$ over S, and by $GM^{\delta}_{\varphi\mu_S}(u)$ the δ-approximation of the maximum v^* of $W^{\delta}_{\mu_S,u}(v)$ over T, i.e.,

$$\begin{aligned} Z^{\delta}_{\mu_T,x}(GM^{\delta}_{g_{\mu_T}}(x)) - \delta &\leq Z^{\delta}_{\mu_T,x}(y^*) \leq Z^{\delta}_{\mu_T,x}(GM^{\delta}_{g_{\mu_T}}(x)) \\ W^{\delta}_{\mu_S,u}(GM^{\delta}_{\varphi\mu_S}(u)) &\leq W^{\delta}_{\mu_S,u}(v^*) \leq W^{\delta}_{\mu_S,u}(GM^{\delta}_{\varphi\mu_S}(u)) + \delta \end{aligned}$$

At a fixed point $x \in S$ we have the following relation between the Gradient Mapping $GM_{g_{\mu_T}}(x)$ and its approximation $GM^{\delta}_{g_{\mu_T}}(x)$,

$$Z_{\mu_T,x}(GM_{g_{\mu_T}}(x)) \leq Z^{\delta}_{\mu_T,x}(GM^{\delta}_{g_{\mu_T}}(x)) + \delta + \varepsilon(\delta, \mu_T\sigma_T)\|GM^{\delta}_{g_{\mu_T}}(x) - x\|_S, \quad (6.27)$$

where $\varepsilon(\delta, \mu_T\sigma_T) = \sqrt{\frac{2\delta}{\mu_S\sigma_S}}\|B\|_{S,T}$. Namely,

$$\begin{aligned}
Z_{\mu_T,x}(GM_{g_{\mu_T}}(x)) &\leq Z_{\mu_T,x}(GM^{\delta}_{g_{\mu_T}}(x)) \\
&\leq g^{\delta}_{\mu_T}(x) + \delta + \langle \nabla g^{\delta}_{\mu_T}(x), GM^{\delta}_{g_{\mu_T}}(x) - x\rangle + \\
&\quad \langle \nabla g_{\mu_T}(x) - \nabla g^{\delta}_{\mu_T}(x), GM^{\delta}_{g_{\mu_T}}(x) - x\rangle \\
&\quad + \frac{1}{2}L_{g_{\mu_T},S}\|GM^{\delta}_{g_{\mu_T}}(x) - x\|^2_S \qquad \text{(Lemma 6.10, statement 1)} \\
&\leq Z^{\delta}_{\mu_T,x}(GM^{\delta}_{g_{\mu_T}}(x)) + \delta \\
&\quad + \|\nabla g_{\mu_T}(x) - \nabla g^{\delta}_{\mu_T}(x)\|^*_S\|GM^{\delta}_{g_{\mu_T}}(x) - x\|_S \\
&\leq Z^{\delta}_{\mu_T,x}(GM^{\delta}_{g_{\mu_T}}(x)) + \delta + \varepsilon(\delta, \mu_T\sigma_T)\|GM^{\delta}_{g_{\mu_T}}(x) - x\|_S. \\
&\quad \text{(Lemma 6.10, statement 2)}
\end{aligned}$$

Algorithm 6 corresponds to the Excessive Gap method using the above defined approximate Gradient Mappings and δ-oracles.

In order to investigate the convergence of Algorithm 6, we will first study the influence of the δ-oracles on the Excessive Gap conditions ($g_{\mu^k_T}(x_k) - \varphi_{\mu^k_S}(u_k) \leq 0 \ \forall\ k$). For this sake, we show that Lemma 5.3 also holds when δ-oracles are used.

Lemma 6.12
For any $x, y \in S$ and $\mu_T > 0$ we have

$$g^{\delta}_{\mu_T}(y) + \langle \nabla g^{\delta}_{\mu_T}(y), x - y\rangle \leq \hat{g}(x) + \langle Bx, u^{\delta}_{\mu_T,y}\rangle - \hat{\varphi}(u^{\delta}_{\mu_T,y}),$$

where $u^{\delta}_{\mu_T,y}$ is a δ-approximation of $u_{\mu_T,y} := \arg\max_{u\in T}\{\langle By, u\rangle - \hat{\varphi}(u) - \mu_T d_T(u)\}$.

Proof. By definition of $g^{\delta}_{\mu_T}$ and convexity of $\hat{g}$ we have for any $x, y \in S$ and $\mu_T > 0$,

$$\begin{aligned}
g^{\delta}_{\mu_T}(y) &+ \langle \nabla g^{\delta}_{\mu_T}(y), x - y\rangle \\
&= \hat{g}(y) + \langle By, u^{\delta}_{\mu_T,y}\rangle - \hat{\varphi}(u^{\delta}_{\mu_T,y}) - \mu_T d_T(u^{\delta}_{\mu_T,y}) + \\
&\quad \langle \nabla\hat{g}(y) + B^T u^{\delta}_{\mu_T,y}, x - y\rangle \\
&\leq \hat{g}(x) + \langle Bx, u^{\delta}_{\mu_T,y}\rangle - \hat{\varphi}(u^{\delta}_{\mu_T,y}) - \mu_T d_T(u^{\delta}_{\mu_T,y}) \\
&\leq \hat{g}(x) + \langle Bx, u^{\delta}_{\mu_T,y}\rangle - \hat{\varphi}(u^{\delta}_{\mu_T,y}).
\end{aligned}$$

The last inequality holds since $d_T(u) \geq 0$ for all $u \in T$ is zero. □

Algorithm 6 Excessive Gap Algorithm with Approximate Oracles

Requires: - A δ-oracle with $\delta > 0$
- $x^o = \min_{x \in S} d_s(x)$ $(d_S(x^o) = 0)$
- An initial smoothing factor $\mu_S^0 = 2\|B\|_{S,T}\sqrt{\frac{D_T}{\sigma_S \sigma_T D_S}}$
- An initial smoothing factor $\mu_T^0 = \|B\|_{S,T}\sqrt{\frac{D_S}{\sigma_S \sigma_T D_T}}$
- An absolute error $\epsilon > 0$

Ensures: An approximate primal solution $\bar{x} \in S$ and an approximate dual solution $\bar{u}$ such that $g(\bar{x}) - \varphi(\bar{u}) \leq \epsilon$

compute $\bar{x}_0 = GM_{g_{\mu_T^0}}(x^o)$ a δ-approximation of

$$\arg \min_{y \in S} \left\{ \langle \nabla \hat{g}(x^o) + B^T u^{\delta}_{\mu_T^k, x^o}, y - x^o \rangle + \tfrac{1}{2} L_{g_{\mu_T^k}, S} \|y - x^o\|_S^2 \right\} \quad \delta\texttt{-oracle}$$

compute $\bar{u}_0 = u_{\mu_S^0, x^o}$, a δ-approximation of $\arg \max_{v \in T} \{\langle Bx^o, v \rangle - \hat{\varphi}(v) - \mu_T^k d_T(v)\}$ $\delta\texttt{-oracle}$

$k = 0$ `Objective functions evaluation`

while $g(\bar{x}_k) - \varphi(\bar{u}_k) > \epsilon$ **do**

  $\tau_k = \frac{2}{k+3}$

  if k is even **then**

    compute $x^{\delta}_{\mu_S^k, \bar{u}_k}$ $\delta\texttt{-oracle}$

    set $\hat{x} = (1 - \tau_k)\bar{x}_k + \tau_k x^{\delta}_{\mu_S^k, \bar{u}_k}$

    compute $u^{\delta}_{\mu_T^k, \hat{x}}$ $\delta\texttt{-oracle}$

    set $\bar{u}_{k+1} = (1 - \tau_k)\bar{u}_k + \tau_k u^{\delta}_{\mu_T^k, \hat{x}}$

    compute $\bar{x}_{k+1} = GM^{\delta}_{g_{\mu_T^k}}(\hat{x})$ $\delta\texttt{-oracle}$

    set $\mu_S^{k+1} = (1 - \tau_k)\mu_S^k$ and $\mu_T^{k+1} = \mu_T^k$

  end if

  if k is odd **then**

    compute $u^{\delta}_{\mu_T^k, \bar{x}_k}$ $\delta\texttt{-oracle}$

    set $\hat{u} = (1 - \tau_k)\bar{u}_k + \tau_k u^{\delta}_{\mu_T^k, \bar{x}_k}$

    compute $x^{\delta}_{\mu_S^k, \hat{u}}$ $\delta\texttt{-oracle}$

    set $\bar{x}_{k+1} = (1 - \tau_k)\bar{x}_k + \tau_k x^{\delta}_{\mu_S^k, \hat{u}}$

    compute $\bar{u}_{k+1} = GM^{\delta}_{\varphi_{\mu_S^k}}(\hat{u})$ $\delta\texttt{-oracle}$

    set $\mu_T^{k+1} = (1 - \tau_k)\mu_T^k$ and $\mu_S^{k+1} = \mu_S^k$

  end if

  $k = k + 1$ `Objective functions evaluation`

end while

set $\bar{x} = \bar{x}_k$ and $\bar{u} = \bar{u}_k$

Details for the even steps of Algorithm 6

if k is even **then**
compute $x^{\delta}_{\mu_S^k,\bar{u}_k}$, a δ-approximation of $\arg\min_{y\in S}\{\langle By,\bar{u}_k\rangle+\hat{g}(y)+\mu_S^k d_S(y)\}$ δ-oracle
set $\hat{x}=(1-\tau_k)\bar{x}_k+\tau_k x^{\delta}_{\mu_S^k,\bar{u}_k}$
compute $u^{\delta}_{\mu_T^k,\hat{x}}$, a δ-approximation of $\arg\max_{v\in T}\{\langle B\hat{x},v\rangle-\hat{\varphi}(v)-\mu_T^k d_T(v)\}$ δ-oracle
set $\bar{u}_{k+1}=(1-\tau_k)\bar{u}_k+\tau_k u^{\delta}_{\mu_T^k,\hat{x}}$
compute $\bar{x}_{k+1}=GM^{\delta}_{g_{\mu_T^k}}(\hat{x})$, a δ-approximation of
$\arg\min_{y\in S}\left\{\langle\nabla\hat{g}(\hat{x})+B^T u^{\delta}_{\mu_T^k,\hat{x}},y-\hat{x}\rangle+\frac{1}{2}L_{g_{\mu_T^k,S}}\|y-\hat{x}\|_S^2\right\}$ δ-oracle
set $\mu_S^{k+1}=(1-\tau_k)\mu_S^k$ and $\mu_T^{k+1}=\mu_T^k$
end if

Details for the odd steps of Algorithm 6

if k is odd **then**
compute $u^{\delta}_{\mu_T^k,\bar{x}_k}$, a δ-approximation of $\arg\max_{v\in T}\{\langle B\bar{x}_k,v\rangle-\hat{\varphi}(v)-\mu_T^k d_T(v)\}$ δ-oracle
set $\hat{u}=(1-\tau_k)\bar{u}_k+\tau_k u^{\delta}_{\mu_T^k,\bar{x}_k}$
compute $x^{\delta}_{\mu_S^k,\hat{u}}$, a δ-approximation of $\arg\min_{y\in S}\{\langle By,\hat{u}\rangle+\hat{g}(y)+\mu_S^k d_S(y)\}$ δ-oracle
set $\bar{x}_{k+1}=(1-\tau_k)\bar{x}_k+\tau_k x^{\delta}_{\mu_S^k,\hat{u}}$
compute $\bar{u}_{k+1}=GM^{\delta}_{\varphi_{\mu_S^k}}(\hat{u})$, a δ-approximation of
$\arg\max_{v\in T}\{\langle\nabla\hat{\varphi}(\hat{u})+Bx^{\delta}_{\mu_S^k,\hat{u}},v-\hat{u}\rangle+\frac{1}{2}L_{\varphi_{\mu_S^k,T}}\|v-\hat{u}\|_T^2\}$ δ-oracle
set $\mu_T^{k+1}=(1-\tau_k)\mu_T^k$ and $\mu_S^{k+1}=\mu_S^k$
end if

Theorem 6.13 (Excessive Gap Condition for Approximate Oracles) *Let $\delta > 0$, $x \in S$, $u \in T$, and $\mu_T, \mu_S > 0$ satisfy the excessive gap condition up to an error $E(\delta)$, i.e.,*

$$g_{\mu_T}(x) \leq \varphi_{\mu_S}(u) + E(\delta).$$

For $\tau \in (0,1)$ compute

$$\begin{aligned}
x^{\delta}_{\mu_S,u} &:= \delta\text{-approximation of } \arg\min_{y\in S} \Phi_{\mu_S^k}(y,u) \\
&= \delta\text{-approximation of } \arg\min_{y\in S} \{\langle By, u\rangle + \hat{g}(y) + \mu_S^k d_S(y)\} \\
\hat{x} &:= (1-\tau)x + \tau x^{\delta}_{\mu_S,u} \\
u^{\delta}_{\mu_T,\hat{x}} &:= \delta\text{-approximation of } \arg\max_{v\in T} \Gamma_{\mu_T^k}(\hat{x}, v) \\
&\quad \delta\text{-approximation of } \arg\max_{v\in T} \{\langle B\hat{x}, u\rangle - \hat{\varphi}(v) - \mu_T^k d_T(v)\} \\
\bar{u} &:= (1-\tau)u + \tau u^{\delta}_{\mu_T,\hat{x}} \\
\bar{x} &:= GM^{\delta}_{g_{\mu_T}}(\hat{x}) \\
\bar{\mu}_S &:= (1-\tau)\mu_S \\
\bar{\mu}_T &:= \mu_T.
\end{aligned}$$

Then, provided that τ is chosen in accordance to

$$\frac{\tau^2}{1-\tau} \leq \frac{\mu_S \sigma_S}{L_{g_{\mu_T},S}}, \tag{6.28}$$

the pairs $(\bar{x}, \bar{\mu}_T)$ and $(\bar{u}, \bar{\mu}_S)$ satisfy the excessive gap condition up to an error $\bar{E}(\delta)$,

$$g_{\mu_T}(\bar{x}) \leq \varphi_{\mu_S}(\bar{u}) + \bar{E}(\delta)$$

where

$$\bar{E}(\delta) := (2-\tau)\left(3\delta + 4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\sqrt{\frac{\delta}{\mu_T D_T}}\right) + (1-\tau)\left(4\sqrt{\delta \mu_S D_S} + E(\delta)\right).$$

Proof. We follow the lines of the proof of Theorem 5.1.

$$\begin{aligned}
\varphi_{\bar{\mu}_S}(\bar{u}) &= -\hat{\varphi}(\bar{u}) + \min_{y\in S}\{\langle By, \bar{u}\rangle + \hat{g}(y) + \bar{\mu}_S d_S(y)\} \\
&= -\hat{\varphi}(\bar{u}) + \min_{y\in S}\{\langle By, \bar{u}\rangle + \hat{g}(y) + (1-\tau)\mu_S d_S(y)\} \\
&\quad (\ \bar{\mu}_S = (1-\tau)\mu_S\ ) \\
&\geq -(1-\tau)\hat{\varphi}(u) - \tau\hat{\varphi}(u^{\delta}_{\mu_S,\hat{x}}) + \\
&\quad \min_{y\in S}\left\{\langle By, (1-\tau)u + \tau u^{\delta}_{\mu_S,\hat{x}}\rangle + \hat{g}(y) + (1-\tau)\mu_S d_S(y)\right\} \\
&\quad (\text{by convexity of } \hat{\varphi} \text{ and definition of } \bar{u})
\end{aligned}$$

$$= \min_{y\in S}\Big\{(1-\tau)\underbrace{[-\hat{\varphi}(u)+\langle By,u\rangle+\hat{g}(y)+\mu_S d_S(y)]}_{A_1}+ \tau\underbrace{[-\hat{\varphi}(u^\delta_{\mu_S,\hat{x}})+\langle By,u^\delta_{\mu_S,\hat{x}}\rangle+\hat{g}(y)]}_{A_2}\Big\}$$

Now consider the expression A_1 as a function of y, that is, $H(y) := -\hat{\varphi}(u) + \langle By,u\rangle+\hat{g}(y)+\mu_S d_S(y)$. The function H is differentiable and strongly convex with convexity parameter $\mu_S\sigma_S$ and has $x_{\mu_S,u}$ as unique minimizer over S. For $x^\delta_{\mu_S,u}$, a δ-approximation of $x_{\mu_S,u}$,

$$\begin{aligned} H(y) &\geq H(x^\delta_{\mu_S,u})+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\theta(\delta,C_S,\mu_S\sigma_S)\\ &= \varphi^\delta_{\mu_S}(u)+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\theta(\delta,C_S,\mu_S\sigma_S)\end{aligned}$$

holds with $\theta(\delta,C_S,\mu_S\sigma_S) := 2\delta+\sqrt{2\delta\mu_S\sigma_S}C_S$ and $C_S = \max_{x,y\in S}\|x-y\|_S$, see Lemma 6.2 statement 2. Thus,

$$\begin{aligned} A_1 &\geq \varphi^\delta_{\mu_S}(u)+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\theta(\delta,C_S,\mu_S\sigma_S)\\ &\geq \varphi_{\mu_S}(u)+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\theta(\delta,C_S,\mu_S\sigma_S)\\ &\quad\text{(Lemma 6.9 statement 1)}\\ &\geq g_{\mu_T}(x)-E(\delta)+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\theta(\delta,C_S,\mu_S\sigma_S)\\ &\quad\text{(Excessive Gap condition)}\\ &\geq g^\delta_{\mu_T}(x)-E(\delta)+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\theta(\delta,C_S,\mu_S\sigma_S)\\ &\quad\text{(Lemma 6.8 statement 1)}\\ &\geq g^\delta_{\mu_T}(\hat{x})+\langle\nabla g^\delta_{\mu_T}(\hat{x}),x-\hat{x}\rangle-E(\delta)-\varepsilon(\delta,\mu_T\sigma_T)\|x-\hat{x}\|_S-\delta\\ &\quad+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\theta(\delta,C_S,\mu_S\sigma_S)\\ &\quad\text{(Lemma 6.10 statement 2)}\\ &= g^\delta_{\mu_T}(\hat{x})+\tau\langle\nabla g^\delta_{\mu_T}(\hat{x}),x-x^\delta_{\mu_S,u}\rangle+\frac{1}{2}\mu_S\sigma_S\|y-x^\delta_{\mu_S,u}\|_S^2-\delta\\ &\quad-\varepsilon(\delta,\mu_T\sigma_T)\tau\|x-x^\delta_{\mu_S,u}\|_S-\theta(\delta,C_S,\mu_S\sigma_S)-E(\delta).\\ &\quad(x-\hat{x}=\tau(x-x^\delta_{\mu_S,u}))\end{aligned}$$

Now we consider expression A_2. Using Lemma 6.12 and the definition of $\hat{x}$, we get

$$\begin{aligned} A_2 &= -\hat{\varphi}(u^\delta_{\mu_S,\hat{x}})+\langle By,u^\delta_{\mu_S,\hat{x}}\rangle+\hat{g}(y)\\ &\geq g^\delta_{\mu_T}(\hat{x})+\langle\nabla g^\delta_{\mu_T}(\hat{x}),y-\hat{x}\rangle\\ &= g^\delta_{\mu_T}(\hat{x})+\langle\nabla g^\delta_{\mu_T}(\hat{x}),y-(1-\tau)x-\tau x^\delta_{\mu_S,u}\rangle\end{aligned}$$

Putting both expressions together we get

$$
\begin{aligned}
\varphi_{\bar{\mu}_S}(\bar{u}) &\geq \min_{y\in S}\{(1-\tau)A_1+\tau A_2\}\\
&\geq \min_{y\in S}\Big\{(1-\tau)\big[g^{\delta}_{\mu_T}(\hat{x})+\tau\langle\nabla g^{\delta}_{\mu_T}(\hat{x}),x-x^{\delta}_{\mu_S,u}\rangle+\frac{1}{2}\mu_S\sigma_S\|y-x^{\delta}_{\mu_S,u}\|^2_S\big]\\
&\quad+\tau\big[g^{\delta}_{\mu_T}(\hat{x})+\langle\nabla g^{\delta}_{\mu_T}(\hat{x}),y-(1-\tau)x-\tau x^{\delta}_{\mu_S,u}\rangle\big]\Big\}\\
&\quad-(1-\tau)\big[\delta+\varepsilon(\delta,\mu_T\sigma_T)\|\tau(x-x^{\delta}_{\mu_S,u})\|_S+\theta(\delta,C_S,\mu_S\sigma_S)+E(\delta)\big]\\
&= \min_{y\in S}\Big\{g^{\delta}_{\mu_T}(\hat{x})+\langle\nabla g^{\delta}_{\mu_T}(\hat{x}),\tau(y-x^{\delta}_{\mu_S,u})\rangle+(1-\tau)\frac{1}{2}\mu_S\sigma_S\|y-x^{\delta}_{\mu_S,u}\|^2_S\Big\}\\
&\quad-(1-\tau)\big[\delta+\varepsilon(\delta,\mu_T\sigma_T)\|\tau(x-x^{\delta}_{\mu_S,u})\|_S+\theta(\delta,C_S,\mu_S\sigma_S)\\
&\quad+E(\delta)\big]\\
&\geq \min_{y\in S}\Big\{g^{\delta}_{\mu_T}(\hat{x})+\langle\nabla g^{\delta}_{\mu_T}(\hat{x}),\tau(y-x^{\delta}_{\mu_S,u})\rangle+\frac{1}{2}L_{g_{\mu_T},S}\|\tau(y-x^{\delta}_{\mu_S,u})\|^2_S\Big\}\\
&\quad-(1-\tau)\big[\delta+\varepsilon(\delta,\mu_T\sigma_T)\|\tau(x-x^{\delta}_{\mu_S,u})\|_S+\theta(\delta,C_S,\mu_S\sigma_S)\\
&\quad+E(\delta)\big]\\
&\quad\text{(Inequality (6.28) : } (1-\tau)\mu_S\sigma_S\geq\tau^2L_{g_{\mu_T},S})\\
&\geq \min_{z\in S}\Big\{g^{\delta}_{\mu_T}(\hat{x})+\langle\nabla g^{\delta}_{\mu_T}(\hat{x}),z-\hat{x}\rangle+\frac{1}{2}L_{g_{\mu_T},S}\|z-\hat{x}\|^2_S\Big\}-(1-\tau)\big[\,\delta\\
&\quad+\varepsilon(\delta,\mu_T\sigma_T)\|\tau(x-x^{\delta}_{\mu_S,u})\|_S+\theta(\delta,C_S,\mu_S\sigma_S)+E(\delta)\big]\\
&\quad\text{(Define } z:=\hat{x}+\tau(y-x^{\delta}_{\mu_S,u}),\ \text{since } \hat{x}=(1-\tau)x+\tau x^{\delta}_{\mu_S,u}\in S,\\
&\quad\text{we have } z\in(1-\tau)x+\tau S\subseteq S)\\
&\geq g^{\delta}_{\mu_T}(\hat{x})+\langle\nabla g^{\delta}_{\mu_T}(\hat{x}),\bar{x}-\hat{x}\rangle+\frac{1}{2}L_{g_{\mu_T},S}\|\bar{x}-\hat{x}\|^2_S-\delta\\
&\quad-(1-\tau)\big[\delta+\varepsilon(\delta,\mu_T\sigma_T)\|x-\hat{x}\|_S+\theta(\delta,C_S,\mu_S\sigma_S)+E(\delta)\big]\\
&\quad\text{(Definition of } \bar{x}:=GM^{\delta}_{g_{\mu_T}}(\hat{x}) \text{ and } \hat{x})\\
&\geq g^{\delta}_{\mu_T}(\bar{x})-2\delta-\varepsilon(\delta,\mu_T\sigma_T)\|\bar{x}-\hat{x}\|_S\\
&\quad-(1-\tau)\big[\delta+\varepsilon(\delta,\mu_T\sigma_T)\|x-\hat{x}\|_S+\theta(\delta,C_S,\mu_S\sigma_S)+E(\delta)\big]\\
&\quad\text{(Lemma 6.10 statement 3)}\\
&\geq g_{\bar{\mu}_T}(\bar{x})-3\delta-\varepsilon(\delta,\mu_T\sigma_T)\|\bar{x}-\hat{x}\|_S\\
&\quad-(1-\tau)\big[\delta+\varepsilon(\delta,\mu_T\sigma_T)\|x-\hat{x}\|_S+\theta(\delta,C_S,\mu_S\sigma_S)+E(\delta)\big]\\
&\quad\text{(Lemma 6.10 statement 1 and } \bar{\mu}_T=\mu_T)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
\bar{E}(\delta) &= 3\delta+\varepsilon(\delta,\mu_T\sigma_T)\|\bar{x}-\hat{x}\|_S\\
&\quad+(1-\tau)\big[\delta+\varepsilon(\delta,\mu_T\sigma_T)\|(x-\hat{x}\|_S+\theta(\delta,C_S,\mu_S\sigma_S)+E(\delta)\big].
\end{aligned}
$$

Since $d_S(x)$ is strongly convex, we have $\|x - x^o\|_S \leq \sqrt{2Ds/\sigma_S}$ for any $x \in S$, where x^o minimizes $d_S(x)$ over S. Therefore, for any $x, y \in S$,

$$\|x - y\|_S \leq \|x - x^o\|_S + \|x^o - y\|_S \leq 2\sqrt{\frac{2Ds}{\sigma_S}}.$$

Then,

$$\begin{aligned}
\bar{E}(\delta) &= (4-\tau)\delta + (2-\tau)\varepsilon(\delta, \mu_T\sigma_T)\|(x - \hat{x}\|_S + (1-\tau)\theta(\delta, C_S, \mu_S\sigma_S) \\
&\quad +(1-\tau)E(\delta) \\
&= (4-\tau)\delta + (2-\tau)\varepsilon(\delta, \mu_T\sigma_T)\|(x - \hat{x}\|_S \ + (1-\tau)(2\delta + \sqrt{2\delta\mu_S\sigma_S}\|y - \|_S) \\
&\quad +(1-\tau)E(\delta) \\
&\quad \text{(Definition of } \theta(\delta, C_S, \mu_S\sigma_S)) \\
&\leq (6-3\tau)\delta + 2(2-\tau)\sqrt{\frac{2\delta}{\mu_T\sigma_T}}\|B\|_{S,T}\sqrt{\frac{2D_S}{\sigma_S}} + 2(1-\tau)\sqrt{2\delta\mu_S\sigma_S}\sqrt{\frac{2D_S}{\sigma_S}} \\
&\quad +(1-\tau)E(\delta) \\
&\quad \text{(Definition of } \varepsilon(\delta, \mu_T\sigma_T) \text{ and } \|x - y\|_S \leq 2\sqrt{\frac{2Ds}{\sigma_S}} \ \forall\ x, y \in S) \\
&= 3(2-\tau)\delta + 4(2-\tau)\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S\sigma_T}}\sqrt{\frac{\delta}{\mu_T D_T}} + 4(1-\tau)\sqrt{\delta\mu_S D_S} \\
&\quad +(1-\tau)E(\delta).
\end{aligned}$$

□

The symmetric result of Theorem 6.13 for the dual step is claimed in the next theorem.

Theorem 6.14
Let $\delta > 0$, $x \in S$, $u \in T$, and $\mu_T, \mu_S > 0$ satisfy the excessive gap condition up to an error $E(\delta)$, i.e.,

$$g_{\mu_T}(x) \leq \varphi_{\mu_S}(u) + E(\delta).$$

For $\tau \in (0,1)$ compute

$$\begin{aligned}
u^{\delta}_{\mu_T,x} &:= \delta\text{-approximation of } \arg\max_{v \in T} \Gamma_{\mu_T^k}(x, v) \\
\hat{u} &:= (1-\tau)u + \tau u^{\delta}_{\mu_T,x} \\
x^{\delta}_{\mu_S,\hat{u}} &:= \delta\text{-approximation of } \arg\min_{y \in S} \Phi_{\mu_S^k}(y, \hat{u}) \\
\bar{x} &:= (1-\tau)x + \tau x^{\delta}_{\mu_S,\hat{u}} \\
\bar{u} &:= GM^{\delta}_{\varphi_{\mu_S}}(\hat{u}) \\
\bar{\mu}_T &:= (1-\tau)\mu_T \\
\bar{\mu}_S &:= \mu_S.
\end{aligned}$$

Then, provided that τ is chosen in accordance to

$$\frac{\tau^2}{1-\tau} \leq \frac{\mu_T \sigma_T}{L_{\varphi_{\mu_S},T}}, \tag{6.29}$$

the pairs $(\bar{x}, \bar{\mu}_T)$ and $(\bar{u}, \bar{\mu}_S)$ satisfy the excessive gap condition up to an error $\bar{E}(\delta)$,

$$g_{\mu_T}(\bar{x}) \leq \varphi_{\mu_S}(\bar{u}) + \bar{E}(\delta)$$

where

$$\bar{E}(\delta) := (2-\tau)\left(3\delta + 4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\sqrt{\frac{\delta}{\mu_S D_S}}\right) + (1-\tau)\left(4\sqrt{\delta \mu_T D_T} + E(\delta)\right).$$

Combining the previous results, we derive the following theorem, which is the key to evaluating the order of the propagated error δ.

Theorem 6.15
Let the pairs of sequences $(\{\bar{x}_k\}_{k\geq 0}, \{\mu_T^k\}_{k\geq 0})$ and $(\{\bar{u}_k\}_{k\geq 0}, \{\mu_S^k\}_{k\geq 0})$ be generated by Algorithm 6. Then, for each $k \geq 0$ the pairs $(\bar{x}_k, \mu_T^k)$ and $(\bar{u}_k, \mu_S^k)$ satisfy the excessive gap condition,

$$g_{\mu_T^k}(\bar{x}_k) \leq \varphi_{\mu_S^k}(\bar{u}_k) + E_k(\delta) \tag{6.30}$$

where

$$\begin{aligned}
E_0(\delta) &= 3\delta + \sqrt{\delta}\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right)^{\frac{1}{2}} \\
E_k(\delta) &= 6\delta k + 2\sqrt{2}\sqrt{\delta}\sum_{i=1}^{k}\left(\sqrt{i+1} + \frac{1}{\sqrt{i}}\right)\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right)^{\frac{1}{2}} + E_0(\delta) \\
&\quad \text{for } k \geq 1.
\end{aligned}$$

Proof. We first prove that Inequality (6.30) holds for $k = 0$.

$$\begin{aligned}
g_{\mu_T^0}(\bar{x}_0) &\leq g^{\delta}_{\mu_T^0}(\bar{x}_0) + \delta \qquad \text{(Lemma 6.8 statement 1)} \\
&\leq g^{\delta}_{\mu_T^0}(x^o) + \langle \nabla g^{\delta}_{\mu_T^0}(x^o), \bar{x}_0 - x^o\rangle + \frac{L_{g_{\mu_T^0},S}}{2}\|\bar{x}_0 - x^o\|_S^2 \\
&\quad + 2\delta + \varepsilon(\delta, \mu_T^0 \sigma_T)\|\bar{x}_0 - x^o\|_S \qquad \text{(Lemma 6.10 statement 3)} \\
&\leq \min_{x\in S}\left\{g^{\delta}_{\mu_T^0}(x^o) + \langle \nabla g^{\delta}_{\mu_T^0}(x^o), x - x^o\rangle + \frac{L_{g_{\mu_T^0},S}}{2}\|x - x^o\|_S^2\right\} \\
&\quad + 3\delta + \varepsilon(\delta, \mu_T^0 \sigma_T)\|\bar{x}_0 - x^o\|_S \qquad \text{(Definition of } GM^{\delta}_{g_{\mu_T^0}}(x^o))
\end{aligned}$$

$$
\begin{aligned}
&\leq \min_{x\in S}\left\{g^{\delta}_{\mu^0_T}(x^o)+\langle\nabla g^{\delta}_{\mu^0_T}(x^o),x-x^o\rangle+\frac{1}{2}\mu^0_T\sigma_T\|x-x^o\|^2_S\right\}\\
&\quad+3\delta+\varepsilon(\delta,\mu^0_T\sigma_T)\|\bar{x}_0-x^o\|_S \qquad (L_{g_{\mu^0_T},S}=\frac{1}{2}\mu^0_T\sigma_T)\\
&\leq \min_{x\in S}\left\{g^{\delta}_{\mu^0_T}(x^o)+\langle\nabla g^{\delta}_{\mu^0_T}(x^o),x-x^o\rangle+\mu^0_T d_S(x)\right\}+3\delta\\
&\quad+\varepsilon(\delta,\mu^0_T\sigma_T)\|\bar{x}_0-x^o\|_S \qquad (d_S(x)\geq\frac{1}{2}\sigma_S\|\bar{x}_0-x^o\|^2_S)\\
&= \min_{x\in S}\left\{\hat{g}(x^o)+\langle Bx^o,u^{\delta}_{\mu^0_T,x^o}\rangle-\hat{\varphi}(u^{\delta}_{\mu^0_T,x^o})-\mu^0_T d_T(u^{\delta}_{\mu^0_T,x^o})\right.\\
&\quad\left.+\langle\nabla\hat{g}(x^o)+B^T u^{\delta}_{\mu^0_T,x^o},x-x^o\rangle+\mu^0_S d_S(x)\right\}+3\delta\\
&\quad+\varepsilon(\delta,\mu^0_T\sigma_T)\|\bar{x}_0-x^o\|_S\\
&\quad(\text{Definition of } g^{\delta}_{\mu^0_T}(x^o) \text{ and } \nabla g^{\delta}_{\mu^0_T}(x^o) \text{ in Lemma 6.8})\\
&\leq \min_{x\in S}\left\{\hat{g}(x)+\langle Bx,u^{\delta}_{\mu^0_T,x^o}\rangle+\mu^0_S d_S(x)\right\}-\hat{\varphi}(u^{\delta}_{\mu^0_T,x^o})+3\delta\\
&\quad+\varepsilon(\delta,\mu^0_T\sigma_T)\|\bar{x}_0-x^o\|_S\\
&\quad(\text{Convexity of } \hat{g}(x) \text{ and } d_T(u)\geq 0\ \forall\ u\in T)\\
&= -\hat{\varphi}(\bar{u}_0)+\min_{x\in S}\left\{\hat{g}(x)+\langle Bx,\bar{u}_0\rangle+\mu^0_S d_S(x)\right\}+3\delta\\
&\quad+\varepsilon(\delta,\mu^0_T\sigma_T)\|\bar{x}_0-x^o\|_S \qquad (\bar{u}_0:=u^{\delta}_{\mu^0_T,x^o})\\
&\leq \varphi^{\delta}_{\mu^0_T}(\bar{u}_0)+3\delta+\varepsilon(\delta,\mu^0_T\sigma_T)\|\bar{x}_0-x^o\|_S\\
&= \varphi^{\delta}_{\mu^0_T}(\bar{u}_0)+3\delta+\sqrt{\frac{2\delta}{\mu^0_T\sigma_T}}\|B\|_{S,T}\|\bar{x}_0-x^o\|_S\\
&\leq \varphi^{\delta}_{\mu^0_T}(\bar{u}_0)+3\delta+\sqrt{4\delta\|B\|_{S,T}}\left(\frac{D_S D_T}{\sigma_S\sigma_T}\right)^{\frac{1}{4}}\\
&\quad(\mu^0_T:=\|B\|_{S,T}\sqrt{\frac{D_S}{\sigma_S\sigma_T D_T}} \text{ and } \|\bar{x}_0-x^o\|_S\leq\sqrt{\frac{2D_S}{\sigma_S}})
\end{aligned}
$$

For $k\geq 1$, we proceed by induction. To simplify the notation we define the constant

$$\Upsilon=4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S\sigma_T}}.$$

Let us show that Inequality (6.30) holds for $k=1$. Using Theorem 6.13 with $\tau=\tau_0$ and $E(\delta)=E_0(\delta)$, we have

$$
\begin{aligned}
g_{\mu^1_T}(\bar{x}_1) &\leq \varphi_{\mu^1_S}(\bar{u}_1)+\bar{E}(\delta)\\
&= \varphi_{\mu^1_S}(\bar{u}_1)+(2-\tau_0)\left(3\delta+\Upsilon\sqrt{\frac{\delta}{\mu^0_T D_T}}\right)+(1-\tau_0)\left(4\sqrt{\delta\mu^0_S D_S}+E_0(\delta)\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \varphi_{\mu_S^1}(\bar{u}_1) + \frac{4}{3}\left(3\delta + 2\sqrt{\delta\Upsilon}\right) + \frac{1}{3}\left(2\sqrt{2\delta\Upsilon} + E_0(\delta)\right) \\
&\quad (\tau_0 = \frac{2}{3},\ \mu_S^0 D_S = \frac{\Upsilon}{2},\ \mu_T^0 D_T = \frac{\Upsilon}{4}) \\
&= \varphi_{\mu_S^1}(\bar{u}_1) + 4\delta + 2\sqrt{2\delta\Upsilon}\left(\frac{2\sqrt{2}}{3} + \frac{1}{3}\right) + \frac{1}{3}E_0(\delta) \\
&\leq \varphi_{\mu_S^1}(\bar{u}_1) + 6\delta + 2\sqrt{2\delta\Upsilon}(\sqrt{2}+1) + E_0(\delta) \\
&= \varphi_{\mu_S^1}(\bar{u}_1) + 6\delta + 2\sqrt{2\delta}(\sqrt{2}+1)\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S\sigma_T}}\right)^{\frac{1}{2}} + E_0(\delta).
\end{aligned}
$$

Now assume that Inequality (6.30) holds for k and show that it also holds for $k+1$. We distinguish the case where k is even from the case where k is odd.

Suppose k is even. From Theorem 6.13 we have

$$g_{\mu_T^{k+1}}(\bar{x}_{k+1}) \leq \varphi_{\mu_S^{k+1}}(\bar{u}_{k+1}) + E_{k+1}(\delta)$$

with

$$E_{k+1}(\delta) = (2-\tau_k)\left(3\delta + \Upsilon\sqrt{\frac{\delta}{\mu_T^k D_T}}\right) + (1-\tau_k)\left(4\sqrt{\delta\mu_S^k D_S} + E_k(\delta)\right).$$

Let us first evaluate $\mu_T^k D_T$ and $\mu_S^k D_S$. Since $\mu_T^k = \frac{2}{k+2}\mu_T^0$ and $\mu_S^k = \frac{1}{k+1}\mu_S^0$ we have

$$
\begin{aligned}
\mu_T^k D_T &= \frac{2}{k+2}\mu_T^0 D_T = \frac{2}{k+2}\|B\|_{S,T}\sqrt{\frac{D_S}{\sigma_S\sigma_T D_T}}D_T = \frac{1}{k+2}\frac{\Upsilon}{2}, \\
\mu_S^k D_S &= \frac{1}{k+1}\mu_S^0 D_S = \frac{1}{k+1}2\|B\|_{S,T}\sqrt{\frac{D_T}{\sigma_S\sigma_T D_S}}D_S = \frac{1}{k+1}\frac{\Upsilon}{2}.
\end{aligned}
$$

Then,

$$
\begin{aligned}
E_{k+1}(\delta) &= \left(2 - \frac{2}{k+3}\right)\left(3\delta + \Upsilon\sqrt{\frac{2\delta(k+2)}{\Upsilon}}\right) \\
&\quad + \left(1 - \frac{2}{k+3}\right)\left(4\sqrt{\frac{\delta\Upsilon}{2(k+1)}} + E_k(\delta)\right) \\
&= \frac{k+2}{k+3}\left(6\delta + 2\sqrt{2}\sqrt{\delta\Upsilon(k+2)}\right) + \frac{k+1}{k+3}\left(2\sqrt{2}\sqrt{\frac{\delta\Upsilon}{k+1}} + E_k(\delta)\right) \\
&\leq 6\delta + 2\sqrt{2}\sqrt{\delta\Upsilon}\left(\sqrt{(k+1)+1} + \frac{1}{\sqrt{k+1}}\right) + E_k(\delta).
\end{aligned}
$$

Note that if k is odd, $\mu_T^k = \frac{2}{k+1}\mu_T^0$ and $\mu_S^k = \frac{1}{k+2}\mu_S^0$ we have

$$\begin{aligned}\mu_T^k D_T &= \frac{2}{k+1}\mu_T^0 D_T = \frac{2}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S}{\sigma_S\sigma_T D_T}}D_T = \frac{1}{k+1}\frac{\Upsilon}{2},\\ \mu_S^k D_S &= \frac{1}{k+2}\mu_S^0 D_S = \frac{1}{k+2}2\|B\|_{S,T}\sqrt{\frac{D_T}{\sigma_S\sigma_T D_S}}D_S = \frac{1}{k+2}\frac{\Upsilon}{2},\end{aligned}$$

and using Lemma 6.14 we also get

$$E_{k+1}(\delta) \le 6\delta + 2\sqrt{2}\sqrt{\delta\Upsilon}\left(\sqrt{(k+1)+1} + \frac{1}{\sqrt{k+1}}\right) + E_k(\delta).$$

Thus,

$$\begin{aligned}E_{k+1}(\delta) &\le 6\delta + 2\sqrt{2}\sqrt{\delta\Upsilon}\left(\sqrt{(k+1)+1} + \frac{1}{\sqrt{k+1}}\right) + E_k(\delta)\\ &\le 6\delta + 2\sqrt{2}\sqrt{\delta\Upsilon}\left(\sqrt{(k+1)+1} + \frac{1}{\sqrt{k+1}}\right)\\ &\quad + 6\delta k + 2\sqrt{2}\sqrt{\delta}\sum_{i=1}^{k}(\sqrt{i+1} + \frac{1}{\sqrt{i}})\sqrt{\Upsilon} + E_0(\delta)\\ &= 6\delta(k+1) + 2\sqrt{2}\sqrt{\delta}\sum_{i=1}^{k+1}(\sqrt{i+1} + \frac{1}{\sqrt{i}})\sqrt{\Upsilon} + E_0(\delta)\\ &= 6\delta(k+1) + 2\sqrt{2}\sqrt{\delta}\sum_{i=1}^{k+1}(\sqrt{i+1} + \frac{1}{\sqrt{i}})\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S\sigma_T}}\right)^{\frac{1}{2}} + E_0(\delta)\end{aligned}$$

□

Having derived the error propagation, we can finally state the accuracy δ of the approximate oracles, which is sufficient for obtaining the same rate of convergence as if exact oracles were available.

Theorem 6.16 (Convergence of Excessive Gap Algorithm with Approximate Oracles— Theorem 7 in [CE05])
Let the pairs of sequences $(\{\bar{x}_k\}_{k\ge0}, \{\mu_T^k\}_{k\ge0})$ *and* $(\{\bar{u}_k\}_{k\ge1}, \{\mu_S^k\}_{k\ge1})$ *be generated by Algorithm 6. Then, for each* $k \ge 1$

$$g(\bar{x}_k) - \varphi(\bar{u}_k) \le \frac{24}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S\sigma_T}} \tag{6.31}$$

holds provided that $\delta = \frac{1}{(k+1)^5}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S\sigma_T}}$.

Proof. Using Theorem 6.15 we get for $k \geq 1$

$$
\begin{aligned}
0 &\leq g(\bar{x}_k) - \varphi(\bar{u}_k) \\
&\quad \text{(Definition and properties of } g_{\mu_T^k} \text{ and } \varphi_{\mu_S^k}\text{)} \\
&\leq g_{\mu_T^k}(\bar{x}_k) - \varphi_{\mu_S^k}(\bar{u}_k) + \mu_S^k D_S + \mu_T^k D_T \\
&\quad \text{(Excessive Gap condition, Theorem 6.15)} \\
&\leq E_k(\delta) + \mu_S^k D_S + \mu_T^k D_T \\
&\leq 6\delta k + 2\sqrt{2}\sqrt{\delta}\sum_{i=1}^{k}(\sqrt{i+1} + \frac{1}{\sqrt{i}})\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right)^{\frac{1}{2}} + E_0(\delta) \\
&\quad + \frac{4}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}} \\
&\leq 6\delta(k+1) + \sqrt{\delta}\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right)^{\frac{1}{2}}\left(2\sqrt{2}\sum_{i=1}^{k}(\sqrt{i+1} + \frac{1}{\sqrt{i}}) + 1\right) \\
&\quad + \frac{4}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}} \\
&\leq 6\delta(k+1) + \sqrt{\delta}\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right)^{\frac{1}{2}} \cdot \\
&\quad \left(2\sqrt{2}\left(\int_2^{k+2}\sqrt{\omega}d\omega + 1 + \int_1^k \frac{1}{\sqrt{\omega}}d\omega\right) + 1\right) + \frac{4}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}} \\
&\leq 6\delta(k+1) + 4\sqrt{2}\sqrt{\delta}\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right)^{\frac{1}{2}}\left(\frac{(k+2)^{\frac{3}{2}}}{3} + k^{\frac{1}{2}}\right) \\
&\quad + \frac{4}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}} \\
&\leq 6\delta(k+1) + 4\sqrt{2}\sqrt{\delta}\left(4\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right)^{\frac{1}{2}}\left((k+1)^{\frac{3}{2}} + k^{\frac{1}{2}}\right) \\
&\quad + \frac{4}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}} \\
&\leq \left(\frac{3}{2}\frac{1}{(k+1)^3} + \frac{2\sqrt{2}}{(k+1)} + 2\sqrt{2} + 1\right)\frac{4}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}} \\
&\quad \left(\delta = \frac{1}{(k+1)^5}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}\right) \\
&\leq \frac{24}{k+1}\|B\|_{S,T}\sqrt{\frac{D_S D_T}{\sigma_S \sigma_T}}
\end{aligned}
$$

Last inequality holds since for $k \geq 1$ we have $\left(\frac{3}{2}\frac{1}{(k+1)^3} + \frac{2\sqrt{2}}{(k+1)} + 2\sqrt{2} + 1\right) \leq 6$. $\square$

Thus, in order to obtain an error of $\epsilon > 0$, δ has to be of order $O(\epsilon^5)$. This suggests that the algorithm could be instable under minor inaccuracies of the oracles. However, the numerical results in Section 9.3 do not support this concern. Note that we have not made any assumptions on the oracles except their accuracy. The last result was presented in Theorem 7 in [CE05]. Note that due to a calculation error we supposed that a δ of order $O(\epsilon^4)$ was sufficient. Nevertheless, such an order of precision is prohibitive and implies that subproblems must be solved to optimality. If possible, Primal-Dual Interior Point methods should be applied given their running time dependence on δ, $O(\log \frac{1}{\delta})$. It may be possible to improve the theoretical result by imposing other requirements on the approximate oracles.

Part II

Applications of the Optimization Methods to Special Linear Problems

7. Uncapacitated Facility Location Problem

In this chapter, we present polynomial time approximation schemes for the linear programming relaxation of the Uncapacitated Facility Location problem (UFL) basing either on the Primal-Dual Subgradient methods (see Chapter 4) or the Excessive Gap method (see Chapter 5). These schemes are analyzed not only from a theoretical point of view but also a practical. Theoretical and numerical results were partially presented in [CE05]. In this paper we focus on the presentation of a polynomial time approximation scheme based on the Excessive Gap method and using the Euclidean norm and the Squared Euclidean norm as prox-functions. We obtained the following result, for $\epsilon > 0$ a polynomial time approximation scheme exists, which delivers a $(1+\epsilon)$-approximation in $O(\frac{1}{\epsilon})$. It improves the dependence on ϵ of the previous best algorithms we were aware of by a factor of $O(\frac{1}{\epsilon})$, [You00], [GK02], and [Chu03]. Here, we extend the presentation to the polynomial time approximation scheme based on Primal-Dual Subgradient methods and we also consider other choices of norms and prox-functions. Note that we focus on the presentation of numerical performance of the developed algorithms. Related results concerning the Facility Location problem with submodular penalties were presented by Chudak and Nagano in [CN07].

7.1 Problem Description

The Uncapacitated Facility Location problem (UFL) consists of a set of potential facility locations $\mathcal{F}$ and a set of clients $\mathcal{D}$ that require service. Building a facility at location $i \in \mathcal{F}$ has an associated fixed cost $f_i \geq 0$, and any open facility can provide an unlimited amount of a certain commodity. Each client $j \in \mathcal{D}$ has a demand $d_j > 0$ that must be shipped from one of the open facilities. If a facility at location $i \in \mathcal{F}$ is used to satisfy the demand of the client $j \in \mathcal{D}$, the service cost incurred is proportional to the distance c_{ij} from i to j. The goal is to determine a subset of the set of potential facility locations at which to open facilities and an assignment of clients to these facilities so as to minimize the overall total cost; that is, the fixed costs of opening the facilities plus the total service cost.

We assume that all the demands are 1 (otherwise we replace c_{ij} with $d_j c_{ij}$) and throughout we let $m := |\mathcal{F}|$ and $n := |\mathcal{D}|$. Of course the UFL problem corresponds to an integer minimization problem but here we consider the simplest linear programming relaxation introduced by Balinski in [Bal65]), which we will refer to as UFL-LP,

$$\begin{aligned}
\text{(UFL-LP)} \quad \text{minimize} \quad & \sum_{i\in\mathcal{F}}\sum_{j\in\mathcal{D}} c_{ij}x_{ij} + \sum_{i\in\mathcal{F}} f_i y_i & & \\
\text{subject to} \quad & \sum_{i\in\mathcal{F}} x_{ij} = 1 & \forall j \in \mathcal{D} & \quad (7.1)\\
& x_{ij} \le y_i & \forall j \in \mathcal{D}, i \in \mathcal{F} & \quad (7.2)\\
& x_{ij} \ge 0. & &
\end{aligned}$$

Any 0-1 feasible solution corresponds to a feasible solution to the UFL problem. Namely, $y_i = 1$ indicates that a facility at location $i \in \mathcal{F}$ is open, whereas $x_{ij} = 1$ means that client $j \in \mathcal{D}$ is serviced by the facility built at location $i \in \mathcal{F}$. The inequalities (7.1) state that each demand point $j \in \mathcal{D}$ must be assigned to some facility, whereas inequalities (7.2) say that clients can only be assigned to open facilities. Thus the linear program UFL-LP is indeed a relaxation of the problem. The linear programming relaxation, UFL-LP, is known to provide excellent lower bounds, and sometimes is referred to as the "strong linear programming relaxation". Namely, it was often observed that optimal solutions of the UFL-LP are integral, see [MF90] and references thereafter. Moreover, Chudak shows in [Chu98] that for the metric case (service costs are proportional to distance), the UFL-LP optimal solution is within a factor of 1.736 of the optimal cost of the UFL problem. Baïou and Barahona recently show in [BB06] that for special instances of the UFL problem, the convex hull of the integer feasible solutions is equal to the polytope of feasible solutions of the UFL-LP.

In order to solve the UFL-LP, Lagrange relaxations have been considered because of its large size ($O(mn)$ variables and constraints). In practice, the algorithms that are known to obtain the best results are those that relax equations (7.1). However these algorithms usually have problems delivering feasible primal solutions (see [BC99]) and do not have a known worst-case analysis of the running time. In contrast, from a theoretical point of view, relaxing constraints (7.2) has proven more useful. In this category, there are the algorithms of [You00], [GK02] and [CS03]. These algorithms are all polynomial time approximation schemes providing in $O(\frac{1}{\epsilon^2})$ iterations an ϵ-relative approximate solution, which corresponds to the previously best known running time dependence on ϵ. We follow this approach as well and derive polynomial time approximation schemes first with a running time dependence $O(\frac{1}{\epsilon^2})$ and secondly with a running time dependence $O(\frac{1}{\epsilon})$, our improvement contribution.

Consider the Lagrange relaxation of inequalities (7.2),

$$\begin{array}{lll} \max\limits_{w} \quad \min\limits_{x,y} & \displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{i=1}^{m} f_i y_i + \sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}(x_{ij} - y_i) & \\ \text{subject to} & x \in \Delta_m^n & \\ & w_{ij} \geq 0 & \forall j \in \mathcal{D}, i \in \mathcal{F} \end{array}$$

where Δ_m^n denotes the Cartesian product of n m-dimensional simplex, i.e., $\Delta_m := \{z \in \mathbb{R}^m \mid \sum_{i=1}^m z_i = 1, z_i \geq 0 \ \forall \ i = 1, \ldots, m\}$ and $\Delta_m^n := \Delta_m \times \cdots \times \Delta_m$ (the notation will be used throughout this chapter). Since we do not require the variable y to be nonnegative, we can assume without loss of generality that for each $i = 1, \ldots, m$, $f_i = \sum_{j=1}^n w_{ij}$. Then, using the change of variables $w_{ij} = f_i u_{ij}$ for $i = 1, \cdots, m$ and $j = 1, \cdots, n$, with $u \in \Delta_n^m$, we get the following representation of the previous Lagrange relaxation,

$$\text{(UFL-LR)} \qquad \max_{u \in \Delta_n^m} \quad \min_{x \in \Delta_m^n} \left\{ \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} f_i u_{ij} x_{ij} \right\}.$$

For the sake of clarity, we will use the notations introduced in Chapter 3. Thus, the primal space Q is Δ_m^n and the dual space P is Δ_n^m. The primal function is

$$f(x) := \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \max_{u \in P} \left\{ \sum_{i=1}^{m}\sum_{j=1}^{n} f_i u_{ij} x_{ij} \right\} \tag{7.3}$$

and the dual function is

$$\psi(u) := \min_{x \in Q} \left\{ \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} f_i u_{ij} x_{ij} \right\}. \tag{7.4}$$

Thus, the sets Q and P with the objective functions $f(x)$ and $\psi(u)$ satisfy Assumption 1.

Note that both functions have the property of being separable either by clients or by facilities. Namely we can rewrite the functions as follows,

$$f(x) := \sum_{i=1}^{m} \left(\sum_{j=1}^{n} c_{ij}x_{ij} + \max_{u_i \in \Delta_n} \sum_{j=1}^{n} f_i u_{ij} x_{ij} \right) \tag{7.5}$$

and

$$\psi(u) := \sum_{j=1}^{n} \left(\min_{x_j \in \Delta_m} \sum_{i=1}^{m} c_{ij}x_{ij} + f_i u_{ij} x_{ij} \right). \tag{7.6}$$

The evaluation of both functions is hence simplified, since we have to compute a sequence of smaller maximization or minimization problems. These optimization problems correspond to minimization or maximization of linear functions over Δ_m, respectively Δ_n, which can be done in $O(m)$, respectively $O(n)$. Hence, the overall complexity of the evaluation of the primal or the objective functions is $O(nm)$

Before considering in detail the developed approximation algorithm for solving UFL-LR and its dual, we show that it is simple to derive a feasible solution, (x, y), for the UFL-LP from a feasible solution $\bar{x} \in \Delta_m^n$, such that the corresponding costs are equal to $f(\bar{x})$. Namely, for each facility $i \in \mathcal{F}$ we define $y_i := \max_{j \in \mathcal{D}} \bar{x}_{ij}$ and for all $i \in \mathcal{F}$ and $j \in \mathcal{D}$ we define $x_{ij} := \bar{x}_{ij}$. Then,

$$\begin{aligned} f(\bar{x}) &= \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}\bar{x}_{ij} + \sum_{i=1}^{m} \max_{u_i \in \Delta_n} \sum_{j=1}^{n} f_i u_{ij} \bar{x}_{ij} \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}\bar{x}_{ij} + \sum_{i=1}^{m} f_i \max_{j \in \mathcal{D}} \bar{x}_{ij} \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{i=1}^{m} f_i y_i. \end{aligned}$$

Similarly, we can derive from a feasible solution $\bar{u} \in \Delta_n^m$ a feasible solution for the dual problem of UFL-LP. In the following consider the dual of UFL-LP,

$$\begin{aligned} \text{(UFL-DP)} \quad \text{maximize} \quad & \sum_{j \in \mathcal{D}} v_j \\ \text{subject to} \quad & \sum_{j \in \mathcal{D}} w_{ij} = f_i \qquad \forall i \in \mathcal{F} \\ & v_j - w_{ij} \leq c_{ij} \qquad \forall j \in \mathcal{D}, i \in \mathcal{F} \\ & w_{ij} \geq 0. \end{aligned}$$

For all $i \in \mathcal{F}$ and $j \in \mathcal{D}$ we define $w_{ij} := f_i \bar{u}_{ij}$ and for all $j \in D$ we define $v_j := \min_{i \in \mathcal{F}}\{c_{ij} + f_i \bar{u}_{ij}\}$. Then,

$$\begin{aligned} \psi(\bar{u}) &= \sum_{j=1}^{n} \min_{x_j \in \Delta_m} \sum_{i=1}^{m} (c_{ij} + f_i \bar{u}_{ij})\, x_{ij} \\ &= \sum_{j=1}^{n} \min_{i \in \mathcal{F}} \; (c_{ij} + f_i \bar{u}_{ij}) \\ &= \sum_{j=1}^{n} v_j. \end{aligned}$$

7.2 Optimization Methods

For applying both methods, Primal-Dual Subgradient method and Excessive Gap method, the first step consists of choosing space norms and then the appropriate prox-functions (differentiable and strongly convex functions). This decision has considerable impact on the tractability, since it generates different algorithms and implies different oracles.

We are going to consider two options. First we use the Euclidean norm for the space norms and the squared Euclidean norm for the prox-functions. Secondly, we consider the Norm 1 and the Entropy function for the prox-functions. Both choices generate approximation functions, which are either separable by client or by facility.

First, let us investigate the subdifferential of the primal function, which does not depend on the chosen norm.

For fixed $x \in Q$, define

$$U_i(x) := \left\{ u_i \in \Delta_n \mid u_i = \arg \max_{v_i \in \Delta_n} \{ \sum_{j=1}^{n} f_i x_{ij} v_{ij} \} \right\} \quad \forall\ i = 1, \ldots, m$$

and $U(x) := U_1(x) \times \cdots \times U_m(x)$. Moreover, define for $u \in U(x)$,

$$\xi_{ij} := c_{ij} + f_i u_{ij} \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n, \tag{7.7}$$

then $\xi \in \partial f(x)$ using Lemma 4.9. Thus, computing a subgradient at x is equivalent to solving

$$\max_{v_i \in \Delta_n} \sum_{j=1}^{n} f_i x_{ij} v_{ij} \qquad \forall\ i = 1, \ldots, m$$

i.e., it is sufficient to evaluate $\max_{\{1 \le j \le n\}} x_{ij}$ for $i = 1, \ldots, m$, which can be done in $O(nm)$, see Algorithm 7.

Algorithm 7 UFL:Subgradient

Input: An evaluation point x
Output: $\xi_x \in \partial f(x)$ and the corresponding feasible dual solution u

for $i = 1$ to m **do**
compute $j^* = \arg \max_{1 \le j \le n} x_{ij}$
set $u_{ij} = \begin{cases} 1 & \text{if } j = j^* \\ 0 & \text{otherwise} \end{cases}$ and $\xi_{ij} = \begin{cases} c_{ij} + f_i u_{ij} & \text{if } j = j^* \\ c_{ij} & \text{otherwise} \end{cases}$
end for

Note that when we compute a subgradient ξ, we also compute a corresponding dual solution u.

The next subsections are divided into two parts; one part concerning the Primal-Dual Subgradients methods and the other part concerning the Excessive Gap method.

7.2.1 Euclidean Norm

We endow the primal space as well as the dual space with Euclidean norms. As prox-functions we consider the squared Euclidean norm. Thus, the primal prox-function is defined as follows

$$\begin{aligned} d_Q(x) &:= \frac{1}{2}\|x - x^o\|^2, \qquad (7.8)\\ \text{where} \quad & x^o := \frac{1}{m}\mathbf{1} = \arg\min_{x\in Q} \|x\|_2 \\ & \sigma_Q = 1 \text{ and } D_Q = \frac{n}{2}\frac{(m-1)}{m} \geq \max_{x\in Q} d_Q(x), \qquad (7.9) \end{aligned}$$

and similarly, the dual prox-function is defined as

$$\begin{aligned} d_P(u) &:= \frac{1}{2}\|u - u^o\|^2, \qquad (7.10)\\ \text{where} \quad & u^o := \frac{1}{n}\mathbf{1} = \arg\min_{u\in P} \|u\|_2 \\ & \sigma_P = 1 \text{ and } D_P = \frac{m}{2}\frac{(n-1)}{n} \geq \max_{u\in P} d_P(u), \qquad (7.11) \end{aligned}$$

where we denote the unit vector by $\mathbf{1}$.

The minimum x^o, respectively u^o, corresponds to the Euclidean projection of $0 \in \mathbb{R}^{mn}$ onto the product of simplices Δ_m^n, respectively Δ_n^m. The maximal value of both prox-functions, D_Q and D_P, are attained at the extreme points of the product of simplices, Δ_m^n and Δ_n^m.

Finally the norm of matrix A is given by

$$\begin{aligned} \|A\|_{Q,P} &:= \max_{\|x\|_2\leq 1}\max_{\|u\|_2\leq 1}\langle Ax, u\rangle \\ &= \max_{\|x\|_2\leq 1}\max_{\|u\|_2\leq 1}\sum_{i=1}^{m}\sum_{j=1}^{n} f_i u_{ij} x_{ij} \\ &\leq \max_{\|x\|_2\leq 1}\max_{\|u\|_2\leq 1}(\max_{1\leq i\leq m} f_i)\|x\|_2\|u\|_2 \\ &\leq (\max_{1\leq i\leq m} f_i) =: f_{\max}. \qquad (7.12) \end{aligned}$$

Primal Dual Subgradient Method

In the different subgradient methods with the exception of the Truncated Dual Averaging method, we assume that the norm of the computed subgradients is bounded (this bound was denoted by L). Since the adjoint norm of the Euclidean norm is the Euclidean norm itself, the norm of the subgradients is also Euclidean. Let x be fixed and consider $\xi_x \in \partial f(x)$ defined as in Equation (7.7) with $u \in U(x)$, then

$$\begin{aligned} \|\xi_x\|^* &= \|\xi_x\|_2 = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (c_{ij} + f_i u_{ij})^2 \right)^{1/2} \\ &\leq \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (c_{ij} + f_i)^2 \right)^{1/2} = \|c + \tilde{f}\|_2 =: L \end{aligned}$$

where $u \in U(x)$, $\tilde{f} \in \mathbb{R}^{nm}$ with $\tilde{f}_{i1} = \cdots = \tilde{f}_{in} = f_i$ for all $1 \leq i \leq m$.

In the Truncated Dual Averaging method, we assume that the subgradients of the primal function $f(x)$ have bounded variations. We next evaluate an upper bound for those variations, which we denoted in Section 4.3 by M. Let x and y be fixed and consider $\xi_x \in \partial f(x)$ and $\xi_y \in \partial f(y)$ defined as in Equation (7.7), then

$$\begin{aligned} \|\xi_x - \xi_y\|_2 &= \left(\sum_{i=1}^{m} \sum_{j=1}^{n} (c_{ij} + f_i u_{ij}^x - c_{ij} - f_i u_{ij}^y)^2 \right)^{1/2} \text{ with } u^x \in U(x),\ u^y \in U(y) \\ &= \left(\sum_{i=1}^{m} \sum_{j=1}^{n} f_i^2 (u_{ij}^x - u_{ij}^y)^2 \right)^{1/2} \leq \left(\sqrt{2} \|\bar{f}\|_2 \right) =: M \end{aligned}$$

where $\bar{f} = (f_i, \cdots, f_m)$.

Now we have all the elements necessary to apply the Primal-Dual Subgradients algorithms to the linear programming relaxation of the UFL problem. Algorithm 8 corresponds to Algorithm 2 using the simple averages sequences (4.12), Algorithm 9 corresponds to Algorithm 2 using the weighted averages sequences (4.13), and Algorithm 10 corresponds to Algorithm 3.

Algorithm 8 Simple Dual Averaging Algorithm for UFL (UFL-SDA-Euclidean)

Input: - $x^o = \frac{1}{m}\mathbf{1}$
- The constants D_Q and L
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 0$ and $\hat{\beta}_0 = 1$
set $x_0 = x^o$
compute $\xi_0 \in \partial f(x_0)$ and u_0 `UFL:Subgradient`
set $\zeta_0 = \xi_0$
set $\bar{x} = x_0$ and $\bar{u} = u_0$.
while $f(\bar{x}) - \psi(\bar{u}) > \epsilon$ **do**
  set $k = k + 1$
  compute $\hat{\beta}_k = \hat{\beta}_{k-1} + \frac{1}{\hat{\beta}_{k-1}}$ and $\beta_k = \frac{L}{\sqrt{2D_Q}}\hat{\beta}_k$
  compute $x_k = \arg\min_{x \in Q} \{\langle \zeta_{k-1}, x \rangle + \beta_k \frac{1}{2}\|x - x^o\|_2^2\}$
  `UFL:Quadratic projection onto` Δ_m^n
  compute $\xi_k \in \partial f(x_k)$ and u_k `UFL:Subgradient`
  set $\zeta_k = \zeta_{k-1} + \xi_k$
  $\bar{x} = \frac{1}{k+1}\sum_{l=0}^{k} x_l$ and $\bar{u} = \frac{1}{k+1}\sum_{l=0}^{k} u_l$ `UFL:Objective functions evaluation`
end while

Algorithm 9 Weighted Dual Averaging Algorithm for UFL (UFL-WDA-Euclidean)

Input: - $x^o = \frac{1}{m}\mathbf{1}$
- The constant D_Q
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 0$, and $\hat{\beta}_0 = 1$
set $x_0 = x^o$
compute $\xi_0 \in \partial f(x_0)$ and u_0 `UFL:Subgradient`
set $\lambda_0 = \frac{1}{\|\xi_0\|_2}$, $\Lambda_0 = \lambda_0$, and $\zeta_0 = \lambda_0 \xi_0$
set $\bar{x} = x_0$ and $\bar{u} = u_0$.
while $f(\bar{x}) - \psi(\bar{u}) > \epsilon$ **do**
  set $k = k + 1$
  compute $\hat{\beta}_k = \hat{\beta}_{k-1} + \frac{1}{\hat{\beta}_{k-1}}$ and $\beta_k = \frac{1}{\sqrt{2D_Q}}\hat{\beta}_k$
  compute $x_k = \arg\min_{x \in Q} \{\langle \zeta_{k-1}, x \rangle + \beta_k \frac{1}{2}\|x - x^o\|_2^2\}$
  `UFL:Quadratic projection onto` Δ_m^n
  compute $\xi_k \in \partial f(x_k)$ and u_k `UFL:Subgradient`
  set $\lambda_k = \frac{1}{\|\xi_k\|_2}$, $\Lambda_k = \Lambda_{k-1} + \lambda_k$, and $\zeta_k = \zeta_{k-1} + \lambda_k \xi_k$
  set $\bar{x} = \frac{1}{\Lambda_k}\sum_{l=0}^{k} \lambda_l x_l$ and $\bar{u} = \frac{1}{\Lambda_k}\sum_{l=0}^{k} \lambda_l u_l$
  `UFL:Objective functions evaluation`
end while

Algorithm 10 Truncated Simple Dual Averaging Algorithm for UFL (UFL-TSDA-Euclidean)

Input: - $x^o := \frac{1}{m}\mathbf{1}$
- The constants D_Q and M
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 1$ and $\beta_1 = 1$
set $x_0 = x^o$
compute $\xi_0 \in \partial f(x_0)$ and u_0 `UFL:Subgradient`
compute $x_1 = \arg\min_{x \in Q} \{\langle \xi_0, x \rangle + \beta_1 \frac{1}{2}\|x - x^o\|_2^2\}$
`UFL:Quadratic projection onto` Δ_m^n
compute $\xi_1 \in \partial f(x_1)$ and u_1 `UFL:Subgradient`
set $\zeta_1 = \xi_1$
$\bar{x} = x_1$ and $\bar{u} = u_1$ `UFL:Objective functions evaluation`

while $f(\bar{x}) - \psi(\bar{u}) > \epsilon$ **do**
  set $k = k + 1$
  compute $x_k = \arg\min_{x \in Q} \{\langle \zeta_{k-1}, x \rangle + \beta_k \frac{1}{2}\|x - x^o\|_2^2\}$
  `UFL:Quadratic projection onto` Δ_m^n
  set $\beta_k = \frac{M}{\sqrt{2D_Q}}\sqrt{k}$
  compute $\xi_k \in \partial f(x_k)$ and u_k `UFL:Subgradient`
  set $\zeta_k = \zeta_{k-1} + \xi_k$
  $\bar{x} = \frac{1}{k}\sum_{l=1}^{k} x_l$ and $\bar{u} = \frac{1}{k}\sum_{l=1}^{k} u_l$ `UFL:Objective functions evaluation`
end while

In order to evaluate the running time of all algorithms, we need to study the complexity of the three operations below, which are the most demanding operations in the algorithms, i.e.,

1) the evaluation of the primal and the dual objective functions, $f(x), x \in \Delta_m^n$ and $\psi(u), u \in \Delta_n^m$

2) the computation of the subgradients of the primal objective function, $\xi_x \in \partial f(x), x \in \Delta_m^n$.

3) the computation of the quadratic projections onto Δ_m^n.

We note that these operations can be efficiently conducted. Namely, as explained at the end of the previous subsection, it takes $O(nm)$ time to evaluate both objective functions. Moreover, using Algorithm 7, a subgradient of the primal objective function as well as a corresponding feasible dual solution can be computed in $O(nm)$ time. The oracle providing quadratic projection onto Δ_m^n remains to be determined.

Towards that end, we consider a general formulation of the quadratic projections onto Δ_m^n,

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij}y_{ij} + \frac{1}{2}b\sum_{j=1}^{n}\sum_{i=1}^{m} y_{ij}^2 \\ \text{subject to} \quad & y \in \Delta_m^n \end{aligned} \tag{7.13}$$

where $a \in \mathbb{R}^{nm}$ and $b \in \mathbb{R}_+$. We observe that this problem is decomposable. Thus, it is sufficient to solve the following n minimization problems,

$$y_j := \arg\min_{x \in \Delta_m} \left\{ \sum_{i=1}^{m} a_{ij}x_i + \frac{1}{2}b\sum_{i=1}^{m} x_i^2 \right\} \quad \forall j = 1, \ldots, n \tag{7.14}$$

and $y = (y_1, \ldots, y_n)$.

From the KKT conditions of a single optimization problem (7.14), we derive the procedure described in Algorithm 11 for computing the general quadratic projection onto Δ_m^n (7.13).

Lemma 7.1
Algorithm 11 delivers in $O(nm \log m)$ time an optimal solution to the following problem

$$\begin{aligned} \textit{minimize} \quad & \sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij}y_{ij} + \frac{1}{2}b\sum_{j=1}^{n}\sum_{i=1}^{m} y_{ij}^2 \\ \textit{subject to} \quad & y \in \Delta_m^n \end{aligned}$$

where $a \in \mathbb{R}^{nm}$ and $b \in \mathbb{R}_+$.

Proof. To show that Algorithm 11 delivers an optimal solution we show that output y and (λ, w) satisfy the KKT conditions for the considered optimization problem. Note that we assume the components of vector a to be ordered, i.e., $a_{1j} \leq a_{2j} \leq \cdots \leq a_{mj}$ for $j = 1, \ldots, n$.

Thus, we show that

1) $a_{ij} + by_{ij} + \lambda_j - w_{ij} = 0$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$

2) $y_{ij} w_{ij} = 0$ for $i = 1, \ldots, m$ and for $j = 1, \ldots, n$ *(orthogonality)*

3) $\sum_{i=1}^{m} y_{ij} = 1$ for $j = 1, \ldots, n$ and $y_{ij} \geq 0$ for $i = 1, \ldots, m$ and for $j = 1, \ldots, n$ *(primal feasibility)*

4) $\lambda \in \mathbb{R}^m$ and $w_{ij} \geq 0$ for $i = 1, \ldots, m$ and for $j = 1, \ldots, n$ *(dual feasibility)*

hold.

Let us fix j and consider $k \geq 1$ such that $y_{kj} = -\frac{1}{b}(\lambda_j + a_{kj}) \geq 0$ with $\lambda_j = -\frac{1}{k}(b + \sum_{i=1}^{k} a_{ij})$.

For $i = 1, \ldots, k$, we have $w_{ij} = 0$ and $y_{ij} = -\frac{1}{b}(\lambda_j + a_{ij}) \geq -\frac{1}{b}(\lambda_j + a_{kj}) \geq 0$ since $a_{ij} \leq a_{kj}$. Thus, the KKT conditions 1), 2), and 4) are satisfied for $i = 1, \ldots, k$ and fixed j.

For proving that they are also satisfied for $i = k+1, \ldots, m$ and fixed j let us first note that for $k+1$,

$$-\frac{1}{b}\left(-\frac{1}{k+1}(b + \sum_{i=1}^{k+1} a_{ij}) + a_{k+1j}\right) < 0$$

and thus $-\frac{1}{k}(b + \sum_{i=1}^{k} a_{ij}) + a_{k+1j} > 0$. Then, for $i = k+1, \ldots, m$,

$$\begin{aligned} y_{ij} &= 0 \\ w_{ij} &= -\frac{1}{k}(b + \sum_{i=1}^{k} a_{ij}) + a_{ij} \geq -\frac{1}{k}(b + \sum_{i=1}^{k} a_{ij}) + a_{k+1j} > 0. \end{aligned}$$

Finally,

$$\sum_{i=1}^{m} y_{ij} = \sum_{i=1}^{k} -\frac{1}{b}(\lambda_j + a_{ij}) = \sum_{i=1}^{k} -\frac{1}{b}(-\frac{1}{k}(b + \sum_{l=1}^{k} a_{lj}) + a_{ij}) = 1 + \sum_{l=1}^{k} \frac{a_{lj}}{b} - \sum_{i=1}^{k} \frac{a_{ij}}{b} = 1.$$

Thus, y and (λ, w) satisfy the KKT conditions for the considered optimization problem.

For fixed j the running time is $O(m \log m + m)$ where the first term is due to the sorting of the components of a, $a_{1j}, \ldots, a_{mj}$. Thus the total complexity of the procedure is $O(nm \log m)$. □

As mentioned in [CE05] and pointed out by one of the referees of the paper, the sorting can be avoided using a variant of the weighted median algorithm, see [KV00], and thus providing a linear time algorithm for solving the quadratic projection onto Δ_m^n. Nevertheless, the numerical results presented in the last section of this chapter concern the algorithms using the non linear version of the quadratic projections onto Δ_m^n, Algorithm 11, which is very simple to implement.

Algorithm 11 UFL:Quadratic projection onto Δ_m^n

Input: - A vector $a \in \mathbb{R}^{nm}$ and a constant $b \in \mathbb{R}_+$
Output: - The optimal solution $y \in \Delta_m^n$ of the quadratic optimization problem described in (7.13).
- A Lagrange dual solution $(\lambda, w) \in \mathbb{R}^{n+nm}$.

Order a such that $a_{1j} \leq a_{2j} \leq \cdots \leq a_{mj}$ for $j = 1, \ldots, n$.
for $j = 1$ to n **do**
  set $\lambda_j = -\frac{1}{m}(b + \sum_{i=1}^{m} a_{ij})$ and $k = m$
  set $y_{kj} = -\frac{1}{b}(\lambda_j + a_{kj})$
  while $y_{kj} < 0$ and $k > 1$ **do**
    $k = k - 1$
    set $\lambda_j = -\frac{1}{k}(b + \sum_{i=1}^{k} a_{ij})$
    set $y_{kj} = -\frac{1}{b}(\lambda_j + a_{kj})$
  end while
  for $i = 1$ to k **do**
    set $y_{ij} = -\frac{\lambda_j + a_{ij}}{b}$ and $w_{ij} = 0$
  end for
  for $i = k + 1$ to m **do**
    set $y_{ij} = 0$ and $w_{ij} = \lambda_j + a_{ij}$
  end for
end for

Now, we have all the required information for the investigation of the running time of the UFL-SDA-Euclidean Algorithm (8), the UFL-WDA-Euclidean Algorithm (9) and the UFL-TSDA-Euclidean Algorithm (10).

Theorem 7.2
Given $\epsilon > 0$, the UFL-SDA-Euclidean Algorithm (8) and the UFL-WDA Algorithm (9) output in $O\left(\frac{1}{\epsilon^2}\|c + \tilde{f}\|_2^2 n^2 m \log m\right)$ time a primal and a dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

Proof. Using the theoretical result presented in Theorem 4.7, we know that if we run either the Simple Dual Averaging Algorithm or the Weighted Dual Averaging Algorithm for $(\frac{9L^2}{\epsilon^2}\frac{D_Q}{\sigma_Q} - 1)$ iterations, we obtain a primal and dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$, $\epsilon > 0$. Thus we need to run

$$\frac{9\|c + \tilde{f}\|_2^2}{\epsilon^2}\frac{n}{2}\frac{(m-1)}{m} \leq \frac{9}{2}\frac{\|c + \tilde{f}\|_2^2 n}{\epsilon^2}$$

of the UFL-SDA-Euclidean Algorithm or the UFL-WDA-Euclidean Algorithm to have a primal and a dual solution with an additive gap of ϵ.

At each iteration we need to compute a subgradient, which requires $O(nm)$ time, a quadratic projection onto the Δ_m^n, which requires $O(nm \log m)$ time, and to evaluate both primal and dual objective functions, which requires $O(nm)$ time. Thus, each iteration takes $O(nm \log m)$ time and thus, the running time of the considered algorithms is

$$O\left(\frac{1}{\epsilon^2}\|c + \tilde{f}\|_2^2 n^2 m \log m\right).$$

□

Theorem 7.3
Given $\epsilon > 0$, the UFL-TSDA-Euclidean Algorithm (10) outputs in $O\left(\frac{1}{\epsilon^2}n^2 m \log m \|\bar{f}\|_2^2\right)$ time a primal and a dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

Proof. Using the theoretical result presented in 4.8, we know that if we run $\frac{8M^2}{\epsilon^2}\frac{D_Q}{\sigma_Q}$ iterations of the Truncated Simple Dual Averaging Algorithm, we obtain a primal and dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$, $\epsilon > 0$. Thus, we need to run UFL-TSDA-Euclidean Algorithm for

$$\frac{16\|\bar{f}\|_2^2}{\epsilon^2}\frac{n}{2}\frac{(m-1)}{m} \leq \frac{8\|\bar{f}\|_2^2}{\epsilon^2}n$$

iterations to have a primal and a dual solution with an additive gap of ϵ.

As for the UFL-SDA-Euclidean Algorithm and the UFL-WDA-Euclidean Algorithm, each iteration takes $O(nm \log m)$. Finally, the running time of the UFL-TSDA Algorithm is

$$O\left(\frac{1}{\epsilon^2}n^2 m \log m \|\bar{f}\|_2^2\right).$$

□

We note that UFL-TSDA-Euclidean Algorithm outperforms the other two algorithms. In the last section of this chapter we confirm this observation with numerical results.

Excessive Gap Method

The first step in the elaboration of the Excessive Gap method is the creation of smooth approximation functions for both primal and dual objective functions. This corresponds to the choice of the prox-function d_Q and the prox-function d_P, defined in (7.8) and (7.10).

Using the same notation as in Section 5.2 we define the following smooth approximations:

$$f_{\mu_P}(\tilde{x}) := \sum_{i=1}^{m}\left(\sum_{j=1}^{n} c_{ij}\tilde{x}_{ij} + \max_{u_i\in\Delta_n}\left\{\sum_{j=1}^{n} f_i u_{ij}\tilde{x}_{ij} - \frac{1}{2}\mu_P\sum_{j=1}^{n}(u_{ij}-u_{ij}^o)^2\right\}\right) \tag{7.15}$$

$$\psi_{\mu_Q}(\tilde{u}) := \sum_{j=1}^{n}\left(\min_{x_j\in\Delta_m}\left\{\sum_{i=1}^{m}(c_{ij}+f_i\tilde{u}_{ij})x_{ij} + \frac{1}{2}\mu_Q\sum_{i=1}^{m}(x_{ij}-x_{ij}^o)^2\right\}\right) \tag{7.16}$$

with the smoothing factors $\mu_Q > 0$ and $\mu_P > 0$. Recall that x^o is the minimizer of $d_Q(x)$ over Q and u^o is the minimizer of $d_T(u)$ over T, see (7.8) and (7.10). We note that both approximation functions, f_{μ_P} and ψ_{μ_Q}, are separable either per client or per facility as the objective functions, f and ψ, themselves.

Recall that we denote by $u_{\mu_P,\tilde{x}}$ the maximum over Δ_n^m of $\left\{\sum_{i=1}^m\sum_{j=1}^n f_i u_{ij}\tilde{x}_{ij} - \frac{1}{2}\mu_P\sum_{i=1}^m\sum_{j=1}^n(u_{ij}-u_{ij}^o)^2\right\}$ and by $x_{\mu_Q,\tilde{u}}$ the minimum over Δ_m^n of $\left\{\sum_{i=1}^m\sum_{j=1}^n(c_{ij}+f_i\tilde{u}_{ij})x_{ij} + \frac{1}{2}\mu_Q\sum_{i=1}^m\sum_{j=1}^n(x_{ij}-x_{ij}^o)^2\right\}$.

The gradient of the smooth approximation functions are Lipschitz continuous and are defined as follows,

$$\nabla f_{\mu_P}(\tilde{x})_{ij} := c_{ij} + f_i u_{\{\mu_P,\tilde{x}\}_{ij}} \qquad \nabla\psi_{\mu_Q}(\tilde{u})_{ij} := f_i x_{\{\mu_Q,\tilde{u}\}_{ij}}$$

for $i = 1,\ldots,m$ and $j = 1,\ldots,n$. Their corresponding Lipschitz constants are

$$L_{f_{\mu_P},Q} = \frac{f_{\max}^2}{\mu_P} \qquad L_{\psi_{\mu_Q},P} = \frac{f_{\max}^2}{\mu_Q}.$$

Hence, computing a gradient demands the use of Algorithm 11 to compute respectively $u_{\mu_P,\tilde{x}}$ and $x_{\mu_Q,\tilde{u}}$.

In Algorithm 12 we tailor the Excessive Gap algorithm (Algorithm 4) to the linear programming relaxation of the UFL problem. The initial smoothing factors μ_Q^0 and μ_P^0 are then given by

$$\mu_Q^0 := 2\|A\|_{Q,P}\sqrt{\frac{D_P}{\sigma_Q\sigma_P D_Q}} = 2f_{\max}\frac{m}{n}\sqrt{\frac{(n-1)}{(m-1)}}$$

and

$$\mu_P^0 := \|A\|_{Q,P}\sqrt{\frac{D_Q}{\sigma_Q\sigma_P D_P}} = f_{\max}\frac{n}{m}\sqrt{\frac{(m-1)}{(n-1)}}.$$

Before specifying Algorithm 12, we study the structure of the primal and the dual Gradient Mapping. Let $\tilde{x} \in \mathbb{R}^{mn}$, $\tilde{u} \in \mathbb{R}^{mn}$, $\mu_Q > 0$, and $\mu_P > 0$. The primal Gradient Mapping at $\tilde{x}$, $GM_{f_{\mu_P}}(\tilde{x})$, is defined as follows,

$$\begin{aligned} GM_{f_{\mu_P}}(\tilde{x}) &:= \arg\min_{y\in Q}\{\langle\nabla f_{\mu_P}(\tilde{x}), y-\tilde{x}\rangle + \frac{L_{f_{\mu_P},Q}}{2}\|y-\tilde{x}\|_2^2\} \\ &= \arg\min_{y\in Q}\{\langle\nabla f_{\mu_P}(\tilde{x}), y\rangle + \frac{L_{f_{\mu_P},Q}}{2}\|y-\tilde{x}\|_2^2\} \end{aligned}$$

and the dual Gradient Mapping at $\tilde{u}$, $GM_{\psi_{\mu_Q}}(\tilde{u})$, is defined as follows,

$$\begin{aligned} GM_{\psi_{\mu_Q}}(\tilde{u}) &:= \arg\max_{v\in P}\{\langle\nabla\psi_{\mu_Q}(\tilde{u}), v-\tilde{u}\rangle - \frac{L_{\psi_{\mu_Q},P}}{2}\|v-\tilde{u}\|_2^2\} \\ &= \arg\min_{v\in P}\{-\langle\nabla\psi_{\mu_Q}(\tilde{u}), v\rangle + \frac{L_{\psi_{\mu_Q},P}}{2}\|v-\tilde{u}\|_2^2\}. \end{aligned}$$

Both optimization problems are quadratic projections onto Δ_m^n, respectively on Δ_n^m. As shown in Lemma 7.2.1, these projections can be solved in either $O(nm\log m)$ or $O(mn\log n)$ with Algorithm 11.

Theorem 7.4 (Theorem 2 and Lemma 1 in [CE05])
Let $n > m$. Then, given $\epsilon > 0$, the UFL-EG-Euclidean Algorithm (12) delivers in $O\left(\frac{1}{\epsilon}f_{max}n^{3/2}m^{3/2}\log n\right)$ time a primal and a dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

Proof. Using the theoretical results in Theorem 5.4, we know that if we run $\frac{4\|A\|}{\epsilon}\sqrt{\frac{D_Q D_P}{\sigma_Q\sigma_P}}$ iterations of the Excessive Gap algorithm, we obtain a primal and dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$, $\epsilon > 0$. Thus we need to run $\frac{2f_{max}}{\epsilon}\sqrt{(n-1)(m-1)}$ iterations of the UFL-EG-Euclidean Algorithm to get a primal and dual solution with an additive gap of ϵ.

At each iteration of the UFL-EG Algorithm we either need to solve two quadratic projections onto Δ_m^n and one quadratic projection onto Δ_n^m or two quadratic projections onto Δ_n^m and one quadratic projection onto Δ_m^n. Assuming that there are more clients to deliver than potential facilities to open, i.e., $n > m$, we have that each iteration takes $O(mn\log(n))$ time. Moreover, recall that the evaluation of the primal and dual functions take $O(nm)$.

Thus finally, the running time of the UFL-EG-Euclidean Algorithm is

$$O\left(\frac{1}{\epsilon}f_{max}n^{3/2}m^{3/2}\log n\right).$$

Algorithm 12 Excessive Gap Algorithm for UFL (UFL-EG-Euclidean)

Input: - $x^o = \frac{1}{m}\mathbf{1} \in \mathbb{R}^{nm}$
- Initial smoothing factors μ_Q^0 and μ_P^0
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 0$
compute $x_0 = GM_{f_{\mu_P}^0}(x^o)$ `UFL:Quadratic projection onto` Δ_m^n
compute $u_0 = u_{\mu_Q^0, x^o}$ `UFL:Quadratic projection onto` Δ_n^m
while $f(x_k) - \psi(u_k) > \epsilon$ **do**
  if k is even **then**
    compute $x_{\mu_P^k, u_k}$ `UFL:Quadratic projection onto` Δ_m^n
    set $\hat{x} = \frac{k+1}{k+3}x_k + \frac{2}{k+3}x_{\mu_P^k, u_k}$
    compute $u_{\mu_Q^k, \hat{x}}$ `UFL:Quadratic projection onto` Δ_n^m
    set $u_{k+1} = \frac{k+1}{k+3}u_k + \frac{2}{k+3}u_{\mu_Q^k, \hat{x}}$
    compute $GM_{f_{\mu_P^k}}(\hat{x})$ `UFL:Quadratic projection onto` Δ_m^n
    set $x_{k+1} = GM_{f_{\mu_P^k}}(\hat{x})$
    set $\mu_Q^{k+1} = \frac{k+1}{k+3}\mu_Q^k$ and $\mu_P^{k+1} = \mu_P^k$
  else
    compute $u_{\mu_Q^k, x_k}$ `UFL:Quadratic projection onto` Δ_n^m
    set $\hat{u} = \frac{k+1}{k+3}u_k + \frac{2}{k+3}u_{\mu_Q^k, x_k}$
    compute $x_{\mu_P^k, \hat{u}}$ `UFL:Quadratic projection onto` Δ_m^n
    set $x_{k+1} = \frac{k+1}{k+3}x_k + \frac{2}{k+3}x_{\mu_P^k, \hat{u}}$
    compute $GM_{\psi_{\mu_Q^k}}(\hat{u})$ `UFL:Quadratic projection onto` Δ_n^m
    set $u_{k+1} = GM_{\psi_{\mu_Q^k}}(\hat{u})$
    set $\mu_Q^{k+1} = \mu_Q^k$ and $\mu_P^{k+1} = \frac{k+1}{k+3}\mu_P^k$
  end if
  set $k = k + 1$
  `UFL:Objective functions evaluation`
end while
set $\bar{x} = x_k$ and $\bar{u} = u_k$

□

Note that the difference with respect to the result presented in [CE05], Theorem 2 and Lemma 1, is the factor $\log n$ due to the sorting required for solving the quadratic projections onto Δ_n^m using Algorithm 11.

The UFL-EG-Euclidean Algorithm outperforms all three Primal-Dual Subgradient algorithms, in the dependence on the input data as well as in the dependence on the absolute error ϵ. The numerical results given in Section 7.4 strongly support this observation.

7.2.2 Entropy Function

We study now another choice of prox-functions and norms. For the space's norms, we consider the following combination of the norm 1 and the Euclidean norm. For $x \in \mathbb{R}^{nm}$ let us write it as $x = (x_1, \ldots, x_n)$ with $x_j \in \mathbb{R}^m$ for $j = 1, \ldots, n$ and endow the primal space with the norm $\|x\|_Q$ defined as follows,

$$\|x\|_Q := \left(\sum_{j=1}^{n} \|x_j\|_1^2 \right)^{1/2} = \left(\sum_{j=1}^{n} \left(\sum_{i=1}^{m} |x_{ij}| \right)^2 \right)^{1/2}.$$

Symmetrically, we endow the dual space with the norm $\|u\|_P$,

$$\|u\|_P := \left(\sum_{i=1}^{m} \|u_i\|_1^2 \right)^{1/2} = \left(\sum_{i=1}^{m} \left(\sum_{j=1}^{n} |u_{ij}| \right)^2 \right)^{1/2},$$

with $u = (u_1, \ldots, u_m)$, $u^i \in \mathbb{R}^n$ for $i = 1, \ldots, m$.

As prox-function, we choose the Entropy function defined for the primal space as follows

$$d_Q(x) \;:=\; n \ln m + \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} \ln x_{ij} \tag{7.17}$$

and for the dual space,

$$d_P(u) \;:=\; m \ln n + \sum_{i=1}^{m} \sum_{j=1}^{n} u_{ij} \ln u_{ij}. \tag{7.18}$$

Note that both functions are well defined over $Q(= \Delta_m^n)$ respectively over $P(= \Delta_n^m)$. Moreover, the convexity parameter $\sigma_Q = 1$ over Q and the convexity parameter $\sigma_P = 1$ over P.

Lemma 7.5 ([Nes05c], Lemma 3)
The prox-function d_Q is smooth and strongly convex with $\sigma_Q = 1$ over the set Q with respect to the norm $\|.\|_Q$.

Proof. Clearly, the function $d_Q(x)$ is smooth over its domain. From theorem of Taylor, we have that $d_Q(x+h) = d_Q(x) + \langle \nabla d_Q(x), h\rangle + \frac{1}{2}\langle d''_Q(\eta)h, h\rangle$ for η a convex combination of x and $(x+h)$, both in ri$Q$ and where the second derivative of d_Q is given by $\langle d''_Q(\eta)h, h\rangle := \sum_{i=1}^m \sum_{j=1}^n \frac{h_{ij}^2}{\eta_{ij}}$.

Now we fix j and using Cauchy-Schwartz we get a lower bound for the second derivative at η. Namely,

$$\|h^j\|_1^2 = \left(\sum_{i=1}^m \mid h_{ij} \mid\right)^2 = \left(\sum_{i=1}^m \frac{h_{ij}}{\sqrt{\eta_{ij}}}\sqrt{\eta_{ij}}\right)^2 \leq \sum_{i=1}^m \frac{h_{ij}^2}{\eta_{ij}} \sum_{i=1}^m \eta_{ij} = \sum_{i=1}^m \frac{h_{ij}^2}{\eta_{ij}}.$$

Summing over j we get $\|h\|_Q^2 \leq \langle d''_Q(\eta)h, h\rangle$ and thus,

$$d_Q(x+h) \geq d_Q(x) + \langle \nabla d_Q(x), h\rangle + \frac{1}{2}\|h\|_Q^2.$$

□

The minimizer of $d_Q(x)$ over Q is $x^o = \frac{1}{m}\mathbf{1}$ and the minimizer of $d_P(u)$ over P is $u^o = \frac{1}{n}\mathbf{1}$. Both can be easily computed using the KKT conditions. The maximum of $d_Q(x)$ over Q is attained at the extreme points of the set Q, namely if the maximum of a convex function over a convex set is attained, it is allways attained at an extreme point of the set. Then, $D_Q := \max_{x\in Q} d_Q(x) = n \ln m$. Respectively, $D_P := \max_{u\in P} d_P(u) = m \ln n$.

The norm of matrix A remains to be computed.

$$\begin{aligned}
\|A\|_{Q,P} &:= \max_{\|x\|_Q\leq 1}\max_{\|u\|_P\leq 1} \langle Ax, u\rangle \\
&= \max_{\|x\|_Q\leq 1}\max_{\|u\|_P\leq 1} \sum_{i=1}^m \sum_{j=1}^n f_i u_{ij} x_{ij} \\
&\leq \max_{\|x\|_Q\leq 1}\max_{\|u\|_P\leq 1} (\max_{1\leq i\leq m} f_i)\|x\|_2\|u\|_2 \\
&\quad (\text{for } w\in\mathbb{R}^n,\ \|w\|_2 \leq \|w\|_1 \leq \sqrt{n}\|w\|_2) \\
&\leq f_{\max} \max_{\|x\|_Q\leq 1}\max_{\|u\|_P\leq 1} \|x\|_Q\|u\|_P = f_{\max}. \qquad (7.19)
\end{aligned}$$

Since $\|.\|_2 \leq \|.\|_1$, it is not suprising that we get the same bound for the norm of A as when we consider Euclidean norms.

Primal Dual Subgradient Method

In order to use the Dual Averaging Method we need to evaluate an upper bound for the norm of the subgradients computed at each step of the method.

Lemma 7.6
Let $\xi : \mathbb{R}^{nm} \longrightarrow \mathbb{R}$ *defined as* $\xi(x) := \langle \xi, x \rangle$. *Then* $\|\xi\|_Q^* = \left(\sum_{j=1}^{2} (\max_{1 \le i \le m} |\xi_{ij}|)^2 \right)^{1/2}$.

Proof.

$$\begin{aligned}
\|\xi\|_Q^* &:= \max_{\|x\|_Q \le 1} \sum_{j=1}^{n} \sum_{i=1}^{m} \xi_{ij} x_{ij} \le \max_{\|x\|_Q \le 1} \sum_{j=1}^{n} \sum_{i=1}^{m} |\xi_{ij}||x_{ij}| \\
&\le \max_{\|x\|_Q \le 1} \sum_{j=1}^{n} (\max_{1 \le i \le m} |\xi_{ij}|) \sum_{i=1}^{m} |x_{ij}| \\
&\le \max_{\|x\|_Q \le 1} \left(\sum_{j=1}^{n} (\max_{1 \le i \le m} |\xi_{ij}|)^2 \right)^{1/2} \left(\sum_{j=1}^{n} (\sum_{i=1}^{m} |x_{ij}|)^2 \right)^{1/2} \\
&\le \left(\sum_{j=1}^{n} (\max_{1 \le i \le m} |\xi_{ij}|)^2 \right)^{1/2} = \left(\sum_{j=1}^{n} \|\xi^j\|_\infty \right)^{1/2}
\end{aligned}$$

Now define $\bar{x}_{kj} := \begin{cases} \xi_{kj} & \text{if } \max_{1 \le i \le m} |\xi_{ij}| = |\xi_{kj}| \\ 0 & \text{otherwise} \end{cases}$ and $x := \frac{\bar{x}}{\|\bar{x}\|_Q}$. Then,

$$\sum_{j=1}^{n} \sum_{i=1}^{m} \xi_{ij} x_{ij} = \sum_{j=1}^{n} \sum_{i=1}^{m} \xi_{ij} \frac{\bar{x}_{ij}}{\|\bar{x}\|_Q} = \frac{\sum_{j=1}^{n} (\max_{1 \le i \le m} |\xi_{ij}|)^2}{(\sum_{j=1}^{n} (\max_{1 \le i \le m} |\xi_{ij}|)^2)^{1/2}} = \|\xi\|_Q^*.$$

□

Recall that $\xi_{ij} = c_{ij} + f_i u_{ij}^x$, $i = 1, \ldots, m$ and $j = 1, \ldots, n$ with $u^x \in U(x)$ is a subgradient of $f(x)$, $x \in \mathbb{R}^{mn}$ (see Equation 7.7). Then,

$$\begin{aligned}
\|\xi\|_Q^* &= \left(\sum_{j=1}^{n} (\max_{1 \le i \le m} |c_{ij} + f_i u_{ij}^x|)^2 \right)^{1/2} \le \left(\sum_{j=1}^{n} (\max_{1 \le i \le m} c_{ij} + f_{max})^2 \right)^{1/2} \\
&\le \sqrt{n}(c_{max} + f_{max}) =: L
\end{aligned}$$

where $c_{max} = \max_{1 \le j \le n} \max_{1 \le i \le m} c_{ij}$.

To use the Truncated Dual Averaging Algorithm we need to bound the subgradient variations. Thus, let x and $y \in \mathbb{R}^{nm}$ and let ξ_x and ξ_y be the subgradient of f at x, respectively the subgradient of f at y. Then, for $u^x \in U(x)$ and $u^y \in U(y)$,

$$\begin{aligned}
\|\xi_x - \xi_y\|_Q^* &= \left(\sum_{j=1}^{n} (\max_{1 \le i \le m} |c_{ij} + f_i u_{ij}^x - (c_{ij} + f_i u_{ij}^y)|)^2 \right)^{1/2} \\
&= \left(\sum_{j=1}^{n} (\max_{1 \le i \le m} f_i |u_{ij}^x - u_{ij}^y|)^2 \right)^{1/2} \\
&\le \left(\sum_{j=1}^{n} f_{max}^2 \right)^{1/2} \le \sqrt{n} f_{max} =: M.
\end{aligned}$$

Now we have all the elements needed to apply the Primal-Dual Subgradients algorithms to the linear programming relaxation of the UFL problem. Algorithm 13 corresponds to Algorithm 2 using the simple averages sequences (4.12), Algorithm 14 corresponds to Algorithm 2 using the weighted averages sequences (4.13), and Algorithm 15 corresponds to Algorithm 3.

Algorithm 13 Simple Dual Averaging Algorithm for UFL (UFL-SDA-Entropy)

Input: - $x^o := \frac{1}{m}\mathbf{1}$
- The constants D_Q and L
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 0$ and $\hat{\beta}_0 = 1$
set $x_0 = x^o$
compute $\xi_0 \in \partial f(x^0)$ and u_0 `UFL:Subgradient`
set $\zeta_0 = \xi_0$
set $\bar{x} = x_0$ and $\bar{u} = u_0$.
while $f(\bar{x}) - \psi(\bar{u}) > \epsilon$ **do**
  set $k = k + 1$
  compute $\hat{\beta}_k = \hat{\beta}_{k-1} + \frac{1}{\hat{\beta}_{k-1}}$ and $\beta_k = \frac{L}{\sqrt{2D_Q}}\hat{\beta}_k$
  compute $x_k = \arg\min_{x \in Q} \{\langle \zeta_{k-1}, x \rangle + \beta_k \sum_{j=1}^{n} \sum_{i=1}^{m} x_{ij} \ln x_{ij}\}$
  `UFL:Entropy projection onto` Δ_m^n
  compute $\xi_k \in \partial f(x_k)$ and u_k `UFL:Subgradient`
  set $\zeta_k = \zeta_{k-1} + \xi_k$
  $\bar{x} = \frac{1}{k+1} \sum_{l=0}^{k} x_l$ and $\bar{u} = \frac{1}{k+1} \sum_{l=0}^{k} u_l$ `UFL:Objective functions`
  `evaluation`
end while

Algorithm 14 Weighted Dual Averaging Algorithm for UFL (UFL-WDA-Entropy)

Input: - $x^o := \frac{1}{m}\mathbf{1}$
- The constant D_Q
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 0$, and $\hat{\beta}_0 = 1$
set $x_0 = x^o$
compute $\xi_0 \in \partial f(x_0)$ and u_0 `UFL:Subgradient`
set $\lambda_0 = \frac{1}{\|\xi_0\|_2}$, $\Lambda_0 = \lambda_0$, and $\zeta_0 = \lambda_0 \xi_0$
set $\bar{x} = x_0$ and $\bar{u} = u_0$.
while $f(\bar{x}) - \psi(\bar{u}) > \epsilon$ **do**
  set $k = k + 1$
  compute $\hat{\beta}_k = \hat{\beta}_{k-1} + \frac{1}{\hat{\beta}_{k-1}}$ and $\beta_k = \frac{1}{\sqrt{2D_Q}}\hat{\beta}_k$
  compute $x_k = \arg\min_{x \in Q} \{\langle \zeta_{k-1}, x \rangle + \beta_k \sum_{j=1}^{n} \sum_{i=1}^{m} x_{ij} \ln x_{ij}\}$
    `UFL:Entropy projection onto` Δ_m^n
  compute $\xi_k \in \partial f(x_k)$ and u_k `UFL:Subgradient`
  set $\lambda_k = \frac{1}{\|\xi_k\|_2}$, $\Lambda_k = \Lambda_{k-1} + \lambda_k$, and $\zeta_k = \zeta_{k-1} + \lambda_k \xi_k$
  set $\bar{x} = \frac{1}{\Lambda_k} \sum_{l=0}^{k} \lambda_l x_l$ and $\bar{u} = \frac{1}{\Lambda_k} \sum_{l=0}^{k} \lambda_l u_l$
    `UFL:Objective functions evaluation`
end while

Algorithm 15 Truncated Simple Dual Averaging Algorithm for UFL (UFL-TSDA-Entropy)

Input: - $x^o := \frac{1}{m}\mathbf{1}$
- The constants D_Q and M
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 1$ and $\beta_1 = 1$
set $x_0 = x^o$
compute $\xi_0 \in \partial f(x_0)$ and u_0 `UFL:Subgradient`
compute $x_1 = \arg\min_{x \in Q} \{\langle \xi_0, x \rangle + \beta_1 \sum_{j=1}^n \sum_{i=1}^m x_{ij} \ln x_{ij}\}$
`UFL:Quadratic projection onto` Δ_m^n
compute $\xi_1 \in \partial f(x_1)$ and u_1 `UFL:Subgradient`
set $\zeta_1 = \xi_1$
$\bar{x} = x_1$ and $\bar{u} = u_1$ `UFL:Objective functions evaluation`
while $f(\bar{x}) - \psi(\bar{u}) > \epsilon$ **do**
  set $k = k + 1$
  compute $x_k = \arg\min_{x \in Q} \{\langle \zeta_{k-1}, x \rangle + \beta_k \sum_{j=1}^n \sum_{i=1}^m x_{ij} \ln x_{ij}\}$
  `UFL:Entropy projection onto` Δ_m^n
  set $\beta_k = \frac{M}{\sqrt{2D_Q}}\sqrt{k}$
  compute $\xi_k \in \partial f(x_k)$ and u_k `UFL:Subgradient`
  set $\zeta_k = \zeta_{k-1} + \xi_k$
  $\bar{x} = \frac{1}{k}\sum_{l=1}^k x_l$ and $\bar{u} = \frac{1}{k}\sum_{l=1}^k u_l$ `UFL:Objective functions evaluation`
end while

In the following we study the complexity of the oracle needed by the three algorithms, Algorithm 13, 14, and 15, i.e., `UFL:Entropy projection onto` Δ_m^n.

First we observe that due to the separability by clients of the Entropy function $d_Q(x)$ we can rewrite the projection onto Δ_m^n as follows: at iteration k we have $x_{k+1} := (x_1, \ldots, x_n)$ where for $j = 1, \ldots, n$

$$x_j := \arg\min_{y \in \Delta_m} \left\{ \sum_{i=1}^{m} \zeta_{k-1_{ij}} y_i + \beta_k \sum_{i=1}^{m} y_i \ln y_i \right\}. \tag{7.20}$$

Using the KKT conditions of a single optimization problem in (7.20), we can show that this problem has a closed form solution.

Namely, let the problem described by Equation (7.21) be a generic formulation of the optimization problem (7.20),

$$\begin{aligned} &\text{minimize} && \sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} y_{ij} + b \sum_{j=1}^{n}\sum_{i=1}^{m} y_{ij} \ln y_{ij} \\ &\text{subject to} && y \in \Delta_m^n \end{aligned} \tag{7.21}$$

where $a \in \mathbb{R}^{nm}$ and $b \in \mathbb{R}_+$, Lemma 7.7 solves Equation (7.21).

Lemma 7.7
Consider the generic optimization problem of Equation (7.21) and let $y \in \mathbb{R}^{nm}$ *and* $(\lambda, w) \in \mathbb{R}^{n+nm}$ *be defined as follows:*

$$y_{ij} := \frac{e^{-a_{ij}/b}}{\sum_{l=1}^{m} e^{-a_{lj}/b}} \qquad \forall\ i = 1, \ldots, m\ j = 1, \ldots, n \tag{7.22}$$

$$\lambda_j := -b\left(-\ln\left(\sum_{l=1}^{m} e^{-a_{lj}/b}\right) + 1\right) \qquad \forall\ j = 1, \ldots, n \tag{7.23}$$

$$w_{ij} := 0 \qquad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n. \tag{7.24}$$

Then, y *is primal feasible and optimal and* (λ, w) *is dual feasible and optimal.*

Proof. We verify that the solution y and (λ, w) satisfies the KKT conditions for the optimization problem described by Equation (7.21):

1. $a_{ij} + b(\ln y_{ij} + 1) + \lambda_j - w_{ij} = 0 \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n$
2. $y_{ij} w_{ij} = 0 \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n$ *(orthogonality)*
3. $\sum_{i=1}^{m} y_{ij} = 1 \quad \forall\ j = 1, \ldots, n$ and $y_{ij} \geq 0 \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n$ *(primal feasibility)*

4. $w_{ij} \geq 0 \quad \forall\, i = 1, \ldots, m,\ j = 1, \ldots, n$ *(dual feasibility)*

Since $w_{ij} = 0$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$, the orthogonality condition (2) as well as the dual feasibility (4) are trivially satisfied. The primal feasibility (3) is also easily verified. Namely, for $i = 1, \ldots, m$ and $j = 1, \ldots, n$, $y_{ij} = \frac{e^{-a_{ij}/b}}{\sum_{l=1}^{m} e^{-a_{lj}/b}} > 0$ and for fixed j we have $\sum_{i=1}^{m} y_{ij} = \sum_{i=1}^{m} (e^{-a_{ij}/b})/(\sum_{l=1}^{m} e^{-a_{lj}/b}) = 1$. Finally,

$$\begin{aligned} a_{ij} \;+\; & b(\ln y_{ij} + 1) + \lambda_j - w_{ij} \\ = \; & a_{ij} + b\left(-a_{ij}/b + \ln\left(1/\sum_{l=1}^{m} e^{-a_{lj}/b}\right)\right) - b\left(\ln\left(1/\sum_{l=1}^{m} e^{-a_{lj}/b}\right) + 1\right) \\ = \; & 0 \end{aligned}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. All KKT conditions are satisfied and thus y is primal feasible and optimal and (λ, w) is dual feasible and optimal. □

We note that the computation of the primal solution y in Lemma 7.7 requires $O(nm)$ time. Now we are able to derive the running time of the Algorithms 13, 14, and 15.

Theorem 7.8
Given $\epsilon > 0$, the UFL-SDA-Entropy Algorithm (13) and the UFL-WDA-Entropy Algorithm (14) output in $O\left(\frac{1}{\epsilon^2}(c_{max} + f_{max})^2 n^3 m \ln m\right)$ time a primal and a dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

Proof. Using the theoretical results in Theorem 4.7, we know that if we run either the Simple Dual Averaging Algorithm or the Weighted Dual Averaging Algorithm for $(\frac{9L^2}{\epsilon^2}\frac{D_Q}{\sigma_Q} - 1)$ iterations, we obtain a primal and dual solution , $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$, $\epsilon > 0$. Thus we need to run

$$\frac{9n(c_{max} + f_{max})^2}{\epsilon^2} n \ln m - 1 \leq \frac{9n^2(c_{max} + f_{max})^2 \ln m}{\epsilon^2}$$

of the UFL-SDA Algorithm or the UFL-WDA Algorithm to have a primal and a dual solution with an additive gap of ϵ.

At each iteration we need to compute a subgradient, which requires $O(nm)$ time, an entropy projection on Δ_m^n, which requires $O(nm)$ time, and to evaluate both primal and dual objective functions, which requires $O(nm)$ time. Thus, each iteration takes $O(nm)$ time and the running time of the considered algorithms is

$$O\left(\frac{1}{\epsilon^2} n^3 m \ln m (c_{max} + f_{max})^2\right).$$

□

Theorem 7.9
Given $\epsilon > 0$, the UFL-TSDA-Entropy Algorithm (15) outputs in $O\left(\frac{1}{\epsilon^2} f_{max}^2 n^3 m \ln m\right)$ time a primal and a dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

Proof. Using the theoretical results in Theorem 4.8, we know that if we run $\frac{8M^2}{\epsilon^2}\frac{D_Q}{\sigma_Q}$ iterations of the Truncated Simple Dual Averaging Algorithm, we obtain a primal and dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$, $\epsilon > 0$. Thus, we need to run UFL-TSDA-Entropy Algorithm for $\frac{8n^2 \ln m f_{max}^2}{\epsilon^2}$ iterations to get a primal and a dual solution with an additive gap of ϵ.

As for the UFL-SDA-Entropy Algorithm and the UFL-WDA-Entropy Algorithm, each iteration takes $O(nm)$. Finally, the running time of the UFL-TSDA-Entropy Algorithm is

$$O\left(\frac{1}{\epsilon^2} f_{max}^2 n^3 m \ln m\right).$$

□

As with the algorithms based on the Euclidean Norm, we note that UFL-TSDA-Entropy Algorithm outperforms the other two algorithms.

Excessive Gap Method

We now apply the Excessive Gap method to the linear programming relaxation of the UFL problem using Entropy functions and Entropy norms for defining the primal and dual smooth approximations,

$$f_{\mu_P}(\tilde{x}) := \sum_{i=1}^{m}\left(\sum_{j=1}^{n} c_{ij}\tilde{x}_{ij} + \max_{u_i \in \Delta_n}\left\{\sum_{j=1}^{n} f_i u_{ij}\tilde{x}_{ij} - \mu_P \sum_{j=1}^{n} u_{ij}\ln u_{ij}\right\}\right) \tag{7.25}$$

$$\psi_{\mu_Q}(\tilde{u}) := \sum_{j=1}^{n}\left(\min_{x_j \in \Delta_m}\left\{\sum_{i=1}^{m}(c_{ij} + f_i\tilde{u}_{ij})x_{ij} + \mu_Q \sum_{i=1}^{m} x_{ij}\ln x_{ij}\right\}\right) \tag{7.26}$$

with the smoothing factors $\mu_Q > 0$ and $\mu_P > 0$. Note that both approximation functions, f_{μ_P} and ψ_{μ_Q}, are separable either per client or per facility as the objective functions, f and ψ, themselves.

The gradient of the smooth approximation functions are Lipschitz continuous and are defined as follows,

$$\nabla f_{\mu_P}(\tilde{x})_{ij} := c_{ij} + f_i u_{\{\mu_P,\tilde{x}\}_{ij}} \qquad \nabla \psi_{\mu_Q}(\tilde{u})_{ij} := f_i x_{\{\mu_Q,\tilde{u}\}_{ij}}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Recall that we denote by $u_{\mu_P,\tilde{x}}$ the maximum of $\left\{\sum_{i=1}^{m}\sum_{j=1}^{n} f_i u_{ij}\tilde{x}_{ij} - \mu_P \sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij}\ln u_{ij}\right\}$ over Δ_n^m and by $x_{\mu_Q,\tilde{u}}$ the minimum of $\left\{\sum_{i=1}^{m}\sum_{j=1}^{n}(c_{ij} + f_i\tilde{u}_{ij})x_{ij} + \mu_Q \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}\ln x_{ij}\right\}$ over Δ_m^n. Their corresponding Lipschitz constants are

$$L_{f_{\mu_P},Q} = \frac{f_{\max}^2}{\mu_P} \qquad L_{\psi_{\mu_Q},P} = \frac{f_{\max}^2}{\mu_Q}.$$

Consider Algorithm 16, which corresponds to the application of the Excessive Gap algorithm (Algorithm 4) to the linear programming relaxation of the UFL problem. The initial smoothing factors μ_Q^0 and μ_P^0 are given by

$$\mu_Q^0 := 2\|A\|_{Q,P}\sqrt{\frac{D_P}{\sigma_Q\sigma_P D_Q}} = 2f_{max}\sqrt{\frac{m\ln n}{n\ln m}}$$

and

$$\mu_P^0 := \|A\|_{Q,P}\sqrt{\frac{D_Q}{\sigma_Q\sigma_P D_P}} = f_{max}\sqrt{\frac{n\ln m}{m\ln n}}.$$

Now, we study the structure of the primal and the dual Gradient Mapping, the main object in the Excessive Gap method. Let $\tilde{x} \in \mathbb{R}^{mn}$, $\tilde{u} \in \mathbb{R}^{mn}$, $\mu_Q > 0$, and $\mu_P > 0$. The primal Gradient Mapping at $\tilde{x}$, $GM_{f_{\mu_P}}(\tilde{x})$, is defined as follows,

$$GM_{f_{\mu_P}}(\tilde{x}) \;:=\; \arg\min_{y\in Q}\left\{\langle\nabla f_{\mu_P}(\tilde{x}), y-\tilde{x}\rangle + \frac{L_{f_{\mu_P},Q}}{2}\|y-\tilde{x}\|_Q^2\right\} \tag{7.27}$$

and the dual Gradient Mapping at $\tilde{u}$, $GM_{\psi_{\mu_Q}}(\tilde{u})$, is defined as follows,

$$GM_{\psi_{\mu_Q}}(\tilde{u}) \;:=\; \arg\max_{v\in P}\left\{\langle\nabla\psi_{\mu_Q}(\tilde{u}), v-\tilde{u}\rangle - \frac{L_{\psi_{\mu_Q},P}}{2}\|v-\tilde{u}\|_P^2\right\}. \tag{7.28}$$

We recall the definitions of both primal and dual norms,

$$\|y\|_Q := \left(\sum_{j=1}^{n}\left(\sum_{i=1}^{m}|x_{ij}|\right)^2\right)^{1/2} \text{ and } \|u\|_P := \left(\sum_{i=1}^{m}\left(\sum_{j=1}^{n}|u_{ij}|\right)^2\right)^{1/2}.$$

The optimization problems (7.27) and (7.28) are either separable per client or per facility. However, they are not as simple as when Euclidean norms are considered. Once a client or a facility is fixed the remaining optimization problems are of the following type.

$$\begin{aligned} opt := \text{minimize} \quad & \sum_{i=1}^{m} a_i(y_i - \bar{y}_i) + \frac{1}{2}b\left(\sum_{i=1}^{m}|y_i-\bar{y}_i|\right)^2 \\ \text{subject to} \quad & y \in \Delta_m \end{aligned} \tag{7.29}$$

where $a \in \mathbb{R}^m$, $\bar{y} \in \Delta_m$, and $b \in \mathbb{R}_+$.

Next lemma is the key for solving the previous optimization problems.

Lemma 7.10 ([Nes05c], Lemma 6)
Let $\|.\|$ be any norm in $\mathbb{R}^n$ and let $\|.\|^*$ be its dual norm. Then, for any $b > 0$ and any h and g

$$\langle g, h\rangle + \frac{1}{2}b\|h\|^2 = \max_s \left\{\langle s, h\rangle - \frac{1}{2b}\|s-g\|^{*2}\right\} \tag{7.30}$$

holds.

As in Section 5 in [Nes05c], we use Lemma 7.10 to rewrite the optimum of problem (7.29) as follows. Note that we consider the norm $\|.\|_1$ and its dual $\|.\|_\infty$.

$$\begin{aligned}
opt &:= \min_{y\in\Delta_m}\left\{\langle a, y-\bar{y}\rangle + \frac{1}{2}b\left(\sum_{i=1}^m |y_i-\bar{y}_i|\right)^2\right\} \\
&= \min_{y\in\Delta_m}\max_s\left\{\langle s, y-\bar{y}\rangle - \frac{1}{2b}(\|s-a\|^*)^2\right\} \\
&= \min_{y\geq 0}\max_{s,\lambda}\left\{\langle s, y-\bar{y}\rangle - \frac{1}{2b}(\|s-a\|^*)^2 + \lambda(1-\langle \mathbf{1}, y\rangle)\right\} \\
&= \max_{s,\lambda}\left\{-\langle s,\bar{y}\rangle - \frac{1}{2b}(\|s-a\|^*)^2 + \lambda \;\middle|\; s_i \geq \lambda \text{ for } i=1,\ldots,m\right\} \\
&= \max_{s,\lambda,\tau}\left\{-\langle s,\bar{y}\rangle - \frac{\tau^2}{2b} + \lambda \;\middle|\; s_i \geq \lambda \text{ and } |s_i-a_i|\leq\tau \text{ for } i=1,\ldots,m\right\}
\end{aligned}$$

Since $|s_i - a_i| \leq \tau$ for $i = 1,\ldots,m$, we have $s_i = \max\{a_i - \tau, \lambda\}$ for $i = 1,\ldots,m$ and $\lambda \leq a_i + \tau$ must hold for $i = 1,\ldots,m$ in order to have a feasible problem. Then,

$$-opt = \min_{\lambda,\tau}\left\{\sum_{i=1}^m \bar{y}_i \max\{a_i-\tau,\lambda\} + \frac{\tau^2}{2b} - \lambda \;\middle|\; \lambda \leq a_{min}+\tau, \tau\geq 0\right\}$$

where a_{min} is the smallest component of vector a.

Since the objective function is decreasing in λ, we have that the optimal $\lambda^* = a_{min} + \tau^*$, where τ^* optimal. Thus,

$$-opt = \min_{\tau\geq 0}\left\{\sum_{i=1}^m \bar{y}_i \max\{a_i - a_{min} - 2\tau, 0\} + \frac{\tau^2}{2b}\right\}. \tag{7.31}$$

Assume now that the components of vector a are ordered increasingly and note that the objective function in (7.31) is defined differently on the following intervals $]-\infty, 0], [0, \frac{a_2-a_1}{2}], [\frac{a_2-a_1}{2}, \frac{a_3-a_1}{2}], \ldots, [\frac{a_{m-1}-a_1}{2}, \frac{a_m-a_1}{2}], [\frac{a_m-a_1}{2}, +\infty[$. In order to find

the optimal τ^* it is sufficient to study the objective function in these intervals, i.e., to check its derivative at the following points $0, \frac{a_2-a_1}{2}, \ldots, \frac{a_m-a_1}{2}$.

Suppose now that the optimal τ^* is in the following interval $[\frac{a_{k-1}-a_1}{2}, \frac{a_k-a_1}{2}]$. Then, either $\tau^* := 2b\sum_{i=k}^m \bar{y}_i$ or $\tau^* := \frac{a_{k-1}-a_1}{2}$ and $s_i^* = \max\{a_i - \tau^*, \lambda^*\} = \max\{a_i - \tau^*, \tau^* + a_1\} = \begin{cases} \tau^* + a_1 & \text{for } i = 1, \ldots, k-1 \\ a_i - \tau^* & \text{for } i = k, \ldots, m \end{cases}$. Moreover $||s^* - a||^* = \max_{1 \le i \le m} |s_i - a_i| = \tau^*$.

The optimal y^* remains to be computed. For that sake we consider Lemma 7.10 again. Namely, we look for $y^* \in \Delta_m$ such that $0 \in \partial_s\{-\langle s^*, y - \bar{y}\rangle + \frac{1}{2b}(\|s^* - a\|^*)^2\}$. Assume for the sake of simplicity that all components of a are different, then by definition of s^* we have that $||s^* - a||^* = \max_i\{|s_i^* - a_i|\}$ is attained at $i = 1$ and $i = k, \ldots, m$. Now, define $\zeta := -y^* + \bar{y} + \frac{\tau^*}{b}(\lambda_1 e_1 - \sum_{i=k}^m \lambda_i e_i)$ with $\lambda_1 + \sum_{i=k}^m \lambda_i = 1$ and $\lambda_1, \lambda_k, \ldots, \lambda_m \ge 0$, and note that $\zeta \in \partial_s\{-\langle s^*, y^* - \bar{y}\rangle + \frac{1}{2b}(\|s^* - a\|^*)^2\}$. Then, we compute y^* by setting $\zeta = 0$ and choosing the convex combination of the vectors $e_1, -e_k, \ldots, -e_m$, i.e., choosing $\lambda_1, \lambda_k, \ldots, \lambda_m$, which generate $y^* \in \Delta_m$.

To continue, we note that to compute y^* we need $O(m \log m)$, which corresponds to the sorting of the vector a. Since for computing the primal gradient mapping $GM_{f_{\mu_P}}(\tilde{x})$ we need to solve a problem of the type of the generic problem (7.29) n times, the evaluation of $GM_{\psi_{\mu_Q}}(\tilde{u})$ requires $O(nm \log m)$. Respectively, for computing the dual gradient mapping we need $O(mn \log n)$.

Now, let us evaluate the running time of UFL-EG-Entropy Algorithm.

Theorem 7.11
Let $n > m$. Then, given $\epsilon > 0$, the UFL-EG-Entropy Algorithm (16) delivers in $O\left(\frac{1}{\epsilon} n^{3/2} m^{3/2} \log n \sqrt{\ln n \ln m} f_{max}\right)$ time a primal and a dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \le \epsilon$.

Proof. Using the theoretical results in Theorem 5.4, we know that if we run $\frac{4\|A\|}{\epsilon}\sqrt{\frac{D_Q D_P}{\sigma_Q \sigma_P}}$ iterations of the Excessive Gap algorithm, we obtain a primal and dual solution, $\bar{x} \in \Delta_m^n$ and $\bar{u} \in \Delta_n^m$, such that $f(\bar{x}) - \psi(\bar{u}) \le \epsilon$, $\epsilon > 0$. Thus we need to run $\frac{2f_{max}}{\epsilon}\sqrt{nm \ln n \ln m}$ iterations of the UFL-EG-Entropy Algorithm to get a primal and dual solution with an additive gap of ϵ. At each iteration of the UFL-EG Algorithm we either need to solve two entropy projections, one onto Δ_m^n and one onto Δ_n^m, which take $O(nm)$ time and one projection of the square of the entropy norm either onto Δ_n^m or into Δ_m^n, which takes either $O(nm \log m)$ or $O(mn \log n)$. Assuming that there are more clients to deliver than potential facilities to open, i.e., $n > m$, we have that each iteration takes $O(mn \log n)$ time. Moreover, recall that the evaluation of the primal and dual functions take $O(nm)$. Finally, the running time of the UFL-EG-Entropy Algorithm is $O\left(\frac{1}{\epsilon} n^{3/2} m^{3/2} \log n \sqrt{\ln n \ln m} f_{max}\right)$. □

Algorithm 16 Excessive Gap Algorithm for UFL (UFL-EG-Entropy)

Input: - $x^o = \frac{1}{m}\mathbf{1} \in \mathbb{R}^{nm}$
- Initial smoothing factors μ_Q^0 and μ_P^0
- An absolute error $\epsilon > 0$

Output: An approximate primal solution $\bar{x} \in \Delta_m^n$ and an approximate dual solution $\bar{u} \in \Delta_n^m$ such that $f(\bar{x}) - \psi(\bar{u}) \leq \epsilon$.

set $k = 0$
compute $x_0 = GM_{f_{\mu_P^0}}(x^o)$ `UFL:Entropy norm projection onto` Δ_m^n
compute $u_0 = u_{\mu_Q^0, x^o}$ `UFL:Entropy projection onto` Δ_n^m
while $f(x_k) - \psi(u_k) > \epsilon$ **do**
  if k is even **then**
    compute $x_{\mu_P^k, u_k}$ `UFL:Entropy projection onto` Δ_m^n
    set $\hat{x} = \frac{k+1}{k+3}x_k + \frac{2}{k+3}x_{\mu_P^k, u_k}$
    compute $u_{\mu_Q^k, \hat{x}}$ `UFL:Entropy projection onto` Δ_n^m
    set $u_{k+1} = \frac{k+1}{k+3}u_k + \frac{2}{k+3}u_{\mu_Q^k, \hat{x}}$
    compute $GM_{f_{\mu_P^k}}(\hat{x})$ `UFL:Entropy norm projection onto` Δ_m^n
    set $x_{k+1} = GM_{f_{\mu_P^k}}(\hat{x})$
    set $\mu_Q^{k+1} = \frac{k+1}{k+3}\mu_Q^k$ and $\mu_P^{k+1} = \mu_P^k$
  else
    compute $u_{\mu_Q^k, x_k}$ `UFL:Entropy projection onto` Δ_n^m
    set $\hat{u} = \frac{k+1}{k+3}u_k + \frac{2}{k+3}u_{\mu_Q^k, x_k}$
    compute $x_{\mu_P^k, \hat{u}}$ `UFL:Entropy projection onto` Δ_m^n
    set $x_{k+1} = \frac{k+1}{k+3}x_k + \frac{2}{k+3}x_{\mu_P^k, \hat{u}}$
    compute $GM_{\psi_{\mu_Q^k}}(\hat{u})$ `UFL:Entropy norm projection onto` Δ_n^m
    set $u_{k+1} = GM_{\psi_{\mu_Q^k}}(\hat{u})$
    set $\mu_Q^{k+1} = \mu_Q^k$ and $\mu_P^{k+1} = \frac{k+1}{k+3}\mu_P^k$
  end if
  set $k = k + 1$
  `UFL:Objective functions evaluation`
end while
set $\bar{x} = x_k$ and $\bar{u} = u_k$

7.3 From an Absolute to a Relative Error

In this section, we construct a decision making procedure A-FEAS, using the algorithms presented in the previous sections as explained in Chapter 3.3. Table 7.1 resumes the running time of the different algorithms for the linear programming relaxation of the UFL problem. We note that all either depend on $\|c\|_2$, $\|\bar{f}\|$, $\|c+\tilde{f}\|_2$, c_{max}, or f_{max}, which we would like to remove. In order to create the decision making

	Running Time		
	dependence of # of iterations		time per iteration
	on ϵ	on problem data	
UFL-SDA-Euclidean	$1/\epsilon^2$	$O(n\|c+\tilde{f}\|_2^2)$	$O(nm\log m)$
UFL-WDA-Euclidean	$1/\epsilon^2$	$O(n\|c+\tilde{f}\|_2^2)$	$O(nm\log m)$
UFL-TSDA-Euclidean	$1/\epsilon^2$	$O(n\|\bar{f}\|_2^2)$	$O(nm\log m)$
UFL-SDA-Entropy	$1/\epsilon^2$	$O(n^2\ln m(c_{max}+f_{max})^2)$	$O(mn)$
UFL-WDA-Entropy	$1/\epsilon^2$	$O(n^2\ln m(c_{max}+f_{max})^2)$	$O(mn)$
UFL-TSDA-Entropy	$1/\epsilon^2$	$O(n^2\ln m f_{max}^2)$	$O(mn)$
UFL-EG-Euclidean	$1/\epsilon$	$O(\sqrt{nm}f_{max})$	$O(mn\log n)$
UFL-EG-Entropy	$1/\epsilon$	$O(\sqrt{nm\ln n\ln m}f_{max})$	$O(mn\log n)$

Tab. 7.1: Running Time of UFL Algorithms for an absolute error $\epsilon > 0$

procedure, we consider the following two results that follow closely [GK02].

Lemma 7.12 (Lemma 2 in [CE05])
Let $(\bar{x},\bar{y})$ be any feasible solution of UFL-LP. Then,

a) if there is some $\bar{y}_{i_o} > 0$ with $f_{i_o} > \sum_{i\in\mathcal{F}}\sum_{j\in\mathcal{D}} c_{ij}\bar{x}_{ij} + \sum_{i\in\mathcal{F}} f_i\bar{y}_i$, we can find a new solution $(\hat{x},\hat{y})$ with strictly smaller objective function value such that $\hat{y}_{i_o} = 0$,

b) if there is some $\bar{x}_{i_oj_o} > 0$ with $c_{i_oj_o} > \sum_{i\in\mathcal{F}}\sum_{j\in\mathcal{D}} c_{ij}\bar{x}_{ij} + \sum_{i\in\mathcal{F}} f_i\bar{y}_i$, we can find a new solution $(\hat{x},\hat{y})$ with strictly smaller objective function value such that $\hat{x}_{i_oj_o} = 0$.

Proof. For sake of clarity let us denote the objective function in UFL-LP by $p(x,y) := \sum_{i\in\mathcal{F}}\sum_{j\in\mathcal{D}} c_{ij}x_{ij} + \sum_{i\in\mathcal{F}} f_iy_i$.

a) We note first that $\bar{y}_{i_o} < 1$ must hold. We define then $(\hat{x},\hat{y})$ as follows,

$$\begin{aligned}
\hat{x}_{ij} &:= \tfrac{\bar{x}_{ij}}{1-\bar{x}_{i_oj}} && \forall\, i\in\mathcal{F}\setminus\{i_o\},\ j\in D,\\
\hat{x}_{i_oj} &:= 0 && \forall\, j\in\mathcal{D},\\
\hat{y}_i &:= \tfrac{\bar{y}_i}{1-\bar{y}_{i_o}} && \forall\, i\in\mathcal{F}\setminus\{i_o\},\\
\hat{y}_{i_o} &:= 0.
\end{aligned}$$

We can easily check the feasibility of $(\hat{x}, \hat{y})$. For fixed $j \in \mathcal{D}$, $\sum_{i\in\mathcal{F}} \hat{x}_{ij} = (\sum_{i\in\mathcal{F}\setminus\{i_o\}} \bar{x}_{ij})/(1-\bar{x}_{i_oj}) = (1-\bar{x}_{i_oj})/(1-\bar{x}_{i_oj}) = 1$ and for $i \in \mathcal{F}$ and $j \in D$, $\hat{x}_{ij} \leq \bar{y}_i/(1-\bar{x}_{i_oj}) \leq \bar{y}_i/(1-\bar{y}_{i_o}) = \hat{y}_i$. Furthermore, $p(\hat{x}, \hat{y}) \leq \frac{1}{1-\bar{y}_{i_o}}(p(\bar{x}, \bar{y}) - f_{i_o}\bar{y}_{i_o}) < \frac{1-\bar{y}_{i_o}}{1-\bar{y}_{i_o}} p(\bar{x}, \bar{y})$.

b) We note first that $\bar{x}_{i_oj_o} < 1$ must hold. We define then $(\hat{x}, \hat{y})$ as follows,

$$\begin{array}{rcll} \hat{x}_{ij} & := & \bar{x}_{ij} & \forall\ i \in \mathcal{F},\ j \in D \setminus \{j_o\}, \\ \hat{x}_{ij_o} & := & \frac{\bar{x}_{ij_o}}{1-\bar{x}_{i_oj_o}} & \forall\ i \in \mathcal{F} \setminus \{i_o\}, \\ \hat{x}_{i_oj_o} & := & 0 & \\ \hat{y}_i & := & \frac{\bar{y}_i}{1-\bar{x}_{i_oj_o}} & \forall\ i \in \mathcal{F}. \end{array}$$

$(\hat{x}, \hat{y})$ is feasible. For fixed $j \neq j_o$, $\sum_{i\in\mathcal{F}} \hat{x}_{ij} = \sum_{i\in\mathcal{F}} \bar{x}_{ij} = 1$ and $\sum_{i\in\mathcal{F}} \hat{x}_{ij_o} = \sum_{i\in\mathcal{F}} \bar{x}_{ij_o}/(1-\bar{x}_{i_oj_o}) - \bar{x}_{i_oj_o}/(1-\bar{x}_{i_oj_o}) = 1$. For all $i \in \mathcal{F}$ and $j \in \mathcal{D} \setminus \{j_o\}$, $\hat{x}_{ij} = \bar{x}_{ij} \leq \bar{y}_i \leq \bar{y}_i/(1-\bar{x}_{i_oj_o}) = \hat{y}_i$. For all $i \in \mathcal{F} \setminus \{i_o\}$, $\hat{x}_{ij_o} \leq \bar{x}_{ij_o}/(1-\bar{x}_{i_oj_o}) \leq y_i/(1-\bar{x}_{i_oj_o}) = \hat{y}_i$. Now, we evaluate the value of the objective function at $(\hat{x}, \hat{y})$, $p(\hat{x}, \hat{x}) \leq \frac{1}{1-\bar{x}_{i_oj_o}}(p(\bar{x}, \bar{y}) - c_{i_oj_o}\bar{x}_{i_oj_o}) < \frac{1-\bar{x}_{i_oj_o}}{1-\bar{x}_{i_oj_o}} p(\bar{x}, \bar{y})$.

□

Suppose that we know that the optimal value OPT of UFL-LP is greater than a certain bound R. Then, we can drop all facility locations with building costs bigger than R and we can prevent assigning clients to facilities with delivering costs greater than R. In the following lemma we derive an upper bound for the optimal value OPT of UFL-LP.

Lemma 7.13 (Lemma 3 in [CE05])
For each client $j \in D$, let $LB_j := \min_{i\in F}(c_{ij} + f_i)$ and $LB := \max_{j\in D} LB_j$. Then, $LB \leq OPT \leq nLB$ and a feasible solution to UFL-LP exists whose cost is also bounded by (n LB). Recall that OPT denotes the optimal value of UFL-LP.

Proof. It is clear that LB_j is a lower bound for each client $j \in \mathcal{D}$. For the upper bound consider the solution obtainend by opening all the facilities that realize the minima of LB_j, $j \in \mathcal{D}$. Then, the solution is feasible and its cost is at most $\sum_{j\in\mathcal{D}} \mathrm{LB}_j \leq n\ \mathrm{LB}$. □

For given $\epsilon, R > 0$, we construct a decision procedure A-FEAS using the UFL-SDA-Euclidean Algorithm as follows. First we remove all facility locations with building costs $f_i > R$ and we forbid a client to be assigned to a facility if the corresponding delivering costs are bigger than R and we run $N = \lceil \frac{9n^2m}{\epsilon^2} - 1 \rceil$ iterations of the algorithm. If the objective value at solution delivered by the algorithm, $\bar{x}_N$, is smaller or equal to $(1+\epsilon)R$, the decision procedure returns $\bar{x}_N$, or else it claims that $\mathrm{OPT} > R$.

	Running Time	
	# of iterations	time per iteration
UFL-SDA-Euclidean	$O(n^2m(\log\log n + \frac{1}{\epsilon^2}))$	$O(nm \log m)$
UFL-WDA-Euclidean	$O(n^2m(\log\log n + \frac{1}{\epsilon^2}))$	$O(nm \log m)$
UFL-TSDA-Euclidean	$O(nm(\log\log n + \frac{1}{\epsilon^2}))$	$O(nm \log m)$
UFL-SDA-Entropy	$O(n^2 \ln m(\log\log n + \frac{1}{\epsilon^2}))$	$O(mn)$
UFL-WDA-Entropy	$O(n^2 \ln m(\log\log n + \frac{1}{\epsilon^2}))$	$O(mn)$
UFL-TSDA-Entropy	$O(n^2 \ln m(\log\log n + \frac{1}{\epsilon^2}))$	$O(mn)$
UFL-EG-Euclidean	$O(\sqrt{nm}(\log\log n + \frac{1}{\epsilon}))$	$O(mn \log n)$
UFL-EG-Entropy	$O(\sqrt{nm \ln n \ln m}(\log\log n + \frac{1}{\epsilon}))$	$O(mn \log n)$

Tab. 7.2: Running Time of UFL Algorithms for a relative error $\epsilon > 0$

Let us show that the previous decision procedure is valid. For $\epsilon > 0$, we need to run $O(\frac{1}{\epsilon^2} n \|c + \tilde{f}\|_2^2)$ iterations of UFL-SDA-Euclidean to have a primal solution $\bar{x}_N$ and a dual solution $\bar{u}_n$ with $f(\bar{x}_N) - \psi(\bar{u}_N) \leq \epsilon$. Note that $\|c + \tilde{f}\|_2 \leq 2\sqrt{nm}R$ since we assume $OPT \leq R$ and we remove the too expensive assignments. Thus, if we run $N = \lceil \frac{9n^2m}{\epsilon^2} - 1 \rceil$ iterations we have

$$f(\bar{x}_N) \leq \epsilon R + \psi(\bar{u}_N) \leq \epsilon R + \text{ OPT } \leq (1 + \epsilon)R.$$

Finally we apply Lemma 3.8 using the initial solution given in Lemma 7.13 to obtain an algorithm, that finds an approximate solution with a relative error of ϵ in $O(n^2m(\log\log n + \frac{1}{\epsilon^2}))$ iterations.

The same idea can be applied using the other algorithms. Table 7.2 resumes their convergence results.

The fastest algorithms we are aware of for the UFL-LP have been developed by Young [You00], Garg and Khandekar [GK02], and Chudak [Chu03]. Their running time is $O(nm^2 \log n(\log\log n + \frac{1}{\epsilon^2})$. With respect to the dependence of the running time on ϵ, the Excessive Gap method improves this result. However, with respect to the dependence of the running time on the input size, our algorithms are less effective.

7.4 Numerical Results

The algorithms used here were tested using randomly generated instances of the UFL problem where the facility locations and the clients are located at random in the unit square $[0, 1] \times [0, 1]$. We proceeded as in [BC99]. We chose n points in the unit square uniformly and independently, each simultaneously corresponding to a

facility location and a client. The cost of delivering the clients are proportional to the Euclidean distance between the clients and the facility locations. All facilities have the same building costs. We considered three different types of building costs. The first type is $\sqrt{n}/10$, the second type is $\sqrt{n}/100$, and the third type is $\sqrt{n}/1000$. Depending on the building costs we get instances with different properties. In general, instances with high building costs are difficult to solve. To avoid numerical problems, we rounded the data to 4 significant digits and multiplied all values by 10^4. Thus, all data entries are integer. All results were obtained using a computer with a cpu running at 2.2 GHz.

We primarily compared the number of iterations that each algorithm required. As expected, the Primal-Dual Subgradient algorithms performed worse than the algorithms based on the Excessive Gap method, see Tables 7.3 and 7.5. Even if the theoretical running time is better for the algorithms using Euclidean norm, we had expected a better practical running time for the methods using the Entropy function, since our feasible spaces are products of simplices. However, when using the Excessive Gap method, the algorithm based on Euclidean norm, see Table 7.3, clearly outperformed the algorithm based on Entropy function, see Table 7.5. In case of the Primal-Dual Subgradient methods, the results show a less clear picture. Depending on the chosen building costs either the algorithms using the Euclidean norm or the algorithms using the Entropy function converged faster.

Tables 7.4 and 7.6 show the maximum number of iterations needed according to theory, which are at least an order of magnitude larger than needed in practice.

Number of Iterations					
n, m	building costs	UFL-EG Euclidean	UFL-SDA Euclidean	UFL-WDA Euclidean	UFL-TSDA Euclidean
500	$\sqrt{n}/10$	42	7463	5769	1091
	$\sqrt{n}/100$	29	100000 (0.084)	100000 (0.083)	331
	$\sqrt{n}/1000$	4	100000 (0.262)	100000 (0.261)	15
1000	$\sqrt{n}/10$	45	8138	6384	1917
	$\sqrt{n}/100$	39	100000 (0.094)	100000 (0.328)	592
	$\sqrt{n}/1000$	9	100000 (0.093)	100000 (0.328)	51
1500	$\sqrt{n}/10$	40	6167	5040	2458
	$\sqrt{n}/100$	38	100000 (0.083)	100000 (0.082)	581
	$\sqrt{n}/1000$	14	100000 (0.338)	100000 (0.337)	99

Tab. 7.3: Comparing the number of iterations required for a relative error $\epsilon = 0.05$ using Euclidean norm (maximal number of iterations allowed: 100000, in brackets relative error at this limit)

Figure 7.1, 7.2, 7.3, and 7.4 show the evolution of the primal value and the dual value

Theoretical Number of Iterations					
n, m	building costs	UFL-EG Euclidean	UFL-SDA Euclidean	UFL-WDA Euclidean	UFL-TSDA Euclidean
500	$\sqrt{n}/10$	573	$1.51*10^7$	$1.51*10^7$	$1.50*10^7$
	$\sqrt{n}/100$	129	$2.06*10^7$ (0.084)	$2.10*10^7$ (0.083)	$5.69*10^7$
	$\sqrt{n}/1000$	32	$8.48*10^6$ (0.262)	$8.54*10^6$ (0.261)	$2.36*10^8$
1000	$\sqrt{n}/10$	904	$3.74*10^7$	$3.74*10^7$	$3.70*10^7$
	$\sqrt{n}/100$	204	$4.16*10^7$ (0.094)	$4.24*10^7$ (0.328)	$1.43*10^8$
	$\sqrt{n}/1000$	45	$1.62*10^7$ (0.093)	$1.62*10^7$ (0.328)	$6.77*10^8$
1500	$\sqrt{n}/10$	1171	$6.40*10^7$	$6.40*10^7$	$6.30*10^7$
	$\sqrt{n}/100$	264	$2.06*10^7$ (0.083)	$9.12*10^7$ (0.082)	$2.40*10^8$
	$\sqrt{n}/1000$	62	$2.74*10^7$ (0.338)	$2.75*10^7$ (0.337)	$1.18*10^9$

Tab. 7.4: Comparing the theoretical number of iterations required for the relative errors attained in Table 7.3 using Euclidean norm

Number of Iterations					
n, m	building costs	UFL-EG Entropy	UFL-SDA Entropy	UFL-WDA Entropy	UFL-TSDA Entropy
500	$\sqrt{n}/10$	1409	78063	4308	15083
	$\sqrt{n}/100$	310	65516	399	707
	$\sqrt{n}/1000$	50	97042	363	11
1000	$\sqrt{n}/10$	2330	100000 (0.058)	67320	35288
	$\sqrt{n}/100$	530	100000 (0.051)	755	1013
	$\sqrt{n}/1000$	111	100000 (0.079)	452	46
1500	$\sqrt{n}/10$	2966	100000 (0.065)	100000 (0.076)	48523
	$\sqrt{n}/100$	689	100000 (0.054)	1354	3021
	$\sqrt{n}/1000$	159	100000 (0.084)	506	97

Tab. 7.5: Comparing the number of iterations required for a relative error $\epsilon = 0.05$ using Entropy function (maximal number of iterations allowed: 100000, in brackets relative error at this limit)

Theoretical Number of Iterations					
n, m	building costs	UFL-EG Entropy	UFL-SDA Entropy	UFL-WDA Entropy	UFL-TSDA Entropy
500	$\sqrt{n}/10$	7107	$1.06*10^7$	$1.05*10^7$	$1.06*10^7$
	$\sqrt{n}/100$	1596	$1.06*10^7$	$1.06*10^7$	$1.06*10^7$
	$\sqrt{n}/1000$	330	$3.37*10^7$	$3.38*10^7$	$3.46*10^7$
1000	$\sqrt{n}/10$	12424	$1.67*10^7$ (0.058)	$2.27*10^7$	$2.32*10^7$
	$\sqrt{n}/100$	2810	$1.59*10^7$ (0.051)	$1.65*10^7$	$1.66*10^7$
	$\sqrt{n}/1000$	620	$1.09*10^7$ (0.079)	$5.57*10^7$	$5.75*10^7$
1500	$\sqrt{n}/10$	17157	$2.17*10^7$ (0.065)	$1.56*10^7$ (0.076)	$3.68*10^7$
	$\sqrt{n}/100$	3844	$1.76*10^7$ (0.054)	$2.06*10^7$	$2.08*10^7$
	$\sqrt{n}/1000$	869	$2.41*10^7$ (0.084)	$6.80*10^7$	$6.80*10^7$

Tab. 7.6: Comparing the theoretical number of iterations required for the relative errors attained in Table 7.5 using Entropy function

with respect to the computed iterations. We plot all values until a relative error of 0.01 is achieved. All figures show the same pattern, namely, that the optimal value is approximated more quickly by the dual values than by the primal values. Again, the methods based on the Euclidean norm perform better than the methods based on the Entropy function, except for the Weighted Dual Average Algorithm (WDA).

We further investigate UFL-EG-Euclidean algorithm for larger instances. We also compared the running time needed by the commercial solver Cplex 11.0 ([Cpl])for solving the same instances to optimality with the running time needed by UFL-EG-Euclidean algorithm to achieve different relative errors. Due to memory requirements of Cplex 11.0, we used a computer with a cpu running at 2.6 GHz and 32G RAM. Nevertheless, we could only use the dual simplex method but not the barrier methods. Table 7.7 shows the corresponding results, with the number of computed iterations in brackets. The advantage of UFL-EG-Euclidean algorithm becomes clear in the larger instances with high building costs. However, with small building cost Cplex 11.0 outperforms the approximation algorithm except when relative errors of 0.05 are sufficient.

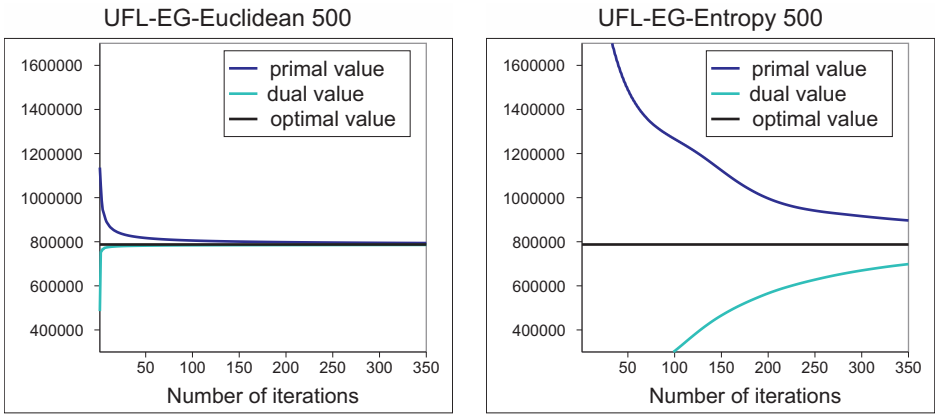


Fig. 7.1: Excessive Gap Method: UFL-EG-Euclidean vs. UFL-EG-Entropy, evolution of primal and dual value

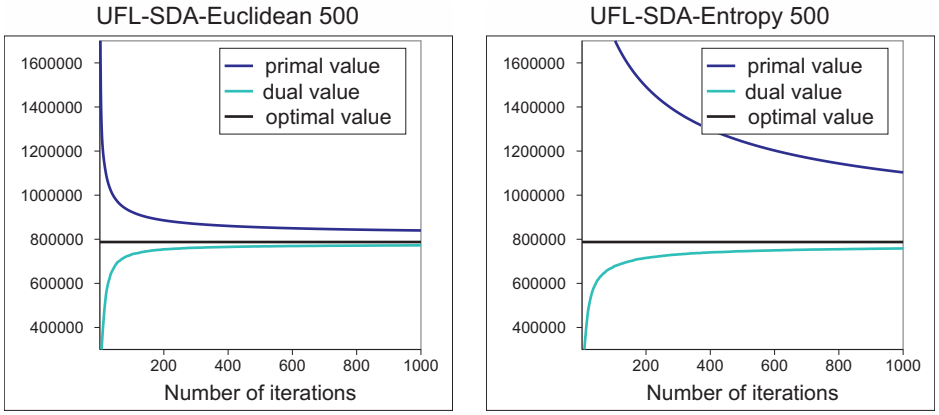


Fig. 7.2: UFL-SDA-Euclidean vs. UFL-SDA-Entropy, evolution of primal and dual value

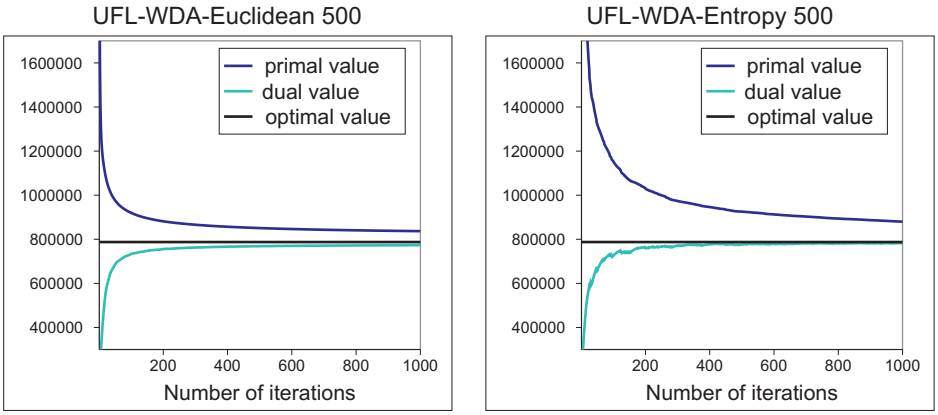


Fig. 7.3: UFL-WDA-Euclidean vs. UFL-WDA-Entropy, evolution of primal and dual value

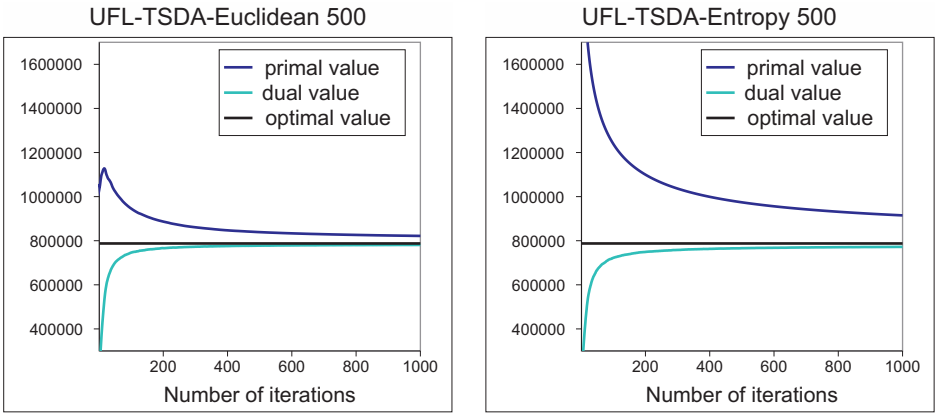


Fig. 7.4: UFL-TSDA-Euclidean vs. UFL-TSDA-Entropy, evolution of primal and dual value

Running Time					
n, m	building costs	Cplex 11.0 (dual simplex)	UFL-EG-Euclidean		
			$\epsilon = 0.05$	$\epsilon = 0.01$	$\epsilon = 0.0075$
2000	$\sqrt{n}/10$	5045s	119s (40)	1435s (483)	2134s (835)
	$\sqrt{n}/100$	328s	131s (40)	977s (316)	1332s (433)
	$\sqrt{n}/1000$	116s	62s (18)	263s (83)	357s (107)
2500	$\sqrt{n}/10$	17821s	184s (38)	1991s (452)	3272s (711)
	$\sqrt{n}/100$	1039s	203s (41)	1613s (344)	2254s (480)
	$\sqrt{n}/1000$	217s	103s (21)	477s (96)	603s (128)
3000	$\sqrt{n}/10$	27645s	230s (38)	2715s (457)	4337s (729)
	$\sqrt{n}/100$	2633s	247s (40)	2273s (360)	3360s (506)
	$\sqrt{n}/1000$	378s	178s (24)	766s (113)	1035s (146)

Tab. 7.7: Cplex 11.0 vs. UFL-EG-Euclidean, comparing running times (in brackets # of completed iterations)

8. Static Traffic Assignment Problem (STAP)

8.1 Motivation and Problem Statement

Starting with the mass production of automobiles in the beginning of the last century, transport analysts and economists and later, mathematicians and computer scientists have considered options for coping with road congestion. From a driver (user) point of view, the highest economic impact of congestion translates into delays. In [War52], Wardrop pointed out in his Second Principle that congestion can only occur if users choose their routes individually in order to optimize their own utility functions, which is usually the case in transportation networks. Thus the main focus of research has been on ways of understanding and possibly relieving congestion in a setting where drivers are free to choose their way. From a game-theoretic point of view, a transportation network is considered at equilibrium when all traffic patterns stabilize (and thus, also the delays) and no driver has any incentive to change his current route (Wardrop's First Principle, [War52]). In this case, we say that the system is at a *user equilibrium* state (UE). The other side of the spectrum is when there is a central decision maker that assigns routes to drivers. In this case, the goal is to collectively optimize the utilization of the network; when this goal is achieved we say that the system is at a *social optimum* state (SO). The existence and uniqueness of either states is a non-trivial question, but they can be guaranteed in certain cases for some mathematical models.

Beckmann et al. were the first to propose and solve a mathematical model to compute both the UE and SO state in [BMW56]. Since then, their model has become standard in transportation networks (e.g., [Nag93], [BMN05] and references therein) and nowadays there are several commercial codes to solve it. In this model, the crucial assumption is the existence of a latency function for each road of the network. As more users use a road, its latency grows, thus making it less attractive. Mathematically the problem usually becomes a minimum cost multicommodity flow problem with non-linear but convex objective in which there are no binding constraints among the flows (the natural road capacity constraints are encoded only through the latency functions).

Parallel to the development of algorithms to solve the underlying mathematical optimization problem, extensions of the Beckmann model with additional constraints have been investigated, which aim at being more realistic. A generalized utility function is considered, where at the same time a latency function and Lagrange multipliers of the additional constraints are used. However, these models have been little studied mainly due to computational issues, see [LP99], and references therein.

More recently Nesterov and De Palma developed an alternative model, [NdP00, NdP03]. In this model, the capacity constraints are kept explicitly in the multicommodity flow problem, and, more crucially the First Wardrop Principle is a consequence of the complementary slackness conditions of a convex optimization problem. In contrast to the Beckmann model, the delays are not computed by means of latency functions, but rather as the Lagrange multipliers of the capacity constraints of a linear multicommodity flow problem.

In the following both models and their main properties are presented. The aim is to provide clarity in the differences of the models both theoretically and practically. However, as they rely on different assumptions, the models cannot be directly compared. Thus, the following measures are considered in order to determine the utility of the models.

- The *price of anarchy*, introduced by Koutsoupias and Papadimitriou in [KP99], defined as the ratio between the total utility at UE and at SO, measures how far the UE is from the best possible use of the network.
- The *Braess paradox*, studied by Braess in [Bra68], describes the counterintuitive phenomenon occurring when adding more resources to a transportation network, e.g., adding a road or a bridge, deteriorates the quality of a UE. Given the significant cost of adding resources to a transportation network, a good model should be able to predict Braess-paradox type of problems before actually making a physical change to the network.

Problem Statement

We consider a traffic network $G = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes, i.e., intersections or zones, and $\mathcal{A}$ is the set of the arcs, i.e., the roads. Each arc $a \in \mathcal{A}$ has a *capacity*, c_a, i.e., the maximal number of cars that can cross the road a during a given period of time, and a *free travel time*, $\bar{t}_a$, the minimal travel time needed to travel through the road a at maximal allowed speed.

The goal of the static traffic assignment problem is to allocate a given set of drivers with fixed origins and destinations on the network in order to attain a *Social Optimum* (SO) state or an *User Equilibrium* (UE) state. At the SO state, the total travel time, i.e., the sum of all drivers' travel time is minimized (second Wardrop principle,

1952). At the UE state, each driver selects his fastest route (first Wardrop principle, 1952). The current state of a traffic network is specified by a *flow*, $s \in \mathbb{R}^{|\mathcal{A}|}$, i.e., where users are driving, and a *travel time*, $t \in \mathbb{R}^{|\mathcal{A}|}$, i.e., how long it takes to travel through the roads of the network.

The set of fixed origins and destinations is denoted by $\mathcal{OD} \subset \mathcal{N} \times \mathcal{N}$. For each fixed origin-destination pair ($\mathcal{OD}$-pair), $k \in \mathcal{OD}$, $d_k > 0$ corresponds to the number of drivers travelling during a given period of time from the origin of k to its destination. We denote by $h^k \in \mathbb{R}^{|\mathcal{A}|}$ the flow of the $\mathcal{OD}$-pair $k \in \mathcal{OD}$ and thus, $s = \sum_{k \in \mathcal{OD}} h^k$.

We assume the number of drivers using a road to be constant during the considered period of time. Thus, we use this problem for studying the network load only during short specific periods of the day, for example peak hours where the average number of drivers using a road can be considered as constant.

8.2 Nesterov & de Palma Model

For Nesterov and de Palma, the capacity c_a of a road $a \in \mathcal{A}$ in a traffic network cannot be violated and as long as there is enough capacity on the roads to allocate to all drivers, there is no slowdown on the roads. At capacity limit, if the flow of drivers is not well managed, congestion and thus delays on the roads may occur. One can characterize the model as a *fluid* model. Assumption 5 resumes the previous remarks.

Assumption 5 ([NdP00, NdP03]) Let (s, t) be a traffic assignment. Then, (s, t) satisfy the following conditions:

- The total flow s_a on an arc $a \in \mathcal{A}$ never exceeds the capacity c_a of this arc, $s_a \leq c_a$.
- Below capacity the travel time t_a on an arc $a \in \mathcal{A}$ is equal to its free travel time $\bar{t}_a$. At capacity limit, it can take any value larger or equal to the free travel time, i.e.,

$$\begin{aligned} \text{if } s_a < c_a \Rightarrow t_a = \bar{t}_a, \\ \text{if } s_a = c_a \Rightarrow t_a \geq \bar{t}_a. \end{aligned}$$

The total travel time is defined as $\sum_{a \in \mathcal{A}} s_a t_a$.

Recall that at SO we assume that the drivers are managed by a central organization, which assigns the drivers on the network in order to minimize the total travel time.

Thus, even for a road with flow at capacity limit, the travel time does not exceed the free flow travel time. For Nesterov and de Palma, computing a traffic assignment at SO is equivalent to solving the following minimum linear cost multicommodity flow problem,

$$\begin{aligned}
\text{(NdP-SO)} \quad \min_{s,h} \quad & \sum_{a\in\mathcal{A}} s_a \bar{t}_a \\
\text{s.t.} \quad & s_a = \sum_{k\in\mathcal{OD}} h_a^k \le c_a \qquad \forall\ a \in \mathcal{A} && (8.1)\\
& Eh^k = \delta^k \qquad \forall\ k \in \mathcal{OD} && (8.2)\\
& h^k \ge 0 \qquad \forall\ k \in \mathcal{OD} && (8.3)
\end{aligned}$$

where E is the node-arc incidence matrix and δ^k is the node demand vector, i.e.,

$$E_{z,a} = \begin{cases} -1 & \text{if node } z \text{ is the tail of arc } a,\\ 1 & \text{if node } z \text{ is the head of arc } a,\\ 0 & \text{otherwise} \end{cases}$$

$$\delta_z^k = \begin{cases} -d_k & \text{if node } z \text{ is the origin of } \mathcal{OD}\text{-pair } k,\\ d_k & \text{if node } z \text{ is the destination of } \mathcal{OD}\text{-pair } k,\\ 0 & \text{otherwise.} \end{cases}$$

The objective function of NdP-SO corresponds to the total travel time since there are no delays and thus the travel time is equal to the free travel time, $t = \bar{t}$. Equation (8.1) ensures that capacity constraints are respected. Equations (8.2) and (8.3) ensure that the demand of each OD-pair is satisfied, i.e., all drivers have to be correctly assigned. We note that this minimization problem models the SO problem in a very natural manner.

Now let us focus on the capacity constraints (8.1). From duality theory the corresponding dual variables are usually considered as the price a user is willing to pay for getting one additional unit of capacity. Nesterov and de Palma interpret it as a penalty, i.e., a delay that the drivers will face for trying to use a road already at capacity limit. We relax (8.1), the capacity restriction, and consider the corresponding Lagrange Dual problem,

$$\max_{u\ge 0} \ \min_{h_k,\ k\in\mathcal{OD}} \left\{ \langle \bar{t}, \sum_{k\in\mathcal{OD}} h^k \rangle + \langle u, \sum_{k\in\mathcal{OD}} h^k - c \rangle \mid Eh^k = \delta^k,\ h^k \ge 0\ \ \forall\ k \in \mathcal{OD} \right\}. \tag{8.4}$$

For fixed u, we observe that the optimization problem (8.4) is decomposable by $k \in \mathcal{OD}$. Then, for each $k \in \mathcal{OD}$, the minimization problem

$$\min_{h_k} \left\{ \langle \bar{t} + u, h^k \rangle - \langle u, c \rangle \mid Eh^k = \delta^k,\ h^k \ge 0 \right\}, \tag{8.5}$$

is a minimum cost flow problem without capacity constraints, where the cost corresponds to the total travel time for assigning drivers of $\mathcal{OD}$-pair k given the travel time $t = \bar{t} + u$. Hence, it is then sufficient to distribute the demand d_k along a shortest path for the commodity k with respect to the given travel time $t = \bar{t} + u$. Having established the duality relation, we observe that for an optimal solution of NdP-SO, s^*, and optimal Lagrange multipliers, u^*, for (8.1) we have

$$\begin{array}{ll} (s^*, \bar{t}) & \text{is a traffic assignment at SO,} \\ (s^*, \bar{t} + u^*) & \text{is a traffic assignment at UE.} \end{array}$$

It is important to note here that the flow is the same at UE and at SO. UE and SO only differ in their travel times, i.e., on the Lagrange multipliers. Thus, from the point of view of traffic management, u^* can be used as an incentive for selfish drivers to reach the SO. One may think of the use of toll (road pricing) or a calibration of maximal allowed speeds and/or flow capacities.

Let us consider again the Lagrange Dual problem (8.4). Denote $\mathcal{P}_k$ the set of all paths between origin and destination of the $\mathcal{OD}$-pair k, and a_P^k the arc incidence vector for each path $P \in \mathcal{P}_k$. Then, the length of the shortest path for the $\mathcal{OD}$-pair k given the travel time t, $T_k(t)$, is defined by

$$T_k(t) = \min_{P \in \mathcal{P}_k} \left\{ \langle a_P^k, t \rangle \right\}. \tag{8.6}$$

Using (8.5) and (8.6), the Lagrange dual of the NdP-SO problem, (8.4), becomes

$$\max_{t \geq \bar{t}} \left\{ \sum_{k \in \mathcal{OD}} d_k T_k(t) - \langle t - \bar{t}, c \rangle \right\}, \tag{8.7}$$

where we replaced $\bar{t} + u$ by t. Theorem 8.1 resumes the previous remarks.

Theorem 8.1 ([NdP00, NdP03])
The arc travel time t^* and the arc flow vector s^* correspond to a traffic assignment at user equilibrium (UE) if and only if t^* is a solution of the following problem

$$\text{(NdP-UE)} \quad \max_{t \geq \bar{t}} \left\{ \sum_{k \in \mathcal{OD}} d_k T_k(t) - \langle t - \bar{t}, c \rangle \right\}, \tag{8.8}$$

and $s^* = c - l^*$, where l^* is a vector of optimal dual multipliers for the inequality constraints $t \geq \bar{t}$ in (8.7).

Note that the flow s^* defined in the previous theorem, i.e., $s^* = c - l^*$ where l^* is the vector of optimal dual multipliers, satisfies the roads capacities. Namely, $c - s^* = l^* \geq 0$ since the feasible dual multipliers are non-negative. The existence

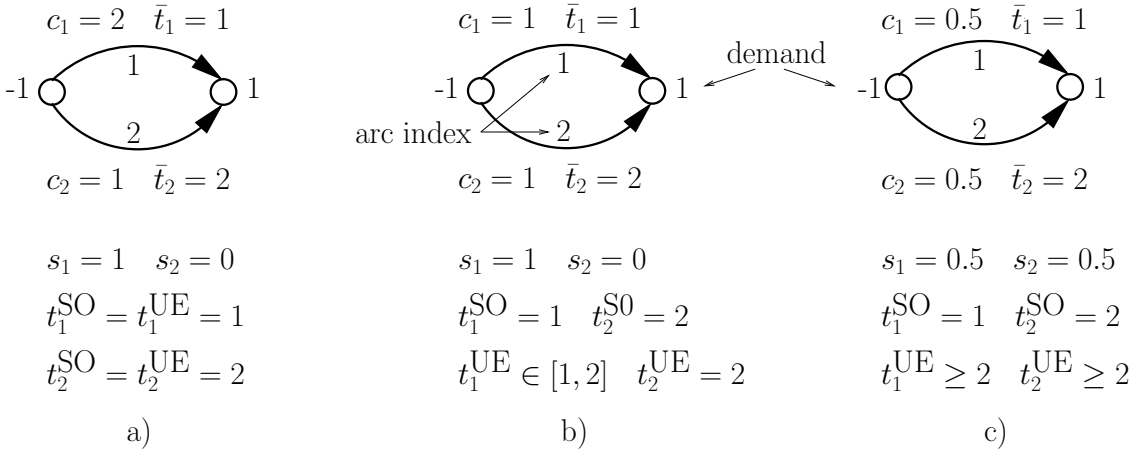


Fig. 8.1: Non uniqueness of delays

and uniqueness of the SO and UE states for the Nesterov & de Palma model is a delicate question. It is obvious that as long as the NdP-SO problem is feasible, the SO state exists but may be non-unique. The existence of UE is more restrictive. If the Lagrange multipliers are not unique the delays and, hence the travel times, cannot be exactly determined. In this case the mathematical model indicates that the distribution of the drivers on the transportation network is unstable with respect to a small change in the capacity of the roads. In the example of Figure 8.1 we have in (a) a situation where delays are unique, and actually equal to zero. In (b), since the upper road is used at its capacity limit, the delay on this road is bounded by the difference in the free travel times of both alternative routes. Finally in (c) we have the flow on both roads at capacity limit and thus, the travel time of both roads have to be equal but the delays may be unbounded.

In case of unboundedness of delays, there is at least one $\mathcal{OD}$-pair having no other alternative route, such that any small decrease of flow capacity make the NdP-SO problem infeasible. One can show that it is enough to find one single road $a \in \mathcal{A}$ for which any reduction of capacity makes the NdP-SO problem infeasible for having unbounded delays and vice versa, see [Rud07].

8.3 Comparison with the Beckmann Model

In this section, we first summarize the basic properties of the Beckmann model and then present some differences between the Nesterov & de Palma and the Beckmann models based on numerical results. For the numerical experiments we consider small networks where the UE and the SO can be solved with high accuracy for both models using commercial solvers. In Chapter 9, numerical results using larger networks and

the methods described in Chapters 4 and 5 are presented. The objective is then to study the numerical performance of the investigated algorithms. Here, we focus on the comparison of the models.

8.3.1 Beckmann Model

In the Beckmann model, one assumes that the travel time for each arc $a \in \mathcal{A}$ only depends on the flow on the arc and it is defined by a latency function $l_a(\cdot)$, which is convex, continuous, non-negative and non-decreasing. Moreover, the capacity constraints are taken into account by choosing $l_a(\cdot)$ such that a violation of the capacity c_a is penalized. Note that this allows solutions where the capacity constraints are violated. The total travel time of a traffic assignment $(s, l(s))$ is defined by $\sum_{a\in\mathcal{A}} l_a(s_a)s_a$. Under these assumptions, the SO is the solution of the following convex optimization problem

$$\begin{aligned} \text{(B-SO)} \qquad \min_{s,h} \quad & \textstyle\sum_{a\in\mathcal{A}} s_a l_a(s_a) & \\ \text{s.t.} \quad & s_a = \textstyle\sum_{k\in\mathcal{OD}} h_a^k & \forall\ a \in \mathcal{A} \\ & Eh^k = \delta^k & \forall\ k \in \mathcal{OD} \\ & h^k \geq 0 & \forall\ k \in \mathcal{OD} \end{aligned}$$

As in the Nesterov & de Palma model, this problem corresponds to a minimum cost multicommodity flow problem having, however, no capacity constraints and an objective function that may be non-linear. Typically used latency functions are the BPR functions ([Bur64]) given by

$$l_a(s_a) := \bar{t}_a \left(1 + \alpha(\frac{s_a}{c_a})^\beta\right), \qquad \alpha, \beta \geq 0, \qquad \forall\ a \in \mathcal{A} \tag{8.9}$$

They penalize the overflow on the roads depending on the values of the parameters α and β, see the example in Figure 8.2. Recall that $\mathcal{P}_k$ denotes the set of all paths for the $\mathcal{OD}$-pair k. We denote by $h_P^k \in \mathbb{R}$ the flow of $\mathcal{OD}$-pair k along path $P \in \mathcal{P}_k$ and note that the total flow s_a on road $a \in \mathcal{A}$ can be then computed as follows

$$s_a = \sum_{k\in\mathcal{OD}} \sum_{\{P\in\mathcal{P}_k, a\in P\}} h_P^k.$$

The travel time of a path P given the total flow s is defined by $l_P(s) = \sum_{a\in P} l_a(s_a)$. The first Wardrop principle in this context can be stated as follows:

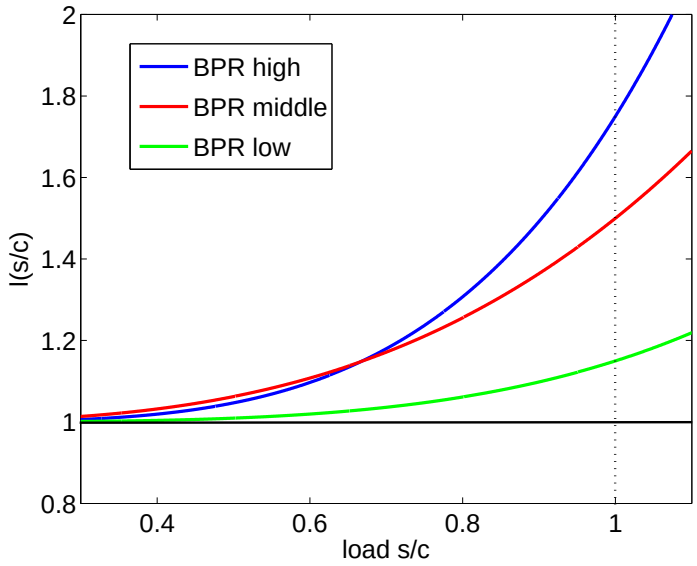


Fig. 8.2: Typical latency functions, BPR, Bureau of Public Road, 1964

Principle (First Wardrop principle, 1952)
The total flow, s^*, satisfies the first Wardrop principle if and only if

$$\forall\ k \in \mathcal{OD} \text{ and } \forall\ P, Q \in \mathcal{P}_k \quad \text{such that } h_P^{k\,*} > 0 \ \Rightarrow \sum_{a \in P} l_a(s_a^*) \leq \sum_{a \in Q} l_a(s_a^*), \quad (8.10)$$

i.e., only the shortest paths are used for each $\mathcal{OD}$-pair.

One can show that condition (8.10) corresponds to the optimality conditions of the following convex optimization problem (see [BMW56] and [Pat94])

$$\begin{array}{lll} \text{(B-UE)} \quad \min_{s,h} & \sum_{a \in \mathcal{A}} \int_0^{s_a} l_a(x)dx & \\ \text{s.t.} & s_a = \sum_{k \in \mathcal{OD}} h_a^k & \forall\ a \in \mathcal{A} \\ & Eh^k = \delta^k & \forall\ k \in \mathcal{OD} \\ & h^k \geq 0 & \forall\ k \in \mathcal{OD} \end{array}$$

Each optimal solution s^* of the optimization problem B-UE corresponds to a traffic assignment, $(s^*, l(s^*))$ at UE for the Beckmann model.

The UE and SO exist always in this setting. Namely, the feasible set

$$\{(h^k)_{k \in \mathcal{OD}} \mid Eh^k = \delta^k, h^k \geq 0 \quad \forall\ k \in \mathcal{OD}\},$$

which is closed but not bounded, can be reduced to the closed and bounded set

$$\{(h^k)_{k\in\mathcal{OD}} \mid Eh^k = \delta^k, h^k \geq 0, h_a^k \quad d_k \ \forall\ a \in \mathcal{A} \quad \forall\ k \in \mathcal{OD}\} \tag{8.11}$$

without affecting the solution set, since both objective functions are continuous, non-negative, and non-decreasing. Then, by Weierstrass' Theorem (Theorem A.2), we have that the minimum of B-SO and the minimum of B-UE are attained in the compact set (8.11). The literature on methods for solving the B-SO and B-UE is vast. See the paper of [BMN05] for an overview. In particular, [BG02] develops an origin-based algorithm efficient for solving the UE problem when highly accurate solutions for small and large-scale instances are required.

Extensions of the Beckmann model, where additional constraints are considered, have also been investigated, e.g. [LP99]. However, they have not been deeply studied. Mathematically, the UE with additional constraints is formulated as follows

$$\begin{aligned}
\text{(B-UEext)} \qquad \min_{s,h} \quad & \textstyle\sum_{a\in\mathcal{A}} \int_0^{s_a} l_a(x)dx & & \\
\text{s.t.} \quad & g_i(s) \leq 0 & \forall i \in \mathcal{I} \quad & \text{(additional constraints)} \\
& s_a = \textstyle\sum_{k\in\mathcal{OD}} \ h_a^k & \forall\ a \in \mathcal{A} & \\
& Eh^k = \delta^k & \forall\ k \in \mathcal{OD} & \\
& h^k \geq 0 & \forall\ k \in \mathcal{OD}, &
\end{aligned}$$

where $\mathcal{I}$ is a subset of indices of arcs, nodes or $\mathcal{OD}$ pairs, and $g_i(s)$ are additional constraints, which are assumed to be convex and continuously differentiable functions. Again, the first Wardrop principle corresponds to the optimality conditions of the convex optimization problem B-UEext. The path's travel time are in this setting generalized as follows

$$t_P(s^*, \zeta^*) = \sum_{a\in P} l_a(s_a^*) + \sum_{i\in\mathcal{I}} \zeta_i^* \left(\sum_{a\in P} \frac{\partial g_i(s^*)}{\partial s_a} \right), \quad \forall\ P \in \mathcal{P}_k,\ \forall\ k \in \mathcal{OD},$$

where s^* is an optimal solution of B-UEext and ζ^* are the Lagrange multipliers corresponding to the additional constraints. Consider for example the special case of flow capacity constraints on the roads, $g_a(s) := s_a - c_a$ for $a \in \mathcal{A}$. For an optimal solution s^* and the optimal Lagrange multipliers ζ_a^* to $g_a(s)$ for $a \in \mathcal{A}$, the generalized travel time for each road $a \in \mathcal{A}$ is given by $t_a(s_a^*) = l_a(s_a^*) + \zeta_a^*$. This corresponds to the Nesterov & de Palma model in the case of constant latency functions, $l_a(s_a) = \bar{t}_a\ \forall\ a \in \mathcal{A}$.

B-UEext is computationally more difficult to solve than B-UE, since the additional constraints are often binding constraints. Moreover, the existence of a UE is a non trivial question as in Nesterov & de Palma model. The non-uniqueness of the dual multipliers implies that the travel times cannot be exactly determined.

8.3.2 Numerical Comparison Based on Small-Scale Networks

The models base the travel times on different assumptions. Therefore, a direct comparison of the travel times is not suitable. Instead, we study the influence of the models' assumptions on the drivers' distribution on the network. Thus, we consider the set of congested roads, the number of paths used per $\mathcal{OD}$-pair, the price of anarchy and the detection of Braess paradoxes.

In this subsection, we consider two small but well investigated instances of the static traffic assignment problem, namely the Sioux Falls and the Anaheim networks, which can be solved with high accuracy (10^{-6}) using commercial solvers. The data concerning these two instances can be found in [BG07] and their main characteristics are shown in Table 8.1.

Instance	Nodes	Roads	$\mathcal{OD}$-pairs
Sioux Falls	24	76	528
Anaheim	416	914	1'405

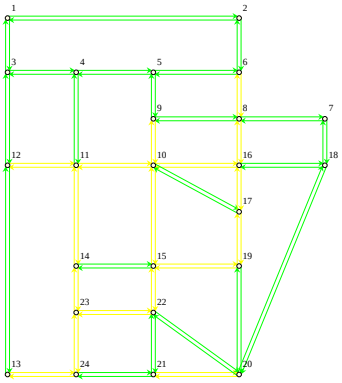
Tab. 8.1: Characteristics of the small networks

For the Beckmann model, three different BPR functions are used, BPR low, middle, and, high. They correspond to the latency function defined at Equation (8.9) for different values of the parameters α and β, see Table 8.2. For the extended Beckmann model, we choose the capacity constraints for the roads as additional constraints.

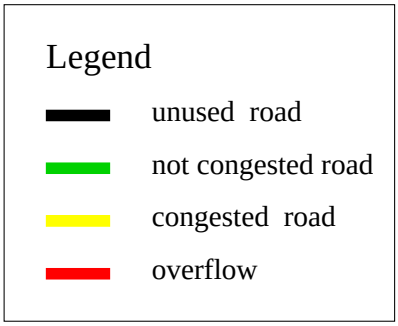
$l_a(s_a) := \bar{t}_a \left(1 + \alpha(\frac{s_a}{c_a})^\beta\right)$		
	α	β
BPR low	0.15	4
BPR middle	0.5	3
BPR high	0.75	4

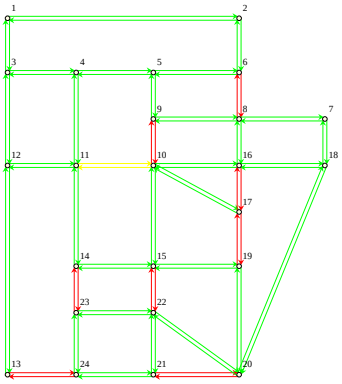
Tab. 8.2: Characteristics of the latency function

First we investigate the set of congested roads at SO and at UE for both models. In Figures 8.3 and 8.4 these sets are depicted for the Sioux Falls network. The roads at capacity limit are drawn in yellow and the roads with overflow are drawn in red. We observe that the set of congested roads provided by the solution of the Nesterov & de Palma model contains the set of congested roads provided by the solutions of

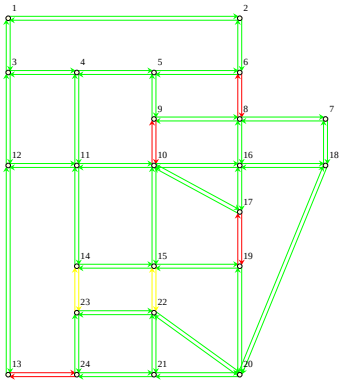


(a) Nesterov & de Palma model

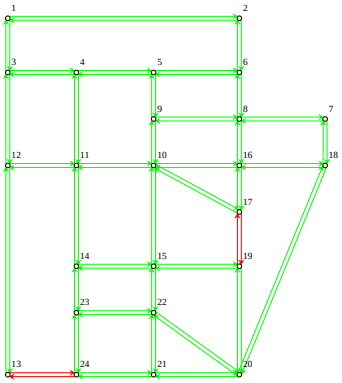




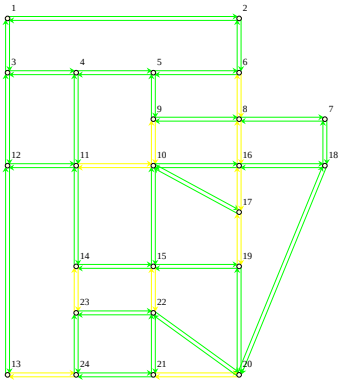
(b) Beckmann model BPR low



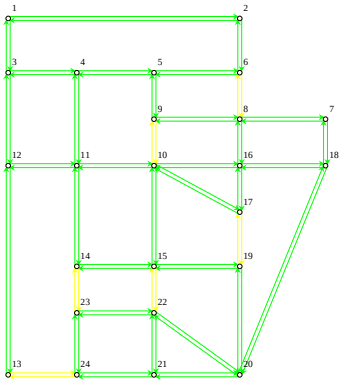
(c) Beckmann model BPR medium



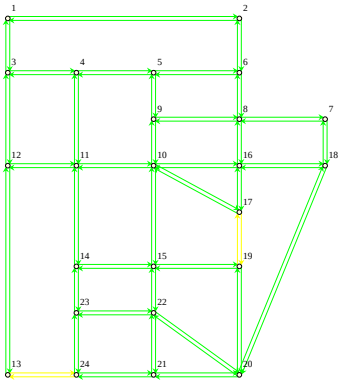
(d) Beckmann model BPR high



(e) Extended Beckmann model BPR low

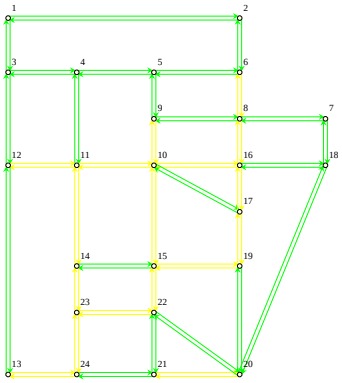


(f) Extended Beckmann model BPR medium

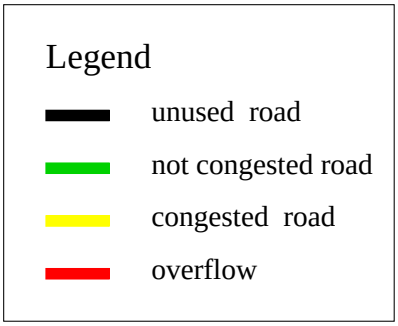


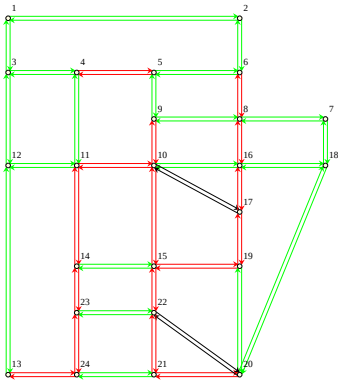
(g) Extended Beckmann model BPR high

Fig. 8.3: Sioux Falls - flow distribution at Social Optimum

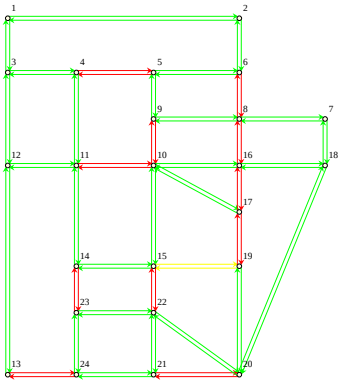


(a) Nesterov & de Palma model

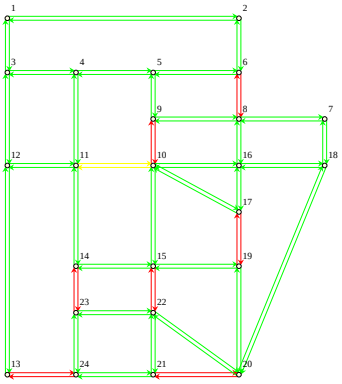




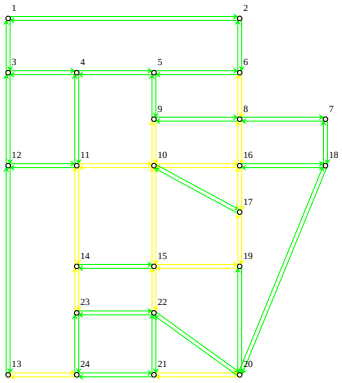
(b) Beckmann model BPR low



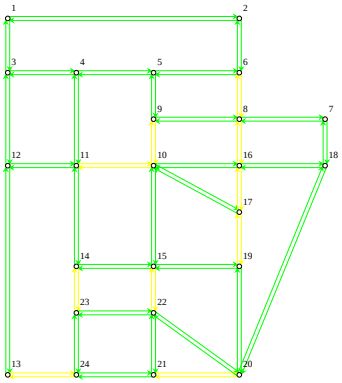
(c) Beckmann model BPR medium



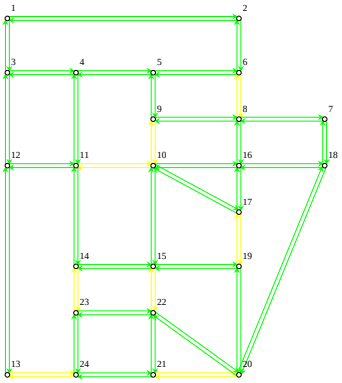
(d) Beckmann model BPR high



(e) Extended Beckmann model BPR low



(f) Extended Beckmann model BPR medium



(g) Extended Beckmann model BPR high

Fig. 8.4: Sioux Falls - flow distribution at User Equilibrium

the considered Beckmann and extended Beckmann models at SO as well as at UE. The same results can be observed for the Anaheim network. Moreover, as expected, the flow distribution delivered by the solutions of the extended Beckmann model is close to the flow distribution obtained by using the Nesterov & de Palma model. Not surprisingly, if the latency function increases the penalty on the overflow, the overflow in the derived traffic assignment is reduced (see Tables 8.3 and 8.4). However, it is interesting to remark that the latency function which best duplicates the solution provided by the Nesterov & de Palma model, both at SO and at UE, is the BPR low function with the standard parameters' values, $\alpha = 0.15$ and $\beta = 4$. These values were defined by the [Bur64].

Model		SO		UE	
		average (%)	std (%)	average (%)	std (%)
Beckmann	BPR low	108.5	2.1	116.4	2.8
	BPR medium	104	1.8	111.2	1.9
	BPR high	102.7	4	109	2.1

Tab. 8.3: Sioux Falls - average overflow at SO and at UE

Model		SO		UE	
		average (%)	std (%)	average (%)	std (%)
Beckmann	BPR low	106.7	5.6	122.7	9.7
	BPR medium	0	0	110	7.9
	BPR high	0	0	103.9	7.7

Tab. 8.4: Anaheim - average overflow at SO and at UE

Model		SO	UE
Nesterov & de Palma		1.057	1.057
Beckmann	BPR low	1.572	1.299
	BPR medium	1.458	1.439
	BPR high	1.545	1.574
Extended Beckmann	BPR low	1.598	1.598
	BPR medium	1.485	1.652
	BPR high	1.545	1.598

Tab. 8.5: Sioux Falls - average number of used paths at SO and at UE

Model		S0	UE
Nesterov & de Palma		1.004	1.004
Beckmann	BPR low	1.357	1.277
	BPR medium	1.443	1.398
	BPR high	1.458	1.440
Extended Beckmann	BPR low	1.322	1.289
	BPR medium	1.443	1.403
	BPR high	1.458	1.436

Tab. 8.6: Anaheim - average number of used paths at SO and at UE

In Tables 8.5 and 8.6, the average numbers of paths used per $\mathcal{OD}$-pair at SO and at UE by both models is summarized. We remark that the flow distribution derived from the Nesterov & de Palma model ($\simeq$ 1 path/$\mathcal{OD}$-pair) uses less paths than the flow distribution derived by the Beckmann models ($\simeq$ 1.4 paths/$\mathcal{OD}$-pair). The addition of capacity constraints on the Beckmann model does not significantly influence the number of paths used.

From our numerical results, we observe that the Nesterov & de Palma model generates traffic assignments where the flow distribution is concentrated as much as possible, whether we look for a SO or a UE state. In contrast, but as expected from the Beckmann model, the more we penalize the overflow the more the flow is spread out over the network.

Price of Anarchy

As already mentioned in the introduction, the price of anarchy was first introduced by [KP99], and it is defined as the ratio between the total utility at UE and at SO. In our context, the total utility corresponds to the total travel time of a traffic assignment (s,t) and is denoted by $U(s,t) = \sum_{a\in\mathcal{A}} s_a t_a$. The price of anarchy is then formulated as follows

$$\text{price of anarchy } = \frac{U(s^{UE},t^{UE})}{U(s^{SO},t^{SO})}, \tag{8.12}$$

where (s^{UE},t^{UE}) corresponds to a traffic assignment at UE and (s^{SO},t^{SO}) to a traffic assignment at SO. In our context, an upper bound on the price of anarchy is a relative measure on how far a UE is off from a best possible network utilization (SO).

The existence of such bounds for the Beckmann model without additional constraints has been intensively investigated in recent years, see [RT00], [RT04],

[CSSM04] and references therein. For example, [Rou03], shows that the price of anarchy is bounded by $O(\frac{d}{\log d})$ when the latency function is a polynomial with non-negative coefficients and degree at most d. Thus, the price of anarchy for the BPR functions is bounded by $O(\frac{\beta}{\log \beta})$ ($= 2$ for $\beta = 4$).

For the Nesterov & de Palma model as well as for the Beckmann model with additional constraints, the existence of such a bound is intrinsically related to the boundedness of the delays, i.e., the Lagrange dual multipliers. We reconsider now the example in Figure 8.1 (c). Using the Nesterov & de Palma model, we get a total travel time at SO of $\frac{3}{2}$. The total travel time at UE is unbounded since any solution distributing half of the flow on each arc with the delays $u_1 = u_2 + 1$, $u_2 \geq 0$, is a traffic assignment at UE. The price of anarchy is then $\frac{2}{3}(1 + u_1)$. The same observation can be made for the extended Beckmann model.

The price of anarchy provides information on the utilization of the network. Table 8.7 summarizes the results of the price of anarchy for the Sioux Falls and the Anaheim networks. We observe that the Nesterov & de Palma model is more pessimistic than the Beckmann model, even if capacity constraints are explicitly considered. The difference in the Anaheim instance is much smaller compared to the Sioux Falls instance, since the first does not correspond to a highly loaded network. In the Anaheim instance, only 0.76 % of the roads are at capacity limit for the Nesterov & de Palma model and 0.66 % for the pessimistic UE given by the Beckmann model.

Model		Sioux Falls	Anaheim
Nesterov & de Palma		1.38	1.008
Beckmann	BPR low	1.026	1.002
	BPR medium	1.039	1.006
	BPR high	1.053	1.007
Extended Beckmann	BPR low	1.003	1.002
	BPR medium	1.016	1.006
	BPR high	1.025	1.008

Tab. 8.7: Price of Anarchy

Braess Paradox

As already mentioned in the introduction, the Braess paradox occurs when adding more resources to a transportation network, for example adding a road or a bridge, deteriorates the quality of a UE. In other words, more resources increase delays for the drivers. [Bra68], was the first to point out this counter-intuitive fact by

exhibiting a simple example using the Beckmann model. This phenomenon can also be interpreted as follows. Suppose we close a road or we increase its free travel time by decreasing the maximal allowed speed in this road. If the utility, i.e., the total travel time, at UE decreases then we also observe a Braess paradox. By Braess roads, we denote roads that worsen the UE under current conditions, i.e. cause a Braess paradox.

In the following we numerically investigate the detection of Braess phenomena on the Sioux Falls network using the three models. For this comparison we increase the free travel time for one road at a time and we look for a decrease in the total travel time at UE. In Table 8.8 the relative improvement of the total travel time is given for the tested roads.

Road	Nesterov & de Palma	Beckmann		
		BPR low	BPR medium	BPR high
15	2.02%	-	-	-
16	2.02%	-	-	-
25	3.52%	-	1.19%	1.36%
26	3.52%	-	1.19%	1.36%

Tab. 8.8: Sioux Falls - Braess Paradox

Since the extended Beckmann model does not detect any Braess paradox, we have omitted it from the table. Road 25 and 26 are seen as Braess roads by the Nesterov & de Palma model as well as by the Beckmann model but only for latency function BPR medium and high. Road 15 and 16 are only considered as Braess roads by the Nesterov & de Palma model. The same observation can be made for the Anaheim network.

The Braess paradox is intrinsically tied to the demands of the $\mathcal{OD}$-pairs. After reducing the demands of all $\mathcal{OD}$-pairs by 30 %, neither the Nesterov & de Palma model nor the Beckmann model detects a Braess paradox. From our numerical results, we note that the detection of the Braess Paradox for the Beckmann model depends as expected, on the choice of the latency function.

8.4 Remarks

The existence of a social optimum (SO) is ensured for both models under minimal requirements. The existence of a user equilibrium (UE) is also easily ensured for the Beckmann model, but it is more restrictive for the Nesterov & de Palma and the

extended Beckmann model. The latter two models use minimum cost multicommodity flow problems with capacity constraints and the corresponding Lagrange dual multipliers for defining a UE. Therefore, the existence of a UE and its uniqueness are intrinsically tied to the existence and uniqueness of the Lagrange dual multipliers.

In the Nesterov & de Palma model, duality theory yields that the flow patterns are equal at SO and at UE and that the travel times differ exactly by the Lagrange dual multipliers. This duality relation provides a natural way to offer an incentive to selfish drivers to reach the SO. On the other hand, with the Beckmann and the extended Beckmann model, traffic managers need to adjust the parameters of the latency function to achieve the same result.

We focused on a computational comparison of the flow distribution at UE and at SO generated by the models. We observed that the set of congested roads in the Nesterov & de Palma model includes the set of congested roads in the Beckmann model. Moreover, the drivers are less spread out in the Nesterov & de Palma model than in the Beckmann model.

In the next chapter we will consider large networks. Our primary goal is to study the numerical performance of the algorithms presented in Chapter 9, nevertheless we will also compare both models.

9. Numerical Results for the Static Traffic Assignment Problem

In this chapter we investigate the numerical performance of the Primal-Dual Subgradient methods and the Excessive Gap method on the Static Traffic Assignment problem using the Nesterov & de Palma model. Recall that we have two mathematical formulations for this problem, the primal formulation corresponding to the Social Optimum (NdP-SO) and the dual formulation corresponding to the User Equilibrium (NdP-UE), see Section 8.2. We apply the Primal-Dual Subgradient methods for solving the dual formulation, i.e., (NdP-UE), and the Excessive Gap method is used for solving the primal formulation (NdP-SO).

We consider instances of STAP from standard networks such as Sioux Falls, Anaheim, or Barcelona, and from real data corresponding to the traffic of the metropolitan region of Zurich. The last network is out of range for commercial LP solvers such as Cplex 11 due to its very large size. We then compare the results obtained by our algorithms with the assignments computed by the commercial solver VISUM [VIS06], which has been developed for approximately solving the Beckmann model.

Before explaining the implementation of the algorithms, we introduce some additional notation and make a few remarks concerning the NdP-SO formulation. The set of roads contains m elements, i.e. $\mid \mathcal{A} \mid = m$, and there are n zones, i.e., $\mid \mathcal{N} \mid = n$. Finally, K $\mathcal{OD}$-pairs exist, i.e., $\mid \mathcal{OD} \mid = K$ and the sum of their demand is denoted by $d_{tot} := \sum_{k \in \mathcal{OD}} d_k$.

The problem NdP-SO corresponds to the traditional minimum cost multicommodity flow problem, where the commodities are the $\mathcal{OD}$-pairs. The literature concerning multicommodity flow problems is very rich due to the large number of applications. Typical examples are message routing in telecommunications, traffic assignment, production scheduling and planning, VLSI design, etc. For an overview on multicommodity flow problems, we refer the reader to the book of Ahuja, Magnanti, and Orlin [AMO93] and the chapters by Minoux and by Lisser and Mahey in [RP06], where multicommodity flow problems in telecommunications are considered. The first book presents more traditional ways for solving this problems, such as Lagrange relaxation, column generation or Dantzig-Wolfe decompositions. In the other book, recent approximation schemes are also presented.

Since we are interested in approximation algorithms, we mention some important results concerning approximation algorithms for the minimum cost multicommodity flow problem. The first approximation scheme was presented by Plotkin, Shmoys, and Tardos in [PST95] with a running time of $\tilde{O}(\epsilon^{-2}K^2m^2)$, where ϵ is the desired accuracy and the notation $\tilde{O}()$ means that logarithm factors are ignored. In [GK96] Grigoriadis and Khachiyan presented an approximation algorithm which improved this result by a factor of $n/(Km)$, i.e., $\tilde{O}(\epsilon^{-2}Kmn)$. A few years later, based on results of Garg and Könemann, [GK98] (different algorithm but same running time as Grigoriadis and Khachian), Fleischer presented in [Fle00] an approximation algorithm with running time $\tilde{O}(\epsilon^{-2}m(m+K))$, which improved the previous bound when a large number of commodities is considered. The first approximation algorithm, whose running time is independent of the number of commodities, $\tilde{O}(\epsilon^{-2}m^2)$, was developed by Karakostas in [Kar02]. His algorithm is also based on the algorithms of Garg and Könemann and Fleischer. In 2005 Villavicencio improved the last result by presenting an approximation algorithm with running time $\tilde{O}(\epsilon^{-2}m)$. For numerical results on some of the above mentioned approximation algorithms, we refer to the paper of Goldberg et al. [GOPS98].

We note that the dependence on ϵ of all algorithms is of order ϵ^2. Using the Excessive Gap method we generate an approximation algorithm whose dependence on ϵ is of order ϵ^{-1}. However, at each iteration our algorithm requires the solution of K minimum quadratic cost flow problems, whereas the others only require the solution of minimum cost flow problems (K or less). The same phenomenon is encountered by Bienstock and Iyengar for the maximum concurrent flow problem in [BI04] as well as in [CE05], where at each iteration a convex quadratic problem has to be solved. Therefore the algorithms with a running time of order ϵ^2 remain more attractive. This is also the case for the approximation algorithm we develop using the Primal-Dual Subgradient methods.

9.1 Primal-Dual Subgradient Algorithms

We apply the Primal-Dual Subgradient methods on the User Equilibrium formulation (NdP-UE) of the Static Traffic Assignment problem,

$$\text{(NdP-UE)} \quad \max_{t \geq \bar{t}} \left\{ \sum_{k \in \mathcal{OD}} d_k T_k(t) - \langle t - \bar{t}, c \rangle \right\} =: UE^* \tag{9.1}$$

where for each $\mathcal{OD}$-pair $k \geq 1$, $T_k(t) = \min_{P \in \mathcal{P}_k} \left\{ \langle a_P^k, t \rangle \right\}$ and $\mathcal{P}_k$ is the set of all possible paths from the origin of the k-th $\mathcal{OD}$-pair to its destination. In the following we denote by UE^* the optimal value of NdP-UE and we use the notation introduced in Chapters 3, 4, and 5.

The objective function, which we want to minimize is defined as

$$f(t) := -\sum_{k\in\mathcal{OD}} d_k T_k(t) + \langle t - \bar{t}, c\rangle \tag{9.2}$$

and the optimization problem is $\min_{t\geq\bar{t}} f(t) = -UE^*$. The feasible space corresponds to the space of travel times on the roads, which are at least equal to the free travel times. However, we cannot estimate the maximal possible travel times in advance. Nevertheless, we assume for the moment that we know a valid upper bound for the travel time, R. This results in a convex and compact feasible space for the travel times $Q := \{t \in \mathbb{R}^m \mid \bar{t} \leq t \leq R\}$. The upper bound R will be iteratively updated during the Primal-Dual Subgradient algorithms by a procedure that we explain later in this section.

The objective function $f(t)$ is not differentiable due to the functions $T_k(t)$, $k \in \mathcal{OD}$. The subdifferential of $f(t)$, $\partial f(t)$, depends on the subdifferential of the functions $T_k(t)$, $\partial T_k(t)$, $k \in \mathcal{OD}$. Namely, $\partial f(t) = -\sum_{k\in\mathcal{OD}} d_k \partial T_k(t) + c$. In order to evaluate these subdifferentials, let us fix $t \in Q$ and $k \geq 1$. Given that the function $T_k(t)$ is defined as a infimum of linear functions in t, its subdifferential is defined as a convex combination of the gradients of the active linear functions, i.e., $\xi^k \in \partial T_k(t)$ if and only if $\xi^k \in \text{conv}\{a_p \mid p \in I_k(t)\}$ with $I_k(t) := \{p \in \mathcal{P}_k \mid T_k(t) = \langle a_p, t\rangle\}$. Therefore, to compute a subgradient of $f(t)$ at each step of the algorithm, it is sufficient to find one path incidence vector in each $I_k(t)$, i.e., to evaluate one shortest path for each $\mathcal{OD}$-pair with respect to the travel time t, for which we apply Dijkstra's Algorithm, see [AMO93]. The latter algorithm computes the shortest paths from one given origin to all other nodes of the network with a running time of $O(m + n\log n)$. Thus, the total running time for computing a subgradient of $f(t)$ for fixed t is $O(K(m + n\log n))$, see Algorithm 17.

In order to specify which dual problem we consider, we rewrite the objective function as follows,

$$\begin{aligned}
f(t) &= -\sum_{k\in\mathcal{OD}} d_k T_k(t) + \langle c, t - \bar{t}\rangle \\
&= -\sum_{k\in\mathcal{OD}} d_k \min_{p\in\mathcal{P}_k} \langle a_p, t\rangle + \langle c, t - \bar{t}\rangle \\
&= \sum_{k\in\mathcal{OD}} d_k \max_{u^k\in\Delta_{|\mathcal{P}_k|}} \left(\sum_{p\in\mathcal{P}_k} \langle -u_p^k a_p, t\rangle\right) + \langle c, t - \bar{t}\rangle \\
&= \max_{u\in\triangle} \langle c - \sum_{k\in\mathcal{OD}} d_k u_p^k a_p, t\rangle - \langle c, \bar{t}\rangle
\end{aligned}$$

where $\triangle = \Delta_{|\mathcal{P}_1|} \times \cdots \times \Delta_{|\mathcal{P}_K|}$. Then,

$$\min_{t\in Q} f(t) = \max_{u\in\triangle} \psi(u),$$

with

$$\psi(u) = \min_{t \in Q} \langle c - \sum_{k \in \mathcal{OD}} d_k u_p^k a_p, t \rangle - \langle c, \bar{t} \rangle, \tag{9.3}$$

defining the considered dual function. As explained in Chapter 4, we can simultaneously compute a subgradient and a dual feasible solution at each step of the Primal-Dual Subgradient algorithms. Namely, for each $k \in \mathcal{OD}$ the convex combination defining $\xi^k \in \partial T_k(t)$ is in $\Delta_{|\mathcal{P}_k|}$. The product of these convex combinations gives a dual feasible solution. Recall that during the algorithm, just one shortest path is computed per $\mathcal{OD}$-pair. Then, for each $k \in \mathcal{OD}$, the corresponding part of the computed dual solution will have zeros as components everywhere expect in the component corresponding to the computed shortest path, which has value 1, see Algorithm 17.

Algorithm 17 NdP-UE:Subgradient

Input: An evaluation point t
Output: $\xi_t \in \partial f(t)$ and the corresponding dual value u

for $k = 1$ to K **do**
compute $p^* = \arg \min_{p \in \mathcal{P}_k} \langle a_p, t \rangle$
set $u_p^k = \begin{cases} 1 & \text{if } p = p^* \\ 0 & \text{otherwise} \end{cases}$ and $\xi_t^k = a_{p^*}$
end for
set $u = (u^1, \ldots, u^K)$ and compute $\xi_t = -\sum_{k \in \mathcal{OD}} d_k \xi_t^k + c$.

Now let us choose a norm for our feasible space as well as a prox-function in order to concretize the Primal-Dual Subgradient algorithms. We endow $\mathbb{R}^m$ with the Euclidean norm and we use the squared Euclidean norm as prox-function. Following the notation of Chapters 3 and 4, we have

$$d_Q(t) := \frac{1}{2}\|t - \bar{t}\|_2^2$$

with $\sigma_Q = 1$ and $\max_{t \in Q} \; d_Q(t) \leq \frac{1}{2}\|R\|_2^2 := D_Q$. We define the prox-function so that its minimizer is the free travel time and, thus, the initial solution for the Primal-Dual Subgradient algorithms is $t_0 = \bar{t}$. Since the dual norm of the Euclidean norm is the Euclidean norm itself, we can bound the norm of the computed subgradient as follows. For fixed t let $\xi_t \in \partial f(t)$, then

$$\|\xi_t\|_2 \leq \| - \sum_{k \in \mathcal{OD}} d_k \mathbf{1} + c\|_2 \leq \sqrt{m} \sum_{k \in \mathcal{OD}} d_k + \|c\|_2 = \sqrt{m} d_{tot} + \|c\|_2 =: L,$$

where $\mathbf{1}$ is the unit vector. We note that the upper bound L depends on the road capacities, c, which usually have large values. In order to remove this dependence,

we consider an upper bound on the norm of subgradients variation. For fixed $t_1, t_2 \in Q$ let $\xi_{t_1} \in \partial f(t_1)$ and $\xi_{t_2} \in \partial f(t_2)$. Then,

$$\begin{aligned}\|\xi_{t_1} - \xi_{t_2}\|_2 &\leq \| - \sum_{k\in\mathcal{OD}} d_k a_{p_k}^{t_1} + c + \sum_{k\in\mathcal{OD}} d_k a_{p_k}^{t_2} - c\|_2 \leq \| \sum_{k\in\mathcal{OD}} d_k (a_{p_k}^{t_2} - a_{p_k}^{t_1})\|_2 \\ &\leq \| \sum_{k\in\mathcal{OD}} d_k \mathbf{1}\|_2 = \sqrt{m}\cdot d_{tot} := M,\end{aligned}$$

where $a_{p_k}^{t_1}$, respectively $a_{p_k}^{t_2}$, is the path incidence vector of one of the shortest paths for $\mathcal{OD}$-pair k given the travel time t_1, respectively t_2. Since $M < L$ we may expect a better convergence from the Truncated Dual Averaging algorithm than the Dual Averaging algorithms.

Now, let us consider the projection we need to calculate at each step of the Primal-Dual Subgradient algorithms. Suppose ζ_{i-1} is the summation of subgradients at step i. For computing the travel time t_i we need to solve $\arg\min_{t\in Q} \{\langle \zeta_{i-1}, t\rangle + \beta_i \frac{1}{2}\|t - \bar{t}\|_2^2\}$ for $\beta_i > 0$. Thus, we have to solve a quadratic minimization problem over a box. Using Lemma 9.1, a solution for the latter optimization problem can be computed in $O(m)$ time.

Lemma 9.1
Let $a, t^{\min}, t^{\max} \in \mathbb{R}^m$ and $b > 0$. Consider the optimization problem

$$\begin{aligned} \textit{minimize} \quad & \sum_{i=1}^{m} a_i t_i + \frac{b}{2}\sum_{i=1}^{m} t_i^2 \\ \textit{subject to} \quad & t^{\min} \leq t \leq t^{\max} \end{aligned} \tag{9.4}$$

and define t^* as follows, $t_i^* := \max\{t_i^{\min}, \min\{-\frac{a_i}{b}, t_i^{\max}\}\} \quad \forall\, i = 1, \ldots, m$. Then, t^* is the optimal solution of problem (9.4).

Proof. The function $t \longmapsto \sum_{i=1}^m a_i t_i + \frac{b}{2}\sum_{i=1}^m t_i^2$ is well defined over $\mathbb{R}^m$ and it is separable by component. For component i we define $g_i(x) = a_i x + \frac{b}{2}x^2$. $g(x)$ is strongly convex, thus its minimum is unique and attained at x_i^* with $g'(x^*) = 0$, i.e., $x_i^* = \frac{-a_i}{b}$. Then,

$$t_i^* = \begin{cases} t_i^{\min} & \text{if } -\frac{a_i}{b} < t_i^{\min} \\ -\frac{a_i}{b} & \text{if } t_i^{\min} \leq -\frac{a_i}{b} \leq t_i^{\max} \\ t_i^{\max} & \text{if } t_i^{\max} < -\frac{a_i}{b} \end{cases}$$

□

In order to evaluate the running time of an iteration of the Primal-Dual Subgradient algorithms for the NdP-UE, we need to know the running time for evaluating the objective function at a given travel time t and the dual function at a given dual

solution u. For evaluating $f(t)$ we need to compute K shortest paths, which takes $O(K(m+n\log n))$. To calculate $\psi(u)$ we need to evaluate the minimum of a linear function over a box, which can be done in $O(m)$. Thus, each iteration of the Primal-Dual Subgradient algorithms takes $O(K(m+n\log n))$.

Suppose the upper bound R on the travel times is correct. The next two theorems resume the convergence rate of the Primal-Dual Subgradient algorithms applied to NdP-UE.

Theorem 9.2
Given $\epsilon > 0$, the Dual Averaging Algorithm, Algorithm 2, using either simple averages sequences (4.12) or weighted averages sequences (4.13) outputs in $O(\frac{K}{\epsilon^2}(m+n\log n)(\sqrt{m}d_{tot}+\|c\|_2)^2\|R\|_2^2)$ time a primal and a dual solution, $t \in Q$ and $u \in \triangle$, such that $f(t)-\psi(u) \leq \epsilon$.

Proof. From Theorem 4.7, we know that if we run the Primal-Dual Subgradient Algorithm 2 using either simple averages sequences (4.12) or weighted averages sequences (4.13) for $(\frac{9L^2}{\epsilon^2}\frac{D_Q}{\sigma_Q}-1)$ iterations, we obtain primal and dual solutions, $t \in Q$ and $u \in \triangle$, such that $f(t)-\psi(u) \leq \epsilon$, $\epsilon > 0$. Thus we need $\frac{9(\sqrt{m}\cdot d_{tot}+\|c\|_2)^2}{\epsilon^2}\frac{\|R\|^2}{2}$ steps to have a primal and a dual solution with an additive gap of ϵ. Since each iteration takes $O(K(m+n\log n))$, the total running time is $O(\frac{K}{\epsilon^2}(m+n\log n)(\sqrt{m}\cdot d_{tot}+\|c\|_2)^2\|R\|_2^2)$. □

Theorem 9.3
Given $\epsilon > 0$, the Truncated Dual Averaging algorithm, Algorithm 3, applied to solve NdP-UE outputs in $O(\frac{Km}{\epsilon^2}(m+n\log n)d_{tot}^2\|R\|_2^2)$ time a primal and a dual solution, $t \in Q$ and $u \in \triangle$, such that $f(t)-\psi(u) \leq \epsilon$.

Proof. From Theorem 4.8, we know that if we run $\frac{8M^2}{\epsilon^2}\frac{D_Q}{\sigma_Q}$ iterations of the Truncated Simple Dual Averaging Algorithm, we obtain a primal and dual solution, $t \in Q$ and $u \in \triangle$, such that $f(x)-\psi(u) \leq \epsilon$, $\epsilon > 0$. Thus, we need $\frac{8m\cdot {d_{tot}}^2}{\epsilon^2}\frac{\|R\|_2^2}{2}$ steps to have a primal and a dual solution with an additive gap of ϵ. Since each iteration takes $O(K(m+n\log n))$, the total running time is $O(\frac{Km}{\epsilon^2}(m+n\log n)\cdot {d_{tot}}^2\|R\|_2^2)$ □

Recall that to correctly define a state of a network we need to know the travel times of the roads as well as the flow on the roads. In the following we will explain how to derive a flow at UE from a solution given by the Primal-Dual Subgradient algorithms. Suppose t^* is an optimal solution of NdP-UE. From the theory in Chapter 8, the flow s^* defined as $s^* := \sum_{k\in\mathcal{OD}} s^{*k}$, $s^{*k} := d_k\partial T_k(t^*)\ \forall\ k \in \mathcal{OD}$, satisfies the UE conditions with t^* and the SO conditions with $\bar{t}$. After N iterations, in addition to a primal solution t_N and a dual solution u_N, the Primal-Dual Subgradient algorithms also deliver a flow $s_N^k = d_k\sum_{p\in\mathcal{P}_k}(u_N^k)_p a_p$ for all $k \in \mathcal{OD}$. Note that we do not consider all paths in the set $\mathcal{P}_k$ but only the shortest paths computed at

each subgradient computation. Moreover, note that such a flow satisfies only the flow balance equations (8.2) but not necessarily the capacity constraints (8.1).

Heuristic for iteratively updating the travel times upper bound R.
Until now, we assumed to know a valid upper bound for the travel times. This bound is important since it affects the value of the averages sequences, which determine the quality of the convergence of the Primal-Dual Subgradient algorithms, see Chapter 4 and Theorems 9.2 and 9.3. In order to be sure that we do not underestimate the possible delays, we must choose a relatively large R. However, this slows the convergence of the algorithms. We apply the following procedure. The algorithm starts with a multiple of the free travel times, i.e., $R = r \cdot \bar{t}$ with $r > 1$. Then, in every iteration, we check if the current travel times solution t is near to the current upper bound R. If for some fixed component i, $R_i - t_i < \theta$, for fixed $\theta > 0$, the value of R_i is doubled and all constants depending on R are updated. Note that this procedure does not affect the dependence on ϵ of the convergence of the algorithm.

Algorithm 18 Update Procedure for Upper Bound R

Input: - An evaluation point t
- Current value of upper bound R
- Distance tolerance $\theta > 0$

Output: Updated upper bound R.

for $i = 1$ to m **do**
set $R_i = \begin{cases} 2R_i & \text{if } R_i - t_i < \theta \\ R_i & \text{otherwise} \end{cases}$
end for

Increasing the algorithms' speed.
Two other important constants in the convergence of the Primal-Dual Subgradient algorithms are the subgradient's norm upper bound L and the upper bound M on the norm of the variation of subgradients. If the estimate of those upper bounds are too large, they slow down the algorithms. In order to improve the algorithms we could run the algorithms with $L/2$, respectively $M/2$. However, convergence of the algorithms is in this case no longer ensured. In order to solve this problem, we consider the following procedure. We first define two new constants $L_{\max} := L$, respectively $M_{\max} = M$ and we set $L = L/\alpha$, respectively $M = M/\alpha$, for $\alpha > 1$. Then, at each iteration the current value of the theoretical absolute gap is computed at the same time as the current absolute gap. If the latter is greater than the theoretical absolute gap, we have an indicator that the upper bounds L or M were underestimated. In this case, the values of L and M are doubled until the current theoretical bound is larger than the current absolute gap, as it should be if the bounds L or M were equal to $L_{\max}$, respectively $M_{\max}$. Each update of the bounds

L or M leads to changes in different parameters of the algorithms. Algorithm 19 corresponds to this procedure, if the Simple Dual Averaging algorithm is used. For the Truncated Simple Dual Averaging we consider Algorithm 20. Note that the Weighted Dual Averaging algorithm does not use the upper bound L.

The values of the constants L and M influence the values of the parameters β_k at iteration k, see Algorithm 2 and 3. Decreasing the value of L or M decreases the value of β_k. When the Algorithm 2 is used with the weighted averages sequences (4.13) the parameter β_k does not dependent on L or M. However, we also decrease it in order to speed up the algorithm. We introduce the dummy value $L = 1$ and apply the same update procedure as for the Simple Dual Averaging algorithm.

Algorithm 19 Update Procedure for Upper Bound L

Input: - Current value of absolute gap, gap $= f(t) - \psi(u)$
- Current value of upper bound, L
- Maximal value of upper bound L, $L_{\max}$
- Current iteration number, i

Output: Updated upper bound, L

theo_gap $= \frac{2\sqrt{2}L}{\sqrt{i+1}}\sqrt{D_Q}$
while gap $>$ theo_gap **do**
  set $L = 2L$
  if $L > L_{\max}$ **then**
    $L = L_{\max}$
  end if
  theo_gap $= \frac{2\sqrt{2}L}{\sqrt{i+1}}\sqrt{D_Q}$
end while

Supposing that we start Simple Dual Averaging algorithm with $L = \frac{L_{max}}{\alpha}$, $\alpha > 1$, respectively the Truncated Simple Dual Averaging algorithm with $M = \frac{M_{max}}{\alpha}$, $\alpha > 1$, then at most $\log \alpha$ runs of the update procedure for upper bound L, respectively M are needed.

Now we have all the elements needed to apply the Primal-Dual Subgradient algorithms to the NdP-UE problem. We change the stopping criterion of Algorithm 2 and 3, i.e., an absolute error sufficiently small, $f(t) - \psi(u) \leq \epsilon,\ \epsilon > 0$, into a relative error. We run the algorithms until a primal solution t and a dual solution u is computed such that

$$\frac{f(t) - \psi(u)}{\psi(u)} \leq \epsilon$$

for a given $\epsilon > 0$. Thus, we consider the following algorithms. Algorithm 21 corresponds to Algorithm 2 using the simple averages sequences (4.12), Algorithm 22 corresponds to Algorithm 2 using the weighted averages sequences (4.13), and

Algorithm 20 Update Procedure for Upper Bound M

Input: - Current value of absolute gap, gap $= f(t) - \psi(u)$
- Current value of upper bound, M
- Maximal value of upper bound M, $M_{\max}$
- Current iteration number, i

Output: Updated upper bound, M

```
theo_gap = (2√2M/√i) √D_Q
while gap > theo_gap do
  set M = 2M
  if M > M_max then
    M = M_max
  end if
  theo_gap = (2√2M/√i) √D_Q
end while
```

Algorithm 23 corresponds to Algorithm 3. Note that both objective functions are not difficult to evaluate. Computing the primal objective function $f(t)$, (9.2), is equivalent to computing a shortest path for each $\mathcal{OD}$-pair. To compute the dual objective function $\psi(u)$, (9.3), we have to compute the minimum of a linear function over a box.

Algorithm 21 Simple Dual Averaging Algorithm for NdP-UE (NdP-UE-SDA)

Input: - Free travel time $\bar{t}$
- The constants D_Q and L
- Upper bound $R = r\bar{t}$, $r > 1$
- Distance tolerance $\theta > 0$
- A relative error $\epsilon > 0$

Output: An approximate primal solution $t \in Q$ and an approximate dual solution $u \in \triangle$ such that $\frac{f(t)-\psi(u)}{\psi(u)} \leq \epsilon$.

set $i = 0$ and $\hat{\beta}_0 = 1$
set $t_0 = \bar{t}$
compute $\xi_0 \in \partial f(t_0)$ and u_0 `NdP-UE:Subgradient`
set $\zeta_0 = \xi_0$
set $t = t_0$ and $u = u_0$.
while $\frac{f(t)-\psi(u)}{\psi(u)} > \epsilon$ **do**
  set $i = i + 1$
  check theoretical absolute gap `L:Update Procedure`
  compute $\hat{\beta}_i = \hat{\beta}_{i-1} + \frac{1}{\hat{\beta}_{i-1}}$ and $\beta_i = \frac{L}{\sqrt{2D_Q}}\hat{\beta}_i$
  compute $t_i = \arg\min_{t \in Q} \{\langle \zeta_{i-1}, t\rangle + \beta_i \frac{1}{2}\|t - \bar{t}\|_2^2\}$
  `NdP-UE:Quadratic Projection on the box` Q
  check upper bound R and update $D_Q = \frac{1}{2}\|R\|_2^2$ if needed
  `R:Update Procedure`
  compute $\xi_i \in \partial f(t_i)$ and u_i `NdP-UE:Subgradient`
  set $\zeta_i = \zeta_{i-1} + \xi_i$
  $t = \frac{1}{i+1}\sum_{l=0}^{i} t_l$ and $u = \frac{1}{i+1}\sum_{l=0}^{i} u_l$ `NdP-UE:Objective functions`
  `evaluation`
end while

Algorithm 22 Weighted Dual Averaging Algorithm for NdP-UE (NdP-UE-WDA)

Input: - Free travel time $\bar{t}$
- The constants D_Q and $L = 1$
- Upper bound $R = r\bar{t}$, $r > 1$
- Distance tolerance $\theta > 0$
- A relative error $\epsilon > 0$

Output: An approximate primal solution $t \in Q$ and an approximate dual solution $u \in \triangle$ such that $\frac{f(t)-\psi(u)}{\psi(u)} \leq \epsilon$.

set $i = 0$ and $\hat{\beta}_0 = 1$
set $t_0 = \bar{t}$
compute $\xi_0 \in \partial f(t_0)$ and u_0 `NdP-UE:Subgradient`
set $\lambda_0 = \frac{1}{\|\xi_0\|_2}$, $\Lambda_0 = \lambda_0$, and $\zeta_0 = \lambda_0 \xi_0$
set $t = t_0$ and $u = u_0$.
while $\frac{f(t)-\psi(u)}{\psi(u)} > \epsilon$ **do**
  set $i = i + 1$
  check theoretical absolute gap `L:Update Procedure`
  compute $\hat{\beta}_i = \hat{\beta}_{i-1} + \frac{1}{\hat{\beta}_{i-1}}$ and $\beta_i = \frac{L}{\sqrt{2D_Q}}\hat{\beta}_i$
  compute $t_i = \arg\min_{t \in Q} \{\langle \zeta_{i-1}, t \rangle + \beta_i \frac{1}{2}\|t - \bar{t}\|_2^2\}$
    `NdP-UE:Quadratic Projection on the box` Q
  check upper bound R and update $D_Q = \frac{1}{2}\|R\|_2^2$ if needed
    `R:Update Procedure`
  compute $\xi_i \in \partial f(t_i)$ and u_i `NdP-UE:Subgradient`
  set $\lambda_i = \frac{1}{\|\xi_i\|_2}$, $\Lambda_i = \Lambda_{i-1} + \lambda_i$, and $\zeta_i = \zeta_{i-1} + \lambda_i \xi_i$
  $t = \frac{1}{\Lambda_i}\sum_{l=0}^{i} \lambda_l t_l$ and $u = \frac{1}{\Lambda_i}\sum_{l=0}^{i} \lambda_l u_l$ `NdP-UE:Objective functions evaluation`
end while

Algorithm 23 Truncated Simple Dual Averaging Algorithm for NdP-UE (NdP-UE-TSDA)

Input: - Free travel time $\bar{t}$
- The constants D_Q and M
- Upper bound $R = r\bar{t}$, $r > 1$
- Distance tolerance $\theta > 0$
- A relative error $\epsilon > 0$

Output: An approximate primal solution $t \in Q$ and an approximate dual solution $u \in \triangle$ such that $\frac{f(t)-\psi(u)}{\psi(u)} \leq \epsilon$.

set $i = 1$ and $\beta_1 = 1$
set $t_0 = \bar{t}$
compute $\xi_0 \in \partial f(t_0)$ and u_0 `NdP-UE:Subgradient`
compute $t_1 = \arg\min_{t \in Q} \{\langle \zeta_0, t \rangle + \beta_1 \frac{1}{2}\|t - \bar{t}\|_2^2\}$
`NdP-UE:Quadratic Projection on the box` Q
compute $\xi_1 \in \partial f(t_1)$ and u_1 `NdP-UE:Subgradient`
set $\zeta_1 = \xi_1$
set $t = t_1$ and $u = u_1$ `NdP-UE:Objective functions evaluation`
while $\frac{f(t)-\psi(u)}{\psi(u)} > \epsilon$ **do**
  set $i = i + 1$
  check theoretical absolute gap `M:Update Procedure`
  compute $t_i = \arg\min_{t \in Q} \{\langle \zeta_{i-1}, t \rangle + \beta_i \frac{1}{2}\|t - \bar{t}\|_2^2\}$
  `NdP-UE:Quadratic Projection on the box` Q
  check upper bound R and update $D_Q = \frac{1}{2}\|R\|_2^2$ if needed
  `R:Update Procedure`
  compute $\beta_i = \frac{M}{\sqrt{2D_Q}}\sqrt{i}$
  compute $\xi_i \in \partial f(t_i)$ and u_i `NdP-UE:Subgradient`
  set $\zeta_i = \zeta_{i-1} + \xi_i$
  $t = \frac{1}{i}\sum_{l=1}^{i} t_l$ and $u = \frac{1}{i}\sum_{l=1}^{i} u_l$ `NdP-UE:Objective functions evaluation`
end while

9.2 Excessive Gap Method

In this section we consider the Social Optimum formulation (NdP-SO) of the Static Traffic Assignment problem.

$$\begin{aligned}
\text{(NdP-SO)} \qquad \text{minimize} \quad & \sum_{a\in\mathcal{A}} s_a \bar{t}_a \\
\text{subject to} \quad & s_a = \sum_{k\in\mathcal{OD}} h_a^k \le c_a \quad \forall\, a \in \mathcal{A} && (9.5)\\
& Eh^k = \delta_k \qquad\qquad \forall\, k \in \mathcal{OD} && (9.6)\\
& h^k \ge 0 \qquad\qquad\quad \forall\, k \in \mathcal{OD}. && (9.7)
\end{aligned}$$

In the following we denote by SO^* the optimal value of the NdP-SO problem. Recall that by strong duality $SO^* = UE^*$.

The NdP-SO problem is a typical minimum cost multicommodity flow problem, where the difficult constraints are the binding constraints, i.e., the capacity constraints (9.5). We start the general procedure described in Chapter 3.3 by relaxing the capacity constraints. The corresponding Lagrange Dual problem is then $\max_{u\ge 0} \psi(u)$, where

$$\psi(u) := \min_h \left\{ \sum_{a\in\mathcal{A}} \sum_{k\in\mathcal{OD}} h_a^k \bar{t}_a + \sum_{a\in\mathcal{A}} u_a \Big(\sum_{k\in\mathcal{OD}} h_a^k - c_a \Big) \;\Big|\; Eh^k = \delta_k, h^k \ge 0 \;\forall\, k \in \mathcal{OD} \right\}.$$

Recall that the dual variables u are interpreted as the delays occurring on the roads in case of congestion. As for the travel times t in the previous section, we cannot estimate the maximal possible delays in advance. Nevertheless, we assume for the moment that we know a valid upper bound for the delays, C. Just as for the travel times, we get a convex and compact feasible space

$$U := \{u \in \mathbb{R}^m \mid 0 \le u \le C\} \subset \mathbb{R}^m.$$

The upper bound C will be iteratively updated during the Excessive Gap method by a similar procedure as the upper bound R for the travel times. This procedure will be explained later in this section.

The primal function is defined as follows

$$f(h) := \max_{u\in U} \left\{ \sum_{a\in\mathcal{A}} \sum_{k\in\mathcal{OD}} h_a^k \bar{t}_a + \sum_{a\in\mathcal{A}} u_a \Big(\sum_{k\in\mathcal{OD}} h_a^k - c_a \Big) \right\} \tag{9.8}$$

and the primal feasible space corresponds to the flow on the $\mathcal{OD}$ pairs, which have to fulfill the flow balance equations (9.6) and be non negative (9.7). This space

is convex but not compact. Thus, we add the following constraints motivated by (9.5),

$$h^k \leq c \qquad \forall\ k \in \mathcal{OD}.$$

Note that a feasible solution of NdP-SO trivially satisfies these constraints. The primal feasible space, which we denote by Π, is now convex and compact and can be considered as a product of K spaces, i.e., $\Pi := \Pi_1 \times \cdots \times \Pi_K \subset \mathbb{R}^{Km}$, where

$$\Pi_k := \{h^k \in \mathbb{R}^m \mid Eh^k = \delta_k,\ 0 \leq h^k \leq c\} \subset \mathbb{R}^m \qquad \forall\ k \in \mathcal{OD}.$$

The dual function is now

$$\psi(u) := \min_{h \in \Pi} \left\{ \sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{OD}} h_a^k \bar{t}_a + \sum_{a \in \mathcal{A}} u_a \Big(\sum_{k \in \mathcal{OD}} h_a^k - c_a \Big) \right\}. \tag{9.9}$$

In order to define the approximation functions for $f(h)$ and $\psi(u)$, $f_{\mu_U}(h)$ respectively $\psi_{\mu_\Pi}(u)$, we endow $\mathbb{R}^{Km}$ and $\mathbb{R}^m$ with the Euclidean norm and we chose as prox-function on Π and U the squared Euclidean norm. Thus,

$$d_\Pi(h) := \frac{1}{2}\|h - h_o\|_2^2 = \frac{1}{2}\sum_{k=1}^{K} \|h^k - h_o^k\|_2^2$$

where $h_o := \arg\min_{h \in \Pi} \|h\|$, i.e., the projection of 0 into Π. The convexity parameter is $\sigma_\Pi = 1$ and the maximum of $d_\Pi(h)$ over Π is bounded by $D_\Pi := \frac{K}{2}\|c\|_2^2$. For the dual space U we have

$$d_U(u) := \frac{1}{2}\|u\|_2^2$$

with convexity parameter $\sigma_U = 1$ and $D_U = \frac{1}{2}\|C\|_2^2$. The prox-functions are then defined as follows. For $\mu_U > 0$ and $\mu_\Pi > 0$,

$$f_{\mu_U}(h) := \sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{OD}} h_a^k \bar{t}_a + \max_{u \in U} \left\{ \sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{OD}} h_a^k u_a - \sum_{a \in \mathcal{A}} u_a c_a - \frac{\mu_U}{2} \sum_{a \in \mathcal{A}} u_a^2 \right\}, \tag{9.10}$$

$$\psi_{\mu_\Pi}(u) := -\sum_{a \in \mathcal{A}} u_a c_a + \sum_{k \mathcal{OD}} \min_{h^k \in \Pi_k} \left\{ \sum_{a \in \mathcal{A}} u_a h_a^k + \sum_{a \in \mathcal{A}} \bar{t}_a h_a^k + \frac{\mu_\Pi}{2}\|h^k - h_0^k\|_2^2 \right\}. \tag{9.11}$$

The operator $B(h, u) := \langle u, Bh \rangle = \sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{OD}} h_a^k u_a$ (see notation of Chapters 3 and 5), corresponds to the following matrix $B = [I_m \dots I_m] \in \mathbb{R}^{m \times Km}$ where $I_m \in \mathbb{R}^{m \times m}$ is the identity matrix. Recall that the Lipschitz constants of the

gradient of $f_{\mu_U}(h)$ and $\psi_{\mu_\Pi}(u)$ depend on the norm of matrix B,

$$\begin{aligned}
\|B\|_2 &= \max_{\|h\|_2\leq 1}\max_{\|u\|_2\leq 1}\sum_{a\in\mathcal{A}}\sum_{k\in\mathcal{OD}} h_a^k u_a \\
&\leq \max_{\|h\|_2\leq 1}\max_{\|u\|_2\leq 1}\left(\sum_{a\in\mathcal{A}}\left(\sum_{k\in\mathcal{OD}} h_a^k\right)^2\right)^{1/2}\left(\sum_{a\in\mathcal{A}} u_a^2\right)^{1/2} \\
&\leq \max_{\|h\|_2\leq 1}\left(\sum_{a\in\mathcal{A}}\left(\sum_{k\in\mathcal{OD}} 1\right)\left(\sum_{k\in\mathcal{OD}}(h_a^k)^2\right)\right)^{1/2} \\
&\leq \sqrt{K}.
\end{aligned}$$

For fixed $h\in\Pi$ and $u\in U$, the gradients are defined as follows

$$(\nabla f_{\mu_U}(h))^k = \bar{t} + u_{\mu_U,h}\ \forall\ k\in\mathcal{OD} \text{ and } \nabla\psi_{\mu_\Pi}(u) = -c + \sum_{k\in\mathcal{OD}} h^k_{\mu_\Pi,u},$$

where

$$u_{\mu_U,h} := \arg\max_{u\in U}\left\{\sum_{a\in\mathcal{A}}\left(\sum_{k\in\mathcal{OD}} h_a^k - c_a\right)u_a - \frac{\mu_U}{2}\sum_{a\in\mathcal{A}} u_a^2\right\} \tag{9.12}$$

and

$$\begin{aligned}
h_{\mu_\Pi,u} &:= \left(h^k_{\mu_\Pi,u}\right)_{k=1,\ldots,K} \qquad (9.13)\\
h^k_{\mu_\Pi,u} &:= \arg\min_{h^k\in\Pi_k}\left\{\sum_{a\in\mathcal{A}}(u_a+\bar{t}_a)h_a^k + \frac{\mu_\Pi}{2}\|h^k - h_0^k\|_2^2\right\} \quad \forall\ k.
\end{aligned}$$

The Lipschitz constants are $L_{f_{\mu_U},\Pi} = \frac{\|B\|_2^2}{\mu_U\sigma_U} = \frac{K}{\mu_U}$ and $L_{\psi_{\mu_\Pi},U} = \frac{\|B\|_2^2}{\mu_\Pi\sigma_\Pi} = \frac{K}{\mu_\Pi}$.

While executing the Excessive Gap method, we need to solve among others the optimization problems (9.12) and (9.13). We note that the optimization problem (9.12) is of the same type as the problem described in the previous section, i.e., the minimization of a quadratic function over a box. Using Lemma 9.1, a solution for the latter optimization problem can be computed in $O(m)$ time. Computing $h_{\mu_\Pi,u}$ is equivalent to computing the K optimization problems described in (9.13). This problem is a Minimum Quadratic Cost Flow problem, which can be solved in polynomial time using Khachian's algorithm ([KTK79],[Kha79]). Since the Minimum Quadratic Cost Flow is interesting per se, we analyze it separately in Chapter 10.

The Gradient Mapping $GM_{f_{\mu_U}}(\tilde{h})$ and $GM_{\psi_{\mu_\Pi}}(\tilde{u})$ at fixed $\tilde{h}\in\Pi$ and $\tilde{u}\in U$ remain to be investigated. Recall that

$$\begin{aligned}
GM_{f_{\mu_U}}(\tilde{h}) &:= \arg\min_{h\in\Pi}\left\{\langle\nabla f_{\mu_U}(\tilde{h}), h-\tilde{h}\rangle + \frac{L_{f_{\mu_U},\Pi}}{2}\|h-\tilde{h}\|_2^2\right\} \\
&= \arg\min_{h\in\Pi}\left\{\sum_{k\in\mathcal{OD}}\left(\langle\bar{t}+u_{\mu_U,\tilde{h}}, h^k-\tilde{h}^k\rangle + \frac{K}{2\mu_u}\|h^k-\tilde{h}^k\|_2^2\right)\right\}
\end{aligned}$$

and

$$\begin{aligned} GM_{\psi_{\mu_\Pi}}(\tilde{u}) &:= \arg \max_{u \in U} \left\{ \langle \nabla \psi_{\mu_\Pi}(\tilde{u}), u - \tilde{u} \rangle - \frac{L_{\psi_{\mu_\Pi}, U}}{2} \|u - \tilde{u}\|_2^2 \right\} \\ &= \arg \max_{u \in U} \left\{ \langle \sum_{k \in \mathcal{OD}} h^k_{\mu_\Pi, \tilde{u}} - c, u - \tilde{u} \rangle - \frac{K}{2\mu_\Pi} \|u - \tilde{u}\|_2^2 \right\}. \end{aligned}$$

We note that for computing $GM_{f_{\mu_U}}(\tilde{h})$ we need to solve K Minimum Quadratic Cost Flow problems, which is a similar problem as computing $h_{\mu_\Pi, u}$ itself for a given $u \in U$, (9.13). Likewise, to compute $GM_{\psi_{\mu_\Pi}}(\tilde{u})$ we need to solve a quadratic minimization problem similar to computing $u_{\mu_U, h}$ for a fixed $h \in \Pi$, (9.12).

Heuristic for updating iteratively the delays upper bound C
Until now we assumed to know a valid upper bound for the delays. As for the Primal-Dual Subgradient algorithm and the upper bound for the travel times R, this bound is important since it affects the value of the initial smoothing factors μ_Π^0 and μ_U^0, which determine the quality of the convergence of the Excessive Gap method, see Chapter 5. We apply the following procedure to iteratively update the upper bound of the delays. The algorithm starts with a multiple of the free travel times, i.e., $C = r \cdot \bar{t}$, $r > 0$. In every iteration, we check if the current delays solution u is near to the current upper bound C. If for some fixed component i, $C_i - t_i < \theta$, for fixed $\theta > 0$, the value of C_i is doubled and all constants depending on C are updated. In particular the smoothing factors are updated as follows. Denote by C_{old} the value of the upper bound C before and by C_{new} after the update. Then, $\mu_{\Pi,\text{new}} = \mu_{\Pi,\text{old}} \frac{\|C_{\text{new}}\|_2}{\|C_{\text{old}}\|_2}$ and $\mu_{U,\text{new}} = \mu_{U,\text{old}} \frac{\|C_{\text{old}}\|_2}{\|C_{\text{new}}\|_2}$.

Algorithm 25 corresponds to the NdP-SO problem including the update procedure for the upper bound of the delays C for the Excessive Gap method (Algorithm 4). Note that both objective functions are not difficult to evaluate. Computing the primal objective $f(h)$, (9.8), is equivalent to computing the maximum of a linear function over a box. To compute the dual objective $\psi(u)$, (9.9), we have to compute for each $\mathcal{OD}$-pair a Minimum Cost Flow problem with non-integer demand.

Algorithm 24 Update Procedure for Upper Bound C

Input: - An evaluation point u
- Current value of upper bound C
- Distance tolerance $\theta > 0$

Output: Updated upper bound C.

for $i = 1$ to m **do**
set $C_i = \begin{cases} 2C_i & \text{if } C_i - u_i < \theta \\ C_i & \text{otherwise} \end{cases}$
end for

Algorithm 25 Excessive Gap Algorithm for NdP-SO (NdP-SO-EG)

Input: - $h_o := \min_{h\in\Pi} d_\Pi(h)$
- Distance tolerance $\theta > 0$
- An bound $C = r\bar{t}, r > 0$
- Initial smoothing factors $\mu_\Pi^0 := 2\frac{\|C\|_2}{\|c\|_2}$ and $\mu_U^0 := K\frac{\|c\|_2}{\|C\|_2}$
- A relative error $\epsilon > 0$

Output: An approximate primal solution $h \in \Pi$ and an approximate dual solution $u \in U$ such that $\frac{f(h)-\psi(u)}{\psi(u)} \leq \epsilon$.

set $i = 0$
compute $h_0 = GM_{f_{\mu_U^0}}(h_o)$ `NdP-SO: Min Quadratic Cost Flow`
compute $u_0 = u_{\mu_\Pi^0, h_o}$ `NdP-SO:Quadratic Projection on the box` U
while $\frac{f(h_i)-\psi(u_i)}{\psi(u_i)} > \epsilon$ **do**
  if i is even **then**
    compute $h_{\mu_U^i, u_i}$ `NdP-SO: Min Quadratic Cost Flow`
    set $\hat{h} = \frac{i+1}{i+3}h_i + \frac{2}{i+3}h_{\mu_U^i, u_i}$
    compute $u_{\mu_\Pi^i, \hat{h}}$ `NdP-SO:Quadratic Projection on the box` U
    set $u_{i+1} = \frac{i+1}{i+3}u_i + \frac{2}{i+3}u_{\mu_\Pi^i, \hat{h}}$
    compute $GM_{f_{\mu_U^i}}(\hat{h})$ `NdP-SO: Min Quadratic Cost Flow`
    set $h_{i+1} = GM_{f_{\mu_U^i}}(\hat{h})$
    set $\mu_\Pi^{i+1} = \frac{i+1}{i+3}\mu_\Pi^i$ and $\mu_U^{i+1} = \mu_U^i$
    check upper bound C and update D_U, μ_U^{i+1}, and μ_Π^{i+1} if needed `C:Update Procedure`
  else
    compute $u_{\mu_\Pi^i, h_i}$ `NdP-SO:Quadratic Projection on the box` U
    set $\hat{u} = \frac{i+1}{i+3}u_i + \frac{2}{i+3}u_{\mu_\Pi^i, h_i}$
    compute $h_{\mu_U^i, \hat{u}}$ `NdP-SO: Min Quadratic Cost Flow`
    set $h_{i+1} = \frac{i+1}{i+3}h_i + \frac{2}{i+3}h_{\mu_U^i, \hat{u}}$
    compute $GM_{\psi_{\mu_\Pi^i}}(\hat{u})$ `NdP-SO:Quadratic Projection on the box` U
    set $u_{i+1} = GM_{\psi_{\mu_\Pi^i}}(\hat{u})$
    set $\mu_\Pi^{i+1} = \mu_\Pi^i$ and $\mu_U^{i+1} = \frac{i+1}{i+3}\mu_U^i$
    check upper bound C and update D_U, μ_U^{i+1}, and μ_Π^{i+1} if needed `C:Update Procedure`
  end if
  set $i = i + 1$
  `NdP-SO:Objective functions evaluation`
end while
set $h = h_i$ and $u = u_i$

9.3 Numerical Results

To compare the algorithms described in the previous sections, i.e., NdP-UE-SDA (Alg. 21), NdP-UE-WDA (Alg. 22), NdP-UE-TSDA (Alg. 23), and NdP-SO-EG (Alg. 25) we ran various simulations testing the parameters of the algorithms. We investigated the relation between the accuracy and the running time, but also the quality of the traffic assignments, e.g., violations of arc capacities or the relation between congested and delayed arcs.

Our pool of problems contains two kinds of instances. The first one consists of standard networks mostly used to test algorithms for solving the Traffic Assignment problem using the Beckmann model, see Section 8.3, which we needed to adapt to the Nesterov & de Palma model. These changes mainly concern the capacities of the arcs. We considered the Sioux Falls instance, the Anaheim instance and the Barcelona instance, which can be downloaded from the website [BG07]. The second kind of instances corresponds to the metropolitan region of Zurich, Switzerland. There are 24 instances, each one describing the traffic during a one hour period of the day. These data were provided by the Federal Office for Territorial Development [Bun05]. All instances are assumed to be feasible. However, we could only compute an optimal solution using the commercial solver Cplex 11.0 ([Cpl]) for the standard instances. All Zurich instances required an amount of computer memory that we did not have. The characteristics of the instances are presented in Table 9.1. Table 9.2 gives an overview of the number of variables in the instances.

Two types of computers have been used for the simulations. For the tests with the standard instances we used a computer with a cpu running at 1GHz and 2G RAM. For the tests with the Zurich instances we used a computer with a cpu running at 2.6GHz and 32G RAM.

Our objective is two-fold. With the tests on the standard instances our focus is on the performance of the algorithms. We tested them against each other and used the optimal solution obtained by Cplex 11.0 as reference solution. With the tests on the Zurich instances, we also tested the performance of the algorithms but at the same time we compared the quality of the traffic assignments with those provided by the commercial software VISUM ([VIS06]), which uses the Beckmann model, see Section 8.3.

Since we are interested in the total travel times at User Equilibrium (UE) and Social Optimum (SO), we present these values in the tables of the numerical results. Primal and dual objective values are neglected since they are related to the total travel times at UE and SO. Note however, that the value of the relative gap between the primal and dual objective functions does not correspond to the relative error between the approximate total travel times at UE, respectively SO, and the reference solution in the following tables.

Name	Arcs	Nodes	$\mathcal{OD}$-pairs	Total demand	Total capacity
Sioux Falls (highly loaded)	76	24	528	49152.78	185000
Anaheim	914	416	1405	99441.95	5511600
Barcelona	2522	1020	7922	184679.56	13745299.36
Zurich 12pm - 1am	14070	6739	193397	7507.77	30045562
Zurich 1am - 2am	14070	6739	150284	5645.29	30045562
Zurich 2am - 3am	14070	6739	186536	7829.39	30045562
Zurich 3am - 4am	14070	6739	217616	12737.84	30045562
Zurich 4am - 5am	14070	6739	377998	39993.77	30045562
Zurich 5am - 6am	14070	6739	514395	176496.92	30045562
Zurich 6am - 7am	14070	6739	570658	216017.67	30045562
Zurich 7am - 8am	14070	6739	526391	128910.50	30045562
Zurich 8am - 9am	14070	6739	506057	98772.35	30045562
Zurich 9am - 10am	14070	6739	508292	100668.31	30045562
Zurich 10am - 11am	14070	6739	520456	141547.35	30045562
Zurich 11am - 12am	14070	6739	522332	154639.18	30045562
Zurich 12am - 1pm	14070	6739	554851	179042.55	30045562
Zurich 1pm - 2pm	14070	6739	532536	146904.05	30045562
Zurich 2pm - 3pm	14070	6739	545516	155177.43	30045562
Zurich 3pm - 4pm	14070	6739	589353	196306.97	30045562
Zurich 4pm - 5pm	14070	6739	618519	235547.11	30045562
Zurich 5pm - 6pm	14070	6739	584180	171376.41	30045562
Zurich 6pm - 7pm	14070	6739	521290	120764.13	30045562
Zurich 7pm - 8pm	14070	6739	450467	69952.95	30045562
Zurich 8pm - 9pm	14070	6739	429427	57537.12	30045562
Zurich 9pm - 10pm	14070	6739	447019	57790.17	30045562
Zurich 10pm - 11pm	14070	6739	397301	40493.03	30045562
Zurich 11pm - 12pm	14070	6739	193397	7507.77	30045562

Tab. 9.1: Networks used in the simulations

Name	Arcs	Nodes	$\mathcal{OD}$-pairs	# variables
Sioux Falls (highly loaded)	76	24	528	$4.01*10^4$
Anaheim	914	416	1405	$1.28*10^6$
Barcelona	2522	1020	7922	$2.00*10^7$
Zurich 2am - 3am	14070	6739	186536	$2.62*10^9$
Zurich 4pm - 5pm	14070	6739	618519	$8.70*10^9$

Tab. 9.2: Total number of variables of the instances used in the simulations

9.3.1 Primal-Dual Subgradients Algorithms

Algorithms' performance and standard instances

The NdP-UE-SDA, NdP-UE-WDA, and NdP-UE-TSDA algorithms have the particularity that they have extra updating procedures that the Primal-Dual Subgradient methods do not have. These are the updating procedure of the upper bound of the primal variables and the updating procedure of the norm of the computed subgradient respectively the norm of the variation of the computed subgradients, see Algorithms 18, 19, and 20.

We first investigated the influence of these procedures on the running times of the algorithms. Table 9.3 resumes the corresponding results. Method 18, in Section 9.1, corresponds to the updating procedure of the upper bound R of the travel time t, the primal variables. In all algorithms, the starting value for R is given by a multiple of the free travel time $\bar{t}$, i.e., $R = r\bar{t}$, $r > 1$. In the second column of Table 9.3, we find the value of r, which leads to the smallest running time. In the fourth column, we have the number of updates of R induced by the choice of r.

We expect that theory overestimates the values of L and M, see Section 9.1. Thus, we set the following values as the maximal values for L, respectively for M,

$$L_{\max} = \sqrt{m} \cdot d_{tot} + \|c\|_2 \quad \text{and} \quad M_{\max} = \sqrt{m} \cdot d_{tot},$$

and use as starting upper bounds $L = \frac{L_{\max}}{l}$, respectively $M = \frac{M_{\max}}{l}$ with $l > 1$. In the third column of Table 9.3, we find the best choice of l, which gives us the starting values of L, respectively M. In the fifth column we have the number of updates of L, respectively M, induced by the choice of l. In the special case of the NdP-UE-WDA algorithm, we set $L_{\max} = 1$ and then $L = \frac{L_{\max}}{l}$, see Section 9.1.

We observe that once the right choices for r and l are made, the algorithms have all the same running time for the same relative gap, with the exception of NdP-UE-TSDA for the Anaheim instance and Barcelona instance. Here we notice that even though the obtained relative gap is the same as for the two other algorithms, NdP-UE-SDA and NdP-UE-WDA, the quality of the solutions is worse. For example, the NdP-UE-TSDA algorithm delivers a traffic assignment at User Equilibrium with no congested arcs but with 86 delayed arcs. For a better quality, we must require a higher relative error, which leads to an increase in running time.

Now let us evaluate the stability of the algorithms with respect to changes in r and l. First, consider Figure 9.1 and Table 9.4.

For the Sioux Falls instance we ran the algorithm NdP-UE-WDA for different values of r and $l = 1$ until a relative error of 0.05 was reached. We notice that the minimum is achieved between $r = 2$ and $r = 4$ for both running time and number of iterations. In particular, we observe a drop in the running time when increasing from $r = 1.75$

Sioux Falls instance, relative gap 0.005							
Algorithm	Best r	Best l	cpu time [s]	# of changes R	# of changes L or M	# of iterations	Total travel time (UE)
NdP-UE-SDA	2	100	2.15	8	1	730	3238.7
NdP-UE-WDA	2	2.7	1.83	8	0	694	3217.74
NdP-UE-TSDA	1.5	65	1.88	13	0	791	3228.44
Cplex 11.0 reference solution	—	—	16	—	—	—	3397.38

Anaheim instance, relative gap 0.01							
Algorithm	Best r	Best l	cpu time [s]	# of changes R	# of changes L or M	# of iterations	Total travel time (UE)
NdP-UE-SDA	1.5	250	1.15	23	45	130	24680.5
NdP-UE-WDA	1.5	3	1.32	18	0	135	24580.2
NdP-UE-TSDA	1.5	40	0.13	7	0	13	24548.8
Cplex 11.0 reference solution	—	—	1276	—	—	—	24923.6

Barcelona instance, relative gap 0.01							
Algorithm	Best r	Best l	cpu time [s]	# of changes R	# of changes L or M	# of iterations	Total travel time (UE)
NdP-UE-SDA	1.8	200	11.75	6	12	202	1239510
NdP-UE-WDA	1.5	30	11.78	5	0	202	1237530
NdP-UE-TSDA	1.5	300	0.14	1	0	2	1260420
Cplex 11.0 reference solution	—	—	334110	—	—	—	1241300

Tab. 9.3: Find best values of upper bounds R, L, and M: NdP-UE-SDA v.s. NdP-UE-WDA v.s. NdP-UE-TSDA

NdP-UE-WDA algorithm, Sioux Falls instance, relative gap 0.005					
r	# of iterations	theoretical # of iterations	cpu time [s]	# of changes R	Total travel time (UE)
1.2	7370	$1.53*10^7$	30.34	22	3191.82
1.4	10621	$6.12*10^7$	41.09	26	3229.34
1.6	12168	$1.38*10^8$	46.83	23	3256.02
1.74	12489	$2.09*10^8$	47.09	19	3255.22
1.75	12537	$2.15*10^8$	49.71	21	3253.3
1.76	6073	$2.21*10^8$	22.92	13	3213.59
1.77	6154	$2.27*10^8$	23.1	16	3215.71
1.8	6399	$2.45*10^8$	24.54	14	3218.07
2	5017	$3.82*10^8$	20.5	12	3209.75
2.2	6017	$5.51*10^8$	22.7	12	3243.69
2.4	3214	$7.49*10^8$	12.84	8	3208.84
2.6	3883	$9.78*10^8$	15.4	10	3230.71
2.8	4251	$1.24*10^9$	17.81	8	3244.47
3	4832	$1.53*10^9$	18.72	7	3255.76
4	4047	$3.46*10^9$	15.99	0	3270.21
5	5964	$6.11*10^9$	25.54	0	3285.48
reference solution	—	—	16	—	3397.38

Tab. 9.4: Varying the starting value of the upper bound $R = r\bar{t}$. Selected results for a 0.005 relative gap—note the jump at 1.75 to 1.76

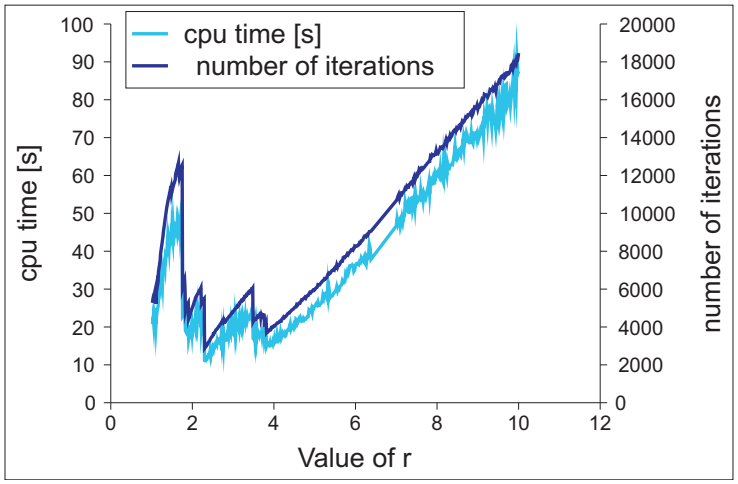


Fig. 9.1: Varying the starting value of the upper bound $R = r\bar{t}$, NdP-UE-WDA algorithm, Sioux Falls instance, relative gap 0.005

to $r = 1.76$. The same kind of behavior is observed with the other algorithms and instances.

Now, consider Table 9.5. For a fixed value of r, we look for the allowed interval for l, that does not increase the running time of the algorithms by more than 30% compared to the best case. The acceptable running times are presented in the fourth column of Table 9.5 and the interval for l is given in the fifth and sixth columns.

We note that in relative values the intervals of l for the NdP-UE-SDA algorithm and the NdP-UE-WDA algorithm are equivalent. Note that for NdP-UE-TSDA, the results are also given in the same table for completeness but they should be interpreted with extreme care since the running time values are too small and measurement errors become important.

A fact of particular importance is that the admissible intervals of l overlap greatly among the networks, which enables us to determine some relatively good values for r and l independent of the networks.

Sioux Falls instance, relative gap 0.005					
Algorithm	r	Min cpu time [s]	Acceptable cpu time[s]	From l	To l
NdP-UE-WDA	2	1.03	1.34	1.6	3
NdP-UE-SDA	2	1.09	1.42	50	100
NdP-UE-TSDA	1.5	0.03	0.04	3.5	10

Anaheim instance, relative gap 0.01					
Algorithm	r	Min cpu time [s]	Acceptable cpu time[s]	From l	To l
NdP-UE-WDA	1.5	1.32	1.72	2	10
NdP-UE-SDA	1.5	1.15	1.5	20	300
NdP-UE-TSDA	1.5	0.13	0.17	30	50

Barcelona instance, relative gap 0.01					
Algorithm	r	Min cpu time [s]	Acceptable cpu time[s]	From l	To l
NdP-UE-WDA	1.5	11.78	15.31	1.5	300
NdP-UE-SDA	1.5	11.75	15.28	1.5	300
NdP-UE-TSDA	1.5	0.14	0.18	1.5	300

Tab. 9.5: Tradeoff between value of l and cpu time.

In order to investigate how the algorithms depend on the network's load, in other words, on the network's congestion, we randomly change the demand of the $\mathcal{OD}$-pairs in the networks. We observe an exponential relation between the total demand and the algorithms' running time. In Figure 9.2, this relaxation is depicted for the Sioux Falls instance and the NdP-UE-WDA algorithm. In this instance 42% of the arcs are congested, thus we randomly decrease the demand of the $\mathcal{OD}$-pairs.

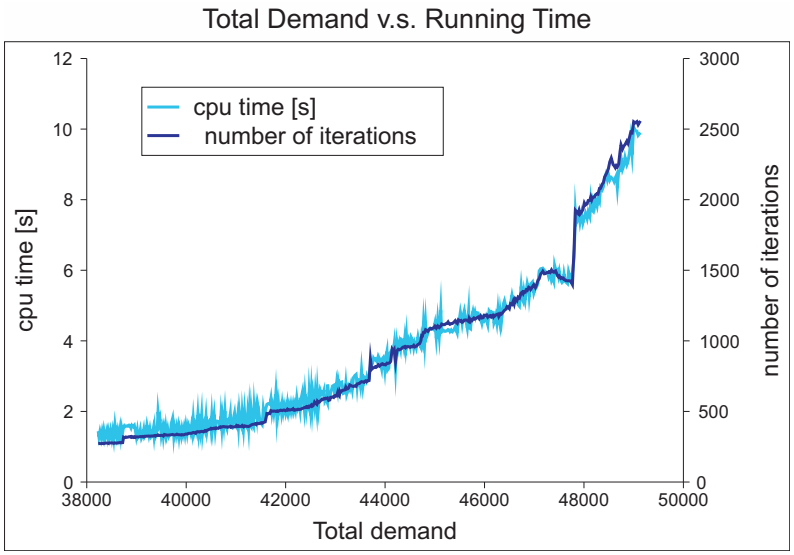


Fig. 9.2: Behavior of cpu time under randomly decrease of the demand—NdP-UE-WDA algorithm, Sioux Falls instance, relative gap 0.005

Until now we have considered the performance of the algorithms from the point of view of running time. In the following we closely investigate the quality of the traffic assignment at UE and at SO delivered by the algorithms. We consider first the total travel time at UE and at SO, and thus the Price of Anarchy, i.e., (total travel time at UE)/(total travel time at SO), see Equation (8.12) in Section 8.3.2. In Tables 9.6 and 9.7 selected results are presented for the NdP-UE-WDA algorithm and the Sioux Falls, respectively Barcelona instances. We investigate the influence of the relative gap on the total travel times. We observe in both tables the same trend. As the relative error decreases the deviation of the total travel time at UE and at SO with respect to the values of the reference solution decreases as expected. Nevertheless, the values obtained for 0.01 relative error are already acceptable. Namely, the deviation of the total travel time at UE from the reference solution is less than 7% for the Sioux Falls and less than 2% for the Barcelona instance. For the total travel time at SO this deviation is negligible for both instances. Note that both total travel times as well as the Price of Anarchy are underestimated by the algorithm. Note also that the small values for the Barcelona instance are normal since there is almost no congestion.

NdP-UE-WDA algorithm, Sioux Falls instance, $r = 2$ and $l = 2$						
Relative error	UE total travel time	deviation from reference solution (%)	SO total travel time	deviation from reference solution (%)	Price of Anarchy	deviation from reference solution (%)
0.0100	3188.58	6.15	2451.45	0.42	1.301	5.75
0.0075	3209.75	5.52	2453.18	0.35	1.308	5.19
0.0050	3244.48	4.50	2455.4	0.26	1.321	4.25
0.0040	3257.75	4.11	2456.48	0.22	1.326	3.90
0.0030	3269.56	3.76	2457.64	0.17	1.330	3.60
0.0020	3280.19	3.45	2458.89	0.12	1.334	3.33
0.0010	3301.51	2.82	2460.26	0.06	1.342	2.76
0.0005	3368.88	0.84	2461.19	0.03	1.369	0.81
reference solution	3397.38	—	2461.82	—	1.38	—

Tab. 9.6: Tradeoff between the relative error and the traffic assignment quality—Total travel time at UE, total travel time at SO, and Price of Anarchy.

An optimal traffic assignment at UE violates no arcs' capacities and the travel time of the routes chosen for all $\mathcal{OD}$-pairs are equal. Moreover, only congested arcs have delays. We next study how well the computed traffic assignment at UE satisfies these requirements in respect to the relative error of the algorithms.

First, we consider Table 9.8, which corresponds to the results obtained using NdP-UE-WDA for the highly loaded Sioux Falls instance. In the fourth column the maximal difference between the travel times of the routes used for the same $\mathcal{OD}$-Matrix is presented. As expected, the difference decreases as the accuracy increases. Then, in the fourth and fifth column we have the evaluation of the violation of the arcs' capacities. First given is the number of arcs where the flow violates the capacity and afterwards the average violation. We consider that an arc's capacity is violated if the flow on the arc exceeds the capacity by more than 1%. Both values, the number of arcs with capacity violation and the average violation, decrease when the accuracy increases. The next column contains the number of congested arcs. An arc is congested if the flow on it is between 0.99 and 1.01 times the capacity. The number of congested arcs increases when the relative error decreases. Namely, the quality of the solution improves since the arc whose flow violates the arc's capacity becomes congested when the relative error decreases. An interesting phenomenon is that the number of delayed arcs decreases with the relative error, however the average delay and the maximal delay increases, see the eighth, ninth, and tenth columns. An arc is delayed if its travel time is at least 1.01 times its free travel

NdP-UE-WDA algorithm, Barcelona instance, $r = 2$ and $l = 3$						
Relative error	UE total travel time	deviation from reference solution (%)	SO total travel time	deviation from reference solution (%)	Price of Anarchy	deviation from reference solution (%)
0.0100	1227940	1.08	1205230	0.15	1.0188	0.93
0.0080	1228300	1.05	1205300	0.15	1.0191	0.9
0.0060	1228790	1.01	1205380	0.14	1.0194	0.87
0.0040	1229550	0.95	1205490	0.13	1.0200	0.82
0.0020	1230960	0.83	1205690	0.11	1.0210	0.72
0.0010	1232460	0.71	1205840	0.10	1.0221	0.61
0.0005	1234210	0.57	1206000	0.09	1.0234	0.48
reference solution	1241300	—	1207050	—	1.0284	—

Tab. 9.7: Tradeoff between the relative error and the traffic assignment quality— Total travel time at UE, total travel time at SO, and Price of Anarchy.

time ($\bar{t}$). The number of congested and delayed arcs converge to similar values. Recall that an arc can be congested at optimum without being delayed, see Section 8.2.

The numerical results presented in Table 9.9 for the Barcelona instance confirm the previous remarks.

Note that solutions delivered by the algorithm for a relative error of 0.01 are already acceptable, i.e. even if the number of arcs with violated arc's capacity is relatively high the maximal violation is very low in respect to the arc's capacity. The solutions delivered for a relative error of 0.001 have less arcs with violated capacities and better ratio between the congested and delayed arcs. However, a large increase of running time is incurred. T he biggest increase resulted for the Sioux Falls instance, where the running time increased from 3.41 seconds to 233.03 seconds.

NdP-UE-WDA algorithm, Sioux Falls instance, $r = 2$ and $l = 2$								
Relative error	# of iterations	cpu time [s]	Maximal $\mathcal{OD}$ travel time difference [s]	# of arcs with violated capacity	Average capacity violation	# of congested arcs	# of delayed arcs	Average delay	Maximal delay
0.0100	733	3.41	0.1	13	0.01	21	31	0.94	2.24
0.0075	1236	5.95	0.11	8	0.01	26	30	1	2.26
0.0050	2546	11.74	0.04	4	0.01	30	30	1.04	2.31
0.0040	3787	17.2	0.04	4	0.01	30	30	1.06	2.38
0.0030	6365	28.02	0.05	0	0	34	30	1.07	2.44
0.0020	13500	59.24	0.01	0	0	32	30	1.09	2.48
0.0010	52128	233.03	0	0	0	32	30	1.11	2.5
0.0005	305524	1363.89	0	0	0	32	30	1.18	2.92
reference solution	—	16	—	—	—	33	30	1.21	3.25

Tab. 9.8: Tradeoff between relative error and traffic assignment quality
— Violation of arc's capacity, arc's congestion, and arc's delay.

NdP-UE-WDA algorithm, Barcelona instance, $r = 2$ and $l = 3$								
Relative error	# of iterations	cpu time [s]	# of arcs with violated capacity	Average capacity violation	Maximal capacity violation	# of congested arcs	# of delayed arcs	Average delay	Maximal delay
0.0100	102	9.03	22	0.05	0.15	18	41	0.27	1.1
0.0080	128	11.56	22	0.05	0.15	18	41	0.27	1.11
0.0060	170	14.99	21	0.04	0.15	18	39	0.28	1.12
0.0040	256	23.51	20	0.04	0.14	20	38	0.29	1.15
0.0020	516	46.9	16	0.04	0.13	21	35	0.32	1.28
0.0010	1048	94.05	14	0.03	0.12	22	35	0.33	1.45
0.0005	2155	193.22	11	0.03	0.11	23	35	0.34	1.56
reference solution	—	334110	—	—	—	35	29	0.45	1.67

Tab. 9.9: Tradeoff between relative error and traffic assignment quality
— Violation of arc's capacity, arc's congestion, and arc's delay.

Zurich instances and NdP-UE-WDA vs. VISUM.

We have tested the Zurich instances on the one hand with the commercial software VISUM, which uses the Beckmann model and on the other hand with the NdP-UE-WDA algorithm applying the Nesterov & de Palma model. Thus, we can make a two-fold comparison, i.e., the running time performance of the two algorithms and the predicted congestions of the two models.

NdP-UE-WDA was calibrated to solve all test instances with a relative accuracy of $\epsilon = 0.002$ and the constants $r = 2$ and $l = 2$. VISUM on the other hand terminates when the calculated travel time relative difference among all routes for the same $\mathcal{OD}$-pair is smaller than 0.05. These tests were conducted on a machine with a cpu running at 1.6 GHz and 512 MB RAM.

For NdP-UE-WDA, two separable running times are listed in Table 9.10. In the first case, only the total flow was calculated, whereas in the second case the algorithm computed individual flows for all $\mathcal{OD}$-pairs including the chosen route.

Table 9.10 presents the running times of the algorithms for all instances. The relation between the demand and running times—second column in case of NdP-UE-WDA—is plotted in Figure 9.3. Note that both running times are scaled in respect to the running time for the period 2am to 3am, i.e., the running times of NdP-UE-WDA are divided by 7'379 and the running times of VISUM are divided by 38.08. Since only 24 instances are available, conclusions should be made with care. However, the plot seems to indicate that while running time increases linearly with the demand for VISUM, the increase is superlinear for NdP-UE-WDA.

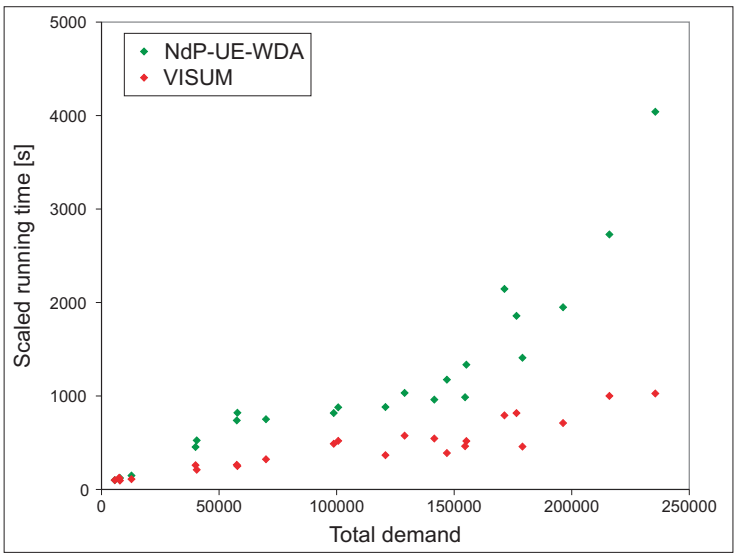
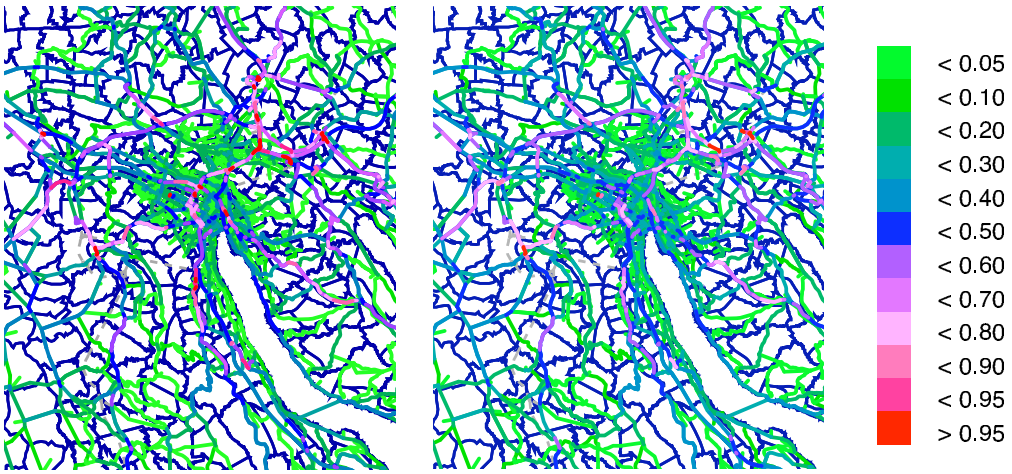


Fig. 9.3: Scaled Running Time — NdP-UE-WDA v.s. VISUM

Hour	Total Demand	NdP-UE-WDA Running time [s] (only total flow)	NdP-UE-WDA Running time [s]	VISUM Running time [s]
12pm - 1am	7507.77	3393	6816	45.33
1am - 2am	5645.29	3263	6010	40.12
2am - 3am	7829.39	3391	7379	38.08
3am - 4am	12737.84	3429	8930	44.27
4am - 5am	39993.77	3840	27294	104.09
5am - 6am	176496.92	4180	111620	327.73
6am - 7am	216017.67	4300	163952	401.36
7am - 8am	128910.50	4230	62063	230.90
8am - 9am	98772.35	4194	49134	196.03
9am - 10am	100668.31	4142	52812	208.50
10am - 11am	141547.35	4176	57754	218.66
11am - 12am	154639.18	4142	59299	185.80
12am - 1pm	179042.55	4303	84671	184.02
1pm - 2pm	146904.05	5192	70581	156.34
2pm - 3pm	155177.43	5271	80216	207.72
3pm - 4pm	196306.97	5422	117155	285.05
4pm - 5pm	235547.11	4724	242833	411.85
5pm - 6pm	171376.41	5365	128929	318.03
6pm - 7pm	120764.13	5121	52996	147.13
7pm - 8pm	69952.95	4995	45164	129.56
8pm - 9pm	57537.12	4199	44391	105.68
9pm - 10pm	57790.17	3961	49313	100.51
10pm - 11pm	40493.03	4107	31547	84.49
11am - 12pm	7507.77	3385	6863	50.19

Tab. 9.10: Running time for NdP-UE-WDA and VISUM for the Zurich instances
1 day = 86'400 s



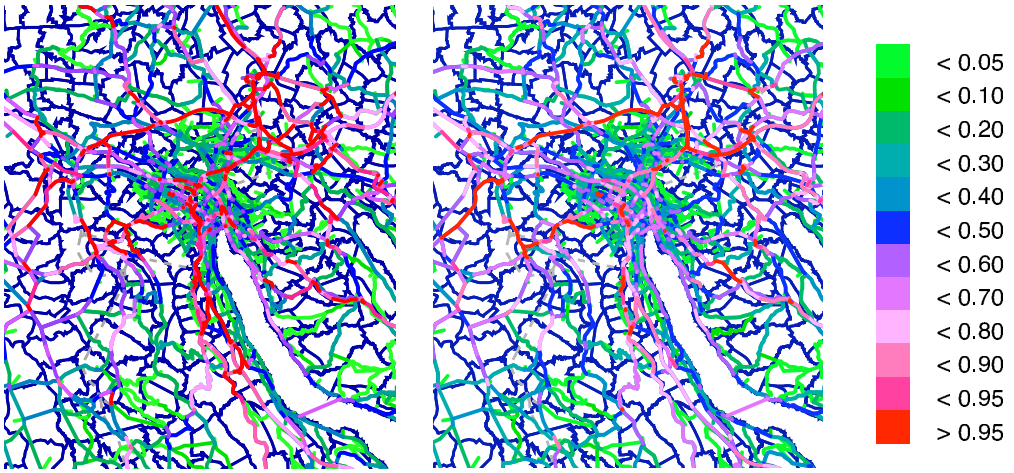
Zurich 12am
NdP-UE-WDA

Zurich 12am
VISUM

Zurich 5pm
NdP-UE-WDA

Zurich 5pm
VISUM

Fig. 9.4: Arcs' load —NdP-UE-WDA v.s. VISUM



Zurich 5pm
NdP-UE-WDA

Zurich 5pm
VISUM

Zurich 10pm
NdP-UE-WDA

Zurich 10pm
VISUM

Fig. 9.5: Arcs' load —NdP-UE-WDA v.s. VISUM

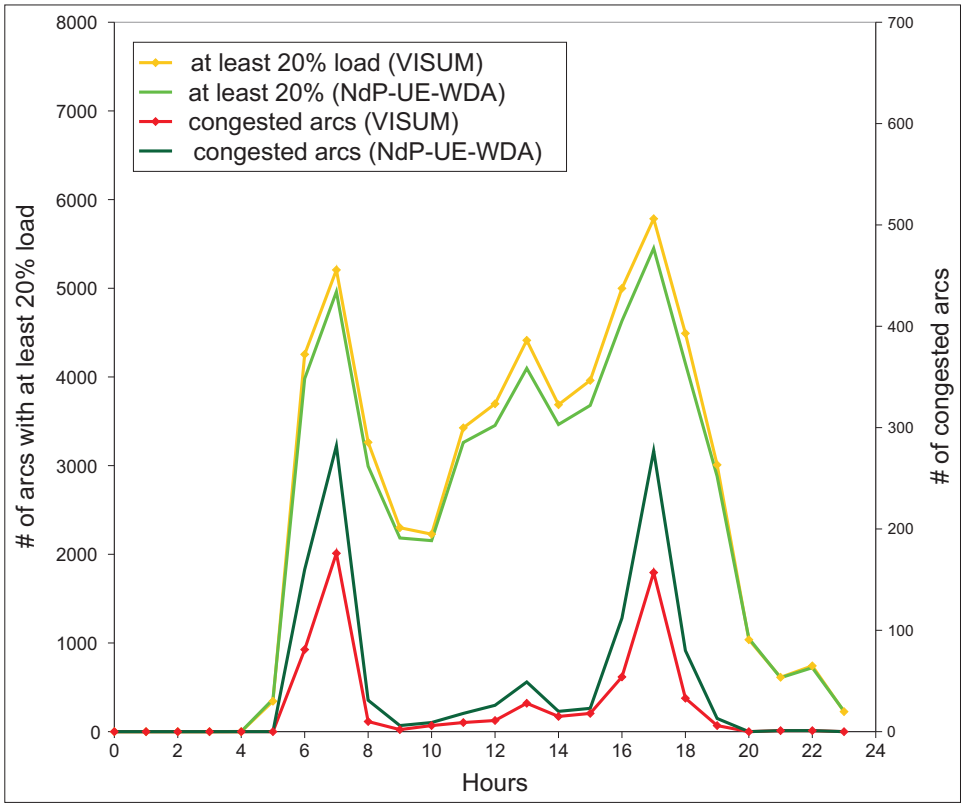


Fig. 9.6: Number of congested arcs and arcs with at least 20% load—NdP-UE-WDA v.s. VISUM

Now, let us compare the flow predictions of the models. Figures 9.4 and 9.5 show the load on the arcs as predicted by the models for some representative hours. Though the figures look very similar, we can see that the Nesterov & de Palma model has more congested arcs during peek hours. In the following we will have a closer look at this result.

Figure 9.6 plots the number of arcs with more than 20% load as well as the number of congested arcs for each model. We can clearly see that the Nesterov & de Palma model results in much more congested arcs while having slightly fewer arcs with more than 20% load compared to the Beckmann model. This can be explained by two effects. First, in the Nesterov & de Palma model the shortest paths will be completely filled before any demand moves to other arcs, which is not the case in the Beckmann model where other routes might receive demand before the shortest route is fully filled. Second, in the Beckmann model capacities are more often violated leading to less demand on other routes. This is shown clearly in Figure 9.7. Finally, let us compare the Price of Anarchy for the two models. Figure 9.8 shows the total travel time ratio of UE to SO for both models. In the Beckmann

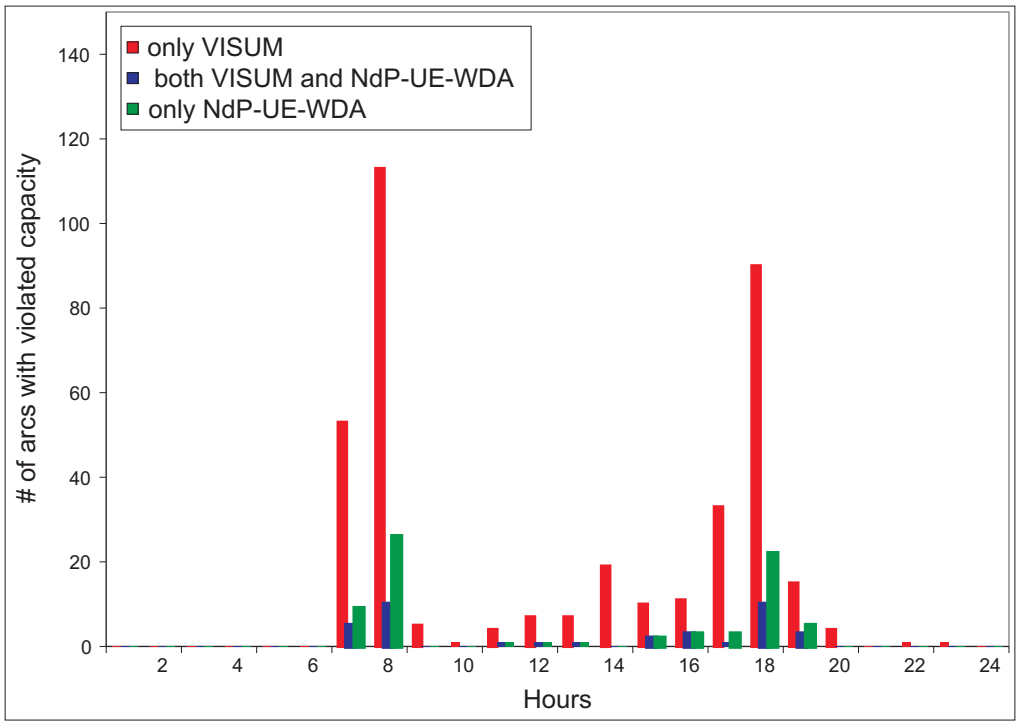


Fig. 9.7: Number of arcs with violated capacity—NdP-UE-WDA v.s. VISUM

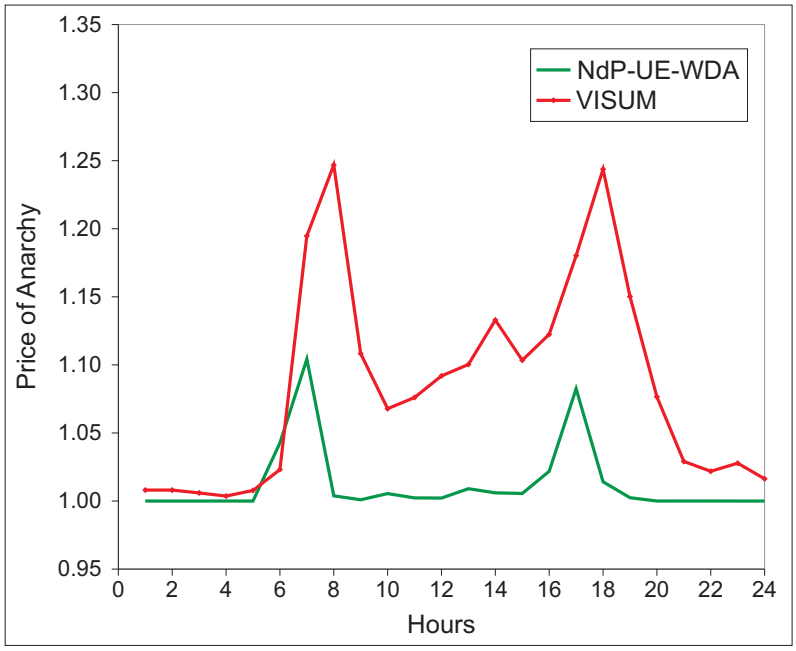


Fig. 9.8: Price of Anarchy—NdP-UE-WDA v.s. VISUM

model the Price of Anarchy is much larger with roughly 25% compared to about 10% for the Nesterov & de Palma model. Moreover, we observe that the Price of Anarchy in the Beckmann model increases with the demand. The Nesterov & de Palma model on the other hand shows the largest Price of Anarchy in the instances one hour before the highest demands.

9.3.2 Excessive Gap Method

Solving the smallest instance, i.e. Sioux Falls, with the Excessive Gap method already took several hours of computing time, which compares extremely bad, see Table 9.11, to the Primal-Dual Subgradient algorithms, which needed a few seconds for this instance, see Table 9.5. This can be explained by the fact that each iteration of the Excessive Gap method is very costly, since as many as three times the number of $\mathcal{OD}$-pairs of Minimum Quadratic Cost Flow (MQCF) problems need to be evaluated. (Chapter 10 presents the techniques used for approximately solving the MQCF problem.) Though each of these subproblems is small and the number of iterations needed for convergence by the Excessive Gap algorithm were smaller than predicted by theory, see Table 9.11, the cost of each iteration prevents the algorithm of being competitive and lead to the decision to not try to solve any larger problem instance. In the following some results and interpretations are given for the SiouxFalls instance.

Sioux Falls instance, relative gap 0.025					
r	# of updates of C	cpu time	# of iterations	theoretical # of iterations	Total travel time (SO)
0.5	2	8h17min	2196	51150	2342.25
1	2	15h17min	4339	104563	2374.82
1.5	2	18h57min	6101	158436	2407.09
reference solution	—	16sec	—	—	2461.82

Tab. 9.11: Find best initial values of upper bound $C = r\bar{t}$ — Excessive Gap using exact oracle (Cplex 11.0)

Table 9.11 shows the influence of the choice of the upper bound C for the delays (see Algorithm 24) on the performance of the Excessive Gap Algorithm 25. These results were obtained using Cplex 11.0 ([Cpl]) in order to solve the MQCF problems within an iteration. Cplex 11.0 uses a Primal-Dual Interior Point method for solving such problems and we interpret the solution delivered at maximal accuracy as an "exact" solution. In Section 10.1, we present the main ideas of Primal-Dual Interior Point methods. Note that the stopping criterion used by Cplex 11.0 does not correspond to the duality gap, see Equation (10.12).

The results show that C is less frequently updated compared to the Primal-Dual Subgradient Algorithms, see Table 9.4. This indicates an underestimation of the delays. Tables 9.12 and 9.13 confirm this. In particular, we note that for the same relative gap, the Primal-Dual Subgradients algorithms deliver solutions closer to the reference solution.

NdP-SO-EG algorithm, Sioux Falls instance $r = 1$ and exact oracle (Cplex 11.0)						
Relative error	UE total travel time	deviation from reference solution (%)	SO total travel time	deviation from reference solution (%)	Price of Anarchy	deviation from reference solution (%)
0.050	2642.06	22.23	2354.06	4.38	1.12	18.84
0.025	2806.85	17.38	2374.82	3.53	1.18	14.49
0.010	2913.04	14.26	2401.05	2.47	1.21	12.32
NdP-UE-WDA[(1)] 0.010	3188.58	6.15	2451.45	0.42	1.301	5.75
reference solution	3397.38	—	2461.82	—	1.38	—

Tab. 9.12: Tradeoff between the relative error and the traffic assignment quality—Total travel time at UE, total travel time at SO, and Price of Anarchy. [(1)] NdP-UE-WDA with $r = 2$ and $l = 2$

NdP-SO-EG algorithm, Sioux Falls instance $r = 1$ and exact oracle (Cplex 11.0)								
Relative error	# of iterations	cpu time [s]	Maximal $\mathcal{OD}$ travel time difference	# of arcs with violated capacity	Average capacity violation	# of congested arcs	# of delayed arcs	Average delay	Maximal delay
0.050	1837	7h44min	0.034	32	0.16	38	62	0.19	4.1
0.025	4339	15h17min	0.017	32	0.16	33	66	0.27	3.25
0.010	22604	46h23min	0.015	30	0.14	33	65	0.31	3.25
NdP-UE-WDA[(1)] 0.010	733	3.41sec	0.1	13	0.01	21	31	0.94	2.24
reference solution	—	16sec	—	—	—	33	30	1.21	3.25

Tab. 9.13: Tradeoff between relative error and traffic assignment quality — Violation of arc's capacity, arc's congestion, and arc's delay.
[(1)] NdP-UE-WDA with $r = 2$ and $l = 2$

Table 9.14 presents results obtained by using approximate instead of exact oracles in the subproblems arising in the Excessive Gap method (MQCF problems). The approximate oracles correspond to an application of a Fast Gradient scheme on a Lagrange relaxation of the MQCF problem, for which details are given in Section 10.2. Additionally, numerical results comparing this method to the Primal-Dual Interior Point method of Cplex 11.0 are presented in Section 10.3.

We consider various values for the stopping criterion of the oracle *stop_fg* = 0.01, 0.001, 0.0001, and 0.00001, see Section 10.3. In case of the Sioux Falls instance, these values correspond to a duality gap of order from 100 to 0.1 respectively. When we chose *stop_fg*= 0.0001, both the results as well as the running time needed were better than those obtained using exact oracles for the same initial upper bound C, see Table 9.15 and 9.16. When a lower accuracy (*stop_fg*= 0.001 or 0.01) was chosen, the number of iterations and running time yet also the quality of the solution decrease also. On the other hand, when we enforced a higher accuracy of (*stop_fg*= 0.00001), the results and the number of iterations needed remained as when *stop_fg*= 0.0001, yet the running time increased due to the additional time needed to solve the subproblems. Remark that, though according to theory an absolute accuracy of the oracle of the order of $O(\epsilon^5)$ is required for an overall accuracy of ϵ, in this piratical instance a much lower accuracy sufficed for the oracle.

Sioux Falls instance, relative gap 0.025 $r = 1$ and approximate oracle (Fast Gradient scheme)				
stopping criterion *stop_fg*	cpu time	# of iterations	theoretical # of iterations	Total travel time (SO)
0.01	6h2min	4331	104456	2369.41
0.001	8h6min	4333	104456	2373.97
0.0001	9h16min	6165	105601	2406.45
0.00001	14h23min	6119	105601	2407.37
reference solution	16sec	—	—	2461.82

Tab. 9.14: Find acceptable value of stopping criterion for the approximate oracles, *stop_fg* — Excessive Gap using $C = \bar{t}$

NdP-SO-EG algorithm, Sioux Falls instance $r = 1$ and approximate oracle ($stop\_fg$= 0.0001)						
Relative error	UE total travel time	deviation from reference solution (%)	SO total travel time	deviation from reference solution (%)	Price of Anarchy	deviation from reference solution (%)
0.050	2642.51	22.22	2354.06	4.34	1.12	18.84
0.025	2990.52	11.98	2406.49	2.22	1.24	10.15
0.010	3216.51	5.32	2450.84	0.45	1.31	5.07
NdP-UE-WDA[(1)] 0.010	3188.58	6.15	2451.45	0.42	1.301	5.75
reference solution	3397.38	—	2461.82	—	1.38	—

Tab. 9.15: Tradeoff between the relative error and the traffic assignment quality— Total travel time at UE, total travel time at SO, and Price of Anarchy — Excessive Gap method with approximate oracle. [(1)] NdP-UE-WDA with $r = 2$ and $l = 2$

NdP-SO-EG algorithm, Sioux Falls instance $r = 1$ and approximate oracle ($stop\_fg = 0.0001$)								
Relative error	# of iterations	cpu time [s]	Maximal $\mathcal{OD}$ travel time difference [s]	# of arcs with violated capacity	Average capacity violation	# of congested arcs	# of delayed arcs	Average delay	Maximal delay
0.050	1837	4h57min	0.034	32	0.16	38	62	0.19	4.1
0.025	6165	9h16min	0.021	26	0.07	35	62	0.39	3.25
0.010	28521	78h43min	0.013	22	0.05	33	46	0.76	3.25
NdP-UE-WDA[1] 0.010	733	3.41sec	0.1	13	0.01	21	31	0.94	2.24
reference solution	—	16sec	—	—	—	33	30	1.21	3.25

Tab. 9.16: Tradeoff between relative error and traffic assignment quality — Violation of arc's capacity, arc's congestion, and arc's delay — Excessive Gap method with approximate oracle. [(1)] NdP-UE-WDA with $r = 2$ and $l = 2$

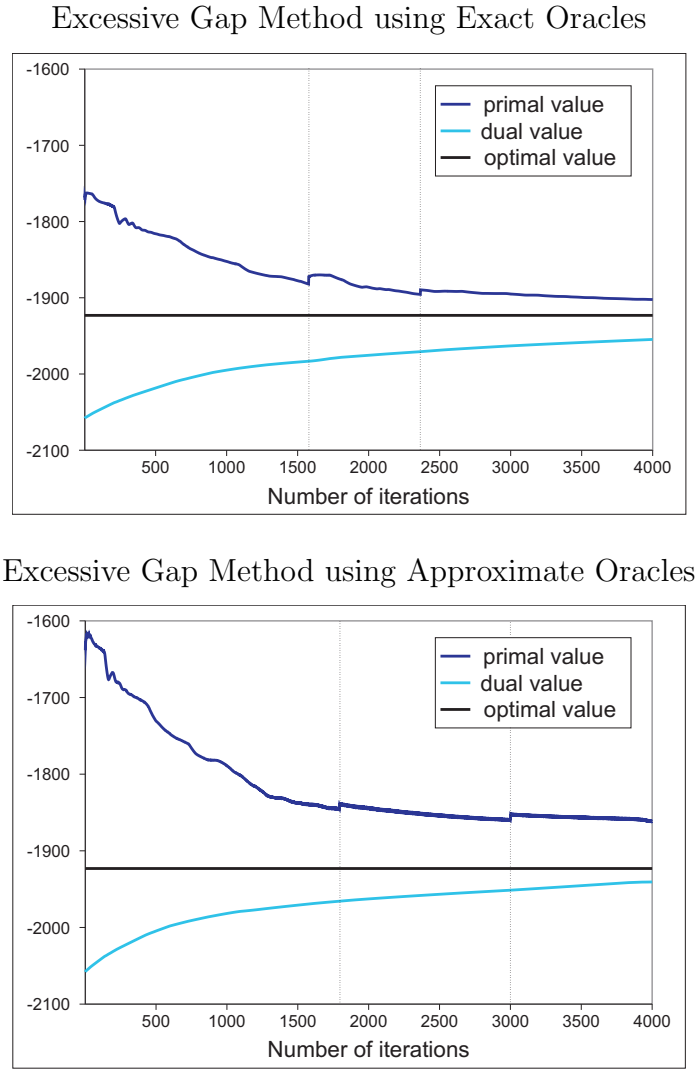


Fig. 9.9: Evolution of primal and dual value

Finally, consider Figure 9.9. As in the case of the linear pogramming relaxation of the Uncapacitated Facility Location problem, we have that the dual value converges faster than the primal value when approximate oracles are used ($stop\_fg = 0.0001$), see Figure 7.1. Yet, for "exact" oracles this is not the case. We notice two jumps in both figures, which correspond to the update of the delays' upper bound C.

9.3.3 Summary

We were able to solve all instances faster than the commercial solver Cplex 11.0 with the Primal-Dual Subgradient algorithms. Note however, that Cplex 11.0 finds

exact solutions while our algorithms deliver approximate solutions within relative errors from 0.01 to 0.001. We found that the algorithms performance was sensitive to the additional updating procedure and therein the chosen values of r and l. However, it was always possible to find values of r and l for each algorithms such that it performed well for all instances. Moreover, once a good choice was made, all algorithms except the truncated simple dual averaging algorithm (NdP-UE-TSDA) showed a similar performance. Note that we did not conduct any further finetuning of the algorithms except for choosing the values for r and l.

Running time increased strongly if a gap of 0.001 instead of 0.01 was required. However, the solution quality regarding arcs with violated capacities and congested or delayed arcs did not improve much, i.e., the additional solution time was not worthwhile.

The Primal-Dual Subgradient algorithm compared badly concerning running time with the commercial software VISUM. Note however, that we solved problems with different complexities. The derived traffic assignments showed as expected that the flow at UE in the Beckmann model is more spread out leading to less congested arcs.

Contrary to the impressive result for the linear programming relaxation of the Uncapacitated Facility Location Problem, the Excessive Gap method performed very poorly for the Traffic Assignment Problem requiring hours to solve small instances. Though, the number of iterations needed remained as small as for the linear programming relaxation of the Uncapacitated Facility Location Problem, the calculation of each iteration was too time-consuming.

We notice that in practice a much lower accuracy is required for the oracle than is predicted by theory.

10. Minimum Separable Quadratic Cost Flow Problem

In this chapter, we consider the minimum separable quadratic cost flow problem. This problem can be solved in polynomial time using Khachian's Algorithm ([Kha79],[KTK79]). However, this algorithm performs poorly in pratice. Thus, starting with Minoux in 1984 ([Min84]) many researchers sought after efficient polynomial algorithms for this problem. For our purpose, we investigate algorithms delivering approximate solutions.

Suppose we have a network with m arcs and n nodes, two of which are special, the source, s, and the sink, t. We denote by $\mathcal{A}$ the set of arcs and by $\mathcal{N}$ the set of nodes. Between the source and the sink a flow with value $\eta > 0$ has to be transported satisfying the arc's capacity, $c \in \mathbb{R}^m$. The target is minimizing the costs, which are incurred by transporting flow over arcs. Here, these costs are separable and quadratic, i.e., for a flow $x \in \mathbb{R}^m$ the corresponding costs are defined as $\langle a, x\rangle + \frac{b}{2}\langle x, x\rangle$, with $a \in \mathbb{R}^m$ and $b \in \mathbb{R}_+$. The minimum separable quadratic cost flow problem is then formulated as follows.

$$\begin{aligned} \text{(MQCF)} \qquad & \text{minimize} && f(x) := a^T x + \frac{b}{2} x^T x \\ & \text{subject to} && Ex = d && (10.1) \\ & && 0 \le x \le c && (10.2) \end{aligned}$$

where $E \in \mathbb{R}^{n\times m}$ corresponds to the node-arc incidence matrix and $d \in \mathbb{R}^m$ to the demand vector, i.e.,

$$E_{z,i} = \begin{cases} -1 & \text{if node } z = t(i), \\ 1 & \text{if node } z = h(i), \\ 0 & \text{otherwise.} \end{cases}$$

$$d_z = \begin{cases} -\eta & \text{if node } z = s \text{ the source}, \\ \eta & \text{if node } z = t \text{ the sink}, \\ 0 & \text{otherwise.} \end{cases}$$

Recall that for an arc $i \in \mathcal{A}$, $t(i)$ denotes its tail and $h(i)$ denotes its head.

In the following, we present two methods for approximately solving the MQCF problem, a Primal-Dual Interior Point method and a Fast Gradient method, which

we then compare numerically. We are interested in methods that deliver good approximation solutions with little numerical effort. The numerical comparisons are presented in the last section of this chapter.

10.1 Primal-Dual Interior Point Method

At the core of most commercial optimization software used for minimizing quadratic convex optimization problems lie Primal-Dual Interior Point methods. The idea of the method is to simplify the problem by replacing the inequality constraints (10.2) with a suitable barrier function added to the objective function $f(x)$, which penalizes all solutions that do not strictly satisfy the inequality constraints (10.2). Namely, we define for $\mu > 0$,

$$\phi_\mu(x) := \frac{1}{\mu} f(x) - \sum_{i=1}^{m} \ln(c_i - x_i) - \sum_{i=1}^{m} \ln(x_i).$$

The function $x \longmapsto -\sum_{i=1}^{m} \ln(c_i - x_i)$ is a barrier function for the set $\{x \in \mathbb{R}^m \mid x \leq c\}$ and the function $x \longmapsto -\sum_{i=1}^{m} \ln(x_i)$ is a barrier function for $\mathbb{R}^n_+$.

We minimize $\phi_\mu(x)$ over the set $\{Ex = d\}$. The corresponding KKT conditions are $\nabla\phi_\mu(x) - E^T y = 0$ and $Ex = d$, where

$$\nabla\phi_\mu(x) = \frac{1}{\mu}(a + bx) + \sum_{i=1}^{m} \frac{1}{c_i - x_i} e_i - \sum_{i=1}^{m} \frac{1}{x_i} e_i,$$

with $e^i \in \mathbb{R}^m$, $e^i_j = 1$ if $j = i$ and $e^i_j = 0$ otherwise for $i = 1, \ldots, m$.

For $\mu > 0$, consider $x(\mu)$ and $\tilde{y}(\mu)$ satisfying the KKT conditions. Then, defining $y(\mu) := \mu\tilde{y}(\mu)$, $s(\mu) := \mu X(\mu)^{-1}\mathbf{1}$ and $z(\mu) := \mu(C - X(\mu))^{-1}\mathbf{1}$ with $X(\mu) := \operatorname{diag}(x(\mu))$ and $C := \operatorname{diag}(c)$, and $\mathbf{1} \in \mathbb{R}^m$, the unit vector, these KKT conditions can be written as follows.

$$a + bx(\mu) + z(\mu) - s(\mu) - E^T y(\mu) = 0 \tag{10.3}$$
$$(C - X(\mu))z(\mu) = \mu\mathbf{1}, \qquad z(\mu) > 0 \tag{10.4}$$
$$X(\mu)s(\mu) = \mu\mathbf{1}, \qquad s(\mu) > 0 \tag{10.5}$$
$$Ex(\mu) = d, \qquad 0 < x(\mu) < c \tag{10.6}$$

Now, suppose we have a solution $(x(\mu), y(\mu), z(\mu), s(\mu))$ satisfying the KKT conditions (10.3), (10.4), (10.5), and (10.6). We first note that $x(\mu)$ is primal feasible, i.e., it is a feasible solution of the MQCF problem. Second, we have that $(y(\mu), z(\mu), s(\mu))$ is a feasible solution of the Lagrange dual of the MQCF problem,

$$\begin{aligned} &\text{maximize} && \psi(y, z, s) := d^T y - c^T z - \frac{1}{2b}\|a - E^T y + z - s\|_2^2 \\ &\text{such that} && s, z \geq 0. \end{aligned}$$

Then, we evaluate the duality gap at $(x(\mu), y(\mu), z(\mu), s(\mu))$.

$$\begin{aligned} f(x(\mu)) &= a^T x(\mu) + \frac{b}{2} x(\mu)^T x(\mu) \\ &= y(\mu)^T E x(\mu) - z(\mu)^T x(\mu) + s(\mu)^T x(\mu) - \frac{b}{2} x(\mu)^T x(\mu) \\ &\quad \text{(using (10.3))} \\ &= y(\mu)^T E x(\mu) - c^T z(\mu) + 2m\mu - \frac{b}{2} x(\mu)^T x(\mu) \\ &\quad \text{(using (10.4), and (10.5))} \\ &= d^T x(\mu) - c^T z(\mu) + 2m\mu - \frac{b}{2} x(\mu)^T x(\mu) \qquad \text{(using (10.6))} \\ &= \psi(y(\mu), z(\mu), s(\mu)) + 2m\mu \qquad \text{(using (10.3))} \end{aligned}$$

Thus, as μ decreases the duality gap (f^* denotes the optimal value of the MQCF problem), the difference

$$f(x(\mu)) - f^* \;\leq\; f(x(\mu)) - \psi(y(\mu), z(\mu), s(\mu)) \;=\; 2m\mu \tag{10.7}$$

decreases.

Starting with a relatively big value $\bar{\mu}$ for μ and a feasible solution $(x(\bar{\mu}), y(\bar{\mu}), z(\bar{\mu}), s(\bar{\mu}))$ of the KKT conditions (10.3), (10.4), (10.5), and (10.6), for $\bar{\mu}$, the main idea of the interior point method is to decrease $\mu = \frac{\bar{\mu}}{t}$, $t > 1$, and to solve approximately the KKT conditions for μ. A Newton method starting at $(x(\bar{\mu}), y(\bar{\mu}), z(\bar{\mu}), s(\bar{\mu}))$ is used for finding $(x(\mu), y(\mu), z(\mu), s(\mu))$. This procedure is repeated until a desired duality gap is achieved. For a duality gap of $\epsilon > 0$ we need to solve at most

$$N = \left\lceil \log\left(\frac{2\bar{\mu}m}{\epsilon}\right) \frac{1}{\log t} \right\rceil$$

KKT conditions, applying the Newton method.

The interior point method is known to be polynomial. Since we use the interior point method of the commercial software Cplex ([Cpl]) as a black box we do not discuss the complexity in detail. For a survey of Primal-Dual Interior Point method for linear and quadratic minimization problems, the reader is referred to [Wri97]. For general convex optimization problems the reader is refereed to [NN94], where Nesterov and Nemirovsi precisely investigated the generation of appropriate barrier functions.

10.2 Fast Gradient Method

We apply a fast gradient method, based on the method presented by Nesterov in [Nes07], to the Lagrange dual of the MQCF obtained by relaxing the demand

constraint (10.1), which are first normalized by the demand η, i.e., we set $\tilde{E}x = \tilde{d}$ with for any node $z \in \mathcal{N}$ and for any arc $j \in \mathcal{A}$

$$\tilde{E}_{z,j} = \begin{cases} -\frac{1}{\eta} & \text{if } z = t(j), \\ \frac{1}{\eta} & \text{if } z = h(j), \\ 0 & \text{otherwise.} \end{cases} \quad \tilde{d}_z = \begin{cases} -1 & \text{if node } z = s, \\ 1 & \text{if node } z = t, \\ 0 & \text{otherwise.} \end{cases}$$

The dual objective function is then defined for any $p \in \mathbb{R}^n$ as follows

$$\psi(p) := \left\{ \min_{0 \le x \le c} \langle a, x \rangle + \frac{b}{2}\langle x, x \rangle + \langle p, \tilde{d} - \tilde{E}x \rangle \right\}. \tag{10.8}$$

In order to evaluate it we first need to compute

$$x_p \;:=\; \arg\min_{0 \le x \le c} \left\{ \langle a - \tilde{E}^T p, x \rangle + \frac{b}{2}\langle x, x \rangle \right\}. \tag{10.9}$$

This minimization problem corresponds to a quadratic projection over a box, whose solution is given as follows

$$\begin{aligned} (x_p)_j &= \max\left\{0, \min\left\{\frac{(\tilde{E}^T p - a)_j}{b}, c_j\right\}\right\} \\ &= \max\left\{0, \min\left\{\frac{1}{b}(\frac{1}{\eta}p_{h(j)} - \frac{1}{\eta}p_{t(j)} - a_j), c_j\right\}\right\} \end{aligned} \tag{10.10}$$

where $\left(\frac{\tilde{E}^T p - a}{b}\right)$ is the optimum of the corresponding unconstrained quadratic minimization problem (see Lemma 9.1). Finally, we get

$$\psi(p) = p_t - p_s + \sum_{j \in J_p} \left((a_j - \frac{1}{\eta}p_{h(j)} + \frac{1}{\eta}p_{t(j)})c_j + \frac{b}{2}c_j^2 \right) - \sum_{j \in I_p} \frac{1}{2b}\left(\frac{1}{\eta}p_{h(j)} - \frac{1}{\eta}p_{t(j)} - a_j\right)^2 \tag{10.11}$$

where

$$\begin{aligned} J_p &:= \left\{ j \in \mathcal{A} \;\middle|\; c_j < \frac{1}{b}(\frac{1}{\eta}p_{h(j)} - \frac{1}{\eta}p_{t(j)} - a_j) \right\}, \\ I_p &:= \left\{ j \in \mathcal{A} \;\middle|\; 0 \le \frac{1}{b}(\frac{1}{\eta}p_{h(j)} - \frac{1}{\eta}p_{t(j)} - a_j) \le c_j \right\}. \end{aligned}$$

Now we compute the gradient of $\psi(p)$, $\nabla\psi(p)$, using the sets J_p and I_p. First we

consider the partial derivatives of $\psi(p)$ at the source and the sink,

$$\begin{aligned}\partial_s\psi(p) &= -1 - \sum_{zs\in J_p}\frac{1}{\eta}c_{zs} + \sum_{sz\in J_p}\frac{1}{\eta}c_{sz} \\ &\quad - \sum_{zs\in I_p}\frac{1}{\eta b}(\frac{1}{\eta}p_s - \frac{1}{\eta}p_z - a_{zs}) + \sum_{sz\in I_p}\frac{1}{\eta b}(\frac{1}{\eta}p_z - \frac{1}{\eta}p_s - a_{sz}) \\ \partial_t\psi(p) &= 1 - \sum_{zt\in J_p}\frac{1}{\eta}c_{zt} + \sum_{tz\in J_p}\frac{1}{\eta}c_{tz} \\ &\quad - \sum_{zt\in I_p}\frac{1}{\eta b}(\frac{1}{\eta}p_t - \frac{1}{\eta}p_z - a_{zt}) + \sum_{tz\in I_p}\frac{1}{\eta b}(\frac{1}{\eta}p_z - \frac{1}{\eta}p_t - a_{tz})\end{aligned}$$

Then, for any other node w, we have

$$\begin{aligned}\partial_w\psi(p) &= -\sum_{zw\in J_p}\frac{1}{\eta}c_{zw} + \sum_{wz\in J_p}\frac{1}{\eta}c_{wz} \\ &\quad - \sum_{zw\in I_p}\frac{1}{\eta b}(\frac{1}{\eta}p_w - \frac{1}{\eta}p_z - a_{zw}) + \sum_{wz\in I_p}\frac{1}{\eta b}(\frac{1}{\eta}p_z - \frac{1}{\eta}p_w - a_{wz})\end{aligned}$$

The Fast Gradient method presented in [Nes07] assumes working with functions having Lipschitz continuous gradient. Moreover the value of the Lipschitz constant influences the convergence of the method. We define the following norm for the dual space, $\|p\|_B := (p^T Bp)^{\frac{1}{2}}$, where the matrix B is a lower estimate of the Hessian of $-\psi(p)$.

For any node w and z we have

$$\partial^2_{ww}\psi(p) = -\frac{1}{\eta^2 b}|\{zw \in I_p\} \cup \{wz \in I_p\}|$$

$$\partial^2_{wz}\psi(p) = \partial^2_{zw}\psi(p) = \frac{1}{\eta^2 b}\begin{cases} 0 & \text{if } wz \notin I_p, zw \notin I_p \\ 1 & \text{if } wz \in I_p, zw \notin I_p \\ 1 & \text{if } wz \notin I_p, zw \in I_p \\ 2 & \text{if } wz \in I_p, zw \in I_p \end{cases}$$

For the elements in the diagonal of the Hessian of $\psi(p)$, we have that for any node w, the set $\{zw \in I_p\} \cup \{wz \in I_p\}$ has at most $\deg_w$ elements, where $\deg_w$ denotes the degree of node w, i.e., the number of arcs having w as head plus the number of arcs having w as tail.

Thus, the diagonal matrix B, with $B_{ww} = \frac{1}{\eta^2 b}\deg_w$ for any node w, satisfy $-\nabla^2\psi(p) \preceq 2B$ and $\sqrt{2}$ is an estimate for the Lipschitz constant L of the gradient of $-\psi$, i.e., $L \approx \sqrt{2}$.

We consider then the Fast Gradient scheme described in Algorithm 26. The stop criterion of the algorithm is the value of the gradient's norm, which is zero at optimum.

Algorithm 26 MQCF—Fast Gradient Scheme

Input: $p_0 = q_0 = 0 \in \mathbb{R}^n$, initial solutions
$L_0 < L_\psi$, estimate for gradient's Lipschitz constant L_ψ
$\alpha_0 \in (0, 1)$ and $A = 0$
$\epsilon > 0$, desired accuracy
Output: $p(\epsilon) \in \mathbb{R}^n$ such that $\|\nabla\psi(p(\epsilon))\|_B^* \leq \epsilon$ and the corresponding flow solution $x(\epsilon)$.

$k = 0$
repeat
  $L = L_k$
  repeat
    find a such that $La^2 - a - A_k = 0$
    set $y = \frac{A_k}{A_k+a} p_k + \frac{a}{A_k+a} q_k$
    compute $\nabla\psi(y)$ and set $\bar{y} = y + \frac{1}{L} B^{-1}\nabla\psi(y)$
    compute $\nabla\psi(\bar{y})$
    if $\langle \nabla\psi(\bar{y}), y - \bar{y}\rangle > \frac{1}{2L}\|\nabla\psi(\bar{y})\|_B^{*\,2}$ **then**
      $L = 2L$
    end if
  until $\langle \nabla\psi(\bar{y}), y - \bar{y}\rangle \leq \frac{1}{2L}\|\nabla\psi(\bar{y})\|_B^{*\,2}$
  $a_{k+1} = a$ and $A_{k+1} = A_k + a$
  set $L_{k+1} = \frac{L}{2}$
  set $p_{k+1} = \bar{y}$ and $q_{k+1} = q_k + a_{k+1} B^{-1}\nabla\psi(p_{k+1})$
  $k = k + 1$
until $\|\nabla\psi(p_{k+1})\|_B^* \leq \epsilon$
set $p(\epsilon) = p_{k+1}$
set $x(\epsilon)_j = \max\{0, \min\{\frac{1}{b}(\frac{1}{\eta} p(\epsilon)_{h(j)} - \frac{1}{\eta} p(\epsilon)_{t(j)} - a_j), c_j\}\}$ for all $j \in \mathcal{A}$.

From Theorem 6 in [Nes07] we have that at each step $k \geq 1$ of Algorithm 26,

$$\psi(p^*) - \psi(p_k) \leq \frac{8 L_\psi \|p^* - p_0\|_B^2}{k^2},$$

where p^* denotes a maximum of ψ over $\mathbb{R}^n$. In this work, We do not provide more details on the analysis of the method. The reader is referred to [Nes07]. However, note that $\bar{y} = y + \frac{1}{L} B^{-1}\nabla\psi(y)$ is a standard gradient step and q_k is the maximum

network	# nodes	# arcs
Anaheim	416	914
Barcelona	1020	2522
Zurich	5835	15184

Tab. 10.1: Networks' characteristics

of the following function

$$\phi_k(q) := \sum_{i=1}^{k} a_i \left(\psi(p_i) + \langle \nabla\psi(p_i), q - p_i\rangle\right) + \frac{1}{2}\|q - p_0\|_B^2$$

which when divided by A_k is a strongly concave approximation of $\psi(q)$. Namely,

$$\phi(k) \geq A_k\psi(q) - \frac{1}{2}\|q - p_0\|_B^2.$$

10.3 Numerical Results

In order to compare the methods presented in the previous sections, i.e., the Primal-Dual Interior Point method (section 10.1) and the Fast Gradient method (section 10.2), we design the following experiment.

Test Instances. First, we create a pool of instances of the MQCF problem based on the networks used in Chapter 9. In particular, we consider the Anaheim, the Barcelona and the Zurich network. Their characteristics are resumed in Table 10.1. Four $\mathcal{OD}$-pairs are chosen for each network in order to define the sources, the sinks and the demands. The linear costs a and the quadratic factor b, are uniformly distributed on given intervals. These intervals are specified at the top of each table with numerical results.

Stopping Criteria. For the Interior Point method, we use the commercial solver Cplex [Cpl]. The Interior Point method in Cplex uses as stopping criterion not the duality gap (10.7), but the duality gap divided by the norm of the primal solution and the norm of the dual solution,

$$\frac{f(x(\mu)) - \psi(y(\mu), z(\mu), s(\mu))}{\|x(\mu)\|_2\|y(\mu), z(\mu), s(\mu)\|_2} = \frac{2m\mu}{\|x(\mu)\|_2\|y(\mu), z(\mu), s(\mu)\|_2}. \tag{10.12}$$

For the Fast Gradient method we use as stopping criterion the norm of the gradient of $\psi(p)$, $\|\nabla\psi(p)\|_B^*$. The algorithm is implemented in C++.

We run the methods for different values of their stopping criteria and numerically compare the delivered solutions. For sake of clarity, we use the following notations in this section.

network	$a_j \in (-1,1)$ $b \in (0,1)$		$a_j \in (-1,1)$ $b \in (0,100)$		$a_j \in (-100,100)$ $b \in (0,1)$	
	objective value	cpu time [s]	objective value	cpu time [s]	objective value	cpu time [s]
Anaheim 1	$1.5*10^8$	0.08	$1.5*10^{10}$	0.08	$1.5*10^8$	0.08
Anaheim 2	$4.3*10^7$	0.07	$4.3*10^9$	0.08	$4.3*10^7$	0.06
Anaheim 3	$9.9*10^7$	0.07	$9.9*10^9$	0.08	$9.9*10^7$	0.05
Anaheim 4	$4.7*10^7$	0.06	$4.7*10^9$	0.07	$4.7*10^7$	0.06
Barcelona 1	$1.5*10^9$	0.22	$1.5*10^{11}$	0.24	$1.5*10^9$	0.18
Barcelona 2	$1.5*10^9$	0.20	$1.5*10^{11}$	0.20	$1.5*10^9$	0.18
Barcelona 3	$7*10^8$	0.09	$7*10^{10}$	0.14	$7.1*10^8$	0.15
Barcelona 4	$5.9*10^8$	0.10	$5.9*10^{10}$	0.09	$5.9*10^8$	0.15
Zurich 1	$4.6*10^6$	0.44	$4.6*10^8$	0.51	$-2*10^6$	0.41
Zurich 2	$4.6*10^6$	0.47	$4.6*10^8$	0.53	$-2*10^6$	0.4
Zurich 3	$2.6*10^6$	0.43	$2.6*10^8$	0.53	$-4*10^6$	0.38
Zurich 4	$7.4*10^6$	0.46	$7.4*10^8$	0.6	$1.5*10^6$	0.41

Tab. 10.2: Reference solution values delivered by CPLEX 11.0 for an accuracy of 10^{-12}

stop_ip: stopping criterion for the Primal-Dual Interior Point method (normalized duality gap)
stop_fg: stopping criterion for the Fast Gradient method ($\|\nabla\psi(p)\|_B^*$)

Investigated values. All of the generated instances are feasible. Namely, we assume that the solution delivered by Cplex at maximal accuracy, 10^{-12}, to be optimal. We consider this solution as the optimal solution and compute the absolute error, the relative error as well as the difference of flow in the arcs. All computed solutions satisfy the capacities constraints (10.2) but not always the demand constraints (10.1). Thus, we also compute $\|Ex - d\|_2$ and $\|Ex - d\|_\infty$.

Results. We first present the values of the solutions delivered by the Primal-Dual Interior Point method for stop_ip = 10^{-12}, see Table 10.2. These shall be the solutions of reference, assumed to be almost optimal. We note that the method is fast. It takes at most 0.53 seconds for the biggest network, i.e., Zurich.

Now, we consider the solutions generated by the Primal-Dual Interior Point method using small values for the stopping criterion. We tested it for stop_ip=0.01, 0.001, 0.0001, and 0.00001. It turns out that the solutions obtained with stop_ip=0.01 are the same as the solutions obtained with stop_ip=0.00001. Thus, the method finds good solutions early. On Table 10.3, we present the results for the Barcelona network. We note that the quality of the solutions are reasonable since the relative

network	absolute error	relative error	max flow difference	average flow difference	cpu time [s]
	$a_j \in (-1,1)$, $b \in (0,1)$				
Barcelona 1	69.362	$4.6{*}10^{-8}$	1.4238	0.0432	0.19
Barcelona 2	20.410	$1.3{*}10^{-8}$	0.8086	0.0213	0.17
Barcelona 3	26.361	$3.7{*}10^{-8}$	0.9410	0.0210	0.09
Barcelona 4	42.179	$7.1{*}10^{-8}$	1.1126	0.0350	0.09
	$a_j \in (-1,1)$, $b \in (0,100)$				
Barcelona 1	36087.3	$2.4{*}10^{-8}$	2.9872	0.1063	0.11
Barcelona 2	2560.5	$1.7{*}10^{-8}$	0.8958	0.0278	0.11
Barcelona 3	2317.3	$3.3{*}10^{-8}$	0.8835	0.0227	0.12
Barcelona 4	13410.6	$2.6{*}10^{-8}$	1.8970	0.0683	0.09
	$a_j \in (-100,100)$, $b \in (0,1)$				
Barcelona 1	471.406	$3.1{*}10^{-7}$	2.9362	0.0763	0.15
Barcelona 2	15.396	$1.0{*}10^{-8}$	1.3911	0.0099	0.14
Barcelona 3	23.087	$3.3{*}10^{-8}$	0.9648	0.0153	0.13
Barcelona 4	26.116	$4.4{*}10^{-8}$	1.0685	0.0196	0.14

Tab. 10.3: Primal-Dual Interior Point method using 0.01 as stopping criterion value

errors are between 10^{-9} and 10^{-6}. The cpu time is slightly better than the cpu time needed to compute the solution of reference.

Now, we consider the Fast Gradient method. In order to get a relative error of 10^{-8}, we need to run the method with stop_fg = 0.0001. Table 10.4 resumes the results. Immediately, we note that the method is much slower than the Primal-Dual Interior Point method even for computing the solution of reference. The Fast Gradient method is 10-40 times slower than the Primal-Dual Interior Point method with stop_ip=0.01. However, we note that the difference of the arc's flow between the solutions delivered by the Fast Gradient method and the reference solution are smaller than the difference of the arc's flow between the solutions delivered by the Primal-Dual Interior Point method with stop_ip =0.01.

As mentioned, we do not expect the demand constraints to be satisfied by the methods. We note that the solutions delivered by the Primal-Dual Interior Point method with stop_ip =0.01 and the solutions delivered by the Fast Gradient method with stop_fg = 0.0001 violate the demand constraints similarly, see Table 10.5. However, these violations are small. Namely, the value of $\|Ex - d\|_2$ is between 10^{-6} and 0.4 and the value of $\|Ex - d\|_\infty$ is between $3 * 10^{-7}$ and $7 * 10^{-5}$.

The last table, Table 10.6, presents the results concerning the cpu time for the Primal-Dual Interior Point method and the Fast Gradient method. The latter

network	absolute error	relative error	max flow difference	average flow difference	cpu time [s]
	$a_j \in (-1,1)$, $b \in (0,1)$				
Barcelona 1	-32.292	$-2.1*10^{-8}$	0.0033	$2.2*10^{-5}$	2.43
Barcelona 2	-24.716	$-1.6*10^{-8}$	0.0062	$2.0*10^{-5}$	2.66
Barcelona 3	2.553	$3.6*10^{-9}$	0.0035	$1.1*10^{-5}$	2.56
Barcelona 4	37.9306	$6.3*10^{-9}$	0.0051	$2.5*10^{-5}$	2.26
	$a_j \in (-1,1)$, $b \in (0,100)$				
Barcelona 1	-109.406	$-7.3*10^{-10}$	0.0043	$8.1*10^{-5}$	4.09
Barcelona 2	468.855	$3.1*10^{-9}$	0.0027	$4.4*10^{-5}$	4.97
Barcelona 3	-387.789	$-5.5*10^{-9}$	0.0026	$3.5*10^{-5}$	3.93
Barcelona 4	-261.015	$-4.4*10^{-9}$	0.0021	$2.5*10^{-5}$	3.56
	$a_j \in (-100,100)$, $b \in (0,1)$				
Barcelona 1	25.577	$1.7*10^{-9}$	0.0009	$1.2*10^{-5}$	2.17
Barcelona 2	2.921	$1.9*10^{-9}$	0.0075	$1.4*10^{-5}$	1.85
Barcelona 3	23.393	$3.3*10^{-8}$	0.0015	$1.8*10^{-5}$	1.85
Barcelona 4	20.229	$3.4*10^{-8}$	0.0003	$1.7*10^{-5}$	1.65

Tab. 10.4: Fast Gradient method using 0.0001 as stopping criterion value

is definitely slower than the Primal-Dual Interior Point method. An interresting phenomenon can be observed. As the values of the quadratic costs decrease in respect to the values of the linear costs, the Primal-Dual Interior Point method as well as the Fast Gradient method become quicker.

We conclude that the Primal-Dual Interior Point method is the method of choice for solving the MQCF problem. It generates good approximate solutions quickly. However, to avoid depedence on Cplex 11.0, the method should be self-implemented.

network	$a_j \in (-1,1)$, $b \in (0,1)$							
	Primal-Dual Interior Point stop_ip=0.01				Fast Gradient stop_fg=0.0001			
	relative error	cpu time[s]	$\|Ex-d\|_2$	$\|Ex-d\|_\infty$	relative error	cpu time[s]	$\|Ex-d\|_2$	$\|Ex-d\|_\infty$
Anaheim 1	$3.9*10^{-8}$	0.07	$1.8*10^{-6}$	$3.1*10^{-7}$	$-3.8*10^{-8}$	0.29	$2.2*10^{-4}$	$6.2*10^{-5}$
Anaheim 2	$9.9*10^{-8}$	0.06	0.0014	$1.1*10^{-6}$	$-4.3*10^{-7}$	0.21	0.0149	$4.4*10^{-5}$
Anaheim 3	$2.9*10^{-7}$	0.06	0.0369	$2.3*10^{-6}$	$3.3*10^{-7}$	0.22	0.1222	$3.5*10^{-5}$
Anaheim 4	$2.0*10^{-7}$	0.05	0.1920	$3.8*10^{-6}$	$1.1*10^{-7}$	0.27	0.3496	$3.7*10^{-5}$
	$a_j \in (-1,1)$, $b \in (0,100)$							
Anaheim 1	$5.7*10^{-8}$	0.07	$1.5*10^{-6}$	$2.2*10^{-6}$	$7.3*10^{-10}$	0.42	$2.4*10^{-5}$	$6.8*10^{-6}$
Anaheim 2	$1.3*10^{-7}$	0.06	0.0039	$6.6*10^{-6}$	$3.8*10^{-9}$	0.32	0.0049	$4.9*10^{-6}$
Anaheim 3	$9.7*10^{-8}$	0.03	0.0627	$2.1*10^{-6}$	$4.6*10^{-9}$	0.39	0.0701	$6.0*10^{-6}$
Anaheim 4	$5.9*10^{-8}$	0.03	0.2505	$1.2*10^{-7}$	$4.5*10^{-9}$	0.38	0.2648	$5.8*10^{-7}$
	$a_j \in (-100,100)$, $b \in (0,1)$							
Anaheim 1	$1.8*10^{-7}$	0.05	$4.2*10^{-5}$	$5.7*10^{-6}$	$-1.2*10^{-8}$	0.27	$2.3*10^{-4}$	$4.1*10^{-5}$
Anaheim 2	$2.7*10^{-7}$	0.05	0.0065	$1.1*10^{-6}$	$-4.1*10^{-8}$	0.25	0.0153	$3.6*10^{-5}$
Anaheim 3	$6.7*10^{-7}$	0.05	0.0805	$3.1*10^{-6}$	$-2.6*10^{-8}$	0.21	0.1235	$1.8*10^{-5}$
Anaheim 4	$3.3*10^{-7}$	0.04	0.2838	$1.2*10^{-6}$	$2.1*10^{-8}$	0.22	0.3515	$2.0*10^{-5}$

Tab. 10.5: Violation of the demand constraints, Primal-Dual Interior Point method with stop_ip=0.01 v.s. Fast Gradient with stop_fg=0.0001

network	$a_j \in (-1,1)$, $b \in (0,1)$							
	Primal-Dual Interior Point stop_ip=0.0001		Fast Gradient stop_fg=0.01		Fast Gradient stop_fg=0.001		Fast Gradient stop_fg=0.0001	
	relative error	cpu time [s]	relative error	cpu time [s]	relative error	cpu time [s]	relative error	cpu time [s]
Zurich 1	$7.9*10^{-7}$	0.36	$-6.9*10^{-6}$	3.64	$1.2*10^{-6}$	8.83	$-3.1*10^{-8}$	15.20
Zurich 2	$9.5*10^{-6}$	0.38	$-5.0*10^{-8}$	1.95	$6.1*10^{-8}$	4.68	$-8.3*10^{-9}$	9.05
Zurich 3	$9.9*10^{-7}$	0.36	$-6.9*10^{-6}$	2.87	$2.1*10^{-7}$	5.78	$7.8*10^{-8}$	11.36
Zurich 4	$8.3*10^{-8}$	0.39	$6.7*10^{-6}$	9.61	$3.1*10^{-6}$	17.02	$1.3*10^{-8}$	26.24
	$a_j \in (-1,1)$, $b \in (0,100)$							
Zurich 1	$1.1*10^{-8}$	0.42	$1.3*10^{-7}$	9.26	$2.8*10^{-8}$	22.26	$6*10^{-9}$	37.09
Zurich 2	$8.7*10^{-9}$	0.46	$-1.6*10^{-7}$	5.46	$2.3*10^{-9}$	12.83	$-1.9*10^{-9}$	20.43
Zurich 3	$4.4*10^{-8}$	0.44	$1.4*10^{-6}$	6.54	$2.7*10^{-8}$	16.95	$-4.6*10^{-9}$	25.51
Zurich 4	$5.3*10^{-8}$	0.49	$5.9*10^{-6}$	22.47	$-3.4*10^{-7}$	38.07	$-2.6*10^{-8}$	61.24
	$a_j \in (-100,100)$, $b \in (0,1)$							
Zurich 1	$-6.2*10^{-6}$	0.33	$3.5*10^{-6}$	3.83	$8.6*10^{-7}$	9.27	$3.5*10^{-8}$	20.81
Zurich 2	$-7.2*10^{-6}$	0.34	$6.9*10^{-6}$	3.45	$3.2*10^{-6}$	7.99	$-2.7*10^{-7}$	18.08
Zurich 3	$-7.2*10^{-6}$	0.31	$4.5*10^{-6}$	3.03	$-2.5*10^{-7}$	7.24	$-1.3*10^{-7}$	17.56
Zurich 4	$3.8*10^{-5}$	0.31	$4.1*10^{-4}$	6.97	$1.4*10^{-5}$	14.41	$4.5*10^{-6}$	33.89

Tab. 10.6: Cpu time: Fast Gradient v.s. Primal-Dual Interior Point method

11. Conclusions

This thesis aimed at improving the understanding of approximation algorithms for solving large scale linear programs from a theoretical as well as practical point of view. We investigated on the one hand a Primal-Dual Subgradient method and on the other hand an approximation scheme combining smoothing techniques with optimal gradient methods, which is called Excessive Gap method. The former has a theoretical convergence dependency of $O(1/\epsilon^2)$ on the desired accuracy of ϵ while the latter needs only $O(1/\epsilon)$. We applied the Excessive Gap method to solve the linear programming relaxation of the Uncapacitated Facility Location problem. By a Lagrangian reformulation, the problem became accessible for the method and, thus, we were able to design a polynomial approximation scheme with a running time of $O(1/\epsilon)$. To our knowledge, the previously best known approximation scheme result was $O(1/\epsilon^2)$, [GK02].

Both investigated algorithms rely on oracles for solving subproblems. We have investigated the dependence of the methods on the accuracy of the oracles. We found that the primal-dual subgradient method requires oracles with an accuracy of ϵ^2 for an overall accuracy of ϵ. The Excessive Gap method requires for the same precision an oracle with an accuracy of ϵ^5 suggesting that it is much less stable. However, we suppose that this result could be improved by finding suitable additional requirements for the oracles. Moreover, the practical results showed that a much lower accuracy sufficed.

To investigate the practical performance of the algorithms, we have applied them to random instances of the Uncapacitated Facility Location Problem and on real-world instances of the Travel Assignment Problem. Exact oracles are available for the former and the Excessive Gap method significantly outperformed the Primal-Dual Subgradient method as expected due to the theoretical running time results. This is not the case for the Travel Assignment Problem, for which we do not have exact oracles for the Excessive Gap method. Here, the Primal-Dual Subgradient method showed better results. We have also found that the bounding of the feasible spaces and their norms as well as the choice of the prox-functions do not only affect the theoretical convergence, but also show to have a big impact on the practical running time.

Finally, we have compared two different formulations of the Traffic Assignment Problem, namely the classic formulation of Beckmann and a recent version of Nes-

terov & de Palma. Test results for travel data of the Zurich metropolitan region show that travel flows are much more concentrated in the second model leading to more predictable congestion.

Results were obtained for the Beckmann model using a commercially available solver, while we applied the Primal-Dual Subgradient method for solving the Nesterov & de Palma model. Our algorithm compared poorly from a computation time point of view, which can be explained by the fact that we evaluated additional information during computation. It would be interesting to see how the algorithm performs if this information was not incorporated.

Appendix

A. Miscellaneous

Theorem A.1
Consider the linear minimization problem LP and its Lagrange relaxation LR,

$$LP := \begin{array}{ll} \min & c^T x \\ & Ax \leq b \\ & x \in Q \end{array} \qquad LR := \begin{array}{lll} \max & \min & \{c^T x + u^T(Ax - b)\} \\ u \geq 0 & x \in Q & \end{array}$$

where $Q := \{x \mid \tilde{A}x \leq \tilde{b}\}$ is a polyhedron. If the linear minimization problem has a finite optimal solution, then $LP = LR$.

Proof. One can easily show that $LP \geq LR$. Then, consider the dual problem of the linear problem

$$DP := \begin{array}{ll} \max & -b^T v - \tilde{b}^T w \\ & -A^T v - \tilde{A}^T w = c \\ & v, w \geq 0 \end{array}$$

Since the linear problem has an optimal solution, there is by strong duality an optimal solution (v^*, w^*) of the dual problem and $DP = LP$. Now we evaluate the Lagrange function $L(u) := \min_{x \in Q} c^T x + u^T(Ax - b)$ at $u = v^*$.

$$\begin{aligned} L(v^*) &= \min_{x \in Q} (c^T + v^{*T} A)x - v^{*T} b \\ &= \min_{x \in Q} -w^{*T} \tilde{A}x - v^{*T} b \\ &\geq -w^{*T} \tilde{b} - v^{*T} b \\ &= DP = LP \end{aligned}$$

Thus $LR = LP$. □

Theorem A.2 (Weierstrass's Theorem, Proposition A.8 in [Ber95])
Let $E \subset \mathbb{R}^n$ be a nonempty closed set and let $f : E \longrightarrow \mathbb{R}$ be lower semi-continuous at all points of E.

a) If E is compact, $x^* \in E$ exists such that $f(x^*) = \inf_{x \in E} f(x)$.

b) If f is coercive, $x^* \in E$ exists such that $f(x^*) = \inf_{x \in E} f(x)$.

Theorem A.3 ([Roc70], Corollary 37.3.2)
Let C and D be a non empty closed convex sets in $\mathbb{R}^m$ and $\mathbb{R}^n$, respectively, and let K be a continuous finite concave-convex function on $C \times D$. If either C or D is bounded, one has

$$\inf_{v\in D} \sup_{u\in C} K(u,v) = \sup_{u\in C} \inf_{v\in D} K(u,v).$$

Theorem A.4 ([Nes03], Theorem 3.1.14)
Let f be a closed convex function. For any $x_0 \in int(dom\ f)$ and $p \in \mathbb{R}^n$ we have

$$f'(x_0;p) = \max\{\langle \xi, p\rangle | \xi \in \partial f(x_0)\}.$$

Theorem A.5 (Danskin's Theorem, Proposition B.25 in [Ber95])
Let $T \subset \mathbb{R}^m$ be a compact set and $\phi : \mathbb{R}^n \times T \longrightarrow \mathbb{R}$ be a continuous function such that the function $x \longmapsto \phi(x,u)$ is convex for each $u \in T$. Then the function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ defined by

$$f(x) = \max_{u\in T} \phi(x,u)$$

is convex and has directional derivative given by

$$f'(x,y) = \max_{u\in T_x} \phi'(x,u;y)$$

where

$$\phi'(x,u;y) = \lim_{\alpha\downarrow 0} \frac{\phi(x+\alpha y,u) - \phi(x,u)}{\alpha}$$

is the directional derivative of $\phi(.,u)$ in direction y and

$$T_x := \{\bar{u} \mid \phi(x,\bar{u}) = \max_{u\in T}\ \phi(x,u)\}.$$

In particular if the set T_x contains one unique point $\bar{u}$ and $\phi(.,u)$ is differentiable at x, then f is differentiable at x, and $\nabla f(x) = \nabla_x \phi(x,\bar{u})$.

Theorem A.6 (Properties of $g(x)$, Danskin's Theorem in [Ber95] and Theorem 1 in [Nes05c]))
Consider the function $g(x)$ defined in (3.10). Then, $g(x)$ has the following properties.
$g(x)$ is a convex and smooth function and its gradient $\nabla g(x)$ is Lipschitz continuous with Lipschitz constant $L_{g,S}$, i.e.,

$$\nabla g(x) = \nabla \hat{g}(x) + B^T u_x \qquad\qquad L_{g,S} = L_{\hat{g},S} + \frac{\|B\|^2_{S,T}}{\beta\sigma_T} \tag{A.1}$$

where $u_x := \arg\ \max_{u\in T}\ \{\langle Bx,u\rangle - \hat{\phi}(u) - \beta d(u)\}$.

Proof. First we show that g is convex. Let $\alpha \in [0,1]$, then

$$\begin{aligned} g\left(\alpha x + (1-\alpha)y\right) &= \hat{g}\left(\alpha x + (1-\alpha)y\right) + \\ &\quad \max_{u\in T} \left\{\langle B(\alpha x + (1-\alpha)y), u\rangle - \hat{\varphi}(u) - \beta d_T(u)\right\} \\ &\leq \alpha\left\{\hat{g}(x) + \max_{u\in T}\left\{\langle Bx, u\rangle - \hat{\varphi}(u) - \beta d_T(u)\right\}\right\} + \\ &\quad (1-\alpha)\left\{\hat{g}(y) + \max_{u\in T}\left\{\langle By, u\rangle - \hat{\varphi}(u) - \beta d_T(u)\right\}\right\} \\ &= \alpha g(x) + (1-\alpha)g(y). \end{aligned}$$

In order to show that g is differentiable we consider Danskin's Theorem (Theorem A.5) for

$$\begin{aligned} \phi : S \times T &\longrightarrow \mathbb{R} \\ (x,u) &\longmapsto \phi(x,u) := \hat{g}(x) + \Gamma_\beta(x,u). \end{aligned}$$

Then, $g(x) = \max_{u\in T}\phi(x,u)$ and is differentiable if and only if the maximizer of $\phi(x,.)$ over T for each $x \in S$ is unique and $\phi(.,u)$ is differentiable. Since $\Gamma_\beta(x,.)$ is strongly concave over T for each $x \in S$, $\phi(x,.)$ is also strongly concave over T for each $x \in S$ and thus, its maximizer is unique and is $u_x = \arg\max_{u\in T}\Gamma_\beta(x,u)$. Moreover,

$$\nabla g(x) = \nabla_x \phi(x,u_x) = \nabla\hat{g}(x) + \nabla_x\Gamma_\beta(x,u_x) = \nabla\hat{g}(x) + B^T u_x,$$

since $\hat{g}$ and $\Gamma_\beta(.,u)$ are differentiable.

We now evaluate the value of the Lipschitz constant $L_{g,S}$. For $x, y \in S$ we have

$$\begin{aligned} \|\nabla g(x) - \nabla g(y)\|_S^* &= \|\nabla\hat{g}(x) - \nabla\hat{g}(y) + B^T u_x - B^T u_y\|_S^* \\ &\leq \|\nabla\hat{g}(x) - \nabla\hat{g}(y)\|_S^* + \|B^T(u_x - u_y)\|_S^*. \end{aligned}$$

Thus we need to bound $\|B^T(u_x - u_y)\|_S^*$. Using the first order condition for $\Gamma_\beta(x,.)$ at u_x and for $\Gamma_\beta(y,.)$ at u_y, i.e.,

$$\begin{aligned} \langle Bx - \nabla\hat{\varphi}(u_x) - \beta\nabla d_T(u_x), u_y - u_x\rangle &\leq 0 \\ \langle By - \nabla\hat{\varphi}(u_y) - \beta\nabla d_T(u_y), u_x - u_y\rangle &\leq 0 \end{aligned}$$

we get

$$\begin{aligned} \langle B(x-y), u_x - u_y\rangle &\geq \langle\nabla\hat{\varphi}(u_x) - \nabla\hat{\varphi}(u_y), u_x - u_y\rangle \\ &\quad +\beta\langle\nabla d_T(u_x) - \nabla d_T(u_y), u_x - u_y\rangle \\ &\geq \beta\langle\nabla d_T(u_x) - \nabla d_T(u_y), u_x - u_y\rangle \\ &\geq \beta\sigma_T\|u_x - u_y\|_T^2. \end{aligned}$$

We used the convexity of $\hat{\varphi}(u)$ to prove the second inequality and the strong convexity of $d_T(u)$ to prove the third inequality. Hence,

$$\begin{aligned}\left(\|B^T(u_x-u_y)\|_S^*\right)^2 &\leq \|B\|_{S,T}^2\|u_x-u_y\|_T^2\\ &\leq \|B\|_{S,T}^2\frac{1}{\beta\sigma_T}\langle B(x-y),u_x-u_y\rangle\\ &\leq \frac{\|B\|_{S,T}^2}{\beta\sigma_T}\|B^T(u_x-u_y)\|_S^*\|x-y\|_S.\end{aligned}$$

Finally, $\|\nabla g(x)-\nabla g(y)\|_S^* \leq \left(L_{\hat{g},S}+\frac{\|B\|_{S,T}^2}{\beta\sigma_T}\right)\|x-y\|_S$. □

B. List of Notation

Part I - Large Scale Linear Programs and Optimization Methods

Chapter 2

LP	linear program
Dual LP	dual program
LR	Langrange dual program
$x \in \mathbb{R}^n$	primal decision variables
$z \in \mathbb{R}^m$	dual variables
$u \in \mathbb{R}^m$	Lagrange dual variables
$c \in \mathbb{R}^n$	linear cost vector
$b \in \mathbb{R}^m$	right-hand side vector
$A \in \mathbb{R}^{m \times n}$	matrix of constraints
$Q \subseteq \mathbb{R}^n$	polyhedron
$\psi(u)$	Lagrange dual objective function

Chapter 3

Problem Description

LP	linear program
LR	Langrange dual program

$x \in \mathbb{R}^n$	primal decision variables
$u \in \mathbb{R}^m$	Lagrange dual variables
$c \in \mathbb{R}^n$	linear cost vector
$b \in \mathbb{R}^m$	right-hand side vector
$A \in \mathbb{R}^{m \times n}$	matrix of constraints

$Q \subset \mathbb{R}^n$	non-empty, convex and compact polytope
$P \subset \mathbb{R}^m$	non-empty, convex and compact polytope
$f(x) := \max_{u \in P} \{c^T x + u^T(Ax - b)\}$	primal objective function
f^*, x^*	minimum and minimizer of $f(x)$ over Q
$\psi(u) := \min_{x \in Q} \{c^T x + u^T(Ax - b)\}$	Lagrange dual objective function
ψ^*, u^*	maximum and maximizer of $\psi(u)$ over P
ϵ	absolute accuracy

Strongly Convex Functions

$S \subseteq \mathbb{R}^n$	convex set or compact convex set
$\|.\|_S$, $\|.\|_S^*$	norm defined over $\mathbb{R}^n$ containing the set S and its dual/adjoint norm
$T \subseteq \mathbb{R}^m$	convex set or compact convex set
$\|.\|_T$, $\|.\|_T^*$	norm defined over $\mathbb{R}^m$ containing the set T and its dual/adjoint norm
$\|B\|_{S,T},\ B \in R^{m \times n}$	norm of matrix B with respect to norm $\|.\|_S$ and norm $\|.\|_T$
$d_S(x)$	prox-function over S with convexity parameter σ_S in respect to norm $\|.\|_S$ and minimizer x^o over S
$d_T(u)$	prox-function over T with convexity parameter σ_T in respect to norm $\|.\|_T$ and minimizer u^o over T

epi g	epigraph of function g
$\nabla g(x)$	gradient of function g
$L_{g,S}$	Lipschitz constant of the gradient of function g
$\hat{g}(x)$	smooth convex function over S
$\hat{\varphi}(u)$	smooth convex function over T
$\Gamma_\beta(x,u)$ $:= \langle Bx, u\rangle - \hat{\varphi}(u) - \beta d_T(u)$	saddle function, which is convex over S for fixed u and concave over T for fixed x, $\beta > 0$
u_x	maximizer of $\Gamma_\beta(x,.)$ over T

From an Absolute to a Relative Error

ϵ	absolute accuracy
ϵ'	relative accuracy
A-FEAS	decision procedure

Chapter 4

$S \subseteq \mathbb{R}^n$	convex set or compact convex set
$\|.\|_S$, $\|.\|_S^*$	norm defined over $\mathbb{R}^n$ containing the set S and its dual/adjoint norm
$d_S(x)$	prox-function over S with convexity parameter σ_S in respect to norm $\|.\|_S$ and minimizer x^o over S
D_S	maximum of $d_S(x)$ over S

$g : S \longrightarrow \mathbb{R}$	closed, finite and convex function
g^*, x^*	minimum and minimizer of $g(x)$ over Q
$\partial g(x)$	subdifferential of g at point x
$\xi_x \in \partial g(x)$	subgradient of g at point x
L	upper bound on subgradients' norm $\forall\ \xi_x \in \partial g(x),\ x \in S, \quad \|\xi_x\|_S^* \leq L$
M	upper bound on subgradients' variation $\forall\ \xi_x \in \partial g(x),\ \xi_y \in \partial g(y),\ x, y \in S, \quad \|\xi_x - \xi_y\|_S^* \leq M$

g_*	conjugate function of g
$\varphi(\zeta) := -g_*(\zeta) + \min\limits_{x \in S} \langle \zeta, x \rangle$	dual function
ζ	dual variables

Chapter 5

$S \subset \mathbb{R}^n$	convex set or compact convex set
$\|.\|_S$, $\|.\|_S^*$	norm defined over $\mathbb{R}^n$ containing the set S and its dual/adjoint norm
$d_S(x)$	prox-function over S with convexity parameter σ_S in respect to norm $\|.\|_S$ and minimizer x^o over S
D_S	maximum of $d_S(x)$ over S

$T \subset \mathbb{R}^m$	convex set or compact convex set
$\|.\|_T$, $\|.\|_T^*$	norm defined over $\mathbb{R}^m$ containing the set T and its dual/adjoint norm
$d_T(x)$	prox-function over T with convexity parameter σ_T in respect to norm $\|.\|_T$ and minimizer x^o over T
D_T	maximum of $d_T(x)$ over T

$\hat{g}(x)$	smooth convex function over S
$L_{\hat{g},S}$	Lipschitz constant of the gradient of $\hat{g}$

$\hat{\varphi}(u)$	smooth convex function over T
$L_{\hat{\varphi},T}$	Lipschitz constant of the gradient of $\hat{\varphi}$

$g(x) := \hat{g}(x) + \max\limits_{u \in T} \{\langle Bx, u\rangle - \hat{\varphi}(u)\}$ primal function
$\varphi(u) := -\hat{\varphi}(u) + \min\limits_{x \in S} \{\langle Bx, u\rangle + \hat{g}(x)\}$ dual function

$g_{\mu_T}(x) := \hat{g}(x) + \max\limits_{u \in T} \{\langle Bx, u\rangle - \hat{\varphi}(u) - \mu_T d_T(u)\}$ lower approximation of $g(x)$
$\varphi_{\mu_S}(u) := -\hat{\varphi}(u) + \min\limits_{x \in S} \{\langle Bx, u\rangle + \hat{g}(x) + \mu_S d_S(x)\}$ upper approximation of $\varphi(u)$

μ_T, μ_S smoothing factors
$L_{g_{\mu_T},S}$ Lipschitz constant of the gradient of g_{μ_T}
$L_{\varphi_{\mu_S},T}$ Lipschitz constant of the gradient of φ_{μ_S}

$Z_{\mu_T,x}(y) := g_{\mu_T}(x) + \langle \nabla g_{\mu_T}(x), y - x\rangle + \frac{L_{g_{\mu_T},S}}{2}\|y - x\|_S^2$ upper approximation of $g(y)$
$W_{\mu_S,u}(v) := \varphi_{\mu_S}(u) + \langle \nabla \varphi_{\mu_S}(u), v - u\rangle - \frac{L_{\varphi_{\mu_S},T}}{2}\|v - u\|_T^2$ lower approximation of $\varphi(v)$

$GM_{g_{\mu_T}}(x) := \arg \min\limits_{y \in S} Z_{\mu_T,x}(y)$ Gradient Mapping of g_{μ_T} at point x
$GM_{\varphi_{\mu_S}}(u) := \arg \max\limits_{v \in T} W_{\mu_S,u}(v)$ Gradient Mapping of φ_{μ_S} at point u

Chapter 6

δ-oracle oracle that delivers an approximate solution of the minimum of a strongly convex function
x^δ δ-approximation, i.e., approximate solution delivered by a δ-oracle

ξ_x^{abc} *abc*-subgradient of a convex function at point x

$\Gamma_{\mu_T}(x, u) := \langle Bx, u\rangle - \hat{\varphi}(u) - \mu_T d_T(u)$ saddle function, which is linear over S for fixed u and concave over T for fixed x
u_x, u_x^δ maximizer of $\Gamma_{\mu_T}(x, .)$ over T and its δ-approximation

$\Phi_{\mu_S}(x, u) := \langle Bx, u\rangle + \hat{g}(x) + \mu_S d_S(x)$ saddle function, which is convex over S for fixed u and linear over T for fixed x
x_u, x_u^δ minimizer of $\Phi_{\mu_S}(., u)$ over S and its δ-approximation

$g_{\mu_T}^\delta(x) := \hat{g}(x) + \Gamma_{\mu_T}(x, u_x^\delta)$ δ-approximate of $g_{\mu_T}(x) := \hat{g}(x) + \Gamma_{\mu_T}(x, u_x)$
$\nabla g_{\mu_T}^\delta(x) := \nabla \hat{g}(x) + B^T u_x^\delta$

$\varphi_{\mu_S}^\delta(u) := -\hat{\varphi}(u) + \Phi_{\mu_S}(x_u^\delta, u)$ δ-approximate of $\varphi_{\mu_S}(u) := -\hat{\varphi}(u) + \Phi_{\mu_S}(x_u, u)$
$\nabla \varphi_{\mu_S}^\delta(u) := -\nabla \hat{\varphi}(u) + Bx_u^\delta$

$Z^{\delta}_{\mu_T,x}(y) := g^{\delta}_{\mu_T}(x) + \langle \nabla g^{\delta}_{\mu_T}(x), y - x\rangle + \frac{L_{g_{\mu_T},S}}{2}\|y-x\|^2_S$ δ-approximation of $Z_{\mu_T,x}(y)$

$W^{\delta}_{\mu_S,u}(v) := \varphi^{\delta}_{\mu_S}(u) + \langle \nabla \varphi^{\delta}_{\mu_S}(u), v - u\rangle - \frac{L_{\varphi_{\mu_S},T}}{2}\|v-u\|^2_T$ δ-approximation of $W_{\mu_S,u}(v)$

$GM^{\delta}_{g_{\mu_T}}(x)$ δ-approximation of the minimizer of $Z^{\delta}_{\mu_T,x}(y)$ over S

$GM^{\delta}_{\varphi_{\mu_S}}(u)$ δ-approximation of the maximizer of $W^{\delta}_{\mu_S,u}(v)$ over T

Part II - Applications of the Optimization Methods to Special Linear Problems

Chapter 7

UFL problem Uncapacitated Facility Location problem

UFL-LP linear programming formulation of the UFL problem

UFL-LR Lagrange relaxation of UFL-LP

UFL-DP dual problem of UFL-LP

$\mathcal{F}$ set of potential facility locations

$\mathcal{D}$ set of clients

m, n number of facility locations and number of clients

f_i cost of building the facility i

d_j demand of client j

c_{ij} cost of serving client j by facility i

x_{ij} primal decision variable, serving or not client j by facility i

y_i primal decision variable, opening or not facility location i

w_{ij} Lagrange dual variable

u_{ij} dual variable ($w_{ij} = f_i u_{ij}$)

$\Delta_m \subset \mathbb{R}^m$ m-dimensional simplex set

$\Delta^n_m \subset \mathbb{R}^{n\times m}$ product of n m-dimensional simplex
$\Delta_m := \{z \in \mathbb{R}^m \mid \sum_{i=1}^m z_i = 1, z_i \geq 0 \ \forall\ i = 1, \ldots, m\}$

$Q = \Delta^n_m$ primal space

$P = \Delta^m_n$ dual space

$f(x) := \sum_{i=1}^m \sum_{j=1}^n c_{ij}x_{ij} + \max\limits_{u\in P} \left\{\sum_{i=1}^m \sum_{j=1}^n f_i u_{ij} x_{ij}\right\}$ primal function

$\psi(u) := \min\limits_{x\in Q} \left\{\sum_{i=1}^m \sum_{j=1}^n c_{ij}x_{ij} + \sum_{i=1}^m \sum_{j=1}^n f_i u_{ij} x_{ij}\right\}$ dual function

UFL-EG-Euclidean	Excessive Gap method using Euclidean norms and Euclidean projections
UFL-EG-Entropy	Excessive Gap method using 1-norms and Entropy functions
UFL-SDA-Euclidean	Simple Dual Averaging Primal-Dual Subgradient method using Euclidean norms and Euclidean projections
UFL-SDA-Entropy	Simple Dual Averaging Primal-Dual Subgradient method using 1-norms and Entropy functions
UFL-WDA-Euclidean	Weighted Dual Averaging Primal-Dual Subgradient method using Euclidean norms and Euclidean projections
UFL-WDA-Entropy	Weighted Dual Averaging Primal-Dual Subgradient method using 1-norms and Entropy functions
UFL-TSDA-Euclidean	Truncated Simple Dual Averaging Primal-Dual Subgradient method using Euclidean norms and Euclidean projections
UFL-TSDA-Entropy	Truncated Simple Dual Averaging Primal-Dual Subgradient method using 1-norms and Entropy functions

Chapter 8

STAP or STA problem	Static Traffic Assignment problem
UE	User Equilibrium
SO	Social Optimum

$G = (\mathcal{N}, \mathcal{A})$	traffic network
$\mathcal{N}$	set of nodes (e.g., intersections)
$\mathcal{A}$	set of arcs (e.g., roads)
$\mathcal{OD} \subset \mathcal{N} \times \mathcal{N}$	set of Origin-Destination pairs or commodities
$\mathcal{OD}$-pair	origin-destination pair

c_a	capacity of arc $a \in \mathcal{A}$
$\bar{t}_a$	free travel time of arc $a \in \mathcal{A}$

d_k	number of drivers travelling during a period of time from origin of $\mathcal{OD}$-pair k to its destination or demand of commodity k
δ_k	demand(drivers) node vector of $\mathcal{OD}$-pair k

E	node-arc incidence matrix

$\mathcal{P}_k$	set of all paths between origin and destination of $\mathcal{OD}$-pair k
a_P^k	arc incidence vector of path $P \in \mathcal{P}_k$.

h^k	flow vector for $\mathcal{OD}$-pair k
s	total flow vector, $f = \sum_{k \in \mathcal{OD}} h^k$
t	travel time vector
(s, t)	traffic assignment
$U(s,t) := \langle s, t \rangle$	total travel time of traffic assignment (s, t)

Nesterov & de Palma model

NdP	Nesterov & de Palma
NdP-SO	linear programming formulation of finding at traffic assignment at social optimum in Nesterov & de Palma model
NdP-UE	convex formulation of finding at traffic assignment at user equilibrium in Nesterov & de Palma model
$T_k(t)$	function given the length of the shortest path for $\mathcal{OD}$-pair k given travel time t.

Beckmann model

B-SO	convex formulation of finding at traffic assignment at social optimum in Beckmann model
B-UE	convex formulation of finding at traffic assignment at user equilibrium in Beckmann model
B-UEext	convex formulation of finding at traffic assignment at user equilibrium in Beckmann model with additional constraints
$l_a(t)$	latency function
BPR	Bureau of Public Road latency function

Chapter 9

m	number of arcs (e.g., roads)
n	number of nodes (e.g., intersections, zones)
K	number of $\mathcal{OD}$-pairs
d_{tot}	sum of the demands of all $\mathcal{OD}$-pairs
SO^*	optimal value of the NdP-SO problem
UE^*	optimal value of the NdP-UE problem

Primal-Dual Subgradient methods

R	upper bound on travel times
$Q := \{t \in \mathbb{R}^m \mid \bar{t} \leq t \leq R\}$	primal feasible space (travel times' space)
$f(t) := \sum_{k \in \mathcal{OD}} d_k T_k(t) - \langle t - \bar{t}, c \rangle$	primal function
$\triangle = \Delta_{\lvert\mathcal{P}_1\rvert} \times \cdots \times \Delta_{\lvert\mathcal{P}_K\rvert}$	dual feasible space
$\psi(u) := \min_{t \in Q} \langle c - \sum_{k \in \mathcal{OD}} d_k u_p^k a_p, t \rangle - \langle c, \bar{t} \rangle$	dual function

$L_{\max}$	largest upper bound on subgradients' norm considered
$M_{\max}$	largest upper bound on subgradients' variation considered

NdP-UE-SDA	Simple Dual Averaging Primal-Dual Subgradient using Euclidean norms and Euclidean projections
NdP-UE-WDA	Weighted Dual Averaging Primal-Dual Subgradient using Euclidean norms and Euclidean projections
NdP-UE-TSDA	Truncated Simple Dual Averaging Primal-Dual Subgradient using Euclidean norms and Euclidean projections

Excessive Gap method

$\Pi_k := \{h^k \in \mathbb{R}^m \mid Eh^k = \delta_k,\ 0 \leq h^k \leq c\}$	feasible space for the flow of $\mathcal{OD}$-pair k
$\Pi := \Pi_1 \times \cdots \times \Pi_K \subset \mathbb{R}^{Km}$	primal feasible space (flows' spaces)

$f(h) := \max_{u \in U} \left\{ \sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{OD}} h_a^k \bar{t}_a + \sum_{a \in \mathcal{A}} u_a (\sum_{k \in \mathcal{OD}} h_a^k - c_a) \right\}$ primal function

C	upper bound on delays
$U := \{u \in \mathbb{R}^m \mid 0 \leq u \leq C\}$	dual feasible space (delays' space)
$\psi(u) := \min_{h \in \Pi} \left\{ \sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{OD}} h_a^k \bar{t}_a + \sum_{a \in \mathcal{A}} u_a (\sum_{k \in \mathcal{OD}} h_a^k - c_a) \right\}$	dual function

NdP-UE-EG Excessive Gap using Euclidean norms and Euclidean projections

Chapter 10

MQCF problem Minimum Quadratic Cost Flow problem

$\mathcal{N}$	set of nodes
$\mathcal{A}$	set of arcs
s	source node
t	sink node
η	demand
$t(i)$	tail of arc i
$h(i)$	head of arc i
a_i	linear cost of arc i
b	quadratic cost
c_i	capacity of arc i
E	node-arc incidence matrix
d	demand vector
$\tilde{E}$	normalized node-arc incidence matrix
$\tilde{d}$	normalized demand vector

stop_ip: stopping criterion for the Primal-Dual Interior Point method
stop_fg: stopping criterion for the Fast Gradient method

Bibliography

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[BA00] F. Barahona and R. Anbil. The volume algorithm: producing primal solutions with the subgradient method (Technical report IBM Watson Research Center 1998). *Mathematical Programming*, 87:385–399, 2000.

[Bal65] M. L. Balinski. Integer programming: methods, uses, computations. *Management Science*, 12(3):253–313, 1965.

[BB06] M. Baïou and F. Barahona. On the integrality of the uncapacitated facility location polytope. Technical report, IBM Watson Research Center, 2006.

[BC99] F. Barahona and F. A. Chudak. Solving large scale uncapacitated facility location problems. In *Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems*. Kluwer Academic Publishers, 1999.

[BdMV06] F. Babonneau, O. du Merle, and J.-P. Vial. Solving Large-Scale Linear Multicommodity Flow Problems with an Active Set Strategy and Proximal-ACCPM. *Operations Research*, 54(1):184–197, 2006.

[Ber95] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.

[BG02] H. Bar-Gera. Origin-based algorithms for the traffic assignment problem. *Transportation Science*, 36:398–417, 2002.

[BG07] H. Bar-Gera. Transportation network test problems, 2007. www.bgu.ac.il/∼bargera/tntp/.

[BI04] D. Bienstock and G. Iyengar. Solving fractional packing problems in $O^*(\frac{1}{\epsilon})$ iterations. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 146–155, 2004.

[Bie02] D. Bienstock. *Potential Function Methods for Approximately Solving Linear Programming Problems: Theory and Practice*. Kluwer Academic Publishers, Boston, 2002.

[Bix02] R. Bixby. Solving real-world linear programs: A decade and more of progress. *Operations Research*, 50(1):3–15, 2002.

[BMN05] D. E. Boyce, H. S. Mahmassani, and A. Nagurney. A retrospective on Beckmann, McGuire and Winsten's Studies in the Economics of Transportation. *Papers in Regional Science*, 84:85–103, 2005.

[BMW56] M. J. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics of Transportation*. CT: Yale University Press, 1956.

[Bra68] D. Braess. Über ein Paradoxon aus der Verkehrsplanung. *Unternehmensforschung 12*, pages 258–268, 1968.

[BT97] D. Bertsimas and J. N. Tsitsiklis. *Introduction to linear optimization*. Athena Scientific, 1997.

[BTMN01] A. Ben-Tal, T. Margalit, and A. Nemirovski. The ordered subsets mirror descent optimization method with applications to tomography. *SIAM Journal on Optimization*, 12(1):79–108, 2001.

[Bun05] Bundesamt für Raumentwicklung (ARE). Fahrtenmatrix Personenverkehr 2000, 2005.

[Bur64] Bureau of Public Road. *Traffic assignment manual*. U.S. Department of Commerce, Urban Planning Division, Washington, DC, 1964.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[CD94] R. Cominetti and J.-P. Dussault. Stable exponential-penalty algorithm with superlinear convergence. *Journal of Optimization Theory and Applications*, 83(2):285–309, 1994.

[CE05] F. A. Chudak and V. Eleuterio. Improved approximation schemes for linear programming relaxations of combinatorial optimization problems. In *Proceedings of the 11th Integer Programming and Combinatorial Optimization Conference*, pages 81–96, 2005.

[CG06] F. A. Chudak and M. Guarisco. High confidence level solutions for stochastic optimization problems, 2006. Internal Report, IFOR, ETH Zurich.

[Chu98] F. A. Chudak. Improved approximation algorithms for uncapacitated facility location. In *Proceedings of the 6th Integer Programming and Combinatorial Optimization Conference*, pages 180–194, 1998.

[Chu03] F. A. Chudak. A simple approximation scheme for the linear programming relaxation of the uncapacitated facility location problem based on a recent algorithm of Garg and Khandekar. Manuscript, 2003.

[Chu05] F. A. Chudak. Special topics in linear programming. Lecture notes, ETH Zurich, 2005.

[CN07] F. A. Chudak and K. Nagano. Efficient solutions to relaxations of combinatorial problems with submodular penalties via the Lovász extension and non-smooth convex optimization. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 79–88, 2007.

[Cpl] Cplex11. ILOG CPLEX 11.0 User's Manual. September 2007, ILOG S.A. and ILOG Inc.

[CS03] F. A. Chudak and D.B. Shmoys. Improved approximation algorithms for the uncapacitated facility location problem. *SIAM Journal on Computing*, 33(1):1–25, 2003.

[CSSM04] J.R. Correa, A.S. Schulz, and N.E. Stier-Moses. Selfish routing in capacitated networks. *Mathematics of Operations Research*, 29(4):961–976, 2004.

[Dan63] G. B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, 1963.

[FG99] A. Frangioni and G. Gallo. A bundle type dual-ascent approach to linear multicommodity min cost flow problems. *INFORMS Journal on Computing*, 11(4):370–393, 1999.

[Fle00] L. Fleischer. Approximating fractional multicommodity flow independent of the number of commodities. *SIAM Journal on Discrete Mathematics*, 13(4):505–520, 2000.

[GHPS07] A. Gilpin, S. Hoda, J. Peña, and T. Sandholm. Gradient-based algorithms for finding nash equilibria in extensive form games. In *Proceedings of the 3rd International Workshop on Internet and Network Economics Conference*, pages 57–69, 2007.

[GK96] M. D. Grigoriadis and L. G. Khachiyan. Approximate minimum-cost multicommodity flows in $\tilde{O}(\epsilon^{-2}KNM)$ time. *Mathematical Programming*, 75(3):477–482, 1996.

[GK98] N. Garg and J. Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, pages 300–309, 1998.

[GK02] N. Garg and R. Khandekar. Fast approximation algorithms for fractional steiner forest and related problems. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 500–509, 2002.

[GOPS98] A. V. Goldberg, J. D. Oldham, S. A. Plotkin, and C. Stein. An implementation of a combinatorial approximation algorithm for minimum-cost multicommodity flow. In *Proceedings of the 6th Conference on Integer Programming and Combinatorial Optimization*, pages 338–352, 1998.

[IPS05] G. Iyengar, D. J. Phillips, and C. Stein. Approximation algorithms for semidefinite packing problems with applications to maxcut and graph coloring. In *Proceedings of the 11th Integer Programming and Combinatorial Optimization Conference*, pages 152–166, 2005.

[Kar84] N. Karmakar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[Kar02] G. Karakostas. Faster approximation schemes for fractional multicommodity flow problems. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 166–173, 2002.

[Kha79] L. G. Khachian. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

[KM72] V. Klee and G. J. Minty. How good is the simplex algorithm. In O. Shisha, editor, *Inequalities III*, pages 159–175. Academic Press, New York, NY, 1972.

[KP99] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, pages 404–413, 1999.

[KTK79] M. K. Kozolov, S. P. Tarasov, and L. G. Khachian. Polynomial solvability of convex quadratic programming. *Soviet Mathematics Doklady*, 20(5):1108–1111, 1979.

[KV00] B. Korte and J. Vygen. *Combinatorial Optimization, Theory and Algorithms*. Springer-Verlag, Berlin, 2000.

[LNN95] C. Lemarechal, A. Nemirovskii, and Y. Nesterov. New variants of bundle methods. *Mathematical Programming*, 69:111–148, 1995.

[LP99] T. Larsson and M. Patriksson. Side constrained traffic equilibrium models – analysis, computation and applications. *Transportation Research B*, 33:233–264, 1999.

[Mar99] R. K. Martin. *Large scale Linear and integer optimization. A unifed approach*. Kluwer Academic Publishers, 1999.

[MF90] P. Mirchandani and R. Francis, eds. *Discrete Location Theory*. John Wiley and Sons, Inc., New York, 1990.

[Min84] M. Minoux. A polynomial algorithm for minimum quadratic cost flows. In *European Journal of Operational Research*, volume 18, pages 377–387, 1984.

[Nag93] A. Nagurney. *Network economics: A variational inequality approach*. Kluwer Academic Publishers, 1993.

[NdP00] Y. Nesterov and A. de Palma. Stable dynamics in transportation systems (CORE Discussion Paper No. 2000/27). Technical report, CORE, UCL, 2000.

[NdP03] Y. Nesterov and A. de Palma. Stationary dynamic solutions in congested transportation networks: summary and perspectives. *Networks and Spatial Economics*, 3(3):371–395, 2003.

[Nes03] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2003.

[Nes05a] Y. Nesterov. Excessive gap technique in nonsmooth convex minimization (CORE Discussion Paper No. 2003/35, 2003). *SIAM Journal on Optimization*, 16(1):235–249, 2005.

[Nes05b] Y. Nesterov. Minimizing functions with bounded variation of subgradients (CORE Discussion Paper No. 2005/79). Technical report, CORE, UCL, 2005.

[Nes05c] Y. Nesterov. Smooth minimization of non-smooth functions (CORE Discussion Paper No. 2003/12, 2003). *Mathematical Programming*, 103(1):127–152, 2005.

[Nes07] Y. Nesterov. Gradient methods for minimizing composite objective function (CORE Discussion Paper No. 2007/76). Technical report, CORE, UCL, 2007.

[Nes09] Y. Nesterov. Primal-dual subgradient methods for convex problems (CORE Discussion Paper No. 2005/67, 2005). *Mathematical Programming*, 120(1):221–259, 2009.

[NN94] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM Publications, 1994.

[NY83] A. Nemirovskii and D. Yudin. *Informational complexity and efficient methods for solution of convex extremal problems*. J. Wiley and Sons, 1983.

[Pat94] M. Patriksson. *The Traffic Assignment Problem – Models and Methods*. VPS, BV, Utrecht, The Netherlands, 1994.

[Pol87] B. T. Polyak. *Introduction to Optimization*. Optimization Software, Inc., Publications Division, 1987.

[PST95] S. A. Plotkin, D. B. Shmoys, and É. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20(2):257–301, 1995.

[Roc70] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[Rou03] T. Roughgarden. The price of anarchy is independent on the network topology. *Journal of Computer and System Sciences*, 67(2):341–364, 2003.

[RP06] M. G. C. Resende and P. M. Pardalos. *Handbook of Optimization in Telecommunications*. Spinger Science + Business Media, New York, 2006.

[RT00] T. Roughgarden and E. Tardos. How bad is selfish routing? In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 93–102, 2000.

[RT04] T. Roughgarden and E. Tardos. Bounding the inefficiency of equilibria in nonatomic congestion games. *Games and Economic Behavior*, 47(2):389–403, 2004.

[Rud07] A. Rudyk. User equilibrium delays in the traffic equilibrium problem. Bachelor thesis, IFOR, ETH Zurich, 2007.

[Van96] R. J. Vanderbei. *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, 1996.

[VIS06] VISUM. VISUM-IV/ÖV 9.44, 2006. PTV Planung Transport Verkehr AG, Germany.

[War52] J G. Wardrop. Some theoretical aspects of road traffic research. In *Proceedings of the Institute of Civil Engineers, London, Part II, Vol.1*, pages 325–378, 1952.

[Wri97] S. J. Wright. *Primal-dual interior-point methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1997.

[You00] N. Young. K-medians, facility location, and the Chernoff-Wald bound. In *Proceedings of the 11th ACM-SIAM Symposium on Discrete Algorithms*, pages 86–95, 2000.

[You01] N. Young. Sequential and parallel algorithms for mixed packing and covering. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 538–546, 2001.

Curriculum Vitae

Personal Data

Name	Vânia Lúcia Dos Santos Eleutério
Date of birth	June 20, 1978
Nationalities	Portuguese, Swiss
Civil status	Single

Education

10/03 – 01/09	Ph.D.–Student in Mathematics, ETH Zurich Dissertation written at the Institute for Operations Research under the supervision of Prof. Dr. H.–J. Lüthi.
10/99 – 03/03	Studies in Mathematics, ETH Zurich
10/97 – 10/99	Studies in Mathematics, ETH Lausanne
08/93 – 06/97	High School, Geneva

Work Experience

04/03 – 12/08	Research and Teaching Assistant, Student Adviser, Department of Mathematics, ETH Zurich