Hyperbolic smoothing in nonsmooth optimization and applications

Alia Al Nuaimat

This thesis is submitted in total fulfilment of the requirement for the degree of Doctor of Philosophy

School of Science, Information Technology and Engineering, Faculty of Science,
Federation University Australia
PO Box 663
University Drive, Mount Helen
Ballarat, VIC 3353, Australia.

2014

Abstract

Nonsmooth nonconvex optimization problems arise in many applications including economics, business and data mining. In these applications objective functions are not necessarily differentiable or convex. Many algorithms have been proposed over the past three decades to solve such problems. In spite of the significant growth in this field, the development of efficient algorithms for solving this kind of problem is still a challenging task.

The subgradient method is one of the simplest methods developed for solving these problems. Its convergence was proved only for convex objective functions. This method does not involve any subproblems, neither for finding search directions nor for computation of step lengths, which are fixed ahead of time. Bundle methods and their various modifications are among the most efficient methods for solving nonsmooth optimization problems. These methods involve a quadratic programming subproblem to find search directions. The size of the subproblem may increase significantly with the number of variables, which makes the bundle-type methods unsuitable for large scale nonsmooth optimization problems. The implementation of bundle-type methods, which require the use of the quadratic programming solvers, is not as easy as the implementation of the subgradient methods. Therefore it is beneficial to develop algorithms for nonsmooth nonconvex optimization which are easy to implement and more efficient than the subgradient methods.

In this thesis, we develop two new algorithms for solving nonsmooth nonconvex optimization problems based on the use of the hyperbolic smoothing technique and apply them to solve the pumping cost minimization problem in water distribution. Both algorithms use smoothing techniques. The first algorithm is designed for solving finite minimax problems. In order to apply the hyperbolic smoothing we reformulate the objective function in the minimax problem and study the relationship between the original minimax and reformulated problems. We also study the main properties of the hyperbolic smoothing function. Based on these results an algorithm for solving the finite minimax

problem is proposed and this algorithm is implemented in GAMS. We present preliminary results of numerical experiments with well-known nonsmooth optimization test problems. We also compare the proposed algorithm with the algorithm that uses the exponential smoothing function as well as with the algorithm based on nonlinear programming reformulation of the finite minimax problem.

The second nonsmooth optimization algorithm we developed was used to demonstrate how smooth optimization methods can be applied to solve general nonsmooth (nonconvex) optimization problems. In order to do so we compute subgradients from some neighborhood of the current point and define a system of linear inequalities using these subgradients. Search directions are computed by solving this system. This system is solved by reducing it to the minimization of the convex piecewise linear function over the unit ball. Then the hyperbolic smoothing function is applied to approximate this minimization problem by a sequence of smooth problems which are solved by smooth optimization methods. Such an approach allows one to apply powerful smooth optimization algorithms for solving nonsmooth optimization problems and extend smoothing techniques for solving general nonsmooth nonconvex optimization problems. The convergence of the algorithm based on this approach is studied. The proposed algorithm was implemented in Fortran 95. Preliminary results of numerical experiments are reported and the proposed algorithm is compared with an other five nonsmooth optimization algorithms. We also implement the algorithm in GAMS and compare it with GAMS solvers using results of numerical experiments.

Statement of Authorship

Except where explicit reference is made in the text of the thesis, this professional thesis contains no material published elsewhere or extracted in whole or in part from a thesis by which I have qualified for or been awarded another degree or diploma. No other persons work has been relied upon or used without due acknowledgment in the main text and bibliography of the thesis.

Signed: _____ Date: _____

Acknowledgement

First of all, I would like to acknowledge my great appreciation to my principal supervisor Associate Professor Adil Bagirov. It has been a great pleasure working with Associate Professor Bagirov over the years; learning from such a great Mathematician. I would like to thank him for accepting me as one of his students and for his great support during my PhD. His care, support, knowledge, availability and guidance was of inestimable value. The appreciation I have toward him is beyond words. The whole time I have spent as his student has definitely been a challenging and stimulating part of my life. Also, I would like to thank both my associate supervisors, Dr David Yost and Dr Andrew Barton, for their valued encouragement and ongoing advice with my research.

In addition to the invaluable support of all my supervisors, this PhD thesis would not have been possible without the financial support of the School of Science, Information Technology and Engineering (SITE). Then too, I would like to recognize Professor Sidney Morris, former head of Graduate School of Information Technology and Mathematical Science (GSITMS), for openly accepting me and financially supporting my studies, from the time of my arrival in Ballarat Australia. I also greatly appreciate the support and help from Dean of SITE, Professor John Yearwood, for allowing me study my PhD study in such encouraging and supportive environment.

As well as all the preceding, it would be remiss of me not to also acknowledge Associate Professor Andrew Stranieri, the Director of Centre for Informatics and Applied Optimization (CIAO), for his direction and timely advice which was always so positive and encouraging. I also wish to sincerely thank Mr Frank Williams for his ongoing efforts and meticulous proof-reading of my research, thereby enriching the final outcome and quality of my thesis.

All of the persons highlighted herein have been most professional in their approach and each in their own special way has enhanced the quality of my research, plus the timely completion of my Doctoral thesis.

Dedication

I dedicate this thesis to my husband Suleiman, my son Talal, my daughter Sally and my parents.

List of publication

Journal papers

1. Bagirov, A.M., Al Nuiamat, A., and Sultanova, N., Hyperbolic smoothing function method for minimax problems, *Optimization*, 62(6), 2013, 759–782.

2. Bagirov, A. M., Jin, L., Karmitsa, N., Al Nuaimat, A., and Sultanova, N., Subgradient method for nonconvex nonsmooth optimization, *Journal of Optimization Theory and Applications*, 157(2), 2013, 416–435.

3. Bagirov, A. M., Barton, A. F., Mala-Jetmarova, H., Al Nuaimat, A., Ahmed, S. T., Sultanova, N., and Yearwood, J., An algorithm for minimization of pumping costs in water distribution systems using a novel approach to pump scheduling, *Mathematical and Computer Modelling*, 57(34), 2013, 873–886.

4. Bagirov, A. M., Ahmed, S. T., Barton, A. F., Mala-Jetmarova, H., Al Nuaimat, A., Sultanova, N. Comparison of metaheuristic algorithms for pump operation optimization. *14th Water Distribution Systems Analysis Conference*, 2012.

5. Bagirov, A. M., Barton, A. F., Mala-Jetmarova, H., Al Nuaimat, A., Ahmed, S. T., Sultanova, N., and Yearwood, J., Minimization of pumping costs in water distribution systems using explicit and implicit pump scheduling, *Hydrology and Water Resources Symposium*, 2012.

6. Bagirov, A.M., Ozturk, G., Sultanova, N., and Al Nuaimat, A., Nonsmooth nonconvex optimization via smooth optimization, (Submitted).

Conference and workshop presentations

1. Al Nuaimat, A., A Generalized Subgradient Algorithm for Unconstrained Nonsmooth, Nonconvex Optimization, *The 2011 IFORS conference in Nonsmooth Optimization III*,

2. Al Nuaimat, A., An algorithm for minimization of pumping costs in water distribution systems using a novel progressive approach for pump scheduling, *The 9th EUROPT Workshop on Advances in Continuous Optimization* July, 2011.

Contents

List of Tables

List of Figures

Introduction

Optimization models are widely used in solving many practical problems including those in economics, operational research, mechanics and optimal control. In many applications optimization problems are nonsmooth, that is in these problems objective and/or constraint functions have discontinuous gradients. Nonsmooth optimization problems are among most difficult in optimization. Over the last four decades a great deal of effort has been devoted to design algorithms for solving nonsmooth optimization problems. To date, problems of nonsmooth optimization have been mainly tackled by variants of the bundle methods [37, 44, 48, 61, 64, 65, 96], subgradient (including the space dilation) methods [88] and algorithms based on smoothing techniques [81].

The subgradient method is one of the simplest methods for solving nonsmooth optimization problem. It was originally developed by N. Shor and then was modified by many authors (see [17, 83, 88] and more recent papers [1, 13, 67, 68, 69]). Its convergence was proved only for convex objective functions. The subgradient method uses one subgradient and one function evaluation at each iteration. It does not involve any subproblems neither for finding search directions nor for computation of step lengths. Moreover, step lengths are fixed ahead of time. Therefore, it is easy to implement this method. Although this method is very slow it is well known that some of its modifications might be more successful for solving large scale problems than other nonsmooth optimization methods. For example space dilation was proposed by Shor[88] to accelerate the direction finding towards the minimum where a linear operator is constructed at each iteration to change the metric of the space (for more of these modifications see [68, 69]). In general, subgradient methods have several important limitations (e.g.lack of implementable stopping test, lack of decrease of the objective function at each iteration, possible poor rate of convergence, etc). Nevertheless, they are extremely popular among practitioners, because of their simplicity of implementation.

To date, bundle methods are considered to be the most efficient methods in nonsmooth optimiza-

tion. The fundamental idea of bundle methods is usually to approximate the subdifferential of the objective function by gathering subgradients from previous iterations in one bundle. By doing this, information about the local behavior of the objective function is obtained. The bundle methods are based on the use of convex models to the objective function and as a result it is efficient for minimization of convex functions, however it is not always efficient for minimization of nonconvex functions. These methods involve a quadratic programming subproblem to find search directions. The size of the subproblem may increase significantly as the number of variables increase which makes the bundle-type methods unsuitable for large scale nonsmooth optimization problems. Therefore most of the versions of the bundle method are not applicable for solving large scale nonsmooth optimization problems (convex or nonconvex). Recently, the limited memory bundle method [46, 45, 40, 41, 42]has been proposed where aggregate subgradients and Quasi-Newton updates for sparse problems are combined to find search directions. At each iteration of this method only three subgradients with a certain type of the quasi- Newton updates are used to find search directions. The limited memory bundle method is a hybrid of the variable metric bundle methods and the limited memory variable metric methods. The method exploits simple aggregation of subgradients, and calculates the search direction using a limited memory approach. As a result, the time-consuming quadratic subproblem appearing in standard bundle methods need not to be solved and the number of stored subgradients is independent of the dimension of the problem. The efficiency of this method has been proved by numerical results [45].

The quasisecant method was introduced in [7]. Unlike bundle methods this method does not rely on the convex model of the objective function f, instead it uses quasisecants which are overestimators of the objective function in some neighborhood of the current point. Results from [7] demonstrate that the quasisecant method is more efficient for solving nonconvex nonsmooth optimization problems than some versions of the bundle method. Moreover, results from [45] show that it is efficient for solving large scale problems. At the same time the quasisecant method requires more function and quasisecant (subgradient) evaluations than the bundle methods.

In [9], a subgradient method for nonsmooth nonconvex optimization problems called SUNNOPT

has been proposed to solve especially nonconvex optimization problems. Similar to the subgradient method, the SUNNOPT method does not contain any subproblem to find either descent direction or step length. A bundle of information of the objective function is used in some neighborhood of the iteration point and the direction finding procedure is very simple. Aggregation of only two subgradients or quasisecants are used in the search direction finding procedure. Versions of SUNNOPT using subgradients and quasisecants have been presented and numerical results are shown to prove its efficiency for both small and large scale problems.

Minimax problems arise in multiple disciplines including engineering design [89], control system design [21], economics [11], machine learning [4]. These problems are nonsmooth problems because of the presence of the "max". There are several different approaches to solve this problem (see, for example, [28, 33]). Moreover, conventional nonsmooth optimization algorithms such as the bundle methods and its variations can also be applied to solve this problem [5, 7, 44, 48, 61]. Any Minimax problem can also be reformulated as a nonlinear programming problem and therefore efficient nonlinear programming techniques can be applied to solve it. One of the methods to solve such problems is applying a smoothing technique to replace the original nonsmooth problem by an approximate smooth one. Smoothing methods bring these problems close to continuously differentiable programming problems which can be solved by the conventional smooth optimization methods. The main feature of smoothing methods is to approximate the nonsmooth problems by a sequence of parameterized smooth continuously differentiable problems. Different smoothing techniques have been developed to replace the maximum function in minimax problem by a smooth function including the exponential and hyperbolic smoothing functions. In [15] Ben-Tal and Teboulle introduce a general smoothing approach that utilizes recession functions to approximate a nondifferentiable optimization problem. It covers several types of problems including minimax problems using the so called recession function. The exponential smoothing function for the maximum function $f(x) = \max_{i=1,\ldots,m} f_i(x)$is as follows

$$F_\varepsilon(x) = \varepsilon \log \sum_{i=1}^{m} exp \frac{f_i(x)}{\varepsilon} \tag{1}$$

where ε is called the precision parameter. The exponential smoothing function has been commonly used to construct many smoothing approximation algorithms for problems with min and max functions. Several smoothing methods are based on using exponential smoothing function (1). The function (1) was first proposed in [50] within the structure of a penalty function method. The precision parameter in the exponential smoothing function may become extremely small too fast and cause ill-conditioning and floating-point overflow. In their paper [81] the authors made an effort to address this issue by introducing a feedback precision adjustment rule whereby the precision parameters are constructed by a subroutine. The goal is to ensure that the precision parameter remains large when away from the solution and is decreased when the solution is approached.

Another smoothing function is a hyperbolic smoothing function which is introduced for the function $\max\{0, t\}$ the first time in [97]:

$$\phi(t, \tau) = \frac{t + \sqrt{t^2 + \tau^2}}{2},$$

where $\tau > 0$ is the precision parameter. In [98], the authors consider a problem of covering plane domains by circles which is modeled as a min-max-min problem which enables them to take advantage of the hyperbolic smoothing function to develop a minimization algorithm which solves a sequence of approximating problems. In the paper [99] this technique was applied to solve the cluster analysis problem using its nonsmooth optimization formulation.

Despite some applications hyperbolic smoothing functions have not been studied extensively so far. In this thesis we study this smoothing technique in detail. In order to apply the hyperbolic smoothing to the finite maximum functions these functions are represented as a sum of the maximum of two functions by adding a new variable. We study the relationship between the set of stationary points of the latter function and that of the original maximum function. The new function is approximated using hyperbolic smoothing functions and differential properties of the approximating function are studied. It is demonstrated that smooth optimization solvers can be applied to minimize the approximating function. We design an algorithm for solving minmax problems using the hyper-

bolic smoothing. Furthermore, we present results of numerical experiments using two solvers from GAMS: CONOPT and SNOPT. We also compare these results with those obtained using exponential smoothing and also nonlinear programming reformulations of minimax problems. Such an approach allows one to solve the finite minimax problem using existing powerful smooth optimization solvers. The smooth function approximates the objective function and this approximation is controlled by the precision parameter(s).

Next we extend this method for solving general nonsmooth nonconvex optimization problems. In this approach the problem of finding search directions is reduced to the minimization of a convex piecewise linear function over the unit ball. The hyperbolic smoothing functions are applied to approximate the convex piecewise linear function by a smooth function which is minimized to find search directions. We present convergence results for the proposed algorithm. The algorithm is implemented in Fortran 95. Results of numerical experiments are reported and the proposed algorithm is compared with another five nonsmooth optimization algorithms. We also implement the algorithm in GAMS and compare it with GAMS solvers using results of numerical experiments.

The proposed methods is applied to solve Pump scheduling problem in water distribution system.

Outline of the Thesis

This thesis is organized into six chapters beginning with a background discussion in Chapter 1 which introduces the reader to nonsmooth optimization fields from nonsmooth optimization analysis to nonsmooth optimization theory. Chapter 2 presents an overview of nonsmooth optimization methods and Chapter 3 describes the application of the hyperbolic smoothing to the finite maximum functions. Then, Chapter 4 provides a description for solving general nonsmooth nonconvex problems. Chapter 5 describes the minimization of pumping costs in water distribution systems. Whilst, Chapter 6 concludes the thesis and gives recommendations for future research directions.

Chapter 1

Background

The chapter is organized as follows. First we introduce basic notations and definitions that will be used throughout the rest of this thesis, followed by basic definitions and results of Nonsmooth Analysis and Optimization. We also present some properties of the subgradients and subdifferentials for convex function. Following that, we give the extension of these concept to a nonconvex case. Finally, we give the necessary optimality conditions and linearizations for locally Lipschitz functions.

1.1 Notations and Definitions

$\mathbb{R}^n$ is the n-dimensional real Euclidean space, we denote by $x = (x_1, \ldots, x_n)$ a point in $\mathbb{R}^n$, and we denote by $\langle x, y\rangle := \sum_{i=1}^{n} x_i y_i$ an inner product of the two points $x, y \in \mathbb{R}^n$, and $\|\cdot\|$ denotes the associated Euclidean norm.

$S_1 := \{x \in \mathbb{R}^n : \|x\| = 1\}$ is the unit sphere, whereas $B_\varepsilon(x) := \{y \in \mathbb{R}^n : \|y - x\| < \varepsilon\}$ is the open ball centered at x with the radius $\varepsilon > 0$. Furthermore, an open ball with radius $\varepsilon > 0$ centered at 0 is denoted by $B_\varepsilon := B_\varepsilon(0_n)$.

A set $S \subset \mathbb{R}^n$ is called convex if $\lambda x + (1-\lambda)y \in S$ for any $x, y \in S$ and $\lambda \in [0, 1]$. A linear combination $\sum_{i=1}^{k} \lambda_i x_i$ is called a convex combination of points $x_1, ..., x_k \in \mathbb{R}^n$ if each $\lambda_i \geq 0$ and $\sum_{i=1}^{k} \lambda_i = 1$. The *convex hull* of a set S, denoted by conv S, is the smallest convex set containing S.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex function if

$$f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda) f(y)$$

for all $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be Locally Lipschitz at (or near) the point x if there exist a numbers $\delta, L > 0$ such that

$$|f(y) - f(z)| \leq L\|f(y) - f(z)\|$$

for all $y, z \in B_\delta(x)$.

1.2 Nonsmooth Analysis

The theory of Nonsmooth analysis was first developed for convex functions. Here we first go over a number of definitions and results for convex analysis. Second, define the subgradient and also the subdifferential of the convex function, and then finally we generalize these results to nonconvex locally Lipschitz functions.

Definition 1. *The directional derivative $f'(x, d)$ of the function f: $\mathbb{R}^n \to \mathbb{R}$ at x in the direction of $d \in \mathbb{R}^n$ is defined as*

$$f'(x; d) = \lim_{t \downarrow 0} \frac{f(x + td) - f(x)}{t}$$

Definition 2. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is called upper semicontinuous at $x \in \mathbb{R}^n$ if for every sequence $\{x_k\}$ converging to x*

$$\limsup_{k \to \infty} f(x_k) \leq f(x),$$

and is called lower semicontinuous if

$$f(x) \leq \liminf_{k \to \infty} f(x_k),$$

Definition 3. *The subdifferential of a convex function $f : \mathbb{R}^n \to \mathbb{R}$ at $x \in \mathbb{R}^n$ is the set*

$$\partial_c f(x) = \{g \in \mathbb{R}^n | f(y) \geq f(x) + \langle g, y - x \rangle, \quad \forall y \in \mathbb{R}^n\}$$

where $g \in \partial_c f(x)$ is called a subgradient of f at x.

For any differentiable function , $\partial_c f(x) = \{\nabla f(x)\}$.

Definition 4. *Let $\varepsilon > 0$. The ε-subdifferential of the convex function $f : \mathbb{R}^n \to \mathbb{R}$ is the set*

$$\partial_\varepsilon f(x) := \{g \in \mathbb{R}^n | f'(x) \geq f(x) + \langle g, x' - x\rangle - \varepsilon, \ \ \forall x' \in \mathbb{R}^n\}$$

each element $\xi \in \partial_\varepsilon f(x)$ is called an ε-subgradient of the convex function f at x.

Theorem 1 (Rademacher)**.** [25] *Let S be an open set. A function $f : S \to \mathbb{R}$ that is locally Lipschitz continuous on S is differentiable almost everywhere on S.*

The next theorem presents the relationship between subdifferential and the directional derivative.

Theorem 2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex function at $x \in R^n$. Then for all $x \in R^n$*

(i) $\partial f(x)$ is a nonempty, convex, and compact set such that $\partial f(x) \subseteq B_L(0)$,

(ii) $\partial_c f(x) = \{g \in R^n : f'(x, d) \geq \langle g, d\rangle\}, \quad \forall d \in R^n$,

(iii) $f'(x, d) = \max\limits_{g \in \partial_c f(x)} \{\langle g, d\rangle\}, \quad \forall d \in R^n$,

where L is a Lipschitz constant of f.

The classical directional derivative and subdifferential, in general, do not exist for locally Lipschitz function. F. Clarke introduced the generalizations of directional derivatives and subdifferentials for locally Lipschitz functions [25].

Definition 5. *The generalized directional derivative $f^0(x, d)$ of the locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ at $x \in R^n$ in the direction $d \in \mathbb{R}^n$ is defined as*

$$f^\circ(x; d) = \limsup_{\substack{y \to x \\ t \downarrow 0}} \frac{f(y + td) - f(y)}{t}.$$

Definition 6. *(Clarke subdifferential)* [25] *The subdifferential $\partial f(x)$ of a locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ at a point $x \in \mathbb{R}^n$ is given by*

$$\partial f(x) := \{g \in \mathbb{R}^n | f^\circ(x, d) \geq \langle g, d\rangle \quad \forall d \in \mathbb{R}^n\}.$$

Here $g \in \partial f(x)$ is called a subgradient of f at x.

According to Rademacher theorem locally Lipschitz functions are differentiable almost everywhere and their subdifferential can also be defined as follows:

$$\partial f(x) = \text{conv} \left\{ \lim_{i \to \infty} \nabla f(x_i) \mid x_i \to x \text{ and } \nabla f(x_i) \text{ exists} \right\}.$$

Definition 7. *A locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ is called regular at x iff it is directionally differentiable at x and*

$$f'(x, d) = f^0(x, d), \ \ \forall d \in \mathbb{R}^n.$$

The following theorem summarizes some properties of the Clarke subdifferential.

Theorem 3. [61] *Let $f : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz function at $x \in R^n$ with constant K. Then*

(i) $\partial f(x)$ is nonempty, convex, compact set such that $\partial f(x) \subseteq B_K(0)$,

(ii) $f^\circ(x; v) = \max\{\langle \xi, v \rangle | \xi \in \partial f(x), \ \ \forall x \in \mathbb{R}^n\}$,

(iii) the mapping $\partial f(\cdot)$ is upper semi-continuous.

Note that, for convex function the Clarke subdifferential coincides with the subdifferential from Definition (3). This means that the Clarke subdiiferential is the generalization of the subdifferential for convex functions. Furthermore, for any locally Lipschitz function f, we have $\nabla f(x) \in \partial f(x)$.

The *Goldstein* ε-subdifferential for locally Lipschitz functions is defined as follows.

Definition 8. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a locally Lipschitz function at $x \in R^n$ $\varepsilon > 0$, then the Goldstein ε-subdifferential of the Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ is the set*

$$\partial_\varepsilon^G f(x) := conv\{\partial f(y) | y \in \bar{B}_\varepsilon(x)\}$$

where, $g \in \partial_\varepsilon^G f(x)$ is called an ε-subgradient of the function f at x.

1.3 Nonsmooth Optimization Theory

In this section, we present basic results from nonsmooth optimization including necessary and sufficient conditions for a minimum. Moreover, linearization of local Lipschitz functions using subgradients will be discussed.

We consider the following unconstrained optimization problem:

$$\text{minimize } \ f(x) \ \text{ subject to } \ x \in \mathbb{R}^n, \tag{1.1}$$

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be locally Lipschitz.

Definition 9. *A point $x^\star \in \mathbb{R}^n$ is called a global minimum of f if it satisfies $f(x^\star) \leq f(x)$ for all $x \in R^n$.*

Definition 10. *A point $x^\star \in \mathbb{R}^n$ is called a local minimum of f if there exists $\epsilon > 0$ such that $f(x^\star) \leq f(x)$ for all $x \in B_\epsilon(x^\star)$.*

Theorem 4. *Let f: $\mathbb{R}^n \to \mathbb{R}$ be a locally Lipschitz at $x^\star \in \mathbb{R}^n$ and attains its local minimum at $x^\star$. Then*

(i) $0 \in \partial f(x^\star)$ *and*

(ii) $0 \in \partial_\varepsilon^G f(x^\star)$,

(iii) $f^\circ(x^\star, d) \geq 0, \ \ \forall d \in \mathbb{R}^n$.

The point $x^* \in \mathbb{R}^n$ is called stationary iff $0 \in \partial f(x^*)$. Stationarity is a necessary condition for local optimality and, in the convex case, it is also sufficient for global optimality.

In order to develop nonsmooth optimization algorithms for unconstrained optimization problem (1.1), linearization of the objective function needs to be built. Using these linearizations we are able to construct a piecewise linear local approximation to the unconstrained optimization problem.

Definition 11. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a locally Lipschitz function at x and let $\xi \in \partial f(x)$ be an arbitrary subgradient. Then the ξ-linearization of f at x is the function $\bar{f}_\xi : \mathbb{R}^n \to \mathbb{R}$ defined by*

$$\bar{f}_\xi(y) := f(x) + \langle \xi, y - x \rangle, \qquad \forall y \in \mathbb{R}^n$$

and the linearization of f at x is the function $\hat{f}_x$: $\mathbb{R}^n \to \mathbb{R}$ such that

$$\hat{f}_x(y) := \max\{\bar{f}_\xi(y) | \xi \in \partial f(x)\}, \ \ \forall y \in \mathbb{R}^n.$$

Some basic properties of the linearization $\hat{f}_x$ are presented in the following theorem.

Theorem 5. *Let f: $\mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz at x. Then the linearization $\hat{f}_x$ is convex and*

(i) $\hat{f}_x(x) = f(x)$,

(ii) $\hat{f}_x(y) = f(x) + f^\circ(x; y - x), \ \ \forall y \in \mathbb{R}^n$,

(iii) $\partial \hat{f}_x(x) = \partial f(x)$.

Finding descent direction for a locally Lipschitz function is an important step of any optimization algorithm. We first give the general definition of a descent direction for any function, then modify it using generalized directional derivatives and subdifferential.

Definition 12. *The direction $d \in \mathbb{R}^n$ is called a descent direction for the function $f : \mathbb{R}^n \to \mathbb{R}$ at $x \in \mathbb{R}^n$ if there exists $\varepsilon > 0$ such that $f(x + td) < f(x), \ \ \forall \ \ t \in (0, \varepsilon]$.*

Theorem 6. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz at x, the direction $d \in \mathbb{R}^n$ is a descent direction for f at x if any of the following holds:*

(i) $f^\circ(x; d) < 0$,

(ii) $\langle \xi, d \rangle < 0, \ \ \forall \xi \in \partial f(x)$,

(iii) $\langle \xi, d \rangle < 0, \ \ \forall \xi \in \partial_\varepsilon^G f(x)$,

(iv) d *is a descent for the linearization* $\hat{f}_x$.

Chapter 2

Nonsmooth Optimization Methods

We consider the following unconstrained minimization problem:

$$\text{minimize } f(x) \text{ subject to } x \in \mathbb{R}^n, \tag{2.1}$$

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be locally Lipschitz and this function is not necessarily either differentiable or convex.

Numerous algorithms have been developed for solving problem (2.1), with similar basic structure. Their main point of difference is based on the specification of search directions and the selection of step size. The general structure of an iterative algorithm for solving the nonsmooth optimization problem (2.1) can be described as follows.

Step 1. Initialization Step: Choose a starting point $x_1 \in R_n$. Set the iteration counter to i:=1.

Step 2. Stopping criterion: If the optimality condition at x_i is satisfied then stop. Otherwise, go to step 3.

Step 3. Direction finding: Find a search direction d_i using either the subgradient $g_i \in \partial f(x_i)$ or some approximation to the subdifferential $\partial f(x_i)$.

Step 4. Line search: Select an appropriate step size $\lambda_i > 0$, and compute $x_{i+1} = x_i + \lambda_i d_i$. Increase i by 1, and go to step 2.

The desirable choices of the direction d_i, and the step size λ_i, have been the principal points of research focus in solving Problem (2.1). There exist two types of numerical techniques that are used

to solve nonsmooth optimization problems: deterministic methods and stochastic methods. Deterministic optimization algorithms guarantee under some assumptions that, starting from a given initial point, an algorithm converges with deterministic steps in a finite number of iterations. On the other hand, stochastic methods cannot adopt the same methodology. Stochastic methods are not guaranteed to achieve the same solution in any single run. Deterministic algorithms have often been shown to be better than stochastic ones at solving large scale nonsmooth nonconvex problems at a considerable computational cost and quality of the solution [53]. In this thesis, we will focus on Deterministic optimization methods. The most popular deterministic optimization algorithms are accordingly divided into three main classes; subgradient methods, bundle methods and methods based on smoothing techniques. In this chapter we will review some of these algorithms.

2.1 Subgradient methods

The subgradient method was originally developed by N. Shor in the mid 1960s and then modified by many authors (see [17, 83, 88] and more recent papers [1, 13, 67, 68, 69]).

It generalizes the steepest descent method of smooth optimization by replacing the gradient by an arbitrary subgradient. The subgradient method uses one subgradient and one function evaluation at each iteration. It does not involve any subproblems either for finding search directions or for computation of step lengths. Therefore, it is easy to implement. In addition it can be immediately applied to a far wider variety of problems. The memory requirement of subgradient methods can be much smaller than the bundle type methods, which means it can be used for extremely large problems for which bundle type methods cannot be used. Although the subgradient method is very slow it is well known that some of its modifications might be more successful for solving large scale problems than other nonsmooth optimization methods [68, 69]. The convergence of the subgradient method was proved only for convex problems.

Consider the minimization problem (2.1). In the smooth case, it is convenient to use the steepest

descent method which takes the ant-gradient direction with fixed step size.

$$d_k = -\nabla f(x_k) \tag{2.2}$$

where $\nabla f(x_k)$ is the gradient at the current iteration point.

The steepest descent method is one of the simplest optimization methods, though easy to implement, however, this method has the disadvantage of the large number of zigzag moves [55]. As a result, the conjugate gradient method has been developed, which is more efficient where gradients at previous iteration points are taken into account to compute the descent search direction. The conjugate gradient method has improved convergence rate the steepest descent method.

In the nonsmooth case of Problem (2.1), the gradient may not always exist at x. We assume that the objective function f is a locally Liptchitz function, where at least one subgradient can be calculated at any given point x, which is a mild assumption about most nonsmooth optimization problems.

The subgradient method is a generalization of steepest descent method (2.2) for smooth optimization problems. The idea of the subgradient method is quite simple by replacing the gradient $\nabla f(x_k)$ by an arbitrary subgradient $\xi_k \in \partial f(x_k)$. If x_k is not a stationary point then $\xi_k \neq 0$ and we compute the search direction by normalizing the subgradient ξ_k:

$$d_k = -\xi_k / \|\xi_k\|. \tag{2.3}$$

Then the subgradient algorithm can be constructed as follows

Algorithm 1. *Subgradient algorithm*

Step 1. Choose a starting point $x_1 \in \mathbb{R}_n$. Set k:=1

Step 2. Given x_k , calculate $f(x_k)$, and an arbitrary subgradient $\xi_k \in \partial f(x_k)$. If the stopping criterion is satisfied, then stop.

Step 3. compute $d_k = -\xi_k / \|\xi_k\|$.

Step 4. Select an appropriate step length $\lambda_k > 0$.

Step 5. Set $x_{k+1} := x_k + \lambda_k d_k$, *put* $k := k + 1$ *and go to step 2.*

There are some requirements for the step size λ_k to guarantee the global convergence. The step length λ_k should be defined so that to satisfy the following conditions:

1. The constant step size λ_k is chosen as a sufficiently small.

2. The step size λ_k satisfies the following condition:

$$\lambda_k > 0, \sum_{k=0}^{\infty} \lambda_k = \infty.$$

The convergence of subgradient methods is proved for only convex problems. To our best knowledge such proof does not exist for nonconvex nonsmooth functions. Moreover, the subgradient method does not guarantee the decrease of the objective at each iteration. In addition, subgradient method does not have practical termination criteria other than a maximum limit on the number of iterations. Poor performance of subgradient methods can happen when the direction is almost orthogonal to a direction pointing toward a minimum. Because of these difficulties with the choice of the step length, there have been many attempts to improve the search and step length [12, 38, 82, 88]. The space dilation methods were introduced [88], which can be considered as a nondifferentiable version of Quasi-Newton methods. These methods are based on accommodating previous iteration information. Similar to the use of the Hessian matrix or its approximation in quasi-Newton approaches, a suitable matrix is generated to multiply the subgradient in order to deflect the direction toward an optimal direction. One approach uses space dilation along the gradient, another approach uses space dilation along the difference of the two successive subgradients.

2.2 Cutting plane methods

Cutting-plane methods were proposed by Cheney and Goldstein [23] as well as by Kelley [47]. The cutting plane methods rely on construction of the lower approximation (underestimation) of the

objective function. This approximation is used to find the minimum of the original function. They are based on minimization of piecewise-affine approximation of the objective function, in which a new direction is obtained from the set of the previous subgradients, as opposed to using only one subgradient at a time, without a memory of past iterations on subgradient methods. The idea behind cutting plane methods is to approximate f using a piecewise linear function, in other words, to replace f by a so-called cutting plane model. At the current iteration k, the piecewise linear approximation of the objective function f can be defined as follows

$$\hat{f}^k(x) := \max\{f(x_j) + \langle \xi_j, x - x_j \rangle | j \in J_k\}, \quad x \in \mathbb{R}^n$$

where ξ_j is the subgradient at the trial points x_j around the current point x_k, $\hat{f}^k(x)$ is an underestimator for convex function f and J_k is a set of subgradients calculated so far. Note that the idea of the Cutting plane is to replace the original function f by its approximation $\hat{f}^k(x)$. The next iterate is then defined where the search direction d_k is calculated by solving the following minimization problem:

$$\min \hat{f}^k(x_k + d) - f(x_k) \quad \text{subject to} \quad d \in \mathbb{R}^n \tag{2.4}$$

which can be modified to the following linear problem.

$$\begin{aligned} \min \quad & \upsilon \\ \text{st.} \quad & \upsilon \geq f(x_j) + \langle \xi_j, x - x_j \rangle, \quad j \in J_k, \end{aligned}$$

The cutting plane method does not always have a solution and is not used in practice due to poor convergence results. Furthermore, Cutting plane method can be very slow when new iterates move too far away from the previous ones. In order to improve the rate of convergence we can obtain the search direction by solving the following local subproblem

$$\begin{aligned} \min \quad & \upsilon + \frac{1}{2}\|d\|^2 \\ \text{st.} \quad & \upsilon \geq -\alpha_j^k + \xi_j^T d \quad \text{for all } j \in J_k, \end{aligned} \tag{2.5}$$

where the regularizing quadratic penalty term is added to guarantee the existence of the solution d_k and to keep the approximation local enough, where the search direction can obtain as follows

$$d_k = -\sum_{j \in J_k} \lambda_j^k \xi_j$$

It is easy to see that the search direction at x_k is obtained as a convex combination of some subgradients at the points y_j, where λ_j^k is the solution of the dual problem.

The main steps of the Cutting plane algorithm are as follows.

Algorithm 2. *(Cutting plane algorithm)*

Step 1. *let $\delta > 0$ a given stopping tolerance, $S \neq \varnothing$ be a compact convex set containing a minimum point of f, choose any starting point $x_1 \in S$, and set $k := 1$.*

Step 2. *If*

$$\|f(x_k) - \hat{f}^{k-1}(x_k)\| < \delta,$$

then stop.

Step 3. *Compute the search direction d_k at $x = x_k$ as a solution of problem (2.5).*

Step 4. *Take $t_k = 1$ constant step size, and set $x_{k+1} := x_k + d_k$, Set $k := k + 1$ and go to Step 2.*

2.3 Bundle methods

Basically, subgradient methods use only one subgradient at each iteration, without a memory of past information iterations. If, instead, the past information obtained so far is kept, it is possible to

define piecewise-affine approximation of the objective function. The bundle method uses piecewise affine approximations to find search directions. These methods are among most efficient in nonsmooth optimization. In what follows we give a review regarding the bundle algorithms.

2.3.1 Standard bundle method

The bundle method was first introduced by C. Lemaréchal in [52] and then modified by many authors [5, 7, 35, 36, 43, 48, 61, 65]. The basic idea of bundle methods is to approximate the subdifferential of the objective function by gathering subgradients from previous iterations into a bundle, which can be used to describe the local behavior of the objective function, as opposed to using only the current subgradients (like in the subgradient method). This subgradient information serves for the construction of a piecewise linear local approximation to the objective function. A search direction for this approximation is usually obtained as a solution of quadratic programming subproblem [61]. The global convergence of bundle methods with a bounded number of stored subgradients can be guaranteed [48].

The basic idea of bundle type algorithms consists of replacing the subdifferential by some polytope approximation. Indeed, if the approximation is sufficiently good the algorithm will find a descent direction and a new point x_{k+1} will decrease the value of the objective function. In case the approximation is bad we stay at x_k and try to improve the approximation by adding further subgradients.

Consider the nonsmooth unconstrained minimization problem (2.1). Bundle method produces a sequence of points $x_k \subset \mathbb{R}^n$ that converges to a global minimum of a convex function f or a stationary point of the nonconvex function f. First, we define the Cutting plane model function $\hat{f}(x)$ that approximates the objective function at the iteration point x_k by

$$\hat{f}_k(x) := \max_{j \in J_k} f(y_k) + \langle \xi_j, x - y_k \rangle - \alpha_j^k, \quad j \in J_k \subset \{1, ..., k\},$$

$$\alpha_j^k = f(x_k) - f(y_k) + \langle \xi_j, x - y_k \rangle \quad \forall j \in J_k.$$

Here $\xi_j \in \partial f(y_j)$ is the subgradient of the trial point $y_j \in \mathbb{R}^n$ from previous iteration, and J_k nonempty index set, α_j^k is called *linearization error* which measures how good the model of the original problem is. For a convex function f the linearization error α_j^k is nonnegative, but for nonconvex case it can be negative. Therefore, the linearization error for nonconvex functions was replaced by the subgradient locality measure β_j^k

$$\beta_j^k = max\{|\alpha_j^k|, \gamma(x_k - y_j)^\omega\},$$

here $\gamma > 0$ is called a *distance measure parameter*, $\omega \geq 1$ is called a *locality measure parameter*. The next iteration is obtained as

$$y_{k+1} := x_k + d_k,$$

where the search direction d_k is calculated as

$$d_k := arg \min_{d \in R^n} \left\{ \hat{f}_k(x_k + d) + \frac{1}{2}\langle d, D_k d\rangle \right\}, \tag{2.6}$$

where D_k is symmetric, positive definite $n \times n$ matrix that accumulates information about the curvature of the objective function f in a ball around the point x_k [62]. The role of the stabilization term $\frac{1}{2}\langle d, D_k d\rangle$ is to keep the approximation $\hat{f}_k$ of the objective function f local enough.

A serious step is taken

$$\begin{aligned} x_{k+1} &= y_{k+1} \\ \text{and} \qquad J_{k+1} &= J_k \cup \{k+1\} \quad \text{for all } j \in J_k, \end{aligned} \tag{2.7}$$

if $f(y_{k+1})$ is significantly less than $f(x_k)$, otherwise, a null step is taken

$$\begin{aligned} x_{k+1} &= x_k \\ \text{and set} \qquad J_{k+1} &= J_k \cup \{k+1\} \quad \text{for all } j \in J_k. \end{aligned} \tag{2.8}$$

By updating the index set J_k we improve the Cutting plane model $\hat{f}_{k+1}$ in both steps. Notice that the quadratic direction finding minimization problem (2.6) can be rewritten as follows

$$\begin{aligned} \min \quad & \nu + \frac{1}{2}\langle d, D_k d\rangle \\ \text{st.} \quad & -\beta_j^k + \langle d, \xi_j\rangle \leq \nu \quad \text{for all } j \in J_k, \end{aligned} \tag{2.9}$$

The solution d_k of problem (2.9) can be found by solving quadratic dual problem as follows

$$d_k = -\sum_{j\in J_k} \lambda_j^k D_{k-1}\xi_j$$

where $\sum_{j\in J_k} \lambda_j = 1$ and $\lambda_j \geq 0$, for all $j \in J_k$.

Bundle methods have been developed for solving problem (2.1), with similar basic structure. Their main point of difference is based on the choice of the Cutting plane approximation $\hat{f}_k$, the linearization error β_j^k and stabilizing matrix D_k [62].

Algorithm 3. *(Standard Bundle algorithm)*

Step 1. *(*Initialization*) Choose the stopping tolerance δ , line search parameter ε, distance measure parameter $\gamma > 0$. Choose any starting point $x_1 \in \mathbb{R}^n$, positive definite matrix D_1, set $y_1 = x_1$, $J_1 = \{1\}$. Set the iteration counter $k := 1$.*

Step 2. *(*Direction finding*) Solve the quadratic dual problem of problem (2.9) to find the direction d_k and the minimal value ϖ_k of problem (2.9).*

Step 3. *(*Stopping criterion*) If*

$$\varpi_k < \delta,$$

then stop with x_k the final solution.

Step 4. *(*Line search*) Determine the step size t_k by a line search algorithm. If*

$$f(x_k + t_k d_k) \leq f(x_k) - \varepsilon t_k \varpi_k$$

do Serious step

$$x_{k+1} := x_k + t_k d_k$$

otherwise do Null step

$$x_{k+1} := x_k$$

Step 5. *(*updating*) Determine the stabilization matrix D_{k+1} using an updating formula. Determine the index set J_{k+1} and go to Step 2.*

The convergence of bundle methods can be proved under some assumptions [48]. The computational results show that the bundle methods perform significantly better than the subgradient method for minimizing nonsmooth convex functions. However, a solution to quadratic Problem (2.9) is time consuming for large scale nonsmooth optimization problems. Several types of bundle methods have been developed to avoid such costly computation. Variable metric bundle type methods are the most powerful method among them, where the time cosuming quadratic Problem (2.9) does not need to be solved.

Bundle methods need relatively large bundles to solve the problems efficiently. In other words, the size of the bundle has to be approximately the same as the number of variables [48]

2.3.2 Variable metric bundle type methods

The variable metric method for convex unconstrained minimization was proposed in [56]. Since this method is relatively robust and efficient even in the nonsmooth case, it has been extended for convex nonsmooth unconstrained minimization in [94]. Variable metric bundle methods are hybrid methods of the standard variable metric method and standard bundle method. The idea of the method is to use only three subgradients for direction determination; one at the current point x_k, the other at a trial point y_{k+1}, and the last aggregated one containing information from past iterations. This means that the dimension of quadratic programming subproblem is only three. Therefore, the size of the bundle does not need to grow with the dimension of the problem. To ensure the global convergence, the matrices are chosen to be uniformly positive definite. However, these methods use dense ap-

proximations of the Hessian matrix to calculate the search direction which become inefficient when the dimension of the problem increases. Practical optimization problems often involve nonsmooth functions of hundreds of variables; which can be a dilemma.

2.3.3 Limited memory bundle methods

For large scale nonsmooth optimization problems, it can be time consuming to use the variable metric bundle methods. Thus, a hybrid method of a variable metric bundle methods[57, 94] and a limited memory variable metric methods [20, 74] has been proposed [40, 41, 42]. In this method the approximation of the Hessian matrix updates uses the information of the last few iterations to define a variable metric approximation. In practice, this means that the approximation of the Hessian matrix is not as accurate as that of the original variable metric bundle methods but both the storage space required and the number of operations used are significantly smaller. The Limited memory bundle method efficiency has been proved by numerical results. At each iteration of this method only three subgradients with a certain type of the quasi Newton updates are used to find search directions. The method exploits simple aggregation of subgradients, and calculates the search direction using a limited memory approach. As a result, the time consuming quadratic subproblem appearing in standard bundle methods need not be solved and the number of stored subgradients is independent of the dimension of the problem. $SR1$ update and $BFGS$ update of the matrix in descent direction finding are used individually for a null step and a serious step to guarantee the convergence of the algorithm.

2.3.4 Quasisecant method

In bundle methods, the subdifferential of the objective function is approximated using those subgradients that provide good underestimators from previous iterations. For convex functions, it is easy to find relevant subgradients from the previous iterations. However for nonconvex functions subgradient information produces only local approximation to the objective function and should be discounted when no longer relevant. Unlike the convex case, in the nonconvex case, in general, not all subgradients provide a tight local approximation to a function. Therefore, subgradients which provide such

an approximation of a function in some neighborhood of a point are of big interest. To address this problem (at least partially) the notion of the quasisecant was introduced in [7]. quasisecants will be explained in greater detail in section (4.1). It was demonstrated that quasisecants can be efficiently computed for some nonsmooth functions. These include convex functions, functions represented as a maximum of a finite number of smooth functions, functions represented as a difference of two convex (DC) functions. Some interesting functions such as functions represented as a maxima of minima of a finite number of smooth convex functions (which includes continuous nonconvex piecewise linear functions), functions represented as a sum of minima of a finite number of smooth convex functions can be easily represented as a difference of two (nonsmooth) convex functions. Quasisecants are approximate subgradients. Quasisecant provide overestimation for the objective function in some neighborhood with a point. The computational results of the quasisecant method for solving well known nonsmooth optimization show that the bundle method performs significantly better than the quasisecant method for convex functions whereas the quasisecant method outperforms the bundle method for nonconvex nonregular problems.

2.4 Methods based on smoothing techniques

Consider the minimization problem

$$\text{minimize } f(x) \text{ subject to } \ x \in \mathbb{R}^n \tag{2.10}$$

where

$$f(x) = \max_{i \in I} f_i(x), \ \ I = \{1, \ldots, m\}, \tag{2.11}$$

and the functions $f_i, \ i \in I$ are continuously differentiable.

These problems are nonsmooth problems because of the presence of the "max" operator and appears in many application , such as vehicle routing (see, for example, [2, 3]), location (see, for example, [16, 32]), resource-allocation (see, for example, [60, 78]), structural optimization (see, for example,

[10, 24]) and many more.

There exist many algorithms for solving Problem (2.10) (see, for example, [28, 33]). Moreover, conventional nonsmooth optimization algorithms such as the bundle methods and its variations can also be applied to solve this problem [5, 7, 44, 48, 61]. Problem (2.10) can also be reformulated as a nonlinear programming problem and therefore efficient nonlinear programming techniques can be applied to solve it.

One of the methods to solve nonsmooth optimization problems is applying a smoothing technique to replace the original nondifferentiable problem by an approximate smooth one. Smoothing techniques bring these problems close to continuously differentiable equations or continuously differentiable programming problems which can be solved by the conventional smooth optimization methods. It has been shown that the smoothing approximation techniques are efficient methods for solving certain specially structured nonsmooth problems. Many of the algorithms are based on reformulating the problem. Smoothing methods have been developed for solving many important optimization problems including min-max problems. The main feature of smoothing methods is to approximate the nonsmooth nondifferentiable problems by a sequence of parameterized smooth continuously differentiable problems, and to trace the smooth path which leads to solutions. The accuracy of the approximation is controlled by some parameter, which is called a smoothing or precision parameter. In the last decade many smooth approximation functions have been developed. A smooth approximation of the ”max” function simplifies the problem and facilitates a platform for us to study related problems.

Recently, different smoothing techniques have been developed to replace the objective function f in Problem (2.10) by a smooth function. Such an approach allows one to solve the finite minimax problem using smooth optimization solvers. The smooth function approximates the objective function f and this approximation is controlled by the precision parameter(s). We can divide smoothing techniques into two main classes. Smoothing techniques from the first class try to smooth the objective function only in some neighborhood of the so-called kink points (points where the function f is not differentiable) whereas smoothing techniques from the second class smooth the objective function f

globally.

The paper [103] introduces different functions to smooth the finite maximum function at the kink points. In the paper [34], the authors reformulate the finite minimax problem replacing the maximum function by the sum of more simple maximum functions and develop smooth approximations of the reformulated function. This smooth function approximates the reformulated function only at the points where the function is not differentiable.

Smoothing techniques from the second class include the exponential and hyperbolic smoothing functions. The general approach for smoothing was introduced in [15] where different smoothing functions, including exponential smoothing function, were considered. The paper [93] considers the logarithmic barrier function of the epigraph of the maximum function to smooth it. In the paper [102], the author introduces a smoothing method for minimax problems based on the exponential smoothing function. He proves its convergence and provides results of preliminary numerical experiments. The paper [81] introduces another version of smoothing technique using exponential smoothing. A feedback precision-adjustment rule is used to update the precision parameter in the exponential smoothing function which allows to avoid the ill-conditioning associated with large precision parameters.

In the paper [100], a truncated exponential smoothing function is introduced which is later combined with a Newton-Armijo algorithm to solve a minimax problem. A generalization of the exponential smoothing algorithm for finite min-max-min problems was considered in [90]. A smoothing technique is applied twice, once to eliminate the inner min operator and once more to eliminate the max operator.

2.4.1 Exponential penalty smoothing method

Ben-Tal and Teboulle [15], introduced a smoothing technique for nondifferentiable optimization problems with maximum objective functions. The tool they used to generate such an approximate problem is through the use of the recession approximate problem by a smooth optimization problem which contains a smoothing parameter. This parameter controls the accuracy of the approximation. This smoothing approach has also been proposed and used to smooth min-max problems by Polak

[81]. This method is called exponential smoothing function.

Consider

$$\Psi(x) = \max_{i=1,...,m} f_i(x), \;\; i = 1, ..., m,$$

where the functions $f_i, \; i = 1, ..., m$ are continuously differentiable.

Let $\mu > 0$ be a smoothing parameter. Define the following function [81]

$$\Psi_\mu(x) = \frac{1}{\mu} \log \sum_{i=1}^{m} exp^{\mu f_i(x)}. \tag{2.12}$$

It follows from (2.12) that

1. $\Psi(x) \leq \Psi_\mu(x) \leq \Psi(x) + \frac{1}{\mu} \log m.$
2. $\Psi_\mu(x)$ is decreasing with respect to μ.
3. $\Psi_\mu(x) \to \Psi(x)$as $\mu \to +\infty$.
4. $\Psi_\mu(x)$ is twice continuously differentiable for all $\mu > 0$.

2.4.2 Hyperbolic smoothing functions

The paper [97] was the first instance where the hyperbolic smoothing function was considered. In the paper [98], the problem of the optimal covering of plane domains by circles was solved by applying a hyperbolic smoothing technique and in the paper [99] this technique was applied to solve the cluster analysis problem using its nonsmooth optimization formulation.

The hyperbolic function was introduced for the following function:

$$f(x) = \max\{0, x\}. \tag{2.13}$$

The hyperbolic smoothing of this function is as follows:

$$\phi_\tau(x) = \frac{x + \sqrt{x^2 + \tau^2}}{2}, \tag{2.14}$$

where $\tau > 0$ is a precision parameter.

Proposition 1. *The function $\phi_\tau(x)$ has the following properties:*

1. *$\phi_\tau(\cdot)$ is an increasing convex C^∞ function;*

2. $f(x) < \phi_\tau(x) \leq f(x) + \dfrac{\tau}{2}$, $\forall x \in \mathbb{R}$.

Proof. The proof is straightforward. □

The hyperbolic function for smoothing the function (2.13) is illustrated in Figure (2.1) where blue curve shows the smoothing function.

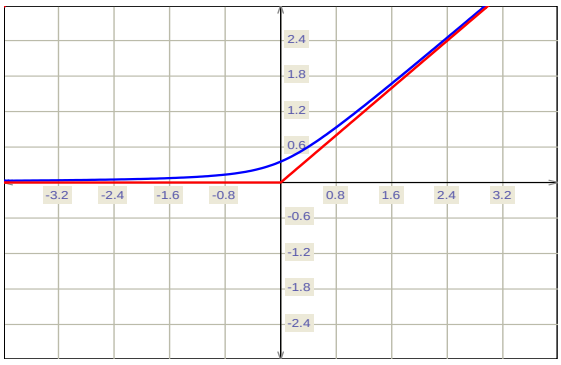


Figure 2.1: Hyperbolic smoothing of the function (2.11).

2.5 Optimization methods in water management

Application of optimization methods in water management is an emerging field in operations research. There are several problems in water management where optimization techniques have been successfully applied, such as design and rehabilitation of water distribution systems, operation of water distribution systems, operations of a reservoir system and groundwater management (see, for example, [49, 70] and also website of International Federation of Operations Research Societies (IFORS)).

Water distribution networks represent one of the largest infrastructure assets of industrial society [91], with energy costs for pumping being a significant part of the operational cost of water distribution

networks worldwide [54, 92]. As such, the optimal pump scheduling can save a significant amount of operational cost of the water distribution system. However, finding the best schedule for a number of pumps, supplying water to satisfy a variable demand and reducing the energy consumption, is a very difficult task. The difficulty is mainly due to the discrete nature of the variables and the size of the solution space [91]. This problem can be formulated as an optimization problem which may have a large number of continuous and discrete variables. It also contains many physical and operational constraints depending on the system.

The problem of efficient scheduling of pumps has been subject to research over the last several decades. In the early stage, optimization models for scheduling pump operations explicitly included hydraulic constraints along with other physical and operational constraints [26, 76, 77]. Because hydraulic constraints are very difficult to fully describe for a given water distribution system, it is not easy to design efficient algorithms based on such an approach.

A review of some early optimization approaches to pump scheduling can be found in [75]. An iterative dynamic programming method was developed in [104] to find an optimal schedule of pump operations. This method uses the forecasted demands for 24 hours, the initial and final conditions in the reservoirs as well as the hydraulic properties of the whole system. A linear programming approach was proposed in [79]. The paper [86] describes a multi-objective optimization formulation using both the energy cost and the pump switching criterion as objective functions. In this paper, the genetic algorithm was modified to solve the multi-objective optimization problem. Various versions of the genetic algorithm were developed in [19, 27] for solving pump optimization problems. In all above mentioned papers, the authors try to explicitly include hydraulic constraints into the optimization model.

In the paper [18], the authors propose an approach which is based on the maximization of the use of low-cost power (e.g. overnight pumping). They formulate the operational optimization of water distribution networks as a mixed integer nonlinear programming problem. The paper [91] proposes an approach to determine a penalty term in the objective function of the pumping cost minimization problem. This term depends on the degree of failure and on the set pressure criteria. A version of the

genetic algorithm was developed based on this approach.

In the paper [54], a schedule of pumps is explicitly defined based on time controlled triggers, where the maximum number of pump switches is specified beforehand. A pump schedule is divided into a series of integers with each integer representing the number of hours for which a pump is active/inactive. An algorithm based on an ant colony optimization was developed to solve the optimal pump scheduling problem. An approach which decomposes a water supply system into several subsystems and a planning period into operational periods was proposed in [71]. The pump discharges are discretized and arranged by heuristic methods in order to reduce the number of times pumps are switched on. A dynamic programming method is consequently applied to solve the optimization problem.

An adaptive search algorithm is proposed in [80]. This algorithm selects which pumps to switch on or off, using a combination of influence coefficients and pipe network pressure readings. When the pressure increases or drops beyond the allowable values, the pump which has the greatest influence and delivers water at least cost is selected to correct the pressure by either turning it on or off as required. The algorithm iterates between the optimization and the simulation models until the optimal solution is found.

The paper [85] develops an approach for determining the optimal scheduling of pumps in the water distribution system with water quality considerations. In this approach, bound constraints on the state variables are incorporated into the objective function using the augmented lagrangian penalty method. The solution of the optimization problem is obtained by interfacing a hydraulic and water quality simulation code, EPANet, with a nonlinear optimization code, GRG2. In the paper [92], the authors propose algorithms based on the combination of the genetic and direct search methods such as the Hooke and Jeeves, and Fibonacci methods for solving the pumping cost minimization problem. It is demonstrated that the hybrid methods are superior to the pure genetic algorithm in finding a good solution quickly when applied to both a test problem and a large existing water distribution system.

The development of hydraulic simulation packages led to the design of more efficient algorithms for solving pumping cost minimization problems. The use of such packages allows to avoid difficulty

with explicit inclusion of hydraulic constraints in the optimization models. This approach, which integrates a hydraulic simulation model with an optimization model, is now widely used to optimize pump operations. It should be noted that algorithms proposed in papers [18, 54, 71, 80, 85, 91, 92] iterate between optimization and simulation models to find optimal solutions to the pumping cost minimization problem.

The discrete (binary) nature of some variables and the size of the solution space are among the main difficulties of optimizing water distribution systems operation [91]. More specifically, the pumping cost minimization problem is a mixed integer nonlinear programming problem. Conventional optimization methods are not directly applicable for solving such problems, because these methods are suitable mostly for optimization problems which have only continuous variables. Population based methods such as evolutionary algorithms and various meta-heuristics are well suited to deal with both discrete and continuous variables. These algorithms have been widely used to solve pumping cost minimization problems. However, population based algorithms have the following drawbacks. Firstly, they require a large number of the objective and constraint function evaluations, which is not acceptable when these evaluations are expensive. Secondly, they are inefficient for solving large scale problems. Thirdly, these algorithms sometimes cannot locate a solution with high accuracy and as a result, they may produce only suboptimal solutions.

Conventional deterministic methods of optimization are more accurate than the population based algorithms. Algorithms for solving pumping cost minimization problems contain both optimization and simulation components, and direct search methods are more suitable for their solution than gradient based or Newton-like methods. The significant advantage of direct search methods is that they do not require any gradient or Hessian information, and can be applied for solving optimization problems with noisy input data. This makes direct search methods attractive for solving pumping cost minimization problem. The papers [85, 92] propose algorithms based on the combination of the population based methods with the various direct search methods. Results presented in these papers demonstrate that such algorithms are able to obtain more accurate solutions than the population based methods. These results also illustrate that population based methods are efficient to generate feasible solutions

to the pumping cost minimization problem, whereas direct search methods can be applied starting from those identified feasible solutions to get more accurate ones.

Different approaches have been proposed to reduce the number of discrete variables in the pumping cost minimization problems. One such approach was proposed in [92] where level of water in tanks is considered as a decision variable.

Chapter 3

Hyperbolic smoothing function method for minimax problems

In this chapter we study hyperbolic smoothing technique in more detail. In order to apply the hyperbolic smoothing to the finite maximum functions we represent them as a sum of maximum of two functions by adding a new variable. We study the relationship between the set of stationary points of the latter function and that of the original maximum function. The new function is approximated using hyperbolic smoothing functions and differential properties of the approximating function are studied. It is demonstrated that smooth optimization solvers can be applied to minimize the approximating function. We present results of numerical experiments using two solvers from GAMS: CONOPT and SNOPT. We also compare these results with those obtained using exponential smoothing and also nonlinear programming reformulations of minimax problems.

The structure of this chapter is as follows. We reformulate the finite maximum function in Section 3.1. Section 3.2 describes the hyperbolic smoothing function for the general maximum functions. The minimization algorithm is described in Section 3.3. Results of numerical experiments are presented in Section 3.4. Section 3.5 concludes the chapter.

3.1 Reformulation of minimax problem

In this section we will reformulate the minimax problem (2.10) to make the application of the hyperbolic smoothing to its objective function possible.

Consider the following maximum function:

$$f(x) = \max_{i \in I} f_i(x), \;\; I = \{1, \ldots, m\}. \tag{3.1}$$

At a point $x \in \mathbb{R}^n$ consider the set:

$$R(x) = \{i \in I : \; f_i(x) = f(x)\}.$$

Using an additional variable $t \in \mathbb{R}$ we introduce the following function:

$$F(x,t) = t + \sum_{i \in I} \max\{0, f_i(x) - t\}. \tag{3.2}$$

For a given (x,t) the index set I can be represented as follows:

$$I = I_1 \cup I_2 \cup I_3,$$

where

$$I_1 \equiv I_1(x,t) = \{i \in I : f_i(x) < t\},$$

$$I_2 \equiv I_2(x,t) = \{i \in I : f_i(x) = t\},$$

$$I_3 \equiv I_3(x,t) = \{i \in I : f_i(x) > t\}.$$

Denote by $\Psi_i(x,t) = \max\{0, f_i(x) - t\}, \; i \in I$. Then for the subdifferential of the function Ψ_i we have

$$\partial \Psi_i(x,t) = \begin{cases} \{0_{n+1}\}, & i \in I_1, \\ \text{co}\,\{0_{n+1}, (\nabla f_i(x), -1)\}, & i \in I_2, \\ \{(\nabla f_i(x), -1)\}, & i \in I_3. \end{cases}$$

Since $f_i, \; i \in I$ are regular functions then Ψ_i are regular as well and therefore we can write the

expression for the subdifferential of F at the point (x,t) as follows:

$$\partial F(x,t) = \{(0_n,1)\} + \sum_{i\in I_1} 0_{n+1} + \sum_{i\in I_2} \operatorname{co}\{0_{n+1},(\nabla f_i(x),-1)\} + \left\{\sum_{i\in I_3}(\nabla f_i(x),-1)\right\}. \quad (3.3)$$

Proposition 2. *Suppose that functions f and F are defined by (2.11) and (3.2), respectively. Then*

$$f(x) = \min_{t\in\mathbb{R}} F(x,t).$$

Proof. For any fixed $x \in \mathbb{R}^n$ define the following function:

$$\varphi_x(t) = t + \sum_{i\in I} \max\{0, f_i(x) - t\}.$$

Observe that the function φ_x is convex piecewise linear and

$$\varphi_x(f(x)) = f(x) = F(x, f(x)).$$

Then the function φ_x is subdifferentiable at any $t \in \mathbb{R}$ and

$$\partial\varphi_x(t) = [1 - |I_2| - |I_3|, 1 - |I_3|]. \quad (3.4)$$

Here $|\cdot|$ stands for the cardinality of a set. For $t = f(x)$ we have that $I_2 = R(x) \neq \emptyset$ and therefore $|I_2| \geq 1$. Moreover for this t one has $I_3 = \emptyset$ and $|I_3| = 0$. Then it follows from (3.4) that

$$0 \in \partial\varphi_x(f(x))$$

and $t = f(x)$ is a global minimizer of φ_x. Furthermore, for any fixed $x \in \mathbb{R}^n$

$$F(x,t) = \varphi_x(t) \geq \varphi_x(f(x)) = f(x) = F(x, f(x)) \quad \forall t \in \mathbb{R}.$$

This completes the proof. □

Proposition 3. *1) Assume that a point $x^* \in \mathbb{R}^n$ is a stationary point of f. Then (x^*, t^*) is a stationary point of the function F where $t^* = f(x^*)$.*

2) Assume that a point (x^*, t^*) is a stationary point of the function F. Then $x^* \in \mathbb{R}^n$ is a stationary point of f.

Proof. 1) First we assume that x^* is a stationary point of the function f and will prove that (x^*, t^*) is a stationary point of the function F where $t^* = f(x^*)$. Since

$$t^* = f(x^*) = \max_{i \in I} f_i(x^*),$$

$t^* \geq f_i(x^*)$ for all $i \in I$ and thus $I_3 = \emptyset$. Moreover, $I_2 \neq \emptyset$ since at least one of the functions $f_i,\ i \in I$ is active at x^*. Then the subdifferential of the function F at the point (x^*, t^*) is as follows:

$$\partial F(x^*, t^*) = \{(0_n, 1)\} + \sum_{i \in I_2} \text{co}\,\{0_{n+1}, (\nabla f_i(x^*), -1)\}. \tag{3.5}$$

It is easy to see that $(\nabla f_i(x^*), 0) \in \partial F(x^*, t^*)$. It is also obvious that $R(x^*) = I_2$ at the point (x^*, t^*). Since x^* is a stationary point of the function f we get that $0_n \in \text{co}\,\{\nabla f_i(x^*) : i \in I_2\}$. Then there exists $\lambda_i, i \in I_2$ such that

$$0_n = \sum_{i \in I_2} \lambda_i \nabla f_i(x^*), \ \ \lambda_i \geq 0, \ \sum_{i \in I_2} \lambda_i = 1. \tag{3.6}$$

The subdifferential $\partial F(x^*, t^*)$ is a polytope and it follows from (3.5) that points $(\nabla f_i(x^*), 0), i \in I_2$ are among extreme points of this polytope. Then (3.6) implies that $0_{n+1} \in \partial F(x^*, t^*)$, that is the point (x^*, t^*) is stationary for the function F.

2) Now assume that (x^*, t^*) is a stationary point of the function F. We will prove that x^* is a stationary point of the function f. There are three cases:

- Case 1: $t^* > f(x^*)$;

- Case 2: $t^* = f(x^*)$;

- Case 3: $t^* < f(x^*)$.

We will consider each of these cases separately.

Case 1. If $t^* > f(x^*) = \max\limits_{i \in I} f_i(x^*)$, then $t^* > f_i(x^*)$ for any $i \in I$ and therefore index sets I_2 and I_3 are empty. Hence it follows from (3.3) that $\partial F(x^*, t^*) = \{(0_n, 1)\}$ and therefore $0_{n+1} \notin \partial F(x^*, t^*)$ which contradicts the fact that (x^*, t^*) is a stationary point of F. This means that Case 1 cannot happen.

Case 2. If $t^* = f(x^*)$ then $t^* \geq f_i(x^*)$ for all $i \in I$. Therefore $I_2 \neq \emptyset$ and $I_3 = \emptyset$. Hence it follows from (3.3) that

$$\partial F(x^*, t^*) = \{(0_n, 1)\} + \sum_{i \in I_2} \text{co}\,\{0_{n+1}, (\nabla f_i(x^*), -1)\}.$$

The subdifferential $\partial F(x^*, t^*)$ is a polytope and any extreme point V of this subdifferential can be expressed in one of the following forms:

- $V = (0_n, 1)$;

- There exists a subset $\bar{I}_2 \subseteq I_2$ such that , $|\bar{I}_2| \geq 1$ and

$$V = \left(\sum_{i \in \bar{I}_2} \nabla f_i(x^*), -|\bar{I}_2| + 1 \right). \tag{3.7}$$

Let $\{w_0, w_1, \ldots, w_K\}$ be a set of extreme points of the subdifferential $\partial F(x^*, t^*)$. Here $w_0 = (0_n, 1)$ and $w_1, \ldots, w_K$ are in the form of (3.7). The number $K > 0$ denotes the total number of extreme points of the form (3.7). Since (x^*, t^*) is a stationary point $0_{n+1} \in \partial F(x^*, t^*)$, there exists $\lambda_k,\ k = 0, \ldots, K$ such that

$$\lambda_0 w_0 + \sum_{k=1}^{K} \lambda_k w_k = 0 \tag{3.8}$$

and

$$\lambda_0 + \sum_{k=1}^{K} \lambda_k = 1, \ \ \lambda_0 \geq 0, \ \ \lambda_k \geq 0,\ k = 1, \ldots, K. \tag{3.9}$$

It is clear that $\lambda_0 \neq 1$. Otherwise all $\lambda_k = 0,\ k = 1, \ldots, K$ and we will have $\lambda_0 w_0 = 0$ which is not possible. This means that $\lambda_k > 0$ for at least one $k = 1, \ldots, K$.

For each extreme point $w_k,\ k = 1, \ldots, K$ there exists $\bar{I}_2 \subseteq I_2$ such that $\bar{I}_2 \neq \emptyset$ and the point w_k can be represented in the form (3.7). Then the point w_k can be expressed as follows: $w_k = (u_k, v_k)$ where

$$u_k = \sum_{i \in \bar{I}_2} \nabla f_i(x^*)$$

and

$$v_k = -|\bar{I}_2| + 1.$$

Then it follows from (3.8) and (3.9) (notice that $w_0 = (0_n, 1)$) that

$$\sum_{k=1}^{K} \lambda_k u_k = 0, \quad \sum_{k=1}^{K} \lambda_k \leq 1, \quad \lambda_k \geq 0.$$

Notice also that $\sum\limits_{k=1}^{K} \lambda_k > 0$. Let $\bar{\lambda} = \sum\limits_{k=1}^{K} \lambda_k$ and we define new coefficients

$$\bar{\lambda}_k = \frac{\lambda_k}{\bar{\lambda}} \geq 0,\ k = 1, \ldots, K.$$

Then

$$\sum_{k=1}^{K} \bar{\lambda}_k u_k = 0, \quad \sum_{k=1}^{K} \bar{\lambda}_k = 1. \tag{3.10}$$

We can rewrite each u_k as

$$u_k = \sum_{i \in I_2} m_{ki} \nabla f_i(x^*),$$

where $m_{ki} = 0$ or $m_{ki} = 1,\ k = 1, \ldots, K,\ i \in I_2$. Moreover,

$$\sum_{i \in I_2} m_{ki} \geq 1 \ \ \forall\, k = 1, \ldots, K.$$

Then it follows from (3.10) that

$$\sum_{k=1}^{K} \bar{\lambda}_k \sum_{i \in I_2} m_{ki} \nabla f_i(x^*) = 0.$$

Reordering it we get

$$\sum_{i \in I_2} \left(\sum_{k=1}^{K} \bar{\lambda}_k m_{ki} \right) \nabla f_i(x^*) = 0. \tag{3.11}$$

Denote by

$$\alpha_i = \sum_{k=1}^{K} \bar{\lambda}_k m_{ki} \geq 0, \ i \in I_2.$$

It is easy to see that

$$\bar{\alpha} = \sum_{i \in I_2} \alpha_i \geq 1.$$

Indeed,

$$\bar{\alpha} = \sum_{i \in I_2} \alpha_i = \sum_{i \in I_2} \sum_{k=1}^{K} \bar{\lambda}_k m_{ki} = \sum_{k=1}^{K} \bar{\lambda}_k \sum_{i \in I_2} m_{ki} \geq \sum_{k=1}^{K} \bar{\lambda}_k = 1.$$

Now (3.11) implies that

$$\sum_{i \in I_2} \alpha_i \nabla f_i(x^*) = 0.$$

Let $\bar{\alpha_i} = \dfrac{\alpha_i}{\bar{\alpha}}$. Then

$$\sum_{i \in I_2} \bar{\alpha_i} \nabla f_i(x^*) = 0, \quad \sum_{i \in I_2} \bar{\alpha_i} = 1, \quad \bar{\alpha_i} \geq 0. \tag{3.12}$$

Since $I_2 = R(x^*)$ it follows from (3.12) that $0_n \in \partial f(x^*) = \operatorname{co} \{\nabla f_i(x^*) : i \in I_2\}$ and therefore x^* is a stationary point of f.

Case 3. Now let us assume that $t^* < f(x^*)$. Then the set I_3 is never empty and there are four possible combinations:

1) $|I_3| = 1, \ I_2 = \emptyset$. Then using (3.3) we get

$$\partial F(x^*, t^*) = \{(\nabla f_j(x^*), 0)\}, \ j \in I_3.$$

Since $0_{n+1} \in \partial F(x^*, t^*)$ it follows that $\nabla f_j(x^*) = 0_n$ and thus x^* is a stationary point of f.

2) $|I_3| = 1$, $I_2 \neq \emptyset$. In this case we have only one active function f_j at x^*,where $j \in I_3$ and $R(x^*) = \{j\}$. Then (3.3) implies that the subdifferential of the function F at (x^*, t^*) is as follows:

$$\partial F(x^*, t^*) = \{(0_n, 1)\} + \sum_{i \in I_2} \text{co}\,\{0_{n+1}, (\nabla f_i(x^*), -1)\} + \{(\nabla f_j(x^*), -1)\}$$

$$= \{(\nabla f_j(x^*), 0)\} + \sum_{i \in I_2} \text{co}\,\{0_{n+1}, (\nabla f_i(x^*), -1)\}.$$

It is clear that this subdifferential is a polytope and any one of its extreme points U can be expressed in one of the following forms:

- $U = (\nabla f_j(x^*), 0)$;
- There exists a subset $\bar{I}_2 \subseteq I_2$ such that , $|\bar{I}_2| \geq 1$ and

$$U = \left(\nabla f_j(x^*) + \sum_{i \in \bar{I}_2} \nabla f_i(x^*), -|\bar{I}_2| \right), \ j \in I_3. \tag{3.13}$$

Let $\{z_0, z_1, \ldots, z_L\}$ be a set of extreme points of the subdifferential $\partial F(x^*, t^*)$. Here $z_0 = (\nabla f_j(x^*), 0)$, $j \in I_3$ and $z_1, \ldots, z_L$ are in the form of (3.13). The number $L > 0$ is the total number of extreme points of the form (3.13). Let $\lambda_i,\ i = 0, \ldots, L$ be coefficients of the points $z_0, \ldots, z_L$ in their convex combination. It is easy to see that no convex combination of these points will give us a 0_{n+1} element unless all $\lambda_i = 0\ i = 1, \ldots, L$ and $\lambda_0 = 1$. Since $0_{n+1} \in \partial F(x^*, t^*)$ we have that $\lambda_0(\nabla f_j(x^*), 0) = 0_{n+1}$ from which follows that $\nabla f_j(x^*) = 0_n$ and $0_n \in \partial f(x^*)$, that is x^* is a stationary point of the function f.

3) $|I_3| > 1$, $I_2 = \emptyset$. Then

$$\partial F(x^*, t^*) = \left\{ \left(\sum_{i \in I_3} \nabla f_i(x^*), -|I_3| + 1 \right) \right\}$$

and therefore $0_{n+1} \notin \partial F(x^*, t^*)$, which contradicts the fact that (x^*, t^*) is a stationary point of F. This means that this case cannot happen.

4) $|I_3| > 1$, $I_2 \neq \emptyset$. In this case it follows from (3.3) that

$$\partial F(x^*, t^*) = \left\{ \left(\sum_{i \in I_3} \nabla f_i(x^*), -|I_3| + 1 \right) \right\} + \sum_{i \in I_2} \text{co} \{0_{n+1}, (\nabla f_i(x^*), -1)\}.$$

Since $-|I_3| + 1 \leq -1$ from the construction of the polytope $\partial F(x^*, t^*)$ it is not difficult to see that for any extreme point the last component will be less than or equal to -1 and therefore $0_{n+1} \notin \partial F(x^*, t^*)$. This contradicts the fact that (x^*, t^*) is a stationary point of F which means that this case cannot happen. □

Remark 1. One can see from the proof of Proposition 3 that if $(x^*, t^*) \in \mathbb{R}^{n+1}$ is a stationary point of the function F then only the following cases are possible:

1) $t^* = f(x^*)$;

2) $t^* < f(x^*)$.

In the second case the set I_3 is a singleton which means that the set $R(x^*)$ is also a singleton. Therefore, the function f is differentiable at the point x^*. In most minimax problems stationary points are the so-called kink points where function f is not differentiable. This means that for most of minimax problems $t^* = f(x^*)$. Moreover, results presented above demonstrate that any stationary point (x^*, t^*) of the function F can represented as $(x^*, f(x^*))$.

Proposition 4. *1) Assume that a point $x^* \in \mathbb{R}^n$ is a local minimizer of f. Then (x^*, t^*) is a local minimizer of the function F, where $t^* = f(x^*)$.*

2) Assume that a point (x^*, t^*) is a local minimizer of the function F. Then $x^* \in \mathbb{R}^n$ is a local minimizer of f.

Proof. 1) If x^* is a local minimizer of f then there exists $\varepsilon > 0$ such that

$$f(x) \geq f(x^*) \ \ \forall x \in B_\varepsilon(x^*).$$

We will prove that $F(x,t) \geq F(x^*,t^*)$ for any $x \in B_\varepsilon(x^*)$ and $t \in \mathbb{R}$. Note that $F(x^*,t^*) = f(x^*)$. Take any $x \in B_\varepsilon(x^*),\ t \in \mathbb{R}$ and consider the following index set $J(x)$:

$$J(x) = \{i \in I :\ f_i(x) - t \geq 0\}.$$

There are two cases:

1. $J(x) = \emptyset$;

2. $J(x) \neq \emptyset$.

In the first case $t > f(x)$, which means that $F(x,t) = t > f(x) \geq f(x^*) = F(x^*,t^*)$. In the second case there is at least one index $j \in J$ for which $f_j(x) = \max\limits_{i \in I} f_i(x) \geq f(x^*)$ and we can write

$$\begin{aligned} F(x,t) &= t + \sum_{i \in I} \max\{0, f_i(x) - t\} \\ &= t + \max\{0, f_j(x) - t\} + \sum_{i \in I, i \neq j} \max\{0, f_i(x) - t\} \\ &= f_j(x) + \sum_{i \in J(x), i \neq j} \max\{0, f_i(x) - t\} \\ &\geq f(x^*) + \sum_{i \in J(x), i \neq j} \max\{0, f_i(x) - t\}. \end{aligned}$$

It is clear that

$$\sum_{i \in J(x), i \neq j} \max\{0, f_i(x) - t\} \geq 0.$$

Then we get that $F(x,t) \geq f(x^*) = F(x^*,t^*)$. Since $x \in B_\varepsilon(x^*),\ t \in \mathbb{R}$ are arbitrary this completes the proof that (x^*,t^*) is a local minimizer of F.

2) Now assume that $(x^*,t^*),\ x^* \in \mathbb{R}^n,\ t^* \in \mathbb{R}$ is a local minimizer of the function F. This means that there exists $\varepsilon > 0$ such that

$$F(x,t) \geq F(x^*,t^*)\ \ \forall x \in B_\varepsilon(x^*),\ t \in \mathbb{R}, |t - t^*| < \varepsilon.$$

According to the proof of Proposition 3 (see Remark 1) only the following two cases are possible:

1) $t^* = f(x^*)$;

2) $t^* < f(x^*)$ and the index set I_3 is a singleton.

In the case 1) we have that there exists $\varepsilon > 0$ such that $F(x^*, t^*) \leq F(x, t)$ for all $x \in B_\varepsilon(x^*)$ and $t, |t - f(x^*)| < \varepsilon$. Since the function f is continuous for $\varepsilon > 0$ there exists $\delta > 0$ such that $|f(x) - f(x^*)| < \varepsilon$ for all $x \in B_\delta(x^*)$. If $\delta_1 = \min\{\varepsilon, \delta\}$ then

$$F(x^*, f(x^*)) \leq F(x, f(x)) \ \ \forall x \in B_{\delta_1}(x^*).$$

It follows from Proposition 2 that

$$f(x^*) = F(x^*, f(x^*)) \leq F(x, f(x)) \leq F(x, t)$$

for all $x \in B_{\delta_1}(x^*)$ and $t \in \mathbb{R}$. This means that

$$f(x^*) \leq \min_{t \in \mathbb{R}} F(x, t) = f(x)$$

for all $x \in B_{\delta_1}(x^*)$, that is the point x^* is a local minimizer of f.

In the case 2) we have that the index set $I_3 = I_3(x^*, t^*)$ is a singleton. Assume that $I_3(x^*, t^*) = \{i\}$ for some $i \in I$. Then it is obvious that

$$f(x^*) = f_i(x^*) = F(x^*, t^*).$$

Since (x^*, t^*) is a local minimizer of F there exists $\varepsilon > 0$ such that

$$F(x^*, t^*) \leq F(x, t) \ \ \forall x \in B_\varepsilon(x^*) \text{ and } \forall t : \ |t - t^*| < \varepsilon.$$

For sufficiently small $\delta > 0$ such that $\delta \in (0, \min\{\varepsilon, f_i(x^*) - t^*\})$ we have

$$I_1(x^*, \bar{t}) = I \setminus \{i\}, \;\; I_2(x^*, \bar{t}) = \emptyset, \; I_3(x^*, \bar{t}) = \{i\}, \; \bar{t} = t^* + \delta.$$

Then it follows from the continuity of the functions $f_i, \; i \in I$ that there exists $\varepsilon_1 \in (0, \varepsilon)$ such that

$$I_1(x, \bar{t}) = I \setminus \{i\}, \;\; I_2(x, \bar{t}) = \emptyset, \; I_3(x, \bar{t}) = \{i\} \;\; \forall x \in B_{\varepsilon_1}(x^*).$$

This means that

$$F(x, \bar{t}) = f_i(x) = f(x) \;\; \forall x \in B_{\varepsilon_1}(x^*).$$

and therefore

$$f(x) = F(x, \bar{t}) \geq F(x^*, t^*) = f(x^*) \;\; \forall x \in B_{\varepsilon_1}(x^*).$$

Thus, x^* is a local minimizer of the function f. □

Remark 2. The reformulation F of the objective function f in minimax problem (2.10) was considered in [34] where the authors also proved that the values of global minima of functions f and F are equal.

3.2 Hyperbolic smoothing of the maximum function

In this section we study hyperbolic smoothing functions for the general finite maximum functions using their reformulation (3.2).

Applying (2.14) we get the following hyperbolic smoothing of the function F:

$$\Phi_\tau(x, t) = t + \sum_{i \in I} \frac{f_i(x) - t + \sqrt{(f_i(x) - t)^2 + \tau^2}}{2}, \; \tau > 0. \tag{3.14}$$

Proposition 5. *For any $x \in \mathbb{R}^n$ and $t \in \mathbb{R}$*

$$0 < \Phi_\tau(x,t) - F(x,t) \leq \frac{m\tau}{2}.$$

Proof. The result follows from Proposition 1. □

The gradient of the function Φ_τ is as follows:

$$\nabla\Phi_\tau(x,t) = (G_{1\tau}(x,t), G_{2\tau}(x,t)) \tag{3.15}$$

where

$$G_{1\tau}(x,t) = \frac{1}{2}\sum_{i\in I}(1+\beta_{i\tau}(x,t))\,\nabla f_i(x), \tag{3.16}$$

$$G_{2\tau}(x,t) = 1 - \frac{1}{2}|I| - \frac{1}{2}\sum_{i\in I}\beta_{i\tau}(x,t). \tag{3.17}$$

$$\beta_{i\tau}(x,t) = \frac{f_i(x)-t}{\sqrt{(f_i(x)-t)^2+\tau^2}}. \tag{3.18}$$

Proposition 6. *Assume that*

$$v = \lim_{\tau\to 0}\nabla\Phi_\tau(x,t).$$

Then $v \in \partial F(x,t)$.

Proof: Rewriting the first component of $\nabla\Phi_\tau(x,t)$ we obtain

$$\begin{aligned} G_{1\tau}(x,t) = \frac{1}{2}\Big[\sum_{i\in I_1}(1+\beta_{i\tau}(x,t))\,\nabla f_i(x) + \sum_{i\in I_2}(1+\beta_{i\tau}(x,t))\,\nabla f_i(x) \\ + \sum_{i\in I_3}(1+\beta_{i\tau}(x,t))\,\nabla f_i(x)\Big]. \end{aligned} \tag{3.19}$$

It is clear that for any $x \in \mathbb{R}^n$ and $t \in \mathbb{R}$

$$\lim_{\tau \to 0} \beta_{i\tau}(x,t) = \begin{cases} -1, & i \in I_1, \\ 0, & i \in I_2, \\ 1, & i \in I_3. \end{cases}$$

Then taking limit as $\tau \to 0$ we get

$$\begin{aligned} \lim_{\tau \to 0} G_{1\tau}(x,t) &= \frac{1}{2}\Big[\sum_{i \in I_1} 0_n + \sum_{i \in I_2} \nabla f_i(x) + \sum_{i \in I_3} 2\nabla f_i(x)\Big] \\ &= \sum_{i \in I_1} 0_n + \sum_{i \in I_2} \frac{1}{2}\nabla f_i(x) + \sum_{i \in I_3} \nabla f_i(x). \end{aligned}$$

Performing similar calculations for the second component of $\nabla \Phi_\tau(x,t)$ we obtain the following

$$\begin{aligned} \lim_{\tau \to 0} G_{2\tau}(x,t) &= 1 - \frac{1}{2}|I| - \frac{1}{2}\left(-\sum_{i \in I_1} 1 + \sum_{i \in I_2} 0 + \sum_{i \in I_3} 1\right) \\ &= 1 - \frac{1}{2}|I_2| - |I_3|. \end{aligned}$$

Therefore

$$v = \lim_{\tau \to 0} \nabla \Phi_\tau(x,t) = \left(\sum_{i \in I_1} 0_n + \sum_{i \in I_2} \frac{1}{2}\nabla f_i(x) + \sum_{i \in I_3} \nabla f_i(x), 1 - \frac{1}{2}|I_2| - |I_3|\right). \tag{3.20}$$

Comparing (3.3) and (3.20) it is easy to see that $v \in \partial F(x,t)$. □

Proposition 7. *Assume that sequences* $\{x_k\},\ \{t_k\}$ *and* $\{\tau_k\}$ *are given such that* $x_k \in \mathbb{R}^n,\ t_k \in \mathbb{R},\ t_k \geq f(x_k)$ *and* $\tau_k > 0,\ k = 1, 2, \ldots$. *Moreover, we assume* $x_k \to x,\ t_k \to t,\ \tau_k \to 0$ *as* $k \to \infty$ *and*

$$v = \lim_{k \to \infty} \nabla \Phi_{\tau_k}(x_k, t_k).$$

Then $v \in \partial F(x,t)$.

Proof. Since $t_k \geq f(x_k)$ we have that $I_3(x_k, t_k) = \emptyset$ for $k > 0$. Moreover, one can see that there exists $k_0 > 0$ such that

$$I_1(x_k, t_k) \supseteq I_1(x, t),$$

$$I_2(x_k, t_k) \subseteq I_2(x, t)$$

for all $k \geq k_0$. We will consider two cases:

1) $t > f(x)$;

2) $t = f(x)$.

In Case 1) $I_2(x, t) = \emptyset$. Therefore $I_1(x, t) = I$ and $I_1(x_k, t_k) = I$ for all $k \geq k_0$. It follows from (4.24) that

$$\nabla \Phi_{\tau_k}(x_k, t_k) = (0_n, 1)$$

for all $k \geq k_0$. Then in this case $v = (0_n, 1)$. Hence (3.3) implies that the subdifferential of the function F at the point (x, t) is:

$$\partial F(x, t) = \{(0_n, 1)\}$$

that is $v \in \partial F(x, t)$.

In Case 2 $I_2(x, t) \neq \emptyset$. Define the following two index sets

$$I_4 = \{i \in I : \ i \in I_1(x_k, t_k) \ \ \forall \, k \geq k_0 \text{ and } f_i(x) = t\},$$

$$\bar{I}_2 = \{i \in I : \ \exists k_1 > 0 \text{ such that } i \in I_2(x_k, t_k) \ \ \forall \, k \geq k_1\}.$$

It is clear that $I_2(x, t) = I_4 \bigcup \bar{I}_2$. For any $i \in I_1(x_k, t_k)$ one can rewrite $\beta_{i\tau_k}(x_k, t_k)$ as

$$\beta_{i\tau_k}(x_k, t_k) = -\sqrt{1 - \frac{\tau_k^2}{(f_i(x_k) - t_k)^2 + \tau_k^2}}.$$

Then we get

$$\lim_{k\to\infty}\beta_{i\tau_k}(x_k,t_k)=\begin{cases}-1, & i\in I_1(x,t)\setminus I_4,\\ -\alpha,\ \alpha\in[0,1], & i\in I_4,\\ 0, & i\in\bar{I}_2.\end{cases}$$

It should be noted that the number α might not be unique. Applying (3.19) for (x_k,t_k) and $\tau_k>0$ we get

$$\begin{aligned}G_{1\tau_k}(x_k,t_k) &= \frac{1}{2}\Big[\sum_{i\in I_1(x_k,t_k)\setminus I_4}(1+\beta_{i\tau_k}(x_k,t_k))\,\nabla f_i(x_k)\\ &\quad+\sum_{i\in I_4}(1+\beta_{i\tau_k}(x_k,t_k))\,\nabla f_i(x_k)+\sum_{i\in I_2(x_k,t_k)}(1+\beta_{i\tau_k}(x_k,t_k))\,\nabla f_i(x_k)\Big].\end{aligned}$$

Then we have

$$\lim_{k\to\infty}G_{1\tau_k}(x,t)=\frac{1}{2}\Big[\sum_{i\in I_1(x,t)}0_n+\sum_{i\in I_4}(1-\alpha)\nabla f_i(x)+\sum_{i\in\bar{I}_2}\nabla f_i(x)\Big].$$

We get the following result for the second component $G_{2\tau_k}(x_k,t_k)$ of the gradient $\nabla\Phi_{i\tau_k}(x_k,t_k)$:

$$\begin{aligned}\lim_{k\to\infty}G_{2\tau_k}(x_k,t_k) &= 1-\frac{1}{2}|I|-\frac{1}{2}\left(-\sum_{i\in I_1}1-\sum_{i\in I_4}\alpha+\sum_{i\in\bar{I}_2}0\right)\\ &= 1-\frac{1}{2}\left((1-\alpha)|I_4|+|\bar{I}_2|\right).\end{aligned}$$

Thus

$$v=\left(\sum_{i\in I_4}\frac{(1-\alpha)}{2}\nabla f_i(x)+\sum_{i\in\bar{I}_2}\frac{1}{2}\nabla f_i(x),1-\frac{1}{2}\left((1-\alpha)|I_4|+|\bar{I}_2|\right)\right).$$

Using (3.3) we can observe that v can be represented as a sum of $(0_n,1)$ and convex combinations of elements with the coefficient $(1-\alpha)/2$ for $i\in I_4$ and with the coefficient $1/2$ for $i\in\bar{I}_2$. Therefore $v\in\partial F(x,t)$. This completes the proof. □

Proposition 8. *Suppose that functions $f_i, i\in I$ are continuously differentiable and their gradients*

∇f_i *are locally Lipschitz. Then the gradient* $\nabla\Phi_\tau$ *is also locally Lipschitz for any given* $\tau > 0$.

Proof. The gradient $\nabla\Phi_\tau(x,t)$ can be rewritten as:

$$\nabla\Phi_\tau(x,t) = \left(\frac{1}{2}\sum_{i\in I}\left(1+\beta_{i\tau}(x,t)\right)\nabla f_i(x), 1 - \frac{1}{2}|I| - \frac{1}{2}\sum_{i\in I}\beta_{i\tau}(x,t)\right). \tag{3.21}$$

It is obvious that the function $\beta_{i\tau}(x,t)$ is locally Lipschitz for any fixed $\tau > 0$. Then it follows from (3.21) that the gradient $\nabla\Phi_\tau$ is also locally Lipschitz. □

Proposition 9. *Suppose that functions* $f_i, i \in I$ *are twice continuously differentiable and* $Q \subset \mathbb{R}^{n+1}$ *is any bounded subset. Then for any* $(x,t) \in Q$ *and* $u = (y,s) \in \mathbb{R}^{n+1}$, $y \in \mathbb{R}^n$, $s \in \mathbb{R}$ *and for given* $\tau > 0$ *there exists an* $L = L(x,t,\tau) < \infty$ *such that*

$$\langle u, \nabla^2\Phi_\tau(x,t)u\rangle \le L\|u\|^2.$$

Proof. Using notations introduced in the proof of Proposition 8 we can write the Hessian of the function Φ_τ as

$$\nabla^2\Phi_\tau(x,t) = \begin{pmatrix} A_{n\times n} & B_n^T \\ B_n & C \end{pmatrix}$$

where

$$A_{n\times n} = \frac{1}{2}\sum_{i\in I}(\nabla^2 f_i(x) + \mu_i(x,t)\nabla f_i(x)^T\nabla f_i(x) + \beta_i(x,t))\nabla^2 f_i(x),$$

$$B_n^T = -\frac{1}{2}\sum_{i\in I}\mu_i(x,t)\nabla f_i(x)^T,$$

$$B_n = -\frac{1}{2}\sum_{i\in I}\mu_i(x,t)\nabla f_i(x),$$

$$C = \frac{1}{2}\sum_{i\in I}\mu_i(x,t).$$

Here

$$\mu_i(x,t) = \frac{\tau^2}{[(f_i(x)-t)^2+\tau^2]^{3/2}}.$$

It is easy to see that $\|\beta_i(x,t)\| \le 1$, $0 \le \mu_i(x,t) \le 1$ and $C \le \frac{1}{2}m$, where $m = |I|$.

$$\langle u, \nabla^2\Phi_\tau(x,t)u\rangle = \langle y, A_{n\times n}y\rangle + 2s\langle y, B_n\rangle + Cs^2.$$

Since $Q \subset \mathbb{R}^{n+1}$ is bounded subset by continuity of gradients $\nabla f_i,\ i \in I$ there exists a number $K < \infty$ such that $\|\nabla f_i(x)\| \le K$ and $\langle y, \nabla^2 f_i(x)y\rangle \le K\|y\|^2$. Then

$$\langle y, A_{n\times n}y\rangle \le \frac{1}{2}m(K^2+2K)\|y\|^2,$$

$$2s\langle y, B_n\rangle \le 2|s|\|y\|\|B_n\| \le mK|s|\|y\|.$$

Therefore

$$2s\langle y, B_n\rangle \le \begin{cases} mKs^2, & \|y\| \le |s|, \\ mK\|y\|^2, & |s| \le \|y\| \end{cases}$$

and

$$\langle u, \nabla^2\Phi_\tau(x,t)u\rangle \le \begin{cases} \frac{1}{2}m(K^2+2K)\|y\|^2 + m(K+\frac{1}{2})s^2, & \|y\| \le |s|, \\ \frac{1}{2}m(K^2+4K)\|y\|^2 + \frac{1}{2}ms^2, & |s| \le \|y\|. \end{cases}$$

Let $L = \max\left\{\frac{1}{2}m(K^2+2K), m(K+\frac{1}{2}), \frac{1}{2}m(K^2+4K), \frac{1}{2}m\right\}$. Hence

$$\langle u, \nabla^2\Phi_\tau(x,t)u\rangle \le L(\|y\|^2+s^2) = L\|u\|^2.$$

□

3.3 Minimization algorithm

In this section we describe an algorithm for solving the finite minimax problem (2.10).

We propose to replace Problem (2.10) by the sequence of the following smooth problems:

$$\text{minimize} \quad \Phi_{\tau_k}(x, f(x)), \tag{3.22}$$

where $\tau_k \to 0$ as $k \to \infty$. Results from Section 3.2 demonstrate that smooth optimization algorithms can be applied to solve Problem (3.22). We will call such algorithms smooth optimization solvers.

Remark 3. It should be noted that one can choose the precision parameter $\tau > 0$ sufficiently small and solve Problem (3.22) only once. However, such an approach may make Problem (3.22) ill-conditioned which will require significantly more computational efforts. The use of the sequence $\{\tau_k\}$ may help to prevent such situations.

We propose the following algorithm for solving Problem (2.10). Let $\{\tau_k\},\ \{\varepsilon_k\}$ be given sequences such that $\tau_k > 0, \varepsilon_k > 0$ and $\tau_k, \varepsilon_k \to 0$ as $k \to \infty$.

Algorithm 4. Algorithm for solving minimax problems.

Step 1 (Initialization). Select any starting point $x_0 \in \mathbb{R}^n$ and set $t_0 := f(x_0),\ k := 0$.

Step 2. Starting from the point x_k apply a smooth optimization solver to Problem (3.22) to find a point $\bar{x}$ such that

$$\|\nabla \Phi_{\tau_k}(\bar{x}, f(\bar{x}))\| < \varepsilon_k. \tag{3.23}$$

Step 3. Set $x_{k+1} := \bar{x},\ t_{k+1} := f(\bar{x}),\ k := k + 1$ and go to Step 2.

Remark 4. For some problems the choice of sequences $\{\tau_k\}$ and $\{\varepsilon_k\}$ might be important. If τ_k quickly converges to 0 then the ill-conditioned behavior of the problem may gradually increase. In this case a large number of iterations is required to satisfy the condition (3.23). In order to avoid this one should ensure that the sequence $\{\tau_k\}$ converges to 0 slower than the sequence $\{\varepsilon_k\}$.

Next we will prove the convergence of Algorithm 4. For the starting point x_0 consider the following set:

$$\mathcal{L}(x_0) = \{x \in \mathbb{R}^n :\ f(x) \leq f(x_0)\}.$$

Proposition 10. *Assume that the set $\mathcal{L}(x_0)$ is bounded for any starting point $x_0 \in \mathbb{R}^n$. Then any accumulation point of the sequence $\{x_k\}$ generated by Algorithm 4 is a stationary point of Problem (2.10).*

Proof. It is clear that $x_k \in \mathcal{L}(x_0)$ for all $k \geq 0$. Since the set $\mathcal{L}(x_0)$ is bounded the sequence $\{x_k\}$ has at least one accumulation point. Assume that x^* is an accumulation point of the sequence $\{x_k\}$ and for the sake of simplicity assume that $x_k \to x^*$ as $k \to \infty$. It follows from Proposition 19 that $0_{n+1} \in \partial F(x^*, f(x^*))$, that is $(x^*, f(x^*))$ is a stationary point of F. Then applying Proposition 3 we get that x^* is a stationary point of Problem (2.10). □

3.4 Numerical results

In this section we present results of testing Algorithm 4 using well-known nonsmooth optimization test problems. We also present comparison of this algorithm with the algorithm based on the exponential smoothing technique as well as with the algorithm based on the nonlinear programming reformulation of the minimax problem (2.10) using numerical results.

In our experiments we use Problems 2.1-7, 2.9-12, 2.14-16, 2.18-25 from Chapter 2 and Problems 3.2, 3.4-9, 3.12, 3.15, 3.17, 3.19, 3.20, 3.22-24 from Chapter 3 of [58]. The description of these problems can be found in Appendix. More specifically we used CB2, WF, SPIRAL, EVD52, Rosen-Suzuki, Polak 6, PBC3, Kowalik-Osborne, Davidson 2, OET5, OET6, EXP, PBC1, EVD61, Filter, Wong 1, Wong 2, Wong 3, Polak 2, Polak 3, Watson, Osborne 2, Crescent, CB3, DEM, QL, LQ, MIFFLIN1, MIFFLIN2, Shor, El-Attar, Gill and Maxq. We do not use all test problems from [58] because for some of them not all input data is available and in some others objective functions are unbounded from below. In addition some of these problems are not minimax problems. It should be noted that all problems from Chapter 2 of [58] (CB2, WF , SPIRAL, EVD52, Rosen-Suzuki, Polak 6, PBC3, Kowalik-Osborne, Davidson 2, OET5, OET6, EXP, PBC1, EVD61, Filter, Wong 1, Wong 2, Wong 3, Polak 2, Polak 3, Watson and Osborne 2) are minimax problems whereas objective functions in problems from Chapter 3 of [58] (Crescent, CB3, DEM, QL, LQ, MIFFLIN1, MIFFLIN2, Shor,

El-Attar, Gill and Maxq) are either maximum functions or composition of maximum functions.

We used two solvers CONOPT and SNOPT from the general algebraic modeling system GAMS for solving smoothing as well as nonlinear programming problems under consideration. CONOPT is a multi-method solver and SNOPT is a large scale sequential quadratic programming (SQP) based nonlinear programming solver. More details on CONOPT and SNOPT as well as on GAMS can be found in [39].

In implementation of Algorithm 4 we choose the sequence $\{\tau_k\}$ as follows: $\tau_{k+1} = 0.2\tau_k,\ k = 1, \ldots, p,\ \tau_1 = 10$. The same sequence was used for the algorithm based on the exponential smoothing. We tried to solve all problems with the relative accuracy 10^{-4}. In order to achieve this accuracy it is sufficient to take $p = 9$. The sequence $\{\varepsilon_k\}$ was defined by default using solvers' accuracy.

It should be noted the sequence $\{\tau_k\}$ can be chosen differently for different problems so as to significantly accelerate convergence of the algorithm, however, it is not clear how it can be done for specific problems.

We analyze the results using the performance profiles introduced in [31]. Given a set of solvers S and a set of problems P one can define performance ratio for each solver as follows:

$$r_{q,s} = \frac{t_{q,s}}{\min\{t_{q,s} : s \in S\}}.$$

Here $t_{q,s}$ stands for CPU time (or number of function evaluations or number of (sub)gradient evaluations) used by solver $s \in S$ for solving problem $q \in P$. Then a parameter r_M is chosen so that $r_M \geq r_{q,s}$ for all $q \in P$ and $s \in S$. Moreover, $r_{q,s} = r_M$ if and only if solver s does not solve problem q. The performance profile $\rho_s(\mu)$ is defined as

$$\rho_s(\mu) = \frac{1}{n_q}\text{size}\left\{q \in P : r_{q,s} \leq \mu\right\}.$$

Here n_q is the number of problems in P. It is clear that $\mu \in [1, r_M]$.

In the performance profiles, the value of $\rho_s(\mu)$ at $\mu = 1$ gives the percentage of test problems for which the corresponding algorithm is the best (it uses least iterations or function calls) and the

value of $\rho_s(\mu)$ at the rightmost abscissa gives the percentage of test problems that the corresponding algorithm can solve, that is, the robustness of the algorithm (this does not depend on the measured performance). Moreover, the relative efficiency of each algorithm can be directly seen from the performance profiles: the higher the particular curve, the better the corresponding algorithm. For more information on performance profiles, see [31].

For all problems we compare the efficiency of the algorithms both in terms of number of iterations (for CONOPT and SNOPT) and function calls (only for SNOPT). We do not compare the CPU time because for most of them the CPU time used by algorithms is close to 0.

Results of numerical experiments are presented in Figures 3.1–3.6. We present results for the single starting points from the literature and for 10 randomly generated starting points, separately. In the next two subsections we discuss results for two types of problems.

3.4.1 Results for unconstrained minimax problems

In this subsection we present results for test problems with maximum objective functions.

Figure 3.1(a) illustrates results obtained using the CONOPT solver with a single starting point given in [58]. These results demonstrate that the use of the NLP reformulation allows one to find the best solutions in almost 70% of cases, whereas the exponential smoothing gives the best results in 20% of cases and the hyperbolic smoothing achieves the best results only in 10% of cases. However, these results also demonstrate that the use of smoothing techniques leads to more reliable algorithms because they solved more problems with the required accuracy than the algorithm based on the NLP reformulation. The latter algorithm solved only about 70% of all problems. Moreover, the use of the NLP reformulation may lead to generation of infeasible solutions which is not the case for smoothing techniques. In this case the algorithm based on the exponential smoothing is slightly better than the algorithm based on the hyperbolic smoothing both in the sense of efficiency and reliability.

Figure 3.1(b) presents results obtained using the CONOPT solver with 10 randomly generated starting points. These results are very similar to those presented in Figure 3.1(a) for a single starting point. One of the main differences is that the algorithm based on the hyperbolic smoothing is slightly

better than the algorithm based on the exponential smoothing in the sense of reliability.

Figures 3.2(a) and (b) illustrate results for the number of iterations obtained using the SNOPT solver with a single starting point and with 10 randomly generated starting points, respectively. These results demonstrate that the algorithm based on the NLP reformulation is more efficient than other two algorithms (it is the best for 85% of problems with a single starting point and almost 55% of problems with 10 starting points). One can see that the algorithm based on the hyperbolic smoothing is more robust than other two algorithms (it solved all problems with a single starting point and 82% of problems with 10 starting points). Again the algorithm based on the NLP reformulation produced infeasible solutions.

Results presented in Figures 3.3(a) and (b) using the number of function calls are similar to those given in Figures 3.2(a) and (b), respectively. Again one can see from these figures that the algorithm based on the NLP reformulation is the most efficient and the algorithm based on the hyperbolic smoothing is the most robust one.

3.4.2 Results for general nonsmooth optimization problems

In this subsection we present results for test problems with both maximum objective functions and objective functions represented as a sum of maximum functions.

Figures 3.4 (a) and (b) present results obtained using the CONOPT with a given single and 10 random starting points, respectively. These results show that the algorithm based on the NLP reformulations is the most efficient and the algorithm based on the exponential smoothing is the most robust for this type of problems.

Results based on the number of iterations and function calls for SNOPT solver with a given single and 10 random starting points are presented in Figures 3.5 (a), (b) and Figures 3.6 (a), (b), respectively. These results show that the algorithm based on the NLP reformulations is again the most efficient and the algorithm based on the hyperbolic smoothing (exponential smoothing in the case of a single starting point) is the most robust for this type of problems.

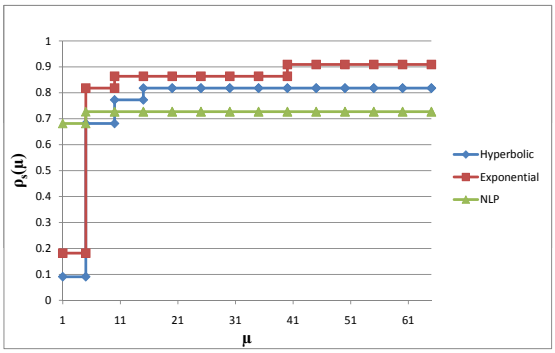
3.5 Conclusions

In this chapter, we studied the hyperbolic smoothing function for the general finite minimax problems. In order to apply the hyperbolic smoothing we reformulated the maximum function using one additional variable. We studied the relationship between the set of stationary points of the original minimax problem and the reformulated one including relationship between their sets of local minimizers. We approximated the maximum objective function by using its reformulation and applying the hyperbolic smoothing function. This smoothing allows us to apply smooth optimization solvers for solving minimax problems. We applied two solvers: CONOPT and SNOPT from the GAMS. We presented results of numerical experiments using nonsmooth optimization test problems with objective functions represented as a maximum of finite smooth functions and also as a sum of maximum functions. We also compared the algorithm based on the hyperbolic smoothing functions with the algorithm based on the exponential smoothing function and also with the algorithm based on the NLP reformulation. Based on the results presented we can draw the following conclusions:

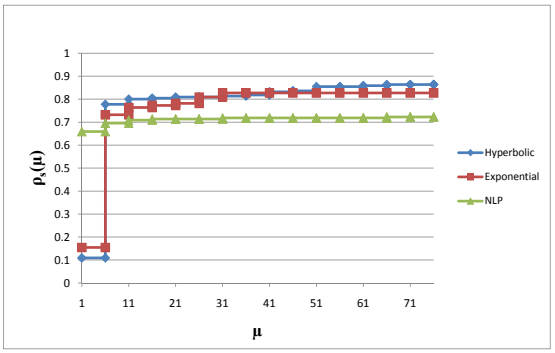
1. The algorithm based on the NLP reformulation is fastest among three algorithms and it is the most efficient, although it may not always find feasible solution which is not the case for the algorithms based on smoothing techniques.

2. Algorithms based on the both hyperbolic and exponential smoothing techniques are more robust than the algorithm based on the NLP reformulation.

3. Results also demonstrate that outcomes of the algorithms based on the smoothing techniques depend on the optimization solver used.

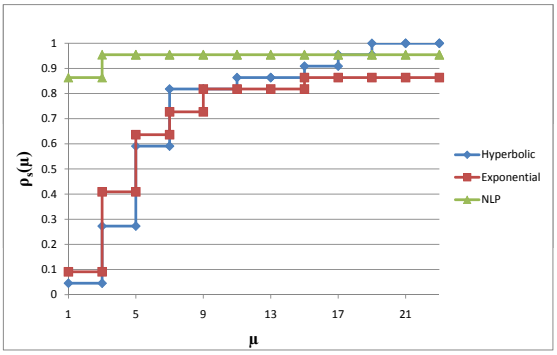


(a) Number of iterations (single starting point)

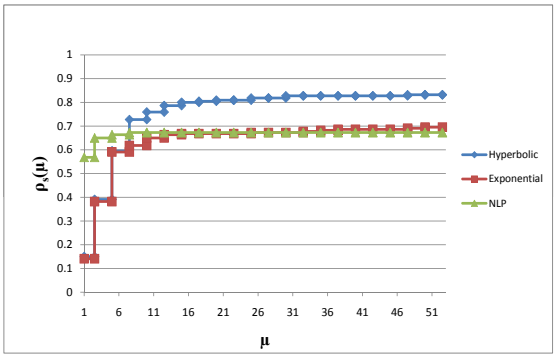


(b) Number of iterations (10 starting points)

Figure 3.1: Number of CONOPT iterations for unconstrained minimax problems.

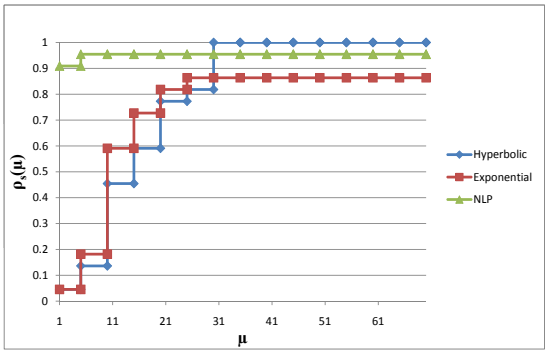


(a) Number of iterations (single starting point)

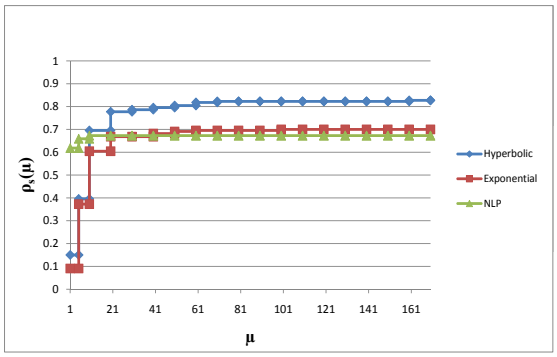


(b) Number of iterations (10 starting points)

Figure 3.2: Number of SNOPT iterations for unconstrained minimax problems.

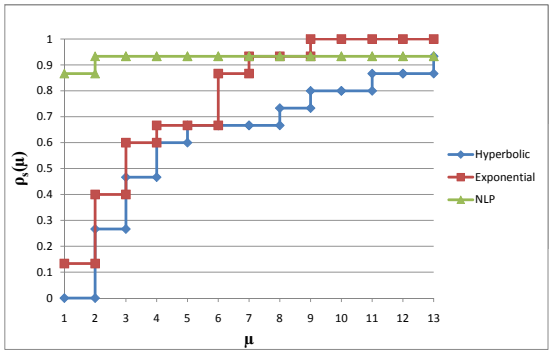


(a) Number of function calls (single starting point)

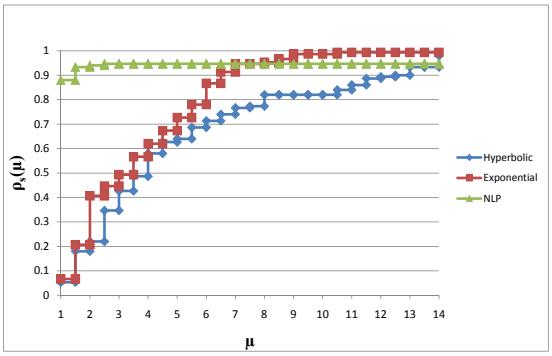


(b) Number of function calls (10 starting points)

Figure 3.3: Number of SNOPT function calls for unconstrained minimax problems.

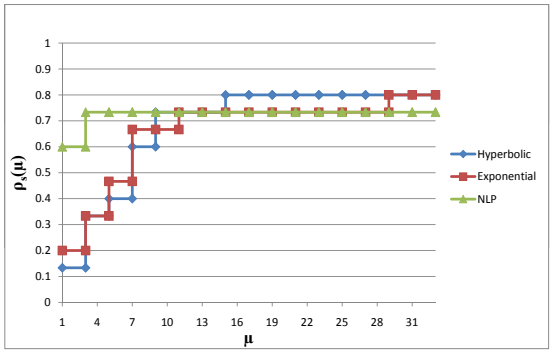


(a) Number of iterations (single starting point)

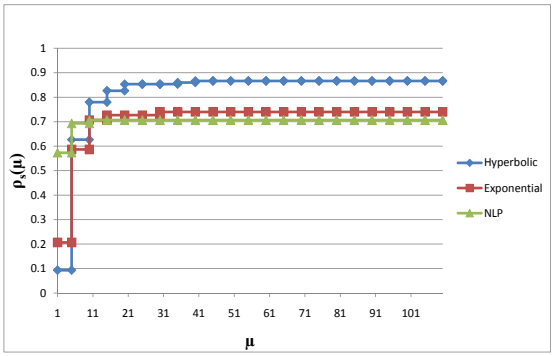


(b) Number of iterations (10 starting points)

Figure 3.4: Number of CONOPT iterations for general nonsmooth optimization problems.

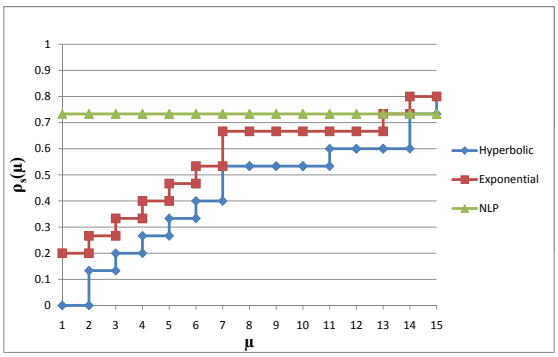


(a) Number of iterations (single starting point)

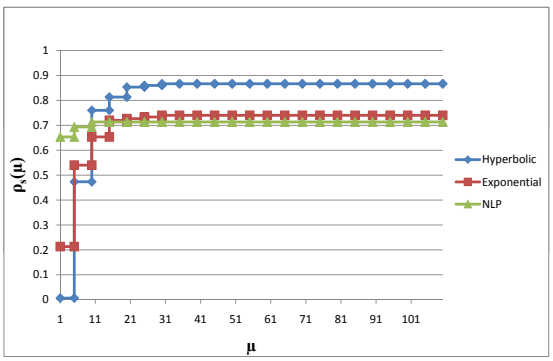


(b) Number of iterations (10 starting points)

Figure 3.5: Number of SNOPT iterations for general nonsmooth optimization problems.



(a) Number of function calls (single starting point)



(b) Number of function calls (10 starting points)

Figure 3.6: Number of SNOPT function calls for general nonsmooth optimization problems.

Chapter 4

Nonsmooth optimization via smooth optimization

In this chapter, we propose a different approach for solving general nonsmooth nonconvex optimization problems. In this approach the problem of finding search directions is reduced to the minimization of a convex piecewise linear function over the unit ball. The hyperbolic smoothing technique is applied to approximate the convex piecewise linear function by a smooth function which is minimized to find search directions. Such an approach allows us to apply powerful methods of smooth optimization for finding search directions in nonsmooth optimization problems.

We study the convergence of the proposed algorithm. The algorithm is implemented in Fortran 95. Results of numerical experiments are reported and the proposed algorithm is compared with five other nonsmooth optimization algorithms. We also implement the algorithm in GAMS and compare it with GAMS solvers using results of numerical experiments.

The structure of the chapter is as follows. We describe an algorithm for the computation of descent directions in Section 4.2. Section 4.3 presents an algorithm for solving the subproblem to find search directions. The proposed minimization algorithm is studied in Section 4.4. We present the results of numerical experiments and their discussion in Section 4.6. Section 4.7 concludes the chapter.

4.1 Quasisecants and their Properties

In this section, we present the definition and some properties of quasisecants. This notion is crucial to design the minimization algorithm in the next sections. Unlike convex case, in general, in nonconvex case not all subgradients provide a good local approximation to a function. Therefore, we try to choose those subgradients which provide either overestimation or underestimation in some neighborhood of a point. Quasisecants are in fact approximate subgradients and they provide overestimation to a function in some neighborhood of a point.

Take any direction $d \in S_1$ and any number $h > 0$. Let

$$\partial_{d,h} f(x) := \bigcup_{t \in [0,h]} \partial f(x + td).$$

The set $\partial_{d,h} f(x)$ is a union of all subdifferentials over the set conv $\{x, x + hd\}$. It is obvious that for any $\varepsilon > 0$ there exists $\delta > 0$ such that

$$\partial_{d,h} f(x) \subset \partial f(x) + B_\varepsilon, \ \forall h \in]0, \delta[.$$

Definition 13. *A vector $v \in \mathbb{R}^n$ is called a quasisecant of a locally Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ at the point x in the direction $d \in S_1$ with the length $h > 0$ iff*

$$f(x + hd) - f(x) \leq h\langle v, d\rangle \tag{4.1}$$

and

$$v \in \partial_{d,h} f(x) + B_{O(h)}. \tag{4.2}$$

Here $O(h) \geq 0$ for all $h \geq 0$ and $O(h) \to 0$ as $h \downarrow 0$. In general, O depends also on x. We call (4.1) *a quasisecant inequality*.

Remark 5. By the Lebourg's mean value theorem (see [25]) for any $x, y \in \mathbb{R}^n$ there exists $\alpha \in]0,1[$ such that

$$f(y) - f(x) = \langle v, y - x \rangle \tag{4.3}$$

for some $v \in \partial f(\alpha x + (1-\alpha)y)$. This theorem demonstrates that quasisecants always exist for locally Lipschitz functions (even with $O(h) \equiv 0$). Unfortunately, one cannot apply Lebourg's theorem to compute quasisecants. Therefore, we relax two conditions in this theorem to define quasisecants. We replace the equality (4.3) by the inequality (4.1) and also the strict condition $v \in \partial f(\alpha x + (1-\alpha)y)$ by the more tractable condition (4.2).

The notion of the quasisecant was first introduced in [7]. It was demonstrated that quasisecants can be efficiently computed for some nonsmooth functions. These include convex functions, functions represented as a maximum of a finite number of smooth functions, functions represented as a difference of two convex (DC) functions. Some interesting functions such as functions represented as a maxima of minima of a finite number of smooth convex functions (which includes continuous nonconvex piecewise linear functions), functions represented as a sum of minima of a finite number of smooth convex functions can be easily represented as a difference of two (nonsmooth) convex functions. However, in general, the computation of quasisecants is not an easy task.

Here we will demonstrate the calculation of quasisecants for convex and DC functions. For the convex function f one has

$$f(x + hd) - f(x) \leq h\langle v, d \rangle, \ \forall v \in \partial f(x + hd).$$

This means that any subgradient $v \in \partial f(x + hd)$ is a quasisecant at the point x in the direction d with the length h. Moreover, for convex functions in (4.2) $O(h) \equiv 0$ for all $h > 0$.

Now consider the function

$$f(x) = f_1(x) - f_2(x),$$

where f_1 and f_2 are convex functions. The function f is quasidifferentiable [29] and its subdifferential

$\overline{\partial} f(x)$ and superdifferential $\underline{\partial} f(x)$ at x are as follows:

$$\overline{\partial} f(x) = \partial f_1(x), \;\; \underline{\partial} f(x) = -\partial f_2(x).$$

Assume that the sets $\partial f_1(x)$ and $\partial f_2(x)$ are polytopes (this condition is satisfied for all DC functions mentioned above).

Take any $v_1 \in \partial f_1(x + hd), \;\; v_2 \in \partial f_2(x)$ and compute $v = v_1 - v_2$. Then it follows from convexity of the functions f_1 and f_2 that

$$f(x + hd) - f(x) \leq h\langle v, d \rangle.$$

However not all vectors v, defined in this way, are approximate subgradients of the function f at x, that is not all such vectors v satisfy the condition (4.2) in the definition of quasisecants. In order to compute vectors which satisfy both conditions (4.1) and (4.2) we use the following scheme from [5, 6]. For sufficiently small number $\alpha \in]0, 1[$ define n vectors:

$$\begin{aligned} e_1(\alpha) &:= (\alpha, 0, \ldots, 0), \\ e_2(\alpha) &:= (\alpha, \alpha^2, 0, \ldots, 0), \\ \ldots &:= \ldots, \\ e_n(\alpha) &:= (\alpha, \alpha^2, \ldots, \alpha^n). \end{aligned}$$

Compute subgradients $v_1 \in \partial f_1(x + hd)$ and $v_2 \in \partial f_2(x)$ such that

$$\begin{aligned} v_1 &:= \text{argmax} \{\langle v, e_j(\alpha) \rangle : \; v \in \partial f_1(x + hd)\}, \; j = 1, \ldots, n, \\ v_2 &:= \text{argmax} \{\langle w, e_j(\alpha) \rangle : \; w \in \partial f_2(x)\}, \; j = 1, \ldots, n. \end{aligned}$$

It is shown in [5, 6] that if the subdifferentials $\partial f_1(x + hd)$ and $\partial f_2(x)$ are polytopes then there exists $\alpha_0 > 0$ such that subgradients v_1 and v_2 exist and they are unique for all $\alpha \in]0, \alpha_0[$. Moreover, the

vector $v = v_1 - v_2$ is an approximate subgradient of f satisfying (4.2). Thus, the vector v, computed using this scheme, satisfies both conditions (4.1) and (4.2) and therefore it is a quasisecant of f at x. Furthermore, this quasisecant can be approximated applying the algorithm from [5, 6] which uses only values of f.

We will use $v(x, d, h)$ to denote the quasisecant v at the point x in the direction d and with the length $h > 0$. We define the set $Q_h(x)$ of all possible quasisecants at the point x with the given length $h > 0$:

$$Q_h(x) := \{w \in \mathbb{R}^n : \exists d \in S_1 \text{ such that } w = v(x, d, h)\}$$

and its closed convex hull

$$W_h(x) := \overline{conv} Q_h(x).$$

Since the subdifferential $\partial f(x)$ is a convex compact set at any $x \in \mathbb{R}^n$ we get that the set $Q_h(x)$ is bounded at any x for given $h > 0$. Moreover, the set $W_h(x)$ is convex and compact.

Finally, we define the set $Q_0(x)$ of limit points of quasisecants at the point x:

$$Q_0(x) := \left\{ w \in \mathbb{R}^n : \exists (\{d_k\} \subset S_1,\ \{h_k > 0\}) \text{ s.t. } \lim_{k\to\infty} h_k = 0 \text{ and } w = \lim_{k\to\infty} v(x, d_k, h_k) \right\}$$

and its convex hull

$$W_0(x) := \text{conv}\, Q_0(x).$$

It is obvious that the set $Q_0(x)$ is compact and the set $W_0(x)$ is convex and compact.

Next we will study relationships between the subdifferential $\partial f(x)$ and the sets $Q_h(x)$, $W_h(x)$, $Q_0(x)$, $W_0(x)$. In order to establish these relationships we need an additional assumption on $O(x, h)$ in (4.2) of the definition of quasisecants.

Assumption 1. *At any given point $x \in \mathbb{R}^n$ there exists $\delta = \delta(x) > 0$ such that $O(y, h) \downarrow 0$ uniformly as $h \downarrow 0$ for all $y \in B_\delta(x)$ that is for any $\varepsilon > 0$ there exists $h(\varepsilon) > 0$ such that $O(y, h) < \varepsilon$ for all $h \in]0, h(\varepsilon)[$ and $y \in B_\delta(x)$.*

Convex functions satisfy this assumption since in this case $O(x,h) \equiv 0$ for all $x \in \mathbb{R}^n$ and $h > 0$.

Proposition 11. *At a given point $x \in \mathbb{R}^n$ for any $\varepsilon > 0$ there exists $h(\varepsilon) > 0$ such that*

$$Q_h(x) \subset \partial f(x) + B_\varepsilon \quad \textit{and} \quad W_h(x) \subset \partial f(x) + B_\varepsilon, \ \forall h \in]0, h(\varepsilon)[.$$

Proof. The proof follows from the upper semicontinuity of the subdifferential mapping and Definition 13 of quasisecants. □

Proposition 12. *Assume that a function f satisfies Assumption 1. Then at a given point $x \in \mathbb{R}^n$ for any $\varepsilon > 0$ there exist $\delta = \delta(\varepsilon) > 0$ and $h(\varepsilon) > 0$ such that*

$$Q_h(y) \subset \partial f(x) + B_\varepsilon \ \textit{and} \ W_h(y) \subset \partial f(x) + B_\varepsilon$$

for all $h \in]0, h(\varepsilon)[$ and $y \in B_\delta(x)$.

Proof. It follows from the definition of the set $Q_h(y)$ that

$$Q_h(y) \subset \bigcup_{d \in S_1} \partial_{d,h} f(y) + B_{O(y,h)}.$$

Since the function f satisfies Assumption 1, for any $\varepsilon > 0$ there exist $\delta_1 = \delta_1(\varepsilon) > 0$ and $h_1(\varepsilon) > 0$ such that $O(y,h) < \varepsilon$ for all $y \in B_{\delta_1}(x)$ and $h \in]0, h_1(\varepsilon)[$. Then we have

$$Q_h(y) \subset \bigcup_{d \in S_1} \partial_{d,h} f(y) + B_\varepsilon$$

for all $y \in B_{\delta_1}(x)$ and $h \in]0, h_1(\varepsilon)[$. Upper semicontinuity of the subdifferential mapping implies

that for $\varepsilon > 0$ there exist $\delta_2 = \delta_2(\varepsilon) > 0$ and $h_2(\varepsilon) > 0$ such that

$$\bigcup_{d \in S_1} \partial_{d,h} f(y) \subset \partial f(x) + B_\varepsilon$$

for all $y \in B_{\delta_2}(x)$ and $h \in]0, h_2(\varepsilon)[$. Then by taking $\delta = \delta(\varepsilon) = \min\{\delta_1(\varepsilon), \delta_2(\varepsilon)\}$ and $h(\varepsilon) = \min\{h_1(\varepsilon), h_2(\varepsilon)\}$ we complete the proof. □

Proposition 13. *Assume that a function $f : \mathbb{R}^n \to \mathbb{R}$ is locally Lipschitz. Then*

$$Q_0(x) \subset \partial f(x) \ \textit{and} \ W_0(x) \subset \partial f(x), \ \ x \in \mathbb{R}^n.$$

Proof. It follows from Proposition 11 and the definition of the sets $Q_0(x)$ and $W_0(x)$. □

Proposition 14. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a locally Lipschitz function at x. Then*

$$\max_{w \in W_0(x)} \langle w, d \rangle \leq f^0(x, d) \ \ \forall d \in \mathbb{R}^n.$$

If, in addition, the function f is also directionally differentiable, then

$$f'(x, d) \leq \max_{w \in W_0(x)} \langle w, d \rangle \ \ \forall d \in \mathbb{R}^n.$$

Proof. The first inequality follows from Proposition 13 and the second one follows from the definitions of quasisecants and the set $W_0(x)$. □

The following corollary follows from Proposition 14.

Corollary 1. *Suppose that the function $f : \mathbb{R}^n \to \mathbb{R}$ is regular at x. Then*

$$\partial f(x) = W_0(x).$$

4.2 Computation of descent directions

Consider the following unconstrained minimization problem:

$$\text{minimize } f(x) \text{ subject to } x \in \mathbb{R}^n \tag{4.4}$$

where the objective function f is locally Lipschitz. This function is not necessarily differentiable or convex. Furthermore, there is not any assumption on the structure of this function.

We start with the definition of the (h, δ)-stationary points (see, also [7]).

Definition 14. *Let $h > 0$ and $\delta > 0$ be given numbers. A point x is called an (h, δ)-stationary point for Problem (4.4) iff:*

$$\min\{\|v\| : v \in W_h(x)\} \leq \delta.$$

For sufficiently small h and δ the (h, δ)-stationary point can also be considered as an approximate stationary point. Indeed, it follows from Proposition 11 that at a point $x \in \mathbb{R}^n$ for any $\varepsilon > 0$ there exists $h(\varepsilon) > 0$ such that

$$W_h(x) \subset \partial f(x) + B_\varepsilon$$

for all $h \in (0, h(\varepsilon))$. If x is an (h, δ)-stationary for some $h \in (0, h(\varepsilon))$ and $\delta > 0$ then

$$0 \in \partial f(x) + B_{\varepsilon+\delta}. \tag{4.5}$$

In this section we present an algorithm for the computation of descent directions of the objective function f in Problem (4.4). Let numbers $h > 0, c \in (0, 1)$ and the tolerance $\delta > 0$ be given.

Algorithm 5. An algorithm for the computation of the descent direction.

Step 1. Choose any $d_1 \in S_1$, compute a quasisecant $v_1 = v(x, d_1, h)$. Set $V_1(x) := \{v_1\}$ and $k := 1$.

Step 2. Compute $\bar{d}$ as the solution to the following minimization problem:

$$\text{minimize} \ \max_{i=1,\ldots,k} \langle v_i, d \rangle \ \text{ s.t. } d \in S_1. \tag{4.6}$$

If $D_k \equiv \max_{i=1,\ldots,k} \langle v_i, \bar{d} \rangle > -\delta$, then stop. Otherwise go to Step 3.

Step 3. If

$$f(x + h\bar{d}) - f(x) \leq chD_k, \tag{4.7}$$

then stop. Otherwise set $d_{k+1} = \bar{d}$ and go to Step 4.

Step 4. Compute a quasisecant $v_{k+1} = v(x, d_{k+1}, h)$, construct the set

$$V_{k+1}(x) = \text{co}\,\{V_k(x) \bigcup \{v_{k+1}\}\},$$

set $k := k + 1$ and go to Step 2.

The most important step in Algorithm 5 is Step 2 where one solves the minimization problem (4.6) to find search directions.

Next we will show that Algorithm 5 is terminating, that is after a finite number of steps we either conclude that x is an (h, δ)-stationary point or find the descent direction. First, we will prove the following propositions.

Proposition 15. *If $D_k > -\delta$, then*

$$\min_{v \in V_k(x)} \|v\| < \delta. \tag{4.8}$$

Proof. Let $\tilde{v}$ be a solution to the following problem:

$$\min \ \frac{1}{2}\|v\|^2 \ \ \text{subject to} \ \ v \in V_k(x). \tag{4.9}$$

If $\tilde{v} = 0$ then (4.8) is true. Now we assume that $\tilde{v} \neq 0$. Since $\tilde{v}$ is the solution to (4.9) it follows from the necessary condition for a minimum that

$$\langle \tilde{v}, v - \tilde{v} \rangle \geq 0, \ \ \forall v \in V_k(x)$$

which means

$$\|\tilde{v}\|^2 \leq \langle \tilde{v}, v \rangle, \ \ \forall v \in V_k(x). \tag{4.10}$$

Since $D_k > -\delta$ then

$$\max_{i=1,\ldots,k} \langle v_i, d \rangle > -\delta, \ \ \forall d \in S_1. \tag{4.11}$$

Consider $\tilde{d} = -\tilde{v}/\|\tilde{v}\|$. Then it follows from (4.11) that there exists $i \in \{1, \ldots, k\}$ such that

$$\langle \tilde{v}, v_i \rangle < \delta \|\tilde{v}\|.$$

Then the proof follows from (4.10). □

Corollary 2. *If $D_k > -\delta$, then the point x is an (h, δ)-stationary point.*

Proof. Since $V_k(x) \subset W_h(x)$ it follows from Proposition 15 that in this case

$$\min_{v \in W_h(x)} \|v\| \leq \delta.$$

This completes the proof. □

Remark 6. It follows from Proposition 15 that if $D_k > -\delta$ in Step 2 of Algorithm 5, then the point $x \in \mathbb{R}^n$ is an approximate stationary point satisfying (4.5) for given $\delta > 0$ and for some $\varepsilon > 0$.

Corollary 3. *If $D_k \geq 0$ then $0 \in V_k(x)$.*

Proof. Assume the contrary, that is $0 \notin V_k(x)$. Then $\tilde{v} \neq 0$, where $\tilde{v}$ is the solution to Problem (4.9).

Since $D_k \geq 0$ then

$$\max_{i=1,\ldots,k} \langle v_i, d \rangle \geq 0, \ \ \forall d \in S_1. \tag{4.12}$$

Consider $\tilde{d} = -\tilde{v}/\|\tilde{v}\|$. Then it follows from (4.12) that there exists $i \in \{1, \ldots, k\}$ such that

$$\langle \tilde{v}, v_i \rangle \leq 0.$$

However, it follows from (4.10) that $\langle \tilde{v}, v_i \rangle \geq \|\tilde{v}\|^2 > 0 \ \forall i \in \{1, ..., k\}$. We have arrived at a contradiction, which completes the proof. □

Proposition 16. *If* $\min\limits_{v \in V_k(x)} \|v\| < \delta$ *then* $D_k > -\delta$.

Proof. Assume the contrary that is $\min\limits_{v \in V_k(x)} \|v\| < \delta$ but $D_k \leq -\delta$. This means that

$$\langle v_i, \bar{d} \rangle \leq -\delta, \ i = 1, \ldots, k,$$

where $\bar{d} \in S_1$ is the solution of the problem (4.6). Let

$$\|\tilde{v}\| = \min_{v \in V_k(x)} \|v\|.$$

Since $\tilde{v} \in V_k(x)$

$$\tilde{v} = \sum_{i \in I} \alpha_i v_i, \quad \sum_{i \in I} \alpha_i = 1, \ \ \alpha_i \in (0, 1], \ i \in I \subseteq \{1, \ldots, k\}.$$

We get

$$\langle \tilde{v}, \bar{d} \rangle \leq -\delta. \tag{4.13}$$

On the other hand

$$\left| \langle \tilde{v}, \bar{d} \rangle \right| \leq \|\tilde{v}\| \|\bar{d}\| = \|\tilde{v}\| < \delta$$

which contradicts (4.13). □

Proposition 17. *Let f be a locally Lipschitz function defined on $\mathbb{R}^n$ and $L > 0$ be a Lipschitz constant at x. Then Algorithm 5 terminates after finite number of steps.*

Proof: If both conditions for the termination of the algorithm are not satisfied, then a quasisecant $v^{k+1} \notin V_k(x)$. Indeed, in this case

$$f(x + h\bar{d}) - f(x) > chD_k.$$

It follows from the definition of the quasisecant v^{k+1} that

$$f(x + h\bar{d}) - f(x) \leq h\langle v_{k+1}, \bar{d}\rangle$$

and therefore

$$\langle v_{k+1}, \bar{d}\rangle > cD_k.$$

Then

$$-\langle v_{k+1}, \bar{d}\rangle < -cD_k \leq -c(-\delta)$$

and

$$\langle v_{k+1}, \bar{d}\rangle > -c\delta. \tag{4.14}$$

Assume the contrary that is $v_{k+1} \in V_k(x)$. Since $D_k \leq -\delta$

$$\langle v_i, \bar{d}\rangle \leq -\delta,\ i = 1, \ldots, k.$$

Since $v_{k+1} \in V_k = \text{co}\,\{v_1, ..., v_k\}$ we can write it as a convex combination of the v_i

$$v_{k+1} = \sum_{i=1}^{k} \alpha_i v_i$$

where $\sum_{i=1}^{k} \alpha_i = 1$ and $\alpha_i \geq 0,\ i = 1, \ldots, k$. Then

$$\langle v_{k+1}, \bar{d} \rangle = \left\langle \sum_{i=1}^{k} \alpha_i v_i, \bar{d} \right\rangle = \sum_{i=1}^{k} \langle \alpha_i v_i, \bar{d} \rangle = \sum_{i=1}^{k} \alpha_i \langle v_i, \bar{d} \rangle \leq -\delta,$$

which contradicts (4.14).

Now we will show that Algorithm 5 is terminating. Assume the contrary. Then Algorithm 5 generates an infinite sequence $\{d_k\}$ of directions $d_k \in S_1$. (4.14) implies that

$$\langle v_k, d_k \rangle > -c\delta, \ \ \forall\, k = 2, 3, \ldots. \tag{4.15}$$

It follows from Theorem 3.1.4 [61] that $\|v\| \leq L$ for all $v \in \partial f(x)$. The direction d_{k+1} is a solution to the minimization problem (4.6). Since we assume that algorithm is not terminating we have

$$\max_{i=1,\ldots,k} \langle v_i, d_{k+1} \rangle \leq -\delta.$$

Therefore d_{k+1} is the solution to the system

$$\langle v_i, d \rangle + \delta \leq 0,\ i = 1, \ldots, k.$$

Then we get

$$\|d_{k+1} - d_j\| > \frac{(1-c)\delta}{L}, \ \ \forall\, j = 2, \ldots, k. \tag{4.16}$$

Indeed, if there exists $j \in \{2, \ldots, k\}$ such that

$$\|d_{k+1} - d_j\| \leq \frac{(1-c)\delta}{L}$$

then we have

$$|\langle v_j, d_{k+1} \rangle - \langle v_j, d_j \rangle| \leq (1-c)\delta.$$

This means that

$$\langle v_j, d_j \rangle \leq \langle v_j, d_{k+1} \rangle + (1-c)\delta \leq -c\delta$$

which contradicts (4.15). The inequality (4.16) can be rewritten as follows:

$$\min_{j=2,\ldots,k} \|d_{k+1} - d_j\| > \frac{(1-c)\delta}{L}.$$

Thus Algorithm 5 generates a sequence $\{d_k\}$ of directions $d_k \in S_1$ such that the distance between d_k and the set of all previous directions is bounded below. Since the set S_1 is bounded the number of such directions is finite. □

4.3 Solving subproblem for finding search directions

In this section we design an algorithm for solving Problem (4.6) in Step 2 of Algorithm 5, which is the most important step in this algorithm. In order to solve this problem we reduce it to the minimization of the convex piecewise linear function over the unit ball. This problem can be solved using two different approaches. In the first approach we replace the constrained problem by the unconstrained one using a distance function, whereas in the second approach we consider the problem as is. In both cases we apply hyperbolic smoothing technique to replace problems by the sequence of smooth problems. Then smooth optimization solvers are applied to solve them. In the rest of this section we describe these two approaches in detail.

Recall that the objective function in Problem (4.6) is as follows:

$$\varphi_k(d) = \max_{i=1,\ldots,k} \langle v_i, d \rangle. \tag{4.17}$$

Then Problem (4.6) can be rewritten as

$$\text{minimize } \varphi_k(d) \quad \text{subject to } \quad d \in S_1. \tag{4.18}$$

In addition to Problem (4.18) we also consider the following convex programming problem:

$$\text{minimize} \quad \varphi_k(d) \quad \text{subject to} \quad d \in B_1 = \{y \in \mathbb{R}^n : \ \|y\| \leq 1\}. \tag{4.19}$$

Denote by G_{1k} and G_{2k} sets of solutions and by D_{1k} and D_{2k} the optimal values of Problems (4.18) and (4.19), respectively. The relationship between these two problems are established in the following proposition.

Proposition 18. *1) If $D_{2k} = 0$ then $D_{1k} \geq 0$. If in this case $D_{1k} = 0$ then $G_{1k} \subset G_{2k}$, otherwise $G_{1k} \not\subset G_{2k}$.*

2) If $D_{2k} < 0$ then $G_{1k} = G_{2k}$ and $D_{1k} = D_{2k}$.

Proof. Notice that the objective function φ_k is positively homogeneous that is

$$\varphi_k(\lambda d) = \lambda \varphi_k(d) \ \ \forall d \in \mathbb{R}^n \text{ and } \lambda \geq 0. \tag{4.20}$$

Since $0 \in B_1$ always $D_{2k} \leq 0$. Moreover, since $S_1 \subset B_1$ one has $D_{1k} \geq D_{2k}$. Therefore we consider only two cases: 1) $D_{2k} = 0$ and 2) $D_{2k} < 0$.

Case 1) If $D_{2k} = 0$ then $D_{1k} \geq 0$, which triggers the stopping criterion in Algorithm 5. Moreover, in this case $0 \in V_k(x)$ according to Corollary 3. Now take any $\bar{d} \in G_{1k}$. If $D_{1k} = 0$ then $\varphi_k(\bar{d}) = 0$ and $\varphi_k(d) \geq \varphi_k(\bar{d}) = 0$ for all $d \in S_1$. Since for any $d \in B_1$ there exists $d_0 \in S_1$ and $\lambda \in [0, 1]$ such that $d = \lambda d_0$. Then it follows from (4.20) that $\varphi_k(d) \geq 0$ for all $d \in B_1$. Therefore $\bar{d} \in G_{2k}$. If $D_{1k} > 0$ then (4.20) implies that $\varphi_k(d) > 0$ for all $d \in B_1, \ d \neq 0$. This means that in this case $G_{2k} = \{0\}$ and therefore $G_{1k} \not\subset G_{2k}$.

Case 2) Now let us consider the case when $D_{2k} < 0$. We will prove that in this case $G_{1k} = G_{2k}$ which means that $D_{1k} = D_{2k}$. Let d_0 be the solution to the problem (4.19). It is clear that $d_0 \neq 0$. Then $d_0 \in S_1$. Indeed, assume that $d_0 \notin S_1$. Then we can find $\bar{d} = \lambda d_0$, where $\lambda > 1$, $\bar{d} \in S_1 \subset B_1$. Applying (4.20) we have

$$\varphi_k(\bar{d}) = \lambda \varphi_k(d_0) = \lambda D_{2k} < D_{2k}$$

which contradicts that d_0 is the solution to Problem (4.19). Then we get that all solutions of Problem (4.19) lie on the unit sphere S_1. This means that $G_{1k} = G_{2k}$ and therefore $D_{1k} = D_{2k}$. □

An important consequence of this proposition is that we can replace the nonconvex minimization problem (4.18) by the convex programming problem (4.19). It is obvious from this proposition that solving Problem (4.19) we get exactly either the same stopping criterion or the same descent direction as can be obtained by solving (4.18).

We apply smoothing techniques to replace Problem (4.19) by the sequence of smooth problems. The hyperbolic smoothing technique is used for this purpose [98, 99]. This technique for minimax problems was studied in [8]. Next, we apply results from [8] to Problem (4.19). Consider the function:

$$F_k(d,t) = t + \sum_{i=1}^{k} \max(0, \langle v_i, d\rangle - t).$$

It is clear that $F_k(d, \varphi_k(d)) = \varphi_k(d)$. It is proved in [8] that the set of minimizers of functions F_k and φ_k coincide when $t = \varphi_k(d)$. Define a function

$$\Phi_k(d) \equiv F_k(d, \varphi_k(d)),\ d \in \mathbb{R}^n.$$

Then we can reformulate the problem (4.19) as follows:

$$\text{minimize } \Phi_k(d) \quad \text{subject to } d \in B_1. \tag{4.21}$$

For a given $(d, \varphi_k(d))$ the index set I can be represented as follows:

$$I = I_1 \cup I_2,$$

where

$$I_1 = \{i \in I : \langle v_i, d\rangle < \varphi_k(d)\},$$

$$I_2 = \{i \in I : \langle v_i, d\rangle = \varphi_k(d)\}.$$

The subdifferential of the function Φ_k at d can be written as follows:

$$\partial\Phi_k(d) = \{(0_n, 1)\} + \sum_{i\in I_2} \text{co}\,\{0_{n+1}, (v_i, -1)\}. \tag{4.22}$$

Applying (3.14) to the function Φ_k we get:

$$H_\tau(d) = \varphi_k(d) + \sum_{i\in I} \frac{\langle v_i, d\rangle - \varphi_k(d) + \sqrt{(\langle v_i, d\rangle - \varphi_k(d))^2 + \tau^2}}{2},\ \tau > 0. \tag{4.23}$$

According to Proposition 5 from [8]

$$0 < H_\tau(d) - \Phi_k(d) \le \frac{k\tau}{2}.$$

The gradient of the function H_τ is as follows:

$$\nabla H_\tau(d) = (G_{1\tau}(d), G_{2\tau}(d)) \tag{4.24}$$

where

$$G_{1\tau}(d) = \frac{1}{2}\sum_{i\in I} (1 + \beta_{i\tau}(d))\, v_i, \tag{4.25}$$

$$G_{2\tau}(d) = 1 - \frac{1}{2}|I| - \frac{1}{2}\sum_{i\in I} \beta_{i\tau}(d). \tag{4.26}$$

$$\beta_{i\tau}(d) = \frac{\langle v_i, d\rangle - \varphi_k(d)}{\sqrt{(\langle v_i, d\rangle - \varphi_k(d))^2 + \tau^2}}. \tag{4.27}$$

From Proposition 6 [8] it follows that if $z = \lim\limits_{\tau\to 0} \nabla\Psi_\tau(d)$ then $z \in \partial F(d, \varphi(d))$. The proof of the following proposition can be found in [8].

Proposition 19. *Assume that sequences $\{d_k\}$ and $\{\tau_k\}$ are given such that $d_k \in \mathbb{R}^n$ and $\tau_k > 0,\ k =$*

$1, 2, \ldots$. *Moreover,* $d_k \to d,\ \tau_k \to 0$ *as* $k \to \infty$ *and*

$$z = \lim_{k \to \infty} \nabla \Psi_{\tau_k}(d_k).$$

Then $z \in \partial F(d, \varphi(d))$.

We therefore can replace a nonsmooth optimization problem (4.21) by the following smooth problem

$$\text{minimize} \quad \Psi_{k\tau}(d) \quad \text{subject to} \quad d \in B_1. \tag{4.28}$$

One can apply any smooth optimization solver to solve the constrained problem (4.28).

Another option is to reduce the constrained problem (4.28) to an unconstrained one. In order to do so we apply Lemma 5.1.5 from [61] to Problem (4.28) and replace it by the following unconstrained problem:

$$\text{minimize} \ \ \Psi_{k\tau}(d) + \widehat{L} g_{B_1}(d) \quad \text{subject to} \quad d \in \mathbb{R}^n \tag{4.29}$$

where $\widehat{L} \geq L$ and $L > 0$ is the Lipschitz constant of the function φ_k which can be computed explicitly as follows:

$$L = \max_{i=1,\ldots,k} \|v_k\|.$$

Let g_{B_1} be a distance function of the set B_1. This function can be expressed as follows:

$$g_{B_1}(d) = \max\{0, \|d\|^2 - 1\}.$$

Now we can apply the hyperbolic smoothing technique to approximate the function g_{B_1}:

$$P_\tau(d) = \frac{\|d\|^2 - 1 + \sqrt{(\|d\|^2 - 1)^2 + \tau^2}}{2}.$$

Then the constrained problem (4.28) can be reformulated as a smooth unconstrained optimization problem:

$$\text{minimize} \ \ \Psi_{k\tau}(d) + \widehat{L} P_\tau(d) \quad \text{subject to} \quad d \in \mathbb{R}^n. \tag{4.30}$$

Smooth optimization techniques can be applied to solve Problem (4.30). In order to solve Problem (4.19) we take a sequence $\{\tau_k\}$ of precision parameters τ where $\tau_k \to +0$ as $k \to \infty$ and solve either Problem (4.29) or Problem (4.30). It is shown in [8] that the sequence of solutions to these problems will converge to the set of solutions of Problem (4.19).

4.4 Minimization algorithms

In this section we design two algorithms for solving Problem (4.4). The first algorithm can find the so-called (h, δ)-stationary points of Problem (4.4) for given $h > 0$ and $\delta > 0$, whereas the second one computes its Clarke stationary points. Finding (h, δ)-stationary points of Problem (4.4) for sufficiently small $h, \delta > 0$ is equivalent to the finding of its approximate stationary points satisfying (4.5).

4.5 Computation of (h, δ)-stationary points

Let $h > 0,\ \delta > 0,\ c_1 \in (0, 1),\ c_2 \in (0, c_1]$ be given numbers. An algorithm for finding (h, δ)-stationary points proceeds as follows.

Algorithm 6. Computation of (h, δ)-stationary points of Problem (4.4).

Step 1. Select any starting point $x_0 \in \mathbf{R}^n$ and set $k = 0$.

Step 2. Apply Algorithm 5 for the computation of the descent direction at $x = x_k$ for given $\delta > 0$ and $c_1 \in (0, 1)$. This algorithm finds $d_k \in S_1$ such that

$$D_k = \max_{i=1,\dots,k} \langle v_i, d_k \rangle = \min_{d \in S_1} \max_{i=1,\dots,k} \langle v_i, d \rangle.$$

Furthermore, either $D_k > -\delta$ or for the search direction $d_k \in S_1$,

$$f(x_k + h d_k) - f(x_k) \le c_1 h D_k. \tag{4.31}$$

Step 3. If $D_k > -\delta$ then stop. Otherwise go to Step 4.

Step 4. Compute $x_{k+1} = x_k + \sigma_k d_k$, where σ_k is defined as follows

$$\sigma_k = \text{argmax}\left\{\sigma \geq 0 : \ f(x_k + \sigma d_k) - f(x_k) \leq c_2 \sigma D_k\right\}.$$

Set $k = k + 1$ and go to Step 2.

In the next proposition we prove that Algorithm 6 is finitely convergent to the set of (h, δ)-stationary points of Problem (4.4).

Proposition 20. *Suppose that function f is bounded below, i.e.*

$$f_* = \inf \ \{f(x) : x \in \mathbf{R}^n\} > -\infty. \tag{4.32}$$

Then Algorithm 6 terminates after finite many iterations $m > 0$ and produces the (h, δ)-stationary point x_m where

$$m \leq m_0 \equiv \left\lfloor \frac{f(x_0) - f_*}{c_2 h \delta} \right\rfloor + 1. \tag{4.33}$$

Proof. Assume the contrary, that is the sequence $\{x_k\}$ generated by Algorithm 6 is infinite and the points x_k are not (h, δ)-stationary points for any $k = 1, 2, \ldots$. This means that

$$D_k \leq -\delta, \ \forall k = 1, 2, \ldots$$

Then descent direction d_k will be found at x_k so that the sufficient decrease condition (4.31) is satisfied:

$$f(x_k + h d_k) - f(x_k) \leq c_1 h D_k \leq c_2 h D_k.$$

It follows from the definition of σ_k that $\sigma_k \geq h$. Therefore, we have

$$\begin{aligned} f(x_{k+1}) - f(x_k) &= f(x_k + \sigma_k d_k) - f(x_k) \\ &< c_2 \sigma_k D_k \\ &\leq c_2 h D_k, \end{aligned}$$

which along with the condition $D_k \leq -\delta$ implies that

$$f(x_{k+1}) \leq f(x_0) - (k+1) c_2 h \delta.$$

Therefore, $f(x_k) \to -\infty$ as $k \to \infty$ which contradicts (4.32). Clearly, the upper bound for the number of iterations m necessary to find the (h, δ)-stationary point is m_0 given by (4.33). □

Remark 7. The use of quasisecants (that is Algorithm 5) allows us to design a very simple procedure for the estimation of the step-length $\sigma_k, k \geq 0$ in Step 4 of Algorithm 6. Since $c_2 \leq c_1$ always $\sigma_k \geq h$. In order to estimate σ_k we define a sequence $\theta_m = mh,\ m \geq 1$. Then σ_k is defined as the largest θ_m satisfying the inequality in Step 4 of Algorithm 6.

Now we design an algorithm for finding stationary points of Problem (4.4), that is points x satisfying the condition $0 \in \partial f(x)$. Let $\varepsilon > 0$ be a tolerance.

Algorithm 7. Computation of stationary points of Problem (4.4).

Step 1. Select sequences $\{h_j\}, \{\delta_j\}$ such that $h_j > 0,\ \delta_j > 0$ and $h_j \to 0,\ \delta_j \to 0$ as $j \to \infty$. Choose any starting point $x_0 \in \mathbf{R}^n$, and set $k = 0$.

Step 2. If $h_k \leq \varepsilon$ and $\delta_k \leq \varepsilon$, then stop with x_k as the final solution.

Step 3. Apply Algorithm 6 starting from the point x_k with $h = h_k$ and $\delta = \delta_k$. This algorithm finds an (h_k, δ_k)-stationary point x_{k+1} after finitely many iterations $m > 0$.

Step 4. Set $k = k + 1$ and go to Step 2.

Next we prove the convergence of Algorithm 7. For point $x_0 \in \mathbf{R}^n$, we consider the level set $\mathcal{L}(x_0) = \{x \in \mathbf{R}^n : \ f(x) \leq f(x^0)\}$.

Proposition 21. *Suppose that the objective function f in Problem (4.4) is locally Lipschitz and satisfies Assumption 1, the set $\mathcal{L}(x_0)$ is bounded for the starting point x_0 and $\varepsilon = 0$ in Algorithm 7. Then any accumulation point of the sequence $\{x_k\}$ generated by this algorithm is a stationary point of Problem (4.4).*

Proof. Since the function f is locally Lipschitz and the set $\mathcal{L}(x_0)$ is bounded, $f_* > -\infty$. Therefore, according to Proposition 20 the sequence of (h_k, δ_k)-stationary points will be generated after a finite number of iterations for all $k > 0$. Since for any $k > 0$, the point x_{k+1} is (h_k, δ_k)-stationary point, it follows from the definition of the (h_k, δ_k)-stationary points that

$$\min\{\|v\| : \ v \in W_{h_k}(x_{k+1})\} \leq \delta_k.$$

It is obvious that $x_k \in \mathcal{L}(x_0)$ for all $k > 0$. The boundedness of the set $\mathcal{L}(x_0)$ implies that the sequence $\{x_k\}$ has at least one accumulation point. Let x^* be an accumulation point and $x_{k_i} \to x^*$ as $i \to \infty$. The inequality above implies that

$$\min\{\|v\| : \ v \in W_{h_{k_i-1}}(x_{k_i})\} \leq \delta_{k_i-1}. \tag{4.34}$$

It follows from Proposition 12 that at the point x^* for any $\tau > 0$ there exists $\eta > 0$ such that

$$W_h(y) \subset \partial f(x^*) + B_\tau$$

for all $y \in B_\eta(x^*)$ and $h \in (0, \eta)$. Since the sequence $\{x_{k_i}\}$ converges to x^*, there exists i_0 such that $x_{k_i} \in B_\eta(x^*)$ for all $i > i_0$. On the other hand, since $\delta_k, h_k \to 0$ as $k \to \infty$ there exists $k_0 > 0$ such that $\delta_k < \tau$ and $h_k < \eta$ for all $k > k_0$. Then there exists i_1 such that $k_i > k_0 + 1$ for all $i > i_1$. Let

$i_2 = \max\{i_0, i_1\}$, It follows that

$$W_{h_{k_i}}(x_{k_i}) \subset \partial f(x^*) + B_\tau \tag{4.35}$$

for all $i > i_2$. (4.34) and (4.35) imply that for any $i > i_2$

$$\min\{\|v\| : \ v \in \partial f(x^*)\} \leq 2\tau.$$

Since τ is arbitrary $0 \in \partial f(x^*)$. This completes the proof. □

4.6 Numerical experiments

In this section we present results of testing and comparison of the proposed method. We call the proposed method SSM - Subgradient Smoothing Method. We compare the SSM with the following nonsmooth optimization algorithms:

1. The Proximal bundle method (PBUN) (see [61]);
2. The Variable metric bundle method (PVAR) [57];
3. The Newton-bundle method (PNEW) [56];
4. The Discrete gradient method (DGM) [5];
5. The Quasisecant method (QSM) [7].

The well-known nonsmooth optimization academic test problems were used to test algorithms. These test problems include Problems 2.1-4, 2.6, 2.10-12, 2.19-25, 3.1-2, 3.4-12, 3.15-20, 3.22-25 from [58]. We do not use all test problems from [58] because for some of them not all input data is available and for some other problems objective functions are unbounded from below. Since we use many starting points for each test problem we excluded problems with many local solutions to make easier comparison of algorithms. Accordingly, we used test problems with very few local solutions. One example of such problems is Problem 2.3 (Spiral). Its graph is given in Figure 4.1.

Parameters in the SSM were chosen as follows: $c_1 = 0.2,\ c_2 = 0.05,\ h_{j+1} = 0.5h_j,\ j \geq 1, h_1 = 1, \delta_j \equiv 10^{-7},\ \forall j \geq 0$. We implemented the SSM as well as DGM and QSM in Lahey Fortran 95 and used Fortran implementation of PBUN, PVAR and PNEW described in [59]. Numerical experiments were carried out on PC Intel(R)Core(TM)2 with CPU 1.86 GHz and 1.97GB of RAM.

We used 20 random starting points for each problem and starting points are the same for all algorithms. All the algorithms tested are so-called local algorithms and they do not intend to find always the global minimum of a nonconvex objective function. Starting from the same point the algorithms may converge to different local minimizers. We say that an algorithm solves the nonconvex nonsmooth optimization problem if it finds its local minimizer even if this local minimizer is different from the global one. An algorithm finds a solution to a problem with a tolerance $\varepsilon > 0$ if

$$|\bar{f} - f_{local}| \leq \varepsilon(1 + |f_{local}|).$$

Here $\bar{f}$ is the value of the objective function at the solution found by an algorithm and f_{local} is the closest to $\bar{f}$ among values of an objective function at its known local minimizers. We analyze the results using the performance profiles introduced in [31].

We compare the efficiency of the algorithms both in terms of number of function and subgradient evaluations. We do not compare the CPU time because for most of test problems the CPU time used by the algorithms is almost 0. We present results with $\varepsilon = 10^{-4}$. In the next two subsections we discuss results for different types of test problems.

4.6.1 Results for unconstrained minimax problems

Results for this type of problem are presented in Figures 4.2 and 4.3. We do not include the DGM in Figure 4.3 because it is the derivative-free method.

Results presented in Figure 4.2 demonstrate that the PNEW is the most efficient for this class of problems and the QSM is the most robust method in terms of function evaluations. The proposed method (SSM) is the second most robust method. It solved more than 90 % of problems. However,

it requires more function evaluations than most other algorithms used in numerical experiments. One can see from Figure 4.3 the PBUN is the most efficient in terms of subgradient evaluations and again the QSM is the most robust methods and the SSM is the second most robust method.

4.6.2 Results for general nonsmooth unconstrained problems

Results for general nonsmooth unconstrained problems are presented in Figures 4.4 and 4.5. Again we do not include the DGM in Figure 4.5.

Results presented in Figures 4.4 and 4.5 are quite similar to those presented in Figures 4.2 and 4.3 for minimax problems. Again the PNEW is the most efficient in terms of function evaluations and the PBUN is the most efficient in terms of subgradient evaluations. The QSM is the most robust method (it solved all problems for all starting points). The proposed algorithm is the second most robust (along with the PVAR). It solved 90 % of problems.

4.6.3 Results with GAMS

In this subsection we demonstrate the GAMS implementation of the proposed algorithm and compare it with the DNLP option of solvers included in GAMS. CONOPT, MINOS and SNOPT solvers were used for this purpose (for details of these solvers, see [39]). The same solvers were used to solve the subproblem for finding descent directions in the proposed method. We selected 10 test problems with different number of variables. For each test problem we used starting points from [58].

Results are given in Table 4.1, where the following notation is used:

- n - number of variables;
- f_{opt} - optimum value;
- f_v - function value obtained by an algorithm;
- n_f, n_s, n_{it} - the number of function evaluations; the number of subgradient evaluations; the number of iterations, respectively.

- N/A means that this feature is not applicable for a given solver.

Results presented in Table 4.1 demonstrate the proposed method is much more efficient than the DNLP option of solvers. The latter solvers failed to solve most of the test problems with required accuracy. Results also show that the use of various solvers for finding search directions in the proposed method may lead to different solutions (and also to a different number of function and subgradient evaluations). The use of these solvers allowed to find solutions with required accuracy, however the solver MINOS failed to find solutions in test problems MAXQ and GOFFIN.

4.7 Conclusions

In this chapter, we developed a new algorithm for solving nonsmooth optimization problems. The main difference between the proposed and other existing nonsmooth optimization methods is that in the proposed method one can use smooth optimization solvers for finding descent directions. This allows one to use powerful smooth optimization methods for solving general nonsmooth optimization problems.

We presented results of numerical experiments using well-known nonsmooth optimization test problems. The proposed algorithm was implemented both in Fortran and GAMS to compare it with other nonsmooth optimization techniques as well as nonsmooth optimization solvers in GAMS. Results demonstrate that the proposed algorithm is one of the most robust algorithms in nonsmooth optimization and it considerably outperforms the DNLP option of solvers in GAMS.

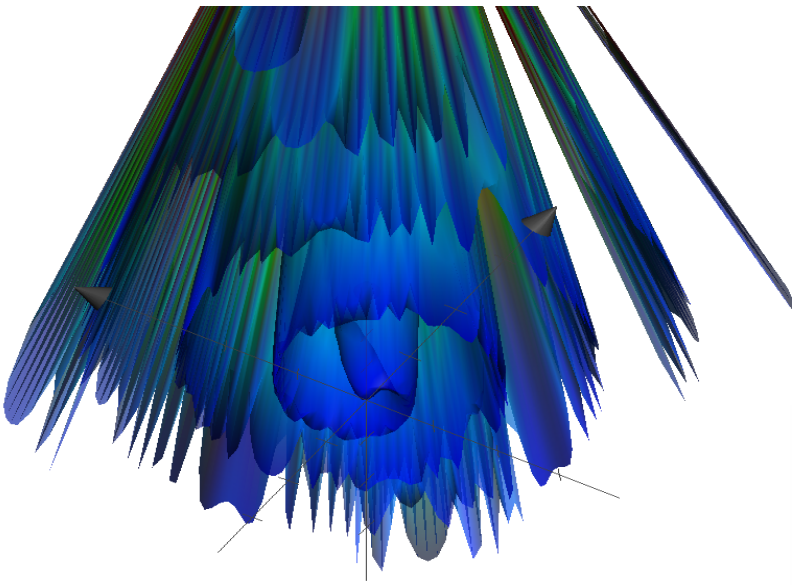


Figure 4.1: Graph of test problem 2.3 (Spiral).

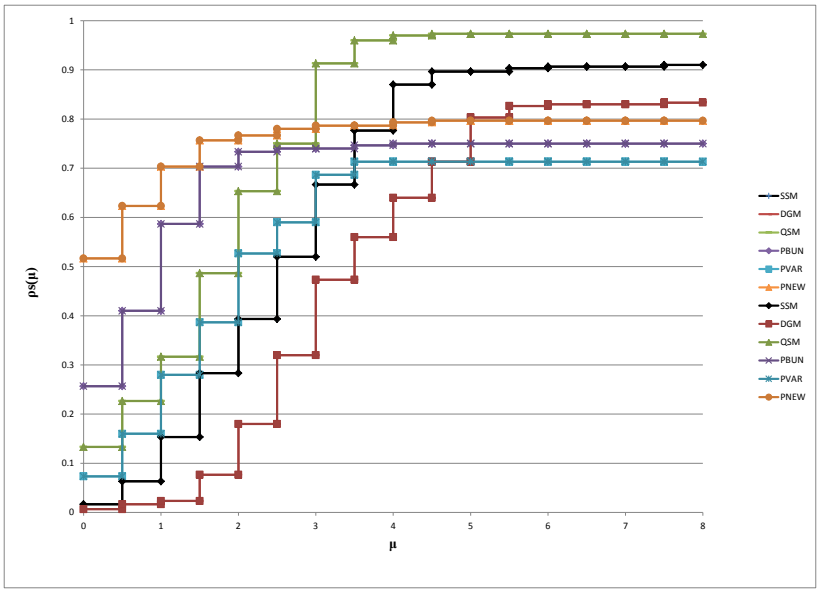


Figure 4.2: Number of function evaluations for unconstrained minimax problems.

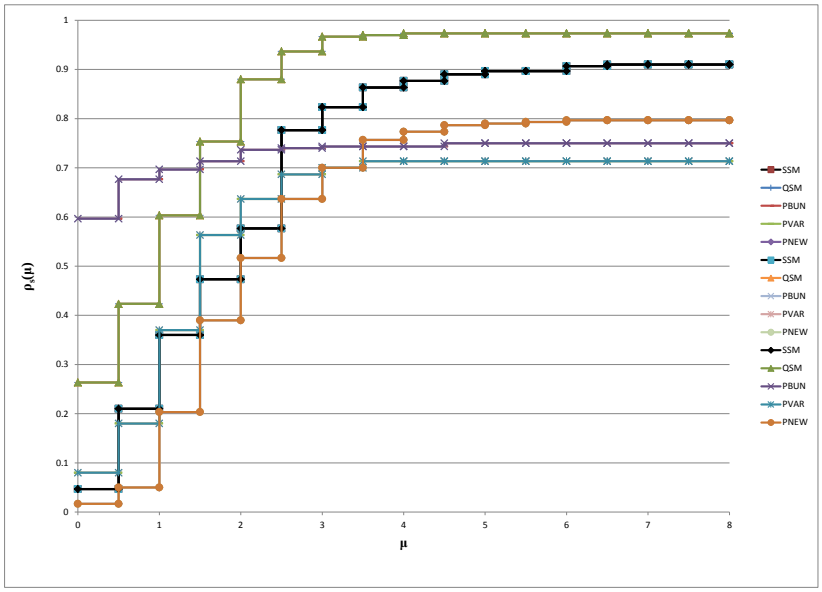


Figure 4.3: Number of subgradient evaluations for unconstrained minimax problems.

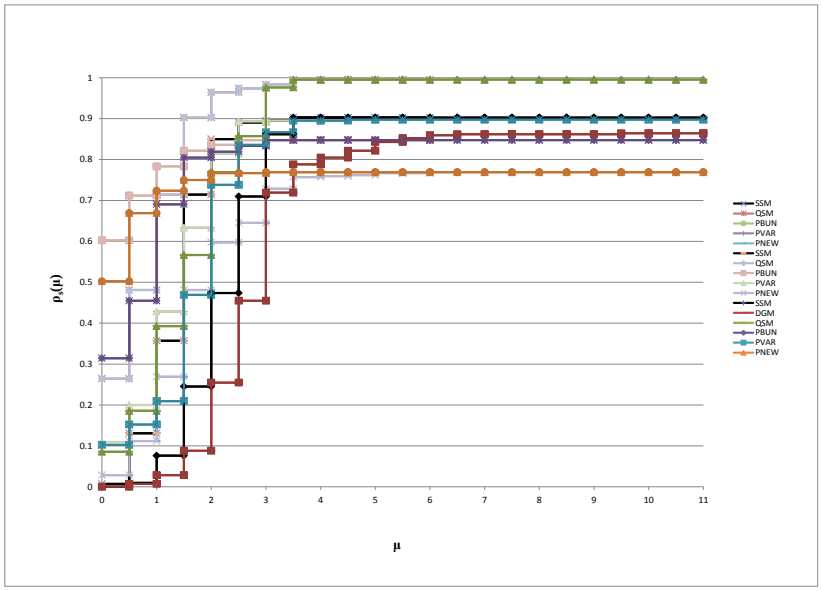


Figure 4.4: Number of function evaluations for general nonsmooth problems.

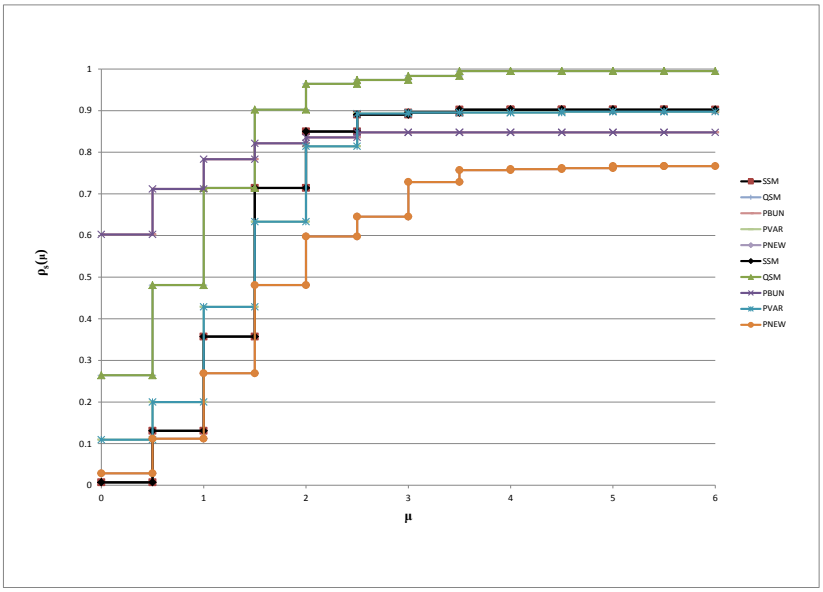


Figure 4.5: Number of subgradient evaluations for general nonsmooth problems.

Table 4.1: GAMS results

Prob.			The method			DNLP		
	n	f_{opt}	f_v	n_f	n_s	f_v	n_f	n_{it}
CONOPT								
CB2	2	1.95222	1.9522	175	47	1.9523	41	N/A
WF	2	0	0.0000	219	52	0.0000	41	N/A
SPIRAL	2	0	0.0000	85810	10873	0.0480	91	N/A
Crescent	2	0	0.0000	130	19	0.0000	16	N/A
EVD52	3	3.59972	3.6000	221	34	3.7595	32	N/A
Wong1	7	680.63006	680.7588	1074	202	700.7282	32	N/A
Polak2	10	54.59815	54.6037	297	91	54.6036	43	N/A
Polak3	11	3.70348	3.7039	301	69	3.9047	73	N/A
Maxq	20	0	0.0000	853	227	0.0000	224	N/A
Goffin	50	0	0.0001	10845	3365	0.0041	39435	N/A
MINOS								
CB2	2	1.95222	1.9522	201	57	1.9523	22	173
WF	2	0	0.0000	219	52	0.0000	7	96
SPIRAL	2	0	0.0000	9294	1306	0.1250	8	77
Crescent	2	0	0.0001	130	19	0.0000	15	125
EVD52	3	3.59972	3.6000	216	35	4.2126	14	145
Wong1	7	680.63006	680.6972	455	130	681.4484	319	1947
Polak2	10	54.59815	54.6036	511	109	60.9528	4	52
Polak3	11	3.70348	3.7037	288	52	4.0400	48	278
Maxq	20	0	62.7313	207	42	0.0000	72	375
Goffin	50	0	52.7149	4577	1307	152.4616	105	704
SNOPT								
CB2	2	1.95222	1.9522	385	111	1.9522	300	1251
WF	2	0	0.0000	219	52	0.0000	18	112
SPIRAL	2	0	0.0000	6316	851	0.1250	9	68
Crescent	2	0	0.0001	130	19	0.0004	26	178
EVD52	3	3.59972	3.6000	216	35	3.5997	194	470
Wong1	7	680.63006	680.6974	361	114	686.0448	100	417
Polak2	10	54.59815	54.6036	265	89	54.6036	38	134
Polak3	11	3.70348	3.7037	778	105	4.0006	54	184
Maxq	20	0	0.0000	2373	713	0.0000	130	611
Goffin	50	0	0.0001	12321	3859	111.4632	303	1321

Chapter 5

Minimization of pumping costs in water distribution systems

The operation of a water distribution system is a complex task which involves scheduling of pumps, regulating water levels of storages, and providing satisfactory water quality to customers at required flow and pressure. Pump scheduling is one of the most important tasks of the operation of a water distribution system as it represents the major part of its operating costs. In this chapter, a novel approach for modeling of explicit pump scheduling to minimize energy consumption by pumps is introduced, which uses a pump's start/end run times as continuous variables, and binary integer variables to describe a pump's status at the beginning of the scheduling period. This is different from other approaches where binary integer variables for each hour are typically used, which is considered very impractical from an operational perspective. The problem is formulated as a mixed integer nonlinear programming problem replaced by the sequence of smooth optimization problems and a new algorithm is developed for its solution. This algorithm is based on the combination of the grid search with the Hooke-Jeeves pattern search method. The performance of the algorithm is evaluated using literature test problems applying the hydraulic simulation model EPANet [84].

Results from this chapter were obtained in collaboration with other colleagues. More specifically, the optimization was developed in collaboration with A. Bagirov, A. Barton, H. Mala Jetmarova and N. Sultanova. Its smoothed version was developed by myself and numerical testing was carried out by S.T. Ahmed.

The structure of this chapter is as follows. The pumping cost minimization problem is formulated

in Section 5.1. An algorithm for solving the optimization problem is presented in Section 5.2. Section 5.3 provides the description of test problems as well as the results of numerical experiments using these test problems. Section 5.4 concludes the chapter.

5.1 Optimization model

In this section, we formulate both the objective function and constraints of the pumping cost minimization problem.

We consider a water distribution network which consists of one main (unlimited) source of water, storages, pump stations, pipes and demand nodes. We assume that initial storage water levels, demands and demand patterns, peak and off peak electricity tariffs are known.

The amount of energy consumed by a pump depends on flow through the pump, head supplied by the pump and wire-to-water efficiency. These parameters can be calculated using a hydraulic simulator (i.e. EPANet) for a known pump schedule. Pump energy costs also depend on the energy price given by electricity tariffs. These tariffs may vary during a scheduling period consisting of an expensive peak and cheaper off-peak periods. In this chapter, we restrict the number of on/off pump switches. However, we do not incorporate the pump maintenance costs and demand charge costs into the objective function. Therefore, our objective function consists of only energy consumed by the pumps.

Combined water volume in all storages can be different at the beginning and end of the scheduling period, which means that it is allowed both to decrease and to increase of the water level in all storages. Nevertheless, the difference between combined water volume at the beginning and end of the scheduling period cannot be greater than some predefined number.

We use the following notations for parameters in the pumping cost minimization problem which are partially adopted from [54]:

- $[T_1, T_2]$ and $[T_3, T_4]$ - off peak intervals and $[T_2, T_3]$ peak interval during one scheduling day. In real applications, for example, $T_1 = 0,\ T_2 = 7,\ T_3 = 22,\ T_4 = 24$;

- N_{SW} - the maximum number of switches for all pumps during one day. We assume that $N_{SW} \geq 2$;
- N_p - number of pumps;
- N_s - number of storages;
- N_d - number of demand nodes;
- R_1 and R_2 - energy tariffs during peak and off-peak intervals, respectively ($\$/kWh$). In real applications $R_1 > R_2$;
- E_{mi} - energy consumption rate of pump m during interval i (kWh/h), $m = 1, \ldots, N_p,\ i = 1, \ldots, N_{SW} - 1$. It can be constant or may depend on flow rate and head. It can be found from the pump's performance curve.
- H_{ji} - head supplied at demand node j during the time period i, $i = 1, \ldots, N_{SW} - 1,\ j = 1, \ldots, N_d$;
- H_j^{min} - minimum head required at the demand node $j,\ j = 1, \ldots, N_d$;
- H_j^{max} - maximum head allowed at the demand node $j,\ j = 1, \ldots, N_d$;
- Q_{mi} - flow rate through pump m during the interval i (m^3/s), $m = 1, \ldots, N_p,\ i = 1, \ldots, N_{SW} - 1$;
- h_{mi} - total dynamic head supplied by the pump m during the interval i (m), $m = 1, \ldots, N_p,\ i = 1, \ldots, N_{SW} - 1$;
- e_m - overall wire-to-water efficiency of the pump $m,\ m = 1, \ldots, N_p$;
- $H_S^{j,min}$ - minimum level of water in the storage $j,\ j = 1, \ldots, N_s$;
- $H_S^{j,max}$ - maximum level of water in the storage $j,\ j = 1, \ldots, N_s$.

We use the following notations for decision variables in the pumping cost minimization problem:

- t_{mi} - time when pump m is switching on or off, $t_{mi} \in [0, 24]$, $i = 1, \ldots, N_{SW}$. $\tau_m = (t_{m1}, \ldots, t_{m,N_{SW}})$ is a vector of start/end run times for the pump m. Let $\tau = (\tau_1, \ldots, \tau_{N_p})$. Here $t_{m1} = 0$ and $t_{m,N_{SW}} = 24$;

- X_m - shows whether pump m is operating during the initial interval $[t_{m1}, t_{m2}]$, $m = 1, \ldots, N_p$. Let $X = (X_1, \ldots, X_{N_p})$. If $X_m = 1$ then the pump m is operating at the initial interval $[t_{m1}, t_{m2}]$ and it is not operating if $X_m = 0$.

Remark 8. We use only one binary variable for each pump, which is sufficient to describe the schedule for one pump for the whole scheduling period (i.e. one day). Indeed, if X_m is known for the first interval $[t_{m1}, t_{m2}]$, then it is $1 - X_m$ for the second interval $[t_{m2}, t_{m3}]$, X_m for the third interval $[t_{m3}, t_{m4}]$ and so on. This significantly reduces the number of binary variables in the pumping cost minimization problem. Moreover, the use of start/end run times as decision variables allows a reduction in the number of continuous variables.

5.1.1 The objective function

Given particular start/end run times $t_{mi},\ i = 1, \ldots, N_{SW}$ at the interval i and initial schedule X_m for pump $m,\ m = 1, \ldots, N_p$, the total cost of energy, in general, is calculated as follows

$$f(X, \tau) = \sum_{m=1}^{N_p} (energy\ consumption\ cost + demand\ charge + pump\ maintenance\ cost).$$

This chapter does not consider the demand charge and pump maintenance cost, and formulates the objective function on a daily (24 hours) basis. Additionally, only two (peak and off-peak) tariff periods are considered.

As already described, information on pump status at the initial time interval is sufficient to determine its status in all subsequent time intervals. Then any pump's on/off intervals can be divided into two parts: the first part contains all intervals where the status of the pump is the same as its status in the initial interval, and the second part contains all other intervals. In order to take into account peak and off peak periods, the cost function for the off peak period (low cost) is written for the whole day

and added to the difference between the peak (high cost) and off peak periods.

The cost function for a given pump m can be expressed as a sum of four functions as follows:

$$F(X_m, \tau_m) = F_1(X_m, \tau_m) + \bar{F}_1(X_m, \tau_m) + F_2(X_m, \tau_m) + \bar{F}_2(X_m, \tau_m), \tag{5.1}$$

$$F_1(X_m, \tau_m) = R_1 X_m \sum_{i=1}^{\bar{N}} E_{m,2i-1}(t_{m,2i} - t_{m,2i-1}), \tag{5.2}$$

$$\bar{F}_1(X_m, \tau_m) = R_1(1 - X_m) \sum_{i=1}^{N_{SW}-\bar{N}-1} E_{m,2i}(t_{m,2i+1} - t_{m,2i}), \tag{5.3}$$

$$F_2(X_m, \tau_m) = (R_2 - R_1) X_m \sum_{i=1}^{\bar{N}} E_{m,2i-1} \max\{0, w_{m,2i} - u_{m,2i-1}\}, \tag{5.4}$$

$$\bar{F}_2(X_m, \tau_m) = (R_2 - R_1)(1 - X_m) \sum_{i=1}^{N_{SW}-\bar{N}-1} E_{m,2i} \max\{0, w_{m,2i+1} - u_{m,2i}\}. \tag{5.5}$$

Here

- E_{mi} is the energy consumption rate at the interval $[t_{mi}, t_{m,i+1}],\ i = 1, \ldots, N_{SW} - 1$ of a pump m which depends on the flow through the pump, head supplied by the pump and efficiency at which it operates:
$$E_{mi} = \frac{K_m Q_{mi} h_{mi}}{e_m},$$
where K_m is a given constant for pump m. For example, $K_m = 0.01019$ [54];
- $\bar{N} = \lfloor N_{SW}/2 \rfloor$ where $\lfloor a \rfloor$ stands for the largest integer number less than or equal to a;
- $u_{mi} = \max\{T_2, t_{mi}\},\ \ w_{mi} = \min\{T_3, t_{mi}\},\ i = 1, \ldots, N_{SW}$ are introduced to find intersections between intervals $[t_{mi}, t_{m,i+1}]$ and the peak period $[T_2, T_3]$.

The function $F_1(X_m, \tau_m)$ defined by (5.2) represents the energy cost for all time intervals where the status of the pump m is the same as its status in the initial interval $[t_{m1}, t_{m2}]$. This function is

0 if this pump is off at the interval $[t_{m1}, t_{m2}]$. The function $\bar{F}_1(X_m, \tau_m)$ defined by (5.3) represents the energy cost for all other intervals. This function is 0 if the pump is on at the interval $[t_{m1}, t_{m2}]$. Both functions are calculated for off peak rate for the whole day. In order to take into account the peak period, we introduce two additional functions $F_2(X_m, \tau_m)$ and $\bar{F}_2(X_m, \tau_m)$, which add the difference between peak and off peak rates in the peak period. The function $F_2(X_m, \tau_m)$ defined by (5.4) represents the cost difference between peak and off peak rates for all intervals where the status of the pump m is the same as its status in the initial interval $[t_{m1}, t_{m2}]$. The function $\bar{F}_2(X_m, \tau_m)$ defined by (5.5) represents the cost difference between peak and off peak rates for all other intervals.

Functions (5.4) and (5.5) are approximated using the hyperbolic smoothing technique. Then we have the following approximations of these functions:

$$F_{2\mu}(X_m, \tau_m) = (R_2 - R_1)X_m$$
$$\sum_{i=1}^{\bar{N}} E_{m,2i-1} \frac{(w_{m,2i} - u_{m,2i-1}) + \sqrt{(w_{m,2i} - u_{m,2i-1})^2 + \mu^2}}{2},$$

$$\bar{F}_{2\mu}(X_m, \tau_m) = (R_2 - R_1)(1 - X_m)$$
$$\sum_{i=1}^{N_{SW}-\bar{N}-1} E_{m,2i} \frac{(w_{m,2i+1} - u_{m,2i}) + \sqrt{(w_{m,2i+1} - u_{m,2i})^2 + \mu^2}}{2}.$$

Then the function $F(X_m, \tau_m)$ can be approximated by the following smooth function:

$$F_\mu(X_m, \tau_m) = F_1(X_m, \tau_m) + \bar{F}_1(X_m, \tau_m) + F_{2\mu}(X_m, \tau_m) + \bar{F}_{2\mu}(X_m, \tau_m), \tag{5.6}$$

Figure 5.1 shows an example of the timeline for one scheduling period (i.e. one day) with five intervals on which pumps switch on and off alternatingly. Assuming that the dashed lines represent intervals during which the pumps are off ($X_m = 0$), and the solid lines represent intervals during which the pumps are on, we will have functions $F_1(X_m, t) = 0$ and $F_2(X_m, t) = 0$, $\bar{F}_1(X_m, t) = R_1[E_{m,2}(t_3 - t_2) + E_{m,4}(t_5 - t_4)]$ and $\bar{F}_2(X_m, t) = (R_2 - R_1)E_{m,2}(t_3 - T_2) + (R_2 - R_1)E_{m,4}(T_3 - t_4)$.

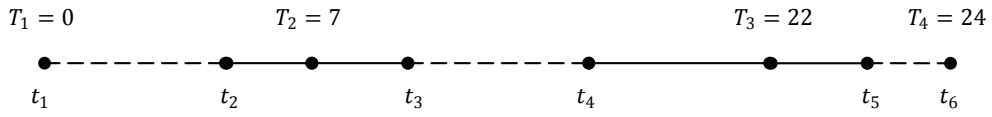


Figure 5.1: An example of a timeline.

Remark 9. It should be noted that off peak may start at midnight in some water distribution networks. This case is even simpler, because there are only two intervals: $[T_1, T_2]$ and $[T_2, T_3]$ where $T_1 = 0$ and $T_3 = 24$. However, in this Chapter we consider a more complex case when the off peak period starts and ends during the day.

The objective function, which is the total pumping cost for the whole water distribution system with N_p pumps, can be written as follows:

$$f_\mu(X, \tau) = \sum_{m=1}^{N_p} F_\mu(X_m, \tau_m). \tag{5.7}$$

Since the functions $F_1, \bar{F}_1$ are nonconvex and functions $F_{2\mu}, \bar{F}_{2\mu}$ are nonconvex, the objective function f is nonconvex.

Remark 10. We consider only pumping cost as the objective function, so it does not include the maintenance costs. Maintenance costs can be measured by using the number of pump switches. Frequent switching of pumps cause wear and tear which, in turn, can increase maintenance costs. As a result, the minimization of the number of pump switches can reduce maintenance costs. However, our objective function allows us to automatically take into account this constraint, since we define the maximum number $N_{SW} - 1$ of time intervals a priori. Therefore, there is no need to additionally define those constraints.

5.1.2 Constraints

In this subsection, we formulate the constraints of the pumping cost minimization problem. These constraints include hydraulic constraints representing conservation of mass of flow and energy, and

system constraints such as minimum and maximum limits on storage levels, and minimum and maximum pressure requirements at demand nodes.

We do not require that storage water levels should be fully recovered by the end of the scheduling period. This means that it is allowed to have some deficit of water in all storages. Thus constraints on storages are determined by their minimum and maximum water levels as well as the combined water volumes in all storages.

Denote by H_S^{jI}, the initial water level, and H_S^{jE}, the final water level in storage $j,\ j = 1, \ldots, N_s$. Then we require that

$$H_S^{j,min} \leq H_S^{jI} \leq H_S^{j,max},$$

$$H_S^{j,min} \leq H_S^{jE} \leq H_S^{j,max},\ \ j = 1, \ldots, N_s.$$

Denote also by D, the ratio of the allowed difference in the combined water volume in all storages (in $\%$). Then we have the following constraints on the volume of water at the end of the scheduling period:

$$-D \leq \frac{V_I - V_E}{V_I} \times 100 \leq D,$$

where

$$V_I = \sum_{j=1}^{N_s} V_{jI},\ \ V_E = \sum_{j=1}^{N_s} V_{jE}.$$

V_{jI} and V_{jE} are volumes of water in storages $j = 1, \ldots, N_s$ at the beginning and end of the scheduling period, respectively.

It is required that consumers are supplied water at adequate pressures. Therefore, the optimization model must include maximum and minimum pressure constraints at customer demand nodes

$$H_j^{min} \leq H_{ji} \leq H_j^{max},\ j = 1, \ldots, N_d.$$

Additionally we require that start/end run times satisfy the following condition:

$$0 = t_{m1} < t_{m2} \leq \ldots \leq t_{m,M-1} \leq t_{mM} = 24,\ M = N_{SW}.$$

We do not explicitly describe constraints representing conservation of mass of flow and energy, because these are maintained by a hydraulic simulator EPANet.

5.1.3 Formulation of optimization problem

The pumping cost minimization problem is reduced to the following optimization problem:

$$\text{minimize} \;\; f_\mu(X, \tau) \tag{5.8}$$

subject to

$$X_m \in \{0, 1\}, \; m = 1, \ldots, N_p, \tag{5.9}$$

$$H_S^{j,min} \leq H_S^{jI} \leq H_S^{j,max}, H_S^{j,min} \leq H_S^{jE} \leq H_S^{j,max}, \;\; j = 1, \ldots, N_s, \tag{5.10}$$

$$-D \leq \frac{V_I - V_E}{V_I} \times 100 \leq D, \tag{5.11}$$

$$H_j^{min} \leq H_{ji} \leq H_j^{max}, \; j = 1, \ldots, N_d, \tag{5.12}$$

$$0 = t_{m1} < t_{m2} \leq \ldots \leq t_{m,N_{SW}-1} \leq t_{m,N_{SW}} = 24. \tag{5.13}$$

The Problem (5.8)-(5.13) contains both continuous (t_{mi}) and integer variables (X_m). The number of continuous variables is $N_p(N_{SW} - 2)$ and the number of integer variables is N_p (which is the number of pumps).

Constraint (5.10) will be maintained by a hydraulic simulator EPANet, so it will not be included in the final objective function. In order to take into account the constraint (5.13), we will use the

following penalty function:

$$P(\tau) = \sum_{k=1}^{m} \sum_{i=1}^{N_{SW}-1} \max\{0, t_{mi} - t_{m,i+1}\}.$$

(5.11) and (5.12) are not maintained by EPANET.

Applying the hyperbolic smoothing to the function $P(\tau)$ we approximate it by the following smooth function:

$$P_\mu(\tau) = \sum_{k=1}^{m} \sum_{i=1}^{N_{SW}-1} \frac{(t_{mi} - t_{m,i+1}) + \sqrt{(t_{mi} - t_{m,i+1})^2 + \mu^2}}{2}.$$

Then Problem (5.8)-(5.13) can be rewritten as follows:

$$\text{minimize } G_\mu(X, \tau) \equiv f_\mu(X, \tau) + \gamma P_\mu(\tau) \tag{5.14}$$

subject to

$$X_m \in \{0, 1\},\ m = 1, \ldots, N_p, \tag{5.15}$$

$$H_S^{j,min} \le H_S^{jI} \le H_S^{j,max}, H_S^{j,min} \le H_S^{jE} \le H_S^{j,max},\ \ j = 1, \ldots, N_s, \tag{5.16}$$

$$-D \le \frac{V_I - V_E}{V_I} \times 100 \le D, \tag{5.17}$$

$$H_j^{min} \le H_{ji} \le H_j^{max},\ j = 1, \ldots, N_d. \tag{5.18}$$

Here $\gamma > 0$ is a penalty coefficient.

This problem is a smooth optimization and can be solved using any smooth optimization algorithm. We use the Hooke-Jeeves algorithm for its solution.

5.2 Solution algorithm and its implementation

In this section, we describe both an algorithm for solving the pumping cost minimization problem (5.14)-(5.18) and its implementation. This algorithm is based on the combination of the optimization and hydraulic simulation. The optimization method is applied to generate pump schedules using only the objective function (5.14) and constraints (5.15), whereas a hydraulic simulator is used to check the hydraulic feasibility of those schedules. Thus hydraulic constraints and limits on storage levels are enforced implicitly by the hydraulic simulator. The constraint (5.18), minimum and maximum pressure heads at demand nodes, was not included in the optimization problem at this stage. Nevertheless, the minimum pressure head was controlled by the simulator. We apply the Hooke-Jeeves method for minimization of the objective function (5.14) and the package EPANet for hydraulic simulation (for details of EPANet system, see [84]).

The Hooke-Jeeves method is a direct search method which does not require gradient information. It is based on two types of moves: exploratory and pattern. Exploratory moves are moves along the coordinate directions. The length of these moves is determined by step sizes which should be initialized by the user. Pattern moves are determined by the first and last points obtained by the exploratory moves. The direction from the first point to the last point is considered as the most favorable search direction. The step length in this phase depends on the distance between these two points, in fact, on the step sizes from the exploratory moves. When no further improvements are made through exploration moves around the base point, the step size can be reduced and the process repeated. If the step size in exploratory moves is less than some predefined threshold, then the method stops and the last base point is accepted as the approximate solution. Details of the Hooke-Jeeves method can be found in [87]. Direct search methods and the Hooke-Jeeves method, in particular, are very suitable for solving optimization problems, where the part of constraints are maintained by simulation.

In order to implement the Hooke-Jeeves method, one should provide the starting point, the initial and final values of the step size in exploratory phase. In the pumping cost minimization problem,

the set of starting schedules is defined using the grid on the start/end run times and combinations of on/offs of pumps. Therefore, the initial value of the step size depends on the size of grids. In our implementation, the initial step size in the exploratory phase is 2 hours. The final value of the step size depends on EPANet's hydraulic time step. In our implementation, the final step size in the exploratory phase of the Hooke-Jeeves method is 10 minutes. For small water networks, values of the binary variable X can be obtained by considering all possible combinations of on/offs. However in large water networks, this variable is also part of optimization.

Figure 5.2 demonstrates a flowchart of the algorithm.

Some explanation on the algorithm follows. In the first step, the initial and smallest values of the step size is set in the Hooke-Jeeves method. In the second step of the algorithm, the grid on start/end run times of pumps and combinations of their on/offs is applied to generate a set of schedules. Such an approach allows the generation of starting points from different parts of the search space. In calculations, we choose the step in the grid search for start/end run times to be four hours. In the third step of the algorithm, the simulation package EPANet is applied to select hydraulically feasible schedules. All these schedules are starting points for the Hook-Jeeves method, which is applied starting from each of these points to find a hydraulically feasible schedule with lower objective function value. Here, the algorithm iterates between the optimization and simulation to check hydraulic feasibility of schedules generated by the Hooke-Jeeves method. Additionally, the EPANet returns dynamic pressure heads and flow rate for each pump which are used to compute the objective function (5.14).

Search directions in the exploratory phase of the Hooke-Jeeves method are determined by the decrease or increase of the combined water volume in all storages. If there is a decrease in this volume, then we allow the Hooke-Jeeves method to only increase the pump(s) operation time. Otherwise, we allow this method to decrease the pump(s) operation time. The smallest time step in the Hooke-Jeeves method is 10 minutes. As a result, the Hooke-Jeeves method finds a set of local solutions to the pumping cost minimization problem. In the last step of the algorithm, we choose the solution among all local minimizers with the lowest objective function value and accept it as the estimate to the global minimizer of the pumping cost minimization problem (5.14)-(5.18).

In our numerical experiments, we use the following values of parameters:

- $T_1 = 0,\ T_2 = 7,\ T_3 = 22,\ T_4 = 24$;
- $N_{SW} = 5$;
- $D = 10\%$;
- $\gamma = 100$.

The proposed algorithm was implemented in programming language C.

5.3 Test problem and numerical results

To verify the efficiency of the proposed algorithm, it has been applied to solve one test problem generated using the software package EPANet and one test problem from the literature. In this section, a description of those test problems and the results obtained applying the proposed algorithm are presented.

The simulation starts at 00:00 in the morning and ends at 23:59 at night. For hydraulic purposes, 10 minute hydraulic time step intervals are used to achieve a reasonable precision.

5.3.1 Example

EPANet Net3 example network is chosen as the test problem for testing the algorithm. The system has two water sources, three elevated water tanks, 120 pipes, 94 nodes and 2 pumping stations. The water distribution system for this test problem is given in Figure 5.3 where the following modifications were made:

1. Status of Pump 10 was kept open instead of closed from the start of the simulation.
2. All control statements for Pumps 10 and 335 were removed. This means that Pump 10 no longer supplies water at a given fixed time of the day. On the other hand, Pump 335 is not controlled by water level of Tank 1 and Link 330 remains closed the whole day.

We use the following values of parameters in the simulation:

- Peak tariff $R_1 = 0.1194\ \$/kWh$;
- Off-peak tariff $R_2 = 0.0244\ \$/kWh$;
- Wire-to-water efficiency of both pumps: $e_1 = e_2 = 0.75$.

Results for this example are presented in Figures 5.3-5.7. The lowest value found for the energy consumption cost (that is the value of the objective function (5.8)) is $\$347.66$. The schedule representing this value is given in Figure 5.4. This schedule corresponds to 4.10% deficit in the total volume of the water in the network.

The graph in Figure 5.5 shows water inflow and outflow into and from the network, respectively, as well as the combined water level in all tanks during the scheduling day.

Graphs in Figure 5.6 demonstrate how water volume in tanks changes during the scheduling period (one day) for the optimal schedule.

Graphs in Figure 5.7 show the water flow from the pumps during one day corresponding to the optimal schedule.

5.4 Conclusions

In this chapter, a novel algorithm was developed for solving pumping cost minimization problems. We proposed a new approach for modeling of explicit pump scheduling by considering pump start/end run times and pump status at the beginning of the scheduling period as decision variables. Such an approach allows for the significant reduction of the number of binary variables in the pumping cost minimization problem. This approach also allows for the easy generation of a set of hydraulically feasible solutions using a hydraulic simulator to cover the whole search space.

An algorithm was developed which involves both optimization and simulation to find the optimal pumping schedule. The Hooke-Jeeves direct search method is applied for optimization starting from feasible points generated by the grid search. The EPANet package was used to perform hydraulic simulations. The performance of the algorithm was evaluated using test problem from the EPANet.

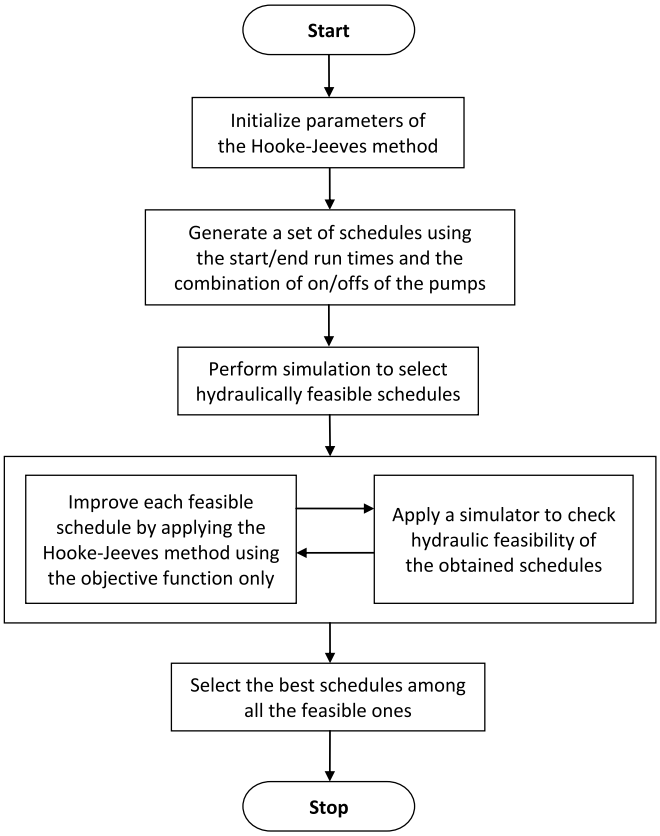


Figure 5.2: The algorithm for pumping cost minimization.

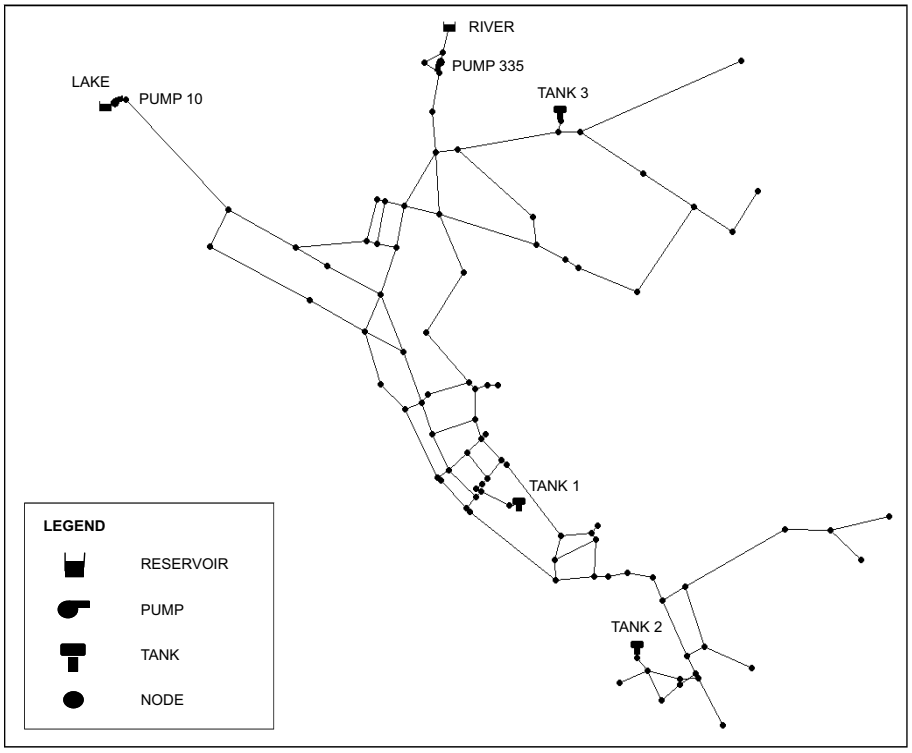


Figure 5.3: The water distribution system.

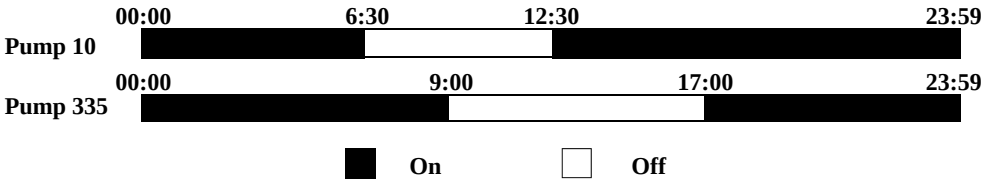


Figure 5.4: The optimal pump schedule.

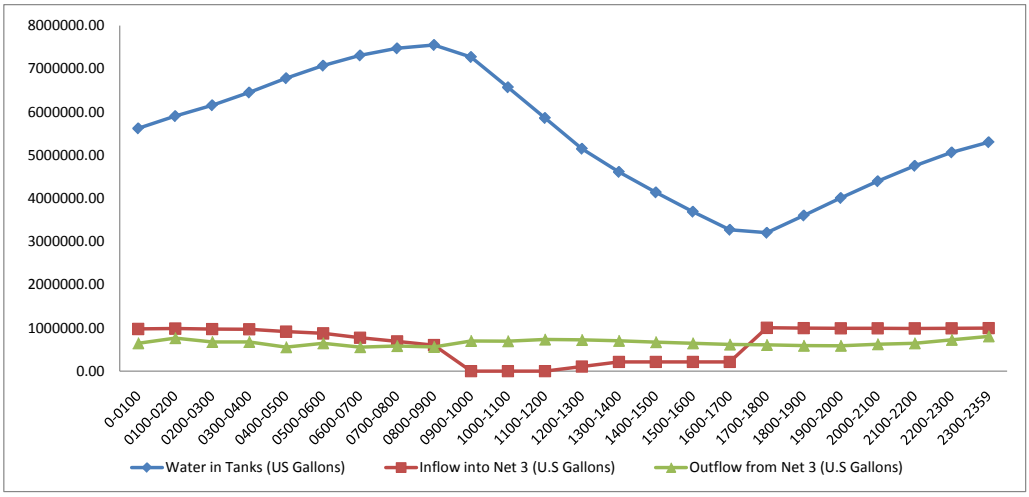


Figure 5.5: Inflow and outflow from the network.

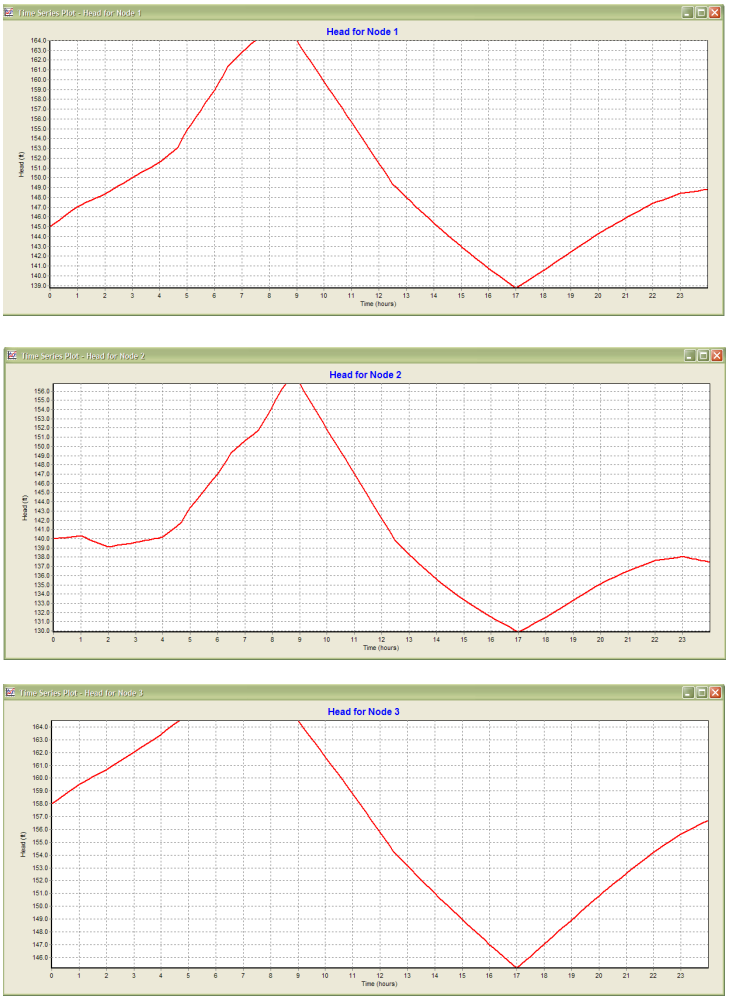


Figure 5.6: Time series water volume graphs for Tanks 1, 2 and 3.

Figure 5.7: Time series water flow graphs for Pumps 10 and 335.

Chapter 6

Conclusion and recommendations for future research

This thesis deals with the problem of solving nonsmooth and in particular, nonconvex optimization problems. More specifically, our aim is to develop new algorithms for solving minimax and more general nonsmooth nonconvex optimization problems. In nonsmooth optimization there are efficient algorithms for solving convex problems however the development of numerical methods for solving nonconvex problems still remains a crucial research topic.

There are several reasons why investigating into nonsmooth nonconvex optimization is important. First, smooth optimization algorithms cannot be applied to solve such problems due to the absence of any convergence results. Second, algorithms for convex nonsmooth optimization are heavily based on the convex models and they are not always efficient for nonconvex problems. Third, the convergence of nonsmooth optimization algorithms which are not based on convex models, such as the subgradient method, can be proved only under convexity assumption.

In this thesis, we have developed two new algorithms for solving nonsmooth nonconvex problems. The first algorithm is developed for solving finite minimax problems, whereas the second algorithm solves general nonsmooth (nonconvex) optimization problems. Both algorithms are based on the use of the hyperbolic smoothing technique. In addition, we studied their convergence and present results of numerical experiments with well-known nonsmooth optimization test problems. The comparison of the proposed methods with other methods is demonstrated using results of numerical experiments.

Our contribution

As previously stated, the design of algorithms for solving nonsmooth optimization problems which are applicable for nonconvex problems is an important task. In order to do that, we designed two methods which are based on the hyperbolic smoothing technique. First, we developed and studied in detail the hyperbolic smoothing method for solving the finite minimax problems. We proved its convergence. The proposed algorithm is easy to implement. It contains simple procedures for finding descent directions and step lengths. We presented results of numerical experiments using well-known nonsmooth optimization test problems including nonsmooth, nonconvex problems. In the second method we designed new nonsmooth optimization algorithms for solving general nonsmooth nonconvex problems by extending the hyperbolic smoothing technique, and using numerical experiments, we demonstrated its performance for nonsmooth nonconvex problems. Finally, we applied them for solving pumping cost minimization problems, indicating their capability in dealing with real world problems and its ability of finding an optimal pump schedule. These two proposed algorithms are explained shortly as follows:

1. Hyperbolic smoothing technique and minimax problems

Despite of some applications, the hyperbolic smoothing technique has not been studied in detail until now. In this thesis we study this smoothing technique in more detail. In order to apply the hyperbolic smoothing to the finite maximum functions they are represented as a sum of the maximum of two functions. We study the relationship between the set of stationary points of the smooth function and that of the original function. The new function is approximated using hyperbolic smoothing functions and differential properties of the approximating function are studied. In short, it is shown that smooth optimization solvers can be used to minimize the approximating function. This algorithm is as simple to implement, and at the same time it is numerically efficient. On the other hand, we implement the algorithm in GAMS and compare it with GAMS solvers using results of numerical experiments. We also compared

the algorithm based on the hyperbolic smoothing functions with the algorithm based on the exponential smoothing function and also with the algorithm based on the NLP reformulation. The results demonstrated that algorithm based on smoothing techniques is robust and most of the time can find feasible solutions, which is not the case for the algorithms based on the NLP reformulation.

2. Solving nonsmooth nonsmooth nonconvex optimization problems

We developed a new algorithm for solving general nonsmooth nonconvex optimization problems. The proposed method allows one to use smooth optimization solvers for finding descent directions. This makes it possible to use powerful smooth optimization algorithms for solving general nonsmooth optimization problems. We presented results of numerical experiments using nonsmooth optimization test problems. The proposed algorithm was implemented in both Fortran and GAMS to compare it with other nonsmooth optimization techniques along with nonsmooth optimization solvers in GAMS. Results show that the new algorithm is robust and it significantly outperforms the DNLP solver in GAMS.

3. Minimization of pumping costs in water distribution systems

We proposed a new method for modeling of pump scheduling to minimize energy usage. This approach is different from others where binary integer variables for each hour are usually used, which is regarded as very unrealistic. The problem is formulated as a mixed integer nonlinear programming problem and a new algorithm is developed for its solution. This algorithm relies on the combination of the grid search with the Hooke-Jeeves pattern search method and hyperbolic smoothing technique. The performance of the algorithm is evaluated using the hydraulic simulation model EPANet.

Future work

1. Constrained nonsmooth optimization

Our focus in this thesis was on unconstrained optimization problems. Though, we might con-

vert the constrained optimization problems in to unconstrained problems, this needs setting some parameters which can make the problem difficult. In this connection, it might be useful to do additional research to extend the proposed methods into constrained problems.

2. Large scale nonsmooth optimization problems

We successfully tested the algorithms on some large scale nonsmooth problems. However algorithms based on smoothing techniques for solving large scale problems have never been studied extensively. It is worthwhile to study such algorithms.

Bibliography

[1] Anstreicher, K.M., and Wolsey, L.A., Two well-known properties of subgradient optimization, *Mathematical Programming*, 120(1), 2009, 213–220. (document), 2.1

[2] Applegate, D., Cook, W., Dash, S., and Rohe, A., Solution of a minmax vehicle routing problem, *INFORMS Journal on Computing*, 14(2), 2002, 132–143. 2.4

[3] Arkin, E.M., Hassin, R., and Levin, A., Approximations for minimum and minmax vehicle routing problems, *J. Algorithms,* 59(1), 2006, 1–18. 2.4

[4] Bagirov, A.M., and Ugon, J., Supervised data classification via max-min separability, *Continuous Optimization*, 35(4), 2005, 175–207. (document)

[5] Bagirov, A.M., Karasozen, B., and Sezer, M., Discrete gradient method: Derivative-free method for nonsmooth optimization, *Journal of Optimization Theory and Applications*, 137(2), 2008, 317–334. (document), 2.3.1, 2.4, 4.1, 4.6

[6] Bagirov, A., and Ganjehlou, A.N., An approximate subgradient algorithm for unconstrained nonsmooth, nonconvex optimization, *Mathematical Methods of Operations Research*, 67(2), 2008, 187–206. 4.1

[7] Bagirov, A.M., and Ganjehlou, A.N., A quasisecant method for minimizing nonsmooth functions, *Optimization Methods and Software*, 25(1), 2010, 3–18. (document), 2.3.1, 2.3.4, 2.4, 4.1, 4.2, 4.6

[8] Bagirov, A.M., Al Nuiamat, A., and Sultanova, N., Hyperbolic smoothing function method for minimax problems, *Optimization*, 62(6), 2013, 759–782. 4.3, 4.3, 4.3, 4.3

[9] Bagirov, A. M., Jin, L., Karmitsa, N., Al Nuaimat, A., and Sultanova, N., Subgradient Method for Nonconvex Nonsmooth Optimization, *Journal of Optimization Theory and Applications*, 157(2), 2013, 416–435. (document)

[10] Banichuk, N.V., Minimax approach to structural optimization problems. *Journal of Optimization Theory and Applications*, 20(1), 1976, 111–127. 2.4

[11] Baumol, W. J., *Economic theory and operations analysis*, 1977. (document)

[12] Bazaraa, M.S, and Sherali, H.D., On the Choice of Step Size in Subgradient Optimization, *European Journal of Operational Research*, 7(4), 1981, 380–388. 2.1

[13] Beltran, C., and Heredia, F.J., An effective line search for the subgradient method, *Journal of optimization theory and applications*, 125(1), 2005, 1–18. (document), 2.1

[14] Ben-Israel, A., A Newton-Raphson method for the solution of systems of equations, *Journal of Mathematical Analysisand Application*, 15(2), 1966, 243–253.

[15] Ben-Tal, A., and Teboulle, M., A smoothing technique for nondifferentiable optimization problems, *Lecture Notes in Mathematics*, Springer Berlin Heidelberg, 1989, 1–11. (document), 2.4, 2.4.1

[16] Berman, O., Drezner, Z., Wang, R.M., and Wesolowsky, G.O., The minimax and maximin location problems on a network with uniform distributed weights, *IIE Trans*, 35(11), 2003, 1017–1025. 2.4

[17] Bertsekas, D.P., *Nonlinear Programming, Athena Scientific*, New York, 1999. (document), 2.1

[18] Biscos, C., Mulholland, M., Le Lann, M.V., Buckley, C.A., and Brouckaert, C.J., Optimal operation of water distribution networks by predictive control using MINLP, *Water Sa*, 29(4), 2004, 393–404. 2.5

[19] Bugeda, G., JA-Dyesidyeri, J.P., and Schoenauer, M., Optimal pump scheduling for water supply using genetic algorithms, *Proceedings of the International Congress on Evolutionary*

Methods for Design, Optimization and Control with Applications to Industrial Problems: EUROGEN 2003, Barcelona, 2003. 2.5

[20] Byrd, R.H., Nocedal, J., and Schnabel, R.B., Representations of quasi-Newton matrices and their use in limited memory methods, *Mathematical Programming,* 63(1-3), 1994, 129–156. 2.3.3

[21] Chow, E., and Willsky, A., Analytical redundancy and the design of robust failure detection systems. Automatic Control, IEEE Transactions on, 29(7), 1984, 603–614. (document)

[22] Chen, X., A verification method for solutions of nonsmooth equations, *Computing*, 58, 1997, 281–294.

[23] Cheney, E.W., and Goldstein, A.A., Newton's method for convex programming and Tcheby cheff approximation, *J.Numerische Mathematik*, 1(1), 1959, 253–268. 2.2

[24] Cherkaev, E., and Cherkaev, A., Minimax optimization problem of structural design, *Comput. Struct*, 86, 2008, 1426–1435 2.4

[25] Clarke, F. H., *Optimization and Nonsmooth Analysis*, New York, John Wiley, 5, 1983. 1, 1.2, 6, 5

[26] COHEN, D., SHAMIR, U., and SINAI, G., Optimal operation of multi-quality water supply systems-II: The Q-H model, *Engineering Optimization*, 32(6), 2000, 687–720. 2.5

[27] Dandy, G.C., Simpson, A.R., and Murphy, L.J., An improved genetic algorithm for pipe network optimization, *Water Resour. Res.,* 32(2), 1996, 449–458. 2.5

[28] Demyanov, V.F., and Malozemov, V.N., *Introduction to minimax,* Wiley, New York, 1974. (document), 2.4

[29] Demyanov, V.F., and Rubinov, A.M., *Constructive Nonsmooth Analysis*. Peter Lang, Frankfurt am Main, 1995. 4.1

[30] Dennis, J.E. and Schnabel, R.B., Numerical methods for unconstrained optimization and nonlinear equations, *Society for Industrial Mathematics*, 1996.

[31] Dolan, E.D., and More, J.J., Benchmarking optimization software with performance profiles, *Mathematical Programming*, 91, 2002, 201–213. 3.4, 4.6

[32] Drezner, Z., Wesolowsky, G.O., Single facility lp-distance minimax location, *SIAM J. Alg. Disc*. Meth, 1, 1980, 315–321. 2.4

[33] Du, D.Z., and Pardalos, P.M. (eds), *Minimax and Applications,* Kluwer Academic Publishers, Dordrecht, 1995. (document), 2.4

[34] Feng, Y., Hongwei, L., Shuisheng, Z., and Sanyang, L., A smoothing trust-region Newton-CG method for minimax problem, *Applied Mathematics and Computation*, 199, 2008, 581–589. 2.4, 2

[35] Fuduli, A., Gaudioso, M., and Giallombardo, G., A DC piecewise affine model and a bundling technique in nonconvex nonsmooth minimization, *Optimization Methods and Software*, 19(1), 2004, 89–102. 2.3.1

[36] Fuduli, A., Gaudioso, M., and Giallombardo, G., Minimizing nonconvex nonsmooth functions via cutting planes and proximity control, *SIAM J. on Optimization*, 14(3), 2004, 743–756. 2.3.1

[37] Fukushima, M., and Qi, L., A globally and superlinearly convergent algorithm for nonsmooth convex minimization, *SIAM Journal on Optimization*, 6(4), 1996, 1106–1120. (document)

[38] Fumero, F., A Modified Subgradient Algorithm for Lagrangean Relaxation, *Computers and Operations Research*, 28(1), 2001,33–52. 2.1

[39] *GAMS: The Solver Manuals,* GAMS Development Corporation, Washington, D.C., 2004. 3.4, 4.6.3

[40] Haarala, N., Large-Scale Nonsmooth Optimization: Variable Metric Bundle Method with Limited Memory. *PhD thesis,* University of Jyväskylä, Department of Mathematical Information Technology, 2004. (document), 2.3.3

[41] Haarala, N., Miettinen, K., and Mäkelä, M.M., New limited memory bundle method for large-scale nonsmooth optimization, *Optimization Methods and Software*, 19(6), 2004, 673–692. (document), 2.3.3

[42] Haarala, N., Miettinen, K., and Mäkelä, M.M., Globally convergent limited memorey bundle method for large-scale nonsmooth optimization, *Mathematical Programming*, 109(1), 2007, 181–205. (document), 2.3.3

[43] Hare, W., and Sagastizábal, C., A Redistributed Proximal Bundle Method for Nonconvex Optimization, *SIAM J. on Optimization*, 20(5), 2010, 2442–2473. 2.3.1

[44] Hiriart-Urruty, J.B., and Lemaréchal, C., Convex Analysis and Minimization Algorithms, *Springer Verlag, Heidelberg*, Vol. 1 and 2, 1993. (document), 2.4

[45] Karmitsa, N., Bagirov, A., and Mäkelä, M. M., Comparing different nonsmooth minimization methods and software *Optimization Methods and Software*, 27(1), 2012, 131–153. (document)

[46] Karmitsa, N.M.S, Mäkelä, M.M., and Ali, M.M., Limited memory interior point bundle method for large inequality constrained nonsmooth minimization, *Applied Mathematics and Computation*, 198(1), 2008, 382-400. (document)

[47] Kelley, Jr, J.E., The cutting-plane method for solving convex programs, *Journal of the Society for Industrial and Applied Mathematics*, 8(4), 1960, 703–712. 2.2

[48] Kiwiel, K.C., Methods of Descent for Nondifferentiable Optimization, *Lecture Notes in Mathematics, Springer-Verlag, Berlin*, 1133, 1985. (document), 2.3.1, 2.3.1, 2.4

[49] Korkmaz, S., Ledoux, E., and Önder, H., Application of the coupled model to the Somme river basin, *Journal of Hydrology,* 366, 2009, 21–34. 2.5

[50] Kort, B.W., and Bertsekas, D.P., A new penalty function method for constrained minimization, *In Decision and Control, 11th Symposium on Adaptive Processes. Proceedings of the 1972 IEEE Conference on*, 11, 1972, 162–166. (document)

[51] Kropat, E., Weber, G.W., and Ruckmann, J.J., Regression analysis for clusters in gene-environment networks based on ellipsoidal calculus and optimization, *Dynamics of Continuous, Discrete and Impulsive Systems Series B,* 17(5), 2010, 639–657.

[52] Lemaréchal, C., Combining Kelley's and conjugate gradient methods, *9th International symposium on mathematical programming*, (Budapest, Hungary, 1976). 2.3.1

[53] Liberti, L., and Kucherenko, S., Comparison of deterministic and stochastic approaches to global optimization, *International Transactions in Operational Research,* 12(3), 2005, 263–285. 2

[54] Lopez-Ibanez, M., Devi Prasad, T., and Paechter, B., Ant colony optimization for optimal control of pumps in water distribution networks, *Journal of Water Resources, Planning and Management, ASCE*, 134(4), 2008, 337–346. 2.5, 5.1, 5.1.1

[55] Luenerger, D. C., Linear and Nonlinear Programming, *2nd Edition, Addition Wesley, Reading, MA*, 1989. 2.1

[56] Lukśan, L., and Vlcek, J., A bundle-Newton method for nonsmooth unconstrained minimization, *Mathematical Programming: Series A and B*, 83(1), 1998,373–391. 2.3.2, 4.6

[57] Lukśan, L., and Vlćek, J., Globally Convergent Variable Metric Method for Convex Nonsmooth Unconstrained Minimization, *Journal of Optimization Theory and Applications*, 102(3), 1999, 593–613. 2.3.3, 4.6

[58] Lukśan, L., and Vlćek, J., Test problems for nonsmooth unconstrained and linearly constrained optimization, *Technical report No. 798, Institute of Computer Science, Academy of Sciences of the Czech Republic*, 2000. 3.4, 3.4.1, 4.6, 4.6.3

[59] Lukśan, L., and Vlćek, J., Algorithm 811: NDA: Algorithms for nondifferentiable optimization, *ACM Transaction on Math. Software,* 27(2), 2001, 193–213. 4.6

[60] Luss, H., Minimax resource-allocation problems: Optimization and parametric analysis, *European Journal of Operational Research*, 60(1), 1992, 76–86. 2.4

[61] Mäkelä, M.M., and Neittaanmaki, P., *Nonsmooth optimization: analysis and algorithms with applications to optimal control*, World Scientific, Singapore, 1992. (document), 3, 2.3.1, 2.4, 4.2, 4.3, 4.6

[62] Mäkelä, M.M., Survey of Bundle Methods for Nonsmooth Optimization, Optimization Methods and Software, *Optimization Methods and Software*, 17(1), 2002, 1–29. 2.3.1, 2.3.1

[63] Martinez, J., and Qi, L., Inexact Newton Methods for Solving Non-smooth Equations, *Journal of Computational and Applied Mathematics*, 60(1), 1995, 127–145.

[64] Mifflin, R., An algorithm for constrained optimisation with semismooth functions, *Mathematics of Operations Research*, 2(2), 1977, 191–207. (document)

[65] Mifflin, R., Sun, D., and Qi, L., Quasi-Newton bundle-type methods for nondifferentiable convex optimization, *SIAM J. on Optimization*, 8(2), 1998, 583–603. (document), 2.3.1

[66] Moré,J.J., Garbow, B. S., and K. E. Hillstrom, Testing unconstrained optimization software, *ACM Transactions on Mathematical Software (TOMS)*, 7(1), 1981, 17–41.

[67] Nedić, A., and Ozdaglar, A., Subgradient methods for saddle-point problems, *J. Optim*. Theory Appl., 142(1), 2009, 205–228. (document), 2.1

[68] Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A., Robust stochastic approximation approach to stochastic programming, *SIAM J. on Optimization,* 19(4), 2009, 1574–1609. (document), 2.1

[69] Nesterov, Y., Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1), 2009, 221–259. (document), 2.1

[70] Nicklow, J., Reed, P., Savic, D., Dessalegne, T., Harrell, L., Chan-Hilton, A., Karamouz, M., Minsker, B., Ostfeld, A., Singh, A., and Zechman, E., State of the Art for genetic algorithms and beyond in water resources planning and management, *Journal of Water Resources, Planning and Management, ASCE,* 136(4), 2010, 412–432. 2.5

[71] Nitivattananon, V., Sadowski, E.C., and Quimpo, R.G., Optimization of water supply system operation, *Journal of Water Resources, Planning and Management, ASCE,* September/October, 1996, 374–384. 2.5

[72] Nie, P.Y., An SQP approach with line search for a system of nonlinear equations, *Mathematical and computer modelling*, 43, 2006, 368–373.

[73] Nie, P.Y., A derivative-free method for the system of nonlinear equations, *Nonlinear analysis: real world applications*, 7, 2006, 378–384.

[74] Nocedal, J., Updating quasi-Newton matrices with limited storage, *Mathematics of computation*, 35(151), 1980, 773–782. 2.3.3

[75] Ormbsee, L.E., and Lansey, K.E., Optimal control of water supply pumping systems, *Journal of Water Resources, Planning and Management, ASCE,* 120(2), 1994, 237–252. 2.5

[76] Ostfeld, A., and Shamir, U., Optimal operation of multiquality networks. I: Steady-state conditions, *Journal of Water Resources Planning and Management*, 119(6), 1993, 645–662. 2.5

[77] Ostfeld, A., and Shamir, U., Optimal Operation of Multiquality Networks. II: Unsteady Conditions, *Journal of Water Resources Planning and Management*, 119(6), 1993, 663–684. 2.5

[78] Pang, J.S., and Yu, C.S., A minmax resource-allocation problem with substitutions, *European Journal of Operational Research*, 41(2), 1989, 218–223. 2.4

[79] Pasha, M.F.K., and Lansey, K., Optimal pump scheduling by linear programming, *In: Proceedings of World Environmental and Water Resources Congress*, Kanzas city, May 17-21, 2009, 395–404. 2.5

[80] Pezeshk, S., and Helweg, O.J., Adaptive search optimization in reducing pump operating costs, *Journal of Water Resources, Planning and Management, ASCE,* January/February, 1996, 57–63. 2.5

[81] Polak, E., Royset, J.O., and Womersley, R.S., Algorithms with adaptive smoothing for finite minimax problems, *Journal of Optimization Theory and Applications,* 119(3), 2003, 459–484. (document), 2.4, 2.4.1

[82] Polyak, B.T., A General Method of Solving Extremum Problems, *Soviet Mathematics*, 8(3), 1967,593–597. 2.1

[83] Polyak, B.T., *Introduction to Optimization*, Optimization Software Inc., New York, 1987. (document), 2.1

[84] Rossman, L.A., *EPANET,* Risk Reduction Engineering Laboratory, *U.S. Environmental Protection Agency,* Cincinnati, Ohio, 2003. [www.epa.gov/ORD/NRMRL/wswrd/epanet.html] 5, 5.2

[85] Sakarya, A.B.A., and Mays, L.W., Optimal operation of water distribution pumps considering water quality, *em Journal of Water Resources, Planning and Management, ASCE,* 126(4), 2000, 210–220. 2.5

[86] Savic, D.A., Walters, G.A., and Schwab, M., Multiobjective genetic algorithms for pump scheduling in water supply, *Evolutionary Computing, AISB*, International Workshop, Selected papers, Springer-Verlag, 1997. 2.5

[87] Schwefel, H.P., *Numerical optimisation of computer models*, Wiley, Chichester, United Kingdom, 1981. 5.2

[88] Shor, N., Minimization methods for non-differentiable functions, Springer, 1985. (document), 2.1, 2.1

[89] Simpson, T.W., Poplinski, J.D., Koch, P. N., and Allen, J.K., Metamodels for computer-based engineering design: survey and recommendations, *Engineering with computers*, 17(2), 2001, 129–150. (document)

[90] Tsoukalas, A., Parpas, P., and Rustem, B., A smoothing algorithm for finite minmaxmin problems, *Optimization Letters,* 3, 2009, 49–62. 2.4

[91] Van Dijk, M., Van Vuuren, S.J., and Van Zyl, J.E., Optimising water distribution systems using a weighted penalty in a genetic algorithm, *Water SA*, 34(5), 2008, 537–548. 2.5

[92] Van Zyl, J.E., Savic, D.A., and Walters, G.A., Operational optimization of water distribution systems using a hybrid genetic algorithm, *Journal of Water Resources, Planning and Management, ASCE,* 130(2), 2004, 160–170. 2.5

[93] Vazquez, F.G., Gunzel, H., and Jongen, H.Th., On logarithmic smoothing of the maximum function, *Annals of Operations Research,* 101, 2001, 209–220. 2.4

[94] Vlček, J., and Lukśan, L., Globally convergent variable metric method for nonconvex nondifferentiable unconstrained minimization, *Journal of Optimization Theory and Applications*, 111(2), 2001,407–430. 2.3.2, 2.3.3

[95] Weber, G.W., Uǵur, Ö., Taylan, P., and Tezel, A., On optimization, dynamics and uncertainty: a tutorial for gene-environment networks, *Discrete Applied Mathematics*, 157(10), 2009, 2494–2513.

[96] Wolfe, P.H., A method of conjugate subgradients of minimizing nondifferentiable convex functions, *Mathematical Programming Study,* 3, 1975, 145–173. (document)

[97] Xavier, A.E., Penalizaćao hiperbólica, I Congresso Latino-Americano de Pesquisa Operacional e Engenharia de Sistemas, 8 a 11 de Novembro, Rio de Janeiro, Brasil, 1982, 468–482. (document), 2.4.2

[98] Xavier, A.E, and Fernandes, A.A., Optimal Covering of Plane Domains by Circles Via Hyperbolic Smoothing, *Journal of Global Optimization*, 31(3), 2005, 493-504. (document), 2.4.2, 4.3

[99] Xavier, A.E., The hyperbolic smoothing clustering method, *Pattern Recognition*, 43, 2010, 731–737. (document), 2.4.2, 4.3

[100] Xiao, Y., and Yu, B., A truncated aggregate smoothing Newton method for minimax problems, *Applied Mathematics and Computation,* 216, 2010, 1868–1879. 2.4

[101] Xu, H., and Chang, X.W., Approximate Newton Methods for Nonsmooth Equations, *Journal of Optimization Theory and Applications*, 93(2), 1997, 373–394.

[102] Xu, S., Smoothing method for minimax problems, *Computational Optimization and Applications,* 20(3), 2001, 267–279. 2.4

[103] Zang, I., A smoothing-out technique for min-max optimization, *Mathematical Programming,* 19, 1980, 61–77. 2.4

[104] Zessler, U., and Shamir, U., Optimal operation of water distribution systems, *Journal of Water Resources Planning and Management,* 115(6), 1989, 735–752. 2.5

Appendix A

Test problems for minimax optimization

Problem 2.1 CB2

$$\begin{aligned} F(x) &= \max_{1\le i\le 3} f_i(x), \\ f_1(x) &= x_1^2 + x_2^4, \\ f_2(x) &= (2-x_1)^2 + (2-x_2)^2, \\ f_3(x) &= 2\exp(x_2 - x_1), \\ \bar{x}_1 &= 2, \quad \bar{x}_2 = 2. \end{aligned}$$

Problem 2.2 WF

$$\begin{aligned} F(x) &= \max_{1\le i\le 3} f_i(x), \\ f_1(x) &= \frac{1}{2}(x_1 + \frac{10x_1}{x_1+0.1} + 2x_2^2), \\ f_2(x) &= \frac{1}{2}(-x_1 + \frac{10x_1}{x_1+0.1} + 2x_2^2), \\ f_3(x) &= \frac{1}{2}(x_1 - \frac{10x_1}{x_1+0.1} + 2x_2^2), \\ \bar{x}_1 &= 3, \quad \bar{x}_2 = 1. \end{aligned}$$

Problem 2.3 SPIRAL

$$
\begin{aligned}
F(x) &= \max(f_1(x), f_2(x)), \\
f_1(x) &= (x_1 - \sqrt{x_1^2 + x_2^2}\cos\sqrt{x_1^2 + x_2^2})^2 + 0.005(x_1^2 + x_2^2), \\
f_2(x) &= (x_2 - \sqrt{x_1^2 + x_2^2}\sin\sqrt{x_1^2 + x_2^2})^2 + 0.005(x_1^2 + x_2^2), \\
\bar{x}_1 &= 1.41831, \quad \bar{x}_2 = -4.79462.
\end{aligned}
$$

Problem 2.4 EVD52

$$
\begin{aligned}
F(x) &= \max_{1 \le i \le 6} f_i(x), \\
f_1(x) &= x_1^2 + x_2^2 + x_3^2 - 1, \\
f_2(x) &= x_1^2 + x_2^2 + (x_3 - 2)^2, \\
f_3(x) &= x_1 + x_2 + x_3 - 1, \\
f_4(x) &= x_1 + x_2 - x_3 + 1, \\
f_5(x) &= 2x_1^3 + 6x_2^2 + 2(5x_3 - x_1 + 1)^2, \\
f_6(x) &= x_1^2 - 9x_3, \\
\bar{x}_i &= 1, \quad i = 1, 2, 3.
\end{aligned}
$$

problem 2.5 Rosen-Suzuki

$$
\begin{aligned}
F(x) &= \max_{1 \le i \le 4} f_i(x), \\
f_1(x) &= x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4, \\
f_2(x) &= f_1(x) + 10(x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_1 - x_2 + x_3 - x_4 - 8), \\
f_3(x) &= f_1(x) + 10(x_1^2 + 2x_2^2 + x_3^2 + 2x_4^2 - x_1 - x_4 - 10), \\
f_4(x) &= f_1(x) + 10(2x_1^2 + x_2^2 + x_3^2 + 2x_1 - x_2 - x_4 - 5), \\
\bar{x}_i &= 0, \quad i = 1, 2, 3, 4.
\end{aligned}
$$

Problem 2.6 Polak 6

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 4} f_i(x),\\
f_1(x) &= (x_1-(x_4+1)^4)^2+(x_2-(x_1-(x_4+1)^4)^4)^2+2x_3^2\\
&\quad +x_4^2-5(x_1-(x_4+1)^4)-5(x_2-(x_1-(x_4+1)^4)^4)-21x_3+7x_4,\\
f_2(x) &= f_1(x)+10((x_1-(x_4+1)^4)^2+(x_2-(x_1-(x_4+1)^4)^4)^2\\
&\quad +x_3^2+x_4^2+(x_1-(x_4+1)^4)-(x_2-(x_1-(x_4+1)^4)^4)+x_3-x_4-8),\\
f_3(x) &= f_1(x)+10((x_1-(x_4+1)^4)^2+2(x_2-(x_1-(x_4+1)^4)^4)^2\\
&\quad +x_3^2+2x_4^2-(x_1-(x_4+1)^4)-x_4-10),\\
f_4(x) &= f_1(x)+10((x_1-(x_4+1)^4)^2+(x_2-(x_1-(x_4+1)^4)^4)^2\\
&\quad +x_3^2+2(x_1-(x_4+1)^4)-(x_2-(x_1-(x_4+1)^4)^4)-x_4-5),\\
\bar{x}_i &= 0, \quad i=1,2,3,4.
\end{aligned}
$$

Problem 2.7 PBC3

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 21} |f_i(x)|,\\
f_i(x) &= \frac{x_3}{x_2}\exp(-t_ix_1)\sin(t_ix_2)-y_i,\\
y_i &= \frac{3}{20}e^{-t_i}+\frac{1}{52}e^{-5t_i}-\frac{1}{65}e^{-2t_i}(3\sin 2t_i+11\cos 2t_i),\\
t_i &= 10(i-1)/20, \quad 1\le i\le 21,\\
\bar{x}_i &= 1, \quad 1\le i\le 3.
\end{aligned}
$$

Problem 2.9 Kowalik-Osborne

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 11} |f_i(x)|,\\
f_i(x) &= \frac{x_1(u_i^2+x_2u_i)}{u_i^2+x_3u_i+x_4}-y_i.
\end{aligned}
$$

i	y_i	u_i
1	0.1957	4.0000
2	0.1947	2.0000
3	0.1735	1.0000
4	0.1600	0.5000
5	0.0844	0.2500
6	0.0627	0.1670
7	0.0456	0.0125
8	0.0342	0.1000
9	0.0323	0.0833
10	0.0235	0.0714
11	0.0246	0.0625

$$\bar{x}_1 = 0.250, \;\; \bar{x}_2 = 0.390, \;\; \bar{x}_3 = 0.415, \;\; \bar{x}_4 = 0.390.$$

Problem 2.10 Davidson 2

$$\begin{aligned} F(x) &= \max_{1 \le i \le 20} |f_i(x)|, \\ f_i(x) &= (x_1 + x_2 t_i - \exp(t_i))^2 + (x_3 + x_4 \sin(t_i) - \cos(t_i))^2. \\ t_i &= 0.2i, \qquad 1 \le i \le 20, \\ \bar{x}_1 &= 25, \quad \bar{x}_2 = 5, \quad \bar{x}_3 = -5, \quad \bar{x}_4 = -1. \end{aligned}$$

Problem 2.11 OET5

$$\begin{aligned} F(x) &= \max_{1 \le i \le 21} |f_i(x)|, \\ f_i(x) &= x_4 - (x_1 t_i^2 + x_2 t_i + x_3)^2 - \sqrt{t_i}. \\ t_i &= 0.25 + 0.75(i-1)/20, \qquad 1 \le i \le 21, \\ \bar{x}_i &= 1.0, \quad 1 \le i \le 4. \end{aligned}$$

Problem 2.12 OET6

$$\begin{aligned} F(x) &= \max_{1\le i\le 21} |f_i(x)|, \\ f_i(x) &= x_1 e^{x_3 t_i} + x_2 e^{x_4 t_i} - \frac{1}{1+t_i}, \\ t_i &= -0.5 + (i-1)/20, \qquad 1 \le i \le 21, \\ \bar{x}_1 &= 1.0, \quad \bar{x}_2 = 1.0, \quad \bar{x}_3 = -3.0, \quad \bar{x}_4 = -1.0. \end{aligned}$$

Problem 2.14 EXP

$$\begin{aligned} F(x) &= \max_{1\le i\le 21} f_i(x), \\ f_i(x) &= \frac{x_1 + x_2 t_i}{1 + x_3 t_i + x_4 t_i^2 + x_5 t_i^3} - \exp(t_i), \\ t_i &= -1 + (i-1)/10, \qquad 1 \le i \le 21, \\ \bar{x}_1 &= 0.5, \quad \bar{x}_2 = 0, \quad \bar{x}_3 = 0, \quad \bar{x}_4 = 0, \quad \bar{x}_5 = 0. \end{aligned}$$

Problem 2.15 PBC1

$$\begin{aligned} F(x) &= \max_{1\le i\le 30} |f_i(x)|, \\ f_i(x) &= \frac{x_1 + x_2 t_i + x_3 t_i^2}{1 + x_4 t_i + x_5 t_i^2} - \frac{\sqrt{(8t_i-1)^2+1}\arctan(8t_i)}{8t_i}, \\ t_i &= -1 + 2(i-1)/29, \qquad 1 \le i \le 30, \\ \bar{x}_1 &= 0, \quad \bar{x}_2 = -1, \quad \bar{x}_3 = 10, \quad \bar{x}_4 = 1, \quad \bar{x}_5 = 10. \end{aligned}$$

Problem 2.16 EVD61

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 51} |f_i(x)|,\\
f_i(x) &= x_1 \exp(-x_2 t_i)\cos(x_3 t_i + x_4) + x_5 \exp(-x_6 t_i) - y_i,\\
y_i &= 0.5e^{-t_i} - e^{-2t_i} + 0.5e^{-3t_i} + 1.5e^{-1.5t_i}\sin 7t_i + e^{-2.5t_i}\sin 5t_i,\\
t_i &= 0.1(i-1), \qquad 1 \le i \le 51,\\
\bar{x}_1 &= 2, \quad \bar{x}_2 = 2, \quad \bar{x}_3 = 7,\\
\bar{x}_4 &= 0, \quad \bar{x}_5 = -2, \quad \bar{x}_6 = 1
\end{aligned}
$$

Problem 2.18 Filter

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 41} |f_i(x)|,\\
f_i(x) &= \left(\frac{(x_1 + (1+x_2)\cos\vartheta_i)^2 + ((1-x_2)\sin\vartheta_i)^2}{(x_3 + (1+x_4)\cos\vartheta_i)^2 + ((1-x_4)\sin\vartheta_i)^2}\right)^{\frac{1}{2}} \cdot\\
&\quad \left(\frac{(x_5 + (1+x_6)\cos\vartheta_i)^2 + ((1-x_6)\sin\vartheta_i)^2}{(x_7 + (1+x_8)\cos\vartheta_i)^2 + ((1-x_8)\sin\vartheta_i)^2}\right)^{\frac{1}{2}} x_9 - y_i,\\
y_i &= |1 - 2t_i|, \quad \vartheta_i = \pi t_i\\
t_i &= 0.01(i-1), \;\; 1 \le i \le 6,\\
t_i &= 0.07 + 0.03(i-7), \;\; 7 \le i \le 20, \;\; t_{21} = 0.5,\\
t_i &= 0.54 + 0.03(i-22), \;\; 22 \le i \le 35,\\
t_i &= 0.95 + 0.01(i-36), \;\; 36 \le i \le 41,\\
\bar{x}_1 &= 0.00, \;\; \bar{x}_2 = 1.00, \;\; \bar{x}_3 = 0.00, \;\; \bar{x}_4 = -0.15,\\
\bar{x}_5 &= 0.00, \;\; \bar{x}_6 = -0.68, \;\; \bar{x}_7 = 0.00, \;\; \bar{x}_8 = -0.72,\\
\bar{x}_9 &= 0.37.
\end{aligned}
$$

Problem 2.19 Wong 1

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 5} f_i(x), \\
f_1(x) &= (x_1-10)^2 + 5(x_2-12)^2 + x_3^4 + 3(x_4-11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 \\
&\quad -4x_6x_7 - 10x_6 - 8x_7, \\
f_2(x) &= f_1(x) + 10(2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 - 127), \\
f_3(x) &= f_1(x) + 10(7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 - 282), \\
f_4(x) &= f_1(x) + 10(23x_1 + x_2^2 + 6x_6^2 - 8x_7 - 196), \\
f_5(x) &= f_1(x) + 10(4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7), \\
\bar{x}_1 &= 1,\ \bar{x}_2 = 2,\ \bar{x}_3 = 0,\ \bar{x}_4 = 4,\ \bar{x}_5 = 0,\ \bar{x}_6 = 1,\ \bar{x}_7 = 1.
\end{aligned}
$$

Problem 2.20 Wong 2

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 9} f_i(x), \\
f_1(x) &= x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3-10)^2 + 4(x_4-5)^2 + (x_5-3)^2 + \\
&\quad 2(x_6-1)^2 + 5x_7^2 + 7(x_8-11)^2 + 2(x_9-10)^2 + (x_{10}-7)^2 + 45, \\
f_2(x) &= f_1(x) + 10(3(x_1-2)^2 + 4(x_2-3)^2 + 2x_3^2 - 7x_4 - 120), \\
f_3(x) &= f_1(x) + 10(5x_1^2 + 8x_2 + (x_3-6)^2 - 2x_4 - 40), \\
f_4(x) &= f_1(x) + 10(0.5(x_1-8)^2 + 2(x_2-4)^2 + 3x_5^2 - x_6 - 30), \\
f_5(x) &= f_1(x) + 10(x_1^2 + 2(x_2-2)^2 - 2x_1x_2 + 14x_5 - 6x_6), \\
f_6(x) &= f_1(x) + 10(4x_1 + 5x_2 - 3x_7 + 9x_8 - 105), \\
f_7(x) &= f_1(x) + 10(10x_1 - 8x_2 - 17x_7 + 2x_8), \\
f_8(x) &= f_1(x) + 10(-3x_1 + 6x_2 + 12(x_9-8)^2 - 7x_{10}), \\
f_9(x) &= f_1(x) + 10(-8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12), \\
\bar{x}_1 &= 2,\ \bar{x}_2 = 3,\ \bar{x}_3 = 5,\ \bar{x}_4 = 5,\ \bar{x}_5 = 1,\ \bar{x}_6 = 2, \\
\bar{x}_7 &= 7,\ \bar{x}_8 = 3,\ \bar{x}_9 = 6,\ \bar{x}_{10} = 10.
\end{aligned}
$$

Problem 2.21 Wong 3

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 18} f_i(x),\\
f_1(x) &= x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3-10)^2 + 4(x_4-5)^2 + (x_5-3)^2\\
&\quad +2(x_6-1)^2 + 5x_7^2 + 7(x_8-11)^2 + 2(x_9-10)^2 + (x_{10}-7)^2 + (x_{11}-9)^2\\
&\quad +10(x_{12}-1)^2 + 5(x_{13}-7)^2 + 4(x_{14}-14)^2 + 27(x_{15}-1)^2 + x_{16}^4 + (x_{17}-2)^2\\
&\quad +13(x_{18}-2)^2 + (x_{19}-3)^2 + x_{20}^2 + 95,\\
f_2(x) &= f_1(x) + 10(3(x_1-2)^2 + 4(x_2-3)^2 + 2x_3^2 - 7x_4 - 120),\\
f_3(x) &= f_1(x) + 10(5x_1^2 + 8x_2 + (x_3-6)^2 - 2x_4 - 40),\\
f_4(x) &= f_1(x) + 10(0.5(x_1-8)^2 + 2(x_2-4)^2 + 3x_5^2 - x_6 - 30),\\
f_5(x) &= f_1(x) + 10(x_1^2 + 2(x_2-2)^2 - 2x_1x_2 + 14x_5 - 6x_6),\\
f_6(x) &= f_1(x) + 10(4x_1 + 5x_2 - 3x_7 + 9x_8 - 105),\\
f_7(x) &= f_1(x) + 10(10x_1 - 8x_2 - 17x_7 + 2x_8),\\
f_8(x) &= f_1(x) + 10(-3x_1 + 6x_2 + 12(x_9-8)^2 - 7x_{10}),\\
f_9(x) &= f_1(x) + 10(-8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12),\\
f_{10}(x) &= f_1(x) + 10(x_1 + x_2 + 4x_{11} - 21x_{12}),\\
f_{11}(x) &= f_1(x) + 10(x_1^2 + 5x_11 - 8x_{12} - 28),\\
f_{12}(x) &= f_1(x) + 10(4x_1 + 9x_2 + 5x_{13}^2 - 9x_{14} - 87),\\
f_{13}(x) &= f_1(x) + 10(3x_1 + 4x_2 + 3(x_{13}-6)^2 - 14x_{14} - 10),\\
f_{14}(x) &= f_1(x) + 10(14x_1^2 + 35x_{15} - 79x_{16} - 92),\\
f_{15}(x) &= f_1(x) + 10(15x_2^2 + 11x_{15} - 61x_{16} - 54),\\
f_{16}(x) &= f_1(x) + 10(5x_1^2 + 2x_2 + 9x_{17}^4 - x_{18} - 68),\\
f_{17}(x) &= f_1(x) + 10(x_1^2 - x_2 + 19x_{19} - 20x_{20} + 19),\\
f_{18}(x) &= f_1(x) + 10(7x_1^2 + 5x_2^2 + x_{19}^2 - 30x_{20}),\\
\bar{x}_1 &= 2,\ \bar{x}_2 = 3,\ \bar{x}_3 = 5,\ \bar{x}_4 = 5,\ \bar{x}_5 = 1,\ \bar{x}_6 = 2,\ \bar{x}_7 = 7,\\
\bar{x}_8 &= 3,\ \bar{x}_9 = 6,\ \bar{x}_{10} = 10,\ \bar{x}_{11} = 2,\ \bar{x}_{12} = 2,\ \bar{x}_{13} = 6,\ \bar{x}_{14} = 15,\\
\bar{x}_{15} &= 1,\ \bar{x}_{16} = 2,\ \bar{x}_{17} = 1,\ \bar{x}_{18} = 2,\ \bar{x}_{19} = 1,\ \bar{x}_{20} = 3.
\end{aligned}
$$

Problem 2.22 Polak 2

$$
\begin{aligned}
F(x) &= \max\{f(x+2e_2), f(x-2e_2)\},\\
f(x) &= \exp(10^{-8}x_1^2 + x_2^2 + x_3^2 + 4x_4^2 + x_5^2 + x_6^2 + x_7^2 + x_8^2 + x_9^2 + x_{10}^2),\\
e_2 &= \text{second column of the unit matrix},\\
\bar{x}_1 &= 100, \quad \bar{x}_i = 0.1, \quad 2 \le i \le 10.
\end{aligned}
$$

Problem 2.23 Polak 3

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 10} f_i(x),\\
f_i(x) &= \sum_{j=0}^{10} \frac{1}{i+j} \exp\left((x_{j+1} - \sin(i-1+2j))^2\right),\\
\bar{x}_i &= 1, \quad 1 \le i \le 11.
\end{aligned}
$$

Problem 2.24 Watson

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 31} |f_i(x)|,\\
f_1(x) &= x_1,\\
f_2(x) &= x_2 - x_1^2 - 1,\\
f_i(x) &= \sum_{j=2}^{n}(j-1)x_j \left(\frac{i-2}{29}\right)^{j-2} - \left[\sum_{j=1}^{n} x_j \left(\frac{i-2}{29}\right)^{j-1}\right]^2, \quad 3 \le j \le 31\\
\bar{x}_i &= 0, \quad 1 \le i \le 20.
\end{aligned}
$$

Problem 2.25 Osborne 2

$$
\begin{aligned}
F(x) &= \max_{1 \le i \le 65} |f_i(x)|, \\
f_i(x) &= y_i - x_1 \exp(-x_5 t_i) - x_2 \exp(-x_6 (t_i - x_9)^2) - x_3 \exp(-x_7 (t_i - x_{10})^2) - \\
&\quad x_4 \exp(-x_8 (t_i - x_{11})^2), \\
t_i &= 0.1(i-1), \quad 1 \le i \le 65.
\end{aligned}
$$

i	y_i	i	y_i	i	y_i
1	1.366	23	0.694	45	0.672
2	1.191	24	0.644	46	0.708
3	1.112	25	0.624	47	0.633
4	1.013	26	0.661	48	0.668
5	0.991	27	0.612	49	0.645
6	0.885	28	0.558	50	0.632
7	0.831	29	0.533	51	0.591
8	0.847	30	0.495	52	0.559
9	0.786	31	0.500	53	0.597
10	0.725	32	0.423	54	0.625
11	0.746	33	0.395	55	0.739
12	0.679	34	0.375	56	0.710
13	0.608	35	0.372	57	0.729
14	0.655	36	0.391	58	0.720
15	0.616	37	0.396	59	0.636
16	0.606	38	0.405	60	0.581
17	0.602	39	0.428	61	0.428
18	0.626	40	0.429	62	0.292
19	0.651	41	0.523	63	0.162
20	0.724	42	0.562	64	0.098
21	0.649	43	0.607	65	0.054
22	0.649	44	0.653		

$$\bar{x}_1 = 1.30,\ \bar{x}_2 = 0.65,\ \bar{x}_3 = 0.65,\ \bar{x}_4 = 0.70,\ \bar{x}_5 = 0.60,\ \bar{x}_6 = 3.00,$$

$$\bar{x}_7 = 5.00,\ \bar{x}_8 = 7.00,\ \bar{x}_9 = 2.00,\ \bar{x}_{10} = 4.50,\ \bar{x}_{11} = 5.50.$$

Appendix B

Test problems for general nonsmooth optimization

Problem 3.1 Rosenbrock

$$\begin{aligned} F(x) &= 100(x_2 - x_1^2)^2 + (1 - x_1)^2, \\ \bar{x}_1 &= -1.2, \quad \bar{x}_2 = 1.0. \end{aligned}$$

Problem 3.2 Crescent

$$\begin{aligned} F(x) &= \max\{x_1^2 + (x_2 - 1)^2 + x_2 - 1, -x_1 - (x_2 - 1)^2 + x_2 + 1\}, \\ \bar{x}_1 &= -1.5, \quad \bar{x}_2 = 2.0. \end{aligned}$$

Problem 3.3 CB2

$$\begin{aligned} F(x) &= \max\{x_1^2 + x_2^4, (2 - x_1)^2 + (2 - x_2)^2, 2e^{-x_1+x_2}\}, \\ \bar{x}_1 &= 1.0, \quad \bar{x}_2 = -0.1. \end{aligned}$$

Problem 3.4 CB3

$$\begin{aligned} F(x) &= \max\{x_1^4 + x_2^2, (2-x_1)^2 + (2-x_2)^2, 2e^{-x_1+x_2}\}, \\ \bar{x}_1 &= 2, \quad \bar{x}_2 = 2. \end{aligned}$$

Problem 3.5 DEM

$$\begin{aligned} F(x) &= \max\{5x_1 + x_2, -5x_1 + x_2, x_1^2 + x_2^2 + 4x_2\}, \\ \bar{x}_1 &= 1, \quad \bar{x}_2 = 1. \end{aligned}$$

Problem 3.6 QL

$$\begin{aligned} F(x) &= \max_{1 \le i \le 3} f_i(x), \\ f_1(x) &= x_1^2 + x_2^2, \\ f_2(x) &= x_1^2 + x_2^2 + 10(-4x_1 - x_2 + 4), \\ f_3(x) &= x_1^2 + x_2^2 + 10(-x_1 - 2x_2 + 6), \\ \bar{x}_1 &= 1, \quad \bar{x}_2 = 5. \end{aligned}$$

Problem 3.7 LQ

$$\begin{aligned} F(x) &= \max\{-x_1 - x_2, -x_1 - x_2 + (x_1^2 + x_2^2 - 1)\}, \\ \bar{x}_1 &= -0.5, \quad \bar{x}_2 = -0.5. \end{aligned}$$

Problem 3.8 MIFFLIN1

$$\begin{aligned} F(x) &= -x_1 + 20\max\{x_1^2 + x_2^2 - 1, 0\}, \\ \bar{x}_1 &= 0.8, \quad \bar{x}_2 = 0.6. \end{aligned}$$

Problem 3.9 MIFFLIN2

$$\begin{aligned} F(x) &= -x_1 + 2(x_1^2 + x_2^2 - 1) + 1.75|x_1^2 + x_2^2 - 1|, \\ \bar{x}_1 &= -1, \quad \bar{x}_2 = -1. \end{aligned}$$

Problem 3.10 Wolfe

$$\begin{aligned} F(x) &= f_1(x), \ \ x_1 \geq |x_2|, \\ F(x) &= f_2(x), \ \ 0 < x_1 \leq |x_2|, \\ F(x) &= f_3(x), \ \ x_1 \leq 0, \\ f_1(x) &= 5\sqrt{9x_1^2 + 16x_2^2}, \\ f_2(x) &= 9x_1 + 16|x_2|, \\ f_3(x) &= 9x_1 + 16|x_2| - x_1^9, \\ \bar{x}_1 &= 3, \quad \bar{x}_2 = 2. \end{aligned}$$

Problem 3.11 Rosen-Suzuki

$$\begin{aligned} F(x) &= \max\{f_1(x), f_1(x) + 10f_2(x), f_1(x) + 10f_3(x), f_1(x) + 10f_4(x)\}, \\ f_1(x) &= x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4, \\ f_2(x) &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_1 - x_2 + x_3 - x_4 - 8, \\ f_3(x) &= x_1^2 + 2x_2^2 + x_3^2 + 2x_4^2 - x_1 - x_4 - 10, \\ f_4(x) &= x_1^2 + x_2^2 + x_3^2 + 2x_1 - x_2 - x_4 - 5, \\ \bar{x}_i &= 0, \quad 1 \leq i \leq 4. \end{aligned}$$

Problem 3.12 Shor

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 10}\{b_i \sum_{j=1}^{5}(x_j - a_{ij})^2\}, \\
A &= \begin{pmatrix}
0 & 0 & 0 & 0 & 0 \\
2 & 1 & 1 & 1 & 3 \\
1 & 2 & 1 & 1 & 2 \\
1 & 4 & 1 & 1 & 2 \\
3 & 2 & 1 & 0 & 1 \\
0 & 2 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 2 & 1 \\
0 & 0 & 2 & 1 & 0 \\
1 & 1 & 2 & 0 & 0
\end{pmatrix}, \quad b = \begin{pmatrix}
1 \\ 5 \\ 10 \\ 2 \\ 4 \\ 3 \\ 1.7 \\ 2.5 \\ 6 \\ 3.5
\end{pmatrix} \\
\bar{x}_1 &= 0, \quad \bar{x}_2 = 0, \quad \bar{x}_3 = 0, \quad \bar{x}_4 = 0, \quad \bar{x}_5 = 1.
\end{aligned}
$$

Problem 3.15 El-Attar

$$
\begin{aligned}
F(x) &= \sum_{i=1}^{50} |x_1 e^{-x_2 t_i}\cos(x_3 t_i + x_4) + x_5 e^{-x_6 t_i} - y_i|, \\
y_i &= 0.5e^{-t_i} - e^{-2t_i} + 0.5e^{-3t_i} + 1.5e^{-1.5t_i}\sin 7t_i + e^{-2.5t_i}\sin 5t_i, \\
t_i &= 0.1(i-1), \quad 1 \le i \le 51, \\
\bar{x}_1 &= 2, \quad \bar{x}_2 = 2, \quad \bar{x}_3 = 7, \quad \bar{x}_4 = 0, \quad \bar{x}_5 = -2, \quad \bar{x}_6 = 1.
\end{aligned}
$$

Problem 3.16 Maxquad

$$
\begin{aligned}
F(x) &= \max_{1\le i\le 5}(x^T A^i x + x^T b^i),\\
A^i_{kj} &= A^i_{jk} = e^{j/k}\cos(jk)\sin(i),\ \ j<k,\\
A^i_{jj} &= \frac{j}{10}|\sin(i)| + \sum_{k\neq j}|A^i_{jk}|,\\
b^i_j &= e^{j/i}\sin(ij),\\
\bar{x}_i &= 0,\quad 1\le i\le 10.
\end{aligned}
$$

Problem 3.17 Gill

$$
\begin{aligned}
F(x) &= \max\{f_1(x), f_2(x), f_3(x)\}\\
f_1(x) &= \sum_{i=1}^{10}(x_i-1)^2 + 10^{-3}\sum_{i=1}^{10}(x_i^2-1/4)^2,\\
f_2(x) &= \sum_{i=2}^{30}\left[\sum_{i=2}^{10} x_j(j-1)(\frac{i-1}{29})^{j-2} - \left(\sum_{j=1}^{10} x_j(\frac{i-1}{29})^{j-1}\right)^2 - 1\right]^2\\
&\quad +x_1^2 + (x_2 - x_1^2 - 1)^2,\\
f_3(x) &= \sum_{i=1}^{10}\left[100(x_i - x_{i-1}^2)^2 + (1-x_i)^2\right],\\
\bar{x}_i &= -0.1,\quad 1\le i\le 10.
\end{aligned}
$$

Problem 3.18 Steiner 2

$$
\begin{aligned}
F(x) &= \sqrt{x_1^2 + x_{m+1}^2} + \sqrt{(\bar{a}_{21}-x_m)^2 + (\bar{a}_{22}-x_{2m})^2} +\\
&\quad \sum_{j=1}^{m} p_j\sqrt{(a_{j1}-x_j)^2 + (a_{j2}-x_{j+m})^2} +\\
&\quad \sum_{j=1}^{m-1}\tilde{p}_j\sqrt{(x_j - x_{j+1})^2 + (x_{j+m} - x_{j+m+1})^2},\quad m=6,
\end{aligned}
$$

$$
\begin{aligned}
\bar{a}_{21} &= 5.5, \ \bar{a}_{22} = -1.0, \\
a_{11} &= 0.0, \ a_{12} = 2.0, \quad p_1 = 2, \ \tilde{p}_1 = 1, \\
a_{21} &= 2.0, \ a_{22} = 3.0, \quad p_2 = 1, \ \tilde{p}_2 = 1, \\
a_{31} &= 3.0, \ a_{32} = -1.0, \ p_3 = 1, \ \tilde{p}_3 = 2, \\
a_{41} &= 4.0, \ a_{42} = -0.5, \ p_4 = 5, \ \tilde{p}_4 = 3, \\
a_{51} &= 5.0, \ a_{52} = 2.0, \quad p_5 = 1, \ \tilde{p}_5 = 2, \\
a_{61} &= 6.0, \ a_{62} = 2.0, \quad p_6 = 1,
\end{aligned}
$$

$$
\begin{aligned}
\bar{x}_1 &= (a_{11} + a_{21})/3, \qquad \bar{x}_{1+m} = (a_{12} + a_{22})/3, \\
\bar{x}_1 &= (\bar{x}_{j-1} + a_{j1} + a_{(j+1)1})/3, \ \bar{x}_{1+m} = (\bar{x}_{j-1+m} + a_{j2} + a_{(j+1)2})/3, \ 2 \le j \le m-1, \\
\bar{x}_1 &= (\bar{x}_{m-1} + a_{m1} + \bar{a}_{21})/3, \quad \bar{x}_{1+m} = (\bar{x}_{2m-1} + a_{m2} + \bar{a}_{22})/3.
\end{aligned}
$$

Problem 3.19 Maxq

$$
\begin{aligned}
F(x) &= \max_{1 \le i \le 20} x_i^2, \\
\bar{x}_i &= i, \quad i = 1, \ldots, 10, \qquad \bar{x}_i = -i, \quad i = 11, \ldots, 20.
\end{aligned}
$$

Problem 3.20 Maxl

$$
\begin{aligned}
F(x) &= \max_{1 \le i \le 20} |x_i|, \\
\bar{x}_i &= i, \quad i = 1, \ldots, 10, \qquad \bar{x}_i = -i, \quad i = 11, \ldots, 20.
\end{aligned}
$$

Problem 3.22 Goffin

$$
\begin{aligned}
F(x) &= 50 \max_{1 \le i \le 50} x_i - \sum_{i=1}^{50} x_i, \\
\bar{x}_i &= i - 25.5, \quad i = 1, 2, \ldots, 50.
\end{aligned}
$$

Problem 3.23 MXHILB

$$\begin{aligned} F(x) &= \max_{1\le i\le 50}\left|\sum_{i=1}^{50}\frac{x_j}{i+j-1}\right|, \\ \bar{x}_i &= 1, \quad i=1,2,\ldots,50. \end{aligned}$$

Problem 3.24 L1HILB

$$\begin{aligned} F(x) &= \sum_{i=1}^{50}\left|\sum_{j=1}^{50}\frac{x_j}{i+j-1}\right|, \\ \bar{x}_i &= 1, \quad i=1,2,\ldots,50. \end{aligned}$$

Problem 3.25 Shell Dual

$$\begin{aligned} F(x) &= 2\left|\sum_{i=1}^{5} d_i x_{i+10}^3\right| + \sum_{i=1}^{5}\sum_{j=1}^{5} c_{ij}x_{i+10}x_{j+10} - , \\ &\quad \sum_{i=1}^{10} b_i x_i + 100\left(\sum_{i=1}^{5}\max(0,P_i(x)) - Q(x)\right), \\ P_i(x) &= \sum_{j=1}^{10} a_{ij}x_j - 2\sum_{j=1}^{5} c_{ij}x_{j+10} - 3d_i x_{i+10}^2 - e_i, \quad 1\le i\le 5, \\ Q(x) &= \sum_{i=1}^{15}\min(0,x_i), \\ \bar{x}_i &= 10^{-4}, \quad i=1,2,\ldots,15, i\neq 7, \quad \bar{x}_7 = 60. \end{aligned}$$