| | |
|---|---|
| **Programa de Doctorat:** | Aplicacions Tècniques i Informàtiques de l'Estadística, la Investigació Operativa i l'Optimització |
| **Departament:** | Estadística i Investigació Operativa |
| **Director i tutor:** | Dr. Narcís NABONA FRANCISCO |

# Applicability of Deterministic Global Optimization to the Short-Term Hydrothermal Coordination Problem

**TESI DOCTORAL**

*presentada per*

**Albert FERRER BIOSCA**

*a*

**LA UNIVERSITAT POLITÈCNICA DE CATALUNYA**

*per a optar al grau de*

**DOCTOR EN MATEMÀTIQUES**

**BARCELONA, GENER 2004**

Albert Ferrer Biosca
Departament de Matemàtica Aplicada I
Universitat Politècnica de Catalunya
Avgda. Dr. Gregorio Marañón, 42-50
08028-Barcelona (Spain)
alberto.ferrer@upc.es

A la Matilde
i al meu pare Lluis i a la meva mare Salut
i al meu germà Marcel

# Acknowledgements

I am indebted to my PhD supervisor, Professor Narcís Nabona Francisco, who has provided indispensable encouragement, help and support during the time-consuming process of preparing this Thesis.

I would like to express my deepest gratitude to Professor Juan Enrique Martínez Legaz for his advice and kind attention throughout the course of this work.

Special thanks are due to the Professors Dinh The Luc and Alex Rubinov for their valuable comments and suggestions that have helped to improve the contents of the Thesis.

I wish to express my sincere thanks to all those other persons who have contributed directly or indirectly to the process of writing this work.

Last but not least, warm thanks go to my wife Matilde, my daughter Alba and my son Damià for their patience and understanding.

# Contents

# List of Tables

# List of Figures

# Notations

Generation Problem

| | |
|---|---|
| $N_e$ | number of reservoirs |
| $N_t$ | number of time intervals |
| $d_j^i$ | water discharges from reservoir $j$ over the $i^{\text{th}}$ time interval |
| $v_j^i$ | volume stored in reservoir $j$ at the end of the $i^{\text{th}}$ time interval |
| $l^i$ | forecast for electricity consumption over the $i^{\text{th}}$ time interval |
| $w_j^i$ | natural water inflow into reservoir $j$ over the $i^{\text{th}}$ time interval |
| $h_j^i$ | power hydrogeneration function of the $j^{\text{th}}$ reservoir into the $i^{\text{th}}$ time interval |
| $s_w^j$ | headwater elevation of the $j^{\text{th}}$ reservoir |
| $\bar{s}_w^j$ | tailwater elevation of the $j^{\text{th}}$ reservoir |
| $\mathsf{s}_j$ | head between the headwater elevation and the tailwater elevation of the $j^{\text{th}}$ reservoir; $\mathsf{s}_j = s_w^j - \bar{s}_w^j$ |
| $\tilde{s}_j$ | equivalent head of the $j^{\text{th}}$ reservoir |
| $s_{vb}$ | basic coefficient in the relationship between the headwater elevation and the volume stored |
| $s_{vl}$ | linear coefficient in the relationship between the headwater elevation and the volume stored |
| $s_{vq}$ | quadratic coefficient in the relationship between the headwater elevation and the volume stored |
| $s_{vc}$ | cubic coefficient in the relationship between the headwater elevation and the volume stored |
| $s_{db}$ | basic coefficient in the relationship between the tailwater elevation and the water discharge |
| $s_{dl}$ | linear coefficient in the relationship between the tailwater elevation and the water discharge |

| | |
|---|---|
| $s_{dq}$ | quadratic coefficient in the relationship between the tailwater elevation and the water discharge |
| $g$ | acceleration due to gravity |
| $\mathsf{d}E_p$ | potential energy increment (or differential) |
| $\mathsf{d}v_j$ | water stored increment (or differential) |
| $t^i$ | $i^{\text{th}}$ time interval |
| $\rho^i_j$ | efficiency coefficient of the power hydrogeneration function $h^j_i$ |
| $k^i_j$ | efficiency and unit conversion coefficient; $k^i_j = \rho^i_j g/t^i$ |
| $f^i_j - g^i_j$ | d.c. representation of the power hydrogeneration function $h^i_j$ |
| $\varphi^i_1(z), \varphi^i_2(z)$ | convex functions to obtain the d.c. constraints and the d.c. objective function $(\varphi_1(z) - \varphi_2(z))$ of the d.c. program of reduced size |

Vector spaces and polynomials

| | |
|---|---|
| $\bigoplus_{i=0}^m F_i$ | direct sum of the vector spaces $F_i$ |
| $\langle e_1 \ldots e_r \rangle$ | the span of the set $\{e_1 \ldots e_r\}$ |
| $I\!\!R_m\,[x_1, ..., x_n]$ | the vector space of polynomials of degree until $m$ |
| $H_i\,[x_1, ..., x_n]$ | the vector space of homogeneous polynomials of degree $i$ |
| $I\!\!R_m\,[X]$ | $R_m\,[X] = R_m\,[x_1, ..., x_n]$ where $X = \{x_1, ..., x_n\}$ |
| $H_i\,[X]$ | $H_i\,[X] = H_i\,[x_1, ..., x_n]$ where $X = \{x_1, ..., x_n\}$ |
| $H^+,\ H^-$ | closed half spaces defined by $H = \{\sum_{i=1}^n a_i \cdot x_i = 0\}$ $H^+ = \{\sum_{i=1}^n a_i \cdot x_i \geq 0\}$, $H^- = \{\sum_{i=1}^n a_i \cdot x_i \leq 0\}$ |
| $q(x_1, \ldots, x_n)$ | polynomial in the variables $x_1, \ldots, x_n$ |
| $\overline{q}(x_0, x_1, \ldots, x_n)$ | homogenization of the polynomial $q(x_1, \ldots, x_n)$ defined by $\overline{q}(x_0, x_1, \ldots, x_n) = x_0^m q(\frac{x_1}{x_0}, \ldots, \frac{x_n}{x_0})$ |
| $B_k$ | the usual bases of the monomials in $H_k\,[x_1, ..., x_n]$ |
| $B^m$ | the usual bases of the monomials in $I\!\!R_m\,[x_1, ..., x_n]$; $B^m = \cup_{k=0}^m B_k$ |
| $K_k(C)$ | the nonempty closed convex cone of the polynomials on $H_k\,[x_1, ..., x_n]$ which are convex on the closed convex set $C$ |
| $K^m(C)$ | the nonempty closed convex cone of the polynomials on $I\!\!R_m\,[x_1, ..., x_n]$ which are convex on the closed convex set $C$ |
| $K(C)$ | to refer to the cones $K_k(C)$ and $K^m(C)$ indistinctly |
| $z + K(C)$ | the cone with apex $z$ |
| $\|z(x)\|_p$ | the $p$-norm in $I\!\!R\,[x_1, ..., x_n]$ for all $p = 0, 1, 2, \ldots$ and $p = \infty$ |
| $\|z(x)\|_{(p,k)}$ | sometimes, it is used to indicate the $p$-norm in $H_k\,[x_1, ..., x_n]$ |

Sets, functions and matrices

| | |
|---|---|
| $\emptyset$ | empty set |
| $I\!\!N$ | the set of the nonnegative integers ($0 \in I\!\!N$) |
| $I\!\!R$ | the set of the real numbers |
| $\overline{I\!\!R}$ | the set of the extended real numbers; $\overline{I\!\!R} := I\!\!R \cup \{-\infty, +\infty\}$ |
| $I\!\!R^n$ | $I\!\!R^n := I\!\!R \times I\!\!R \times \ldots \times I\!\!R$, $n$-times |
| $I\!\!R^{n\times m}$ | $I\!\!R^{n\times m} := I\!\!R^n \times I\!\!R^m$ |
| $B(x,\delta)$ | ball with center $x$ and radius $\delta > 0$; $B(x,\delta) := \{z : \ \|z-x\| < \delta\}$ |
| $\text{cone}(A)$ | the cone generated by $A$ (the smallest convex cone which contains $A$) |
| $\text{int}\ C$ | interior of $C$; the largest open set contained in $C$ |
| $\text{cl}\ C$ | closure of $C$; the smallest closed set containing $C$ |
| $D \setminus C$ | difference of the sets $D$ and $C$; $D \setminus C := \{x : \ x \in D,\ x \notin C\}$ |
| $\text{ext}\ C$ | exterior of $C$; $\text{ext}\ C := \text{int}\ (I\!\!R^n \setminus C)$ |
| $\partial C$ | the boundary of the set $C$; $\partial C := \text{cl}\ C \cap \text{cl}\ (I\!\!R^n \setminus C)$ |
| $\text{proj}_{I\!\!R^n} D$ | the projection of $D$ on $I\!\!R^n$ |
| $f \circ g$ | the composition of mappings $g : X \to Y$ and $f : Y \to Z$, i.e, $(f \circ g)(x) := f(g(x))$ |
| $\text{argmin} f(S)$ | the set of all global minimizers of the function $f$ on the set $S$ |
| $\nabla f(x)$ | gradient of the function f at the point x; |
| $\partial f(x)$ | subdiferential of the function f at the point x; |
| $\mathcal{DC}(A)$ | the class of d.c. functions on the set $A \subset I\!\!R^n$ |
| $\mathcal{C}^2(A)$ | the class of functions whose second partial derivatives are continuous everywhere on $A$, $A$ open set |
| $I\!\!R^{n*m}$ | the vector space of matrices with $n$ rows and $m$ columns |
| $Det\ (M)$ | determinant of the square matrix $M \in I\!\!R^{n*n}$ |
| $M \succeq N$ | when $M - N$ is positive semidefinite for any two symmetric matrices $M$ and $N$ in $I\!\!R^{n*n}$ |

# Preliminaries

## Convex sets

A set $A \subset I\!\!R^n$ is called a *convex* set if it contains any line segment between a pair of its points. The *dimension* of a convex set is the dimension of its affine hull (the smallest affine set containing $A$). A convex set $A \subset I\!\!R^n$ is said to be of *full dimension* if $\dim(A) = n$. Let $A$, $B$ be convex sets of $I\!\!R^n$ and let $t \in I\!\!R$. Then, the sets

- $A \cap B$,
- $A + B := \{z \in I\!\!R^n : z = x + y,\ x \in A,\ y \in B\}$,
- $tA := \{z \in I\!\!R^n : z = tx,\ x \in A,\ t \in I\!\!R\}$,

are convex sets. Given a set $X \subset I\!\!R^n$, the intersection of all convex sets which contain $X$ is called the *convex hull* of $X$ and it is denoted by $\text{conv}(X)$. The convex hull of $X$ coincides with all the *convex combinations* of its elements ($\sum_{i=1}^m t_i x_i$ with $x_i \in X$, $t_i \geq 0$, $\sum_{i=1}^m t_i = 1$, $m \in I\!\!N$) and it is the smallest convex set containing $X$.

Let $C \subset I\!\!R^n$ be a nonempty closed convex set, and consider $y \notin C$. Then there exists a hyperplane $H := \{x \in I\!\!R^n : c^t x = b\}$, with $b \in I\!\!R$ and non zero $c \in I\!\!R^n$ such that

1. $y \notin H^+$ ($y \in \text{int}(H^-)$),
2. $C \subset H^+$,

where $H^+ := \{x \in I\!\!R^n : c^t x \geq b\}$ and $H^- := \{x \in I\!\!R^n : c^t x \leq b\}$ are the closed halfspaces defined by $H$. An immediate consequence of this property is that a nonempty closed convex set $C$ is the intersection of all closed halfspaces containing $C$. In order to express $C$ as the intersection of halfspaces, we only need hyperplanes $H$ which contain a boundary point of $C$ (*supporting hyperplanes* of $C$), i.e., $H \cap C \neq \emptyset$ and $C \subset H^+$ or $C \subset H^-$ .

A set $M \subset \mathbb{R}^n$ is called a *cone* if $tM \subset M$ for all $t > 0$. Consider $a \in \mathbb{R}$ then, $C := a + M$ is called a cone with *apex* $a$. A cone which contains no line is said to be *pointed* in this case, 0 and $a$ are called the *vertex* of $M$ and $C$, respectively.

A set $M \subset \mathbb{R}^n$ is a convex cone if and only if

1. $tM \subset M$ for all $t > 0$,
2. $M + M \subset M$,

which is equivalent to saying that the cone $M$ contains all the positive linear combinations of its elements. Let $A$ be a convex set of $\mathbb{R}^n$. Denote by cone$(A)$ the smallest convex cone that contains $A$, which is said to be the cone *generated* by $A$. We can see that cone$(A) = \cup_{t>0}(tA)$.

## Convex functions

Let $\overline{\mathbb{R}} := \mathbb{R} \cup \{-\infty, +\infty\}$ be the set of the *extended real numbers* with the well-known rules of calculus with these new elements $+\infty$ and $-\infty$, and the meaningless situations such that $+\infty - \infty$ and $\pm\infty / \pm\infty$, among others, that must be avoided. A function $f : A \to \overline{\mathbb{R}}$ on a set $A \subset \mathbb{R}^n$, is said to be an *extended real-valued function* on $A$. The sets

- $\text{dom}(f) := \{x \in A : f(x) < +\infty\}$ and
- $\text{epi}(f) := \{(x, t) \in A \times \mathbb{R} : f(x) \leq t\} \subset \mathbb{R}^n \times \mathbb{R}$

are named the *effective domain* and the *epigraph* of $f$, respectively. If $\text{dom}(f) \neq \emptyset$ and $f(x) > -\infty$ for all $x \in A$, then $f$ is said to be a *proper* function.

A function $f : A \subset \mathbb{R}^n \to \overline{\mathbb{R}}$ is said to be *convex* on $A$ when epi$(f)$ is a convex set in $\mathbb{R}^n \times \mathbb{R}$. This is equivalent to say that $A$ is a convex set in $\mathbb{R}^n$ and

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)$$

for any $x_1,\ x_2 \in A$, $0 \leq \lambda \leq 1$ and the right hand side is defined. If the inequality is strict for $x_1 \neq x_2$ and $0 < \lambda < 1$, then the function is said to be *strictly convex.* It can be proved that if $f$ is convex on $A$ then

$$f\left(\sum_{i=1}^{m} \lambda_i x_i\right) \leq \sum_{i=1}^{m} \lambda_i f(x_i),$$

where $m \in I\!N$, $x_i \in A$, $0 \leq \lambda_i \leq 1$, $i = 1, \ldots, m$ and $\sum_{i=1}^{m} \lambda_i = 1$. Let $A \subset I\!R^n$ be a convex set. A function $f : A \to \overline{I\!R}$ is said to be *concave* (*strictly concave*) on $A$ when the function $-f$ is convex (*strictly convex*) on $A$. Many properties of convex functions can be deduced from corresponding properties of convex sets.

The following algebraic properties are a direct consequence of the definition of a convex function. Let $f_i : A \to \overline{I\!R}$, $i = 1, \ldots, m$ be proper convex functions on the convex set $A \subset I\!R^n$, then

- $\sum_{i=1}^{m} \alpha_i f_i(x)$, $\alpha_i \geq 0$, $i = 1, \ldots, m$ is convex,
- max $\{f_i(x), i = 1, \ldots, m\}$ is convex,
- sup $\{f(x), f \in \mathcal{F}\}$, where $\mathcal{F}$ is a family of proper convex functions on $A$, is also a convex function.

Convex functions have interesting continuity and differentiability properties which are very useful in optimization. Let $A \subset I\!R^n$ be a nonempty convex set of full dimension $n$.

- A convex function $f : A \to I\!R$ is continuous at every interior point of $A$. If $A = I\!R^n$, then $f$ is continuous everywhere. For $A \neq I\!R^n$ discontinuities can only be found at the boundary of $A$.
- If $f : A \to I\!R$ is differentiable on $A$, $A$ open convex set, then $f$ is convex if and only if $f(y) \geq f(x) + (y-x)^t \nabla f(x)$ for every $x, y \in A$. It is strictly convex if and only if the inequality is strict for $x \neq y$.
- If $f : A \to I\!R$ is twice-differentiable on $A$, $A$ open convex set, then $f$ is convex if and only if its Hessian matrix $H(x)$ is positive semidefinite for every $x \in A$, i.e., $y^t H(x) y \geq 0$ for every $x \in A$ and $y \in I\!R^n$. If $H(x)$ is positive definite for every $x \in A$, then $f$ is strictly convex.

Given a proper function $f : A \subset I\!R^n \to \overline{I\!R}$. A vector $p \in I\!R^n$ is said to be a *subgradient* of $f$ at a point $x \in A$ if $f(y) \geq f(x) + p^t(y-x)$ for all $y \in A$. The set of all subgradients at the point $x$ is said to be the *subdifferential* of the function $f$ at the point $x$, and it is denoted by $\partial f(x)$. A function $f$ is called *subdifferentiable* at $x$ if $\partial f(x) \neq \emptyset$.

Let $f : A \subset I\!\!R^n \to \overline{I\!\!R}$ be a proper convex function on $A$, $A$ a convex set. For any bounded set $S \subset \mathrm{int}(\mathrm{dom}(f))$ the set

$$\bigcup_{x \in S} \partial f(x)$$

is nonempty and bounded. In particular, $\partial f(x)$ is nonempty and bounded at every $x \in \mathrm{int}(\mathrm{dom}(f))$. If $f$ is differentiable at $x \in A$ then $\partial f(x) = \{\nabla f(x)\}$.

## D.c. functions

Let $f$ be a real valued function defined on a convex set $A \subset I\!\!R^n$. The function $f$ is called a *d.c. function* on $A$ if it can be expressed as a difference of two convex functions on $A$, i.e., there exist convex functions $f_1$ and $f_2$ on $A$ such that

$$f(x) = f_1(x) - f_2(x) \ \forall x \in A.$$

The pair of functions $(f_1, f_2)$ is said to be a *d.c. representation* of $f$ on $A$. Moreover, the functions $f_1$ and $f_2$ are called the *first component* and *the second component* respectively of the current d.c. representation of $f$ on $A$. On the other hand, a function $f$ is said to be *d.c. at a point* $x \in A$ if there exists a convex neighborhood $U_x$ of $x$ such that $f$ is d.c. on $U_x \cap A$. If $f$ is d.c. at every point of $A$, it is said to be *locally d.c.* on $A$. Every locally d.c. function on $A \subset I\!\!R^n$, $A$ an open or closed convex set, is d.c. on $A$ (see Hartman [25]). Moreover, it can be proved that every function $f \in C^2(A)$, $A$ open or closed convex set, is a d.c. function on $A$ (see Ellaia [14]). The set of the d.c. functions on $A$, denoted by $DC(A)$, is the vector space generated by the cone of convex functions on $A$. Given $f \in DC(A)$, it is evident that there are infinitely d.c. representations of $f$. Denote by $D_f(A)$ the set

$$D_f(A) := \{(f_1, f_2) : f(x) = f_1(x) - f_2(x) \ \forall x \in A, \ f_1, \ f_2 \text{ convex on } A\}.$$

When a d.c. representation $(f_1, f_2)$ of $f$ is available, then we can always obtain a new d.c. representation of $f$ in the form $(f_1 + g, f_2 + g)$, where both components are strictly convex by adding a strictly convex function $g(x)$ (a simple choice is $g(x) = t\|x\|^2$ with $t > 0$). $DC(A)$ has some interesting properties with respect to operations frequently encountered in optimization (see Hiriart-Urruty [27] or Horst et al [32]).

- Every linear combination of a finite number of d.c. functions is a d.c. function, which is a consequence of well known properties of convex and concave functions.

- Let $(p_i, q_i)$, $i = 1, \ldots, m$ be d.c. representations of the functions $f_i$, $i = 1, \ldots, m$. Then, $\max\{f_1(x), \ldots, f_m(x)\}$ is a d.c. function because we can write

$$\max\{f_1(x), \ldots, f_m(x)\} = \max\left\{p_i(x) + \sum_{j=1,\ j\neq i}^{m} q_j(x),\ i = 1, \ldots, m\right\} - \sum_{j=1}^{m} q_j(x).$$

- $\min\{f_1(x), \ldots, f_m(x)\}$ is a d.c. function because we know

$$\min\{f_1(x), \ldots, f_m(x)\} = -\max\{-f_1(x), \ldots, -f_m(x)\}.$$

- Also, we can see that $|f(x)| := \max\{f(x), -f(x)\}$, $f^+(x) := \max\{0, f(x)\}$ and $f^-(x) := \min\{0, f(x)\}$ are d.c. functions.

- The product of a pair of nonnegative-valued convex functions $q_1$ and $q_2$ is a d.c. function because we can write

$$q_1(x)q_2(x) = \frac{1}{2}(q_1(x) + q_2(x))^2 - \frac{1}{2}(q_1^2(x) + q_2^2(x)).$$

- Let $f_1$ and $f_2$ be d.c. functions on $A$, $A$ open or closed convex set in $I\!R^n$. Then, $f_1(x)f_2(x)))$ and, if for all $x \in A$, $f_2(x) \neq 0$, the quotient $f_1(x)/f_2(x)))$ are d.c. functions on $A$ (see Hartman [25]).

Some authors provide interesting theoretical d.c. representation results but no practical means to get them. D.c. functions were considered by Alexandrov [1] and Landis [44]. Some time later Hartman [25] states that every locally d.c. function on $A \subset I\!R^n$, $A$ open or closed convex set, is d.c. on $A$. Bougeard [6] proves that if $f \in C^2(A)$ then there exists a d.c. representation $(f_1, f_2)$ of $f$ in which $f_1 \in C^2(A)$ and $f_2 \in C^\infty(A)$. Penot and Bougeard [50] establish a similar result with more global assumptions. Indeed, let $A$ be an open convex set of a finite dimensional normed vector space, then any lower$-C^2$ function $f$ on $A$, in particular any $f \in C^2(A))$, can be written as $f = f_1 - f_2$ with $f_1$ and $f_2$ convex and $f_2 \in C^\infty(A)$. Moreover, every lower$-C^2$ function can be characterized by its (locally) decomposability as a sum of a convex continuous and a concave quadratic function (see [77]).

# Introduction

## Motivation

A global optimization programming problem has the general form

$$\begin{array}{rl} \text{minimize} & f(x) \\ \text{subject to:} & x \in S, \end{array}$$

where $S$ is a set contained in $\mathbb{R}^n$ and the minimizer is understood in the global sense, i.e., we are interested in points $x^* \in S$ satisfying

$$f(x^*) \;\leq\; f(x) \;\; \forall x \in S.$$

The set of *global minimizers* is denoted by $\text{argmin} f(S)$ and, at each $x^* \in \text{argmin} f(S)$, the corresponding value $f(x^*)$ is said to be the *global minimum* of the function $f$ at the point $x^*$ over the set $S$. On the other hand, a point $x^o \in S$ is said to be a *local minimizer* of the function $f$ over the set $S$, if there exists a neighborhood $V$ of $x^o$ satisfying

$$f(x^o) \leq f(x) \;\; \forall x \in S \cap V.$$

The corresponding value $f(x^o)$ is said to be the *local minimum* of the function $f$ at the point $x^o$ over the set $S$. If the set $S$ can be described as

$$S := \{x : g_i(x) \leq 0, i = 1, \ldots, m\}$$

and if all functions involved in the program are in $C^1(A)$, $A$ an open set containing $S$, the following Karush-Kuhn-Tucker ($KKT$) conditions hold at $x^o \in S$. There exist $\lambda_i \geq 0,\; i = 1, \ldots, m$, such that

1. $\lambda_i g_i(x^o) = 0,\; i = 1, \ldots, m,$
2. $\nabla f(x^o) + \sum_{i=1}^m \lambda_i \nabla g_i(x^o) = 0,$

provided that the constraints at the local minimizer $x^o$ are regular. When both the objective function $f$ and the feasible set $S$ are convex the program is said to be convex and, in this case, it is well known that every local minimizer is a global one.

Suppose that by using a standard local technique of nonlinear programming we have obtained a Karush-Kuhn-Tucker point of the above program. We must then stop the procedure because no local method can tell us whether the obtained point is a global optimizer or not and, in the latter case, how to proceed to obtain a better feasible point. It is this possibility of becoming trapped at a stationary point that causes the failure of local methods and motivates the need to develop global ones. In practice, direct problem formulations are not convenient, and it is thus necessary to transform them into alternative ones that are more suitable for algorithmic purposes. In any approach to global optimization methods it is essential to understand the mathematical structure of the problem under consideration. A careful analysis of this structure can provide insight into the most relevant properties of the problem and suggest efficient methods for solving it. In recent years, some papers have described deterministic global optimization procedures to solve problems whose objective function can directly be expressed as difference of convex functions (*d.c. function*), and the feasible domain is a convex set. They are a special class of global optimization programs named *d.c. programs* which are described in Subsection 1.4.3. The Multisource Weber Problem, the Facility Location Problem with limited distances, the Stochastic Transportation-Location Problem and the Stochastic Transportation Problem belong to this special class of d.c. programs. In Pey-Chun Chen et al [51], both the Multisource Weber Problem and the Facility Location Problem with limited distances are reformulated as a concave minimization problem, which is the simplest class of global optimization problem. In K. Holmberg et al [30], the Stochastic Transportation-Location Problem and the Stochastic Transportation Problem are reduced to d.c. optimization problems whose objective functions are separable d.c. functions and the feasible domains are defined by the transportation constraints. In this case, the procedure suggested in [30] takes advantage of these two special structures so an efficient rectangular subdivision branching method can be used to solve them.

This Thesis has been motivated by the interest in applying deterministic global optimization procedures to problems in the real world with no special structure. We have focused on the Short-Term Hydrothermal Coordination of Electricity Generation Problem (also named the *Generation Problem* in this Thesis) where the objective function and the nonlinear constraints are polynomials of degree up to four (see [26]). Its solution has important economic and technical implications. In the

Generation Problem neither a representation of the involved functions as difference of convex functions is at hand nor we can take advantage of any special structure of the problem. Hence, a very general problem such as

$$\begin{array}{rc} \text{minimize} & f(x) \\ \text{subject to:} & x \in S \subset I\!R^n, \end{array}$$

does not seem to have any mathematical structure conducive to computational implementations. However, when $f(x)$ is a continuous function and $S$ is a nonempty closed set the problem can be transformed into an equivalent problem expressed by

$$\begin{array}{rc} \text{minimize} & l(z) \\ \text{subject to:} & z \in D \setminus \text{int}C, \end{array}$$

which is said to be a *canonical d.c. program*, where $l(z)$ is a linear function and $D$ and $C$ are closed convex sets (see Section 1.3 for details). Thus, we can see that every continuous global optimization problem has a *mathematical complementary convex structure* ($D\setminus\text{int}C$) also called the *d.c. structure*. The mathematical complementary convex structure is not always apparent and, even when it is explicit, a lot of work still remains to be done to bring it into a form amenable to efficient computational implementations. The attractive feature of the mathematical complementary convex structure is that it involves convexity. Thus, we can use analytical tools from convex analysis like subdifferential and supporting hyperplane. On the other hand, since convexity is involved in the reverse convex property, these tools must be used in some specific way and combined with combinatorial tools like cutting planes, branch and bound and outer approximation.

## Objectives

A program expressed by

$$\begin{array}{rc} \text{minimize} & f(x) \\ \text{subject to:} & h_i(x) \leq 0, \; i = 1, \ldots, m, \\ & x \in S, \end{array}$$

is said to be a *d.c. program* when $S$ is a closed convex set into $I\!R^n$ and the functions $f(x)$ and $h_i(x)$, $i = 1, \ldots, m$ are *d.c. functions*, i.e., they are expressed explicitly as a difference of two convex functions on $S$. At the expense of introducing additional variables, any d.c. program can be transformed to a program with a complementary convex structure. While it is not too difficult to prove theoretically that a given function is a d.c. function, it is often very problematical to obtain an effective *d.c. representation* of a d.c. function as a difference of convex functions. For this reason our first objective is

O1.- how to write effectively a function as a difference of convex functions.

After that, our second objective is

O2.- to convert the Generation Problem into an equivalent reverse convex programming problem (see Section 1.4 for details) and develop a deterministic global optimization procedure to solve it.

Having solved the problem of finding a d.c. representation of a polynomial we then come up against another even more complicated problem, that is, if the computational efficiency depends on the d.c. representation of the functions. Our third objective is to answer the questions:

O3.- is there any d.c. representation that improves the computational efficiency? If the answer to this question is affirmative, then what is the best d.c. representation of a d.c. function (*optimal d.c. representation*) from a computational point of view and how can it be obtained?

Finally,

O4.- we want to compare, for the nonconvex Generation Problem, the solutions obtained by applying the deterministic global optimization algorithm and the solutions obtained with a local optimization package. This comparison will shed light on two topics:

- how far the solutions obtained by the local optimization package are from the global optimizer, and
- up to which problem size the global procedure developed can be applied in practice.

## Contributions

In this Thesis, we have

O1a.- described a new method for obtaining a d.c. representation of polynomials based on the fact that the set of $m^{th}$ powers of homogeneous polynomials of degree 1 is a generating set for the vector space of homogeneous polynomials of degree $m$,

O1b.- developed a procedure, using *MAPLE* Symbolic Calculator, which allows us to search for bases and to obtain a d.c. representation of the homogeneous components of a polynomial. Alternative bases in order to obtain a different d.c. representation of a polynomial can be used,

O2a.- described and written out a procedure in *FORTRAN* to convert the Generation Problem into an equivalent reverse convex programming problem expressed by:

$$\begin{array}{rc} \text{minimize} & f(x) - t \\ \text{subject to:} & g(x) - t \leq 0, \\ & h(x) - t \geq 0, \\ & Ax \leq b, \end{array}$$

where $A$ is a real $m \times n$ matrix, $b \in \mathbb{R}^m$ and $f(x),\ g(x)$ and $h(x)$ are convex functions on $\mathbb{R}^n$,

O2b.- described and written out an *adapted algorithm* in *FORTRAN* and in *C* by modification of the *combined outer approximation and cone splitting conical algorithm for canonical d.c. programming* from [74]. Since the above-mentioned programming problem is unbounded we use prismatical subdivisions instead of conical ones so that it is not necessary to find a subdivision vertex as in the case of conical subdivisions in [74]. Moreover, the adapted algorithm uses prismatic branch and select technique with polyhedral outer approximation subdivisions, in such a way that only linear programming problems have to be solved. To solve them, we have used the *MINOS* package in the case of the algorithm in *FORTRAN* and the *CPLEX* callable library in the case of the algorithm in *C*.

O2c.- established theoretically the convergence to a global optimizer of the adapted algorithm, O3a.- applied the concept of *least deviation decomposition* from [46] to obtain an optimal d.c. representation of a polynomial function in the normed space of the polynomials with the Euclidean norm, in order to improve the computational efficiency of our algorithm.

O3b.- described and written out an algorithm in *FORTRAN* by using an interior point method to solve semi-infinite quadratic programming problems with linear constraints to obtain these optimal d.c. representation (see [11], [37] and [78] for more details),

O4a.- used *MINOS* to check all gradients of the d.c. program instances,

O4b.- used *MINOS* as a local optimization package to compare its solutions with the solutions of the adapted algorithm.

It should be pointed out that the adapted algorithm is more general than it seems to be because it can be used to solve d.c. programming problems with convex constraints expressed by

$$\begin{array}{rl} \text{minimize} & f(x) - g(x) \\ \text{subject to:} & h_i(x) \leq 0, i = 1, \ldots, m, \end{array}$$

which, by introducing an additional variable $t$, can be transformed into an equivalent convex minimization problem subject to an additional reverse convex constraint in the form

$$\begin{array}{rl} \text{minimize} & f(x) - t \\ \text{subject to:} & g(x) - t \geq 0, \\ & h(x) \leq 0, \end{array}$$

with $h(x) = \max\{h_i(x) :, \; i = 1, \ldots, m\}$. This is a different way of solving a d.c. program with convex constraints than the algorithm proposed in [34] and [33] which transforms it into an equivalent concave minimization problem and uses prismatic branch and bound technique with polyhedral outer approximation subdivisions.

## Contents

Chapter 1 begins with a brief historical note about optimization methods, in which deterministic global optimization has a history of over thirty years. Then, we introduce the common general mathematical complementary convex structure underlying in global optimization problems. In this Thesis, this general mathematical complementary convex structure provides the foundation for reducing a nonconvex global optimization problem (for which a d.c. representation of both the objective function and constraints can be obtained) to a canonical form. Moreover, any global optimization method must address, directly or indirectly, the question of how to transcend a given feasible solution, which may be a local minimizer or an stationary point. It is this possibility to be trapped at an stationary point which causes the failure of local methods. Hence, various methods, which have been proposed to solve global optimization problems, are briefly mentioned.

In Chapter 2 we describe the Generation Problem, whose functions are d.c. functions because they are polynomials. Thus, by using the properties of the d.c. functions (see [27] and [32]) and the flow balance equations at all nodes of the replicated hydronetwork (which are the linear constraints of the Generation Problem), we describe the Generation Problem as an equivalent canonical d.c. programming problem of reduced size. It should be stressed that several transformations can be used to obtain an equivalent reverse convex program. From the structure of the functions in the Generation Problem, we rewrite it as a more suitable equivalent reverse convex program in order to obtain an advantageous adaptation of the *combined outer approximation and cone splitting conical algorithm for d.c. programming* as described in [74].

Chapter 3 introduces the concepts and properties, which allow us to obtain an explicit representation of a polynomial as a difference of convex polynomials (Corollary 3.3.4), based on the fact that the set of $m^{th}$ powers of homogeneous polynomials of degree 1 is a generating set for the vector space of homogeneous polynomials of degree $m$ (Proposition 3.3.2). Also, we compare our procedure to obtain a d.c. representation of a polynomial with the procedure described by Konno, Thach and Tuy [42], emphasizing its advantages and applying it to the polynomials of the Generation Problem. Moreover, we present a procedure, using $MAPLE$ Symbolic Calculator, which allows us to search for bases and to obtain a d.c. representation of the homogeneous components of a given polynomial.

Chapter 4 is devoted to describing the adapted global optimization algorithm and its basic operations. Moreover, we prove the convergence of the adapted algorithm by using a prismatical subdivision process together with an outer approximation procedure. The adapted global optimization algorithm is an advantageous adaptation of the combined outer approximation and cone splitting conical algorithm for d.c. programming in [74]. Since our equivalent reverse convex program is unbounded we use prismatical subdivisions instead of conical ones (as used in [74]). Hence, it is not necessary to find any subdivision vertex as in the case of conical subdivisions.

In Chapter 5, we announce *the minimal norm problem* by using the concept of *least deviation decomposition* (*LDD*) described in Luc D.T. et al [46] in order to obtain the optimal d.c. representation of a polynomial function, which allow us more efficient implementations by reducing the number of iterations of the adapted global optimization algorithm. The minimal norm problem can be transformed into an equivalent semidefinite program or into an equivalent semi-infinite quadratic programming problem with linear constraints. We discuss the suitability of use a quadratic semi-infinite algorithm in place of a semidefinite one.

Chapter 6 is devoted to describing a quadratic semi-infinite algorithm, which is an adaptation of the linear semi-infinite algorithm developed by J.Kaliski et al in [37], and its basic operations. We propose a build-up and build-down strategy, introduced by Den Hertog in [11] for standard linear programs that use a logarithmic barrier method. It should be pointed out that Chapters 5 and 6 are closely connected. They are presented separately for the sake of clarity.

Finally, in Chapter 7 computational results are given and conclusions are discussed.

# Chapter 1

# Overview and scope of global optimization

## 1.1 Historical notes

Modern techniques of optimization constitute a scientific discipline whose origins can be traced back to the birth of the digital computer. Optimization is concerned with the analysis of solutions and the development of procedures for solving problems of the form

$$\begin{array}{rl} \text{minimize} & f(x) \\ \text{subject to:} & g_i(x) \leq 0, i = 1, 2, \ldots, m \\ & x \in X \subset I\!\!R^n, \end{array} \tag{1.1}$$

where $f(x)$ and $g_i(x), i = 1, 2, \ldots, m$, are real-valued functions defined on a domain $X \subset I\!\!R^n$. Hence, $f(x)$ is the objective function which measures the quality of the solution, $S := \{x \in X : g_i(x) \leq 0, i = 1, 2, \ldots, m\}$ is the set of feasible points (or feasible domain), and $x \in S$ are said to be the decision variables. If $m = 0$ and $S$ defines an hyperrectangle of $I\!\!R^n$, the problem is said to be unconstrained; if $n = 1$ the problem is univariate otherwise the problem is said to be multivariate. *Linear programming* deals with optimization problems where $f(x)$ is a linear function and the feasible domain $S$ is a set defined by linear inequalities. During World War II, George B. Dantzig developed the simplex method for solving linear programming models of logistics and operational military problems. At the same time, and independently, Leonid V. Kantorovich developed the method of resolving multipliers for linear programs and applied them to problems such as equipment work distribution. The simplex method consists of the selection of an optimizer within the finite set of vertices of the convex polytope defined by the linear constraints. A programming problem is said to be *combinatorial* when an optimizer must be sought within a finite set of candidate points. *Integer programming* imposes additional integrality constraints on some subsets of the decision variables. Almost all combinatorial optimization problems can be modelled as integer programs. Methods such as cutting-plane, branch and bound, branch and cut, column generation, decomposition techniques and polyhedral combinatorics have served as powerful tools for solving integer programming problems. However, because of the complexity (NP-hard, etc) of these problems, many instances cannot be solved exactly in a polynomial time. Methods which find approximate optimizers (suboptimal solutions) to these instances have been developed in recent times. *Heuristics*, such as genetic algorithms by Holland in 1975 [29], simulated annealing by Kirkpatrick et al. in 1983 [41], GRASP by Feo and Resende in 1995 [15], Tabu search by Glover and Laguna in 1997 [22], find good quality approximate optimizers in reasonable computational times. *Network optimization* is another important field of combinatorial optimization. Many early network algorithms, such as minimum spanning tree by Kruskal in 1956 [43], Prim in 1957 [54] and shortest-path programs by Dijkstra in 1959 [12] are still used today. In 1962 the book Flows in networks by Ford and Fulkerson [19] appeared. Recent data structure developments have contributed to the use of network algorithms for solving large real-world optimization problems.

In *nonlinear optimization* the constraints, which define the feasible domain $S$, and/or the objective function $f(x)$ are nonlinear functions. In the early 1960's, J.B. Rosen published the *gradient-projection* method for nonlinear programming (see [58] and [59]) which instigated further research into algorithms for nonlinear optimization. The interplay between continuous nonlinear optimization and combinatorial optimization motivated the development of new algorithmic techniques for large-scale problems. For instance, the field of *interior point* methods uses nonlinear programming for solving linear programs. In 1979 Leonid Khachiyan published the ellipsoid method [39], the first polynomial time algorithm for solving linear programs which come from nonlinear programming was published in 1970 by Shor ([67] and [66]), and it is drastically different from most previous approaches to linear programming. In 1984 Narendra Karmarkar published a polynomial time interior point algorithm for linear programming [38], which is the origin of modern interior point methods. Variants of interior point methods were shown to perform well in practice and they extended the limits of the dimension of problems that could be solved.

When a problem is convex, it is well known that every local minimizer is global. In many practical problems in which the convexity of the objective function or the

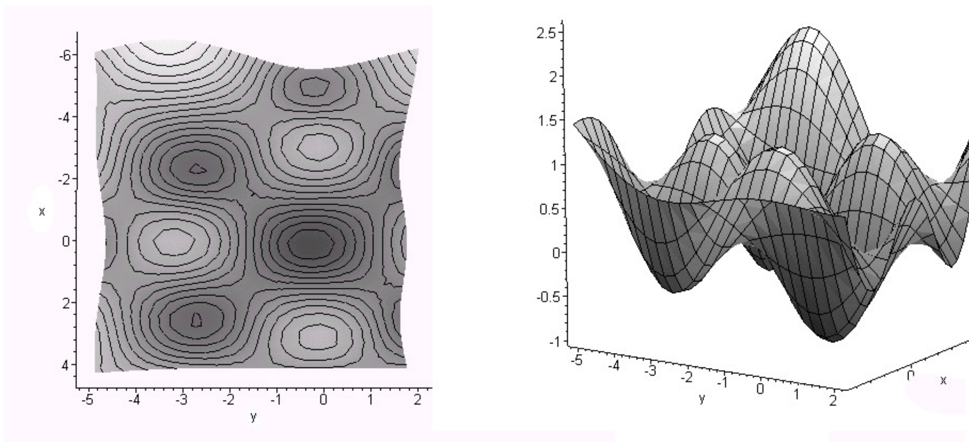

Figure 1.1: $f(x,y) := \frac{3}{100}(x^2 + y^2) - cos(x)cos(y)$

constraints cannot be verified, it is reasonable to assume that the problems have multiple local optimizers (multiextremal problems, see Figure 1.1). *Global optimization* deals with the computation and characterization of global optimums of multiextremal problems and it aims at solving the very general program (1.1). Any global optimization method must address the question of *how to transcend a given feasible point, if there is one, or else how to produce evidence that the given point is already globally optimal.* Global optimization techniques are substantially different from local ones and, among others, employ combinatorial tools such as cutting-plane, branch and bound, branch and cut and so on. Most people considered until the mid 1980's that heuristic and stochastic local searches were more practical and reliable approaches for solving these inherently difficult classes of problems. In recent years, despite the inherent difficulty of global optimization, remarkable developments in this field have been made. The emergency of powerful workstations enabled one to solve a number of small to medium size global optimization problems by *general purpose deterministic algorithms* in a practical amount of time. They have been applied to some important classes of problems such as concave minimization, reverse convex programs, d.c. programs and Lipschitz optimization. Unfortunately we usually observe a rapid increase in computation time as the size of the instance increases if it has no special structure. The first textbook on this subject was published in 1990 by Horst and Tuy [32]. In this textbook the authors discussed the overall theoretical framework and general purpose deterministic algorithms for locating a global optimum of a multiextremal problem. The last years also witnessed the emergence of the *Journal of Global Optimization* as well as the increase in research activities with several specialized conferences on global optimization and applications.

## 1.2 Classification of global optimization methods

As we have seen before any global optimization method must address the question of how to transcend a given feasible point or else to produce evidence that the given point is already globally optimal. Numerical methods that have been proposed can be classified into three categories by distinguishing the available guarantees: heuristic methods, approximation methods, and systematic methods, which include deterministic methods.

### 1.2.1 Heuristic methods

Heuristic methods are used for solving global optimization problems whose structure is difficult to analyze. They contain all methods in which no theoretical justification can be established to find a global optimizer. In these methods, we have some grounds for believing that the feasible point obtained is sufficiently near the optimum or that there is a high probability of it being so. Among the most popular are:

- *Multi-start methods* which are intended to locate as many local minimizers as possible by applying standard local optimization algorithms initiated at multiple starting points which are generated stochastically.
- *Simulated annealing methods* which are intended to transcend a local solution by using some probabilistic procedure which may accept a temporary increase in the objective function.
- *Genetic algorithms* where the optimizer is searched by a procedure imitating the natural genetic selection of the fittest individuals.

Also, heuristic algorithms of *tabu search* type have been used to solve global optimization problems difficult to tackle otherwise. In contrast to deterministic methods, heuristic methods offer almost no guarantee of global optimality for the obtained solution and hardly provide any information about how far this solution could be from the optimum.

### 1.2.2 Approximation methods

Approximation methods transform the original problem by means of suitable approximations into a simpler global optimization problem that is more tractable. Solving the approximate problem yields an approximate solution for the original problem, and local optimization from this approximate solution often gives the global optimizer of the original problem provided that the approximation was good enough or a good local minimizer otherwise. Of great practical importance is the approximation by the *mixed integer linear programs* (MILP).

### 1.2.3 Systematic methods

When only black box function evaluation routines are available, we can use some kind of systematic methods (black box methods) which will usually find a global optimizer with certainty in a finite time, but we can never know when this is the case. Thus, for systematic black box methods, stopping must be based on heuristic recipes. On the other hand, deterministic methods contain all methods that theoretically guarantee a global optimizer via a predictable amount of work. Predictable means relative to known problem characteristics such as a d.c. structure, Lipschitz constants or other global information needed for the convergence proof. These methods include, among others, cutting plane, successive approximation and partitioning methods together with concepts such as concavity cuts, outer approximation schemes and branch and bound techniques. However, a weakness of these methods is that for problems without any particular structure, they can only solve, in practice, problem instances of limited size. The amount of work is usually exponential with respect to the problem characteristics. Sometimes, heuristic and probabilistic choices also play a role in systematic methods, mainly to provide cheaply a good local optimizer which benefits the systematic search.

We list some systematic global optimization codes with short comments on scope and method.

The codes listed use black box function evaluation routines, and have heuristic stopping rules.

**DIRECT (Divide Rectangles).** It is a branching code in *FORTRAN* which use function values only, by Gablonsky and Kelley (2001). *DIRECT* is based on branching and a Pareto principle for box selection (see [20]). http://www4.ncsu.edu/∼jmgablon/.

**MCS (Multilevel Coordinate Search).** A *MATLAB* program for bound constrained global optimization by Huyer and Neumaier (1999). *MCS* uses function values only and is based on branching and sequential quadratic programming (see [35]). http://solon.cma.univie.ac.at/∼neum/software/mcs/.

**LGO (Lipschitz Global Optimization).** An integrated development environment for global optimization with Lipschitz continuous objective function and constraints by Janos Pintér (1996). *LGO* is based on branching and estimation of Lipschitz constants, constraints other than simple bounds are handled by $L_1$ penalty terms and interior convex constraints by projection penalties (see [52]). http://is.dal.ca/∼jdpinter/lgoide.html/.

The following codes use global information and stop in a finite time with the guarantee that the global minimizer is found. In difficult cases storage or time limits may be exceeded, however, leading to appropriate error messages.

**BARON (Branch and Reduce Optimization Navigator).** A general purpose solver for optimization problems with nonlinear constraints and/or integer variables by Ryoo and Sahinidis (1996). *BARON* is based on branching and box reduction using convex relaxation and Lagrange multiplier techniques (see [62]). http://archimedes.scs.uiuc.edu/baron/baron.html.

**$\alpha$BB (alpha Branch and Bound).** A branch and bound code for nonlinear programs by Androulakis, Maranas and Floudas (1995). It is based on branching and bound by convex underestimation using interval analysis to write nonlinearities in difference of convex functions form (see [3]). http://titan.princeton.edu/soft.html#abb. The site contains a description only, no code.

Recently, the **Cutting Angle Method** of Andramonov, Rubinov et al. (see [2] and [61]) has been developed for solving global optimization problems. A new version of this method can be found in [5]. In this method the objective function $f(x)$ is a Lipschitz function and the feasible domain $S$ is the unit simplex. Many problems of unconstrained and constrained optimization can be reduced to this form by a transformation of variables and penalization.

In this Thesis we are interested in deterministic global optimization methods which rely on a d.c. structure of a d.c. program. These methods have been proved to be particularly successful for analyzing and solving a variety of highly structured problems (see [30] and [51]).

## 1.3 General complementary convex mathematical structure

A general problem such as (1.1) does not seem to have any mathematical structure receptive to computational implementations. However, by taking into account the fact that every point in $I\!R^n$ has a base of convex neighborhoods, when $f(x)$ is a continuous function and $S$ is a nonempty closed set the problem (1.1) can be transformed into an equivalent problem with a linear objective function and a complementary convex mathematical structure (or d.c. structure) expressed by

$$\text{minimize } l(z) \text{ subject to: } z \in D \setminus \text{int } C, \tag{1.2}$$

where $l(z)$ is a linear function and $D$ and $C$ are closed convex sets. The program (1.2) is said to be a *canonical d.c. program.* For the convenience of making the exposition self-contained we quote some results with proofs from Tuy [74].

**Proposition 1.3.1** *Let $S \subset I\!R^n$ be a closed convex set. Then for every $y \in I\!R^n \setminus S$, let $B(y, \delta_y) \subset I\!R^n \setminus S$ be a ball with center $y$ and radius $\delta_y > 0$. The following equalities are true:*

$$S = \bigcap_{y \notin S} I\!R^n \setminus B(y, \delta_y) = \{x \in I\!R^n : g(x) - \|x\|^2 \leq 0\},$$

*where $g(x) :=$ sup $\{2xy - \|y\|^2 + \delta_y^2,\ y \notin S\}$ is a convex lower semi-continuous function ( l.s.c.) (see [56]).*

PROOF: For all $y \notin S$ there exists a ball $B(y, \delta_y)$ of radius $\delta_y > 0$ satisfying $B(y, \delta_y) \subset I\!R^n \setminus S$ or equivalently $S \subset I\!R^n \setminus B(y, \delta_y)$. Then $S \subset \cap_{y \notin S} I\!R^n \setminus B(y, \delta_y)$. Moreover, if $x \in \cap_{y \notin S} I\!R^n \setminus B(y, \delta_y)$ and $x \notin S$, then $x \notin B(x, \delta_x)$ which is a contradiction. So $\cap_{y \notin S} I\!R^n \setminus B(y, \delta_y) \subset S$.

On the other hand, $x \in S \Leftrightarrow \|y - x\| \geq \delta_y$, for all $y \notin S$. Then, we have

$$\begin{array}{lcll} \|y-x\|^2 & = & \|y\|^2 + \|x\|^2 - 2xy \geq \delta_y^2, \text{ for all } y \notin S & \text{followed by} \\ \|x\|^2 & \geq & 2xy - \|y\|^2 + \delta_y^2, \text{ for all } y \notin S & \text{and} \\ \|x\|^2 & \geq & g(x) = \sup \{2xy - \|y\|^2 + \delta_y^2, y \notin S\}, & \end{array}$$

where $g(x) :=$ sup $\{2xy - \|y\|^2 + \delta_y^2,\ y \notin S\}$ is the pointwise supremum of a family of affine functions. Thus, $x \in S \Leftrightarrow g(x) - \|x\|^2 \leq 0$. □

**Proposition 1.3.2** *Let $S \subset I\!\!R^n$ be a closed set. There exist closed convex sets $D$ and $C$ into $I\!\!R^{n+1}$ such that $S$ is the projection of $D \setminus int\ C$ on $I\!\!R^n$, i.e.,*

$$S = proj_{\boldsymbol{R}^n}(D \setminus int\ C).$$

Proof: Using the results of the Proposition 1.3.1 we have that

$$g(x) - \|x\|^2 \leq 0 \Leftrightarrow \text{ there exists } t \in I\!\!R \text{ with } g(x) - t \leq 0 \text{ and } \|x\|^2 - t \geq 0.$$

Thus, $D = \{(x,t) \in I\!\!R^{n+1} :\ g(x) - t \leq 0\}$ and $C = \{(x,t) \in I\!\!R^{n+1} :\ \|x\|^2 - t \leq 0\}$ are closed convex sets and verify $S = \text{proj}_{\boldsymbol{R}^n}(D \setminus \text{int } C)$. □

**Proposition 1.3.3** *Problem (1.1), with continuous objective function and closed feasible set, can be transformed into an equivalent canonical d.c. program expressed by*

$$\textit{minimize } l(z) \textit{ subject to: } z \in D \setminus int\ C,$$

*where $l(z)$ is a linear function and $D$ and $C$ are closed convex sets.*

Proof: The program (1.1) can be expressed by the equivalent programming problem

$$\text{minimize } t \text{ subject to: } f(x) - t \leq 0,\ x \in S, \tag{1.3}$$

where $\{(x,t) \in I\!\!R^{n+1} :\ f(x) - t \leq 0, x \in S\}$ is a closed set. From Proposition 1.3.2 $\{(x,t) \in I\!\!R^{n+1} :\ f(x) - t \leq 0, x \in S\} = \text{proj}_{\boldsymbol{R}^{n+1}}(D \setminus \text{int } C)$ with $D$ and $C$ closed convex sets into $I\!\!R^{n+2}$. We obtain the expression (1.2) by setting $z = (x,t)$. □

Hence, we can see that every continuous global optimization problem can be expressed by an equivalent canonical d.c. program.

A canonical d.c. program is said to be *regular* when the feasible set satisfies the property $D \setminus \text{int } C = \text{cl } (D \setminus C)$. Also, a canonical d.c. program is said to be *essentially nonconvex* when it satisfies the inequality

$$\min\{l(x) : x \in D\} < \min\{l(x) : x \in D \setminus \text{int } C\}. \tag{1.4}$$

Denote

$$D(\gamma) := \{x \in D : l(x) \leq \gamma\} \text{ with } \gamma \in I\!\!R.$$

Assume the problem (1.2) to be regular and essentially nonconvex. Let $\bar{x} \in D \setminus \text{int } C$ be a feasible point and set $\bar{\gamma} = l(\bar{x})$. Then, we can announce the propositions

**Proposition 1.3.4** *If $\bar{x} \in D \setminus int\ C$ is a global optimizer of (1.2) then $\bar{x} \in \partial C \cap \partial D$.*

PROOF: Suppose that $\bar{x} \notin \partial C$, then $\bar{x} \notin C$ and $\bar{x} \in D \setminus C = D \cap \mathbb{R}^n \setminus C$ that is an open set by the topology induced in $D$ from the topology in $\mathbb{R}^n$. Hence, there exists a ball $B(\bar{x}, \delta)$ such that $B(\bar{x}, \delta) \cap D \subset D \setminus C$ and if $0 < t \leq \delta/\|\bar{x} - w\| < 1$. From the convexity of the function $l(x)$ every point $z = (1-t)\bar{x} + tw$ satisfies

$$l(z) \leq (1-t)l(\bar{x}) + tl(w) = l(\bar{x}) + t(l(w) - l(\bar{x})) < l(\bar{x}),$$

which is a contradiction. On the other hand, suppose that $\bar{x} \notin \partial D$. From $\bar{x} \in \partial C$ every ball $B(\bar{x}, \delta)$ satisfies

$$B(\bar{x}, \delta) \cap D(\gamma) \cap \text{int } D \neq \emptyset \text{ and } B(\bar{x}, \delta) \cap D(\gamma) \cap \text{ext } D \neq \emptyset.$$

Consider $z_0 \in B(\bar{x}, \delta) \cap D(\gamma) \cap \text{ext } D$. Then every point $z = (1-t)z_0 + tw$ with $0 \leq t \leq 1$ satisfies

$$l(z) \leq (1-t)l(z_0) + tl(w) \leq (1-t)l(\bar{x}) + tl(w) = l(\bar{x}) + t(l(w) - l(\bar{x})) < l(\bar{x}),$$

by using the convexity of the function $l(x)$. Hence, as $z_0 \in D \setminus C$ and $w \in D \cap C$ there exists $\tilde{z} \in \partial C \cap [w, z_0]$ satisfying $l(\tilde{z}) < l(\bar{x})$ which is a contradiction. □

**Proposition 1.3.5** *The following assertions are true:*

*i) if $D(\bar{\gamma}) \setminus C \neq \emptyset$, then there exists a feasible point strictly better than $\bar{x}$ lying on $\partial C \cap D$,*

*ii) $\bar{x}$ is a global optimizer of (1.2) $\Leftrightarrow$ $D(\bar{\gamma}) \setminus C = \emptyset$ (or $D(\bar{\gamma}) \subset C$).*

PROOF: ii)
$\Rightarrow$(necessary)
Let $x^0 \in D(\bar{\gamma}) \setminus C$. Then, $l(x^0) = l(\bar{x}) = \bar{\gamma}$ so that $x^0$ is an optimal global solution of (1.2). Thus, $x^0 \in \partial C$ (i.e. $x^0 \in C$) is a contradiction.
$\Leftarrow$(sufficient)
Let $x^* \in D \setminus \text{int } C$ be a better feasible solution than $\bar{x}$. Then, $l(x^*) < l(\bar{x})$ and we can find an open ball $B := B(x^*, \delta)$, of center $x^*$ and radius $\delta$, such that for all $x \in B$ we have 1) $l(x) < l(\bar{x})$. On the other hand, from $x^* \in D \setminus \text{int } C = \text{cl}(D \setminus C)$ it follows that $B \cap (D \setminus C) \neq \emptyset$ so every point $x \in B \cap (D \setminus C)$ verifies 2) $x \in D$ and $x \notin C$. Thus, from 1) and 2) we have $x \in (D(\bar{\gamma}) \setminus C) = \emptyset$ which is a contradiction.

□

## 1.4 Classes of nonconvex problems with a d.c. structure

In many classes of global optimization problems, convexity is present in a reverse sense. In this circumstance we have minimization of concave functions subject to convex constraints (concave minimization), minimization of convex functions on feasible domains which are the intersection of convex sets and complements of convex sets (reverse convex programming) and global optimization of functions which can be expressed as a difference of two convex functions (d.c. programming).

### 1.4.1 Concave minimization

A program expressed by

$$\text{minimize } f(x) \text{ subject to: } x \in S, \tag{1.5}$$

where $f(x)$ is a concave function on $I\!R^n$ and $S \subset I\!R^n$ is a nonempty closed convex set, is said to be a *concave minimization program.* This is the simplest class of global optimization problems with a d.c. structure. By defining the closed convex sets $D := \{(x,t) \in I\!R^{n+1} : x \in S\}$ and $C := \{(x,t) \in I\!R^{n+1} : f(x) \leq t\}$, it is equivalent to the canonical d.c. program expressed by

$$\text{minimize } t \text{ subject to : } (x,t) \in D \setminus \operatorname{int} C. \tag{1.6}$$

### 1.4.2 Reverse convex programming

A program expressed by

$$\text{minimize } f(x) \text{ subject to: } g(x) \leq 0,\ \ h(x) \geq 0 \tag{1.7}$$

where $f(x)$, $g(x)$ and $h(x)$ are convex functions on $I\!R^n$ is said to be a *reverse convex program.* This differs from a convex program only by the presence of the constraint $h(x) \geq 0$, which is said to be a *reverse convex constraint.* The reverse convex program (1.7) is equivalent to the canonical d.c. program expressed by

$$\text{minimize } t \text{ subject to: } (x,t) \in D \setminus \operatorname{int} C, \tag{1.8}$$

where $D$ and $C$ are closed convex sets into $I\!R^{n+1}$ defined by

$$D := \{(x,t) \in I\!R^{n+1} : f(x) - t \leq 0,\ \ g(x) \leq 0\} \text{ and } C := \{(x,t) \in I\!R^{n+1} : h(x) \leq 0\}.$$

### 1.4.3 D.c. programming

A program expressed by

$$\text{minimize } f(x) \text{ subject to: } x \in S,\ h_i(x) \le 0,\ i = 1, \dots, m, \tag{1.9}$$

is said to be a *d.c. program* when $S$ is a closed convex set into $I\!R^n$ and the functions $f(x)$, $h_i(x) \le 0,\ i = 1, \dots, m$ are d.c. functions and they can be explicitly expressed as a difference of two convex functions on $I\!R^n$. By introducing the additional variable $t \in I\!R$ the program (1.9) can be transformed to the equivalent d.c. program

$$\text{minimize } t \text{ subject to: } x \in S,\ f(x) - t \le 0,\ h_i(x) \le 0,\ i = 1, \dots, m, \tag{1.10}$$

with linear objective function. Furthermore, the d.c. inequalities

$$f(x) - t \le 0,\ h_i(x) \le 0,\ i = 1, \dots, m,$$

can be replaced by a single d.c. inequality

$$r(x,t) := \max\ \{f(x) - t \le 0,\ h_i(x) \le 0,\ i = 1, \dots, m\} \le 0.$$

Suppose that for each d.c. function $f(x)$, $h_i(x) \le 0,\ i = 1, \dots, m$ a d.c. representation is known. Then, by using the properties of the d.c. functions (see Preliminaries) we can write $r(x,t) = p(x,t) - q(x,t)$, where $p(x,t)$ and $q(x,t)$ are convex functions. Hence, by introducing the new additional variable $z \in I\!R$, the d.c. inequality

$$r(x,t) = p(x,t) - q(x,t) \le 0$$

is equivalent to the system

$$p(x,t) - z \le 0,\ q(x,t) - z \ge 0,$$

where the first inequality is convex and the second is said to be *reverse convex.* Finally, setting

$$D := \{(x,t,z) \in I\!R^{n+2} : p(x,t) - z \le 0,\ x \in S\}$$

and

$$C := \{(x,t,z) \in I\!R^{n+2} : q(x,t) - z \le 0\},$$

we can see that the d.c. program (1.9) is equivalent to the canonical d.c. program

$$\text{minimize } t \ \text{ subject to: } (x,t,z) \in D \setminus \operatorname{int} C$$

### 1.4.4 Continuous optimization

It has been proved in Proposition 1.3.3 that the problem of minimizing a continuous function over a closed set can be reduced to a canonical d.c. program. This is a theoretical result and, in any case, continuous optimization problems are the most difficult to solve from the viewpoint of the complementary mathematical convex structure.

As above-mentioned, in this Thesis we are interested in deterministic global optimization methods which rely on a d.c. structure of a d.c. program. The main question is how to find a good d.c. structure of a given d.c. program. We will solve this problem when the functions of the d.c. program are polynomial functions.

# Chapter 2

# The Generation Problem and its complementary convex structure

## 2.1 The short-term hydrothermal coordination of electricity generation problem

We want to apply deterministic global optimization procedures to the problem of the short-term hydrothermal coordination of the electricity generation (Heredia and Nabona [26]). Its importance stems from the economic and technical implications that the solution to this problem has for electric utilities with a mixed, hydro and thermal, generation system.

Given a short-term time period subdivided into time intervals the aim is to find values of hydro and thermal generation for each time interval in the period so that the demand of electricity is met for each time interval, a number of constraints are satisfied, and the generation cost of thermal units is minimized. The model contains the replicated hydronetwork through which the temporary evolution of the reservoir system is represented (Figure 2.1 shows us the replicated hydronetwork with only two reservoirs and where the time period has been subdivided into four time intervals). We use $N_e$ to indicate the number of reservoirs, $N_t$ to indicate the number of time intervals, $j$ to indicate the $j^{\text{th}}$ reservoir, $j = 1 \dots N_e$ and $i$ to indicate the $i^{\text{th}}$ time interval, $i = 1 \dots N_t$. It should be observed that,

- the *variables* are the *water discharges* $d_j^i$ from reservoir $j$ over the $i^{\text{th}}$ interval and the *volume stored* $v_j^i$ in reservoir $j$ at the end of the $i^{\text{th}}$ time interval,

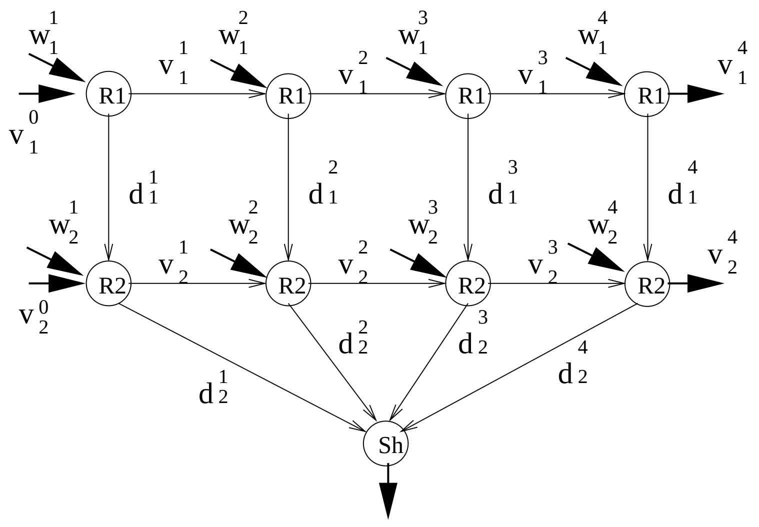

Figure 2.1: Four intervals and two reservoirs replicated hydronetwork

- in each time interval $i$, the water discharge from reservoir $R_1$ to reservoir $R_2$ establishes a link between the reservoirs,

- the volume stored at the end of the time interval $i$ and the volume stored at the beginning of the time interval $i+1$ are the same in each reservoir $R_j$, which establishes a link between each reservoir from the time interval $i$ to $i+1$,

- the volumes stored at the beginning and at the end of the time period, $v_j^0$ and $v_j^{N_t}$, $j = 1 \dots N_e$ respectively, are known (they are not variables). Acceptable forecasts for electricity consumption $l^i$, $i = 1 \dots N_t$ and for natural water inflow $w_j^i$, $i = 1 \dots N_t$, $j = 1 \dots N_e$ into the reservoirs of the hydrogeneration system must be available at each time interval,

- the electrical power generated at each reservoir $j$ in each time interval $i$ depends on the initial and final volumes, $v_j^{i-1}$ and $v_j^i$ respectively, and the volume of the water discharge $d_j^i$. This dependence will be expressed by the notation $h_j^i(v_j^{i-1}, v_j^i, d_j^i)$ and will be called the *power hydrogeneration function* of the $j^{\text{th}}$ reservoir over the $i^{\text{th}}$ time interval.

### 2.1.1 The power hydrogeneration function in a reservoir

At each reservoir $j$ the *head* $\mathsf{s}_j$ depends on the *headwater elevation* $s_w^j$ and the *tailwater elevation* $\bar{s}_w^j$. The latter varies as a function of the water discharge $d_j$.

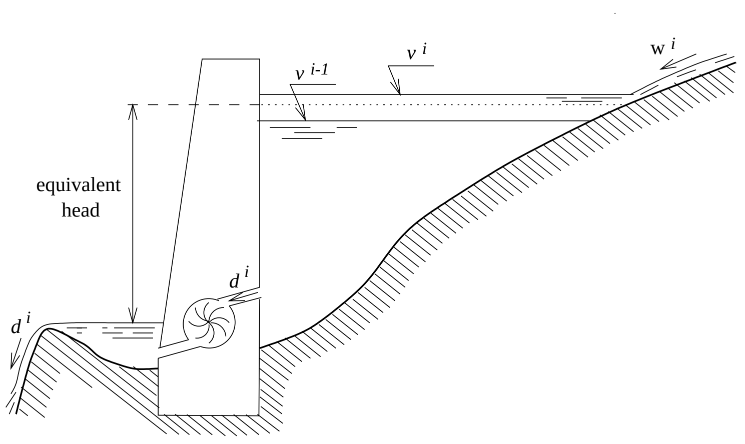

Figure 2.2: Cross-section of a reservoir

For zero discharge, the *downstream pool* is flat and the head becomes equal to the headwater elevation, so we can write

$$\mathsf{s}_j = s_w^j - \bar{s}_w^j. \tag{2.1}$$

The relationship between the headwater elevation $s_w^j$ and the volume stored $v_j$ can be expressed approximately by the polynomial

$$s_w^j = s_{vb} + s_{vl}v_j + s_{vq}v_j^2 + s_{vc}v_j^3, \tag{2.2}$$

where $s_{vb}$, $s_{vl}$, $s_{vq}$ and $s_{vc}$ are the basic, linear, quadratic and cubic technological coefficients respectively, which depend on each reservoir and need to be known. Also, the relationship between the tailwater elevation $\bar{s}_w$ and the water discharge $d_j$ can be expressed in the form

$$\bar{s}_w = s_{dl}d_j + s_{dq}d_j^2, \tag{2.3}$$

where $s_{dl}$ and $s_{dq}$ are technological coefficients similar to before. Hence, the head $\mathsf{s}_j$ depends on the volume stored $v_j$ and the water discharge $d_j$, and it can be written

$$\mathsf{s}_j(v_j, d_j) = s_{vb} + s_{vl}v_j + s_{vq}v_j^2 + s_{vc}v_j^3 - s_{dl}d_j - s_{dq}d_j^2. \tag{2.4}$$

The relationship between the variation in potential energy $\mathsf{d}E_p$ and the variation in the water stored $\mathsf{d}v_j$ can be expressed approximately by

$$\mathsf{d}E_p = g\mathsf{s}_j(v_j, d_j)\mathsf{d}v_j,$$

where $g$ is the *gravity*. Thus, the potential energy $E_p$ in the $i^{\text{th}}$ time interval, corresponding to the initial and final volumes $v_j^{i-1}$ and $v_j^i$ respectively, can be obtained by calculating

$$E_p = \int_{v_j^{i-1}}^{v_j^i} \mathsf{d}E_p = g \int_{v_j^{i-1}}^{v_j^i} \mathsf{s}_j(v_j, d_j)\mathsf{d}v_j. \tag{2.5}$$

Define the *equivalent head* $\tilde{s}_j$ by the expression

$$E_p = \tilde{s}_j g(v_j^i - v_j^{i-1}). \tag{2.6}$$

From the expressions (2.5) and (2.6) the equivalent head $\tilde{s}_j$ can be obtained in function of the initial volume $v_j^{i-1}$, the final volume $v_j^i$ and the volume of the water discharge $d_j^i$, by evaluating the expression

$$\tilde{s}_j = \frac{1}{v_j^i - v_j^{i-1}} \int_{v_j^{i-1}}^{v_j^i} \mathsf{s}_j(v_j, d_j)\mathsf{d}v_j. \tag{2.7}$$

Hence, after carrying out the necessary calculations we obtain

$$\begin{aligned} \tilde{s}_j \; = \; & s_{vb} + \tfrac{s_{vl}}{2}(v_j^{i-1} + v_j^i) + \tfrac{s_{vq}}{3}(v_j^i - v_j^{i-1})^2 + s_{vq}v_j^{i-1}v_j^i + \\ & + \tfrac{s_{vc}}{4}((v_j^{i-1})^2 + (v_j^i)^2)(v_j^{i-1} + v_j^i) - s_{dl}d_j^i - s_{dq}(d_j^i)^2. \end{aligned} \tag{2.8}$$

The main feature of this formulation is that the power hydrogeneration function is expressed approximately by a polynomial function. The power hydrogeneration function $h_j^i$ of the $j^{\text{th}}$ reservoir in the $i^{\text{th}}$ time interval can be obtained by considering that the energy transferred to the *turbine* by the water discharge is approximately equivalent to the potential energy of the weight of the water discharge $d_j^i$ from the equivalent head $\tilde{s_j}$, i.e., the energy transferred is $\tilde{s}_j g d_j^i$. Then, the power hydrogeneration function $h_j^i$ of the $j^{\text{th}}$ reservoir in the $i^{\text{th}}$ time interval can be expressed by

$$h_j^i = \delta \rho_j^i \frac{g d_j^i \tilde{s}_j}{t^i}, \tag{2.9}$$

where $\delta$ is the *unit conversion coefficient*, $0 \leq \rho_j^i \leq 1$ is the *efficiency coefficient*, which is a concave function of the water discharge $d_j^i$ (during the time interval $t^i$) and the equivalent head $\tilde{s}_j$. Finally, after substituting (2.8) in (2.9) we have the new expression of the power hydrogeneration function

$$\begin{aligned} h_j^i(v_j^{i-1}, v_j^i, d_j^i) \;\cong\; & k_j^i \tfrac{d_j^i}{t^i}[s_{vb} + \tfrac{s_{vl}}{2}(v_j^{i-1} + v_j^i) + \tfrac{s_{vq}}{3}(v_j^i - v_j^{i-1})^2 + \\ & + s_{vq}v_j^{i-1}v_j^i + \tfrac{s_{vc}}{4}((v_j^{i-1})^2 + (v_j^i)^2)(v_j^{i-1} + v_j^i) - \\ & - s_{dl}d_j^i - s_{dq}(d_j^i)^2], \end{aligned} \tag{2.10}$$

where $k_j^i := \delta\rho_j^i g$ is the *efficiency and unit conversion coefficient.*

### 2.1.2 The Generation Problem

The *objective function*, which will be minimized, is the generation cost function of thermal units,

$$\sum_{i=1}^{N_t} c^i \left( l^i - \sum_{j=1}^{N_e} h^i_j(v^{i-1}_j, v^i_j, d^i_j) \right), \tag{2.11}$$

where $l^i$, $i = 1, \ldots, N_t$, are the forecast for electricity consumption in the $i^{\text{th}}$ time interval. The *linear constraints* are the flow balance equations at all nodes of the network,

$$v^i_j - v^{i-1}_j - d^i_{j-1} + d^i_j = w^i_j \quad j = 1, ..., N_e, \ i = 1, ..., N_t, \tag{2.12}$$

the *nonlinear constraints* are the thermal production with generation bounds,

$$\underline{g} \le l^i - \sum_{j=1}^{N_e} h^i_j(v^{i-1}_j, v^i_j, d^i_j) \le \overline{g} \quad i = 1, ..., N_t \tag{2.13}$$

and there are positive bounds on all variables,

$$\underline{d_j} \le d^i_j \le \overline{d_j} \qquad j = 1, ..., N_e, \ i = 1, ..., N_t \tag{2.14a}$$

$$\underline{v_j} \le v^i_j \le \overline{v_j} \quad j = 1, ..., N_e, \ i = 1, ..., N_t - 1. \tag{2.14b}$$

Hence, the problem can be expressed by the program

$$\boxed{\begin{array}{rll} \text{minimize} & \sum_{i=1}^{N_t} c^i \left( l^i - \sum_{j=1}^{N_e} h^i_j(v^{i-1}_j, v^i_j, d^i_j) \right) & \\ \text{subject to:} & \underline{g} \le l^i - \sum_{j=1}^{N_e} h^i_j(v^{i-1}_j, v^i_j, d^i_j) \le \overline{g}, & i = 1, ..., N_t, \\ & v^i_j - v^{i-1}_j - d^i_{j-1} + d^i_j = w^i_j, & j = 1, ..., N_e, \\ & & i = 1, ..., N_t \\ & \underline{d_j} \le d^i_j \le \overline{d_j}, & j = 1, ..., N_e, \\ & & i = 1, ..., N_t \\ & \underline{v_j} \le v^i_j \le \overline{v_j}, & j = 1, ..., N_e. \\ & & i = 1, ..., N_t - 1 \end{array}} \tag{2.15}$$

## 2.2 The Generation Problem as a d.c. program

A polynomial is a d.c. function on $\mathbb{R}^n$ because every function whose second partial derivatives are continuous on $\mathbb{R}^n$ is a d.c. function on $\mathbb{R}^n$. Let

$$h^i_j(v^{i-1}_j, v^i_j, d^i_j) = f^i_j(v^{i-1}_j, v^i_j, d^i_j) - g^i_j(v^{i1}_j, v^i_j, d^i_j), \tag{2.16}$$

be a d.c. representation of the power hydrogeneration function, where $f_j^i(v_j^{i-1}, v_j^i, d_j^i)$ and $g_j^i(v_j^{i1}, v_j^i, d_j^i)$ are convex functions defined on a convex set which contains the feasible domain of the program (2.15). Then, by defining the convex functions

$$F^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) = c^i \left( l^i + \sum_{j=1}^{N_e} g_j^i(v_j^{i-1}, v_j^i, d_j^i) \right), \; i = 1, ..., N_t, \tag{2.17}$$

$$G^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) = c^i \sum_{j=1}^{N_e} f_j^i(v_j^{i-1}, v_j^i, d_j^i), \; i = 1, ..., N_t \tag{2.18}$$

$$F(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) = \sum_{i=1}^{N_t} F^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) \tag{2.19}$$

and

$$G(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) = \sum_{i=1}^{N_t} G^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots), \tag{2.20}$$

a d.c. representation of all functions within (2.15) can be obtained.

### 2.2.1 D.c. representation of the nonlinear constraints

By using the expressions (2.17) and (2.18) in

$$l^i - \sum_{j=1}^{N_e} h_j^i(v_j^{i-1}, v_j^i, d_j^i) = \left( l^i + \sum_{j=1}^{N_e} g_j^i(v_j^{i-1}, v_j^i, d_j^i) \right) - \sum_{j=1}^{N_e} f_j^i(v_j^{i-1}, v_j^i, d_j^i), \tag{2.21}$$

a d.c. representation of the nonlinear constraints of the program (2.15) can be obtained:

$$c^i \underline{g} \le F^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) - G^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) \le c^i \overline{g}, \; i = 1, \ldots, N_t. \tag{2.22}$$

These constraints are equivalent to the $2N_t$ d.c. constraints:

$$G^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) - F^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) + c^i \underline{g} \le 0 \quad i = 1, ..., N_t, \tag{2.23}$$

$$F^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) - G^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) - c^i \overline{g} \le 0 \quad i = 1, ..., N_t. \tag{2.24}$$

### 2.2.2 D.c. representation of the objective function

By using the expressions (2.19) and (2.20) in

$$\sum_{i=1}^{N_t} c^i \left( l^i - \sum_{j=1}^{N_e} h_j^i(v_j^{i-1}, v_j^i, d_j^i) \right) = \sum_{i=1}^{N_t} F^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) - \sum_{i=1}^{N_t} G^i(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots),$$

a d.c. representation of the objective function of the program (2.15) can be obtained

$$\sum_{i=1}^{N_t} c^i \left( l^i - \sum_{j=1}^{N_e} h_j^i(v_j^{i-1}, v_j^i, d_j^i) \right) = F(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) - G(\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots).$$

Defining $n = N_e(2N_t - 1)$, $m = N_e N_t$ and $x = (\ldots, v_j^{i-1}, v_j^i, d_j^i, \ldots) \in I\!R^n$, and by expressing the linear constraints in the form $Ax = b$ the program (2.15) can be rewritten as a d.c. program

$$\boxed{\begin{array}{rcl} \text{minimize} & F(x) - G(x) & \\ \text{subject to:} & (G^i(x) + c^i \underline{g}) - F^i(x) \le 0, & i = 1, ..., N_t, \\ & F^i(x) - (G^i(x) + c^i \overline{g}) \le 0, & i = 1, ..., N_t, \\ & Ax = b, & \\ & \underline{x} \le x \le \overline{x}, & \\ & A \in I\!R^{m*n}, b \in I\!R^m, \underline{x}, \overline{x} \in I\!R^n, & \end{array}} \tag{2.25}$$

which involves linear equality constraints.

### 2.2.3 The d.c. program of reduced size

The matrix $A$ of the linear constraints in (2.25) can be written as $A = [B, N]$ where $B$ is a basic nonsingular square matrix. Let $y$ and $z$ be the basic and nonbasic coordinates corresponding to the matrices $B$ and $N$, respectively. Then, $x = (y, z)$ and $y = B^{-1}(b - Nz)$, so that it is possible to reduce the size of the d.c. program (2.25) by defining the functions $\varphi_1(z) = F(x)$, $\varphi_2(z) = G(x)$, $\varphi_1^i(z) = F^i(x)$ and $\varphi_2^i(z) = G^i(x)$. By using these functions in (2.25) we have an equivalent d.c. program of reduced size expressed by

$$\boxed{\begin{array}{rll} \text{minimize} & \varphi_1(z) - \varphi_2(z) & \\ \text{subject to:} & (\varphi_2^i(z) + c^i\underline{g}) - \varphi_1^i(z) \leq 0, & i = 1, ..., N_t, \\ & \varphi_1^i(z) - (\varphi_2^i(z) + c^i\overline{g}) \leq 0, & i = 1, ..., N_t, \\ & \underline{b} \leq Mz \leq \overline{b}, & \\ & \underline{z} \leq z \leq \overline{z}, & \end{array}} \tag{2.26}$$

where $M = B^{-1}N$, $\underline{b} = B^{-1}b - \overline{y}$ and $\overline{b} = B^{-1}b - \underline{y}$.

## 2.3 Equivalent reverse convex programs

It should be stressed that by applying properties of the d.c. functions, several different transformations can be used to obtain an equivalent reverse convex program to the d.c. program (2.26).

### 2.3.1 The equivalent canonical d.c. program

The d.c. program (2.26) can be transformed into an equivalent d.c. program with a linear objective by adding the variable $\tau$ in the form:

$$\begin{array}{rll} \text{minimize} & \tau & \\ \text{subject to:} & \varphi_1(z) - \varphi_2(z) - \tau \leq 0, & \\ & (\varphi_2^i(z) + c^i\underline{g}) - \varphi_1^i(z) \leq 0, & i = 1, ..., N_t, \\ & \varphi_1^i(z) - (\varphi_2^i(z) + c^i\overline{g}) \leq 0, & i = 1, ..., N_t, \\ & \underline{b} \leq Mz \leq \overline{b}, & \\ & \underline{z} \leq z \leq \overline{z}. & \end{array} \tag{2.27}$$

The nonlinear constraints in (2.27) can be expressed using a single constraint by defining

$$r(z, \tau) = \max\{\varphi_1(z) - (\varphi_2(z) + \tau),\ (\varphi_2^i(z) + c^i\underline{g}) - \varphi_1^i(z),\ \varphi_1^i(z) - (\varphi_2^i(z) + c^i\overline{g}),\ i = 1, ..., N_t\},$$

so that (2.27) can be written

$$\begin{array}{rl} \text{minimize} & \tau \\ \text{subject to:} & r(z, \tau) \leq 0, \\ & \underline{b} \leq Mz \leq \overline{b}, \\ & \underline{z} \leq z \leq \overline{z}. \end{array} \tag{2.28}$$

From the properties of the d.c. functions (see Preliminaries or [27] and [32] for more information), a d.c. representation of $r(z,\tau) = p(z,\tau) - q(z,\tau)$ can be obtained by using the convex functions

$$p(x,\tau) = \max \left\{ \begin{array}{ll} 2\varphi_1(z) + \varphi_2(z) + N_t c^i \overline{g}, & \\ \varphi_1(z) + 2\varphi_2(z) + \tau + (\varphi_2^i(z) - \varphi_1^i(z) + c^i \underline{g}) + N_t c^i \overline{g} & i = 1,...,N_t, \\ \varphi_1(z) + 2\varphi_2(z) + \tau + (\varphi_1^i(z) - \varphi_2^i(z) - c^i \overline{g}) + N_t c^i \overline{g} & i = 1,...,N_t \end{array} \right\}$$

and

$$q(z,\tau) = \varphi_1(z) + 2\varphi_2(z) + \tau + N_t c^i \overline{g}.$$

A more suitable d.c. representation of $r(z,\tau)$ can be obtained by defining the convex functions

$$p_0(z,\tau) = \max \left\{ \begin{array}{ll} 2\varphi_1(z) - \tau, & \\ \varphi_1(z) + \varphi_2(z) + (\varphi_2^i(z) - \varphi_1^i(z) + c^i \underline{g}) & i = 1,...,N_t, \\ \varphi_1(z) + \varphi_2(z) + (\varphi_1^i(z) - \varphi_2^i(z) - c^i \overline{g}) & i = 1,...,N_t \end{array} \right\}$$

and

$$q_0(z,\tau) = \varphi_1(z) + \varphi_2(z).$$

Then, we can write

$$p(z,\tau) = \varphi_2(z) + \tau + N_t c^i \overline{g} + p_0(z,t) \tag{2.29}$$

and

$$q(z,\tau) = \varphi_2(z) + \tau + N_t c^i \overline{g} + q_0(z,t), \tag{2.30}$$

so a new d.c. representation of $r(z,t)$ can be obtained

$$r(z,\tau) = p(z,\tau) - q(z,\tau) = p_0(z,\tau) - q_0(z,\tau). \tag{2.31}$$

By introducing a new variable $t$, the constraint $r(z,\tau) \leq 0$, can be replaced by an equivalent pair of convex and reverse convex constraints

$$\begin{array}{l} p_0(z,\tau) - t \leq 0, \\ q_0(z,\tau) - t \geq 0, \end{array} \tag{2.32}$$

respectively. Hence, by defining the closed convex sets

$$D = \{(z,\tau,t) : \; p_0(z,\tau) - t \leq 0, \; \underline{b} \leq Mz \leq \overline{b}, \; \underline{z} \leq z \leq \overline{z}\}$$

and

$$C = \{(z,\tau,t) : \; q_0(z,\tau) - t \leq 0\}$$

the d.c. program (2.28) is equivalent to the canonical d.c. program

$$\boxed{\begin{array}{ll} \text{minimize} & \tau \\ \text{subject to:} & (z,\tau,t) \in D \setminus \text{int}C. \end{array}} \tag{2.33}$$

### 2.3.2 A more advantageous equivalent reverse convex program

To solve the program (2.33) we need to find a vertex to the conical subdivisions by solving an initial convex program, and bound the closed convex sets of the resultant complementary mathematical convex structure. In this section, the d.c. representation of the functions in the d.c. program are used to find a suitable equivalent reverse convex program in order to obtain, by using a prismatical subdivision process (see definition in Chapter 4 or see [33] and [34] for more information), an advantageous adaptation of the *combined outer approximation and cone splitting conical algorithm for canonical d.c. programming* as described in [74].

The pair of constraints (2.32) can be expressed by the $2N_t+1$ convex constraints

$$\begin{array}{rl} 2\varphi_1(z) - \tau - t \leq 0, & \\ \varphi_1(z) + \varphi_2(z) + (\varphi_2^i(z) - \varphi_1^i(z) + c^i\underline{g}) - t \leq 0 & i = 1, ..., N_t, \\ \varphi_1(z) + \varphi_2(z) + (\varphi_1^i(z) - \varphi_2^i(z) - c^i\overline{g}) - t \leq 0 & i = 1, ..., N_t \end{array} \tag{2.34}$$

and the reverse convex constraint

$$\varphi_1(z) + \varphi_2(z) - t \geq 0 \tag{2.35}$$

Hence, by defining the closed convex sets

$$\Omega = \left\{ (z,t) : \begin{array}{c} \varphi_1(z) + \varphi_2(z) + (\varphi_2^i(z) - \varphi_1^i(z) + c^i\underline{g}) - t \leq 0, \\ \varphi_1(z) + \varphi_2(z) + (\varphi_1^i(z) - \varphi_2^i(z) - c^i\overline{g}) - t \leq 0, \\ i = 1, ..., N_t, \\ \underline{b} \leq Mz \leq \overline{b}, \\ \underline{z} \leq z \leq \overline{z} \end{array} \right\}$$

and

$$\Delta = \{(z,t) : \ \varphi_1(z) + \varphi_2(z) - t \leq 0\},$$

and by using as objective function the convex function $2\varphi_1(z) - t$ a new canonical d.c. program equivalent to the d.c. program (2.26) can be written

$$\boxed{\begin{array}{ll} \text{minimize} & 2\varphi_1(z) - t \\ \text{subject to:} & (z,t) \in \Omega \setminus \text{int}\Delta. \end{array}} \tag{2.36}$$

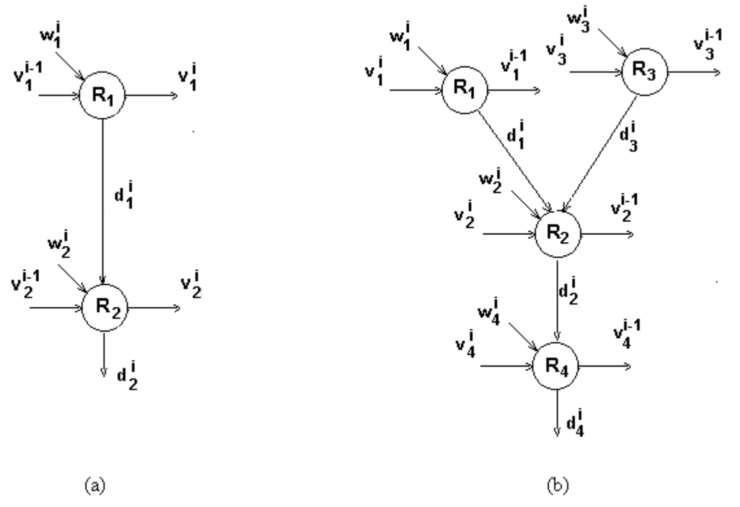

Figure 2.3: The two basic models for obtaining the replicated hydronetwork of the instances of the hydrogeneration systems.

Table 2.1: Characteristics of the hydrogeneration systems

| *Problem* | *Nodes* | *Intervals* | *Dimension* | *Linear const.* | *Nonlinear const.* |
|---|---|---|---|---|---|
| | $N_e$ | $N_t$ | $N_e(2N_t-1)$ | $N_eN_t$ | $N_t$ |
| *gp*2*e*02*i* | 2 | 2 | 6 | 4 | 2 |
| *gp*2*e*03*i* | 2 | 3 | 10 | 6 | 3 |
| *gp*4*e*03*i* | 4 | 3 | 20 | 12 | 3 |
| *gp*4*e*04*i* | 4 | 4 | 28 | 16 | 4 |
| *gp*2*e*08*i* | 2 | 8 | 30 | 16 | 8 |

## 2.4 Characteristics of the hydrogeneration systems

The characteristics of the instances of the hydrogeneration systems can be found in Table 2.1, where we use $N_e$ to indicate the number of reservoirs and $N_t$ to indicate the number of time intervals. The names of the hydrogeneration systems in Table 2.1 are denoted by **gp**$n$**e**$m$**i**, where $n = N_e$, $m = N_t$. Figure 2.3 displays the two basic models for obtaining the replicated hydronetwork of the instances of the hydrogeneration systems used in this Thesis. Model $(a)$ corresponds with the instances where $n = 2$, and model $(b)$ with the instances where $n = 4$. In this point, we must give some explanations about how the constants, the variables and the functions are measured in the Generation Problem.

- We know that the power hydrogeneration function $h_j^i$ over the $i^{\text{th}}$ interval at the reservoir $j$ is measured in $Kw$ and it is approximated by a polynomial function of fourth degree in the variables $v_j^{i-1}$, $v_j^i$ and $d_j^i$, which are measured in $Hm^3$.

$$\begin{aligned} h_j^i(v_j^{i-1}, v_j^i, d_j^i) &= k_j^i \tfrac{d_j^i}{t^i}[s_{vb} + \tfrac{s_{vl}}{2}(v_j^{i-1} + v_j^i) + \tfrac{s_{vq}}{3}(v_j^i - v_j^{i-1})^2 + \\ &\quad + s_{vq} v_j^{i-1} v_j^i + \tfrac{s_{vc}}{4}((v_j^{i-1})^2 + (v_j^i)^2)(v_j^{i-1} + v_j^i) - \\ &\quad - s_{dl} d_j^i - s_{dq}(d_j^i)^2]. \end{aligned}$$

- The coefficients $s_{vb}(m)$, $s_{vl}(m/Hm^3)$, $s_{vq}(m/Hm^6)$, $s_{vc}(m/Hm^9)$, $s_{dl}(m)$ and $s_{dq}(m/Hm^3)$ are technological coefficients which depend on each reservoir.

- The time intervals $t_i$, $i = 1, \ldots, N_t$ and the time period $t_p = \sum_{i=1}^{N_t} t_i$ are measured in *hours*.

- The bounds for the water discharge are denoted by $\underline{d_j}$, for the minimum, and by $\overline{d_j}$, for the maximum, and they are measured in $Hm^3/h$ for all $j = 1, ..., N_e$. Hence, we must take into account the inequalities

$$t^i \underline{d_j} \leq d_j^i \leq t^i \overline{d_j},$$

for all $i = 1, ..., N_t$ and $j = 1, ..., N_e$.

- The efficiency and unit conversion coefficient to maintain the coherence of the measurement scale is

$$k_j^i = 3.6 * 10^{-6} \rho_j^i g,$$

with $\rho_j^i = \rho(v_j^{i-1}, v_j^i, \frac{d_j^i}{t^i})$ verifying $0 \leq \rho_j^i \leq 1$ and $g = 9.8 m/s^2$. The efficiency coefficient of the turbine $\rho_j^i$ depends on the variables $v_j^{i-1}$, $v_j^i$ i $d_j^i/t_i$ but, for the sake of simplicity, in our problem this coefficient is a concave polynomial function of second degree, which only depends on the variable $d_j^i/t^i$ as follows.

$$\rho_j^i := (k_q)_j \left(\frac{d_j^i}{t^i}\right)^2 + (k_l)_j \left(\frac{d_j^i}{t^i}\right),$$

with $(k_q)_j$ measured in $(Hm^3/h)^{-2}$ and $(k_l)_j$ measured in $(Hm^3/h)^{-1}$ for all $i = 1, ..., N_t$ and $j = 1, ..., N_e$.

- Also, the natural water inflows $w_j$ are measured in $Hm^3/h$ for all $j = 1, ..., N_e$ and $w_j^i = w_j t^i$, where $w_j^i$ are measured in $Hm^3$ for all $i = 1, ..., N_t$ and $j = 1, ..., N_e$.

Table 2.2: The technological coefficients at each reservoir

| Reservoir | $s_{vb}$ | $s_{vl}$ | $s_{vq}$ | $s_{vc}$ |
|---|---|---|---|---|
| $R_1$ | $0.385835e+2$ | $0.191407e+0$ | $-.430230e-4$ | $0.485060e-7$ |
| $R_2$ | $0.114000e+2$ | $0.116999e+0$ | $-.193817e-4$ | $0.309579e-7$ |
| $R_3$ | $0.385835e+2$ | $0.191407e+0$ | $-.430230e-4$ | $0.485060e-7$ |
| $R_4$ | $0.114000e+2$ | $0.116999e+0$ | $-.193817e-4$ | $0.309579e-7$ |
| | $s_{dl}$ | $s_{dq}$ | $k_l$ | $k_q$ |
| $R_1$ | $0.581395e-4$ | $0.253515e-6$ | $0.355556e+1$ | $-.395062e+1$ |
| $R_2$ | $0.133333e-3$ | $0.133333e-5$ | $0.416667e+1$ | $-.578704e+1$ |
| $R_3$ | $0.581395e-4$ | $0.253515e-6$ | $0.355556e+1$ | $-.395062e+1$ |
| $R_4$ | $0.133333e-3$ | $0.133333e-5$ | $0.416667e+1$ | $-.578704e+1$ |

The generation bounds are $\underline{g} = 0.01\ Kwh$ and $\overline{g} = 150\ Kwh$. The time period has been established to $t_p = 24$ for each instance of Table 2.1. The technological coefficients and the coefficients of $\rho^i_j$, $j = 1, ..., N_e$ of the reservoirs used in this Thesis are described in Table 2.2. The bounds on the variables and the volumes stored at the beginning and at the end of the time period are in Table 2.3, and they are the same for all instances in Table 2.1.

Table 2.3: Bounds on the volumes ($Hm^3$) and the water discharges ($Hm^3/h$), and the volumes stored at the beginning and at the end of the time period at each reservoir ($Hm^3$)

| Reservoir | $\underline{v}$ | $\overline{v}$ | $\underline{d}$ | $\overline{d}$ | $v_{ini}$ | $v_{fin}$ |
|---|---|---|---|---|---|---|
| $R_1$ | $0.100e+3$ | $0.400e+3$ | $0.00e0$ | $0.250e+3$ | $0.344e+3$ | $0.344e+3$ |
| $R_2$ | $0.500e+2$ | $0.200e+3$ | $0.00e0$ | $0.200e+3$ | $0.150e+3$ | $0.150e+3$ |
| $R_3$ | $0.100e+3$ | $0.400e+3$ | $0.00e0$ | $0.250e+3$ | $0.344e+3$ | $0.344e+3$ |
| $R_4$ | $0.500e+2$ | $0.200e+3$ | $0.00e0$ | $0.200e+3$ | $0.150e+3$ | $0.150e+3$ |

Table 2.4 shows the forecasts for electricity consumption and the natural water inflow, into the reservoirs of the hydrogeneration systems, at each time interval.

Table 2.4: Forecasts for electricity consumption ($Kw$) and the natural water inflow ($Hm^3/h$) at each time interval for the reservoirs of the instances of the hydrogeneration systems

| $gp2e02i$ | $t_1$ | $t_2$ | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $l$ | $0.16e5$ | $0.13e4$ | | | | | | |
| $w_1$ | $0.70e2$ | $0.70e2$ | | | | | | |
| $w_2$ | $0.40e2$ | $0.40e2$ | | | | | | |
| $gp2e03i$ | $t_1$ | $t_2$ | $t_3$ | | | | | |
| $l$ | $0.10e4$ | $0.10e4$ | $0.90e3$ | | | | | |
| $w_1$ | $0.70e2$ | $0.70e2$ | $0.70e2$ | | | | | |
| $w_2$ | $0.40e2$ | $0.40e2$ | $0.40e2$ | | | | | |
| $gp2e08i$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ | $t_8$ |
| $l$ | $3.62e2$ | $3.62e2$ | $3.62e2$ | $3.62e2$ | $3.62e2$ | $3.62e2$ | $3.62e2$ | $3.62e2$ |
| $w_1$ | $0.7e2$ | $0.7e2$ | $0.7e2$ | $0.7e2$ | $0.7e2$ | $0.7e2$ | $0.7e2$ | $0.7e2$ |
| $w_1$ | $0.4e2$ | $0.4e2$ | $0.4e2$ | $0.4e2$ | $0.4e2$ | $0.4e2$ | $0.4e2$ | $0.4e2$ |
| $gp4e03i$ | $t_1$ | $t_2$ | $t_3$ | | | | | |
| $l$ | $0.200e3$ | $0.135e3$ | $0.110e3$ | | | | | |
| $w_1$ | $0.130e1$ | $0.215e1$ | $0.155e1$ | | | | | |
| $w_2$ | $0.100e1$ | $0.200e1$ | $0.150e1$ | | | | | |
| $w_3$ | $0.155e1$ | $0.115e1$ | $0.205e1$ | | | | | |
| $w_4$ | $0.125e1$ | $0.150e1$ | $0.125e1$ | | | | | |
| $gp4e04i$ | $t_1$ | $t_2$ | $t_3$ | $t4$ | | | | |
| $l$ | $0.150e3$ | $0.185e3$ | $0.160e3$ | $0.150e3$ | | | | |
| $w_1$ | $0.100e1$ | $0.175e1$ | $0.100e1$ | $0.125e1$ | | | | |
| $w_2$ | $0.750e0$ | $0.100e1$ | $0.175e1$ | $0.100e1$ | | | | |
| $w_3$ | $0.125e1$ | $0.750e0$ | $0.100e1$ | $0.175e1$ | | | | |
| $w_4$ | $0.100e1$ | $0.100e1$ | $0.750e1$ | $0.100e1$ | | | | |

# Chapter 3

# How to obtain a d.c. representation of a polynomial

## 3.1 Introduction

The main problem i this Chapter is *how to obtain a d.c. representation of a d.c. function*. While it is not too difficult to prove theoretically that a given function is d.c. it is often problematic to obtain an effective d.c. representation of a d.c. function. We are interested in d.c. representations of polynomials. The aim of this Chapter is to provide a new procedure (see A.Ferrer [16]) to obtain an explicit representation of a polynomial as a difference of convex polynomials (Corollary 3.3.4), based on the fact that the set of $m^{th}$ powers of homogeneous polynomials of degree 1 is a generating set for the vector space of homogeneous polynomials of degree $m$ (Proposition 3.3.2). Also, we compare our procedure with the procedure described by Konno, Thach and Tuy [42], emphasizing its advantages and applying it to the polynomials of our problem.

## 3.2 Some relevant properties of polynomials and convex functions

In this section some useful lemmas are stated and the references, where their proofs can be found, are also given. Lemmas without specific references are proved.

Let $I\!R_m[x_1, \ldots, x_n]$ be the vector space of polynomials of degree less than or equal to $m$ and let $H_i\left[x_1,\ldots,x_n\right]$ be the vector space of homogeneous polynomials of degree $i$ (generated by monomials $x_1^{t_1}x_2^{t_2}\ldots x_n^{t_n}$, with $t_1+t_2+\ldots+t_n=i$, which form a basis).

**Lemma 3.2.1** *(Lang [45]) The following properties hold:*

*(i) If $i\le m$ then $H_i\left[x_1,\ldots,x_n\right]$ is a vector subspace of $I\!R_m\left[x_1,\ldots,x_n\right]$*

*(ii) $s_i$=dim $H_i\left[x_1,\ldots,x_n\right]=C^i_{n+i-1}:=\begin{pmatrix} n+i-1\\ i\end{pmatrix}$*

*(iii) $I\!R_m\left[x_1,\ldots,x_n\right]=\bigoplus_{i=0}^m H_i\left[x_1,\ldots,x_n\right]$*

Let $p(x_1,\ldots,x_n)\in I\!R^m\left[x_1,\ldots,x_n\right]$. Thus, we can write

$$p(x_1,\ldots,x_n)=\sum_{i=0}^m q_i(x_1,\ldots,x_n),$$

where $q_i(x_1,\ldots,x_n)\in H_i\left[x_1,\ldots,x_n\right]$ $i=0,1,\ldots,m$.

If $\{m_i^j(x_1,\ldots,x_n);\ j=1,2,\ldots,s_i\}$ is the usual monomial basis of the vector space of homogeneous polynomials of degree $i$ then

$$q_i(x_1,\ldots,x_n)=\sum_{j=1}^{s_i}\lambda_j m_i^j(x_1,\ldots,x_n).$$

**Lemma 3.2.2** *(Avriel [4]) Let $f: D\subset I\!R^n\longrightarrow I\!R$ be a convex function in the convex set $D$ and let $\Psi: B\subset I\!R\longrightarrow I\!R$ be a nondecreasing proper convex function in the convex set $B$. If $f(D)\subset B$ then $\Psi\circ f: D\subset I\!R^n\longrightarrow I\!R$ is convex (where $\Psi\circ f$ means the function $(\Psi\circ f)(x)=\Psi(f(x))$ for all $x\in D$ ).*

**Lemma 3.2.3** *Let $g(x):=\left(\sum_{i=1}^n a_ix_i\right)^m$ be a function which is the $m^{th}$ power of a linear function where $m$ is a nonnegative integer. Define the closed half spaces:*

$$H_m^+=\left\{(x_1,\ldots,x_n)\in I\!R^n:\quad \sum_{i=1}^n a_ix_i\ge 0\right\}\tag{3.1a}$$

$$H_m^-=\left\{(x_1,\ldots,x_n)\in I\!R^n:\quad \sum_{i=1}^n a_ix_i\le 0\right\}\tag{3.1b}$$

*The following results hold:*

*i) If $m$ is even then $g(x)$ is a convex function on $I\!R^n$.*

*ii) If $m$ is odd then $g(x)$ is convex on $H_m^+$ and concave on $H_m^-$.*

*iii) All expressions of the form*

$$h(x) = \sum_{j=1}^{s} \lambda_j \left( \sum_{i=1}^{n} a_i^j x_i \right)^{m_j}, \tag{3.2}$$

*where $\lambda_j > 0$ and $m_j \in I\!N$ are convex functions on the convex set*

$$D = \left\{ (x_1, \ldots, x_n) \in I\!R^n : \quad \sum_{i=1}^{n} a_i^j x_i \geq 0, \text{ for all } j \in J \right\}, \tag{3.3}$$

*where $J = \{j : m_j \text{ is odd}\}$.*

Proof: Let $\Psi(t) = t^m$ be a nondecreasing proper convex function on $I\!R_+$. By applying Lemma 3.2.2 to the linear functions $\sum_{i=1}^n a_i x_i$ and $-\sum_{i=1}^n a_i x_i$ on $H_m^+$ and $H_m^-$ respectively we can prove *i*) and *ii*). As regards *iii*), consider, for all $j = 1 \ldots s$, the set $D = \cap_{j=1}^s D_j$ where $D_j = H_{m_j}^+$ when $m_j$ is odd, and $D_j = I\!R^n$ when $m_j$ is even. It is easy to see that $D$ is a convex subset of each convex set $D_j$, $j = 1 \ldots s$. Hence, each term $\left(\sum_{i=1}^n a_i^j x_i\right)^{m_j}$ is a convex function on $D$ and since $\lambda_j > 0$, for all $j = 1 \ldots s$, it follows that $h(x)$ is a convex function on $D$. □

**Lemma 3.2.4** *(Cox et al [9]) Let $q(x_1, \ldots, x_n) \in I\!R_m[x_1, \ldots, x_n]$ be a polynomial of degree $m$.*

*i) Let $q = \sum_{i=0}^m q_i$ be the expansion of $q$ as the sum of its homogeneous components where $q_i$ has total degree $i$. Then*

$$\overline{q}(x_0, x_1, \ldots, x_n) = \sum_{i=0}^{m} q_i(x_1, \ldots, x_n) x_0^{m-i} \tag{3.4}$$

*is homogeneous polynomial of total degree $m$ in $H_m[x_0, x_1, \ldots, x_n]$. We will call $\overline{q}$ the homogenization of $q$.*

*ii) The homogenization of $q$ can be computed using the formula*

$$\overline{q}(x_0, x_1, \ldots, x_n) = x_0^m q\left(\frac{x_1}{x_0}, \ldots, \frac{x_n}{x_0}\right). \tag{3.5}$$

*iii) Taking $x_0 = 1$ we thus get $q(x_1, \ldots, x_n) = \overline{q}(1, x_1, \ldots, x_n)$. For this reason we will call $q$ the dehomogenization of the polynomial $\overline{q}$.*

*iv) Let $F(x_0,\dots,x_n)$ be homogeneous polynomial and let $x_0^s$ be the highest power of $x_0$ dividing $F$. If $f(x_1,\dots,x_n)=F(1,x_1,\dots,x_n)$ is a dehomogenization of $F$, then*

$$F = x_0^s \overline{f}. \tag{3.6}$$

*(v) Let $\Phi$ be the function*

$$\begin{array}{rcc} \Phi : I\!R_m\left[x_1,\dots,x_n\right] & \longmapsto & H_m\left[x_0,x_1,\dots,x_n\right] \\ q & \longmapsto & \Phi(q)=\overline{q} \end{array} \tag{3.7}$$

*then $\Phi$ is a real vector space isomorphism.*

## 3.3 D.c. representation of a polynomial as a difference of convex polynomials

In this section an explicit decomposition of a polynomial as a difference of convex polynomials is given by using polynomials which are $m^{th}$ powers of homogeneous polynomials of degree 1. These polynomials are a generating set for the vector space of homogeneous polynomials of degree $m$.

The following lemma is well known. We include a proof for the sake of completeness.

**Lemma 3.3.1** *Let $\alpha_0,\alpha_1,\dots,\alpha_m$ be $m+1$ distinct elements of $I\!R$; then the family of $m+1$ polynomials*

$$B=\left\{(x+\alpha_0)^m,(x+\alpha_1)^m,\dots,(x+\alpha_m)^m\right\}$$

*is a basis of the vector space $I\!R_m\left[x\right]$.*

Proof: We know that the set

$$B_0=\left\{1,x,x^2,\dots,x^m\right\} \tag{3.8}$$

is a basis of $I\!R_m\left[x\right]$. Then, for all $i=0,1,\dots,m$, we can write

$$(x+\alpha_i)^m = x^m + C_m^1 x^{m-1}\alpha_i + C_m^2 x^{m-2}\alpha_i^2 + \dots + C_m^k x^{m-k}\alpha_i^k + \dots + \alpha_i^m \tag{3.9}$$

and calculate the determinant

$$\Delta = \begin{vmatrix} 1 & ... & 1 & ... & 1 \\ C_m^1\alpha_0 & ... & C_m^1\alpha_i & ... & C_m^1\alpha_m \\ C_m^2\alpha_0^2 & ... & C_m^2\alpha_i^2 & ... & C_m^2\alpha_m^2 \\ ... & ... & ... & ... & ... \\ C_m^k\alpha_0^k & ... & C_m^k\alpha_i^k & ... & C_m^k\alpha_m^k \\ ... & ... & ... & ... & ... \\ C_m^m\alpha_0^m & ... & C_m^m\alpha_i^m & ... & C_m^m\alpha_m^m \end{vmatrix} = C_m^1 ... C_m^m \begin{vmatrix} 1 & ... & 1 & ... & 1 \\ \alpha_0 & ... & \alpha_i & ... & \alpha_m \\ \alpha_0^2 & ... & \alpha_i^2 & ... & \alpha_m^2 \\ ... & ... & ... & ... & ... \\ \alpha_0^m & ... & \alpha_i^m & ... & \alpha_m^m \end{vmatrix} \tag{3.10}$$

where the expression

$$\begin{vmatrix} 1 & ... & 1 & ... & 1 \\ \alpha_0 & ... & \alpha_i & ... & \alpha_m \\ \alpha_0^2 & ... & \alpha_i^2 & ... & \alpha_m^2 \\ ... & ... & ... & ... & ... \\ \alpha_0^m & ... & \alpha_i^m & ... & \alpha_m^m \end{vmatrix} = \prod_{i=0}^{m} \prod_{j=i+1}^{m} (\alpha_i - \alpha_j) \neq 0 \tag{3.11}$$

is the nonzero Vandermonde determinant which completes the proof. □

**Proposition 3.3.2** *(Chambadal et al [7] Ex.111, p.498) Let $H_m[x_1, x_2, \ldots, x_n]$ be the vector space of the homogeneous polynomials of total degree $m$. Then the set $\{p^m : p \in H_1[x_1, x_2, \ldots, x_n]\}$ of $m^{th}$ powers of homogeneous polynomials of degree 1 is a generating set for $H_m[x_0, x_1, \ldots, x_n]$. We have*

$$H_m[x_1, x_2, \ldots, x_n] = \langle p^m : p \in H_1[x_1, x_2, \ldots, x_n] \rangle \tag{3.12}$$

*(the notation $\langle \ldots \rangle$ means the span of the set it contains).*

Proof: The proof is by induction on $n$. It is obvious for the case $n = 1$ because $H_m[x_1] = \langle x_1^m \rangle$. For the case $n = 2$, consider $h \in H_m[x_1, x_2]$. According the Lemma 3.2.4 $\Phi$ is an isomorphism, then there exists $q \in I\!R_m[x_1, x_2]$ where $h = \Phi(q)$. Let $\alpha_0, \alpha_1, \ldots, \alpha_m \in I\!R$ be a collection of different numbers. From the Lemma 3.3.1 there exist $\lambda_0, \lambda_1, \ldots, \lambda_m \in I\!R$ such that $q(x_2) = \sum_{i=0}^m \lambda_i (x_2 + \alpha_i)^m$, whence

$$h(x_1, x_2) = x_2^m q\left(\frac{x_1}{x_2}\right) = x_2^m \sum_{i=0}^m \lambda_i \left(\frac{x_1}{x_2} + \alpha_i\right)^m = \sum_{i=0}^m \lambda_i (x_1 + \alpha_i x_2)^m.$$

This proves that the polynomials $(x_1 + \alpha_i x_2)^m$ $i = 0, 1, \ldots, m$ generate $H_m[x_1, x_2]$. Let us assume that our assertion holds for the case $n - 1$. We will now prove it for the case $n$.

Let $h\left(x_1,\ x_2,\ldots,x_{n-1},1\right) \in \mathbb{R}_m\left[x_1,\ldots,x_{n-1}\right]$ be the dehomogenization of the polynomial $h\left(x_1,x_2,\ldots,x_n\right) \in H_m\left[x_1,x_2,\ldots,x_n\right]$ with respect to the variable $x_n$. Then, we can write

$$h\left(x_1,\ x_2,\ldots,x_{n-1},1\right) = \sum_{s=0}^{m} h_s, \text{ with } h_s \in H_s\left[x_1,x_2,\ldots,x_{n-1}\right], \tag{3.13}$$

and by induction

$$h_s\left(x_1,\ x_2,\ldots,x_{n-1}\right) = \sum_i \mu_{s,i}(P_{s,i})^s \tag{3.14}$$

with

$$P_{s,i} = a_1^{s,i}x_1 + \ldots + a_{n-1}^{s,i}x_{n-1}. \tag{3.15}$$

Then, we have

$$h\left(x_1,\ x_2,\ldots,x_n\right) = \sum_{s=0}^{m} x_n^{m-s} h_s\left(x_1,\ x_2,\ldots,x_{n-1}\right) \tag{3.16}$$

and using the expression (3.14) within (3.16) we can write

$$h\left(x_1,\ x_2,\ldots,x_n\right) = \sum_{s=0}^{m} x_n^{m-s}\left(\sum_i \mu_{s,i}(P_{s,i})^s\right) = \sum_{i,\ 0\le s\le m} \mu_{s,i}x_n^{m-s}(P_{s,i})^s. \tag{3.17}$$

Every homogeneous polynomial $x_n^{m-s}(P_{s,i})^s$ in the variables $x_n$ and $P_{s,i}$ can be expressed as a linear combination of $m^{\text{th}}$ powers $(\alpha_{s,i,j}P_{s,i} + \beta_{s,i,j}x_n)^m$

$$x_n^{m-s}(P_{s,i})^s = \sum_j \rho_{s,i,j}\left(\alpha_{s,i,j}P_{s,i} + \beta_{s,i,j}x_n\right)^m. \tag{3.18}$$

Combining (3.15) with (3.18) we obtain

$$x_n^{m-s}(P_{s,i})^s = \sum_j \rho_{s,i,j}\left(\alpha_{s,i,j}a_1^{s,i}x_1 + \ldots + \alpha_{s,i,j}a_{n-1}^{s,i}x_{n-1} + \beta_{s,i,j}x_n\right)^m. \tag{3.19}$$

Substituting the expression (3.19) in (3.17) we obtain $h\left(x_1,x_2,\ldots,x_n\right)$ as a linear combination of $m^{\text{th}}$ powers of homogeneous polynomials of total degree 1, which proves the proposition. $\square$

**Corollary 3.3.3** *Let $q_i\left(x_1,x_2,\ldots,x_n\right)$ be an homogeneous polynomial of total degree $i$. Then, there exist bases $B_{(i)}$ of $i^{th}$ powers of homogeneous polynomials of total degree 1, $B_{(i)} = \left\{(a_1^j x_1 + \ldots + a_n^j x_n)^i,\ j = 1\ldots s_i\right\}$ so that*

$$q_i\left(x_1,x_2,\ldots,x_n\right) = \sum_{j=1}^{s_i} \mu_j(a_1^j x_1 + \ldots + a_n^j x_n)^i, \tag{3.20}$$

*with $\mu_j \in \mathbb{R},\ j = 1,\ldots,s_i$. (Recall that $s_i$=dim $H_i\left[x_1,\ldots,x_n\right]$, see Lemma 3.2.1.)*

**Corollary 3.3.4** *Let $I^+ = \{j; \mu_j > 0\}$ and $I^- = \{j; \mu_j < 0\}$. Then, in the convex set $D_i = \left\{(x_1, \ldots, x_n) \in I\!R^n : \ \sum_{k=1}^n a_k^j x_k \geq 0,\ j = 1 \ldots s_i \text{ with } \mu_j \neq 0\right\}$ if $i$ is even or $D_i = I\!R^n$ if $i$ is odd, we have*

$$q_i\left(x_1, x_2, \ldots, x_n\right) = \sum_{j \in I^+} \mu_j \left(\sum_{k=1}^n a_k^j x_k\right)^i - \sum_{j \in I^-} (-\mu_j) \left(\sum_{k=1}^n a_k^j x_k\right)^i, \tag{3.21}$$

*where both components are convex. Hence, each polynomial $p(x) \in I\!R_m\left[x_1, \ldots, x_n\right]$, $p = \sum_{i=0}^m q_i$ can be expressed as a d.c. function on the convex set $D = \cap_{i=0}^m D_i$.*

PROOF: This result follows from Corollary 3.3.3 and Lemma 3.2.3. □

Corollaries 3.3.3 and 3.3.4 provide a procedure to get d.c. representations of polynomials expressed in terms of suitable bases of $i^{\text{th}}$ powers (see Appendix A, where a procedure by using $MAPLE$ symbolic calculator has been described).

### 3.3.1 Alternative procedure to obtain a d.c. representation

It should be noted that the proof of Proposition 3.3.2 gives us an alternative procedure, different from the search for bases, for obtaining d.c. representations of polynomials. We only need to know d.c. representations of monomials in $H_i\left[x_1, x_2\right],\ i = 2, \ldots, m$. If we know that

$$xy = \frac{1}{4}(x+y)^2 - \frac{1}{4}(x-y)^2 \tag{3.22}$$

and

$$x^2 y^2 = \frac{1}{12}(x+y)^4 + \frac{1}{12}(x-y)^4 - \left(\frac{1}{6}x^4 + \frac{1}{6}y^4\right) \tag{3.23}$$

then, we can obtain a d.c. representation of $xyz^2 \in I\!R_4\left[x, y, z\right]$.

Using (3.22) we can write

$$xyz^2 = (xy)z^2 = \left(\frac{1}{4}(x+y)^2 - \frac{1}{4}(x-y)^2\right) z^2 = \frac{1}{4}(x+y)^2 z^2 - \frac{1}{4}(x-y)^2 z^2.$$

From (3.23) we have

$$(x+y)^2 z^2 = \frac{1}{12}(x+y+z)^4 + \frac{1}{12}(x+y-z)^4 - \left(\frac{1}{6}(x+y)^4 + \frac{1}{6}z^4\right)$$

and

$$(x-y)^2 z^2 = \frac{1}{12}(x-y+z)^4 + \frac{1}{12}(x-y-z)^4 - \left(\frac{1}{6}(x-y)^4 + \frac{1}{6}z^4\right),$$

whence

$$xyz^2 = \frac{1}{48}(x+y+z)^4 + \frac{1}{48}(x+y-z)^4 + \frac{1}{24}(x-y)^4 - \\ - \left(\frac{1}{24}(x+y)^4 + \frac{1}{48}(x-y+z)^4 + \frac{1}{48}(x-y-z)^4\right),$$

which is a d.c. representation of $xyz^2$ on $\mathbb{R}_4[x,y,z]$.

To use this procedure we need to know the d.c. representations of monomials in $H_i\left[x_1,x_2\right]$, $i = 2,\dots,m$, which is not always possible. On the other hand, we must take into account that using this procedure the $i^{\text{th}}$ powers of the linear combination may be dependent, whence the importance of using bases of $i^{\text{th}}$ powers of homogeneous polynomials of degree 1 to obtain d.c. representations.

## 3.4 Advantages of using $i^{th}$ powers of homogeneous polynomials of degree 1

The procedure described by Konno, Thach and Tuy [42] to obtain a d.c. representation of a polynomial uses the fact that the monomials $x_i^{t_i}$, $i = 1,\dots,n$ are convex functions on $\mathbb{R}^n_+$(positive orthant). Hence, by applying successively the property

$$\phi_1(x)\phi_2(x) = \frac{1}{2}(\phi_1(x)+\phi_2(x))^2 - \frac{1}{2}(\phi_1(x)^2+\phi_2(x)^2), \tag{3.24}$$

where $\phi_1$ and $\phi_2$ are convex functions on $\mathbb{R}^n_+$, a d.c. representation of the monomials

$$x_1^{t_1}x_2^{t_2}\dots x_n^{t_n}, \text{ with } t_1+t_2+\dots+t_n = i$$

on $\mathbb{R}^n_+$ can be obtained.

**Example 3.4.1**

$$xyz^3 = (xy)z^3 = \left(\frac{1}{4}(x+y)^2 - \frac{1}{4}(x-y)^2\right)z^3 = \frac{1}{4}(x+y)^2z^3 - \frac{1}{4}(x-y)^2z^3.$$

*Since $z^3$ is convex on $\mathbb{R}^3_+$ we can write*

$$(x+y)^2z^3 = \frac{1}{2}((x+y)^2+z^3)^2 - \frac{1}{2}((x+y)^4+z^6)$$

*and*

$$(x-y)^2z^3 = \frac{1}{2}((x-y)^2+z^3)^2 - \frac{1}{2}((x-y)^4+z^6).$$

*Hence,*

$$xyz^3 = p_1(x,y,z) - p_2(x,y,z),$$

*is a d.c. representation of* $xyz^3 \in I\!R_5[x,y,z]$ *on* $I\!R^3_+$*, in which*

$$p_1(x,y,z) = \frac{1}{8}((x+y)^2 + z^3)^2 + \frac{1}{8}((x-y)^4 + z^6)$$

*and*

$$p_2(x,y,z) = \frac{1}{8}((x+y)^4 + z^6) + \frac{1}{8}((x-y)^2 + z^3)^2$$

*are convex polynomials in* $I\!R_6[x,y,z]$ *on* $I\!R^3_+$*.*

We have presented an alternative procedure which only requires a suitable basis change of $i^{\text{th}}$ powers of homogeneous polynomials of degree 1 to obtain a d.c. representation

$$q_i(x_1,\ldots,x_n) = \sum_{j\in I^+} \mu_j \left(\sum_{k=1}^{n} a_k^j x_k\right)^i - \sum_{j\in I^-} (-\mu_j) \left(\sum_{k=1}^{n} a_k^j x_k\right)^i, \tag{3.25}$$

on suitable domains. The main advantages of using $m^{th}$ powers of homogeneous polynomials of degree 1 are:

i) Solely basis changes within $H_i[x_1,\ldots,x_n]$ are used for obtaining a d.c. representation of a polynomial (in place of individual d.c. representation for monomials). Every polynomial $p(x)$ can be expressed as a difference of convex polynomials where both components are convex polynomials of the same degree as $p(x)$ (this fact is an essential difference compared to the d.c. representations by Konno, Thach and Tuy [42], where this property is not true).

ii) To obtain different d.c. representations of a polynomial function we can use different bases in our procedure (see Appendix A).

Hence, applications to real problems, which are described by using polynomial functions, are possible.

## 3.5 D.c. representation of the power hydrogeneration function

Ordering by increasing degrees the power hydrogeneration function (2.10) and substituting the coefficients $a = k_j^i$, $b = s_{vb}$, $c = 0.5s_{vl}$, $d = s_{vq}/3$, $e = 0.25s_{vc}$, $f = s_{dl}$

and $g = s_{dq}$ and the variables $x = d_j^i$, $y = v_j^{i-1}$ and $z = v_j^i$ leads to

$$\begin{aligned} h(x,y,z) = abx + acx(y+z) - afx^2 + adx(z-y)^2 - \\ -agx^3 + 3adxyz + aex\left(y^2+z^2\right)(y+z). \end{aligned} \tag{3.26}$$

If we obtain a d.c. representation of its nonlinear homogeneous components then we can decompose it. Let $h_2$, $h_3$ and $h_4$ be power hydrogeneration homogeneous components of degree 2, 3 and 4 respectively. Hence, we have

$$h_2(x,y,z) = acx(y+z) - afx^2, \tag{3.27}$$

the homogeneous polynomial component of degree 2,

$$h_3(x,y,z) = adx(z-y)^2 + 3adxyz - agx^3, \tag{3.28}$$

the homogeneous polynomial component of degree 3, and

$$h_4(x,y,z) = aex\left(y^2+z^2\right)(y+z), \tag{3.29}$$

the homogeneous polynomial component of degree 4.

We get a suitable vector space basis using the *MAPLE* Symbolic Calculator (see Appendix A)

### 3.5.1 D.c. representation of the homogeneous polynomial component of degree 2

Let $B_2$ be the initial basis in $H_2[x,y,z]$ with

$$B_2 = \left\{x^2, y^2, z^2, xy, xz, yz\right\}. \tag{3.30}$$

It is easy to prove that the set

$$B_{(2)} := \left\{x^2, y^2, z^2, (x+y)^2, (x+z)^2, (y+z)^2\right\} \tag{3.31}$$

is another $H_2[x,y,z]$ basis. Every polynomial of $B_{(2)}$ can be represented using polynomials of $B_2$, as we can see in Table 3.1.

Let $C$ be the matrix which represents the change of coordinates from $B_{(2)}$ (coordinates $X$) to $B_2$ (coordinates $Y$).

$$C = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix}. \tag{3.32}$$

Table 3.1: Change of basis in the space of the polynomials

| $B_{(2)}$ | representation in $B_2$ | Coordinates |
|---|---|---|
| $x^2$ | $x^2$ | $(1,0,0,0,0,0)$ |
| $y^2$ | $y^2$ | $(0,1,0,0,0,0)$ |
| $z^2$ | $z^2$ | $(0,0,1,0,0,0)$ |
| $(x+y)^2$ | $x^2+y^2+2xy$ | $(1,1,0,2,0,0)$ |
| $(x+z)^2$ | $x^2+z^2+2xz$ | $(1,0,1,0,2,0)$ |
| $(y+z)^2$ | $y^2+z^2+2yz$ | $(0,1,1,0,0,2)$ |

Then, using the expression $X = C^{-1}Y$ we obtain the d.c. representation of $h_2(x,y,z)$.

$$h_2(x,y,z) = \frac{ac}{2}\left[(x+y)^2 + (x+z)^2\right] - \left[(ac+af)\,x^2 + \frac{ac}{2}\left(y^2+z^2\right)\right] \tag{3.33}$$

Note that, in this case, another possible d.c. representation is via the diagonalization of the symmetric matrix $A$ such that $h_2(x) = X^tAX$. To simplify the expressions we suppose $a=1$, $c=2$ and $f=1$ in (3.27). Hence

$$A = \begin{pmatrix} -1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}. \tag{3.34}$$

The nonzero eigenvalues of $A$ are 1 and $-2$. Moreover, the matrix $C$ of the orthonormal eigenvectors corresponding to the eigenvalues of the matrix $A$ is

$$C = \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{-2}{\sqrt{6}} & 0 \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} \end{pmatrix}. \tag{3.35}$$

Thus, using $(X^tC)D(C^tX)$, we obtain a shorter d.c. representation

$$h_2(x,y,z) = 1\left(\frac{x+y+z}{\sqrt{3}}\right)^2 - 2\left(\frac{-2x+y+z}{\sqrt{6}}\right)^2. \tag{3.36}$$

### 3.5.2 D.c. representation of the homogeneous polynomial component of degree 3

In this case, the initial basis is the set

$$B_3 = \left\{x^3, y^3, z^3, x^2y, x^2z, xy^2, xz^2, y^2z, yz^2, xyz\right\} \tag{3.37}$$

and we can choose

$$\begin{aligned} B_{(3)} := \ & \left\{x^3, y^3, z^3, (x+2y)^3, (x+2z)^3, (y+2z)^3, \right. \\ & \left. (2x+y)^3, (2x+z)^3, (2y+z)^3, (x+y+z)^3\right\} \end{aligned} \tag{3.38}$$

as the new $H_3[x, y, z]$ basis. Then, the representation of the polynomial $h_3(x, y, z)$, in the basis $B_{(3)}$ as a difference of convex polynomials is

$$\begin{aligned} h_3 = \ & a(d-g)x^3 + \tfrac{ad}{12}(x+2y)^3 + \tfrac{ad}{12}(x+2z)^3 + \tfrac{ad}{6}(x+y+z)^3 - \\ & - \left(\tfrac{ad}{2}(y^3+z^3) + \tfrac{ad}{36}(y+2z)^3 + \tfrac{ad}{12}(2x+y)^3 + \tfrac{ad}{12}(2x+z)^3 + \tfrac{ad}{36}(2y+z)^3\right). \end{aligned} \tag{3.39}$$

It should be noted that in practice, we are interested in d.c. representations of a polynomial only on the orthant $x \geq 0$. Taking into account that the odd cases lead to restrictions on the domain where the polynomial components are convex (see Lemma 3.2.3), we can use linear expressions, into $m^{\text{th}}$ powers, with positive coefficients to obtain d.c. representations, which hold on the orthant $x \geq 0$.

### 3.5.3 D.c. representation of the homogeneous polynomial component of degree 4

The initial basis is

$$\begin{aligned} B_4 = \ & \{x^4, y^4, z^4, x^3y, x^3z, xy^3, xz^3, y^3z, yz^3, \\ & x^2y^2, x^2z^2, y^2z^2, x^2yz, xy^2z, xyz^2\} \end{aligned} \tag{3.40}$$

and let

$$\begin{aligned} B_{(4)} = \ & \left\{x^4, y^4, z^4, (x+y)^4, (x+z)^4, (y+z)^4, (y-x)^4, (z-x)^4 \right. \\ & (y-z)^4, (x+y+z)^4, (x+y-z)^4, (x-y+z)^4, \\ & \left. (x+2y+z)^4, (x+y+2z)^4, (x+2y+3z)^4\right\} \end{aligned} \tag{3.41}$$

be a vector space basis for $H_4[x, y, z]$. The representation of the polynomial $h_4(x, y, z)$ as a difference of convex polynomials is

$$\begin{aligned} h^{(4} = \ & \tfrac{ae}{24}\left(4x^4 + (y-z)^4 + (x+2y+z)^4 + (x+y+2z)^4\right) - \\ & -\tfrac{ae}{24}\left(8y^4 + 8z^4 + 5(y+z)^4 + (y-x)^4 + (z-x)^4 + 4(x+y+z)^4\right). \end{aligned} \tag{3.42}$$

# Chapter 4

# The global optimization algorithm

## 4.1 Introduction

As was seen in Section 2.3.2, the following is one of the forms in which the Generation Problem can be written as a reverse convex program.

$$\begin{array}{rl} \text{minimize} & f(x) - t \\ \text{subject to:} & g(x) - t \leq 0, \\ & h(x) - t \geq 0, \\ & Ax \leq b, \end{array} \tag{4.1}$$

where $A$ is a real $m \times n$ matrix, $b \in I\!R^m$ and $f(x)$, $g(x)$ and $h(x)$ are proper convex functions on $I\!R^n$. In this Chapter, it is our aim to present a deterministic global optimization algorithm for solving reverse convex programming problems of the above-mentioned form. We will use the following notation

$$\begin{array}{rcl} D & := & \{(x,t) \in I\!R^n \times I\!R : \ Ax \leq b, \ g(x) - t \leq 0\}, \\ C & := & \{(x,t) \in I\!R^n \times I\!R : \ h(x) - t \leq 0\}, \\ C_\epsilon & := & \{(x,t) \in I\!R^n \times I\!R : \ h(x) - t + \epsilon \leq 0\}, \ \epsilon > 0, \\ D_\alpha & := & \{(x,t) \in D : \ f(x) - t \leq \alpha\}, \ \alpha \in I\!R, \end{array} \tag{4.2}$$

where $D$, $C$, $C_\epsilon$ and $D_\alpha$ are closed convex sets. Hence, the programming problem (4.1) can be rewritten in the form

$$\text{minimize } \{f(x) - t : (x,t) \in D \backslash \text{int } C\}. \tag{4.3}$$

Note that the sets $D$ and $C$ are not bounded but $D \setminus \text{int } C$ is a compact set when $Ax \leq b$ defines a polytope in $I\!R^n$ (see Figure 4.1). Algorithms for solving determin-

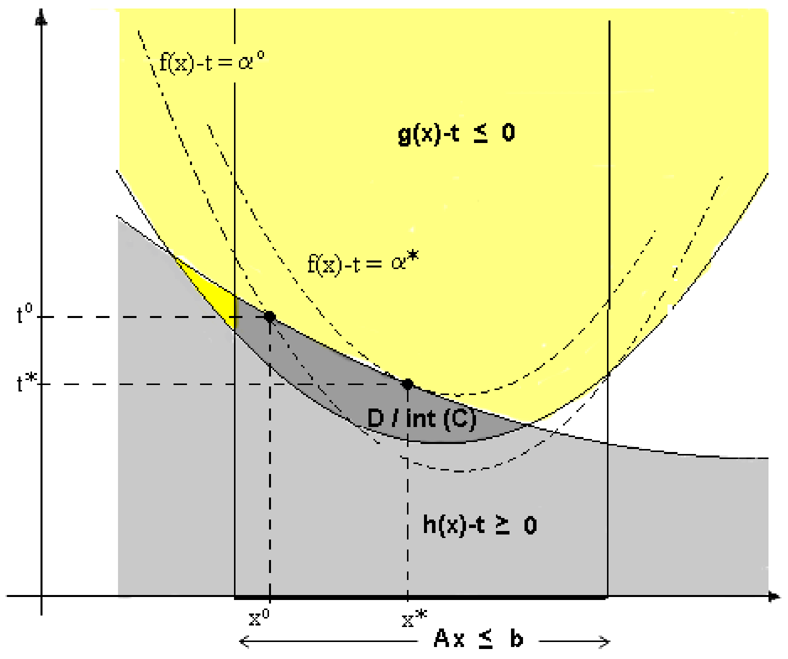

Figure 4.1: Feasible set of the reverse convex program (4.3) and some level curves of its objective function $f(x) - t$

istic reverse convex programs are described, for example, in [32] and [74]. All these algorithms begin by obtaining an initial point by solving a convex program. Without this point these algorithms cannot work because the point is the vertex of a conical subdivision process. In contrast, our algorithm, though designed in a similar spirit to algorithms used for solving reverse convex programs, has several differences and advantages. Among others, we will use prismatical subdivisions in place of conical ones so that it will not be necessary to solve an initial convex program.

In the next sections we will describe the basic operations of the algorithm which combines a prismatical subdivision process with polyhedral outer approximation in such a way that only linear programs have to be solved. Also, the convergence of the algorithm will be proved.

## 4.2 Global optimal and global $\epsilon$-optimal solutions

In the following we assume the programs (4.1) and (4.3) equivalent (i.e, (4.3) is a different way to express (4.1)), $D \backslash C \neq \emptyset$ and that for every feasible point $(x_0, t_0)$

which verifies $t_0 = g(x_0) = h(x_0)$ we have

$$B(x_0, \epsilon) \cap \{x : Ax \leq b, g(x) < h(x)\} \neq \emptyset, \tag{4.4}$$

for all $\epsilon > 0$.

**Lemma 4.2.1** *All the global optimal solutions of the programming problem (4.3) are in $D \cap \partial C$.*

PROOF: Suppose that the assertion is false. Then, there exists a global optimal solution $(x^*, t^*)$ which satisfies $h(x^*) - t^* > 0$. Let $t^o > 0$ be the real number in such a way that $h(x^*) - (t^* + t^o) = 0$. Then, the feasible point $(x^*, t^* + t^o)$ satisfies $f(x^*) - t^* > f(x^*) - (t^* + t^o)$ and $(x^*, t^*)$ cannot be a global optimal solution. □

**Lemma 4.2.2** *The programming problem (4.3) is regular, i.e.,*

$$D \backslash int\ C = cl\ (D \backslash C).$$

PROOF: We have $D\backslash \text{int } C \supset \text{cl } (D\backslash C)$ because $\text{cl } (D\backslash C)$ is the smallest closed set containing $D\backslash C$ and $D\backslash \text{int } C$ is a closed set in $I\!R^n \times I\!R$. On the other hand, let $(x^o, t^o) \in D\backslash \text{int } C$. Thus, there exists a sequence $\{(x_k, t_k),\ k = 1, 2, \ldots\}$ of points of $D\backslash C$ that converge to $(x^o, t^o)$. Indeed, three cases are possible:

1. $Ax_0 \leq b$ and $g(x_0) \leq t_0 < h(x_0)$. In this case $(x_0, t_0) \in D\backslash C$,
2. $Ax_0 \leq b$ and $g(x_0) < t_0 = h(x_0)$. In this case, the sequence

$$\{(x_0, t_0 - (t_0 - g(x_0))/2^k),\ k = 1, 2, \ldots\}$$

   of points of $D\backslash C$ converge to $(x^o, t^o)$,
3. $Ax_0 \leq b$ and $g(x_0) = t_0 = h(x_0)$. By choosing

$$x_k \in B(x_0, 1/k) \cap \{x : Ax \leq b, g(x) < h(x)\} \neq \emptyset,$$

   for every $k = 1, 2, \ldots$, and taking $t_k = (h(x_k) + g(x_k))/2$ we can see that the sequence $\{(x_k, t_k),\ k = 1, 2, \ldots\}$ of points of $D\backslash C$ converge to $(x^o, t^o)$.

Hence, $(x^o, t^o) \in \text{cl } (D\backslash C)$ in $I\!R^n \times I\!R$ and $D\backslash \text{int } C \subset \text{cl } (D\backslash C))$, which proves the lemma. □

**Lemma 4.2.3** *Let $(x^*, t^*)$ be a feasible solution of a regular program (4.3) which comes from the programming problem (4.1). The following assertions are equivalences:*

*i) $(x^*, t^*)$ is a global optimal solution,*

*ii) $D_{\alpha^*} \subset C$ with $\alpha^* = f(x^*) - t^*$.*

PROOF: i) $\Rightarrow$ ii)
Consider $(x_0, t_0) \in D_{\alpha^*}$ with $(x_0, t_0) \notin C$. Then $h(x_0) - t_0 > 0$ and from $f(x_0) - t_0 = f(x^*) - t_* = \alpha^*$ follows that $(x_0, t_0)$ is an optimal global solution of (4.3) so $h(x_0) - t_0 = 0$ which is a contradiction.
ii) $\Rightarrow$ i)
Let $(\bar{x}, \bar{t}) \in D \setminus \text{int}C$ be a better feasible solution than $(x^*, t^*)$. Then $f(\bar{x}) - \bar{t} < f(x^*) - t^*$ and we can find an open ball $B := B((\bar{x}, \bar{t}), r)$, of center $(\bar{x}, \bar{t})$ and radius $r$, such that for all $(x, t) \in B$ we have $f(x) - t < f(x^*) - t_*$. From $(\bar{x}, \bar{t}) \in D \setminus \text{int } C = \text{cl } (D \setminus C)$ follows that $B \cap \text{cl } (D \setminus C) \neq \emptyset$ so every point $(x, t) \in B \cap \text{cl } (D \setminus C)$ verify $(x, t) \in D$ and $(x, t) \notin C$ which is a contradiction. □

In practice, we only need to find the global optimum within a prescribed tolerance. Hence, the programming problem (4.3) can be considered solved when, given $\epsilon > 0$ (a nonnegative real number), a feasible solution $(x^*_\epsilon, t^*_\epsilon)$ has been found which satisfies

$$f(x^*_\epsilon) - t^*_\epsilon - \epsilon \leq f(x) - t, \text{ for all } (x, t) \in D \setminus \text{int } C. \tag{4.5}$$

Such a feasible solution is called a *global $\epsilon$-optimizer*. By defining

$$\alpha^* := \inf\{f(x) - t : (x, t) \in D, \ h(x) - t > 0\},$$

it is easily seen that the set $D_{\alpha^*} \setminus \text{int } C_\epsilon$ coincides with the set of $\epsilon$-optimizers of the programming problem (4.3). The algorithm for solving the problem (4.3) that we present in this Chapter, is an adaptation of the *Combined OA/CS Conical Algorithm for CDC* as described in Tuy [74], which corresponds to the specific structure of this program. In the next sections a branching process, in which every partition set is a simplicial prism, will be elaborated, and an outer approximation process will be described by means of a sequence of polyhedrons generated through suitable piece-wise linear functions.

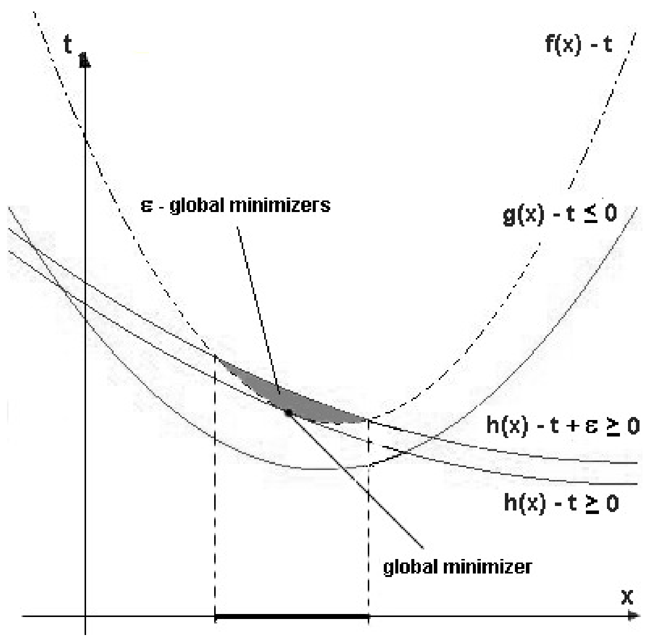

Figure 4.2: The sets of the $\epsilon$-global minimizers and global minimizers of a reverse convex program as used by the modified algorithm

## 4.3 Subdivision processes

This section is devoted to study the properties of the prismatical subdivision processes (introduced by Horst et al. [34] but without a systematic study of its properties). *The basic prismatical subdivision property* (see Proposition 4.3.6), which is fundamental to prove the convergence of our global optimization algorithm, is enunciated and proved. This study is carried out by using the simplicial subdivision properties, which have been completely developed by Tuy [74].

### 4.3.1 Simplicial subdivision processes

In this subsection we enounce the properties of the simplicial subdivision processes without proofs (see [74] for details).

Let $v^1, \ldots, v^{n+1}$ be $n+1$ points affinely independent, i.e., the $n$ vectors $v^1 - v^{n+1}, \ldots, v^n - v^{n+1}$ are linearly independent. Then, the set

$$[v^1, \ldots, v^{n+1}] := \left\{ v = \sum_{i=1}^{n+1} \lambda_i v^i, \sum_{i=1}^{n+1} \lambda_i = 1, \lambda_i \geq 0, i = 1, \ldots, n+1 \right\}$$

is called the $n$-simplex of vertices $v^1,\dots,v^{n+1}$. It is clear that $[v^1,\dots,v^{n+1}] = \operatorname{conv}\{v^1,\dots,v^{n+1}\}$. Let $Z = [v^1,\dots,v^{n+1}]$ be a $n$-simplex. Thus, if $v \in Z$ and $v \notin \{v^1,\dots,v^{n+1}\}$ we can write

$$v = \sum_{i=1}^{n+1} \lambda_i v^i, \ \sum_{i=1}^{n+1} \lambda_i = 1, \ \lambda_i \geq 0 \text{ and } \lambda_i \neq 1, \ i = 1,\dots,n+1.$$

Consider the set $I = \{i : \lambda_i > 0\}$. From $Z$, we can obtain a new $n$-simplex $Z(i,v)$ by replacing (in $Z$) the vertex $v^i$ by $v$ for each $i \in I$, i.e.,

$$Z(i,v) = [v^1,\dots,v^{i-1},v,v^{i-1},\dots,v^{n+1}], \text{ for each } i \in I.$$

Hence, $Z = \cup_{i\in I} Z(i,v)$ and each pair $Z(i,v)$, $Z(j,v)$, $i \neq j$ intersects at the most at common boundaries and comprises a subdivision of $Z$. This subdivision is called a *radial subdivision* of the simplex $Z$. Each $n$-simplex $Z(i,v)$ is referred to as a *child* of the $n$-simplex $Z$. A radial subdivision is said to be *proper*, if it consists of at least two members. An important special case is when $v = \alpha v^k + (1-\alpha)v^h$ with $0 < \alpha \leq 1/2$ is a point on the longest edge $[v^k, v^h]$ of the simplex $Z$. Then, the radial subdivision is called a *bisection* of the ratio $\alpha$. When $\alpha = 1/2$, the bisection is said to be *exact*.

Consider an infinite sequence of nested simplices

$$Z_0 \supset Z_1, \dots, \supset Z_k \supset \dots$$

such that $Z_{k+1}$ is a child of $Z_k$ in a subdivision via $v^k \in Z_k$. Such a sequence is called a *filter* and the simplex $Z_\infty := \cap_{j=0}^{\infty} Z_j$ is the *limit of the filter*. Let $\delta(D)$ be the diameter of the set $D$ as measured by the Euclidean distance. For instance, for a simplex $Z$, $\delta(Z)$ is the length of the longest edge of $Z$. A filter is said to be *exhaustive* if $\delta(Z_i) \longrightarrow 0$ when $i \longrightarrow \infty$, i.e., the filter shrinks to a single point.

**Example 4.3.1** *In Figure 4.3 the simplex $Z_0 := [v1, v2, v3]$ with $v1 = (0,0)$, $v2 = (1,0)$ and $v1 = (0,1)$ occurs in both graphics (a) and (b). In case (a) we have constructed a filter from $Z_0$ by using exact bisections as follows:*

$$Z_0 \supset Z_1 \supset Z_2 \supset Z_3 \supset \dots$$

*where $Z_1 := [v1, v2, v4]$, $Z_2 := [v1, v5, v4]$, $Z_3 := [v6, v5, v4]$ and so on. The diameter of $Z_0$ is $\delta(Z_0) = \sqrt{2}$ and the diameter of each simplex satisfies the relationship $\delta(Z_{i+1}) = \delta(Z_i)/\sqrt{2}, i = 0,1,2,\dots$ so the filter obtained is exhaustive because*

$$\delta(Z_i) = \sqrt{2}\left(\frac{1}{\sqrt{2}}\right)^{i-1} \longrightarrow 0 \ (\textit{ by letting } i \longrightarrow \infty).$$

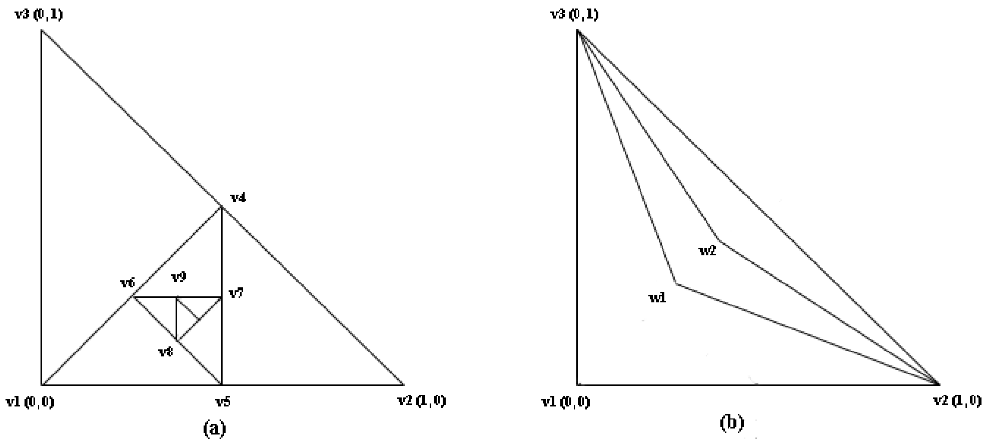

Figure 4.3: Exhaustive and nonexhaustive filters in a simplicial subdivision process for an initial simplex $Z_0 = [v1, v2, v3]$

*In case $(b)$ we have constructed a filter of $Z_0$ in which each simplex $Z_i$ is subdivided by using its barycenter $w_i$ in such a way that $Z_i = [w_i, v2, v3], i = 1, 2, \ldots$. Then, the diameter of each simplex is a constant with value $\delta(Z_i) = \sqrt{2}, i = 0, 1, 2, \ldots$. From the relationship*

$$\delta(Z_i) = \sqrt{2} \longrightarrow \sqrt{2} \text{ ( by letting } i \longrightarrow \infty)$$

*we see that the filter is nonexhaustive. Moreover, in this last case, the filter satisfies the property $ii)$ of the Lemma forthcoming 4.3.2 (by taking $\rho = \sqrt{5}/3$).*

The next lemma describes the conditions under which a filter is exhaustive.

**Lemma 4.3.2** *Let $Z_k = [v_k^1, \ldots, v_k^{n+1}]$, $k \in I\!N$ be a filter of simplices such that any $Z_{k+1}$ is a child of $Z_k$ in a subdivision via $v^k \in Z_k$ and we assume that,*

*i) for infinitely many $k$ the subdivision of $Z_k$ is a bisection,*

*ii) there exists a constant $0 < \rho < 1$ such that for every $k \in I\!N$,*

$$max\{\|v^k - v_k^i\|,\ i = 1, \ldots, n+1\} \leq \rho\delta(Z_k). \tag{4.6}$$

*Then the filter is exhaustive.*

The condition $ii)$ is essential for exhaustiveness of the filter. It implies in particular that the bisections in $i)$ are bisections of ratio no less than $1 - \rho > 0$.

**Theorem 4.3.3** *Let $\{Z_k,\ k = 1, 2, \ldots\}$ be a filter of simplices and for each $k$ let $v^k$ be a point in $Z_k$ which is not a vertex of $Z_k$. We assume that*

*i) for infinitely many $k$, $Z_{k+1}$ is a child of $Z_k$ via a bisection of ratio no less than a constant $\alpha > 0$,*

*ii) for all other $k$, $Z_{k+1}$ is a child of $Z_k$ in a subdivision via $v^k$.*

*Then at least one accumulation point of the sequence $\{v^k,\ k = 0, 1, 2, \ldots\}$ is a vertex of $Z_\infty = \cap_{k=1}^{\infty} Z_k$ (note that $Z_\infty$ could be a $p$-simplex with $0 < p < n + 1$).*

**Corollary 4.3.4** *A filter of simplices $\{Z_k,\ k = 0, 1, 2, \ldots\}$, where every $Z_{k+1}$ is a child of $Z_k$ via a bisection of ratio no less than a constant $\alpha \in ]0, 1/2]$, is exhaustive.*

Let $\Omega$ be a set of $I\!R^n$. A finite collection $\Theta$ of simplices covering $\Omega$ is referred to as a *simplicial net* for $\Omega$ and a simplicial net for $\Omega$ is said to be a *refinement* of another net if it is obtained by subdividing one or more members of the latter and replacing them with their partitions. A *simplicial subdivision process* for $\Omega$ is a sequence of nets $\Theta_0, \Theta_1, \ldots$, each of which, except the first one, is a refinement of its predecessor.

**Proposition 4.3.5** *An infinite simplicial subdivision process $\Theta_0, \Theta_1, \ldots,$ generates at least one filter.*

A simplicial subdivision process $\Theta_0, \Theta_1, \ldots,$ is *exhaustive* if every filter $\{Z_k,\ k = 0, 1, 2, \ldots\}$ generated by the process is exhaustive. From Corollary 4.3.4 it follows that a subdivision process consisting only of bisections of ratio at least $\alpha \in ]0, 1/2]$ is exhaustive.

## 4.3.2 Prismatical subdivision processes

In this subsection the properties of the prismatical subdivision processes are carried out (by using the simplicial subdivision properties above-mentioned), and *the basic prismatical subdivision property* is stated and proved.

Let $Z$ be an $n$-simplex in $I\!R^n$. The set

$$T(Z) := \{(x,\ t) \in I\!R^n \times I\!R :\ x \in Z\}$$

is referred to as a *simplicial prism* of base $Z$. All simplicial prisms have $n+1$ edges that are parallel lines to the $t$-axis. Each edge passes through the $n+1$ vertices of $Z$. Then, each radial subdivision $Z^1, \ldots, Z^r$, of the simplex $Z$ via a point $z \in Z$ induces a *prismatical subdivision* of the prism $T(Z)$ in subprisms, $T(Z^1), \ldots, T(Z^r)$, via the line through $z$ parallel to the $t$-axis. A prismatical subdivision for $T(Z)$ is said to be a *bisection* of ratio $\rho$ if it is induced by a bisection of ratio $\rho$ of $Z$. A filter of simplices

$$Z_1 \supset Z_2 \supset \ldots \supset Z_k \supset \ldots$$

induce a *filter of prisms*,

$$T_1 \supset T_2 \supset \ldots \supset T_k \supset \ldots$$

with $T_k = T(Z_k)$, $k = 1, 2, \ldots$. Also, every $T_{k+1}$ is called a *child* of $T_k$. Moreover, a filter of prisms is said to be *exhaustive* if it is induced by an exhaustive filter of simplices, i.e., $T_\infty = \cap_{k=0}^{\infty} T_k$ is a parallel line to the $t$-axis.

**Proposition 4.3.6** *(The basic prismatical subdivision property) Consider*

$$C := \{(x,t) \in I\!R^n \times I\!R : h(x) - t \leq 0)\},$$

*as defined in (4.2). Let $\{T_k,\ k = 0,1,2,\ldots\}$ be a filter of prisms (with $n+1$ edges) and let $z_k,\ k = 0,1,2,\ldots$ be the subdivision points in the simplex $Z_k,\ k = 0,1,2,\ldots$. Let $q_k,\ k = 0,1,2,\ldots$ be the points where the parallel line to the $t$-axis through the points $z_k,\ k = 0,1,2,\ldots$ meets the simplex spanned by the $n+1$ intersection points of the edges of $T_k$ with $\partial C$. We assume that:*

*i) For infinitely many $k$, $T_{k+1}$ is a child of $T_k$ in a bisection of ratio no less than a constant $\alpha > 0$,*

*ii) For all other $k$, $T_{k+1}$ is a child of $T_k$ in a subdivision via the parallel line to the $t$-axis through the point $z_k \in Z_k$.*

*Then, at least one accumulation point $q$ of the sequence $\{q_k,\ k = 0,1,2,\ldots\}$ satisfies $q \in \partial C$.*

Proof: Let $Z_k$ be the $n$-simplex $Z_k = [v_k^1, \ldots, v_k^{n+1}]$ and let $q_k^i$ be the point where the parallel line to the $t$-axis through the point $v_k^i$ meets $\partial C$, so $q_k \in [q_k^1, \ldots, q_k^{n+1}]$. Since $Z_1$ is bounded, by taking a subsequence if necessary, we may assume that

$$\{v_{k_s}^i\} \to v^i (s \to \infty),\ i = 1, \ldots, n+1.$$

We thus obtain $Z_\infty = \text{conv}\{v^1, \ldots, v^{n+1}\}$. Nevertheless, $v^1, \ldots, v^{n+1}$ may not be affinely independent, i.e., vert $(Z_\infty) \neq \{v^1, \ldots, v^{n+1}\}$. There is no loss of generality in assuming that $Z_\infty$ is a $p-1$-simplex, $0 < p \leq n+1$, which is expressed in the form $Z_\infty = [v^1, \ldots, v^p]$. Hence, we can write

$$v^{p+j} = \sum_{i=1}^{p} \alpha_i^j v^i, \tag{4.7}$$

where $\sum_{i=1}^{p} \alpha_i^j = 1,\ \alpha_i^j \geq 0,\ i = 1, \ldots, p$, for all $j = 1, \ldots, n-p+1$. On the other hand, we know that

$$z_{k_s} = \sum_{i=1}^{n+1} \lambda_i^{k_s} v_{k_s}^i, \tag{4.8}$$

with $\sum_{i=1}^{n+1} \lambda_i^{k_s} = 1$ and $\lambda_i^{k_s} \geq 0,\ i = 1, \ldots, n+1$. From Theorem 4.3.3, at least one accumulation point of the sequence $\{z_k\}$ is a vertex of $Z_\infty$. Suppose that $\{z_{k_s}\} \to v^1 (s \to \infty)$. By letting $s \to \infty$, we obtain $\{\lambda_i^{k_s}\} \to \lambda_i$, with $\sum_{i=1}^{n+1} \lambda_i = 1$ and $\lambda_i \geq 0,\ i = 1, \ldots, n+1$. Thus, we have

$$z_{k_s} = \sum_{i=1}^{n+1} \lambda_i^{k_s} v_{k_s}^i \to v^1 = \sum_{i=1}^{n+1} \lambda_i v^i. \tag{4.9}$$

Replacing the right hand side of the second equality in the expression (4.9) with (4.7) we have

$$v^1 = \lambda_1 v^1 + \ldots + \lambda_p v^p + \lambda_{p+1} \left( \sum_{k=1}^{p} \alpha_k^1 v^k \right) + \ldots + \lambda_{n+1} \left( \sum_{k=1}^{p} \alpha_k^{n-p+1} v^k \right). \tag{4.10}$$

By calculating and rearranging terms, we can rewrite (4.10) in the form

$$\begin{array}{rcl} v_1 & = & (\lambda_1 + \lambda_{p+1}\alpha_1^1 + \ldots + \lambda_{n+1}\alpha_1^{n-p+1})v^1 \\ & & +(\lambda_2 + \lambda_{p+1}\alpha_2^1 + \ldots + \lambda_{n+1}\alpha_2^{n-p+1})v^2 \\ & & + \ldots \\ & & +(\lambda_p + \lambda_{p+1}\alpha_p^1 + \ldots + \lambda_{n+1}\alpha_p^{n-p+1})v^p. \end{array} \tag{4.11}$$

In the second term of (4.11) we have the barycentric coordinates of the vertex $v^1$ in aff $\{v^1, \ldots, v^p\}$ (affine hull of the set $\{v^1, \ldots, v^p\}$). Since a point is solely represented by its barycentric coordinates, then we have

$$\begin{array}{rcl} 1 & = & \lambda_1 + \lambda_{p+1}\alpha_1^1 + \ldots + \lambda_{n+1}\alpha_1^{n-p+1}, \\ 0 & = & \lambda_2 + \lambda_{p+1}\alpha_2^1 + \ldots + \lambda_{n+1}\alpha_2^{n-p+1}, \\ & & \ldots \\ 0 & = & \lambda_p + \lambda_{p+1}\alpha_p^1 + \ldots + \lambda_{n+1}\alpha_p^{n-p+1}. \end{array} \tag{4.12}$$

We can see that each summand in the second term of the equalities in (4.12) is a nonnegative number. Then, we can deduce the following:

1. $\lambda_s = 0,\ s = 2, \dots, p$.
2. From $\lambda_{p+k}\alpha_s^k = 0,\ k = 1, \dots, n-p+1,\ s = 2, \dots, p$, and $\sum_{s=1}^{p} \alpha_s^k = 1$, we have
$$\lambda_{p+k} = \lambda_{p+k} \sum_{s=1}^{p} \alpha_s^k = \sum_{s=1}^{p} \lambda_{p+k}\alpha_s^k = 0,\ k = 1, \dots, n-p+1.$$
3. As a consequence of 1 and 2 we have $\lambda_1 = 1$.

Finally, we obtain

$$\left(\lambda_1^{k_s}, \dots, \lambda_{n+1}^{k_s}\right) \to (1, 0, \dots, 0) \text{ by letting } s \to \infty. \tag{4.13}$$

On the other hand, from the expression

$$q_{k_s} = \sum_{i=1}^{n+1} \lambda_i^{k_s} \left(v_{k_s}^i, h(v_{k_s}^i)\right) = \left(\sum_{i=1}^{n+1} \lambda_i^{k_s} v_{k_s}^i, \sum_{i=1}^{n+1} \lambda_i^{k_s} h(v_{k_s}^i)\right) = \left(z_{k_s}, \sum_{i=1}^{n+1} \lambda_i^{k_s} h(v_{k_s}^i)\right), \tag{4.14}$$

we have

$$q_{k_s} = \left(z_{k_s}, \sum_{i=1}^{n+1} \lambda_i^{k_s} h(v_{k_s}^i)\right) \to q = \left(v^1, h(v^1)\right) \text{ by letting } s \to \infty, \tag{4.15}$$

which proves that $q \in \partial C$. $\square$

## 4.4 Outline of the method

A brief outline of the algorithm is given in this section. Properties which prove its convergence will be demonstrated later (see Section 4.7).

At each iteration the procedure involves some basic operations as follows:

- *Branching:* a selected prism $T(Z)$ is divided into a finite number of subprisms by using a simplicial partition of $Z$.
- *Outer approximation:* a new polyhedral $P_k$ is obtained by using a cutting plane to cut off a part of $P_{k-1}$ in such a way that a sequence of convex polyhedral $P_0, P_1, \dots$ is constructed satisfying
$$P_0 \supset P_1 \supset \dots \supset P_k \supset \dots \supset D_{\alpha^*} \supset D_{\alpha^*} \backslash \text{int } C.$$

- *Delete rule:* prisms are deleted which contain feasible solutions worse than the best obtained so far.

The basic operations used in the algorithm are related to the optimum $\mu(T)$ and the optimizer $(x(T), t(T))$ of the linear program

$$\begin{aligned} \mu(T) = \text{maximize} \quad & a^t x - t - \rho \\ \text{subject to:} \quad & (x,t) \in T \cap P, \end{aligned} \tag{4.16}$$

where $T := T(Z)$ is a prism with $Z = [v^1, \dots, v^{n+1}]$, $P$ is the current convex polyhedral and $a^t x - t - \rho$ is the unique hyperplane passing through the points $(v^i, h(v^i)), i = 1, \dots, n+1$. The optimizer of (4.16) is the point of the polyhedral $T \cap P$ with the greatest distance to the hyperplane $a^t x - t - \rho$. When $\mu(T) \le 0$ then $T(Z)$ is deleted because $T \cap P \subset C$ (see Lemma 4.6.2). On the other hand, if a prism $T$ is selected for division (see Remark 1 at the end of this section), which is associated with the largest value $\max\{\mu(T) : \mu(T) > 0\}$, a refined partition of $T$ is constructed and a new cut is added to the polyhedral $P$ to obtain a new convex polyhedral. From the solution $(x(T), t(T))$ the new point $(\bar{x}, \bar{t})$ is obtained with $\bar{x} := x(T)$ and $\bar{t} := \max\{h(x(T)), g(x(T))\}$. If $h(x(T)) - t(T) > 0$, a new cut $l(x,t)$ is added through the point $(\bar{x}, \bar{t})$ to obtain a new convex polyhedral

$$P \cap \{(x,t) : l(x,t) \le 0\}.$$

The cut $l(x,t) := (x - \bar{x})^T p - t + c$ is defined as follows:

- if $h(x(T)) \ge g(x(T))$, then $p$ is a subgradient of the function $f(x)$ at the point $x(T)$ and $c = h(x(T))$,
- if $h(x(T)) < g(x(T))$, then $p$ is a subgradient of the function $g(x)$ at the point $x(T)$ and $c = g(x(T))$.

The procedure continues until all generated prisms have been deleted. At this stage, we have

i) a partition $\{T_i, i = 1, 2, \dots\}$ of the initial prism $T_0 = \cup_{i=1}^{\infty} T_i$ with $\mu(T_i) < 0, i = 1, 2, \dots$

ii) a sequence of convex polyhedrons $P_0, P_1, \dots$ such that

$$P_0 \supset P_1 \supset \dots \supset D_{\alpha^*} \supset D_{\alpha^*} \backslash \text{int } C,$$

where each polyhedral $P_j, j = 1, 2 \dots$ is obtained by using a cutting plane to cut off a part of $P_{j-1}, j = 1, 2 \dots,$

iii) a bounded sequence of points $\{(\bar{x}_k, \bar{t}_k), k = 0, 1, \ldots\}$ so that there exists a subsequence $\{k_i\}$ such that $\{(\bar{x}_{k_i}, \bar{t}_{k_i})\} \to (x^*, t^*)$.

Hence, we have that for all $i = 1, 2, \ldots$ there exists an index $k$ such that $T_i \cap P_k \subset C$. Since $T_i \cap D_{\alpha^*} \subset T_i \cap P_k \subset C$ for all $i = 1, 2, \ldots$ we can write

$$C \supset \cup_{i=1}^{\infty}(T_i \cap D_{\alpha^*}) = \cup_{i=1}^{\infty}(T_i) \cap D_{\alpha^*} = T_0 \cap D_{\alpha^*} = D_{\alpha^*}. \tag{4.17}$$

On the other hand, the point $(x^*, t^*)$ as defined in iii) satisfies

$$(x^*, t^*) \in D_{\alpha^*} \backslash \text{int } C \text{ and } \alpha^* = f(x^*) - t^*. \tag{4.18}$$

From the Lemma 4.2.3 we can deduce that the point $(x^*, t^*)$ is a global minimizer.

## 4.5 Initialization of the algorithm

Consider $T_0 := T(Z_0)$ an initial prism, with $Z_0 := [v_0^1, \ldots, v_0^{n+1}]$ a $n$-simplex of $I\!R^n$ which contains the polytop $\{x \in I\!R^n : Ax \leq b\}$. Let $P_0$ be an initial convex polyhedral

$$P_0 := \{(x, t) : Ax \leq b, l_i(x, t) \leq 0, i = 1, \ldots, n + 1\},$$

with

$$l_i(x, t) := (x - v_0^i)^T p_i - t + c_i, i = 1, \ldots, n + 1$$

defined in the form:

- if $h(v_0^i) \geq g(v_0^i)$, then $p_i$ is a subgradient of the function $f(x)$ at the vertex $v_0^i \in Z$ and $c_i = h(v_0^i)$ (see the line $AD$ in Figure 4.4 through the point $A = (v1, h(v1)))$,
- if $h(v_0^i) < g(v_0^i)$, then $p_i$ is a subgradient of the function $g(x)$ at the vertex $v_0^i \in Z$ and $c_i = g(v_0^i)$ (see the line $CD$ in Figure 4.4 through the point $C = (v2, g(v2)))$.

On the other hand, let $H_0$ be the uniquely defined hyperplane through the points $(v_0^i, h(v_0^i)),\ i = 1, 2, \ldots, n + 1$ (see the line $AB$ in Figure 4.4), i.e.,

$$H_0 := \{(x, t) \in I\!R^n \times I\!R : a_0^t x - t - \rho_0 = 0\},$$

with $a_0 \in I\!R^n$ and $\rho_0 \in I\!R$ (note that $a_0^t v_0^i - h(v_0^i) - \rho_0 = 0,\ i = 1, 2, \ldots, n + 1)$.

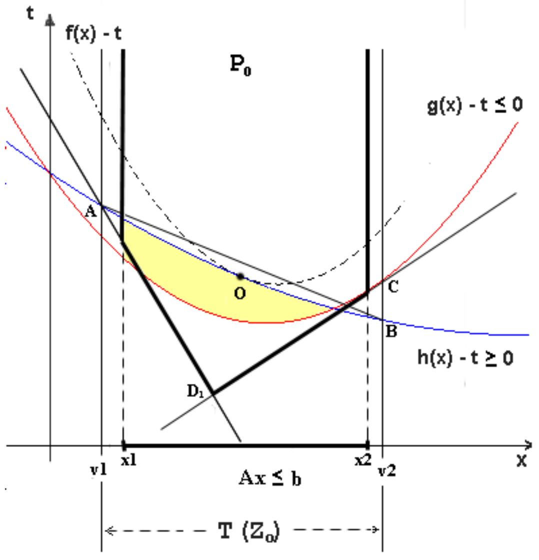

Figure 4.4: *In this graphic the segment* $[x_1, x_2]$ *represents the linear constraints* $Ax \leq b$*. The line through the points A, B is the uniquely defined hyperplane through the points* $(v1, h(v1))$ *and* $(v2, h(v2))$*. The space within the line in bold represents the initial polyhedral* $P_0$*. Point D is the optimizer for the program (4.16) with the feasible set* $T_0 \cap P_0$ *with* $T_0 = T(Z_0)$ *and* $Z_0 := [v_1, v_2]$*. Point O is the desired global optimizer.*

**Example 4.5.1** *In Figure 4.4 the initial set* $T_0 \cap P_0$ *is defined by the lines AB, AD, BD and the vertical lines through the points* $x1$ *and* $x2$ *respectively. Moreover, point D is the optimizer of the linear program (4.16). Since in Figure 4.4 we have* $\mu(T_0) > 0$ *then we can subdivide the prism* $T_0$ *by using point D. Hence, we have the prismatical subdivision* $T_0 = T(Z_1) \cup T(Z_2)$ *(see Figure 4.5 (a) for* $T(Z_1)$ *and Figure 4.5 (b) for* $T(Z_2)$*). Thus, we can see that* $\mu(T_1) \leq 0$ *and the subdivision* $T(Z_1)$ *can be deleted, and that* $\mu(T_2) > 0$ *and* $T(Z_2)$ *can again be subdivided by using the new point F (see Figure 4.5 (b)).*

## 4.6 Outer approximation process

Let $\psi^\epsilon(x)$ be the proper convex function defined as

$$\psi^\epsilon(x) := \max\,\{h(x) + \epsilon,\ g(x)\}, \tag{4.19}$$

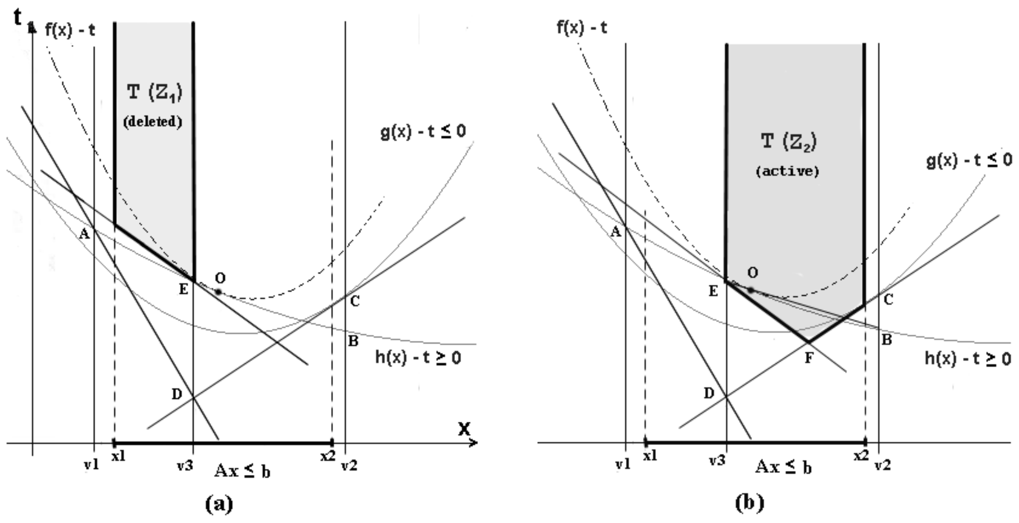

Figure 4.5: The prismatic subdivision process of the prism $T_0$ in subprisms $T(Z_1)$ and $T(Z_2)$

where, in this section, $\epsilon$ represents a nonnegative real number and $\psi := \psi^0$.

**Lemma 4.6.1** *Consider $M := \{x_1, \ldots, x_m\}$ a finite set of points in $I\!R^n$. Let $p_i$ be a subgradient of the function $f(x)$ at the point $x_i \in M$ if $h(x_i) + \epsilon \geq g(x_i)$ or, else let $p_i$ be a subgradient of the function $g(x)$ at point $x_i \in M$ if $h(x_i) < g(x_i)$. Thus, the function*

$$\psi_M^\epsilon(x) := max\{(x - x_i)^T p_i + \psi^\epsilon(x_i),\ i = 1, 2, \ldots, m\} \tag{4.20}$$

*(see Figure 4.6) satisfies the following properties:*

*i) $\psi_M^\epsilon(x)$ is piece-wise linear and proper convex function on $I\!R^n$,*

*ii) $P_M := \{(x,t) \in I\!R^n \times I\!R : \psi_M^\epsilon(x) - t \leq 0\}$ is a polyhedron.*

*If $N$ is a finite set in $I\!R^n$ and $M \subseteq N$ then*

*iii) $\psi_M^\epsilon(x) \leq \psi_N^\epsilon(x)\ \forall x \in I\!R^n$,*

*iv) $P_M \supseteq P_N$.*

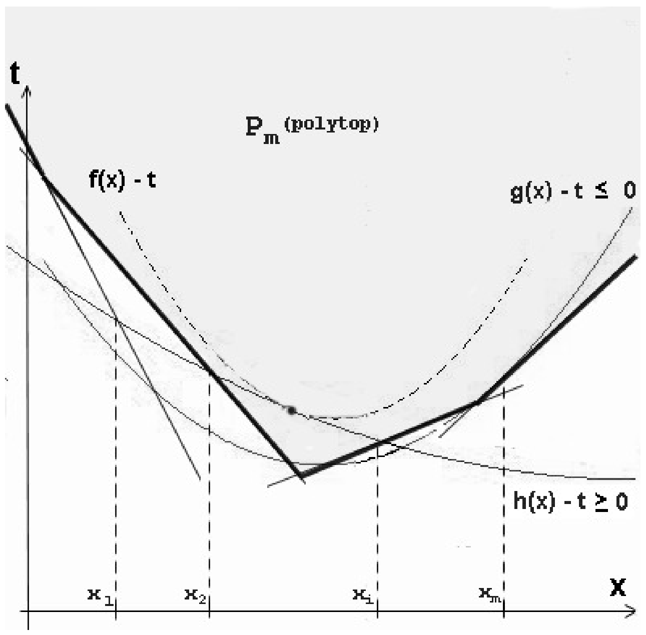

Figure 4.6: The piece-wise linear and proper convex function $\psi^{\epsilon}_{M}(x)$

PROOF: Obvious. $\square$

Consider the filter of prisms $T_0 \supset T_1 \supset \ldots \supset T_k \supset \ldots$ where every $T_k := T(Z_k)$ is a prism, which is induced by a radial subdivision of the simplex $Z_{k-1}$ (where $T_{k-1} = T(Z_{k-1})$) via a suitable point of $Z_{k-1}$ and let $z_{k-1} \in Z_{k-1}$ be the point $z_{k-1} = x(Z_{k-1})$ obtained by solving the linear program (4.16). Let

$$P_k = \{(x,\ t) \in I\!R^n \times I\!R : Ax \leq b,\ \psi^{\epsilon}_{M_k}(x) - t \leq 0\}$$

be the polyhedral generated from the set $M_k$, which contains the vertices of $Z_0$ and the points $z_0,\ z_1, \ldots, z_{k-1}$ generated in the subdivision process. Then, the sequence of convex polyhedrons $P_i, i = 0, 1, 2, \ldots$ defined as before satisfies

$$P_0 \supset P_1 \supset \ldots \supset P_k \supset \ldots \supset D_{\alpha^*} \backslash \text{int}\ C_\epsilon.$$

In what follows, the function $\psi^{\epsilon}_{M_k}(x)$ will be denoted by $\psi^{\epsilon}_{k}(x)$.

**Lemma 4.6.2** *Let $(x(T_k), t(T_k))$ and $\mu(T_k)$ be the optimizer and the optimum respectively of the linear program*

$$max\ \{a_k^t x - t - \rho_k :\ (x,t) \in T_k \cap P_k\}, \tag{4.21}$$

*where* $a_k^t x - t - \rho_k = 0$ *is the hyperplane passing through the points* $(v_k^i, h(v_k^i))$, $i = 1, 2, \ldots, n+1$ *with* $Z_k = [v_k^1, \ldots, v_k^{n+1}]$. *Thus, we have*

*i) if* $\mu(T_k) > 0$ *then* $(x(T_k), t(T_k))$ *doesn't lie on any edge of* $T_k$,

*ii) if* $\mu(T_k) \leq 0$ *then* $T_k \cap P_k \subset C$.

PROOF:

i) Suppose that the optimizer of (4.21) lies on an edge of $T_k$. Hence, there exists a vertex $v^i \in \{v_k^1, \ldots, v_k^{n+1}\}$ such that $(v^i, t_i)$ satisfies $\mu(T_k) = a_k^t v^i - t_i - \rho_k > 0$. On the other hand, the vertex $v^i$ satisfies that $\psi_k^\epsilon(v^i) \geq \psi(v^i) \geq h(v^i)$ so $\psi_k^\epsilon(v^i) - h(v^i) \geq 0$ and $a_k^t v^i - h(v^i) - \rho_k = 0$. Thus, $h(v^i) > t_i$ and $\psi_k^\epsilon(v^i) - t_i > \psi_k^\epsilon(v^i) - h(v^i) \geq 0$ which implies that $(v^i, t_i) \notin P_k$, which is a contradiction.

ii) Let $(x^o, t^o) \in T_k \cap P_k$ be a feasible point of the linear program (4.21). Then, from the hypothesis in ii), we deduce that $a_k^t x^o - t^o - \rho_k \leq \mu(T_k) \leq 0$ so that $(x^o, t^o) \in T_k \cap P_k \cap H_+$. From $x^o \in Z_k$ we can write the expression $x^o = \sum_{i=1}^{n+1} \lambda_i^o v^i$ with $\sum_{i=1}^{n+1} \lambda_i^o = 1$. From the convexity of the function $h(x)$ we obtain the inequality $h(x^o) = h(\sum_{i=1}^{n+1} \lambda_i^o v^i) \leq \sum_{i=1}^{n+1} \lambda_i^o h(v^i)$. Finally, from the definition of the hyperplane $H$ we know that each point $(v^i, h(v^i))$, $i = 1, \ldots, n+1$ verifies the equality $a_k^t v^i - h(v^i) - \rho_k = 0$ so that we have $h(v^i) = a_k^t v^i - \rho_k$. Hence, we can write

$$h(x^o) - t_o \leq \sum_{i=1}^{n+1} \lambda_i^o h(v^i) - t_o = \sum_{i=1}^{n+1} \lambda_i^o (a_k^t v^i - \rho_k) - t_o \leq$$

$$\leq a_k^t \sum_{i=1}^{n+1} \lambda_i^o v^i - \rho_k \sum_{i=1}^{n+1} \lambda_i - t_o = a_k^t x^o - \rho_k - t_o \leq \mu(T_k) \leq 0,$$

which proves that $(x^o, t^o) \in C$. □

## 4.7 The algorithm and its convergence

In this section the algorithm will be described and its convergence will be proved. In each iteration $k$ of the algorithm, we denote by $\Phi_k$ the set of all the current prismatical subdivisions, by $\Psi_k$ the prismatical subdivisions of the prism associated with the biggest value $\mu(Z^*)$ and by $\Gamma_k$ the set of the prismatical subdivisions with $\mu(Z) > 0$.

```
Initialization:
$\alpha_0 \leftarrow +\infty$; $\Gamma_0 \leftarrow \emptyset$; $\Phi_0 \leftarrow \emptyset$;
Determine a simplex $Z_0 \supset \{x \in I\!R^n : Ax \leq b\}$, its vertex set $V_0$ and the prism
$T_0 = T(Z_0)$;
When one feasible solution $(\bar{x}, \bar{t})$ is at hand and it is the best, then $\alpha_0 \leftarrow f(\bar{x}) - \bar{t}$;
$M_0 \leftarrow V_0 \cup \{\bar{x}\}$; $P_0 \leftarrow \{(x,\ t) \in I\!R^n \times I\!R : Ax \leq b,\ \psi_0^\epsilon(x) - t \leq 0\}$;
Solve $\mu(T_0) = \max\{a_0^t x - t - \rho_0 :\ (x,\ t) \in T_0 \cap P_0\}$, with $(x(T_0), t(T_0))$ a basic
optimizer;
if $\mu(T_0) > 0$ then $\Phi_0 \leftarrow \{T_0\}$; $\Gamma_0 \leftarrow \{T_0\}$; $T_0^* \leftarrow T_0$; $Z_0^* \leftarrow Z_0$; end if
stop $\leftarrow$ false; $k \leftarrow 0$;

while stop=false do
    if $\Gamma_k = \emptyset$ then
        if $\alpha_k = +\infty$ then the problem is infeasible;
        else $(\bar{x}, \bar{t})$ is an optimizer; end if
        stop$\leftarrow$ true;
    else
        if $h(x(T)) - t(T) + \epsilon > 0$ for some $T \in \Gamma_k$ then
            $M_{k+1} \leftarrow M_k \cup \{x(T)\}$;
            $P_{k+1} \leftarrow \{(x,\ t) \in I\!R^n \times I\!R : Ax \leq b,\ \psi_{k+1}^\epsilon(x) - t \leq 0\}$;
            if $\psi(x(T)) = h(x(T))$ and $f(x(T)) - h(x(T)) < \alpha_k$ then
                $(\bar{x},\ \bar{t}) \leftarrow (x(T), h(x(T))$; $\alpha_{k+1} \leftarrow f(\bar{x}) - \bar{t}$;
            end if
        end if
        Split $Z_k^*$ via the chosen normal rule (see remark 1) to obtain a partition
        $T_{k_1}, \ldots, T_{k_r}$ of $T_k^*$; $\Psi_{k+1} \leftarrow \{T_{k_1}, \ldots, T_{k_r}\}$;
        For all prism $T_{k_i} \in \Psi_{k+1}$ solve the linear program:
        $\mu(T_{k_i}) = \max\{a_{k_i}^t x - t - \rho_{k_i} :\ (x,\ t) \in T_{k_i} \cap P_{k+1}\}$,
        with $(x(T_{k_i}), t(T_{k_i}))$ basic optimizer;
        $\Phi_{k+1} \leftarrow (\Gamma_k \setminus \{T_k^*\}) \cup \Psi_{k+1}$; $\Gamma_{k+1} \leftarrow \{T \in \Phi_{k+1} :\ \mu(T) > 0\}$;
        Choose $T_{k+1}^* \in$ argmax $\{\mu(T) :\ T \in \Gamma_{k+1}\}$;
    end if
    $k \leftarrow k + 1$;
end while
```

**Remark 1** *From the property i) of Lemma 4.6.2 it seems natural to subdivide the selected prism defined in the algorithm $T(Z_k^*)$ via the point $x(Z_k^*) \in Z_k^*$. A subdivision of this kind for $T(Z_k^*)$ will be referred to as an $\omega$-subdivision and the strategy* *of choosing $\omega$-subdivisions in all iterations is called $\omega$-strategy. Until now, attempts to prove the convergence of procedures using $\omega$-strategy have been unsuccessful. This situation has caused a shift to the use of the alternative bisection strategy which is always convergent and takes the midpoint of a longest edge of selected simplex $Z_k^*$ to subdivide $T(Z_k^*)$. Thus, as far as the choice of a subdivision strategy is concerned, there is a conflict between convergence and efficiency. Although the subdivision strategy guarantees convergence it is usually slow. On taking into account the Proposition 4.3.6 we find that the way out of this difficulty is to use a mixed strategy which combines $\omega$-subdivisions with infinite bisections. A subdivision process is said to be normal if it involves bisections mixed between $\omega$-subdivisions and the bisections occur infinitely in every filter generated by the process. For instance, every $N = 5$ $\omega$-subdivisions performed a bisection could be included (see [73] for additional information). From here on we will always use a normal subdivision strategy, also known as a normal rule.*

### 4.7.1 Convergence of the algorithm

Let $\sigma(x)$ be the proper convex function defined as

$$\sigma(x) := \max\,\{f(x) - \alpha^*,\ g(x)\}. \tag{4.22}$$

Each generated point $(x(T_k), t(T_k))$ in the algorithm will be denoted by $(x_k, t_k)$. Thus, for each generated point $(x_k, t_k)$ we can consider the cuts:

- $r_k(x,t) := p_k^T(x - x_k) - (t - \sigma(x_k))$ with $p_k$ a subgradient of the function $\sigma(x)$ at the point $x_k$,
- $s_k(x,t) := p_k^T(x - x_k) - (t - \psi^\epsilon(x_k))$ with $p_k$ a subgradient as defined in Lemma 4.6.1.

**Lemma 4.7.1** *Let $\{(x_k, t_k)\}$ be the sequence obtained in the algorithm by solving the linear problems (4.21). Thus, we have that $\psi^\epsilon(x_k) \leq \sigma(x_k)$ and the sequences $\{(x_k, t_k)\}$, $\{(x_k, \psi^\epsilon(x_k))\}$ and $\{(x_k, \sigma(x_k))\}$ are bounded.*

PROOF: We have either $\psi^\epsilon(x_k) = g(x_k)$ or $\psi^\epsilon(x_k) = h(x_k) + \epsilon$. When $\psi^\epsilon(x_k) = g(x_k)$ then obviously $\psi^\epsilon(x_k) \leq \sigma(x_k)$. Otherwise $\bar{t_k} = \psi^\epsilon(x_k) = h(x_k) + \epsilon$, so $(x_k, \bar{t_k})$ is a feasible point and $f(x_k) - \bar{t_k} \geq \alpha^*$. Then we can write $f(x_k) - \alpha^* \geq \bar{t_k} = h(x_k) + \epsilon$ and also $\psi^\epsilon(x_k) \leq \sigma(x_k)$.

On the other hand, the functions $\psi^\epsilon(x)$ and $\sigma(x)$ are continuous on the polytope defined by $Ax \leq b$ (which is a compact set). From $t_k < \psi^\epsilon(x_k) \leq \sigma(x_k)$ we can deduce that the sequences $\{(x_k, t_k)\}$, $\{(x_k, \psi^\epsilon(x_k))\}$ and $\{(x_k, \sigma(x_k))\}$ are bounded.

□

**Lemma 4.7.2** *The cuts $s_k(x,t)$ and $r_k(x,t)$ strictly separate each generated point $(x_k, t_k)$ (of the sequence $\{(x_k, t_k)\}$ obtained in the algorithm) from $D_{\alpha^*}$.*

Proof: Obviously, for all $(x,t) \in D_{\alpha^*}$ we have $\sigma(x) - t \leq 0$. Then $r_k(x,t) = p_k^T(x - x_k) - (t - \bar{t_k}) \leq \sigma(x) - t \leq 0$ by using the convexity of the function $\sigma(x) - t$. On the other hand, let $\bar{t_k} = \psi^\epsilon(x_k)$ and $(x,t) \in D_{\alpha^*}$. Then we can write

$$s_k(x,t) = p_k^T(x - x_k) - (t - \bar{t_k}) \leq \begin{cases} g(x) - t \leq 0 \\ \text{or} \\ f(x) - \alpha^* - t \leq 0. \end{cases} \tag{4.23}$$

Moreover, from $t_k < \psi^\epsilon(x_k) \leq \sigma(x_k)$ we obtain

$$r_k(x_k, t_k) = -(t_k - \sigma(x_k)) \geq -(t_k - \psi^\epsilon(x_k)) = s_k(x_k, t_k) > 0, \tag{4.24}$$

which proves the lemma. □

From Lemma 4.7.1 we know that the sequence $\{(x_k, t_k)\}$ is bounded and that there exits a subsequence $\{k_i\}$ such that $\{(x_{k_i}, t_{k_i})\} \to (x^*, t^*)$.

**Lemma 4.7.3** *The following assertions are true:*

*i)* $r_{k_i}(x_{k_i}, t_{k_i}) = -(t_{k_i} - \sigma(x_{k_i})) \to 0$ *when* $i \to +\infty$ *and* $\sigma(x^*) = t^*$,

*ii)* $s_{k_i}(x_{k_i}, t_{k_i}) = -(t_{k_i} - \psi^\epsilon(x_{k_i})) \to 0$ *when* $i \to +\infty$ *and* $\psi^\epsilon(x^*) = t^*$.

Proof: From (4.24) we have

$$\lim_{i \to +\infty} r_{k_i}(x_{k_i}, t_{k_i}) \geq 0. \tag{4.25}$$

On the other hand, if $k$ is fixed we have $r_k(x_{k_i}, t_{k_i}) \leq 0$ for all $k_i > k$. Then, from $i \to +\infty$, we obtain $r_k(x^*, t^*) \leq 0$. Otherwise, for all $k$, we can write $r_k(x,t) = r_k(x^*, t^*) + p_k^T(x - x^*) - (t - t^*)$. Hence, the relationship

$$r_{k_i}(x_{k_i}, t_{k_i}) \leq p_{k_i}^T(x_{k_i} - x^*) - (t_{k_i} - t^*) \tag{4.26}$$

can be obtained. Moreover, we know that $\{p_k\}$ is a bounded sequence (see [74] Theorem 2.6). Then, letting $i \to +\infty$ in (4.26) we obtain

$$\lim_{i \to +\infty} \quad r_{k_i}(x_{k_i}, t_{k_i}) \quad \leq 0. \tag{4.27}$$

From (4.25) and (4.27) we can deduce that $r_{k_i}(x_{k_i}, t_{k_i}) \to 0$ and as a direct consequence we have $\sigma(x^*) = t^*$. The same proof holds true by using $s_{k_i}(x_{k_i}, t_{k_i})$ in place of $r_{k_i}(x_{k_i}, t_{k_i})$, which proves the lemma. □

From the preceding lemmas and by using the Proposition 4.3.6 we can enounce the following result.

**Proposition 4.7.4** *The algorithm can only be infinite if $\epsilon = 0$ and in this case any accumulation point of the sequence $\{(x_k, t_k)\}$ is an optimizer for the program (4.3). Moreover, if $\epsilon > 0$ then the algorithm is finite and an $\epsilon$-approximate optimizer can be obtained.*

PROOF: Let $(x^*, t^*)$ be an accumulation point of the sequence $\{(x_k, t_k)\}$. From $h(x_k) - t_k > 0$ we obtain $h(x^*) - t^* \geq 0$. On the other hand, we know that $h(x^*) - t^* + \epsilon = 0$, which is a contradiction unless $\epsilon = 0$. In this case, every point $(x^o, t^o) \in D_{\alpha^*}$ satisfies $g(x^o) - t^o \leq 0$, $f(x^o) - t^o \leq \alpha^*$ and $Ax^o \leq b$. Suppose that $h(x^o) - t^o > 0$. This implies that $f(x^o) - t^o \leq \alpha^* < f(x^o) - t^o$, which is a contradiction. Thus, the point $(x^o, t^o)$ must satisfy $h(x^o) - t^o + \epsilon \leq 0$ and therefore $(x^o, t^o) \in C$, i.e., $D_{\alpha^*} \subset C$. The optimality criterion, together with the regularity assumption, implies that $(x^*, t^*)$ is a global optimizer with optimum $\alpha^*$. □

## 4.8 Appropriate linear program routine by using barycentric coordinates

From the point of view of implementation it is necessary to rewrite the linear program (4.16) in a more suitable way by using barycentric coordinates. Consider $Z = [v^1, \ldots, v^{n+1}]$ and $(x, t) \in T(Z) \cap P$. Then, we have that $x \in Z$ and we can write $x = \sum_{i=1}^{n+1} \lambda_i v^i$ with $\lambda_i \geq 0, i = 1, \ldots, n+1$ and $\sum_{i=1}^{n+1} \lambda_i = 1$. Equivalently, we can write $x = V\lambda$ where $V \in \mathcal{M}(n, n+1)$ has as columns the coordinates of the vertices $v^1, \ldots, v^{n+1}$ and $\lambda = (\lambda_1, \ldots, \lambda_{n+1})^t$. By using the barycentric coordinates the objective function of the linear program (4.16) can be rewritten as follows.

$$\begin{array}{rcll}
a^t x - t - \rho & = & a^t(\sum_{i=1}^{n+1} \lambda_i v^i) - t - \rho & = \\
 & = & \sum_{i=1}^{n+1} \lambda_i (a^t v^i) - t - \rho & = \\
 & = & \sum_{i=1}^{n+1} \lambda_i (h(v^i) + \rho) - t - \rho & = \\
 & = & \sum_{i=1}^{n+1} \lambda_i h(v^i) + (\sum_{i=1}^{n+1} \lambda_i)\rho - t - \rho & = \\
 & = & \sum_{i=1}^{n+1} \lambda_i h(v^i) + \rho - t - \rho & = \sum_{i=1}^{n+1} \lambda_i t_i - t.
\end{array}$$

On the other hand, the polyhedral $P$ can be expressed in the form

$$P := \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : Mx + rt \leq s\}$$

with $M \in \mathcal{M}(m,n)$ and $r, s \in \mathbb{R}^m$. Hence, we can rewrite the set of constraints with the expression

$$Mx + rt = MV\lambda + rt \leq s,$$

where $\sum_{i=1}^{n+1} \lambda_i = 1$ and $\lambda_i \geq 0, i = 1, \ldots, n+1$. Thus, the linear program (4.16) can be expressed in a more suitable form as follows.

$$\begin{array}{rc}
\text{maximize} & \sum_{i=1}^{n+1} \lambda_i t_i - t \\
\text{subject to:} & MV\lambda + rt \leq s, \\
 & \sum_{i=1}^{n+1} \lambda_i = 1, \\
 & \lambda_i \geq 0, i = 1, \ldots, n+1.
\end{array} \tag{4.28}$$

The main advantage of the linear program (4.28) is the importance of the vertices $V = \{v^1, \ldots, v^{n+1}\}$ (of the current subdivision) that permit a more efficient calculus of the objective function and of the set of constraints. We can calculate the coefficients of the objective function by using the expression $t_i = h(v^i), i = 1, \ldots, n+1$. Moreover, we can see that the matrix $M$ and the vectors $r$ and $s$ (which define the current polyhedral $P$) have the same expression for every linear program. Hence, the set of constraints only depends on the vertices $V$. Note that in an outer approximation procedure cuts are always *conjunctive*, i.e., the polyhedron resulting from the cuts is always the intersection of all the cuts performed.

## 4.9 Differences between the algorithm put forward and other existing algorithms

To conclude this Chapter it only remains to add that the adapted algorithm is more general that it seems because it can be used to solve d.c. programming problems with linear constraints of the form

$$\begin{array}{rc} \text{minimize} & f(x) - h(x) \\ \text{subject to:} & Ax \leq b, \end{array} \tag{4.29}$$

where $A$ is a real $m \times n$ matrix, $b \in I\!R^m$ and $f(x)$ and $h(x)$ are proper convex functions on $I\!R^n$. By introducing an additional variable $t$ the program (4.29) can be transformed into the equivalent convex minimization problem subject to an additional reverse convex constraint

$$\begin{array}{rc} \text{minimize} & f(x) - t \\ \text{subject to:} & h(x) - t \geq 0, \\ & Ax \leq b. \end{array} \tag{4.30}$$

Hence, we can use our algorithm to solve (4.30) by taking $g(x) := -\infty$ in (4.1). Moreover, we can add convex constraints in a general sense to (4.29) and then we can solve d.c. programming problems in the form

$$\begin{array}{rc} \text{minimize} & f(x) - h(x) \\ \text{subject to:} & \varphi_i(x) \leq 0,\ i = 1, \ldots, m, \\ & Ax \leq b, \end{array} \tag{4.31}$$

where $\varphi_i(x), i = 1, \ldots, m$ are proper convex functions on $I\!R^n$. As before, by introducing an additional variable $t$ the program (4.31) can be transformed into the equivalent convex minimization problem subject to an additional reverse convex constraint

$$\begin{array}{rc} \text{minimize} & f(x) - t \\ \text{subject to:} & h(x) - t \geq 0, \\ & \varphi(x) \leq 0, \\ & Ax \leq b, \end{array} \tag{4.32}$$

with $\varphi(x) := \max\{\varphi_i(x),\ i = 1, \ldots, m\}$. When the point obtained by solving the linear program is infeasible (which can only happen if $\varphi(x(T)) > 0$), we add the constraint

$$l(x,t) := p^t(x - x(T)) + \varphi(x(T)), \tag{4.33}$$

where $p$ is a subgradient of the function $\varphi(x)$. This is a different way to solve (4.31) comparing with the algorithm proposed in Horst R. et al. [34] and Horst R. et al.

[33] which uses prismatical subdivisions but transforms (4.31) into an equivalent concave minimization problem.

On the other hand, the reasoning behind adding a constraint in the form described in (4.33) can also be applied to a programming problem expressed

$$\begin{array}{rc} \text{minimize} & f(x) - t \\ \text{subject to:} & g(x) - t \leq 0, \\ & h(x) - t \geq 0, \\ & \varphi(x) \leq 0, \\ & Ax \leq b, \end{array} \tag{4.34}$$

which is the result of adding a convex constraint $\varphi(x) \leq 0$ to the set of constraints of the program (4.1).

# Chapter 5

# How to improve a d.c. representation of a polynomial

## 5.1 Introduction

Given a d.c. function there are infinitely many d.c. representations, which express it as a difference of convex functions. From a computational point of view, what should it mean that a d.c. representation is "better" that another one?. Having solved the problem of finding a d.c. representation of a polynomial we then come up with another even more complicated problem, that is, if the computational efficiency depends on the d.c. representation of the functions. Hence, interesting questions can be formulated such as:

- Does exist a d.c. representation that improves the computational efficiency?
- If the answer to this question is affirmative, then what is the best d.c. representation of a d.c. function from a computational point of view and how can it be obtained?

This Chapter is devoted to introducing the necessary concepts for answering these questions which cannot be answered directly. We have found that the simplest approach is to work backwards from theoretical concepts, formulating a problem in such a way that a new d.c. representation of the polynomial functions can be obtained by using the concept of minimal d.c. representation of a polynomial (which will be explained below) in the normed space of the polynomials. In order to obtain such a minimal d.c. representation of a polynomial function, we state, in section

5.3, *the minimal norm problem* by using the concept of *least deviation decomposition* (*LDD*) described in Luc D.T. et al. [46]. After that, in Chapter 7, we will describe how these new d.c. representation improve the computational efficiency of the global optimization algorithm by reducing the number of iterations to find a global optimal solution. Notice, that the question whether or not the given minimal d.c. representation is the best d.c. representation from a computational viewpoint (as it seems to be) cannot be completely answered.

## 5.2 The normed space of the polynomials

Let $\mathbb{R}_m[x_1, ..., x_n]$ and $H_i[x_1, ..., x_n]$, $i = 0, 1, \ldots, m$ be the vector spaces of polynomials of degree less than or equal to $m$ and of homogeneous polynomials of degree $i$ respectively. Let $B^m := \{f_i(x), i \in I^m\}$ and $B_k := \{f_i(x), i \in I_k\}$ be the usual bases of the monomials in $\mathbb{R}_m[x_1, ..., x_n]$ and $H_k[x_1, ..., x_n]$ index-linked by the sets $I^m$ and $I_k$ respectively. Hence, each polynomial $z(x) \in \mathbb{R}_m[x_1, ..., x_n]$ can be written in the form $z(x) = \sum_{i \in I^m} a_i f_i(x)$ (and the same for polynomials in $H_k[x_1, ..., x_n]$). Both vector spaces can be defined as normed spaces by using the $p$-norms $\|z(x)\|_p := (\sum_{i \in I} \mid a_i \mid^p)^{1/p}, p = 1, 2, \ldots$(for $p = 2$ we have the Euclidean norm) and the $\infty$-norm $\|z(x)\|_\infty := \max\{\mid a_i \mid, i \in I\}$(Tchebychev norm). The notation $\|z(x)\|_{(p,k)}$ is used to indicate the $p$-norm in $H_k[x_1, ..., x_n]$ for all $p = 0, 1, 2, \ldots$ and $p = \infty$.

**Lemma 5.2.1** *Set $z(x) = \sum_{i=0}^m z_i(x)$, where $z_i(x) \in H_i[x_1, ..., x_n]$, $i = 0, 1, \ldots, m$. Then, the following relationship between the norms can be deduced:*

$$\|z(x)\|_p^p = \|z_0(x)\|_{(p,0)}^p + \|z_1(x)\|_{(p,1)}^p + \ldots + \|z_m(x)\|_{(p,m)}^p \tag{5.1}$$

*and*

$$\|z\|_\infty := \ max\{\|z_0(x)\|_{(\infty,0)}, \|z_1(x)\|_{(\infty,1)}, \ldots, \|z_m(x)\|_{(\infty,m)}\}. \tag{5.2}$$

Proof: Obvious. □

On the other hand, let $C \subset \mathbb{R}^n$ be a closed convex set and let $K^m(C)$ and $K_k(C)$ be the nonempty closed convex cones of the polynomials in $\mathbb{R}_m[x_1, ..., x_n]$ and $H_k[x_1, ..., x_n]$ respectively, which are convex on $C$.

In the following all the properties to be deduced can be applied to both normed spaces. For simplicity, we use the character $\mathbb{E}$ to denote indistinctly both normed spaces with $K(C)$ the respective above-mentioned convex cones. Moreover, we denote by $\|z\|$ any of the norms defined above where $z$ denotes a polynomial in place of $z(x)$.

## 5.3 The Minimal Norm problem

Let $(y_1, y_2), (w_1, w_2) \in K(C) \times K(C)$ be a couple of d.c. representations of $z$ on $C \subset \mathbb{R}^n$ closed convex set. We define

$$(y_1, y_2) \text{ is better than } (w_1, w_2) \text{ with respect to } \|.\| \Leftrightarrow \|y_1 + y_2\| \leq \|w_1 + w_2\|.$$

A d.c. representation of $z$ is *minimal*, if it is better that any other d.c. representation of $z$. From the practical viewpoint of how to obtain the minimal d.c. representation of $z$ we will describe *the minimal norm problem.* Consider $z \in \mathbb{E}$ and let $(y_1, y_2) \in K(C) \times K(C)$ such that $z = y_1 - y_2$, i.e., a d.c. representation of $z$ on $C$, and define $v := y_1 + y_2$, which is a convex polynomial on $C$, i.e, $v \in K(C)$. Hence, we can write

$$y_1 = \frac{z+v}{2} \text{ and } y_2 = \frac{v-z}{2}, \tag{5.3}$$

so $v = -z + 2y_1 = z + 2y_2$ and we have $v \in \{-z + 2K(C)\} \cap \{z + 2K(C)\}$. Thus, the minimal norm problem can be expressed as follows: *given $z \in \mathbb{E}$ we want to find $v \in \{-z + 2K(C)\} \cap \{z + 2K(C)\}$ with minimal norm.* The minimal norm problem is equivalent to the programming problem

$$\text{minimize } \{\|v\| : v \in \{-z + 2K(C)\} \cap \{z + 2K(C)\}\}. \tag{5.4}$$

Hence, by using the expressions in (5.3), the optimal solution $v^*$ of the problem (5.4) give us an optimal d.c. representation for $z = y_1^* - y_2^*$ where

$$y_1^* = \frac{z+v^*}{2} \text{ and } y_2^* = \frac{v^*-z}{2}.$$

Taking into account the definitions given in Luc D.T. et al. [46] the pair $(y_1^*, y_2^*)$ is called the *least deviation decomposition (LDD)* of the polynomial $z$ on $C$.

**Example 5.3.1** *(Homogeneous polynomials of degree 1) Let $z(x,y) = 2 - 3x + 4y$ be a polynomial in $\mathbb{R}_1[x,y] = H_0[x,y] \oplus H_1[x,y]$ (note that $H_0[x,y] = \mathbb{R}$). We want to obtain the LDD of $z$ on the convex compact set $C \subset \mathbb{R}^2$. In this example we have $K^1(C) = \mathbb{R}_1[x,y]$. Then, we can see that the polynomial $v^*(x,y) = 0$ solves the* *programming problem (5.4) and the pair $(y_1^*(x,y), y_2^*(x,y))$, with $y_1^*(x,y) = z(x,y)/2$ and $y_2^*(x,y) = -z(x,y)/2$, is the LDD of $z(x,y)$ on $C$. Thus, we can write*

$$z(x,y) = y_1^*(x,y) - y_2^*(x,y) = (1 - \frac{3}{2}x + 2y) - (-1 + \frac{3}{2}x - 2y).$$

*This is a general procedure so if we have the polynomial $z(x_1, x_2, \dots, x_n) \in H_1[x_1, x_2, \dots, x_n]$, i.e., $z(x_1, x_2, \dots, x_n) = \sum_{i=1}^n a_i x_i$ then we obtain the LDD of the convex polynomial $z(x_1, x_2, \dots, x_n)$ by using the following d.c. representation*

$$z(x_1, x_2, \dots, x_n) = \sum_{i=1}^n \frac{a_i}{2} x_i - \sum_{i=1}^n \frac{-a_i}{2} x_i.$$

*Notice that the d.c. representation*

$$z(x_1, x_2, \dots, x_n) = \sum_{i=1}^n a_i x_i - 0$$

*is not the LDD of the polynomial.*

**Example 5.3.2** *(Nonconvex polynomials with one variable) Let $z(x) = 5x^3$ be a polynomial in $H_3[x]$. We want to obtain the LDD of $z$ on the convex compact set $C = [-r, r] \subset I\!R$ with $r > 0$. Consider $v(x) = ax^3$ then, we can write $(v+z)(x) = (a+5)x^3$ and $(v-z)(x) = (a-5)x^3$ so that, from the convexity of the polynomials $(v+z)(x)$ and $(v-z)(x)$, we can deduce:*

$$\text{i) } (v+z)''(x) = 6(a+5)x \geq 0 \text{ and when } \begin{cases} x > 0 & \text{then} \quad a > -5 \\ x < 0 & \text{then} \quad a < -5 \\ x = 0 & \text{then} \quad a \in I\!R \end{cases}$$

$$\text{ii) } (v-z)''(x) = 6(a-5)x \geq 0 \text{ and when } \begin{cases} x > 0 & \text{then} \quad a > \;\; 5 \\ x < 0 & \text{then} \quad a < \;\; 5 \\ x = 0 & \text{then} \quad a \in I\!R \end{cases}$$

*Hence, if $x \geq 0$ then $a \geq 5$ so that the minimal norm is obtained for $a = 5$. Thus, $v(x) = 5x^3$ and the pair $(5x^3, 0)$ is a LDD of $z = 5x^3 - 0$ on [0,r]. On the other hand, if $x < 0$ then $a < -5$ and the minimal norm is obtained for $a = -5$ as above. Thus, $v(x) = -5x^3$ and the pair $(0, -5x^3)$ is a LDD of $z = 0 - (-5x^3)$ on $[-r, 0]$, and we can write the following d.c. representation*

$$5x^3 = max\{5x^3, 0\} - max\{-5x^3, 0\} \text{ on } C = [-r, r],$$

*where $max\{5x^3, 0\}$ and $max\{-5x^3, 0\}$ are convex functions on $I\!R$.*

**Example 5.3.3** *(Convex polynomials with one variable) Let $z = 3x^4$ be a polynomial in $H_4[x]$. We want to obtain the LDD of $z$ on the convex compact set $C = [-r, r] \subset I\!R$ with $r > 0$. Consider $v(x) = ax^4$. As in the previous example*

*i) from* $(v+z)''(x) = 12(a+3)x^2 \geq 0$, *we deduce that if* $\begin{cases} x \neq 0 & then \quad a > -3 \\ x = 0 & then \quad a \in I\!R \end{cases}$

*ii) from* $(v-z)''(x) = 12(a-3)x^2 \geq 0$, *we deduce that if* $\begin{cases} x \neq 0 & then \quad a > 3 \\ x = 0 & then \quad a \in I\!R \end{cases}$

*Hence, for all $x \in [-r, r]$ we have that $a \geq 3$ so that the minimal norm is obtained for $a = 3$. Thus, $v(x) = 3x^4$ and the pair $(3x^4, 0)$ is a LDD of $z = 3x^4 - 0$ on [-r,r]. Then, we can write*

$$3x^4 = 3x^4 - 0 \text{ on } C = [-r, r].$$

When the $LDD$ of a polynomial $z$ is difficult to obtain then, the relationship between $I\!R_m[x_1, ..., x_n]$ and $H_k[x_1, ..., x_n]$, $k = 0, 1, \ldots, m$,

$$I\!R_m[x_1, \ldots, x_n] = \bigoplus_{k=0}^{m} H_k[x_1, \ldots, x_n], \tag{5.5}$$

allows us to obtain an alternative d.c. representation of the polynomial $z$ by using the $LDD$ of its homogeneous summands. This new d.c. representation of $z$, generally, will not be the optimal but it will improve the initial d.c. representation.

**Proposition 5.3.4** *(The decomposition property) Let $C \subset I\!R^n$ be a closed convex set, let $z = \sum_{k=0}^m z_k$ be a polynomial with $z_k \in H_k[x_1, ..., x_n]$ for each $k = 0, 1, \ldots, m$, and consider $(y^o_{1,k},\ y^o_{2,k})$ the LDD of $z_k$ on $C$ for each $k = 0, 1, \ldots, m$. Then, the pair $(y^o_1,\ y^o_2)$ where $y^o_1 = \sum_{k=0}^m y^o_{1,k}$ and $y^o_2 = \sum_{k=0}^m y^o_{2,k}$ is a d.c. representation of $z$ on $C$ with*

$$\| v^o \|_p \quad \geq \quad \| v^* \|_p \ \textit{ for all } p = 0, 1, 2, \ldots \textit{ and } p = \infty. \tag{5.6}$$

*Moreover, if 5t ha–ens that $y^*_1 = \sum_{k=0}^m y^*_{1,k}$ and $y^*_2 = \sum_{k=0}^m y^*_{2,k}$ with $y^*_{1,k}$ and $y^*_{2,k}$ are polynomials in $K_k(C)$ (which is not always true), for all $k = 0, 1, \ldots, m$, then*

$$\| v^o \|_p \quad = \quad \| v^* \|_p \ \textit{ for all } p = 0, 1, 2, \ldots \textit{ and } p = \infty. \tag{5.7}$$

Proof: Let $(y^*_1, y^*_2)$ be a $LDD$ of $z$ on $C$ so the polynomial $v^* = y^*_1 + y^*_2$ solve the minimal norm program. Hence, we can write

$$\| v^o \|_p \quad \geq \quad \| v^* \|_p \ \text{ for all } p = 0, 1, 2, \ldots \text{ and } p = \infty. \tag{5.8}$$

where $v^o = y_1^o + y_2^o$. On the other hand, if we can write $y_1^* = \sum_{k=0}^{m} y_{1,k}^*$ and $y_2^* = \sum_{k=0}^{m} y_{2,k}^*$ where $y_{1,k}^*$ and $y_{2,k}^*$ are polynomials in $K_k(C)$, for all $k = 0, 1, \ldots, m$ thus, $v_k^* = y_{1,k}^* + y_{2,k}^*$ satisfies $\| v_k^* \|_{(k,p)}^p \geq \| v_k^o \|_{(k,p)}^p$ and $\| v_k^* \|_\infty \geq \| v_k^o \|_\infty$, for all $k = 0, 1, \ldots, m$ because the pair $(y_{1,k}^o,\ y_{2,k}^o)$ is the $LDD$ of $z_k$ on $C$, for all $k = 0, 1, \ldots, m$. Hence, we can write

$$\| v^* \|_p^p = \sum_{k=0}^{m} \| v_k^* \|_{(k,p)}^p \geq \sum_{k=0}^{m} \| v_k^o \|_{(k,p)}^p = \| v^o \|_p^p \tag{5.9}$$

and

$$\| v^* \|_\infty = \max\{\| v_k^* \|_{(k,\infty)}\}_{k=0,1,\ldots,m} \geq \max\{\| v_k^o \|_{(k,\infty)}\}_{k=0,1,\ldots,m} = \| v^o \|_\infty, \tag{5.10}$$

which proves the proposition. □

The minimal norm problem can be transformed into an equivalent semi-definite program or into an equivalent semi-infinite quadratic programming problem with linear constraints.

## 5.4 Relationships between semidefinite and semi-infinite programming

Let $M$ be a matrix $M \in I\!R^{n*n}$. The inequality $M \succeq 0$ means that the matrix $M$ is positive semidefinite. The linear matrix inequality of the form

$$M(x) := M_0 + x_1 M_1 + \ldots + x_m M_m \succeq 0 \tag{5.11}$$

is a convex inequality in the variable $x = (x_1, \ldots, x_m) \in I\!R^m$, where the matrices $M_i \in I\!R^{n*n}$ satisfy $M_i^t = M_i$. We can immediately see that (5.11) is equivalent to an infinite set of linear inequalities

$$\lambda^t M(x)\lambda := \lambda^t M_0 \lambda + x_1(\lambda^t M_1 \lambda) + \ldots + x_m(\lambda^t M_m \lambda) \geq 0 \tag{5.12}$$

for all $\lambda$ in the compact set $S^n := \{\lambda \in I\!R^n : \ \| \lambda \| = 1\}$. A convex quadratic inequality in the form $(Qx+b)^t(Qx+b) + c^t x + d \leq 0$, where $x \in I\!R^n$, can also be written in matrix form as

$$\begin{bmatrix} I & Qx+b \\ (Qx+b)^t & -(c^t x + d) \end{bmatrix} \succeq 0. \tag{5.13}$$

This can be expressed as an affine combination of symmetric matrices

$$\begin{bmatrix} I & Qx+b \\ (Qx+b)^t & -(c^tx+d) \end{bmatrix} := M_0 + x_1M_1 + \ldots + x_nM_n \succeq 0, \tag{5.14}$$

where

$$M_0 = \begin{bmatrix} I & b \\ b^t & -d \end{bmatrix}, \; M_i = \begin{bmatrix} 0 & q_i \\ q_i^t & -c_i \end{bmatrix}, \text{ for all } i = 1, \ldots, n$$

with $Q = [q_1, \ldots, q_n]$.

In semidefinite programming ($SDP$) a linear function is minimized subject to a constraint where an affine combination of symmetric matrices is positive semidefinite:

$$\begin{array}{rc} \text{minimize} & c^tx \\ \text{subject to:} & M(x) := M_0 + x_1M_1 + \ldots + x_mM_m \succeq 0, \\ & M_i^t = M_i, i = 0, 1, \ldots, m. \end{array} \tag{5.15}$$

The dual problem of (5.15) is

$$\begin{array}{rc} \text{maximize} & -\mathbf{Tr}M_0Z \\ \text{subject to:} & \mathbf{Tr}M_iZ = c_i, i = 1, \ldots, m, \\ & Z \succeq 0. \end{array} \tag{5.16}$$

Applying duality results we know that the optimal values of (5.15) and (5.16) are equal if at least one of the problems is strictly feasible.

A general convex quadratically constrained quadratic program (QCQP)

$$\begin{array}{rc} \text{minimize} & (Q_0x+b)^t(Q_0x+b) + c_0^tx + d_0 \\ \text{subject to:} & (Q_ix+b_i)^t(Q_ix+b_i) + c_i^tx + d_i \le 0, i = 1, \ldots, r \end{array} \tag{5.17}$$

can be written, by adding a new variable $t$, in the form

$$\begin{array}{rl} \text{minimize} & t \\ \text{subject to:} & \begin{bmatrix} I & Q_0x+b \\ (Q_0x+b)^t & -(c_0^tx+d_0-t) \end{bmatrix} \succeq 0, \\ & \begin{bmatrix} I & Q_ix+b_i \\ (Q_ix+b_i)^t & -(c_i^tx+d_i) \end{bmatrix} \succeq 0, i = 1, \ldots, r, \end{array} \tag{5.18}$$

which is a semidefinite program with variables $x \in \mathbb{R}^m$ and $t \in \mathbb{R}$. Moreover, the linear program (LP)

$$\begin{array}{rc} \text{minimize} & c^tx \\ \text{subject to:} & Ax+b \ge 0 \end{array} \tag{5.19}$$

can be expressed as a semidefinite program by defining $M(x) :=$ diag $(Ax + b)$ i.e. $M_0 =$ diag $(b)$ and $M_i =$ diag $(a_i)$, $i = 1, \dots, m$, where $A = [a_i, \dots, a_m] \in I\!R^{n*m}$.

Semidefinite programming unifies, among others, the standard problems of linear and quadratic programming and can be regarded as an extension of linear programming where the componentwise inequalities between vectors are replaced by matrix inequalities or, equivalently, the first orthant is replaced by the cone of semidefinite matrices. Consequently, most interior-point methods for linear programming have been generalized to semidefinite programs. As in linear programming, these methods have polynomial worst-case complexity. While it is interesting to note that QCQPs can be represented as semidefinite programming problems, it may not be a good idea algorithmically. The semidefinite programming formulation will be less efficient, especially when the matrices $Q_i$ have high rank. A more efficient interior-point method for $QCQP$'s can be developed by using Nesterov and Nemirovsky's formulation as a problem over the second-order cone (see Nesterov et al.[49]).

By using the expression (5.12) equivalent to (5.11) we can express the $SDP$ (5.15) as the following semi-infinite linear program

$$\begin{array}{ll} \text{minimize} & c^t x \\ \text{subject to:} & \lambda^t M(x) \lambda \geq 0 \text{ for all } \lambda \in S^n. \end{array} \tag{5.20}$$

## 5.5 The equivalent semi-infinite and semidefinite programs for solving the Minimal Norm problem

A peculiarity of the minimal norm problem (5.4) is that, in the case of the Euclidean norm, it can be transformed into an equivalent semi-infinite quadratic programming problem with linear constraints. The feasible set of the problem (5.4) can be described by the expression $(v \pm z)(x) \in K(C)$ i.e. $(v \pm z)(x)$ are convex polynomials, which means that the Hessian matrices $\nabla^2 (v \pm z)(x) = \sum_{i \in I} (v_i \pm a_i) \nabla^2 f_i(x)$ are positive semidefinite. Hence, we can write

$$\lambda^t \nabla^2 (v \pm z)(x) \lambda \geq 0 \quad \forall \lambda \in S^n, \ \forall x \in C \subset I\!R^n, \tag{5.21}$$

where $S^n = \{x \in I\!R^n : \ \|x\| = 1\}$, or equivalently

$$\lambda^t \sum_{i \in I} (v_i \pm a_i) \nabla^2 f_i(x) \lambda \geq 0 \quad \forall \lambda \in S^n \quad , \forall x \in C \subset I\!R^n. \tag{5.22}$$

By substituting the constraint set of (5.4) by the equivalent constraint set (5.22), the problem (5.4) can be transformed into the equivalent semi-infinite quadratic programming problem

$$\begin{cases} \text{minimize} & \|v\|^2 = \sum_{i\in I} v_i^2 \\ \text{subject to:} & \lambda^t \sum_{i\in I}(v_i \pm a_i)\nabla^2 f_i(x)\lambda \geq 0, \\ & \forall \lambda \in S^n \quad , \forall x \in C \subset I\!R^n, \end{cases} \tag{5.23}$$

which depends on a family of parameters $x$ and $\lambda$, or into the equivalent semidefinite quadratic programming problem

$$\begin{cases} \text{minimize} & \|v\|^2 = \sum_{i\in I} v_i^2 \\ \text{subject to:} & \sum_{i\in I}(v_i \pm a_i)\nabla^2 f_i(x) \succeq 0, \\ & \forall x \in C \subset I\!R^n, \end{cases} \tag{5.24}$$

which depends solely on the parameter $x$. The set $C$ will usually be a convex compact set in the form $C = \prod_{i=1}^n [r_i, t_i]$.

## 5.6 Example to compare the semi-infinite and semidefinite procedures

In this section we present an example to compare the semi-infinite and semidefinite procedures in order to obtain the $LDD$ of a polynomial. Let $z(x,y) = xy + 3x^2y$ be a polynomial in $H_2\,[x,y] \oplus H_3\,[x,y] \subset I\!R_3\,[x,y]$. In this example, in order to compare the results, we will obtain the $LDD$ of the polynomial $z(x,y)$ on the convex compact set $C = [5,20] \times [5,20] \subset I\!R^2$ and the $LDD$ of the polynomials $xy$ and $3x^2y$ in two different ways. Firstly, by using the semi-infinite algorithm described in Chapter 6, which solves the semi-infinite problem (5.23). Secondly, by using the semidefinite algorithm obtained from [64], which solves the semidefinite problem (5.24).

**The constraint set of this example:** In order to determine the constraint set of this example consider the set of homogeneous polynomials of degree 2

$$B_2 := \{f_1(x,y) := x^2,\ f_2(x,y) := xy,\ f_3(x,y) := y^2\},$$

and the set of homogeneous polynomials of degree 3

$$B_3 := \{f_4(x,y) := x^3,\ f_5(x,y) := x^2y,\ f_6(x,y) := xy^2,\ f_7(x,y) := y^3\},$$

which are the monomials of the usual bases in $H_2[x,y]$ and $H_3[x,y]$ respectively. Moreover, the set $B = B_2 \cup B_3$ is a new base for $H_2[x,y] \oplus H_3[x,y]$. Hence, the coordinates of the polynomial $z(x,y)$ can be written $z(x,y) \equiv (0,\ 1,\ 0,\ 0,\ 3,\ 0,\ 0)$. For the sake of simplicity, we denote the coordinates of a given polynomial $v(x,y) \in H_3[x,y] \oplus H_2[x,y]$ by using the first seven letters of the alphabet in the expression $(a,\ b,\ c,\ d,\ e,\ f,\ g)$. As we know how to calculate the Hessian of each monomial function $f_i(x,y)$, $i = 1,2,\ldots,7$, e.g. the Hessian of the monomial $f_5(x,y) = x^2y$ is

$$\nabla^2 f_5(x,y) = \begin{bmatrix} 2y & 2x \\ 2x & 0 \end{bmatrix},$$

then by using the expression $\nabla^2(v \pm z)(x,y) = \sum_{i=1}^{7}(v_i \pm z_i)\nabla^2 f_i(x,y)$, we can obtain the Hessians $\nabla^2(v \pm z)(x,y)$ of the convex functions $(v \pm z)(x,y)$ in the form

$$\begin{bmatrix} 6dx + 2(e \pm 3)y + 2a & 2(e \pm 3)x + 2fy + (b \pm 1) \\ 2(e \pm 3)x + 2fy + (b \pm 1) & 2fx + 6gy + 2c \end{bmatrix}, \tag{5.25}$$

which are positive-semidefinite matrices where the values $a,\ b,\ c,\ d,\ e,\ f$ and $g$ are the variables, and $x, y$ are the parameters which satisfy the property $(x,y) \in [0,25] \times [0,25]$. On the other hand, a positive-semidefinite matrix must satisfy

$$[\lambda_1, \lambda_2] \begin{bmatrix} 6dx + 2(e \pm 3)y + 2a & 2(e \pm 3)x + 2fy + (b \pm 1) \\ 2(e \pm 3)x + 2fy + (b \pm 1) & 2fx + 6gy + 2c \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq 0,$$

with $\lambda_1^2 + \lambda_2^2 = 1$, which allows the linear constraints of the semi-infinite program (5.23) to be written as follows:

$$\begin{aligned} &(2\lambda_1^2)a + (2\lambda_1\lambda_2)b + (2\lambda_2^2)c + (6x\lambda_1^2)d + 2(y\lambda_1^2 + 2x\lambda_1\lambda_2)e + \\ &+2(x\lambda_2^2 + 2y\lambda_1\lambda_2)f + (6y\lambda_2^2)g \pm (6y\lambda_1^2 + 12x\lambda_1\lambda_2 + 2\lambda_1\lambda_2) \geq 0, \end{aligned} \tag{5.26}$$

where the values $a,\ b,\ c,\ d,\ e,\ f$ and $g$ are the variables, and $x, y, \lambda_1$ and $\lambda_2$ are the parameters which satisfy the properties $(x,y) \in [0,25] \times [0,25]$ and $\lambda_1^2 + \lambda_2^2 = 1$. Dividing the inequality (5.26) by 2 and substituting the term $\pm(6y\lambda_1^2 + 12x\lambda_1\lambda_2 + 2\lambda_1\lambda_2)$ by its absolute value we can write the new equivalent constraint set

$$\begin{aligned} &\lambda_1^2 a + \lambda_1\lambda_2 b + \lambda_2^2 c + 3x\lambda_1^2 d + (y\lambda_1^2 + 2x\lambda_1\lambda_2)e + \\ &+(x\lambda_2^2 + 2y\lambda_1\lambda_2)f + 3y\lambda_2^2 g - \mid 3y\lambda_1^2 + 6x\lambda_1\lambda_2 + \lambda_1\lambda_2 \mid \geq 0. \end{aligned} \tag{5.27}$$

The parameters $\lambda_1$ and $\lambda_2$ are linked via the relation $\lambda_1^2 + \lambda_2^2 = 1$, so for fixed $x$ and $y$ the pairs $(\lambda_1, \lambda_2)$ and $(-\lambda_1, -\lambda_2)$ define the same constraint. To avoid this repetition we can consider the new parameter $\omega \in [0,\pi]$ in such a way that $\lambda_1 = \cos\omega$ and $\lambda_2 = \sin\omega$.

**Remark 2** *In general, the relation $\lambda_1^2 + \lambda_2^2 + \ldots + \lambda_n^2 = 1$ between the parameters $\lambda_1, \lambda_2, \ldots, \lambda_n$ can be described by using spherical coordinates. Indeed, consider the usual base $\{e_1, e_2, \ldots, e_n\}$ of the vectors of $I\!R^n$. Let $\vec{OP}$ be the vector position of the point $(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and define $\omega_{n-1} = ang(\vec{OP}, e_n)$ (angle between the vectors $\vec{OP}$ and $e_n$), $\omega_{n-2} = ang(\vec{OP}_{n-1}, e_{n-1})$ where $\vec{OP}_{n-1}$ is the projection of the vector $\vec{OP}$ on $0+ < e_1, e_2, \ldots, e_{n-1} >$ parallel to the direction $< e_n >$, ..., $\omega_i = ang(\vec{OP}_{i+1}, e_{i+1})$ where $\vec{OP}_{i+1}$ is the projection of the vector $\vec{OP}$ on $0+ < e_1, e_2, \ldots, e_{i+1} >$ parallel to the direction $< e_{i+2}, \ldots, e_n >$, ..., $\omega_1 = ang(\vec{OP}_2, e_2)$ where $\vec{OP}_2$ is the projection of the vector $\vec{OP}$ on $0+ < e_1, e_2 >$ parallel to the direction $< e_3, \ldots, e_n >$. Then we can write*

$$\left\{\begin{array}{lcr} \lambda_1 & = & \sin\omega_1 \sin\omega_2 \ldots \sin\omega_{n-2} \sin\omega_{n-1}, \\ \lambda_2 & = & \cos\omega_1 \sin\omega_2 \ldots \sin\omega_{n-2} \sin\omega_{n-1}, \\ \ldots & \ldots & \ldots\ldots\ldots\ldots \quad , \\ \lambda_{n-2} & = & \cos\omega_{n-3} \sin\omega_{n-2} \sin\omega_{n-1}, \\ \lambda_{n-1} & = & \cos\omega_{n-2} \sin\omega_{n-1}, \\ \lambda_n & = & \cos\omega_{n-1}, \end{array}\right. \tag{5.28}$$

*where $\omega_1 \in [0, 2\pi[$ and $\omega_i \in [0, \pi], i = 2,\ n-1$.*

**An initial strictly interior feasible point for this example:** The algorithm of semi-infinite programming which solves the program (5.23) needs an initial feasible point strictly interior to the constraint set (5.26). In the following we explain how to obtain it. We know that the set of homogeneous polynomials of degree 2

$$U_2 := \{g_1 := x^2,\ g_2 := (x+y)^2,\ g_3 := y^2\},$$

and the set of homogeneous polynomials of degree 3

$$U_3 := \{g_4 := x^3,\ g_5 := (x+y)^3,\ g_6 := (2x+y)^3,\ g_7 := y^3\}$$

are new bases for $H_2[x, y]$ and $H_3[x, y]$ respectively. Moreover, the set $U = U_2 \cup U_3$ is a new base for $H_2[x, y] \oplus H_3[x, y]$. The change from the coordinates $X_U$ of a polynomial in the base $U$ to the coordinates $X_B$ in the base $B$ can be expressed by using the matrix

$$C := \left[\begin{array}{ccc|cccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 8 & 0 \\ 0 & 0 & 0 & 0 & 3 & 12 & 0 \\ 0 & 0 & 0 & 0 & 3 & 6 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right],$$

in the form $X_B = CX_U$. The columns of the matrix $C$ are the components of the polynomials in the new base $U$ expressed in the usual base $B$. Therefore, the change from the coordinates $X_B$ in the base $B$ to the coordinates $X_U$ in the base $U$ can be obtained by the inverse transformation $X_U = C^{-1}X_B$ i.e.

$$\begin{bmatrix} -0.5 \\ 0.5 \\ -0.5 \\ -3.0 \\ -1.0 \\ 0.5 \\ 0.5 \end{bmatrix}_U = \left[\begin{array}{ccc|cccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 8 & 0 \\ 0 & 0 & 0 & 0 & 3 & 12 & 0 \\ 0 & 0 & 0 & 0 & 3 & 6 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right]^{-1} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 3 \\ 0 \\ 0 \end{bmatrix}_B .$$

Also, we can see that the matrices

$$C_1 := \begin{bmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \end{bmatrix} \text{ and } C_2 := \begin{bmatrix} 1 & 1 & 8 & 0 \\ 0 & 3 & 12 & 0 \\ 0 & 3 & 6 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

give us the change from the coordinates $X_{U_2}$ and $X_{U_3}$ in the bases $U_2$ and $U_3$ to the coordinates $X_{B_2}$ and $X_{B_3}$ in the usual bases $B_2$ and $B_3$ of the homogeneous components $xy$ and $3x^2y$ respectively. As before, we can write $X_{B_i} = C_iX_{U_i}$ for $i = 2, 3$ i.e.

$$\begin{bmatrix} -0.5 \\ 0.5 \\ -0.5 \end{bmatrix}_{U_2} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}_{B_2}$$

and

$$\begin{bmatrix} -3.0 \\ -1.0 \\ 0.5 \\ 0.5 \end{bmatrix}_{U_3} = \begin{bmatrix} 1 & 1 & 8 & 0 \\ 0 & 3 & 12 & 0 \\ 0 & 3 & 6 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 3 \\ 0 \\ 0 \end{bmatrix}_{B_3} .$$

Thus, we can obtain a d.c. representation of both terms of the polynomial $z(x, y) = xy + 3x^2y$ in the forms

$$xy = 0.5(x + y)^2 - 0.5(x^2 + y^2)$$

and

$$3x^2y = 0.5(2x + y)^3 + 0.5y^3 - (3x^3 + (x + y)^3).$$

Hence, by using the convex polynomials

$$y_1(x, y) = 0.5(x + y)^2 + 0.5(2x + y)^3 + 0.5y^3$$

and

$$y_2(x,y) = 0.5(x^2+y^2) + 3x^3 + (x+y)^3,$$

which can be expanded to obtain the expressions

$$y_1(x,y) = 0.5x^2 + xy + 0.5y^2 + 4x^3 + 6x^2y + 3xy^2 + y^3$$

and

$$y_2(x,y) = 0.5x^2 + 0.5y^2 + 4x^3 + 3x^2y + 3xy^2 + y^3,$$

we can write

$$z(x,y) = y_1(x,y) - y_2(x,y).$$

Thus, the polynomial $y_1(x,y)+y_2(x,y)$, which has the coordinates $(1,1,1,8,9,6,2)$, is a feasible point of the constraint set (5.26).

To make sure that we have an strictly interior feasible point to the constraint set we can add the convex polynomial $q(x,y) = x^2+y^2$ to both terms of the current d.c. representation of $z(x,y)$ so a suitable new d.c. representation of the polynomial can be obtained by the expression

$$z(x,y) = (y_1(x,y) + q(x,y)) - (y_2(x,y) + q(x,y)).$$

Then, the polynomial $v(x,y) := y_1(x,y) + y_2(x,y) + 2q(x,y)$ has the coordinates $(3,1,3,8,9,2,6,2)$, which represent a feasible point strictly interior to the constraint set (5.26).

**The optimal solutions and the objective values for this example:** By using a SIP algorithm we have obtained the optimal solution

$$v^* = .0932x^2 - .0110xy + .0611y^2 + 2.6291x^3 + 1.0286x^2y + .6860xy^2 + 1.7550y^3 \quad (5.29)$$

with the objective value $\|v^*\|^2 = 11.534$ for the polynomial $xy + 3x^2y$. Moreover, by using the same procedure as mentioned above, we can obtain the optimal solutions of the components $xy$ and $3x^2y$, respectively

$$v_1^*(x,y) = 0.5492x^2 + 0.0xy + 0.4551y^2 \quad (5.30)$$

with the objective value $\|v_1^*\|^2 = 0.509$ and

$$v_2^*(x,y) = 2.6247x^3 + 1.0377x^2y + 0.6745xy^2 + 1.7551y^3 \quad (5.31)$$

with the objective value $\|v_2^*\|^2 = 11.501$.

We have also used a *SDP* algorithm, obtained from Wu, P. et al.[64], to solve the same problems. Then, the optimal solutions are

$$v^* = 0.1684x^2 - 1.1498xy - 1.2717y^2 + 3.0977x^3 + 1.0297x^2y + 1.4457xy^2 + 0.6469y^3, \quad (5.32)$$

with the objective value $\|v^*\|^2 = 14.094$,

$$v_1^*(x,y) = 0.5x^2 + 0.0xy + 0.5y^2, \quad (5.33)$$

with the objective value $\|v_1^*\|^2 = 0.5$, and

$$v_2^*(x,y) = 2.7835x^3 + 0.9420x^2y + 0.8661xy^2 + 1.3150y^3, \quad (5.34)$$

with the objective value $\|v_2^*\|^2 = 11.115$.

**Remark 3** *We can see that for the polynomials $xy$ and $3x^2y$ the solutions obtained by using SDP are better than the solutions obtained by using SIP (although still very similar). Moreover, it must be noted that for the polynomials $xy$, $3x^2y$ and $xy + 3x^2y$ the optimal solutions obtained by using SIP satisfy the decomposition property (see Proposition 5.3.4)*

$$11.534 = \|v^*\|^2 \leq \|v_1^*\|^2 + \|v_2^*\|^2 = 12.01$$

*but, for the optimal solutions obtained by using SDP this is not true*

$$14.094 = \|v^*\|^2 \geq \|v_1^*\|^2 + \|v_2^*\|^2 = 11.615.$$

*For this reason, as described in Chapter 6, we have chosen the semi-infinite formulation to obtain the LDD of a polynomial function, so an interior point algorithm, for solving semi-infinite quadratic programming problems with linear constraints, has been implemented by following the semi-infinite linear programming procedures described in Kaliski J. et al. [37] and in Zhi-Quan et al. [78].*

**The *LDD* for the functions in this example:** By using the best optimal solutions found, and the expressions(5.3) we can obtain the *LDD* of the polynomials $xy + 3x^2y$, $3x^2y$ and $xy$ on the convex compact set $C = [5,20] \times [5,20] \subset I\!R^2$ as follows.

First, we use the solution (5.29) to obtain the *LDD* of the polynomial $xy + 3x^2y$

$$\begin{aligned} y_1^* &= .0466x^2 + .4945xy + .03055y^2 + 1.31455x^3 + 2.0143x^2y + .3430xy^2 + .8775y^3, \\ y_2^* &= .0466x^2 - .5055xy + .03055y^2 + 1.31455x^3 - .9857x^2y + .3430xy^2 + .8775y^3. \end{aligned} \quad (5.35)$$

and we can write

$$xy + 3x^2y = y_1^*(x, y) - y_2^*(x, y). \tag{5.36}$$

Then, by using the best solution (5.34) we obtain the *LDD* of $3x^2y$

$$\begin{aligned} y_1^*(x, y) &= 1.31235x^3 + 2.0188x^2y + .33725xy^2 + .87755y^3, \\ y_2^*(x, y) &= 1.31235x^3 - .9811x^2y + .33725xy^2 + .87755y^3. \end{aligned} \tag{5.37}$$

and

$$3x^2y = y_1^*(x, y) - y_2^*(x, y). \tag{5.38}$$

Finally, by using the best solution (5.33) we obtain the *LDD* of $xy$

$$\begin{aligned} y_1^*(x, y) &= .25x^2 + .5xy + .25y^2 = \tfrac{1}{4}(x + y)^2, \\ y_2^*(x, y) &= .25x^2 - .5xy + .25y^2 = \tfrac{1}{4}(x - y)^2, \end{aligned} \tag{5.39}$$

and

$$xy = \frac{1}{4}(x + y)^2 - \frac{1}{4}(x - y)^2. \tag{5.40}$$

**Remark 4** *This last result (5.40) is interesting because, in this example, we can express the quadratic function $xy$ in the form*

$$xy = [x, y] \begin{bmatrix} 0 & .5 \\ .5 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \tag{5.41}$$

*where the symmetric matrix in (5.41) has the eigenvalues $-.5$ and $.5$ and the matrix of its eigenvectors is*

$$C := \begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 \\ -\sqrt{2}/2 & -\sqrt{2}/2 \end{bmatrix}. \tag{5.42}$$

*We have the following relationship between the above-mentioned matrices*

$$\begin{bmatrix} 0 & .5 \\ .5 & 0 \end{bmatrix} = C^t \begin{bmatrix} -.5 & 0 \\ 0 & .5 \end{bmatrix} C. \tag{5.43}$$

*Substituting the expression (5.43) in (5.41) we obtain*

$$\begin{aligned} xy &= [x, y] \begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 \\ -\sqrt{2}/2 & -\sqrt{2}/2 \end{bmatrix} \begin{bmatrix} -.5 & 0 \\ 0 & .5 \end{bmatrix} \begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 \\ -\sqrt{2}/2 & -\sqrt{2}/2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \\ &= [\tfrac{\sqrt{2}}{2}x - \tfrac{\sqrt{2}}{2}y, -\tfrac{\sqrt{2}}{2}x - \tfrac{\sqrt{2}}{2}y] \begin{bmatrix} -.5 & 0 \\ 0 & .5 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{2}}{2}x - \frac{\sqrt{2}}{2}y \\ -\frac{\sqrt{2}}{2}x - \frac{\sqrt{2}}{2}y \end{bmatrix} = \\ &= -.5\left(\tfrac{\sqrt{2}}{2}x - \tfrac{\sqrt{2}}{2}y\right)^2 + .5\left(\tfrac{\sqrt{2}}{2}x + \tfrac{\sqrt{2}}{2}y\right)^2. \end{aligned} \tag{5.44}$$

*Hence, we can see that $xy$ is equal to the linear combination of the homogenous polynomials of degree 2 obtained from the linear equations which serve to obtain the new eigenvector bases of the symmetric matrix in (5.41) where the coefficients are the eigenvalues of this matrix. By expanding the expression (5.44) we obtain*

$$xy \;=\; \tfrac{1}{4}\,(x+y)^2 - \tfrac{1}{4}\,(x-y)^2\,,$$

*which is the LDD of the polynomial $xy$ obtained in a different way.*

*Had this procedure always given similar results we would have been able to use it to obtain the LDD of a quadratic function analytically. Also, we would have been able to use it to verify whether a quadratic function is convex. The example 5.6.1 shows that this analytical procedure is not useful in some cases.*

**Example 5.6.1** *Let $z(x,y) := (x-y)^2 = x^2 - 2xy + y^2$ be a convex polynomial in $H_2\,[x,y]$. Since $z(x,y)$ is convex, we can consider $z(x,y) = y_1(x,y) - y_2(x,y)$ as the initial d.c. representation of the polynomial with $y_1(x,y) = x^2 - 2xy + y^2$ and $y_2(x,y) = 0$. Then, the polynomial $v(x,y) = y_1(x,y) + y_2(x,y)$ satisfies $\|v(x,y)\|^2 = 6.0$. Note that this initial d.c. representation of the polynomial $z(x,y)$ is not the LDD in this example. Indeed, by using the algorithm of semi-infinite programming described in Chapter 6 to solve the program (5.23), we came to the optimal solution $v^*(x,y) = 1.4354x^2 - 1.2596xy + 1.3148y^2$ with the objective value $\|v^*\|^2 = 5.375$. Hence, by using the expressions (5.3) we obtain the polynomials*

$$\begin{aligned} y_1^*(x,y) &= 1.2177x^2 - 1.6298xy + 1.1574y^2,\\ y_2^*(x,y) &= .2177x^2 + .3702xy + .1574y^2. \end{aligned}$$

*so the LDD of $x^2 - 2xy + y^2$ can be written*

$$x^2 - 2xy + y^2 = y_1^*(x,y) - y_2^*(x,y).$$

*It should be observed that if we calculate the determinant of the Hessians of both functions*

$$\nabla^2(y_1^*) = \begin{bmatrix} 2.4354 & -1.6298 \\ -1.6298 & 2.3148 \end{bmatrix} \text{ and } \nabla^2(y_2^*) = \begin{bmatrix} .4354 & .3702 \\ .3702 & .3148 \end{bmatrix},$$

*we obtain Det $(\nabla^2 y_1^*) = 2.981$ and Det $(\nabla^2 y_2^*) = .358 * 10^{-7}$, which is additional proof that $y_1^*(x,y)$ and $y_2^*(x,y)$ are convex polynomials.*

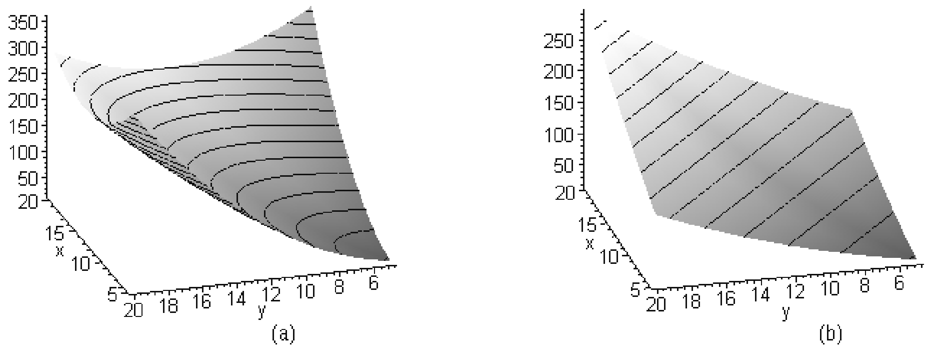

Figure 5.1: Graphics (a) and (b) are respectively the graphics of the convex polynomials $y_1^*(x, y)$ and $y_2^*(x, y)$, which define the $LDD$ of the polynomial $(x - y)^2$.

In Figure (5.1) we can see the graphics of the convex functions $y_1^*(x, y)$ and $y_2^*(x, y)$.

# Chapter 6

# Solving semi-infinite quadratic programs. Numerical results

## 6.1 Introduction

The problem under consideration is a semi-infinite quadratic programming problem with infinitely many linear constraints $(SIQP)$. Let $\mathcal{T} \subset I\!R^s$ be an infinite compact set of parameters, and let $t \to a_t$ and $t \to b_t$ be continuous functions on $\mathcal{T}$ with $a_t \in I\!R^n$ and $b_t \in I\!R$. By defining

$$\mathcal{G} := \{x \in I\!R^n : a_t x + s_t = b_t, \| a_t \| = 1, s_t \geq 0, t \in \mathcal{T}\}, \tag{6.1}$$

the semi-infinite quadratic programming problem can be expressed by

$$\text{minimize } \left\{ \frac{1}{2} x^t Q x + c^t x \ : \ x \in \mathcal{G} \right\}. \tag{6.2}$$

The standard methods of optimization solve problems with a finite number of variables and constraints. Nevertheless, semi-infinite quadratic programming problems have infinite constraints. In order to overcome this difficulty we will use Reemtsen's straightforward generalization (see [55]) of a theorem evolved by Gustafson in [23] on the computational equivalence of a semi-infinite linear program and a standard linear program, as follows.

**Theorem 6.1.1** *Given a nonlinear semi-infinite programming problem*

$$\textit{maximize } \{c^t x : \ f_t(x) \leq 0, \ t \in \mathcal{T}\}, \tag{6.3}$$

*where $\mathcal{T}$ is a compact set of parameters, $f_t(x)$ are continuous on $\mathcal{T}$ and the feasible set $\mathcal{F} = \{x \in I\!R^n | \; f_t(x) \leq 0, \; t \in \mathcal{T}\}$ is convex and compact. Then, there exists a finite set $\mathcal{N} \subset \mathcal{T}$ such that the semi-infinite program (6.3) and the convex program*

$$\text{maximize } \{c^t x : \; f_t(x) \leq 0, \; t \in \mathcal{N}\} \tag{6.4}$$

*are* **computationally equivalent** *(which means that we cannot find any difference between the feasible sets of both problems within a given computational precision).*

Notice that this result does not imply that the optimal solutions to both programs are close to each other, but it does mean that the optimal values are identical within the given computational precision. To obtain the computational equivalence the number of constraints indexed by $\mathcal{N}$ must be extremely large and this is still a drawback.

In this chapter we propose a build-up and down strategy, introduced by Den Hertog in [11], for standard linear programs that use the logarithmic barrier method. The algorithm which will be described is an adaptation of the linear semi-infinite algorithm developed by Kaliski, J. et al. in [37] to the case of a convex quadratic objective function. To enable us to understand the build-up and down strategy, we analyze in Section 6.7 the effect of adding a constraint and in Section 6.8 the effect of deleting a constraint on the distance to the central path respectively. We also prove the finite convergence (finite number of steps) of the algorithm, which depends on the desired accuracy, the radius of the largest Euclidean ball contained in the feasible set and the problem dimension.

## 6.2 Build-up and build-down strategies

Let $\mathcal{T}$ be an $s$ dimensional convex compact set of parameters in the form

$$\mathcal{T} = \prod_{i=1}^{s} [u_i, v_i],$$

with $u_i, v_i \in I\!R$ and $[u_i, v_i] := \{t_i \in I\!R : u_i \leq t_i \leq v_i\}$, $i = 1, \ldots, s$. The key in determining the accuracy and speed of the algorithm is the discretization strategy applied to the set of parameters $\mathcal{T}$.

An initial strategy, called the **static mesh technique**, can be defined by taking a sufficiently large grid of $k_i$ points for each $i^{\text{th}}$ dimension of the parameter space

$$I_i := \{u_i + (j-1)(v_i - u_i)/k_i, j = 1, \ldots, k_i\}.$$

Thus, a grid of mesh points $\mathcal{N} := \prod_{i=1}^{s} I_i \subset \mathcal{T}$ can be obtained, and the discretized problem defined by this finite grid of mesh points will be solved. While theoretically sufficient the static mesh technique has significant numerical impediments in practice, because when the dimension of the parameter space increases the number of constraints is tremendously high.

The procedure described by Den Hertog [11] begins with a small finite subset $\mathcal{Q} \subset \mathcal{N}$ i.e. the procedure begins with the problem

$$\text{minimize } \left\{ \frac{1}{2} x^t Q x + c^t x : a_t x + s_t = b_t, \| a_t \| = 1, s_t \geq 0, t \in \mathcal{Q} \right\} \tag{6.5}$$

instead of problem (6.4). The procedure follows the central path until the current iterate violates a constraint in $\mathcal{N}$ which is not in $\mathcal{Q}$. Then, this violated constraint is added (build-up) to the current system of constraints and returns to the previous iterate to continue the process. On the other hand, if the slack value of a constraint in the current iterate is large enough i.e. it is nonbinding in an optimal solution, we remove it (build-down) from the current system of constraints and recenter when necessary. Consequently, all iterates are feasible for the computationally equivalent problem with the constraints in $\mathcal{N}$. This process is repeated until the iterate is close to the optimal solution.

Otherwise, the build-up and down strategy employed by Kaliski, J. et al. [37], which is called the **dynamic mesh strategy**, uses a variable mesh which automatically self-adjusts to achieve machine level precision within the local area with the highest probability of finding an infeasible (or near infeasible) point for the proposed solution. The procedure starts with the problem (6.5), the mesh position $t_0 \in \mathcal{Q}$ with the lowest slack $s_{t_0}$ and a relatively coarse mesh $\mathcal{N}_0$ around the position $t_0$,

$$\mathcal{N}_0 = \prod_{i=1}^{s} I_i^0,$$

where $I_i^0 := \{u_i^0 + (j-1)(v_i^0 - u_i^0)/k_i, j = 1, \ldots, k_i + 1\}$ with $u_i^0 = u_i$, $v_i^0 = v_i$, $i = 1, \ldots, s$. $\mathcal{N}_0$ is the initial coarse grid of mesh points where $k_i$, $i = 1, \ldots, s$ are big enough. If, while checking the initial mesh, a constraint is found to be violated at the current iterate i.e. $s_t \leq 0$, $t \in \mathcal{N}_0$, then the procedure returns this mesh position $t$, and $\mathcal{Q} = \mathcal{Q} \cup \{t\}$. If no violation is found, the mesh position $t$ where there is a constraint with the lowest slack $s_t$, is checked more closely by building a new submesh $\mathcal{N}_1$ around the position $t$,

$$\mathcal{N}_1 = \prod_{i=1}^{s} I_i^1.$$

As before, $I_i^1 := \{u_i^1 + (j-1)(v_i^1 - u_i^1)/k_i, j = 1, \ldots, k_i + 1\}$ and $u_i^0 < u_i^1$, $v_i^0 > v_i^1$, $i = 1, \ldots, s$. The new submesh $\mathcal{N}_1$ is then checked for feasibility. We continue the procedure by generating additional submesh

$$\mathcal{N}_k = \prod_{i=1}^{s} I_i^k,$$

where $I_i^k := \{u_i^k + (j-1)(v_i^k - u_i^k)/k_i, j = 1, \ldots, k_i + 1\}$ and $u_i^{k-1} < u_i^k$, $v_i^k > v_i^{k-1}$, $i = 1, \ldots, s$, until an infeasibility is found or the local mesh precision $\tau_{msh}$ is exceeded, i.e.,

$$\max \left\{ \frac{v_i^k - u_i^k}{k_i}, i = 1, \ldots, s \right\} \leq \tau_{msh}.$$

The nearest located constraint position is returned, if any constraint is found to be violated.

Table 6.1: Dynamic mesh procedure for *SIQP*

```
Procedure: Dynamic_Mesh
Input:
the mesh position $t \in \mathcal{Q}$ with the lowest slack $s_t > 0$;
the initial coarse grid of mesh points $\mathcal{N}_0$ around the position $t$, where
$\tau_c(i) := (v_i - u_i)/k_i$ is the initial mesh precision for the $i^{\text{th}}$ parameter, $i = 1, \ldots, s$;
$x$ is the current iterate to be evaluated;
$\tau_{msh}$ is the mesh precision locally considered;
Output:
$t$ is the mesh position for the most (nearly) violated constraint;
$s_t$ is the constraint slack value for the most (nearly) violated constraint;
begin
    while $\tau_c(i) > \tau_{msh}$ do
        let $t$ be the position in the current mesh with the smallest $s_t$;
        if $s_t \leq 0$ then
            return $t$, $s_t$;
        else
            $\tau_c(i) := \tau_c(i)/k_i$;
            enmesh $\prod_{i=1}^{s} k_i$ points with $\tau_c(i)$ spacing around $t$;
        end if
    end while
    return $t$, $s_t$;
end
```

For the sake of simplicity, we consider that $\mathcal{G}$ contains the constraints $0 \leq x_i \leq 1, i = 1, \ldots, n$, whose index set will be denoted as $\mathcal{J}_0$. In addition to this index set a small subset $\mathcal{J} \subset \mathcal{T} \backslash \mathcal{J}_0$ is chosen to form the initial discretization $\mathcal{Q} := \mathcal{J}_0 \cup \mathcal{J}$ of the set constraints. In Table 6.1 we can see the dynamic mesh algorithm which describe the dynamic mesh strategy.

## 6.3 Some relevant properties of linear varieties, projectors and convex programs

In this section some useful lemmas, which will be used in the next sections, are enunciated. In the following, the vector space of vectors in $I\!R^n$ is denoted by $I\!E_n$.

### 6.3.1 Distance between a point and a linear variety

#### Geometric interpretation

Let $V$ be a linear variety of $I\!R^n$ expressed by $V := \{x \in I\!R^n : Ax = b\}$, with the matrix $A \in \mathcal{M}(m, n)$ of full rank i.e. $\text{rank}(A) = m \leq n$, and where $b \in I\!R^m$. We want to obtain an expression to calculate the distance between a point $p \in I\!R^n$ and a linear variety $V$ which will be denoted by $d(p, V)$. We know that

$$d(p, V) := \min \ \{d(p, x) : x \in V\},$$

or equivalently

$$d(p, V) = \min \ \{\|x - p\| : Ax = b\}.$$

**Lemma 6.3.1** *If the matrix $A \in \mathcal{M}(m, n)$ has full rank there exist the matrices $(A^tA)^{-1} \in \mathcal{M}(m)$ and $(AA^t)^{-1} \in \mathcal{M}(n)$.*

**Lemma 6.3.2** *Consider the following vector subspaces of $I\!E_n$*

$$F := \{u \in I\!E_n : Au = 0\} \text{ and } G := \{v \in I\!E_n : v = A^t\lambda, \lambda \in I\!R^m\},$$

*with the matrix $A \in \mathcal{M}(m, n)$ of full rank. Thus, the linear varieties $W := p + G$ and $V := \{x \in I\!R^n : Ax = b\}$ are perpendicular and $I\!E_n = F \oplus G$.*

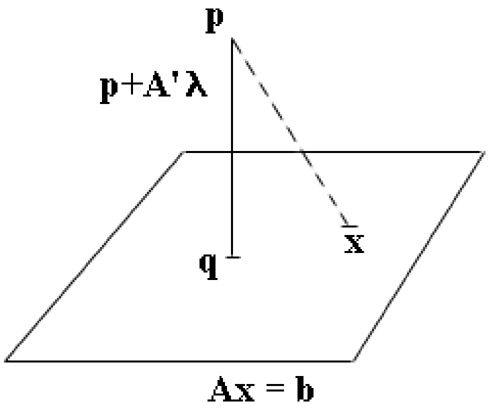

Figure 6.1: The distance from the point $p$ to the linear variety $Ax = b$ can be calculated by using $d(p, q) = \|A^t(AA^t)^{-1}(b - Ap)\|$, where $q$ is the point in $Ax = b$ which satisfies the equation $x = p + A^t\lambda$

.

Proof: Consider $u \in F$ and $v \in G$. Then we can write

$$u^t v = v^t u = (A^t\lambda)^t u = \lambda^t A u = \lambda^t(Au) = 0$$

so we can see that $F$ and $G$ are orthogonal and $F \cap G = \{0\}$. Moreover, we know that $F + G \subset \mathbb{E}_n$, $\dim(F) = n - m$ and $\dim(G) = m$ (because $\text{rank}(A) = \text{rank}(A^t) = m$). From the relationship

$$\dim(F + G) = \dim(F) + \dim(G) - \dim(F \cap G) = n,$$

we have that $\mathbb{E}_n = F + G$ and, together with the property $F \cap G = \{0\}$, we can deduce $\mathbb{E}_n = F \oplus G$, which is the desired conclusion. □

To calculate $d(p, V)$ we only need to obtain the point $q \in V \cap W$ i.e. $q = p + A^t\lambda$ verifying $Aq = b$ (see Figure 6.3.1) and then calculate $d(p, q)$. From the relationship $Aq = b$ we can write

$$A(p + A^t\lambda) = b$$

and by expanding this expression we have

$$Ap + AA^t\lambda = b,$$

so we can calculate $\lambda$ in the form

$$\lambda = (AA^t)^{-1}(b - Ap)$$

and $q = p + A^t(AA^t)^{-1}(b - Ap)$. Hence,

$$d(p, V) = d(p, q) = \|A^t(AA^t)^{-1}(b - Ap)\| \tag{6.6}$$

which is an analytical expression for calculating the distance between a point and a linear variety.

### Interpretation within the field of optimization

The problem can be written as

$$\begin{array}{rl} \text{minimize} & \|p - x\|^2 \\ \text{subject to:} & Ax = b. \end{array}$$

Then, the optimal solution of this program can be obtained directly by using the multiplier rule of the Kuhn-Tucker theorem. Solving the system

$$\left\{ \begin{array}{rcll} \frac{\partial L}{\partial x} & = & 2(x - p) + A^t\lambda & = 0, \\ \frac{\partial L}{\partial \lambda} & = & Ax - b & = 0, \end{array} \right.$$

which derives from the Lagrange function $L(x, \lambda) = \|p - x\|^2 + \lambda^t(Ax - b)$, we have the solution $x = p - \frac{1}{2}A^t\lambda$ where $\lambda = 2(AA^t)^{-1}(Ap - b)$. Thus, the distance is

$$\|p - x\| = \|A^t(AA^t)^{-1}(b - Ap)\|,$$

as before.

## 6.3.2 Projectors

A matrix $P \in \mathcal{M}(n, n)$ is said to be a projector when $P^2 = P$. Therefore $P^t$ is also a projector because $(P^t)^2 = P^t$. Let $Z$ be the matrix which verifies $PZ = 0$. Consider the vector subspaces of $\mathbb{E}_n$

$$F := \{u \in \mathbb{E}_n : u = Z\lambda, \lambda \in \mathbb{R}^n\}$$

known as the null space of $P$ and

$$G := \{v \in \mathbb{E}_n : v = P\lambda, \lambda \in \mathbb{R}^n\}$$

said to be the range space (or column space) of $P$.

**Lemma 6.3.3** *When $P$ is a projector then we have that $\mathbb{E}_n = F \oplus G$. In this case, the rank space $G$ is said to be the projection of $P$ parallel to $F$.*

PROOF: Let $u$ be a vector $u \in F \cap G$. Then, from $u \in G$ we have $u = P\lambda$ so $Pu = P^2\lambda = P\lambda = u$. From $u \in F$ we have $u = Z\lambda_0$ so $Pu = PZ\lambda_0 = 0$. Hence, we obtain $u = Pu = 0$ and we can deduce that $F \cap G = \{0\}$. On the other hand, each vector $u \in \mathbb{E}_n$ can be written $u = (u - Pu) + Pu$ with $u - Pu \in F$ and $Pu \in G$ so $\mathbb{E}_n = F + G$ and the lemma is proved. $\square$

**Lemma 6.3.4** *The eigenvalues of a projector $P$ are $0$ or $1$.*

PROOF: In effect, if we consider $u$ an eigenvector of $P$ then $Pu = tu$, $t \in \mathbb{R}$, so we have

$$Pu = P^2u = tPu = t^2u$$

and we can deduce that $tu = t^2u$ or equivalently $0 = t(t-1)u$, which means that $t = 0$ or $t = 1$. $\square$

**Example 6.3.5** *Consider $Ax + s = b$, $s > 0$ with the matrix $A$ of full rank and $s := (s_1, s_2, \dots, s_m)$. By defining $I_n$ as the identity matrix and $S := diag(s_1, s_2, \dots, s_m)$, then we can see that the matrix*

$$P := S^{-1}A(A^tS^{-2}A)^{-1}A^tS^{-1}$$

*is a projector of $\mathbb{E}_n$ i.e. $P^2 = P$. In this example we also have that $P^t = P$. Moreover, the vectors subspaces $F$ and $G$ are*

$$F := \{u \in \mathbb{E}_n : u = (P - I_n)\lambda, \lambda \in \mathbb{R}^n\},$$

*given that $P(P - I_n) = 0$, and*

$$G := \{v \in \mathbb{E}_n : v = P\lambda, \lambda \in \mathbb{R}^n\}.$$

*Because of the linear system $(A^tS^{-2}A)\bar{\lambda} = A^tS^{-1}\lambda$ is always determinate for each $\lambda \in \mathbb{R}^n$ the subspace $G$ can be rewritten*

$$G = \{v \in \mathbb{E}_n : v = S^{-1}A\bar{\lambda}, \bar{\lambda} \in \mathbb{R}^n\}$$

*where $\bar{\lambda} = (A^tS^{-2}A)^{-1}A^tS^{-1})\lambda$. In this example $F$ and $G$ are orthogonal subspaces. Let $u \in F$ be with $u = (P - I_n)\lambda_1$ and let $v \in G$ be with $v = P\lambda_2$ then we have*

$$v^tu = \lambda_2^tP(P - I_n)\lambda_1 = \lambda_2^t(P^2 - P)\lambda_1 = \lambda_2^t(P - P)\lambda_1 = 0.$$

*$P$ is said to be an orthogonal projector and $G$ the orthogonal projection of $P$ parallel to $F$. Hence,*

$$\|Pu\| \leq \|u\|$$

*because $\|Pu\|^2 = u^tP^tPu = u^tPPu = u^tP^2u = u^tPu \leq \|u\|\|Pu\|$, therefore $\|Pu\|(\|Pu\| - \|u\|) \leq 0$, which proves the last inequality. As a consequence of this inequality and by assumption that $v \notin F$, $\|v\| = 1$ and $v^tPv = 1$ we have*

$$(v \notin F, \|v\| = 1 \text{ and } v^tPv = 1) \Rightarrow (\|Pv\|^2 = 1 \text{ and } v \in G),$$

*because if $v \notin G$ then $v^t(Pv - v) \neq 0$ (since $Pv - v \in F$) and $v^tPv - v^tv = 1 - v^tv \neq 0$ i.e. $\|v\|^2 \neq 1$ which would be a contradiction.*

### 6.3.3 General properties of dual programming problems

Consider the programming problem

$$\begin{array}{rc} \text{minimize} & f(x) \\ \text{subject to:} & g(x) \leq 0, \\ & x \in U \subset I\!R^n, \end{array} \tag{6.7}$$

where $f : I\!R^n \to I\!R$ and $g : I\!R^n \to I\!R^m$. The Lagrangian function of (6.7) is

$$L(x, y) := f(x) + y^tg(x), \tag{6.8}$$

while $y \in I\!R^m_+$ are referred to as Lagrange multipliers. We define the dual function as

$$\Phi(y) := \inf \{L(x, y) : x \in U\}, \tag{6.9}$$

with the domain $D := \{y \in I\!R^m_+$ and $\Phi(y)$ exists $\}$. This function is concave on each convex subset of $D$. Hence, the dual program of (6.7) can be defined

$$\begin{array}{rc} \text{maximize} & \Phi(y) \\ \text{subject to:} & y \in D. \end{array} \tag{6.10}$$

From (6.7) and (6.10) it is easy to see that for all $x$ feasible for (6.7) and for all $y$ feasible for (6.10) we have

$$\Phi(y) \leq f(x).$$

Hence, we have

$$\sup \{\Phi(y) : y \in D\} \leq \inf \{f(x) : g(x) \leq 0, x \in U \subset I\!R^n\}.$$

As a consequence, we can deduce that if one of the programs (6.7) or (6.10) is unbounded then the other program is infeasible. When the program (6.7) is convex then we have the equality

$$\sup\,\{\Phi(y) : y \in D\} = \inf\,\{f(x) : g(x) \leq 0, x \in U \subset I\!R^n\},$$

whenever a suitable constraint qualification holds.

## 6.4 Analysis of the discretized problems

Suppose $\{1, 2, \ldots, m\} \subset \mathcal{T}$. Let

$$\begin{array}{cc} \text{minimize} & f(x) = \frac{1}{2}x^tQx + c^tx \\ \text{subject to:} & a_ix + s_i = b_i, s_i \geq 0, i = 1, 2, \ldots, m, \end{array} \tag{6.11}$$

be a discretized problem of (6.2) which has a strictly feasible solution and the feasible set of constraints is bounded. Consider $b := (b_1, b_2, \ldots, b_m)^t$ and

$$A = \begin{pmatrix} a_1^1 & a_1^2 & \ldots & a_1^i & \ldots & a_1^n \\ a_2^1 & a_2^2 & \ldots & a_2^i & \ldots & a_2^n \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ a_j^1 & a_j^2 & \ldots & a_j^i & \ldots & a_j^n \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ a_m^1 & a_m^2 & \ldots & a_m^i & \ldots & a_m^n \end{pmatrix}, \tag{6.12}$$

which can be assumed of full rank. From (6.8), the Lagrangian function of the program (6.11) is

$$L(x, y) = \frac{1}{2}x^tQx + c^tx + y^t(Ax - b),$$

where $y \in I\!R^m_+$. From (6.9), the dual function of the program (6.11) is

$$\Phi(y) = \min\,\{L(x, y) : x \in I\!R^n\}.$$

Since, in this case, $L(x, y)$ is a positive definite quadratic function in the variable $x$, we have that $\Phi(y)$ exist for all $y$ and it can be obtained by solving the system

$$\frac{\partial L(x, y)}{\partial x} = Qx + c + A^ty = 0 \Leftrightarrow x = -Q^{-1}(c + A^ty).$$

We have $\Phi(y) = L(-Q^{-1}(c + A^ty), y)$. Thus, the dual program of (6.11) can be written

$$\begin{array}{cc} \text{maximize} & \Phi(y) = L(-Q^{-1}(c + A^ty), y) \\ \text{subject to:} & y \in I\!R^m_+. \end{array}$$

By substituting $x = -Q^{-1}(c + A^t y)$ in the terms of the expression

$$\Phi(y) \quad = \quad \tfrac{1}{2}x^t Q x + c^t x + y^t(Ax - b),$$

we can write

$$\begin{array}{lcl}
\frac{1}{2}x^tQx & = & \frac{1}{2}c^tQ^{-1}c + \frac{1}{2}y^tAQ^{-1}c + \frac{1}{2}c^tQ^{-1}A^ty + \frac{1}{2}y^tAQ^{-1}A^ty, \\
c^tx & = & -c^tQ^{-1}c - c^tQ^{-1}A^ty, \\
y^t(Ax-b) & = & -y^tAQ^{-1}c - y^tAQ^{-1}A^ty - y^tb,
\end{array}$$

so

$$\begin{array}{lcl}
\Phi(y) & = & -\frac{1}{2}c^tQ^{-1}c - \frac{1}{2}y^tAQ^{-1}A^ty - y^tAQ^{-1}c - y^tb \\
& = & -\frac{1}{2}(c^tQ^{-1}c + y^tAQ^{-1}c) - \frac{1}{2}(y^tAQ^{-1}A^ty + y^tAQ^{-1}c) - y^tb \\
& = & -\frac{1}{2}(c^t + y^tA)Q^{-1}c - \frac{1}{2}(y^tAQ^{-1}(A^ty + c) - y^tb \\
& = & -\frac{1}{2}(-x^tQ)Q^{-1}c - \frac{1}{2}(y^tAQ^{-1}(-Qx) - y^tb \\
& = & -\frac{1}{2}x^tc + \frac{1}{2}y^tAx - y^tb \\
& = & -\frac{1}{2}x^tc + \frac{1}{2}(-Qx - c)^t - y^tb \\
& = & -\frac{1}{2}x^tQx - y^tb.
\end{array}$$

Thus, the dual problem of (6.11) can be rewritten

$$\begin{array}{rc}
\text{maximize} & h(x,y) = -\frac{1}{2}x^tQx - y^tb \\
\text{subject to:} & Qx + c + A^ty = 0, \\
 & y \geq 0.
\end{array} \tag{6.13}$$

When the primal and dual feasible points are $x$ and $(x, y)$ respectively we know that the **gap** between the objective function of (6.11) and (6.13) is

$$\frac{1}{2}x^tQx + c^tx - (-\frac{1}{2}x^tQx - y^tb) = y^ts. \tag{6.14}$$

On the other hand, a barrier function for (6.11) is

$$F_\mu(x) = \frac{x^tQx/2 + c^tx}{\mu} - \sum_{i=1}^{m} \ln\ s_i, \tag{6.15}$$

with $s_i = b_i - a_i^t x > 0, i = 1, 2, \ldots, m$, and where $\mu > 0$ indicates the parameter of barrier. The gradient $\nabla F_\mu(x)$ and the Hessian $H(x, \mu)$ of the barrier function (6.15) can be expressed by

$$\nabla F_\mu(x) = \frac{1}{\mu}(x^tQ + c^t) + A^tS^{-1}e \tag{6.16}$$

and

$$H(x,\mu) = \frac{1}{\mu}Q + A^tS^{-2}A. \tag{6.17}$$

The barrier function is a convex function with only one global minimum, which is an interior point of the feasible domain of the problem (6.11). The necessary and sufficient first order optimality conditions for the optimal point of the barrier function are:

$$\frac{\partial F_\mu(x)}{\partial x} = \frac{1}{\mu}(Qx + c) + A^t S^{-1} e = 0 \tag{6.18}$$

where $s \geq 0$ and $e := (1, 1, \ldots, 1)^t$. By using $y := \mu S^{-1} e \geq 0$ in the expression (6.18) the first order optimality conditions can be rewritten

$$\begin{array}{rcl} Ax + s & = & b,\ s \geq 0, \\ Qx + c + A^t y & = & 0,\ y \geq 0, \\ Sy & = & \mu e. \end{array} \tag{6.19}$$

Notice, that when $\mu = 0$ the first order optimality conditions (6.19) are the first order optimality conditions for the primal and the dual programs (6.11) and (6.13) respectively. The solution $(x(\mu),\ y(\mu),\ s(\mu))$ of the system (6.19) defines the central paths $x(\mu)$ and $(x(\mu), y(\mu))$ of the primal and dual problems respectively. From the third equality in (6.19) we can deduce that the duality gap in this solution satisfies

$$y(\mu)^t s(\mu) = s(\mu)^t y(\mu) = \mu m. \tag{6.20}$$

It is well-known (see Den Hertog [11]) that $x(\mu)$ and $y(\mu)$ are continuously differentiable. From (6.14) and (6.20) it can be seen that $x(\mu)$ and $(x(\mu), y(\mu))$ converge to the optimal primal and optimal dual solutions respectively when $\mu$ converges to 0. The next lemma proves that along the primal path the objective function of the primal problem is decreasing and along the dual path the objective function of the dual problem is increasing.

**Lemma 6.4.1** *Let $(x(\mu),\ y(\mu),\ s(\mu))$ be the solution of the system (6.19), which defines the central paths $x(\mu)$ and $(x(\mu), y(\mu))$ of the primal and dual problems respectively. The function $f(x(\mu))$ where $f(x)$ is the objective function of the primal problem (6.11) is monotonically decreasing and the function $h(x(\mu), y(\mu))$ where $h(x, y)$ is the objective function of the dual problem (6.13) is monotonically increasing, as $\mu$ decreases.*

PROOF: Consider $\mu_1 < \mu_2$. Since $x(\mu_1)$ and $x(\mu_2)$ minimize $F_{\mu_1}(x)$ and $F_{\mu_2}(x)$ respectively we have

$$F_{\mu_1}(x(\mu_1)) \leq F_{\mu_1}(x(\mu_2))$$

and

$$F_{\mu_2}(x(\mu_2)) \leq F_{\mu_2}(x(\mu_1)).$$

Thus, we can write the equivalent expressions

$$\frac{f(x(\mu_1))}{\mu_1} - \sum_{i=1}^{m} \ln s_i(\mu_1) \leq \frac{f(x(\mu_2))}{\mu_1} - \sum_{i=1}^{m} \ln s_i(\mu_2)$$

and

$$\frac{f(x(\mu_2))}{\mu_2} - \sum_{i=1}^{m} \ln s_i(\mu_2) \leq \frac{f(x(\mu_1))}{\mu_2} - \sum_{i=1}^{m} \ln s_i(\mu_1).$$

By adding these two inequalities we obtain

$$\frac{f(x(\mu_1))}{\mu_1} - \frac{f(x(\mu_2))}{\mu_1} \leq \frac{f(x(\mu_1))}{\mu_2} - \frac{f(x(\mu_2))}{\mu_2},$$

or equivalently

$$\left(\frac{1}{\mu_1} - \frac{1}{\mu_2}\right)(f(x(\mu_1)) - f(x(\mu_2))) \leq 0.$$

As $\mu_1 < \mu_2$, we have that $\frac{1}{\mu_1} > \frac{1}{\mu_2}$ and $f(x(\mu_1)) \leq f(x(\mu_2))$ so the first part of the lemma follows.

The second part of the lemma can be proved in a similar way by using the dual logarithmic barrier function

$$G_\mu(x,y) := -\frac{h(x,y)}{\mu} - \sum_{i=1}^{m} \ln y_i, \text{ with } Qx + c + A^t y = 0.$$

As before, we have

$$-\frac{h(x(\mu_1), y(\mu_1))}{\mu_1} - \sum_{i=1}^{m} \ln y_i(\mu_1) \leq -\frac{h(x(\mu_2), y(\mu_2))}{\mu_1} - \sum_{i=1}^{m} \ln y_i(\mu_2)$$

and

$$-\frac{h(x(\mu_2), y(\mu_2))}{\mu_2} - \sum_{i=1}^{m} \ln y_i(\mu_2) \leq -\frac{h(x(\mu_1), y(\mu_1))}{\mu_2} - \sum_{i=1}^{m} \ln y_i(\mu_1).$$

Then, by adding the two inequalities we obtain

$$\frac{-h(x(\mu_1), y(\mu_1)) + h(x(\mu_2), y(\mu_2))}{\mu_1} \leq \frac{-h(x(\mu_1), y(\mu_1)) + h(x(\mu_2), y(\mu_2))}{\mu_2},$$

or equivalently

$$\left(\frac{1}{\mu_1} - \frac{1}{\mu_2}\right)(h(x(\mu_2), y(\mu_2)) - h(x(\mu_1), y(\mu_1)))) \leq 0,$$

so $h(x(\mu_2), y(\mu_2)) \leq h(x(\mu_1), y(\mu_1))$ when $\mu_1 < \mu_2$, which proves the second part of the lemma. $\square$

## 6.5 Distance from a noncentered point to the central path

A noncentered point of the primal program (6.11) is a feasible point which does not satisfy the system (6.19). To calculate the distance from the noncentered points to the central path, several measures can be introduced. The first is analogous to the Roos and Vial measure for linear programming (see for instance Den Hertog [11] or Zhi-Quan Luo et al. [78]). For each interior point $x$ we define the measure

$$\delta_1(x,\mu) := \min{}_y \left\{ \left\| \frac{Sy}{\mu} - e \right\| : Qx + c + A^t y = 0 \right\}, \tag{6.21}$$

which can be considered as the distance to the central path. The unique solution to the problem (6.21) is denoted by $y(x,\mu)$.

**Lemma 6.5.1** *For the distance $\delta_1(x,\mu)$ we have the following properties:*

*i) if $x$ is an interior point of the feasible domain of the problem (6.11) then*

$$\delta_1(x,\mu) = 0 \Leftrightarrow x = x(\mu),$$

*and moreover*

$$\delta_1(x,\mu) = 0 \Rightarrow y(x,\mu) = y(\mu).$$

*ii) By using the vector $v_1(x,\mu)$ satisfying*

$$(A^t S^{-2} A) v_1(x,\mu) = -\nabla F_\mu(x)^t, \tag{6.22}$$

*we can obtain the expression*

$$\delta_1(x,\mu) = \| S^{-1} A v_1(x,\mu) \|. \tag{6.23}$$

PROOF: The property i) can easily be verified. To verify ii) we define $t := \frac{Sy}{\mu}$ which is equivalent to $y = \mu S^{-1} t$. Hence (6.21) can be rewritten

$$\delta_1(x,\mu) = \min{}_t \left\{ \|t - e\| : (A^t S^{-1}) t = -\frac{Qx + c}{\mu} \right\},$$

where the right hand side of the equality is the distance from the point $e$ to the linear variety defined by $(A^t S^{-1}) t = -\frac{Qx+c}{\mu}$. By taking into account the expression (6.6) (which calculates the distance between a point and a linear variety) we obtain

$$\delta_1(x,\mu) = \left\| (S^{-1} A) v_1(x,\mu) \right\|,$$

where using (6.16) $v_1(x,\mu)$ satisfies

$$(A^tS^{-2}A)v_1(x,\mu) = -\frac{Qx+c}{\mu} - A^tS^{-1}e = -\nabla F_\mu(x)^t,$$

which completes the proof. □

Another measure for the distance to the central path can be obtained by using the norm defined by the Hessian $H(x,\mu)$ defined in (6.17), because the Hessian of a convex function is positive definite. Hence, we can define the new measure

$$\delta_2(x,\mu) = \|v_2(x,\mu)\|_H := \sqrt{v_2(x,\mu)^t H(x,\mu) v_2(x,\mu)}, \tag{6.24}$$

where $v_2(x,\mu)$ is the Newton step for the barrier function from the point $x$, i.e., the solution to the linear system

$$H(x,\mu)v_2(x,\mu) = -\nabla F_\mu(x)^t = -\frac{Qx+c}{\mu} - A^tS^{-1}e. \tag{6.25}$$

In this case, we also have that if $x$ is an interior point of the feasible domain of the problem (6.11), then

$$\delta_2(x,\mu) = 0 \Leftrightarrow x = x(\mu).$$

As demonstrated in Den Hertog [11] there exists a close connection between both measures. Let $\mu$ be a nonnegative real number and let $x$ be a feasible point of (6.11) with $y(x,\mu)$ the optimal solution to the problem (6.21). Then, we have

$$\delta_2^2(x,\mu) = -v_2(x,\mu)^t \nabla F_\mu(x)^t \le \delta_1^2(x,\mu). \tag{6.26}$$

## 6.6 Relationship between the solutions to the semi-infinite and the discretized problems

The following lemmas are needed to relate the approximate solutions to (6.11) and (6.2).

**Lemma 6.6.1** *Let $x$ be a feasible point of (6.11) and let $y(x,\mu)$ be the optimal solution to the problem (6.21) such that $\delta_1(x,\mu) \le 1$. Then, $y(x,\mu)$ is dual feasible and $\mu(m - \delta_1(x,\mu)\sqrt{m}) \le s^t y(x,\mu) \le \mu(m + \delta_1(x,\mu)\sqrt{m})$.*

PROOF: The last of the following equivalences

$$\delta_1(x,\mu) = \left\|\frac{Sy(x,\mu)}{\mu} - e\right\| \le 1 \Leftrightarrow -e \le \frac{Sy(x,\mu)}{\mu} - e \le e \Leftrightarrow$$

$$\Leftrightarrow 0 \leq \frac{Sy(x,\mu)}{\mu} \leq 2e \Leftrightarrow 0 \leq y(x,\mu) \leq 2\mu S^{-1}e,$$

shows that $y(x,\mu) \geq 0$ which together the property that $y(x,\mu)$ is the optimal solution to the problem (6.21) proves that it is dual feasible. On the other hand, from the Cauchy Schwarz inequality we have

$$\delta_1(x,\mu)\sqrt{m} = \left\| \frac{Sy(x,\mu)}{\mu} - e \right\| \|e\| \geq \left| \frac{s^t y(x,\mu)}{\mu} - m \right|.$$

Then, by using the properties of the absolute value we obtain the following inequalities

$$m - \delta_1(x,\mu)\sqrt{m} \leq \frac{s^t y(x,\mu)}{\mu} \leq m + \delta_1(x,\mu)\sqrt{m}.$$

Multiplying all the terms of the above inequalities by $\mu$ we can deduce

$$\mu(m - \delta_1(x,\mu)\sqrt{m}) \leq s^t y(x,\mu) \leq \mu(m + \delta_1(x,\mu)\sqrt{m}), \tag{6.27}$$

which proves the lemma. □

Lemma 6.6.1 shows that if we can find a feasible point $x$ of (6.11) making $y(x,\mu)$ the optimal solution to the problem (6.21) and such that $\delta_1(x,\mu) \leq 1$ and $\mu \leq \epsilon/(m+\sqrt{m})$ then $x$ is an $\epsilon$-minimizer of (6.11) because from (6.27) we can write

$$\frac{1}{2}x^tQx + c^tx - q^* \leq s^t y(x,\mu) \leq \mu(m + \delta_1(x,\mu)\sqrt{m}) \leq \mu(m+\sqrt{m}) \leq \epsilon,$$

where $q^*$ is the optimal value of (6.11) or (6.13).

The next lemma shows that $x$ is an $\epsilon$-minimizer of (6.2) when it satisfies the additional condition of being feasible for (6.2).

**Lemma 6.6.2** *Let $\mathcal{T}$ be a compact set of $I\!R^s$ and let $\mathcal{N}$ be a finite subset of $\mathcal{T}$. Suppose that the mappings $t \rightarrow a_t$ and $t \rightarrow b_t$ are continuous functions on $\mathcal{T}$ (where $a_tx - b_t \leq 0,\ t \in \mathcal{T}$ define the set constraints of (6.2)). If $x \in \mathcal{G}$ and satisfies $\delta_1(x,\mu) \leq 1$ with $\mu \leq \epsilon/(m+\sqrt{m})$, then $x$ is an $\epsilon$-minimizer of (6.2).*

Proof: Let $\mathcal{Q}$ be a finite set satisfying $\mathcal{N} \subset \mathcal{Q} \subset \mathcal{T}$. In that follows $\bar{A}$ will denote the matrix associated with the set $\mathcal{Q}$. Then, the variable $\bar{y}^t := (y(x,\mu), 0)$ is a dual feasible solution to the problem

$$q_r^* := \min\left\{\frac{1}{2}x^tQx + c^tx :\ a_tx - b_t \leq 0,\ t \in \mathcal{Q}\right\},$$

because $Qx + c + \bar{A}^t\bar{y} = Qx + c + A^t y = 0$ and $\bar{y} \geq 0$ (with $r := \operatorname{card}(\mathcal{Q})$). Furthermore, the duality gap $\bar{s}^t\bar{y} = s^t y$ remains unchanged. Hence, by using Lemma 6.6.1 we obtain the inequality

$$\frac{1}{2}x^tQx + c^tx - q_r^* \leq \bar{s}^t\bar{y} = s^ty \leq \mu(m + \delta_1(x,\mu)\sqrt{m}) \leq \epsilon. \tag{6.28}$$

Now, by applying Reemtsen's generalization of the Gustafson Theorem (see [55]), which takes into account the compactness of $\mathcal{T}$ and the continuity of the functions $t \to a_t$ and $t \to b_t$, the values $q_r^*$ converge to the minimum $q^*$ of the problem (6.2) (for certain sequences of index sets $\mathcal{Q}$ with increasing cardinality). Hence, letting $r \to +\infty$ in the inequality (6.28), we obtain

$$\frac{1}{2}x^tQx + c^tx - q^* \leq \epsilon,$$

which shows that $x$ is indeed an $\epsilon$-minimizer of (6.2). □

## 6.7 The effect of adding a constraint

In the following we denote by $(A^tS^{-2}A)_{\mathcal{Q}}$ the matrix whose columns are in the index set $\mathcal{Q}$ and we assume that it has full rank. Moreover, we define

$$\|a\|_{\mathcal{Q}} := \sqrt{a^t(A^tS^{-2}A)_{\mathcal{Q}}^{-1}a} \tag{6.29}$$

where $a \in I\!\!R^n$. Sometimes, the index set $\mathcal{Q}$ will be referred to as $\{1, 2, \ldots, m\}$ and

$$\chi_i := \frac{s_i}{\|a_i\|_{\mathcal{Q}}}, i = 1, 2, \ldots, m$$

can be interpreted as the distance to the $i^{\text{th}}$ constraint in a certain metric.

Suppose that a new constraint $a_{m+1}^t x \leq b_{m+1}$ is added with $s_{m+1} > 0$ the corresponding slack variable. The next lemma shows that if the new constraint is far enough away from the current iterate then adding the constraint hardly influences the distance $\delta_1(x,\mu)$. Otherwise, we have a bound for the new distance $\bar{\delta}_1(x,\mu)$.

**Lemma 6.7.1** *The new distance $\bar{\delta}_1(x,\mu)$ is bounded in the form*

$$\bar{\delta}_1(x,\mu) \leq \begin{cases} \frac{1+\delta_1\chi_{m+1}}{\sqrt{1+\chi_{m+1}^2}} & \text{if } \chi_{m+1} > \delta_1, \\ \\ 1 + \delta_1^2 & \text{if } \chi_{m+1} \leq \delta_1. \end{cases} \tag{6.30}$$

*Notice that if $\chi_{m+1}$ is large then adding the new constraint $a_{m+1}^t x \leq b_{m+1}$ hardly influences the new distance $\bar{\delta}_1(x,\mu)$.*

PROOF: From the definition (6.21) we have

$$\bar{\delta}_1(x,\mu)^2 = \min_{(\bar{y},\xi)} \left\{ \left\| \frac{1}{\mu} \begin{pmatrix} S\bar{y} \\ s_{m+1}\xi \end{pmatrix} - \begin{pmatrix} e \\ 1 \end{pmatrix} \right\|^2 : Qx + c + A^t\bar{y} + \xi a_{m+1} = 0 \right\}$$

that can be written

$$\bar{\delta}_1(x,\mu)^2 = \min_{(\bar{y},\xi)} \left\{ \left\| \frac{S\bar{y}}{\mu} - e \right\|^2 + \left( \frac{s_{m+1}\xi}{\mu} - 1 \right)^2 : Qx + c + A^t\bar{y} + \xi a_{m+1} = 0 \right\}.$$

Consider $\triangle y := \bar{y} - y(x,\mu)$, where $y(x,\mu)$ solves the problem (6.21). Hence,

$$\bar{\delta}_1(x,\mu)^2 = \min_{(\triangle y,\xi)} \left\{ \left\| \frac{Sy(x,\mu)}{\mu} - e + \frac{S\triangle y}{\mu} \right\|^2 + \left( \frac{s_{m+1}\xi}{\mu} - 1 \right)^2 : A^t\triangle y + \xi a_{m+1} = 0 \right\}.$$

From the relationship

$$\left\| \frac{Sy(x,\mu)}{\mu} - e + \frac{S\triangle y}{\mu} \right\|^2 \leq \left( \left\| \frac{Sy(x,\mu)}{\mu} - e \right\| + \left\| \frac{S\triangle y}{\mu} \right\| \right)^2 = \left( \delta_1 + \left\| \frac{S\triangle y}{\mu} \right\| \right)^2$$

we can write

$$\bar{\delta}_1(x,\mu)^2 \leq \min_{(\triangle y,\xi)} \left\{ \left( \delta_1 + \left\| \frac{S\triangle y}{\mu} \right\| \right)^2 + \left( \frac{s_{m+1}\xi}{\mu} - 1 \right)^2 : A^t\triangle y + \xi a_{m+1} = 0 \right\}.$$

The objective value of the program in the second term of this expression can be substituted by [1]

$$\min_{(\triangle y,\xi)} \left\{ (\delta_1 + \|S\triangle y\|)^2 + (s_{m+1}\xi - 1)^2 : A^t\triangle y + \xi a_{m+1} = 0 \right\}$$

so we can write

$$\bar{\delta}_1(x,\mu)^2 \leq \min_{(\triangle y,\xi)} \left\{ (\delta_1 + \|S\triangle y\|)^2 + (s_{m+1}\xi - 1)^2 : A^t\triangle y + \xi a_{m+1} = 0 \right\}.$$

[1]Let $k$ be a nonzero real number. It is clear that the programming problems

$$(x^* - a)^2 + (y^* - b)^2 = \min_{(x,y)} \{(x-a)^2 + (y-b)^2 : tx + sy = 0\}$$

and

$$(\frac{\bar{x}}{k} - a)^2 + (\frac{\bar{y}}{k} - b)^2 = \min_{(x,y)} \{(\frac{x}{k} - a)^2 + (\frac{y}{k} - b)^2 : tx + sy = 0\}$$

have the same objective value because both points $(kx^*, ky^*)$ and $(\frac{\bar{x}}{k}, \frac{\bar{y}}{k})$ are feasible points for these programs.

By using the expression (6.6), which calculates the distance from a point to a linear variety, we have that $\triangle y = -\xi S^{-2}A(A^tS^{-2}A)^{-1}a_{m+1}$ minimizes $\|S\triangle y\|$ over $A^t\triangle y + \xi a_{m+1} = 0$. Then,

$$\|S\triangle y\|^2 = \xi^2 a_{m+1}^t (AS^{-2}A^t)^{-1} a_{m+1} = \xi^2 \|a_{m+1}\|_Q^2$$

and

$$\|S\triangle y\| = |\xi| \|a_{m+1}\|_Q$$

so we can write

$$\bar{\delta}_1(x,\mu)^2 \leq \min_\xi \left\{ (\delta_1 + |\xi|\|a_{m+1}\|_Q)^2 + (s_{m+1}\xi - 1)^2 \right\}.$$

Defining the function

$$\psi(\xi) := (\delta_1 + |\xi|\|a_{m+1}\|_Q)^2 + (s_{m+1}\xi - 1)^2,$$

its optimal values must be found among the values

$$\xi = \frac{s_{m+1} - \delta_1\|a_{m+1}\|_Q}{\|a_{m+1}\|_Q^2 + s_{m+1}^2} \text{ and } \xi = \frac{s_{m+1} + \delta_1\|a_{m+1}\|_Q}{\|a_{m+1}\|_Q^2 + s_{m+1}^2},$$

which have been obtained by solving the equation $\psi'(\xi) = 0$ i.e.

$$2\|a_{m+1}\|_Q (\delta_1 + \xi\|a_{m+1}\|_Q) + 2s_{m+1}(s_{m+1}\xi - 1) = 0$$

when $|\xi| = \xi$ or

$$-2\|a_{m+1}\|_Q (\delta_1 - \xi\|a_{m+1}\|_Q) + 2s_{m+1}(s_{m+1}\xi - 1) = 0$$

when $|\xi| = -\xi$. By substituting these values in the function $\psi(\xi)$ we can see that it is minimal for

$$\xi = \frac{s_{m+1} - \delta_1\|a_{m+1}\|_Q}{\|a_{m+1}\|_Q^2 + s_{m+1}^2} = \frac{\chi_{m+1} - \delta_1}{\|a_{m+1}\|_Q(1 + \chi_{m+1}^2)}$$

with the objective value

$$\psi\left(\frac{s_{m+1} - \delta_1\|a_{m+1}\|_Q}{\|a_{m+1}\|_Q^2 + s_{m+1}^2}\right) = \frac{(1 + \delta_1\chi_{m+1})^2}{1 + \chi_{m+1}^2}.$$

If $\chi_{m+1} > \delta_1$, then

$$\bar{\delta}_1^2(x,\mu) \leq \frac{(1 + \delta_1\chi_{m+1})^2}{1 + \chi_{m+1}^2},$$

or else $\chi_{m+1} \leq \delta_1$ and we can write

$$\bar{\delta}_1^2(x,\mu) \leq \frac{(1+\delta_1\chi_{m+1})^2}{1+\chi_{m+1}^2} \leq \frac{(1+\delta_1^2)^2}{1+\chi_{m+1}^2} \leq (1+\delta_1^2)^2,$$

which proves the lemma. □

Let $x(\mu)$ be the center for the barrier function $F_\mu(x)$, and let $\bar{x}(\mu)$ be the new center for the barrier function $\bar{F}_\mu(x)$ which has been obtained after adding the new constraint $a_{m+1}^t x \leq b_{m+1}$ to the set constrains of the programming problem (6.11). Denote by $\bar{s}_{m+1}(\mu) := b_{m+1} - a_{m+1}^t \bar{x}(\mu)$ and $s_{m+1}(\mu) := b_{m+1} - a_{m+1}^t x(\mu)$ the slack variables for the new and the old centers respectively. The next lemma states the relationship between $\bar{s}_{m+1}(\mu)$ and $s_{m+1}(\mu)$.

**Lemma 6.7.2** *Let $a_{m+1}^t x \leq b_{m+1}$ be the constraint to be added. Then,*

$$\bar{s}_{m+1}(\mu) \geq s_{m+1}(\mu).$$

Proof: If $s_{m+1}(\mu) \leq 0$ the lemma is true because $\bar{s}_{m+1}(\mu) > 0$. Otherwise, we can suppose that both centers $x(\mu)$ and $\bar{x}(\mu)$ are feasible points for the old and the new feasible sets. Then, using 6.15, we have $F_\mu(x(\mu)) \leq \bar{F}_\mu(x(\mu))$ and $\bar{F}_\mu(\bar{x}(\mu)) \leq F_\mu(\bar{x}(\mu))$ and we can write

$$\begin{array}{rclclcl} \ln(s_{m+1}(\mu)) & = & F_\mu(x(\mu)) - \bar{F}_\mu(x(\mu)) & \leq & F_\mu(x(\mu)) - \bar{F}_\mu(\bar{x}(\mu)) & & \\ & \leq & F_\mu(x(\mu)) - \bar{F}_\mu(\bar{x}(\mu)) & \leq & F_\mu(\bar{x}(\mu)) - \bar{F}_\mu(\bar{x}(\mu)) & = & \ln(\bar{s}_{m+1}(\mu)). \end{array}$$

which prove the lemma. □

The next lemma gives us an upper bound for the barrier function value of the new center after adding a constraint. In the following we refer to the barrier function after adding a new cut as $F_\mu^{m+1}(x)$, and to the new center as $x^{m+1}(\mu)$. Moreover, we denote

$$\|a_t\|_m := \|a_t\|_{\mathcal{Q}} \tag{6.31}$$

and

$$\mathcal{G}_m := \{x \in I\!R^n : \ a_t x + s_t = b_t, \| \ a_t \ \| = 1, s_t \geq 0, t \in \mathcal{Q}\}, \tag{6.32}$$

where $\mathcal{Q} = \{1,2,\ldots,m\}$.

**Lemma 6.7.3** *Suppose that a new cut $a_{m+1}^t x \leq b_{m+1}$ is added to the set $\mathcal{G}_m$ to obtain $\mathcal{G}_{m+1}$. Let $x$ be an interior point of $\mathcal{G}_m$ with $\delta_1(x,\mu) \leq 0.25$ and the current* *slack value* $s_{m+1} = b_{m+1} - a_{m+1}^t x > 0$. *Then*

$$F_\mu^{m+1}(x) - F_\mu^{m+1}(x^{m+1}(\mu)) \leq \frac{1}{3} + max \left\{0, \ln \frac{4}{\chi_{m+1}}\right\}$$

*where* $\chi_{m+1} = s_{m+1}/\|a_{m+1}\|_m$.

Proof: Consider $\chi_{m+1} \geq 4$. According to Lemma 6.7.1 and taking into account that $\frac{t}{\sqrt{1+t^2}} \leq 1$ we can write the inequalities

$$\bar{\delta}_1(x,\mu) \leq \frac{1+\delta_1\chi_{m+1}}{\sqrt{1+\chi_{m+1}^2}} \leq \delta_1 + \frac{1}{\sqrt{1+\chi_{m+1}^2}} \leq \frac{1}{4} + \frac{1}{\sqrt{1+16}} \leq \frac{1}{2}.$$

From the property (see Den Hertog [11] pag 41)

$$F_\mu(x) - F_\mu(x(\mu)) \leq \frac{\delta_1^2(x,\mu)}{1-\delta_1^2(x,\mu)}.$$

when $\delta_1(x,\mu) < 1$, we can write

$$F_\mu^{m+1}(x) - F_\mu^{m+1}(x^{m+1}(\mu)) \leq \frac{\bar{\delta}_1^2}{1-\bar{\delta}_1^2} \leq \frac{1/4}{1-1/4} = \frac{1}{3}. \tag{6.33}$$

On the other hand, suppose $\chi_{m+1} < 4$. In this case we add an auxiliary constraint in the form $a_0^t x + s_0 = b_0$ with $\chi_0 = s_0/\|a_0\|_0 = 4$. Hence, we can write

$$\begin{aligned} F_\mu^{m+1}(x) - F_\mu^{m+1}(x^{m+1}(\mu)) &= F_\mu^0(x^0(\mu)) - F_\mu^{m+1}(x^{m+1}(\mu)) \\ &+ F_\mu^0(x) - F_\mu^0(x^0(\mu)) \\ &+ F_\mu^{m+1}(x) - F_\mu^0(x). \end{aligned} \tag{6.34}$$

The first term is smaller than or equal to 0, since the barrier function in the center increases after shifting a constraint (see Den Hertog [11]). For the second term, the property (6.33) gives us again the inequality

$$F_\mu^0(x) - F_\mu^0(x^0(\mu)) \leq \frac{1}{3}.$$

Finally, for the third term

$$F_\mu^{m+1}(x) - F_\mu^0(x) = \ln s_0 - \ln s_{m+1} = \ln \frac{4\|a_0\|_0}{s_{m+1}} = \ln \frac{4}{\chi_{m+1}}.$$

Substituting this expressions into (6.34) we obtain

$$F_\mu^{m+1}(x) - F_\mu^{m+1}(x^{m+1}(\mu)) \leq \frac{1}{3} + \ln \frac{4}{\chi_{m+1}}. \tag{6.35}$$

The expressions (6.33) and (6.35) prove the lemma. □

The following corollary estimates the variation in the barrier function after a new cut is added.

**Corollary 6.7.4** *The hypotheses are the same as above. If* $\chi_{m+1} \leq 4$ *then*

$$F_\mu^m(x) - F_\mu^{m+1}(x^{m+1}(\mu)) \leq \frac{1}{3} + \ln(4\|a_{m+1}\|_m).$$

Proof: From $\chi_{m+1} \leq 4$ we have

$$F_\mu^{m+1}(x) - F_\mu^{m+1}(x^{m+1}(\mu)) \leq \frac{1}{3} + \ln\frac{4}{\chi_{m+1}}.$$

or equivalently

$$F_\mu^m(x) - F_\mu^{m+1}(x^{m+1}(\mu)) \leq \frac{1}{3} + \ln\frac{4\|a_{m+1}\|_m}{s_{m+1}} + \ln s_{m+1} = \frac{1}{3} + \ln(4\|a_{m+1}\|_m),$$

which proves the corollary. □

Table 6.2: Center and add constrains procedure for *SIQP*

```
Procedure: Center_and_Add_constraints
Input:
$0 < \zeta < 1$ is a centering parameter;
$\mathcal{Q} = \mathcal{J}_0 \cup \mathcal{I}$ is the index set of current constraints;
$x$ is the current iterate;
Output:
$\mathcal{Q}$ is the new set of current constraints;
$x$ is the centered solution from $\mathcal{Q}$;
begin
    while $\delta(x,\mu) > \zeta$ do
        $\tilde{x} := x$;
        compute $v := v(x,\mu)$ by using (6.22);
        $\tilde{\lambda} :=$ arg min $_{\lambda>0}\{F_\mu(x+\lambda v) : s_t > 0, t \in \mathcal{Q}\}$;
        $x := x + \tilde{\lambda} v$;
        if $\exists t \notin \mathcal{Q} : s_t \leq 0$ (by using the Dynamic_Mesh procedure) then
            $x := \tilde{x}$;
            $\mathcal{Q} := \mathcal{Q} \cup \{t\}$;
        end if
    end while
end
```

To complete the analysis, it is necessary to bound the maximum number of cuts generated as well as the number of iterations required to recenter the iterate after a new cut is introduced. The next lemma provides an upper bound for $F_\mu^m(x_m(\mu))$.

**Lemma 6.7.5** *We assume that the feasible set $\mathcal{G}$ (defined in (6.1)) contains an Euclidean ball of radius $\nu_0$ centered at $\widetilde{x}$. Then, for all $m \geq 2n$ there exists a constant $M$ such that*

$$F_\mu^m(x_m(\mu)) \leq \frac{M\sqrt{n}}{\mu} - m \ln \nu_0. \tag{6.36}$$

Proof: Consider the inequality

$$F_\mu^m(x_m(\mu)) \leq F_\mu^m(\widetilde{x}) = \frac{\frac{1}{2}\widetilde{x}^t Q \widetilde{x} + c^t \widetilde{x}}{\mu} - \sum_{i=1}^m \ln s_i(\widetilde{x}).$$

We have that $\|\widetilde{x}\|_\infty \leq 1$. By defining $M := \|\frac{1}{2}Q\widetilde{x} + c\|$, the Cauchy-Schwartz inequality implies

$$\left|\widetilde{x}^t \left(\frac{1}{2}Q\widetilde{x} + c\right)\right| \leq M\sqrt{n},$$

or equivalently

$$-M\sqrt{n} \leq \widetilde{x}^t \left(\frac{1}{2}Q\widetilde{x} + c\right) \leq M\sqrt{n}. \tag{6.37}$$

Taking into account that $s_i(\widetilde{x}) \geq \nu_0$, $i = 1, \ldots, m$ the lemma is proved. $\square$

Notice, that we have assumed that the feasible set $\mathcal{G}$ contains an Euclidean ball of radius $\nu_0$ centered at $\widetilde{x}$ (which is an interior point of $\mathcal{G}$). Moreover, we have $\mathcal{G} \subset \mathcal{G}_m \subset [0,1]^n$. Hence, there exists an Euclidean ball of radius $\nu_1$ centered at $\widetilde{x}$ which contains $[0,1]^n$ and the following relationship can be written.

$$B(\widetilde{x}, \nu_0) \subset \mathcal{G} \subset \mathcal{G}_m \subset [0,1]^n \subset B(\widetilde{x}, \nu_1). \tag{6.38}$$

Now, we will obtain simple bounds for the slack variables $s_i$, $i = 1, \ldots, m$.

**Lemma 6.7.6** *Let $\mathcal{G}_j$, $j = 1, \ldots, m$ be as defined in (6.32). Then, we have*

*i)* $0 \leq s_j \leq 1$, $j = 1, \ldots, 2n$,

*ii)* $0 \leq s_j \leq \sqrt{n} + \nu_1$, $j = 2n+1, \ldots, m$.

Proof:

i) From $0 \leq x_j \leq 1$, $j = 1, \ldots, n$, the slack variables $s_j = 1 - x_j$ and $s_{n+j} = x_j$, $j = 1, \ldots, n$ satisfy $0 \leq s_j \leq 1$, $j = 1, \ldots, 2n$.

ii) For all $j = 2n+1, \ldots, m$, we have

$$\begin{array}{rcll} 0 < s_j & = & b_j - a_j^t x & \\ & = & s_j(\widetilde{x}) + a_j^t \widetilde{x} - a_j^t x & \\ & \leq & \nu_1 + \|a_j\| \|\widetilde{x} - x\| & \\ & \leq & \nu_1 + \|\widetilde{x} - x\| & (\text{ from } \|a_j\| = 1) \\ & \leq & \nu_1 + \sqrt{n} & (\text{ from } \mathcal{G} \subset [0,1]^n). \end{array}$$

□

**Lemma 6.7.7** *Let $z^j = (z_1^j, \ldots, z_n^j)$ be the iterate after adding the $j^{th}$ cut to $\mathcal{G}_{j-1}$, $j = 2n+1, \ldots, m$ and the centering is performed. Then, we have*

*i) $Z_n^{-2} + (I - Z_n)^{-2} \succeq \text{diag}\,(8, \ldots, 8)$, where $Z_n = \text{diag}\,(z_1^j, \ldots, z_n^j)$, $j = 2n+1, \ldots, m$,*

*ii) $\|a_j\|_{j-1}^2 \leq \frac{1}{8}$, $j = 2n+1, \ldots, m$.*

Proof:

i) From $\frac{1}{t^2} + \frac{1}{1-t^2} \geq 8$, $t \in ]0,1[$, so $\frac{1}{(z_i^j)^2} + \frac{1}{(1-z_i^j)^2} \geq 8$, $i = 1, \ldots, n$. Thus, $Z_n^{-2} + (I - Z_n)^{-2} \succeq \text{diag}\,(8, \ldots, 8)$.

ii) From the definitions (6.29) and (6.31) we can write

$$\begin{array}{rcll} \|a_j\|_{j-1}^2 & = & a_j^t \left( \sum_{i=1}^{j-1} \frac{a_i a_i^t}{s_i^2(z^j)} \right)^{-1} a_j & \\ & \leq & a_j^t \left( \sum_{i=1}^{2n} \frac{a_i a_i^t}{s_i^2(z^j)} \right)^{-1} a_j & (\text{from Sherman-Morrison}) \\ & = & a_j^t \left( \sum_{i=1}^{n} \left( \frac{a_i a_i^t}{(z_i^j)^2} + \frac{a_i a_i^t}{(1-z_i^j)^2} \right) \right)^{-1} a_j & \\ & = & a_j^t \left( Z_n^{-2} + (I - Z_n)^{-2} \right)^{-1} a_j & (Z_n = \text{diag}(z_1^j, \ldots, z_n^j)) \\ & \leq & \frac{1}{8} a_j^t a_j = \frac{1}{8} & (\text{ from } \|a_j\| = 1). \end{array}$$

□

The following lemma uses a construction developed by Nesterov [48] which bounds the Hessian and is necessary in order to prove finite convergence.

**Lemma 6.7.8** *Let $m \geq 2n$ and let $s$ be $s = s_m(x)$ with $x \in \mathcal{G}_m$. Define*

$$B^{2n} := 8I, \ B^{m+1} := B^m + \frac{a_{m+1}a_{m+1}^t}{(\nu_1 + \sqrt{n})^2}.$$

*Then,*

$$A_m^t S_m^{-2} A_m \succeq B^m.$$

PROOF:

$$\begin{array}{rcl} A_m^t S_m^{-2} A_m & = & Z_n^{-2} + (I - Z_n)^{-2} + \sum_{i=2n+1}^m \frac{a_i a_i^t}{s_i^2} \\ & \succeq & Z_n^{-2} + (I - Z_n)^{-2} + \frac{1}{(\nu_1+\sqrt{n})^2} \sum_{i=2n+1}^m a_i a_i^t \\ & \succeq & 8I + \frac{1}{(\nu_1+\sqrt{n})^2} \sum_{i=2n+1}^m a_i a_i^t \\ & = & B^m. \end{array}$$

□

By defining $(\omega^m)^2 := a_{m+1}^t (B^m)^{-1} a_{m+1}$ we obtain

$$(\omega^m)^2 \geq a_{m+1}^t (A_m^t S_m^{-2} A_m)^{-1} a_{m+1} = \|a_{m+1}\|_m^2. \tag{6.39}$$

**Lemma 6.7.9**

$$\sum_{j=2n+1}^m (\omega^m)^2 \leq 2n(\nu_1 + \sqrt{n})^2 \ln\left(1 + \frac{m+1-2n}{8n(\nu_1 + \sqrt{n})^2}\right).$$

PROOF: By using the identity

$$\det\,(Q + qq^t) = \det\,(Q)(1 + q^t Q^{-1} q), \text{ for all } Q \succeq 0, q \in I\!R^n,$$

we have

$$\begin{array}{rcl} \det\, B^{m+1} & = & \det\, \left(B^m + \frac{1}{(\nu_1+\sqrt{n})^2} a_{m+1} a_{m+1}^t\right) \\ & = & \det\, B^m \left(1 + \frac{(\omega^m)^2}{(\nu_1+\sqrt{n})^2}\right). \end{array}$$

Hence,

$$\ln\det\, B^{m+1} = \ln\det\, B^m + \ln\left(1 + \frac{(\omega^m)^2}{(\nu_1 + \sqrt{n})^2}\right).$$

From the inequality $\ln(1+t) \geq t/2$, $t \in [0,1]$ we obtain

$$\begin{array}{rcl} \ln\det\, B^{m+1} & \geq & \ln\det\, B^m + (\omega^m)^2 / \left(2(\nu_1 + \sqrt{n})^2\right) \\ & \geq & \ln\det\, B^{2n} + \sum_{j=2n+1}^m (\omega^j)^2 / \left(2(\nu_1 + \sqrt{n})^2\right) \\ & \geq & n \ln 8 + \sum_{j=2n+1}^m (\omega^j)^2 / \left(2(\nu_1 + \sqrt{n})^2\right). \end{array}$$

The geometric inequality [2] implies

$$\frac{1}{n}\ln\det B^{m+1} \le \ln\frac{\text{trace } B^{m+1}}{n} = \ln\left(8+\frac{m+1-2n}{n(\nu_1+\sqrt{n})^2}\right),$$

so we can write

$$\sum_{j=2n+1}^{m}\frac{(\omega^j)^2}{2(\nu_1+\sqrt{n})^2} \le n\ln\left(8+\frac{m+1-2n}{n(\nu_1+\sqrt{n})^2}\right) - n\ln 8$$

or

$$\sum_{j=2n+1}^{m}(\omega^j)^2 \le 2n(\nu_1+\sqrt{n})^2\ln\left(1+\frac{m+1-2n}{8n(\nu_1+\sqrt{n})^2}\right).$$

$\square$

**Remark 5** *As an additional information, remember that given a $n\times n$ matrix $A$ both the determinant and the trace are coefficients of its polynomial characteristic. Moreover, when the matrix $A$ is symmetric then all its eigenvalues are real numbers. The arithmetic-geometric mean inequality is then satisfied by using the eigenvalues of the matrix $A$. Hence, the inequality*

$$\frac{1}{n}\ln \mathit{det}\, A \le \ln\frac{\mathit{trace}\ A}{n}$$

*can be deduced from the invariance of the determinant and the trace of $A$.*

The next proposition proves that the number of cuts which can be generated by the procedure *Center and add constrains* is finite because the value $F_\mu^m(x^m(\mu))$ decreases slower than a linear function of $m$, which is a upper bound of $F_\mu^m(x^m(\mu))$ (see (6.36)).

**Proposition 6.7.10** *Fix $\mu$. A feasible solution is found the first time that $m$ satisfies*

$$-\frac{(M+\bar{M})\sqrt{n}}{(m+1)\mu}+\ln\nu_0-\frac{m+1-2n}{m+1}\alpha \ge \frac{1}{2}\ln\frac{n+2n(\nu_1+\sqrt{n})^2\ln\left(1+\frac{m+1-2n}{8n(\nu_1+\sqrt{n})^2}\right)}{m+1}$$

*with $\alpha = \frac{1}{3}+\ln 4$.*

[2]For any two positive vectors $u, v \in \mathbb{R}^n$,

$$\prod_{j=1}^{n}\left(\frac{u_j}{v_j}\right)^{v_j} \le \left(\frac{\sum_{j=1}^n u_j}{\sum_{j=1}^n v_j}\right)^{\sum_{j=1}^n v_j}.$$

When $v$ is the all-one vector $e$ the geometric inequality reduces to the arithmetic-geometric mean inequality (we refer to [24] for a proof).

PROOF: Consider $x^{2n}(\mu) := (x_1^{2n}(\mu), \dots, x_n^{2n}(\mu))$. From (6.37) we can write

$$F_\mu^{2n}(x^{2n}(\mu)) = \frac{x^{2n}(\mu)^t Q x^{2n}(\mu)/2 + c^t x^{2n}(\mu)}{\mu} - \sum_{j=1}^{n} (\ln x_j^{2n}(\mu) + \ln(1 - x_j^{2n}(\mu))).$$

Using the concavity of $\ln t$ we obtain

$$F_\mu^{2n}(x^{2n}(\mu)) \geq \frac{x^{2n}(\mu)^t Q x^{2n}(\mu)/2 + c^t x^{2n}(\mu)}{\mu} - n \ln \frac{1}{2}.$$

Define $\bar{M} = \min \left\{ \|Qx/2 + c\|^2 : x \in [0,1]^n \right\}$. Hence, by using the Cauchy-Schwartz inequality (as in expression (6.37)) and remembering that $\mathcal{G} \subset \mathcal{G}_m \subset [0,1]^n$ we deduce

$$F_\mu^{2n}(x^{2n}(\mu)) \geq -\frac{\bar{M}\sqrt{n}}{\mu} - n \ln \frac{1}{2}. \tag{6.40}$$

From the Corollary 6.7.4 we can deduce the inequality

$$F_\mu^{2n}(x^{2n}(\mu)) - F_\mu^{m+1}(x^{m+1}(\mu)) \leq (m + 1 - 2n)\alpha + \frac{1}{2} \sum_{j=2n+1}^{m+1} \ln \|a_j\|_{j-1}^2. \tag{6.41}$$

Substituting (6.40) in (6.41) and rearranging terms we obtain

$$F_\mu^{m+1}(x^{m+1}(\mu)) \geq -\frac{\bar{M}\sqrt{n}}{\mu} - n \ln \frac{1}{2} - (m + 1 - 2n)\alpha - \frac{1}{2} \sum_{j=2n+1}^{m+1} \ln \|a_j\|_{j-1}^2.$$

By using (6.36) we have

$$\frac{M\sqrt{n}}{\mu} - (m+1)\ln \nu_0 \geq -\frac{\bar{M}\sqrt{n}}{\mu} - n \ln \frac{1}{2} - (m + 1 - 2n)\alpha - \frac{1}{2} \sum_{j=2n+1}^{m+1} \ln \|a_j\|_{j-1}^2,$$

or equivalently

$$\frac{(M + \bar{M})\sqrt{n}}{\mu} + n \ln \frac{1}{2} - (m+1)\ln \nu_0 \geq -(m + 1 - 2n)\alpha - \frac{1}{2} \sum_{j=2n+1}^{m+1} \ln \|a_j\|_{j-1}^2.$$

Rearranging terms and dividing by $m+1$ we obtain

$$-\frac{(M + \bar{M})\sqrt{n}}{(m+1)\mu} + \ln \nu_0 - \frac{m+1-2n}{m+1}\alpha \leq \frac{1}{2(m+1)} \left( 2n \ln \frac{1}{2} + \sum_{j=2n+1}^{m+1} \ln \|a_j\|_{j-1}^2 \right).$$

From the concavity of $\ln t$ we can write

$$\frac{1}{2(m+1)} \left( 2n \ln \frac{1}{2} + \sum_{j=2n+1}^{m+1} \ln \|a_j\|_{j-1}^2 \right) \leq \frac{1}{2} \ln \frac{n + \sum_{j=2n+1}^{m+1} \|a_j\|_{j-1}^2}{m+1}.$$

From the expression (6.39) we deduce

$$\frac{1}{2}\ln\frac{n+\sum_{j=2n+1}^{m+1}\|a_j\|_{j-1}^2}{m+1} \leq \frac{1}{2}\ln\frac{n+\sum_{j=2n+1}^{m+1}(\omega^j)^2}{m+1}.$$

From the lemma (6.7.9) we have

$$\frac{1}{2}\ln\frac{n+\sum_{j=2n+1}^{m+1}(\omega^j)^2}{m+1} \leq \frac{1}{2}\ln\frac{n+2n(\nu_1+\sqrt{n})^2\left(1+\frac{m+1-2n}{8n(\nu_1+\sqrt{n})^2}\right)}{m+1}.$$

Finally, the inequality

$$-\frac{(M+\bar{M})\sqrt{n}}{(m+1)\mu}+\ln\nu_0-\frac{m+1-2n}{m+1}\alpha \leq \frac{1}{2}\ln\frac{n+2n(\nu_1+\sqrt{n})^2\ln\left(1+\frac{m+1-2n}{8n(\nu_1+\sqrt{n})^2}\right)}{m+1}$$

can be written.

Notice that the left hand side of the above inequality is equivalent to a constant of the order $\ln\nu_0$ and the right hand side decreases to $-\infty$ when $m$ increases. Hence, the value of $m$ increases until the left hand side is greater than the right hand side, the inequality changes, and a suitable feasible solution is found. This process is always finite so the number of cuts generated must also be finite. $\square$

## 6.8 The effect of deleting a constraint

Suppose that the constraint $a_k^t x + s_k = b_k$, $k \in \mathcal{Q}$ is removed from the given problem, while assuming that the remaining constraint matrix still has full rank. Denote $\bar{\mathcal{Q}} := \mathcal{Q}\setminus\{k\}$ and $\bar{\chi}_k = s_k/\|a_k\|_{\bar{\mathcal{Q}}}$. $\bar{A}$ denotes the matrix obtained from $A$ by removing the $k^{\text{th}}$-row. Hence, we have the relationship

$$\bar{A}^t\bar{S}^{-2}\bar{A} = A^tS^{-2}A - \frac{a_k a_k^t}{s_k^2}.$$

Since $\bar{A}^t\bar{S}^{-2}\bar{A}$ is invertible, from Sherman-Morrison's formula we obtain the relationship between the inverses

$$(\bar{A}^t\bar{S}^{-2}\bar{A})^{-1} = (A^tS^{-2}A)^{-1} + \frac{(A^tS^{-2}A)^{-1}a_k a_k^t(A^tS^{-2}A)^{-1}}{s_k^2 - a_k^t(A^tS^{-2}A)^{-1}a_k},$$

which will be used in the next lemma.

**Lemma 6.8.1** *Let $k \in \mathcal{Q}$ be the index of the removed constraint for the given problem, with $s_k = b_k - a_k^t x$ the slack value. We assume that the remaining constraint matrix still has full rank. Then,*

$$\chi_k > 1 \text{ and } \bar{\chi}_k = \sqrt{\chi_k^2 - 1}.$$

PROOF: Multiplying Sherman- Morrison's formula from the left by $a_k^t$ and from the right by $a_k$ we obtain

$$a_k^t(\bar{A}^t\bar{S}^{-2}\bar{A})^{-1}a_k = a_k^t(A^tS^{-2}A)^{-1}a_k + \frac{a_k^t(A^tS^{-2}A)^{-1}a_ka_k^t(A^tS^{-2}A)^{-1}a_k}{s_k^2 - a_k^t(A^tS^{-2}A)^{-1}a_k},$$

which can be expressed in the form

$$\|a_k\|_{\bar{\mathcal{Q}}}^2 = \|a_k\|_{\mathcal{Q}}^2 + \frac{\|a_k\|_{\mathcal{Q}}^4}{s_k^2 - \|a_k\|_{\mathcal{Q}}^2} = \frac{s_k^2\|a_k\|_{\mathcal{Q}}^2}{s_k^2 - \|a_k\|_{\mathcal{Q}}^2},$$

or equivalently

$$\frac{1}{\|a_k\|_{\bar{\mathcal{Q}}}^2} = \frac{s_k^2 - \|a_k\|_{\mathcal{Q}}^2}{s_k^2\|a_k\|_{\mathcal{Q}}^2} = \frac{1}{\|a_k\|_{\mathcal{Q}}^2} - \frac{1}{s_k^2}.$$

Hence, by multiplying both terms of this equality by the value $s_k^2$ we obtain

$$\frac{s_k^2}{\|a_k\|_{\bar{\mathcal{Q}}}^2} = \frac{s_k^2}{\|a_k\|_{\mathcal{Q}}^2} - 1,$$

or equivalently

$$0 \leq \bar{\chi}_k^2 = \chi_k^2 - 1.$$

Suppose that $\chi_k^2 = 1$ and define $P := S^{-1}A(A^tS^{-2}A)^{-1}A^tS^{-1}$, which is an orthogonal projector. It is clear that the following assertions are equivalents

i) $\chi_k^2 = 1$,

ii) $a_k^t(A^tS^{-2}A)^{-1}a_k = s_k^2$,

iii) $e_k^tS^{-1}A(A^tS^{-2}A)^{-1}A^tS^{-1}e_k = e_k^tPe_k = 1$.

From $\|e_k\| = 1$ and $e_k^tPe_k = 1$, by using the properties explained in the example 6.3.5, we can deduce that $e_k = S^{-1}A^t\lambda$ or equivalently $s_ke_k^t = \lambda^tA$. This means that if $Ax = 0$ then $x_k = 0$ i.e. the $k^{\text{th}}$-column of $A$ does not have linear dependence on the other columns of the matrix $A$. Hence, $A$ decreases in rank when the $k^{\text{th}}$-row of $A$ is removed. This is a contradiction because we have assumed that the new matrix $\bar{A}$ has full rank. Therefore, $\chi_k^2 > 1$, which proves the lemma. □

**Lemma 6.8.2** *Suppose that the constraint $a_k^t x + s_k = b_k$, $k \in \mathcal{Q}$ is deleted. The new distance $\bar{\delta}_1(x,\mu)$ is bounded in the form*

$$\bar{\delta}_1(x,\mu) \le \delta_1(x,\mu) + \frac{1+\delta_1(x,\mu)}{\bar{\chi}_k}.$$

Proof: After removing the constraint $a_k^t x + s_k = b_k$, $k \in \mathcal{Q}$ we have the new matrices $\bar{A}$ and $\bar{S}$. Hence, we can write

$$\bar{\delta}_1(x,\mu) = \min_{\bar{y}} \left\{ \left\| \frac{\bar{S}\bar{y}}{\mu} - e \right\| : Qx + c + \bar{A}^t \bar{y} = 0 \right\}. \tag{6.42}$$

The point $y(x,\mu)$ which solves the minimization problem for $\delta_1(x,\mu)$ can be rewritten in the form

$$y(x,\mu) = \begin{pmatrix} \eta \\ \bar{y}(x,\mu) \end{pmatrix}$$

and, in this case, the set constraints of the program (6.42) can be rewritten as

$$Qx + c + \bar{A}^t \bar{y}(x,\mu) + \eta a_k = 0.$$

Thus,

$$\delta_1^2(x,\mu) = \left\| \frac{Sy(x,\mu)}{\mu} - e \right\|^2 = \left\| \frac{\bar{S}\bar{y}(x,\mu)}{\mu} - e \right\|^2 + \left( \frac{s_k \eta}{\mu} - 1 \right)^2$$

and we can deduce

$$\delta_1(x,\mu) \ge \left\| \frac{\bar{S}\bar{y}(x,\mu)}{\mu} - e \right\| \tag{6.43}$$

and

$$\delta_1(x,\mu) \ge \left| \frac{s_k \eta}{\mu} - 1 \right|.$$

From the last inequality we obtain

$$\delta_1(x,\mu) - 1 \le \frac{s_k \eta}{\mu} \le \delta_1(x,\mu) + 1,$$

which is equivalent to

$$-\frac{\delta_1(x,\mu)+1}{s_k} \le \frac{\delta_1(x,\mu)-1}{s_k} \le \frac{\eta}{\mu} \le \frac{\delta_1(x,\mu)+1}{s_k}$$

so we have

$$\left| \frac{\eta}{\mu} \right| \le \frac{\delta_1(x,\mu)+1}{s_k}. \tag{6.44}$$

By defining $\triangle y := \bar{y} - \bar{y}(x,\mu)$ we can rewrite (6.42) in the form

$$\bar{\delta}_1(x,\mu) = \min_{\triangle y} \left\{ \left\| \frac{\bar{S}\bar{y}(x,\mu)}{\mu} - e + \frac{\bar{S}\triangle y}{\mu} \right\| : \bar{A}^t \triangle y = \eta a_k \right\}.$$

Hence, we obtain

$$\begin{aligned}\bar{\delta}_1(x,\mu) &\leq \min_{\triangle y}\left\{\left\|\tfrac{\bar{S}\bar{y}(x,\mu)}{\mu}-e\right\|+\left\|\tfrac{\bar{S}\triangle y}{\mu}\right\| : \bar{A}^t\triangle y=\eta a_k\right\}\\ &\leq \delta_1(x,\mu)+\min_{\triangle y}\left\{\tfrac{1}{\mu}\left\|\bar{S}\triangle y\right\| : \bar{A}^t\triangle y=\eta a_k\right\}\\ &= \delta_1(x,\mu)+\tfrac{\eta}{\mu}\left\|a_k\right\|_{\bar{Q}}\\ &\leq \delta_1(x,\mu)+\tfrac{1+\delta_1(x,\mu)}{s_k}\left\|a_k\right\|_{\bar{Q}},\end{aligned}$$

where the inequalities are deduced from (6.43) and (6.44) respectively, and the equality is deduced by using the expression (6.6), which calculates the distance from a point to a linear variety. Hence, we have that

$$\triangle y=\eta\bar{S}^{-2}\bar{A}^t(\bar{A}\bar{S}^{-2}\bar{A}^t)^{-1}a_k=\eta S_{\bar{Q}}^{-2}A_{\bar{Q}}^t(A_{\bar{Q}}S_{\bar{Q}}^{-2}A_{\bar{Q}}^t)^{-1}a_k$$

minimizes $\|\bar{S}\triangle y\|$ on $\bar{A}^t\triangle y=\eta a_k$. By using $\bar{\chi}_k=s_k/\|a_k\|_{\bar{Q}}$ we can write

$$\bar{\delta}_1(x,\mu)\leq\delta_1(x,\mu)+\frac{1+\delta_1(x,\mu)}{\bar{\chi}_k},$$

which proves the lemma. □

To analyze the complexity in this case we suppose that $s_m=b_m-a_k^tm\geq 4$ in an iterate near the path. Lemma 6.8.2 gives

$$\bar{\delta}_1(x,\mu)\leq\delta_1(x,\mu)+\frac{(1+\delta_1(x,\mu))\|a_k\|_{m-1}}{s_m}=\frac{1}{4}+\frac{1+1/4}{4}\frac{1}{\sqrt{2}}<\frac{1}{2}.$$

From this point we have to recenter, and only a finite number of constraints can be added.

## 6.9 The build-up and down quadratic semi-infinite logarithmic barrier algorithm

The algorithm approximately follows the central path, which leads to an optimal solution to the problem. This is accomplished by repeated reduction of the barrier parameter $\mu$. When an iterate becomes infeasible or near infeasible (Dynamic_Mesh procedure), a new cut is added by using the Center_and_Add_constraints procedure (build-up), and the algorithm attempts to move to a new central point. When a

Table 6.3: Delete constraints procedure for *SIQP*

```
Procedure: Delete_constraints
Input:
$\alpha_d$ is a deleting parameter value;
$\mathcal{Q} = \mathcal{J}_0 \cup \mathcal{I}$ is the index set of the current constraints;
$x$ is the current iterate;
Output:
$\mathcal{Q}$ is the new set of current constraints;
$x$ is the centered for the index set $\mathcal{Q}$;
begin
    for $t \in \mathcal{Q}\backslash\mathcal{J}_0$ do
        if $\exists t \in \mathcal{Q}\backslash\mathcal{J}_0 : s_t \geq \alpha_d$ then
            $\mathcal{Q} := \mathcal{Q}\backslash\{t\}$;
            if $\delta(x,\mu) > \zeta$ then Center_and_Add_constraints;
        end if
    end for
end
```

constraint is nonbinding in an optimal solution it will be removed by using the Delete_constraints procedure (build-down) from our current system. Then, we re-center as necessary with respect to the updated constraint subset and by using the current barrier parameter $\mu$. The main difficulty is to guarantee that $\mu$ can be successfully reduced in such a way that the algorithm terminates finitely. Because of difficulties with numeric stability the slack variables must be checked. The algorithm finishes when the smallest slack value is smaller than a threshold value, which can be set, for instance, at machine precision (or another suitable value). The details of the method are given below.

**Proposition 6.9.1** *Let $\mu_0$ be the initial barrier parameter and let $\epsilon$ be a positive number. The algorithm ends with a gap $s^t y(x,\mu) \leq \epsilon$ after at most*

$$\frac{1}{\theta} \ln \frac{(m+\sqrt{m})\mu_0}{\epsilon}$$

*iterations.*

Proof: The algorithm stop when $\mu_k = (1-\theta)^k \mu_0 \leq \epsilon/(m+\sqrt{m})$. Indeed, taking logarithms

$$k \ln(1-\theta) \leq \ln \frac{\epsilon}{(m+\sqrt{m})\mu_0}$$

Table 6.4: Logarithmic barrier algorithm for *SIQP*

```
The build-up and build-down algorithm
Input:
$\mu = \mu_0$ is the initial barrier parameter value;
$\bar{\mu}$ is the machine precision or another suitable stopping parameter;
$\epsilon > 0$ is the accuracy of the final solution;
$0 < \theta < 1$ is the reduction parameter to the barrier parameter $\mu$;
$0 < \zeta < 1$ is a centering parameter;
$\mathcal{Q}$ is the initial subset of constraints with $\mid \mathcal{Q} \mid = m \geq 2n$;
$x := x_0$ is a given interior point with $y(x, \mu)$ the solution
to the problem (6.21) and such that $\delta(x, \mu) < \zeta$;
Output:
$x$ is an $\epsilon$-minimizer of (6.2);
begin
    while $\mu > \epsilon/(m + \sqrt{m})$ do
        Delete_constraints;
        $\mu := (1 - \theta)\mu$;
        Center_and_Add_constraints;
    end while
end
```

or equivalently

$$-k \ln(1 - \theta) \geq \ln \frac{(m + \sqrt{m})\mu_0}{\epsilon}.$$

From $\theta \leq -\ln(1 - \theta)$, we can write

$$k \geq \frac{1}{\theta} \ln \frac{(m + \sqrt{m})\mu_0}{\epsilon}.$$

From Lemma 6.6.1, if $\delta_1(x, \mu) \leq 1$ then

$$s^t y(x, \mu) \leq \mu_k (m + \sqrt{m}) \leq \frac{\epsilon}{(m + \sqrt{m})}(m + \sqrt{m}) = \epsilon.$$

$\square$

## 6.10 Semi-infinite quadratic test problem and numerical results

Consider

$$\mathcal{G} = \left\{ (x_1, x_2, x_3) : (x_1 - a)^2 + (x_2 - b)^2 + (x_3 - c)^2 \leq r^2 \right\}.$$

The solution of the quadratic problem

$$min \left\{ \|(x_1, x_2, x_3)\|^2 : (x_1, x_2, x_3) \in \mathcal{G} \right\}, \tag{6.45}$$

can be obtained directly (see Table 6.5 ) by using the expression

$$\left(1 - \frac{r}{\sqrt{a^2 + b^2 + c^2}}\right)(a, b, c).$$

On the other hand, by using the polar coordinates $(r, \theta_1, \theta_2)$ with $0 \leq \theta_1 \leq 2\pi$ and $0 \leq \theta_2 \leq \pi$, the tangent hyperplanes to $\partial\mathcal{G}$ can be expressed in the form

$$\sum_{i=1}^{3} a_i(\theta_1, \theta_2) x_i = d(\theta_1, \theta_2),$$

where

$$\begin{aligned} a_1(\theta_1, \theta_2) &= \cos\theta_1 \sin\theta_2, \\ a_2(\theta_1, \theta_2) &= \sin\theta_1 \sin\theta_2, \\ a_3(\theta_1, \theta_2) &= \cos\theta_2, \\ d(\theta_1, \theta_2) &= r + a\cos\theta_1 \sin\theta_2 + b\sin\theta_1 \sin\theta_2 + c\cos\theta_2. \end{aligned}$$

Thus, we have

$$\mathcal{G} = \{(x_1, x_2, x_3) : \sum_{i=1}^{3} a_i(\theta_1, \theta_2) x_i \leq d(\theta_1, \theta_2), (\theta_1, \theta_2) \in [0, 2\pi] \times [0, \pi]\}$$

and the problem (6.45) can be rewritten as the semi-infinite quadratic programming problem

$$\begin{array}{rc} \text{minimize} & \|(x_1, x_2, x_3)\|^2 \\ \text{subject to:} & \sum_{i=1}^{3} a_i(\theta_1, \theta_2) x_i \leq d(\theta_1, \theta_2), \\ & 0 \leq \theta_1 \leq 2\pi, 0 \leq \theta_2 \leq \pi. \end{array} \tag{6.46}$$

This example is used to illustrate the concepts which are dealt with in this chapter, in order to directly obtain the solution for the problem (6.46). Then, we compare it with the solution calculated by using the semi-infinite algorithm. In Table 6.5, it can be seen the characteristics and exact solutions of the problem (6.46). In Table 6.6, it can be seen the solution calculated from the semi-infinite algorithm, where *iter* means the number of total iterations, *fact* indicates the number of matrix factorizations, *cons* are the number of total constraints used, *obj.val.* is the objective value and *time* is the CPU time in seconds. For all instances, the number of subdivisions for both parameter intervals $[0, 2\pi]$ and $[0, \pi]$ is 512, the initial $\mu$ is 1000 and the precision $10e-5$. Moreover, the number of initial constraints, which define the initial bounded polytope, is 6 for instances of dimension 2, and 8 for instances of dimension 3. Notice, that the initial polytope include the constrains that are simple bounds on the variables as it can be seen, for example, in Figure 6.2.

Table 6.5: Characteristics and exact instance solutions of the semi-infinite quadratic test problem

| *Instance* | *a* | *b* | *c* | *radius* | *distance* |
|---|---|---|---|---|---|
| c001d2 | 2.0 | 2.0 | 0.0 | 1.75 | 1.0784 |
| c002d2 | 2.0 | 3.0 | 0.0 | 1.75 | 1.8556 |
| c003d2 | 4.0 | 2.0 | 0.0 | 1.75 | 2.7221 |
| c004d2 | 10.0 | 15.0 | 0.0 | 9.00 | 9.0278 |
| c005d2 | 4.0 | 4.0 | 0.0 | 4.00 | 1.6568 |
| c006d2 | 4.0 | 4.0 | 0.0 | 6.00 | 0.0000 |
| c001d3 | 2.0 | 2.0 | 2.0 | 1.75 | 1.7141 |
| c002d3 | 2.0 | 3.0 | 3.0 | 1.75 | 2.9404 |
| c003d3 | 4.0 | 2.0 | 3.0 | 1.75 | 3.6351 |
| c004d3 | 10.0 | 15.0 | 20.0 | 9.00 | 17.9258 |
| c005d3 | 4.0 | 4.0 | 4.0 | 4.00 | 2.9282 |
| c006d3 | 3.0 | 3.0 | 3.0 | 6.00 | 0.0000 |

## 6.11 Graphical description and numerical results of the instance $c003d2$

In the Figure 6.2 we have a graphical description for computational results of the instance $c003d2$, in which we can see how the added constraints are concentrated around the optimal solution. The numerical results directly obtained, by application of the *SIQP* algorithm to the instance $c003d2$, are included in Appendix B.

Table 6.6: Instance solutions of the semi-infinite quadratic test problem by using semi-infinite algorithm

| *Instance* | *iter* | *fact* | *cons* | *obj.val.* | *time* |
|---|---|---|---|---|---|
| c001d2 | 16 | 56 | 12 | 1.0784 | 14.23 |
| c002d2 | 16 | 50 | 12 | 1.8855 | 12.41 |
| c003d2 | 16 | 62 | 15 | 2.7221 | 15.60 |
| c004d2 | 16 | 57 | 15 | 9.0277 | 15.65 |
| c005d2 | 16 | 55 | 12 | 1.6568 | 17.13 |
| c006d2 | 15 | 29 | 6 | 0.0000 | 5.99 |
| c001d3 | 16 | 70 | 21 | 1.7141 | 232.06 |
| c002d3 | 16 | 66 | 20 | 2.9404 | 215.80 |
| c003d3 | 16 | 77 | 24 | 3.6351 | 268.09 |
| c004d3 | 16 | 74 | 26 | 17.9258 | 239.03 |
| c005d3 | 16 | 69 | 18 | 2.9282 | 262.88 |
| c006d3 | 15 | 26 | 6 | 0.0000 | 4.72 |

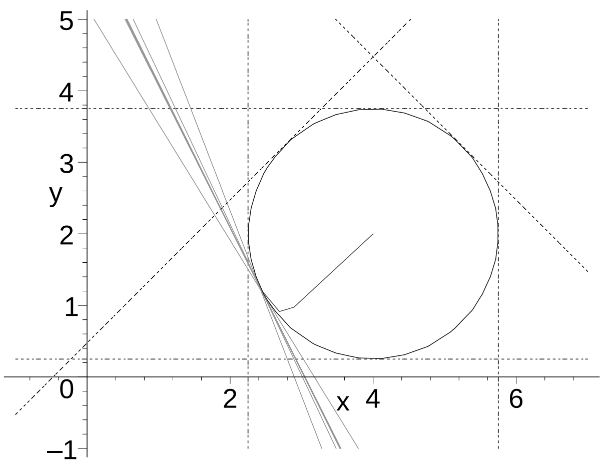

Figure 6.2: Description of the instance *c*003*d*2 by using semi-infinite algorithm. We can see how the added constraints are concentrated around the optimal solution.

## 6.12 Optimal d.c. representation of the power hydrogeneration functions

In Tables 6.7, 6.8, 6.9 and 6.10, $z(x)$ represents the coefficients of the power hydrogeneration function, the pair $(f(x), g(x))$ is a suitable d.c. representation of $z(x)$

$(z(x) = f(x) - g(x))$ and $v(x) = f(x) + g(x)$. In Tables 6.7 and 6.8 we describe the original d.c. representation of the power hydrogeneration function for every reservoir in the hydrogeneration systems. On the other hand, the optimal d.c. representation of the power hydrogeneration functions are described in Tables 6.9 and 6.10. From the optimal d.c. representation of the power hydrogeneration functions we can obtain a more efficient d.c. representation of the whole functions in the problem (2.15), but notice, that they are not the optimal d.c. representation for the whole functions in the problem, which would have required to solve a very difficult semi-infinite programming problem to obtain them.

Table 6.7: Initial d.c. representation of the power hydrogeneration function for the reservoirs $R1$ and $R3$

| $n$ | Basis | $z(x)$ | $f(x)$ | $g(x)$ | $v(x)$ |
|---|---|---|---|---|---|
| 1 | $x^2$ | $-.581395E-04$ | $0.957033E-01$ | $0.957614E-01$ | $0.191465E+00$ |
| 2 | $y^2$ | $0.000000E+00$ | $0.478517E-01$ | $0.478517E-01$ | $0.957033E-01$ |
| 3 | $z^2$ | $0.000000E+00$ | $0.478517E-01$ | $0.478517E-01$ | $0.957033E-01$ |
| 4 | $xy$ | $0.957033E-01$ | $0.957033E-01$ | $0.000000E+00$ | $0.957033E-01$ |
| 5 | $xz$ | $0.957033E-01$ | $0.957033E-01$ | $0.000000E+00$ | $0.957033E-01$ |
| 6 | $yz$ | $0.000000E+00$ | $0.000000E+00$ | $0.000000E+00$ | $0.000000E+00$ |
| | | | | norm | 0.270731E+00 |
| 7 | $x^3$ | $-.380000E-11$ | $0.191214E-03$ | $0.191214E-03$ | $0.382427E-03$ |
| 8 | $y^3$ | $-.540000E-11$ | $0.119509E-03$ | $0.119509E-03$ | $0.239017E-03$ |
| 9 | $z^3$ | $-.540000E-11$ | $0.119509E-03$ | $0.119509E-03$ | $0.239017E-03$ |
| 10 | $x^2y$ | $-.170000E-11$ | $0.143410E-03$ | $0.143410E-03$ | $0.286821E-03$ |
| 11 | $x^2z$ | $-.170000E-11$ | $0.143410E-03$ | $0.143410E-03$ | $0.286821E-03$ |
| 12 | $xy^2$ | $-.143410E-03$ | $0.717051E-04$ | $0.215115E-03$ | $0.286821E-03$ |
| 13 | $xz^2$ | $-.143410E-03$ | $0.717051E-04$ | $0.215115E-03$ | $0.286821E-03$ |
| 14 | $y^2z$ | $-.161000E-11$ | $0.717051E-04$ | $0.717051E-04$ | $0.143410E-03$ |
| 15 | $yz^2$ | $-.161000E-11$ | $0.717051E-04$ | $0.717051E-04$ | $0.143410E-03$ |
| 16 | $xyz$ | $-.143410E-03$ | $0.000000E+00$ | $0.143410E-03$ | $0.143410E-03$ |
| | | | | norm | 0.807015E-03 |
| 17 | $x^4$ | $-.450000E-15$ | $0.303163E-07$ | $0.303163E-07$ | $0.606326E-07$ |
| 18 | $y^4$ | $0.181000E-14$ | $0.909488E-07$ | $0.909488E-07$ | $0.181898E-06$ |
| 19 | $z^4$ | $-.497000E-14$ | $0.909488E-07$ | $0.909488E-07$ | $0.181898E-06$ |
| 20 | $x^3y$ | $0.000000E+00$ | $0.606326E-07$ | $0.606326E-07$ | $0.121265E-06$ |
| 21 | $x^3z$ | $-.181000E-14$ | $0.606325E-07$ | $0.606326E-07$ | $0.121265E-06$ |
| 22 | $xy^3$ | $0.121265E-06$ | $0.181898E-06$ | $0.606326E-07$ | $0.242530E-06$ |
| 23 | $y^3z$ | $0.360000E-14$ | $0.181898E-06$ | $0.181898E-06$ | $0.363795E-06$ |
| 24 | $xz^3$ | $0.121265E-06$ | $0.181898E-06$ | $0.606326E-07$ | $0.242530E-06$ |
| 25 | $yz^3$ | $-.720000E-14$ | $0.181898E-06$ | $0.181898E-06$ | $0.363795E-06$ |
| 26 | $x^2yz$ | $0.000000E+00$ | $0.242530E-06$ | $0.242530E-06$ | $0.485060E-06$ |
| 27 | $xy^2z$ | $0.121265E-06$ | $0.363795E-06$ | $0.242530E-06$ | $0.606325E-06$ |
| 28 | $xyz^2$ | $0.121265E-06$ | $0.363795E-06$ | $0.242530E-06$ | $0.606325E-06$ |
| 29 | $x^2y^2$ | $0.000000E+00$ | $0.151581E-06$ | $0.151581E-06$ | $0.303163E-06$ |
| 30 | $x^2z^2$ | $-.810000E-14$ | $0.151581E-06$ | $0.151581E-06$ | $0.303163E-06$ |
| 31 | $y^2z^2$ | $-.540000E-14$ | $0.272846E-06$ | $0.272846E-06$ | $0.545693E-06$ |
| | | | | norm | 0.139059E-05 |

Table 6.8: Initial d.c. representation of the power hydrogeneration function for the reservoirs $R2$ and $R4$

| $n$ | Basis | $z(x)$ | $f(x)$ | $g(x)$ | $v(x)$ |
|---|---|---|---|---|---|
| 1 | $x^2$ | $0.000000E+00$ | $0.584999E-01$ | $0.584999E-01$ | $0.117000E+00$ |
| 2 | $y^2$ | $0.000000E+00$ | $0.292500E-01$ | $0.292500E-01$ | $0.584999E-01$ |
| 3 | $z^2$ | $0.000000E+00$ | $0.292500E-01$ | $0.292500E-01$ | $0.584999E-01$ |
| 4 | $xy$ | $0.584999E-01$ | $0.584999E-01$ | $0.000000E+00$ | $0.584999E-01$ |
| 5 | $xz$ | $0.584999E-01$ | $0.584999E-01$ | $0.000000E+00$ | $0.584999E-01$ |
| 6 | $yz$ | $0.000000E+00$ | $0.000000E+00$ | $0.000000E+00$ | $0.000000E+00$ |
| | | | | norm | $0.165463E+00$ |
| 7 | $x^3$ | $-.169000E-11$ | $0.861410E-04$ | $0.861410E-04$ | $0.172282E-03$ |
| 8 | $y^3$ | $-.240000E-11$ | $0.538381E-04$ | $0.538381E-04$ | $0.107676E-03$ |
| 9 | $z^3$ | $-.240000E-11$ | $0.538381E-04$ | $0.538381E-04$ | $0.107676E-03$ |
| 10 | $x^2y$ | $-.720000E-12$ | $0.646058E-04$ | $0.646058E-04$ | $0.129212E-03$ |
| 11 | $x^2z$ | $-.720000E-12$ | $0.646058E-04$ | $0.646058E-04$ | $0.129212E-03$ |
| 12 | $xy^2$ | $-.646058E-04$ | $0.323029E-04$ | $0.969087E-04$ | $0.129212E-03$ |
| 13 | $xz^2$ | $-.646058E-04$ | $0.323029E-04$ | $0.969087E-04$ | $0.129212E-03$ |
| 14 | $y^2z$ | $-.720000E-12$ | $0.323029E-04$ | $0.323029E-04$ | $0.646058E-04$ |
| 15 | $yz^2$ | $-.720000E-12$ | $0.323029E-04$ | $0.323029E-04$ | $0.646058E-04$ |
| 16 | $xyz$ | $-.646058E-04$ | $0.000000E+00$ | $0.646058E-04$ | $0.646058E-04$ |
| | | | | norm | $0.363557E-03$ |
| 17 | $x^4$ | $-.290000E-15$ | $0.193487E-07$ | $0.193487E-07$ | $0.386973E-07$ |
| 18 | $y^4$ | $0.115000E-14$ | $0.580460E-07$ | $0.580460E-07$ | $0.116092E-06$ |
| 19 | $z^4$ | $-.317000E-14$ | $0.580460E-07$ | $0.580460E-07$ | $0.116092E-06$ |
| 20 | $x^3y$ | $0.000000E+00$ | $0.386973E-07$ | $0.386973E-07$ | $0.773946E-07$ |
| 21 | $x^3z$ | $-.115000E-14$ | $0.386973E-07$ | $0.386973E-07$ | $0.773946E-07$ |
| 22 | $xy^3$ | $0.773947E-07$ | $0.116092E-06$ | $0.386973E-07$ | $0.154789E-06$ |
| 23 | $y3z$ | $0.230000E-14$ | $0.116092E-06$ | $0.116092E-06$ | $0.232184E-06$ |
| 24 | $xz^3$ | $0.773946E-07$ | $0.116092E-06$ | $0.386973E-07$ | $0.154789E-06$ |
| 25 | $yz^3$ | $-.460000E-14$ | $0.116092E-06$ | $0.116092E-06$ | $0.232184E-06$ |
| 26 | $x^2yz$ | $0.000000E+00$ | $0.154789E-06$ | $0.154789E-06$ | $0.309579E-06$ |
| 27 | $xy^2z$ | $0.773947E-07$ | $0.232184E-06$ | $0.154789E-06$ | $0.386973E-06$ |
| 28 | $xyz^2$ | $0.773946E-07$ | $0.232184E-06$ | $0.154789E-06$ | $0.386973E-06$ |
| 29 | $x^2y^2$ | $0.000000E+00$ | $0.967433E-07$ | $0.967433E-07$ | $0.193487E-06$ |
| 30 | $x^2z^2$ | $-.519000E-14$ | $0.967433E-07$ | $0.967433E-07$ | $0.193487E-06$ |
| 31 | $y^2z^2$ | $-.350000E-14$ | $0.174138E-06$ | $0.174138E-06$ | $0.348276E-06$ |
| | | | | norm | $0.887511E-06$ |

Table 6.9: Optimal d.c. representation of the power hydrogeneration function for the reservoirs $R1$ and $R3$ obtained by using the semi-infinite algorithm

| $n$ | Basis | $z(x)$ | $f(x)$ | $g(x)$ | $v(x)$ |
|---|---|---|---|---|---|
| 1 | $x^2$ | $-.581395E-04$ | $0.324835E-01$ | $0.325416E-01$ | $0.650251E-01$ |
| 2 | $y^2$ | $0.000000E+00$ | $0.336645E-01$ | $0.336645E-01$ | $0.673291E-01$ |
| 3 | $z^2$ | $0.000000E+00$ | $0.208359E-01$ | $0.208359E-01$ | $0.416718E-01$ |
| 4 | $xy$ | $0.957033E-01$ | $0.562928E-01$ | $-.394105E-01$ | $0.168823E-01$ |
| 5 | $xz$ | $0.957033E-01$ | $0.440984E-01$ | $-.516049E-01$ | $-.750651E-02$ |
| 6 | $yz$ | $0.000000E+00$ | $0.218007E-01$ | $0.218007E-01$ | $0.436015E-01$ |
| | | | | norm | $0.112874E+00$ |
| 7 | $x^3$ | $-.253515E-06$ | $0.103109E-04$ | $0.105644E-04$ | $0.208753E-04$ |
| 8 | $y^3$ | $0.000000E+00$ | $0.363911E-04$ | $0.363911E-04$ | $0.727822E-04$ |
| 9 | $z^3$ | $0.000000E+00$ | $0.373155E-04$ | $0.373155E-04$ | $0.746311E-04$ |
| 10 | $x^2y$ | $0.000000E+00$ | $0.516392E-04$ | $0.516392E-04$ | $0.103278E-03$ |
| 11 | $x^2z$ | $0.000000E+00$ | $0.509149E-04$ | $0.509149E-04$ | $0.101830E-03$ |
| 12 | $xy^2$ | $-.143410E-03$ | $-.709266E-04$ | $0.724837E-04$ | $0.155707E-05$ |
| 13 | $xz^2$ | $-.143410E-03$ | $-.726222E-04$ | $0.707881E-04$ | $-.183406E-05$ |
| 14 | $y^2z$ | $0.000000E+00$ | $0.404350E-04$ | $0.404350E-04$ | $0.808701E-04$ |
| 15 | $yz^2$ | $0.000000E+00$ | $0.390559E-04$ | $0.390559E-04$ | $0.781119E-04$ |
| 16 | $xyz$ | $-.143410E-03$ | $-.728244E-04$ | $0.705858E-04$ | $-.223860E-05$ |
| | | | | norm | $0.212110E-03$ |
| 17 | $x^4$ | $0.000000E+00$ | $0.137176E-08$ | $0.137176E-08$ | $0.274351E-08$ |
| 18 | $y^4$ | $0.000000E+00$ | $0.604150E-07$ | $0.604150E-07$ | $0.120830E-06$ |
| 19 | $z^4$ | $0.000000E+00$ | $0.472810E-07$ | $0.472810E-07$ | $0.945619E-07$ |
| 20 | $x^3y$ | $0.000000E+00$ | $0.492811E-08$ | $0.492811E-08$ | $0.985621E-08$ |
| 21 | $x^3z$ | $0.000000E+00$ | $0.466677E-08$ | $0.466677E-08$ | $0.933354E-08$ |
| 22 | $xy^3$ | $0.121265E-06$ | $0.669363E-07$ | $-.543288E-07$ | $0.126076E-07$ |
| 23 | $y^3z$ | $0.000000E+00$ | $0.408544E-07$ | $0.408544E-07$ | $0.817089E-07$ |
| 24 | $xz^3$ | $0.121265E-06$ | $0.540632E-07$ | $-.672019E-07$ | $-.131386E-07$ |
| 25 | $yz^3$ | $0.000000E+00$ | $0.357566E-07$ | $0.357566E-07$ | $0.715132E-07$ |
| 26 | $x^2yz$ | $0.000000E+00$ | $0.465939E-07$ | $0.465939E-07$ | $0.931878E-07$ |
| 27 | $xy^2z$ | $0.121265E-06$ | $0.588472E-07$ | $-.624179E-07$ | $-.357080E-08$ |
| 28 | $xyz^2$ | $0.121265E-06$ | $0.581846E-07$ | $-.630805E-07$ | $-.489580E-08$ |
| 29 | $x^2y^2$ | $0.000000E+00$ | $0.649961E-07$ | $0.649961E-07$ | $0.129992E-06$ |
| 30 | $x^2z^2$ | $0.000000E+00$ | $0.553756E-07$ | $0.553756E-07$ | $0.110751E-06$ |
| 31 | $y^2z^2$ | $0.000000E+00$ | $0.360574E-07$ | $0.360574E-07$ | $0.721147E-07$ |
| | | | | norm | $0.280964E-06$ |

Table 6.10: Optimal d.c. representation of the power hydrogeneration function for the reservoirs $R2$ and $R4$ obtained by using the semi-infinite algorithm

| $n$ | Basis | $z(x)$ | $f(x)$ | $g(x)$ | $v(x)$ |
|---|---|---|---|---|---|
| 1 | $x^2$ | $0.000000E+00$ | $0.224546E-01$ | $0.224546E-01$ | $0.449092E-01$ |
| 2 | $y^2$ | $0.000000E+00$ | $0.123096E-01$ | $0.123096E-01$ | $0.246192E-01$ |
| 3 | $z^2$ | $0.000000E+00$ | $0.130552E-01$ | $0.130552E-01$ | $0.261103E-01$ |
| 4 | $xy$ | $0.584999E-01$ | $0.293609E-01$ | $-.291390E-01$ | $0.221824E-03$ |
| 5 | $xz$ | $0.584999E-01$ | $0.293687E-01$ | $-.291312E-01$ | $0.237497E-03$ |
| 6 | $yz$ | $0.000000E+00$ | $0.130665E-01$ | $0.130665E-01$ | $0.261330E-01$ |
| | | | | norm | $0.631485E-01$ |
| 7 | $x^3$ | $0.000000E+00$ | $0.615022E-05$ | $0.615022E-05$ | $0.123004E-04$ |
| 8 | $y^3$ | $0.000000E+00$ | $0.171830E-04$ | $0.171830E-04$ | $0.343659E-04$ |
| 9 | $z^3$ | $0.000000E+00$ | $0.168556E-04$ | $0.168556E-04$ | $0.337111E-04$ |
| 10 | $x^2y$ | $0.000000E+00$ | $0.228710E-04$ | $0.228710E-04$ | $0.457420E-04$ |
| 11 | $x^2z$ | $0.000000E+00$ | $0.232906E-04$ | $0.232906E-04$ | $0.465811E-04$ |
| 12 | $xy^2$ | $-.646058E-04$ | $-.322440E-04$ | $0.323618E-04$ | $0.117836E-06$ |
| 13 | $xz^2$ | $-.646058E-04$ | $-.326406E-04$ | $0.319652E-04$ | $-.675366E-06$ |
| 14 | $y^2z$ | $0.000000E+00$ | $0.171015E-04$ | $0.171015E-04$ | $0.342030E-04$ |
| 15 | $yz^2$ | $0.000000E+00$ | $0.169891E-04$ | $0.169891E-04$ | $0.339782E-04$ |
| 16 | $xyz$ | $-.646058E-04$ | $-.323943E-04$ | $0.322115E-04$ | $-.182802E-06$ |
| | | | | norm | $0.951617E-04$ |
| 17 | $x^4$ | $0.000000E+00$ | $0.229213E-09$ | $0.229213E-09$ | $0.458427E-09$ |
| 18 | $y^4$ | $0.000000E+00$ | $0.523446E-07$ | $0.523446E-07$ | $0.104689E-06$ |
| 19 | $z^4$ | $0.000000E+00$ | $0.424769E-07$ | $0.424769E-07$ | $0.849539E-07$ |
| 20 | $x^3y$ | $0.000000E+00$ | $0.999162E-09$ | $0.999162E-09$ | $0.199832E-08$ |
| 21 | $x^3z$ | $0.000000E+00$ | $0.515736E-08$ | $0.515736E-08$ | $0.103147E-07$ |
| 22 | $xy^3$ | $0.773946E-07$ | $0.446634E-07$ | $-.327313E-07$ | $0.119321E-07$ |
| 23 | $y^3z$ | $0.000000E+00$ | $0.730828E-07$ | $0.730828E-07$ | $0.146166E-06$ |
| 24 | $xz^3$ | $0.773946E-07$ | $0.367809E-07$ | $-.406137E-07$ | $-.383276E-08$ |
| 25 | $yz^3$ | $0.000000E+00$ | $0.147884E-07$ | $0.147884E-07$ | $0.295768E-07$ |
| 26 | $x^2yz$ | $0.000000E+00$ | $0.291790E-07$ | $0.291790E-07$ | $0.583579E-07$ |
| 27 | $xy^2z$ | $0.773946E-07$ | $0.415851E-07$ | $-.358095E-07$ | $0.577556E-08$ |
| 28 | $xyz^2$ | $0.773946E-07$ | $0.378847E-07$ | $-.395099E-07$ | $-.162520E-08$ |
| 29 | $x^2y^2$ | $0.000000E+00$ | $0.829985E-08$ | $0.829985E-08$ | $0.165997E-07$ |
| 30 | $x^2z^2$ | $0.000000E+00$ | $0.193607E-06$ | $0.193607E-06$ | $0.387214E-06$ |
| 31 | $y^2z^2$ | $0.000000E+00$ | $0.174353E-06$ | $0.174353E-06$ | $0.348705E-06$ |
| | | | | norm | $0.562078E-06$ |

# Chapter 7

# Numerical results and conclusions

## 7.1 Introduction

The modified algorithm discussed in Chapter 4 has been implemented and run for four different test problems, which are used to illustrate all the different possible situations where deterministic global optimization algorithms can be applied. We remark on good results and also point out some case instances where performance is poor.

All instances of the test problems are of the same size, but when we use different d.c. representations of the objective function, their behavior is very different as regards the number of iterations and subdivisions used by the modified algorithm. This is a relevant result of the Thesis. For this reason, when there is a possible choice of d.c. representation of the objective function, great care must be taken in its selection. On the other hand, in all cases the modified algorithm detects a good feasible point in a neighborhood of a global optimizer very quickly, even if it is not necessarily the final solution, and this can be considered a positive result.

By using the method developed in Chapter 3 we can convert the Generation Problem into an equivalent reverse convex programming problem in such a way that the modified algorithm can be applied. From the optimal d.c. representation of the power hydrogeneration functions we can obtain a more efficient d.c. representation of the whole functions in the Generation Problem, but note that they are not the optimal d.c. representation for the whole functions in the Generation Problem. $MINOS$ 5.5 has been used to solve all problem instances and also to check all gradients of the functions in the instances of the Generation Problem. Numerical results and $CPU$

requirements of instances solved of the Generation Problem are given and explained in this Chapter.

All required *CPU* times reported refer to the compute used NETSERVER LC2000 U3 of Hewlett Packard with 4 GB of *RAM* and 2 *CPU* of 1000 MHz Pentium III. Moreover, to compare different speeds of solution, instances of the Generation Problem have been solved with a computer Compaq AlphaServer HPC320: 8 nodes ES40 (4 EV68, 833 MHz, 64 KB/8 MB), 20 GB of main memory, 1.128 GB on disk and top speed of 53.31 Gflop/s, connected with Memory Channel II de 100 MB/s. By using this *high performance* computer the *CPU* time can be reduced to one third.

## 7.2 Global optimization test problems and numerical results

The algorithm discussed in Chapter 4 has been implemented and run for four different instances, which are described in this section. In Tables 7.2, 7.3, 7.4 and 7.5 *Case* is the number of the case instance, $\epsilon$ is the precision, $D.c.(K)$ is a nonnegative real number $k$ such that $f(x) + k\left(\sum_{j=1}^{n} x_j^2\right)$ is a convex function, in which the function $f(x)$ represents the nonconvex objective function of the current instance, *Iter* indicates the number of iterations required, *Msdv* indicates the maximum number of subdivisions that have been simultaneously active, *Tsdv* indicates the total number of subdivisions performed by the modified algorithm, *Mdepht* indicates the maximum depth reached for the subdivision procedure, *Obj. Val* indicates the optimum obtained by the modified algorithm and *Time* is the *CPU* time in seconds.

### 7.2.1 The class of test problems $HPTnXmY$

The following class of test problems can be found in [33] and they turn out to be rich enough to produce typical numerical results. We seek an $\epsilon$-solution in the sense mentioned in Chapter 4 of

$$\begin{array}{rl} \text{minimize} & -\sum_{i=1}^{m} 1/\left(\|x-a^i\|^2 + c_i\right) \\ \text{subject to} & x \in I\!R^n, 0 \le x_j \le 10, j = 1, \dots, n \end{array} \tag{7.1}$$

where $a^i \in \{x \in \mathbb{R}^n : 0 \leq x_j \leq 10, 1 \leq j \leq n\}$ and $c_i > 0$. The initial simplex $S_0$ for this class of test problems is

$$S_0 = \left\{ x \in \mathbb{R}^n_+ : \sum_{j=1}^{n} x_j \leq 10n \right\}.$$

By using the convex function $k\left(\sum_{j=1}^n x_j^2\right)$ with $k > 0$, we can obtain a d.c. representation of the objective function in (7.1) as follows. Consider $f(x) = \sum_{i=1}^m f_i(x)$, with $f_i(x) := 1/\left(\|x - a^i\|^2 + c_i\right)$ and $x \in \mathbb{R}^n$. Hence, we can write

$$f(x) = \left( f(x) + k\sum_{j=1}^{n} x_j^2 \right) - \left( k\sum_{j=1}^{n} x_j^2 \right), \tag{7.2}$$

with $k$ a real number such that $f(x) + k\sum_{j=1}^n x_j^2$ is a convex function. The different instances of the test problem 7.1 are denoted by $HPTnXmY$ where $X$ represents the dimension and $Y$ means the number of local optimal solutions of the instance. In Table 7.2 we consider the instance $HPTn2m10$ with the parameters $c_i$, $a^i_j$, $i = 1, \ldots, 10$, $j = 1, 2$ from the Table 7.1. Figure 7.1 shows a plot of this instance. Table 7.2 displays relevant results on the computational effort required to minimize the function $f(x)$ by means of our algorithm with different d.c. representations of the objective function, which are defined by the values $k = 0.5$, $k = 5$ and $k = 50$, and the precisions $\epsilon_i = 10^{-i}$, $i = 1, 2, 3$.

Table 7.1: Parameters for the test problem $HPTnXmY$

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $c_i$ | 0.70 | 0.73 | 0.76 | 0.79 | 0.82 | 0.85 | 0.88 | 0.91 | 0.94 | 0.97 |
| $a^i_1$ | 4.0 | 2.5 | 7.5 | 8.0 | 2.0 | 2.0 | 4.5 | 8.0 | 9.5 | 5.0 |

### 7.2.2 The class of test problems $TnXrY$

Let $x \in \mathbb{R}^n$ be $x = (x_1, \ldots, x_n)$. A reduced version of the test problem

$$\begin{array}{rl} \text{minimize} & f(x) = \Pi_{i=1}^n (x_i^2 + c_i x_i) \\ \text{subject to} & Ax \leq b, \\ & d_i \leq x_i \leq e_i, i = 1, \ldots, n, \end{array} \tag{7.3}$$

where $A \in \mathbb{R}^{m*n}$ and $b \in \mathbb{R}^m$, can be found in [74]. The names of the different instances of the test problem 7.3 are denoted by $TnXrY$, where $X$ is the dimension

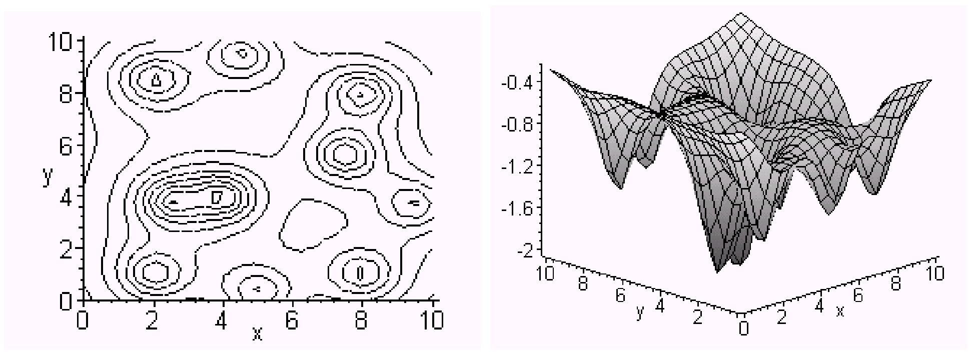

Figure 7.1: Plot of the objective function in (7.1) for the instance $HPTn2m10$

Table 7.2: Computational results for the instance $HPTn2m10$

| *Case* | $\epsilon$ | *D.c.(K)* | *Iter* | *Msdv* | *Tsdv* | *Mdepht* | *Obj.val.* | *Time* |
|---|---|---|---|---|---|---|---|---|
| 1 | $\epsilon_1$ | 0.5 | 312 | 55 | 645 | 23 | −2.1423 | 2.39 |
| 2 | $\epsilon_2$ | 0.5 | 354 | 55 | 732 | 28 | −2.1423 | 3.17 |
| 3 | $\epsilon_3$ | 0.5 | 403 | 55 | 832 | 31 | −2.1423 | 3.85 |
| 4 | $\epsilon_1$ | 5 | 371 | 55 | 762 | 26 | −2.1424 | 3.34 |
| 5 | $\epsilon_2$ | 5 | 408 | 55 | 840 | 31 | −2.1424 | 3.93 |
| 6 | $\epsilon_3$ | 5 | 449 | 55 | 925 | 33 | −2.1424 | 4.50 |
| 7 | $\epsilon_1$ | 50 | 335 | 55 | 694 | 26 | −2.1411 | 2.82 |
| 8 | $\epsilon_2$ | 50 | 342 | 55 | 708 | 26 | −2.1411 | 2.91 |
| 9 | $\epsilon_3$ | 50 | 451 | 55 | 936 | 37 | −2.1411 | 4.84 |

and $Y$ means the number of linear constraints of the instance. Figure 7.2 shows a plot of the objective function of the instance $Tn2r0$, where $d_i = -1$, $e_i = 1$, $i = 1, \dots, n$ and the coefficients of the objective function in (7.3) are $c_1 = 0.09$ and $c_2 = 0.1$. For numerical tests, we have chosen the instance $Tn2r4$ with the same parameters $c_1 = 0.09$ and $c_2 = 0.1$ for the objective function in (7.3) and the feasible domain defined as follows.

$$\{(x_1, x_2) : Ax \leq b, -2 \leq x_i \leq 1, i = 1, 2\},$$

where

$$A = \begin{bmatrix} 1 & -1 \\ -1 & 1 \\ 1 & 1 \\ -1 & -1 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 2.5 \\ 1 \\ 3.5 \end{bmatrix}.$$

As before, by using the convex function $k\left(\sum_{j=1}^{n} x_j^2\right)$ many different d.c. representations of the objective function can be obtained in the form

$$f(x) = \left(f(x) + k\sum_{j=1}^{n} x_j^2\right) - \left(k\sum_{j=1}^{n} x_j^2\right).$$

Table 7.3 displays the numerical results for the instance $Tn2r4$ with different d.c. representations of the objective function, which are defined by the values $k = 7.5$, $k = 8$ and $k = 8.5$, and the precisions $\epsilon_i = 10^{-i}$, $i = 1, 2, 3$.

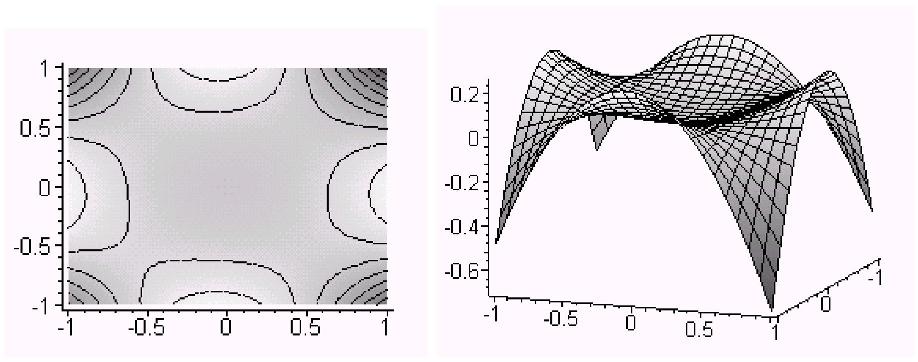

Figure 7.2: Plot of the objective function for the instance $Tn2r4$

Table 7.3: Computational results for the instance $Tn2r4$

| $Case$ | $eps$ | $D.c.(K)$ | $Iter$ | $Msdv$ | $Tsdv$ | $Mdepht$ | $Obj.val$ | $Time$ |
|---|---|---|---|---|---|---|---|---|
| 1 | $\epsilon_1$ | 7.5 | 706 | 247 | 1547 | 12 | $-8.3882$ | 9 |
| 2 | $\epsilon_2$ | 7.5 | 5304 | 1999 | 11395 | 16 | $-8.3882$ | 650 |
| 3 | $\epsilon_3$ | 7.5 | 20000 | 14020 | 43560 | 19 | $-8.3882$ | 16099 |
| 4 | $\epsilon_1$ | 8.0 | 867 | 312 | 1901 | 11 | $-8.3882$ | 16 |
| 5 | $\epsilon_2$ | 8.0 | 6305 | 2424 | 13518 | 15 | $-8.3882$ | 970 |
| 6 | $\epsilon_3$ | 8.0 | 20000 | 16384 | 43504 | 18 | $-8.3882$ | 15855 |
| 7 | $\epsilon_1$ | 8.5 | 989 | 355 | 2175 | 12 | $-8.3882$ | 22 |
| 8 | $\epsilon_2$ | 8.5 | 7332 | 2736 | 15708 | 15 | $-8.3882$ | 1371 |
| 9 | $\epsilon_3$ | 8.5 | 20000 | 17653 | 43343 | 18 | $-8.3882$ | 15711 |

### 7.2.3 The instance $HPBr1$

The instance

$$\begin{array}{rl} \text{minimize} & xy = \frac{1}{4}(x+y)^2 - \frac{1}{4}(x-y)^2 \\ \text{subject to:} & x - y \leq 5.7, \\ & -2 \leq x \leq 3, \\ & -3 \leq y \leq 4, \end{array} \tag{7.4}$$

which will be denoted by $HPBr1$, is a nonconvex programming problem with the optimal solution on $x - y = 5.7$. The objective function of $HPBr1$ is an homogeneous polynomial of degree two with two variables (in this case it is a hyperbole). From Section 5.6 we know that the d.c. representation of $xy$ in (7.4) is the optimal. Alternative d.c. representations of $xy$, which are not optimal, are

(1) $xy = \frac{1}{2}(x+y)^2 - \frac{1}{2}(x^2+y^2)$, and

(2) $xy = \frac{1}{2}(x^2+y^2) - \frac{1}{2}(x-y)^2$.

Table 7.4 displays the results of minimizing the function $xy$ with these different d.c. representations of the objective function $xy$, and the precisions $\epsilon_i = 10^{-i}$, $i = 1, 2, 3$.

Table 7.4: Computational results for the instance $HPBr1$

| *Case* | $\epsilon$ | *D.c.* | *Iter* | *Msdv* | *Tsdv* | *Mdepht* | *Obj.val.* | *Time* |
|---|---|---|---|---|---|---|---|---|
| 1 | $\epsilon_1$ | *opt* | 15 | 3 | 32 | 7 | −8.1184 | 0.02 |
| 2 | $\epsilon_2$ | *opt* | 26 | 4 | 57 | 11 | −8.1220 | 0.06 |
| 3 | $\epsilon_3$ | *opt* | 32 | 6 | 69 | 11 | −8.1220 | 0.07 |
| 4 | $\epsilon_1$ | (1) | 24 | 6 | 51 | 8 | −8.1159 | 0.05 |
| 5 | $\epsilon_2$ | (1) | 34 | 6 | 73 | 11 | −8.1188 | 0.07 |
| 6 | $\epsilon_3$ | (1) | 54 | 8 | 113 | 13 | −8.1225 | 0.12 |
| 7 | $\epsilon_1$ | (2) | 51 | 13 | 108 | 9 | −8.1048 | 0.14 |
| 8 | $\epsilon_2$ | (2) | 104 | 22 | 226 | 12 | −8.1223 | 0.39 |
| 9 | $\epsilon_3$ | (2) | 163 | 32 | 354 | 14 | −8.1224 | 0.77 |

### 7.2.4 The instance $COSr0$

The instance

$$\begin{aligned} \text{minimize} \quad & f(x,y) := 0.03(x^2+y^2) - cos(x) * cos(y) \\ \text{subject to:} \quad & -6 \le x \le 4, \\ & -5 \le y \le 2, \end{aligned} \tag{7.5}$$

which will be denoted by $COSr0$, is a multiextremal programming problem with minimizer $(0,0)$ and minimum $-1$, as we can see in Figure 1.1 (Chapter 1). The function $k(x^2+y^2)$, $k > 0$ allows us to obtain many different d.c. representations of the objective function $f(x,y)$ in (7.5), as follows.

$$f(x,y) = (f(x,y) + k(x^2+y^2)) - k(x^2+y^2).$$

Table 7.5 displays the results of minimizing the function $f(x,y)$ with different d.c. representations of the objective function, which are defined by the values $k = 0.5$, $k = 1$ and $k = 1.5$, and the precisions $\epsilon_i = 10^{-i}$, $i = 1,2,3$.

Table 7.5: Computational results for the instance $COSr0$

| *Case* | $\epsilon$ | *D.c.(K)* | *Iter* | *Msdv* | *Tsdv* | *Mdepht* | *Obj.val.* | *Time* |
|---|---|---|---|---|---|---|---|---|
| 1 | $\epsilon_1$ | 0.5 | 532 | 180 | 1144 | 12 | $-0.9920$ | 7 |
| 2 | $\epsilon_2$ | 0.5 | 1163 | 312 | 2478 | 15 | $-0.9999$ | 29 |
| 3 | $\epsilon_3$ | 0.5 | 1948 | 344 | 4157 | 18 | $-0.9999$ | 86 |
| 4 | $\epsilon_1$ | 1.0 | 1335 | 481 | 2905 | 13 | $-0.9981$ | 41 |
| 5 | $\epsilon_2$ | 1.0 | 3579 | 1040 | 7678 | 16 | $-0.9998$ | 362 |
| 6 | $\epsilon_3$ | 1.0 | 6230 | 1330 | 13369 | 19 | $-0.9999$ | 1297 |
| 7 | $\epsilon_1$ | 1.5 | 2184 | 804 | 4747 | 13 | $-0.9964$ | 122 |
| 8 | $\epsilon_2$ | 1.5 | 6858 | 2120 | 14702 | 17 | $-0.9997$ | 1692 |
| 9 | $\epsilon_3$ | 1.5 | 12675 | 2745 | 27289 | 20 | $-0.9999$ | 6535 |

## 7.3 Characteristics of the generation systems and numerical results

The characteristics of the reservoirs in the generation systems are in Table 2.2 and the characteristics of the generation systems can be found in Table 2.1. Table 7.6 displays the numerical results and $CPU$ requirements of solved instances of the

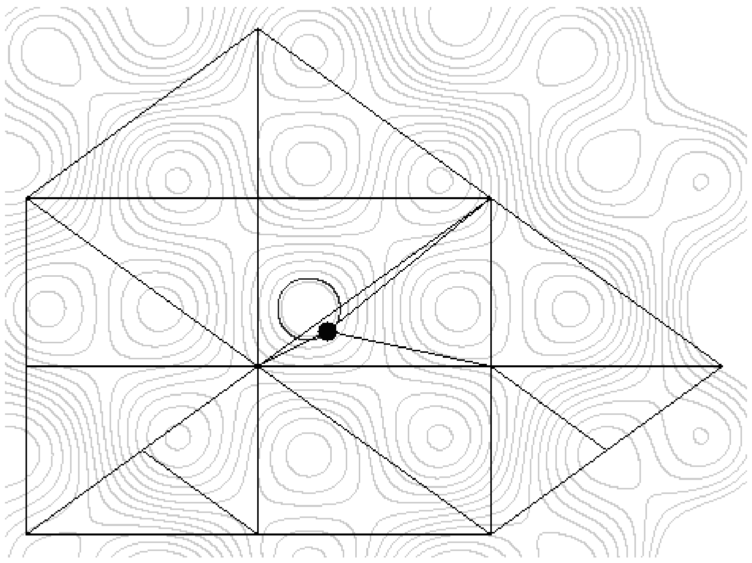

Figure 7.3: Incumbent and subdivisions at the iteration 18 for the case number 2 in Table 7.5

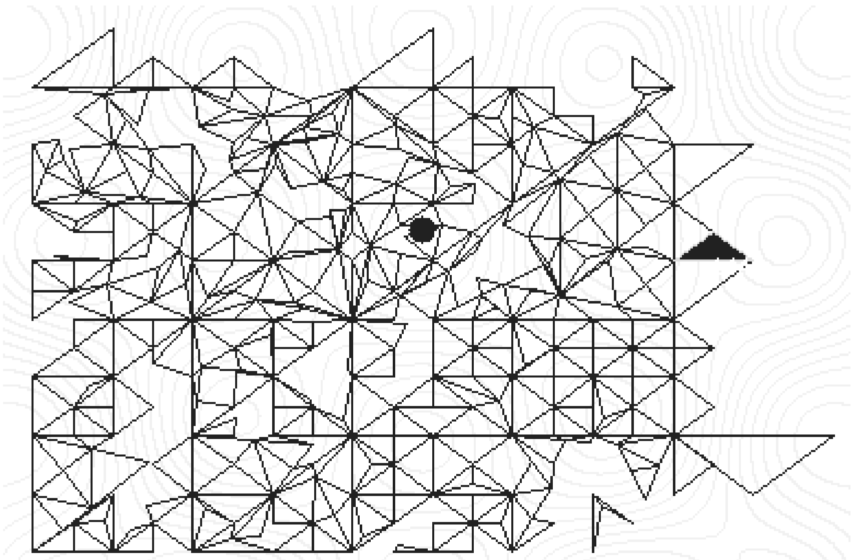

Figure 7.4: Incumbent and subdivisions at the iteration 677 for the case number 2 in Table 7.5. Notice that the incumbent, which has been found at this iteration, is already the global $\epsilon$-solution.

Generation Problem. The names of the different instances in Table 7.6 are denoted by **Cgp**$n$**e**$m$**i**, when $k^i$, in (2.10), is a constant or **Vgp**$n$**e**$m$**i**, when $k^i$ depends

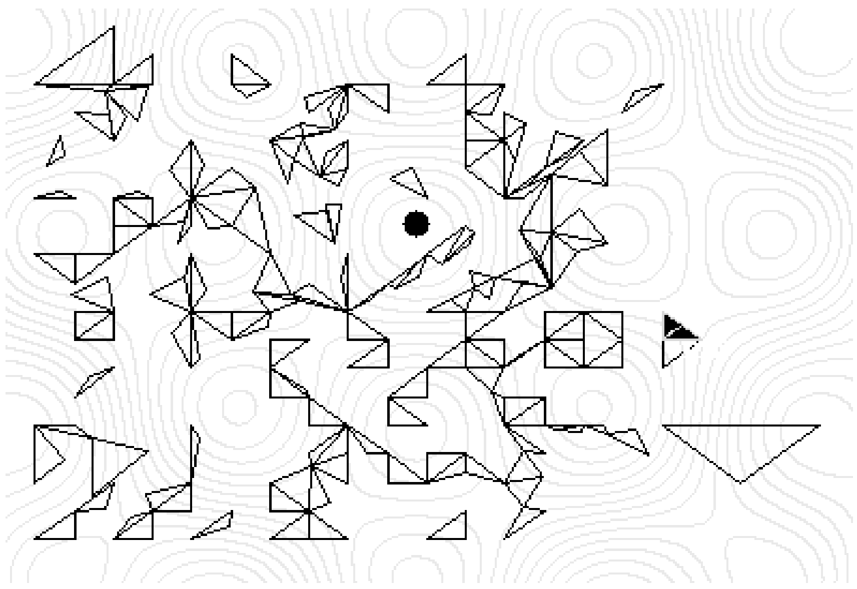

Figure 7.5: Incumbent and subdivisions at the iteration 976 for the case number 2 in Table 7.5. Notice that the optimal solution $(0, 0)$ is an isolated point because it doesn't belong to any active subdivision. Moreover, no active subdivision contains a feasible point better than the incumbent. Hence, we can deduce that the algorithm, from the current iteration, cannot further improve the incumbent.

on water discharges. Both **Cgp**$n$e$m$**i** and **Vgp**$n$e$m$**i** are related to the generation system **c**$n$e$m$**i**. In column *D.c.* of the Table 7.6 we write *Opt* when the instance is solved using the optimal d.c. representation of the power hydrogeneration functions, or $Ini$ when the original d.c. representation is used. Moreover, as before, *Iter* indicates the number of iterations required, *Msdv* indicates the maximum number of subdivisions that have been simultaneously active, *Tsdv* indicates the total number of subdivisions performed by the modified algorithm, *MINOS* indicates the optimal value of the solution obtained by $MINOS$, *Obj.Val* indicates the optimum obtained by the modified algorithm and *Time* is the $CPU$ time in seconds. The maximum number of iterations allowed in the algorithm is 10000 and the bound for $Mdepth$ has been established at 20. In all problem instances the precision used is $\epsilon = 0.01$.

Table 7.6: Results and CPU requirements of the instances of the Generation Problem solved

| *Num* | *Instance* | *D.c.* | *Iter* | *Msdv* | *Tsdv* | *MINOS* | *Obj.Val* | *Time* |
|---|---|---|---|---|---|---|---|---|
| 1 | $Cgp2e02i$ | Ini | 245 | 81 | 524 | 2806.26 | 2806.26 | 3 |
| 2 | | Opt | 100 | 28 | 217 | | 2806.26 | 1 |
| 3 | $Cgp2e03i$ | Ini | 4697 | 237 | 8067 | 2759.39 | 2759.39 | 570 |
| 4 | | Opt | 4315 | 320 | 7363 | | 2759.39 | 464 |
| 5 | $Cgp4e03i$ | Ini | 10000 | 2064 | 14226 | 3304.17 | 3304.23 | 4466 |
| 6 | | Opt | 10000 | 1180 | 12787 | | 3304.23 | 3046 |
| 7 | $Cgp4e04i$ | Ini | 10000 | 2340 | 12861 | 3205.41 | 3205.60 | 6933 |
| 8 | | Opt | 10000 | 2345 | 12803 | | 3205.71 | 5029 |
| 9 | $Cgp2e08i$ | Ini | 10000 | 9737 | 19873 | 2524.82 | 2525.57 | 14126 |
| 10 | | Opt | 10000 | 8703 | 18979 | | 2525.65 | 10188 |
| 11 | $Vgp2e02i$ | Ini | 10000 | 2829 | 11127 | 2778.10 | 2778.10 | 1408 |
| 12 | | Opt | 10000 | 2092 | 10109 | | 2778.10 | 1363 |
| 13 | $Vgp2e03i$ | Ini | 10000 | 1898 | 16205 | 2723.88 | 2724.26 | 2105 |
| 14 | | Opt | 7243 | 902 | 12060 | | 2724.75 | 1222 |
| 15 | $Vgp4e03i$ | Ini | 10000 | 8142 | 19673 | 3277.84 | 3278.17 | 6169 |
| 16 | | Opt | 10000 | 4329 | 14651 | | 3278.23 | 5056 |
| 17 | $Vgp4e04i$ | Ini | 10000 | 3463 | 13264 | 3152.37 | 3152.40 | 5214 |
| 18 | | Opt | 10000 | 3796 | 13597 | | 3152.85 | 7311 |

## 7.4 Conclusions

From a computational point of view we must point out the following.

1. On observing the numerical results, we can deduce that the required computational time is related to both the number of iterations and the performed subdivisions, which depend on the desired precision, the size of the instance and the d.c. representation of the functions. This last result is new and has never before been described. In the case of polynomial functions special attention must be paid to this last result, because of the concept of Least Deviation Decomposition, which can be applied to improve the d.c. representation of a polynomial from a computational view-point. The better the d.c. representation is, the less iterations and subdivisions are needed to find an $\epsilon$-solution, as can be seen in Tables 7.4 and 7.6. When d.c. functions which are not polynomials are used, a similar result is observed by using the convex function $k\left(\sum_{j=1}^{n} x_j^2\right)$. The higher the constant $k$ is, the more iterations are needed, as

we can see in Tables 7.2, 7.3 and 7.5. Nevertheless, we must point out exceptions to this rule for the case instances 7, 8 and 9 in Table 7.2. Actually, for d.c. functions in general, no concept of optimal d.c. representation has been defined which explains the above-mentioned exceptions.

2. By using the modified algorithm, a good feasible point is often detected very quickly and most of the computational effort is devoted to verifying its optimality. Although the optima obtained in various instances of every problem remains within the required precision, many more iterations are required to create small nested subdivisions in a neighborhood of an optimal solution if high precision is needed.

3. During the execution of the algorithm we can arrive at an iteration, in which the optimal solution of the problem is an isolated point because it doesn't belong to any active subdivision. Moreover, no active subdivision contains a feasible point better than the incumbent, which is already an $\epsilon$-optimal solution. Hence, we can deduce that the algorithm, from the current iteration, can't improve the incumbent. All this can be seen in Figures 7.3, 7.4 and 7.5.

4. Although the algorithm is unable to improve the incumbent, the number of subdivisions can keep on increasing, and this situation gets worse depending on the size of the instance. This fact also explains the limitation represented by the size of the instances with regards to the computational time for deterministic algorithms that use outer approximation and subdivision procedures (mainly, when the instance has no special structure). It should be noted that cuts are always *conjunctive* in the outer approximation procedure, i.e., the polyhedron resulting from the cuts is the intersection of all the cuts performed. Hence, the number of constraints increases at each iteration as well as the complexity of the sublinear programs that are to be solved.

5. Other devices could have been used, such as the combination of global optimization tools with standard nonlinear local optimization, but we have preferred to study the behavior of the global optimization tools working on their own in the modified algorithm. Sometimes, to facilitate the procedure, subdivisions can be deleted when they have reached sufficient depth, i.e., they are sufficiently small. Other strategies must be taken into account to solve instances of larger dimensions efficiently.

The method developed in Chapter 3 has show itself to be very useful for obtaining a d.c. representation of a polynomial function. By using this method we can convert the Generation Problem into an equivalent reverse convex programming problem in such a way that the adapted algorithm can be applied. As can be seen in table 7.6, the instances with a constant coefficient of efficiency and unit conversion (Cgp*nem*i) seem to work well, and we can find good values for the optimal. On the other hand, instances with a variable coefficient of efficiency (Vgp*nem*i) have worse optimal values for the objective functions, but all solutions are very near to the solution found by $MINOS$. Of course, this is not an ideal situation but it is not as bad as we might suppose.

The interior point algorithm developed for semi-infinite quadratic programming problems with linear constraints, together with the concept of Least Deviation Decomposition, allows us to obtain a better d.c. representation of each polynomial function of the Generation Problem, in such a way that the computational efficiency of the adapted algorithm is improved. From a computational standpoint and on observing Table 7.6, the efficiency of using the optimal d.c. representation of the power hydrogeneration functions is obvious. Observe in Table 7.6 that for instances with *Opt* in column *D.c.*, i.e., those with optimized d.c. representation of the hydrogeneration functions, there is an overall decrease of the computation time with respect to the computation time of case instances with $Ini$ in column *D.c.*, i.e., those with the initial d.c. representation of the power hydrogeneration functions. It is obvious that for purposes of speed, global optimization depends on the use of high performance computers and parallel computation. Nevertheless, I am sure that there exist a lot of available mathematical results (such as the concept of Least Deviation Decomposition) which could be used in order to obtain more efficient implementations for problems, both with and without any specific structure.

## 7.5 Publications originated by this Thesis

- Ferrer, A., *Representation of a polynomial function as a difference of convex polynomials, with an application.* Lectures Notes in Economics and Mathematical Systems vol. 502, pgs. $189 - -207$. Springer-Verlag, Berlin. 2001

- Ferrer, A., *Applying global optimization to a problem in short-term hydrothermal scheduling.* Accepted for publication in the Proceedings of the 7th International Symposium on Generalized Convexity and Monotonicity held in Hanoi, Vietnam from the 27th to the 31st of August 2002. The edited book will be published by Kluwer Academic Publishers.

## 7.6 Topics for further research

Further research will be geared towards both the mathematical theory of global optimization and software development. The following topics are mid-term projects.

- The study of the behavior of the different optimal d.c. representations of a polynomial by using the different norms within the normed space of the polynomials.
- The extension of the concept of optimal d.c. representation from the polynomials functions to d.c. functions.
- The search for deterministic strategies to obtain more efficient implementation for problems of larger dimensions, either with or without any specific structure.
- The use of high performance computers and parallel computation to speed up global optimization.
- The development of a web of public domain with test problems and a software library for the performance of global optimization analysis.

# Appendix A

In this appendix a procedure is described using the *MAPLE* Symbolic Calculator to search for bases and to obtain d.c. representations of each homogeneous component of a polynomial. The inputs are: number of variables $n$, degree of homogeneous polynomial $m$, a base of $m^{\text{th}}$ powers and the homogeneous polynomial $f$ for which we want to obtain a d.c. representation. The procedure gives us several outputs. The outputs $f_1$ and $f_2$ are the d.c. components of the homogeneous polynomial $f$. The output $f = f_1 + f_2$ is to verify that the final solution is correct. If a set of $m^{\text{th}}$ powers is not a base then the procedure stops and it shows an error message of error. Thus, we must search for alternative base of $m^{\text{th}}$ powers. We show how to obtain a d.c. representation of the homogeneous polynomial

$$x_1(x_3 - x_2)^2 - x_1^3 + 5x_1x_2x_3,$$

using the base (3.38), as an example of application of this procedure.

```
> restart:
```

```
with(linalg):
```

---

Input variables (n), degree of homogeneous polynomial (m):

---

```
n:=3:
m:=3:
```

---

```
d:=binomial(n+m-1, m):
u:=[seq(0,k=1..d)]:
xx:=[seq(x[k],k=1..n)]:
aa:=[seq(1,k=1..n)]:
xxx:=(evade(aa &* xx))^m:
H:=[op(expand(xxx))]:
```

```
M:=[seq(H[k]/coeffs(H[k]),k=1..d)]:
```

$$M := [x_1{}^3,\, x_1{}^2\, x_2,\, x_1{}^2\, x_3,\, x_1\, x_2{}^2,\, x_1\, x_2\, x_3,\, x_1\, x_3{}^2,\, x_2{}^3,\, x_2{}^2\, x_3,\, x_2\, x_3{}^2,\, x_3{}^3]$$

```
a := array(1..d):
```

---

Input base of $m^{\text{th}}$ powers:

---

```
a[1]:=xx[1]^3:
a[2]:=xx[2]^3:
a[3]:=xx[3]^3:
a[4]:=(xx[1]+2*xx[2])^3:
a[5]:=(xx[1]+2*xx[3])^3:
a[6]:=(xx[2]+2*xx[3])^3:
a[7]:=(2*xx[1]+xx[2])^3:
a[8]:=(2*xx[1]+xx[3])^3:
a[9]:=(2*xx[2]+xx[3])^3:
a[10]:=(xx[1]+xx[2]+xx[3])^3:
```

---

Input polynomial from which we want to obtain a d.c. representation

---

```
yyy:=xx[1]*(xx[3]-xx[2])^2-xx[1]^3+5*xx[1]*xx[2]*xx[3];
```

---

```
expand(yyy):
HH:=[op(expand(yyy))]:
nn:=nops(expand(yyy)):
```

$$yyy := x_1\,(x_3 - x_2)^2 - x_1{}^3 + 5\, x_1\, x_2\, x_3$$

$$x_1\, x_3{}^2 + 3\, x_1\, x_2\, x_3 + x_1\, x_2{}^2 - x_1{}^3$$

```
for i from 1 by 1 to nn
do
for k from 1 by 1 to d
do
if (member(M[k], {HH[i]/coeffs(HH[i])})) then
u[k]:=coeffs(HH[i]):
fi:
od:
```

```
od:
print(u);
```

$$[-1, 0, 0, 1, 3, 1, 0, 0, 0, 0]$$

```
v:= array(1..d,1..d):for j from 1 by 1 to d
do
for i from 1 by 1 to d
do
v[j,i]:=0
od;
od;
for j from 1 by 1 to d
do
if (j<n+1) then
L:=[expand(a[j])]:
nn:=1
else
L:=[op(expand(a[j]))]:
nn:=nops(expand(a[j])):
fi:
for i from 1 by 1 to nn
do
for k from 1 by 1 to d
do
if (member(M[k], {L[i]/coeffs(L[i])})) then
v[k,j]:=coeffs(L[i]):
fi:
od:
od:
od:
C:=evalm(v^(-1)):
ff:=evalm(C &* u):
ff1 := array(1..d):
ff2 := array(1..d):
for i from 1 by 1 to d
do
if (ff[i]>0) then
ff1[i]:=ff[i]:
```

```
ff2[i]:=0:
else
ff2[i]:=-ff[i]:
ff1[i]:= 0:
fi;
od:
f1:=evalm(a &* ff1);
f2:=evalm(a &* ff2);
f:=expand(f1-f2);
```

$$f1 := \frac{2}{3}{x_1}^3 + \frac{1}{6}{x_2}^3 + \frac{1}{6}{x_3}^3 + \frac{1}{36}(x_1 + 2\,x_2)^3 + \frac{1}{36}(x_1 + 2\,x_3)^3 + \frac{1}{2}(x_1 + x_2 + x_3)^3$$

$$f2 := \frac{1}{12}(x_2 + 2\,x_3)^3 + \frac{5}{36}(2\,x_1 + x_2)^3 + \frac{5}{36}(2\,x_1 + x_3)^3 + \frac{1}{12}(2\,x_2 + x_3)^3$$

$$f := 3\,x_1\,x_2\,x_3 + x_1\,{x_2}^2 + x_1\,{x_3}^2 - {x_1}^3$$

# Appendix B

In this appendix we show the file results obtained by the application of the semi-infinite quadratic algorithm to the instance $c004d2$, where *ncons* is the number of added constrains, *iter*1 indicates the number of iterations of the algorithm, *iter*2 is the number of inner iterations of the algorithm to close the current point $x$ to the central path until the *distance* ($\delta(x, \mu)$) is small enough. The procedure continues until the barrier parameter $\mu$ verifies $\mu \leq \epsilon/(ncons + \sqrt{ncons})$, with $\epsilon$ the required precision. This is accomplished by repeated reduction of $\mu$. The algorithm performances for each major iterations (*iter*1) a series of minor iterations (*iter*2) in order to get closer to the central path and center the current feasible point. For example, if we observe the sixth major iteration in the file results, four minor iterations have been performed to obtain a point close and close to the central path, i.e., the distance value is less than the stipulated bound (0.25). Moreover, a new constraint is added (*ncons*= 10), although, as can be seen in the first major iteration, this is not always the case. Also, it can be observed that the values of the parameters $\mu$ and *eps* (precision) are reduced at each major iteration. The procedure ends when *eps* becomes smaller then $\mu$.

```
------------------------------------------------------------------------
SEMI-INFINITE QUADRATIC ALGORITHM
------------------------------------------------------------------------
n          2
ncons0     6
eps        1.000000000000000E-005
toler      1.000000000000000E-005
mu         1000.00000000000
reduc_par  0.750000000000000
cent_par   0.250000000000000
add_par    1.000000000000000E-009
```

```
alfa0: 0.0 beta0: 6.283184
ksdv:  512
initial point:0.1000000E+020.1500000E+02

---------------------------------------------------------------------------
|ncons|iter1|iter2|     mu      |  distance   |  obj.val.   |     eps     |
---------------------------------------------------------------------------
|    6|    1|    1|0.2500000E+03|0.2447255E+00|0.1363662E+02|             |
---------------------------------------------------------------------------
|    6|    1|    1|0.2500000E+03|0.2447255E+00|0.1363662E+02|0.2112372E+04|
---------------------------------------------------------------------------
|    6|    2|    1|0.6250000E+02|0.2171031E+00|0.1030750E+02|             |
---------------------------------------------------------------------------
|    6|    2|    1|0.6250000E+02|0.2171031E+00|0.1030750E+02|0.5280931E+03|
---------------------------------------------------------------------------
|    7|    3|    1|0.1562500E+02|0.8338529E+00|0.1030750E+02|             |
|    7|    3|    2|0.1562500E+02|0.2904390E+00|0.9188550E+01|             |
|    7|    3|    3|0.1562500E+02|0.2443715E+00|0.9713749E+01|             |
---------------------------------------------------------------------------
|    7|    3|    3|0.1562500E+02|0.2443715E+00|0.9713749E+01|0.1507149E+03|
---------------------------------------------------------------------------
|    8|    4|    1|0.3906250E+01|0.1674584E+01|0.9713749E+01|             |
|    8|    4|    2|0.3906250E+01|0.3608236E+00|0.9303694E+01|             |
|    8|    4|    3|0.3906250E+01|0.2328531E-01|0.9198383E+01|             |
---------------------------------------------------------------------------
|    8|    4|    3|0.3906250E+01|0.2328531E-01|0.9198383E+01|0.4229854E+02|
---------------------------------------------------------------------------
|    9|    5|    1|0.9765625E+00|0.1076663E+01|0.9198383E+01|             |
|    9|    5|    2|0.9765625E+00|0.2271900E+00|0.9109379E+01|             |
---------------------------------------------------------------------------
|    9|    5|    2|0.9765625E+00|0.2271900E+00|0.9109379E+01|0.1171875E+02|
---------------------------------------------------------------------------
|    9|    6|    1|0.2441406E+00|0.1556006E+01|0.9071785E+01|             |
|    9|    6|    2|0.2441406E+00|0.5326418E+00|0.9049502E+01|             |
|    9|    6|    3|0.2441406E+00|0.3482103E+00|0.9040308E+01|             |
|    9|    6|    4|0.2441406E+00|0.1779073E+00|0.9044921E+01|             |
---------------------------------------------------------------------------
|    9|    6|    4|0.2441406E+00|0.1779073E+00|0.9044921E+01|0.2929688E+01|
---------------------------------------------------------------------------
```

```
|   10|    7|    1|0.6103516E-01|0.2215039E+01|0.9044921E+01|             |
|   10|    7|    2|0.6103516E-01|0.5846586E+00|0.9034330E+01|             |
|   10|    7|    3|0.6103516E-01|0.1282503E+00|0.9032929E+01|             |
-------------------------------------------------------------------------
|   10|    7|    3|0.6103516E-01|0.1282503E+00|0.9032929E+01|0.8033617E+00|
-------------------------------------------------------------------------
|   10|    8|    1|0.1525879E-01|0.1533907E+01|0.9029887E+01|             |
|   10|    8|    2|0.1525879E-01|0.2326075E+00|0.9028341E+01|             |
-------------------------------------------------------------------------
|   10|    8|    2|0.1525879E-01|0.2326075E+00|0.9028341E+01|0.2008404E+00|
-------------------------------------------------------------------------
|   11|    9|    1|0.3814697E-02|0.1101192E+01|0.9028341E+01|             |
|   11|    9|    2|0.3814697E-02|0.4377216E-01|0.9028044E+01|             |
-------------------------------------------------------------------------
|   11|    9|    2|0.3814697E-02|0.4377216E-01|0.9028044E+01|0.5461359E-01|
-------------------------------------------------------------------------
|   11|   10|    1|0.9536743E-03|0.1595118E+01|0.9027896E+01|             |
|   11|   10|    2|0.9536743E-03|0.3036814E+00|0.9027821E+01|             |
|   11|   10|    3|0.9536743E-03|0.2829643E-01|0.9027802E+01|             |
-------------------------------------------------------------------------
|   11|   10|    3|0.9536743E-03|0.2829643E-01|0.9027802E+01|0.1365340E-01|
-------------------------------------------------------------------------
|   11|   11|    1|0.2384186E-03|0.1073689E+01|0.9027775E+01|             |
|   12|   11|    2|0.2384186E-03|0.6726161E+00|0.9027775E+01|             |
|   12|   11|    3|0.2384186E-03|0.4269529E+00|0.9027781E+01|             |
|   12|   11|    4|0.2384186E-03|0.1444421E+00|0.9027780E+01|             |
-------------------------------------------------------------------------
|   12|   11|    4|0.2384186E-03|0.1444421E+00|0.9027780E+01|0.3686929E-02|
-------------------------------------------------------------------------
|   12|   12|    1|0.5960464E-04|0.1687946E+01|0.9027769E+01|             |
|   12|   12|    2|0.5960464E-04|0.4530882E+00|0.9027763E+01|             |
|   12|   12|    3|0.5960464E-04|0.9430605E-01|0.9027760E+01|             |
-------------------------------------------------------------------------
|   12|   12|    3|0.5960464E-04|0.9430605E-01|0.9027760E+01|0.9217323E-03|
-------------------------------------------------------------------------
|   12|   13|    1|0.1490116E-04|0.1326261E+01|0.9027758E+01|             |
|   13|   13|    2|0.1490116E-04|0.9634092E+00|0.9027758E+01|             |
|   13|   13|    3|0.1490116E-04|0.1769063E+00|0.9027758E+01|             |
-------------------------------------------------------------------------
```

```
|   13|   13|    3|0.1490116E-04|0.1769063E+00|0.9027758E+01|0.2474420E-03|
---------------------------------------------------------------------------
|   13|   14|    1|0.3725290E-05|0.1537173E+01|0.9027757E+01|             |
|   13|   14|    2|0.3725290E-05|0.3129436E+00|0.9027757E+01|             |
|   13|   14|    3|0.3725290E-05|0.1378576E+00|0.9027757E+01|             |
---------------------------------------------------------------------------
|   13|   14|    3|0.3725290E-05|0.1378576E+00|0.9027757E+01|0.6186050E-04|
---------------------------------------------------------------------------
|   13|   15|    1|0.9313226E-06|0.1584843E+01|0.9027756E+01|             |
|   13|   15|    2|0.9313226E-06|0.2455574E+00|0.9027756E+01|             |
---------------------------------------------------------------------------
|   13|   15|    2|0.9313226E-06|0.2455574E+00|0.9027756E+01|0.1546512E-04|
---------------------------------------------------------------------------
|   14|   16|    1|0.2328306E-06|0.1566053E+01|0.9027756E+01|             |
|   15|   16|    2|0.2328306E-06|0.3535572E+00|0.9027756E+01|             |
---------------------------------------------------------------------------
|   15|   16|    3|0.2328306E-06|0.3535572E+00|0.9027756E+01|0.4394209E-05|
---------------------------------------------------------------------------

Final results
--------------
total fact  :=   57
mu          :=   2.328306436538696E-007
obj.val.    :=   9.02775639915243
optimal solution:(0.5007750E+01, 0.7511513E+01)
CPU time    :=   15.65
--------------
```

# Bibliography

[1] Alexandrov, A. D. On surfaces which may be represented by a difference of convex functions (in Russian). *Izvestia Akademii Nauk Kazakhskoj SSSR, Seria Fiziko-Matematicheskikh*, 3:3–20, 1949.

[2] Andramonov, M. Yu, Rubinov, A. M. and Glover, B. M. Cutting angle method in global optimization. *Applied Mathematics Letters*, 12:95–100, 1999.

[3] Androulakis I. P., Maranas C. D. and Floudas C. A. $\alpha$BB: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7(4):337–363, 1995.

[4] Avriel, M. *Nonlinear programming analysis and methods.* Englewood Cliffs, NS Printice-Hall, Inc., 1976.

[5] Bagirov, A. M. and Rubinov, A. M. Global minimization of increasing positively homogeneous functions over the unit simplex. *Annals of Operations Research*, 98:171–187, 2000.

[6] Bougeard, M. *Contribution à la théorie de Morse en dimension finie.* PhD thesis, Université de Paris IX, Paris, 1978.

[7] Chambadal, L. and Ovaert, J. L. *Algèbre Linéaire et Algèbre Tensorielle.* Dunod Université. Dunod, Paris, 1968.

[8] Clarke, F. H. *Nonsmooth analysis and optimization*, volume 5 of *Classics in Applied Mathematics.* John Wiley & Sons, Inc., New York, second edition, 1990.

[9] Cox, D., Little, J. and O'Shea, D. *Ideals, varieties and algorithms. An introduction to computational algebraic geometry and commutative algebra.* Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1992.

[10] CPLEX, CPLEX Optimization Inc., http:/www.cplex.com. *CPLEX Callable Library.*

[11] den Hertog, D. *Interior point approach to linear, quadratic and convex programming.* Mathematics and its Applications. Kluwer Academic Publishers, 2nd edition, 1997.

[12] Dijkstra, E. W. A note on two problems in connection with graphs. *Numeriskche Mathematik*, 1:269–271, 1959.

[13] Dixon, L. C. W. Global optima without convexity. Technical report, Numerical Optimization Center, Hatfield Polytechnic, Hatfield, England, 1978.

[14] Ellaia, R. *Contribution a l'analyse et l'optimisation de difference de functions convexes.* PhD thesis, Université Paul Sabatier de Toulouse (Sciences), 1984.

[15] Feo, T. A. and Resende, M. G. C. Greedy randomised adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

[16] Ferrer, A. Representation of a polynomial function as a difference of convex polynomials, with an application. *Lectures Notes in Economics and Mathematical Systems*, 502:189–207, 2001.

[17] Floudas, C. A. and Pardalos, P. M. *A collection of test problems for constrained global optimization algorithms*, volume 268 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin-Heidelberg, 1990.

[18] Floudas, C. A. and Pardalos, P. M., editors. *Recent advances in global optimization.* Princeton Series in Computer Science. Princeton University Press, 1991.

[19] Ford, L. R. Jr. and Fulkerson, D. R. *Flows in networks.* Princeton University Press, 1962.

[20] Gablonsky, J. M. and Kelley, C. T. A locally-biased form of the DIRECT algorithm. *Journal of Global Optimization*, 21:27–37, 2001.

[21] Gill, P. E., Murray, W. and Wright, M. H. *Practical optimization.* Academic Press, London, 1989.

[22] Glover, F. and Laguna, M. *Tabu search.* Kluwer Academic Publishers, 1997.

[23] Gustafson, S. A. On numerical analysis in semi-infinite proramming. *Lecture Notes in Control and Information Sciences*, 15:51–65, 1979.

[24] Hardy, G. H., Littlewood, J. E. and Pólya, G. *Inequalities.* Cambridge University Press, Cambridge, 1934.

[25] Hartman, P. On functions representable as a difference of convex functions. *Pacific Journal of Mathematics*, 9:707–713, 1959.

[26] Heredia, F. J. and Nabona, N. Optimum short-term hydrothermal scheduling with spinning reverse through network flows. *IEEE Trans. on Power Systems*, 10(3):1642–1651, 1995.

[27] Hiriart-Urruty, J. B. Generalized differentiability, duality and optimization for problems dealing with differences of convex functions. *Lectures Notes in Economics and Mathematical Systems*, 256:37–38, 1985.

[28] Hiriart-Urruty, J. B. and Lemarechal, C. *Convex analysis and minimization algorithms*, volume 305 and 306 of *Grundleheren der Mathematischen Wissenschaften*. Springer Verlag, Berlin, 1993.

[29] Holland, J. H. *Adaptation in natural and artificial systems.* University of Michigan Press, 1975.

[30] Holmberg, K. and Tuy, H. A production-transportation problem with stochastic demand and concave production costs. *Mathematical Programming*, 85:157–179, 1999.

[31] Hong, S. R. and Sahinidis, N. V. A branch and bound approach to global optimization. *Journal of Global Optimization*, 8:107–138, 1996.

[32] Horst, R. and Tuy, H. *Global optimization. Deterministic approaches.* Springer-Verlag, Heidelberg, first edition, 1990.

[33] Horst, R., Pardalos, P. M. and Thoai, Ng. V. *Introduction to gobal optimization.* Kluwer Academic Publishers, Dordrecht, first edition, 1995.

[34] Horst, R., Phong, T. Q., Thoai , Ng. V. and de Vries, J. On solving a d.c. programming problem by a sequence of linear programs. *Annals of Operations Research*, 25:1–18, 1990.

[35] Huyer, W. and Neumaier, A. Global optimization by multilevel coordinate search. *Journal of Global Optimization*, 14:331–355, 1999.

[36] Jeyakumar, V. and Glover, B. M. Characterizing global optimality for dc optimization problems under convex inequality constraints. *Journal of Global Optimization*, 8:171–187, 1996.

[37] Kaliski, J., Haglin, D., Roos, C. and Terlaky, T. Logarithmic barrier decomposition methods for semi-infinite programming. *Int.Trans.Oper.Res.*, 4(4):285–303, 1997.

[38] Karmarkar, N. K. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[39] Khachiyan, L. G. A polynomial time algorithm in linear programming (English translation). *Soviet Mathematics Doklady*, 20:191–194, 1979.

[40] Khachiyan, L. G. A polynomial time algorithm in linear programming (in Russian). *Doklady Akademii Nauk SSSR*, 244:1093–1096, 1979.

[41] Kirkpatrick, S. ,Gellat, C. D. Jr. and Vecchi, M. P. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[42] Konno, H., Thach, P. T. and Tuy, H. *Optimization on low rank nonconvex structures.* Kluwer Academic Publishers, New York, 1997.

[43] Kruskal, J. B. On the shorted spanning subtree of a graph and the travelling salesman problem. *Proceedings of the American Mathematical Society*, 7:48–50, 1956.

[44] Landis, E. M. On functions representable as the difference of two convex functions. *Dokl. Akad. Nauk SSSR*, 80:9–11, 1951.

[45] Lang, S. *Algebra.* Addison-Wesley, 1965.

[46] Luc, D. T., Martínez-Legaz, J. E. and Seeger, A. Least deviation decomposition with respect to a pair of convex sets. *Journal of Convex Analysis*, 6(1):115–140, 1998.

[47] MINOS, Systems Optimization Laboratory, Stanford University, http://www.sbsi-sol-optimize.com. *Software package for solving large-scale optimization problems.*

[48] Nesterov, Y. Cutting plane algorithms from analytic centers: efficiency estimates. *Mathematical Programming*, 69:149–176, 1995.

[49] Nesterov, Y. and Nemirovsky, A. *Interior-point polynomial methods in convex programming*, volume 13 of *Studies in Applied Mathematics.* SIAM, Philadelphia, 1994.

[50] Penot, J. P. and Bougeard, M. L. Approximation and decomposition properties of some classes of locally d.c. functions. *Mathematical Programming*, 41:195–227, 1988.

[51] Pey-Chun Chen, Hansen, P., Jaumard, B. and Tuy, H. Solution of the multi-source weber and conditional weber problems by d.c. programming. *Operations Research*, 46(4):548–562, 1998.

[52] Pintér, J. D. *Global optimization in action*, volume 6 of *Nonconvex Optimization and its Applications*. Kluwer Academic Publishers, Dordrecht / Boston / London, 1996.

[53] Piyavskii, S. A. An algorithm for finding the absolute extremum of a function. *USSR Computational Mathematics and Mathematical Phisics*, 12:57–67, 1972.

[54] Prim, R. C. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36:1389–1401, 1957.

[55] Reemtsen, R. Some other approximation methods for semi-infinite optimization problems. *Journal of Computational and Applied Mathematics*, 53:87–108, 1994.

[56] Rockafellar, R. T. *Convex Analysis*. Princeton University Press, Princeton, 1970.

[57] Roos, C., Terlaky, T. and Vial, J.-Ph. *Theory and algorithms for linear optimization*. Series in Discrete Mathematics and Optimization. John Wiley & Sons, Inc., Chichester, 1998.

[58] Rosen, J. B. The gradient projection method for nonlinear programming. Part I. Linear constraints. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, 8(1):181–217, Mar 1960.

[59] Rosen, J. B. The gradient projection method for nonlinear programming. Part II. Non-linear constraints. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, 9:514–553, 1961.

[60] Rosen, J. B. and Pardalos, P. M. . Global optimization of large-scale constrained concave quadratic problems by separable programming. *Mathematical Programming*, 34:163–174, 1986.

[61] Rubinov, A. M. and Andramonov, M. Yu. Lipschitz programming via increasing convex-along-rays functions. *Optimization Methods and Software*, 10:763–781, 1999.

[62] Ryoo, H. S. and Sahinidis, N. V. A branch-and-reduce approach to global optimization. *Journal of Global Optimization*, 8(2):107–139, 1996.

[63] Sahinidis, N. V. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8(2):201–205, 1996.

[64] SDPSOL, Wu, P. and Boyd, S., http://www.stanford.edu/ boyd/SDPSOL.html. *Solver for semidefinite programs with matrix structure.*

[65] Shectman, J. P. and Sahinidis, N. V. A finite algorithm for global minimization of separable concave programs. *Journal of Global Optimization*, 12:1–36, 1998.

[66] Shor, N. Z. Convergence rate of the gradient descent method with dilatation of the space. *Cybernetics*, 6(2):102–108, 1970.

[67] Shor, N. Z. Utilization of the operation of space dilatation in the minimization of convex functions. *Cybernetics*, 6(1):7–15, 1970.

[68] Strekalovsky, A. S. *On global optimality conditions for d.c. programming problems.* Optimization and Optimal Control. Irkutsk State University, 1997.

[69] Strekalovsky, A. S. and Tsevendorj, I. Testing the $\mathcal{R}$-strategy for a reverse convex problem. *Journal of Global Optimization*, 13:61–74, 1998.

[70] Thach, P. T. and Tuy, H. Global optimization under Lipschitzian constraints. *Japan Journal of Mathematics*, 4:205–217, 1987.

[71] Thach, P. T. and Tuy, H. The relief indicator method for constrained global optimization. *Naval Research Logistics*, 37:473–497, 1990.

[72] Tuan, H. D. Can linear programs be used to test global optimization algorithms? *Computing*, 59:91–93, 1997.

[73] Tuy, H. Effect of the subdivision strategy on convergence and efficiency of some global optimization algorithms. *Journal of Global Optimization*, 1(1):23–26, 1991.

[74] Tuy, H. *Convex analysis and global optimization.* Kluwer Academic Publishers, Dordrecht, first edition, 1998.

[75] Vandenberghe, L. and Boyd, S. Positive-definite programming. *Mathematical Programming: State of the Art*, 1994.

[76] Vandenberghe, L. and Boyd, S. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

[77] Vial, J.-P. Strong and weak convexity of sets and functions. *Math. Oper.Res.*, 8:231–259, 1983.

[78] Zhi-Quan, L., Roos, C. and Terlaky, T. Complexity analysis of logarithmic barrier decomposition methods for semi-infinite linear programming. *Applied Numerical Mathematics*, 29:379–394, 1999.