People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

Mohamed Khider University, Biskra

Faculty of Exact Sciences and Natural and Life Sciences

DEPARTEMENT OF MATHEMATICS

Thesis presented with a view to obtaining the Diploma :

MASTER of Mathematics

Option: Analysis

By

Menadi Ahmed

Title :

Interior Point Methods for Linear Optimization Problem

Membres du Comité d'Examen :

Pr.	Chemcham Madani	UMKB	President
Pr.	Amrane Houas	UMKB	Advisor
Pr.	Bellagoun Abdelghani	UMKB	Examinator

June 2024

Dedicace

I dedicate this humble work to:

The souls of my parents.

My dear family,

especially my wife and children.

Acknowledgements

First, I would like to thank my advisor, **Pr. Amrane Houas**, for supervising this work.

I also extend my gratitude to the members of the examination committee, **Pr. Chemcham Madani and Pr. Bellagoun Abdelghani**, for their efforts and valuable guidance.

I would also like to thank all the professors in the Department of Mathematics at Mohamed Khider University of Biskra.

I also to thank, Ms. Loubna Zernadji.

Contents

List of Figures

List of Tables

Introduction

Linear programming (LP) is considered one of the most significant advancements in economic theory after World War II, rapidly developing through the combined efforts of mathematicians, business leaders, military commanders, statisticians, and economists. It is often cited as one of the greatest successes of operations research.

In essence, a linear program involves optimizing (maximizing or minimizing) a linear function of several variables, subject to linear constraints. When Klee and Minty (1972) discovered an example showing that the simplex method could require an exponential number of iterations, the search for a polynomial algorithm for linear programming was launched. Khachiyan provided an answer in 1979 by showing that the ellipsoid method, developed in the early 1970s for nonlinear programming, had an overall complexity of when applied to linear programming.

In 1984, Karmarkar presented a polynomial algorithm with an efficient complexity in practice, based on an interior-point method. Interest in these methods, primarily developed since the 1960s by Dikin (1967) and Fiacco and McCormik (1968), has seen a revival for nonlinear problems and opened up a new domain for linear problems. Interior-point methods have proven effective for large-scale optimization problems due to their polynomial nature. Den Herty (1994) classified interior-point methods into three categories: affine methods, potential reduction methods, and central path methods.

Researchers have continued to refine these methods by developing new kernel func-

tions, such as those introduced by Peng et al. in 2001 and Bai et al. in 2004, which have improved the algorithmic complexity of interior-point methods. More recently, Bouafia et al. proposed a generalized kernel function in 2016, theoretically and numerically demonstrating its superior efficiency for large-step methods based on trigonometric kernel functions.

We present our work in 3 chapters:

Chapter 1: provides a foundation in convex analysis, crucial for understanding the optimization landscape, including concepts such as convex sets and functions. It also introduces basic optimization problem types and optimality conditions.

Chapter 2: it covers the general theory of linear optimization, duality, and the simplex method before delving into the specifics of IPMs, including the primal, dual, and primal-dual methods.

Chapter 3: introduces the Central Trajectory Method via Kernel Function, a novel approach that enhances traditional IPMs by leveraging kernel functions. It discusses the primal-dual method based on kernel functions, the qualifications of kernel functions, the Newton direction, and a generic algorithm for implementation.

Chapter 1

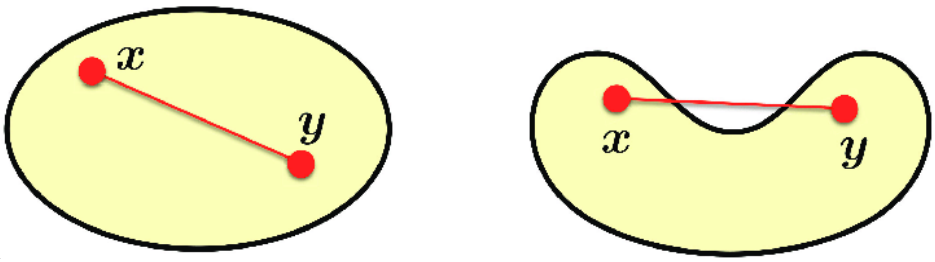
Preliminaries and Fundamentals

1.1 Convex Analysis

1.1.1 Convex sets

Definition 1.1.1 *A set C is convex if the line segment between any two points in C lies in C, i.e. $\forall x_1, x_2 \in C, \forall \theta \in [0,1]$*

$$\theta x_1 + (1-\theta)x_2 \in C.$$



1.286800in4.800600in

Figure 1.1: Example of a convex set (left) and a non-convex set (right).

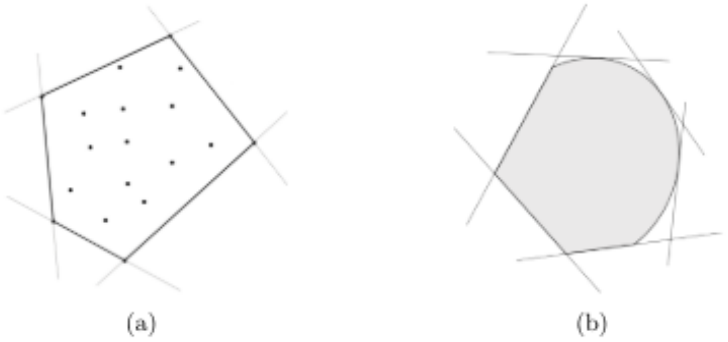
Simple examples of convex sets are:

- The empty set $\varnothing$, the singleton set $\{x_0\}$, and the complete space $\mathbb{R}^n$;

• Lines $\{a^T x = b\}$, line segments, hyperplanes $\{A^T x = b\}$, and halfspaces $\{A^T x \leq b\}$;

• Euclidian balls $B(x_0, \epsilon) = \{x | ||x - x_0||_2 \leq \epsilon\}$.

We can generalize the definition of a convex set above from two points to any number of points n. A convex combination of points $x_1, x_2, \ldots, x_k \in C$ is any point of form $\theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_k x_k$, where $\theta_i \geq 0$, $i = 1...k$, $\sum_{i=1}^{k} \theta_i = 1$. Then, a set C is convex if any convex combination of points in C is in C.



2.135200in4.926800in

Figure 1.2: (a) Representation of a convex set as the convex hull of a set of points. (b) Representation of a convex set as the intersection of a (possibly infinite) number of halfspaces.

We can take this even further to infinite countable sums: C convex if $\forall x_i \in C, \theta_i \geq 0, i = 1, 2, \ldots, \sum_{i=1}^{\infty} \theta_i = 1$:

$$\sum_{i=1}^{\infty} \theta_i x_i \in C,$$

if the series converges.

1.1.2 Convex functions

A convex function is a function defined on a convex domain such that, for any two points in the domain, the segment between the two points lies above the function curve between them. We will show below that this definition is closely connected to

the concept of a convex set: a function f is convex if and only if its epigraph, the set of all points above the function graph, is a convex set. We restate these results more precisely:

Definition 1.1.2 *A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $dom(f)$ is a convex set and if $\forall x, y \in dom(f)$, $\forall \theta \in [0, 1]$, we have:*

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y).$$

The epigraph of a function $f : \mathbb{R}^n \to \mathbb{R}$ is the set of points

$$epi(f) = \{(x, t) | x \in dom(f), t \geq f(x)\}.$$

Lemma 1.1.1 *The function f is convex if the set $epi(f)$ is convex.*

1.1.3 Convexity and Derivatives

Definition 1.1.3 *Let $f : U \subset E \to F \subset \mathbb{R}$, with U an open subset of $E \subset \mathbb{R}^n$ and a a point in U. We say that f is a differentiable function at a if there exists a linear map $L \in L(E, F)$ such that*

$$\lim_{x \to a} \frac{\|f(x) - f(a) - L(x - a)\|_F}{\|x - a\|_E} = 0.$$

The map L is then unique and is called the differential of f at a, denoted by $df(a)$.

Definition 1.1.4 *We say that f is twice differentiable at a if f is differentiable on a neighborhood of a, and if df is itself differentiable at a.*

We say that f is twice differentiable on U if f is twice differentiable at every point in U.

Definition 1.1.5 *Let f be a differentiable function at every point in $U \subset C \subset \mathbb{R}^n$. We can consider the mapping*

$$df : U \to L(E, F)$$
$$x \to df(x).$$

If df is continuous on U, we say that f is continuously differentiable on U, or alternatively, that f is of class C^1 on U.

Definition 1.1.6 *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function, its gradient at the point x is expressed as*

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(x) \\ \frac{\partial f}{\partial x_2}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{pmatrix}.$$

Definition 1.1.7 *Let U be an open set in $\mathbb{R}^n$, and f, be a function twice differentiable at a, where all second-order partial derivatives are defined at a. Then the matrix*

$$\mathbf{Hess}(a) = \begin{pmatrix} \frac{\partial^2 f}{\partial^2 x_1}(a) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(a) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(a) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(a) & \frac{\partial^2 f}{\partial^2 x_2}(a) & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(a) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(a) & \frac{\partial^2 f}{\partial x_n \partial x_2}(a) & \cdots & \frac{\partial^2 f}{\partial^2 x_n}(a) \end{pmatrix},$$

is called the hessian matrix of f at a.

Proposition 1.1.1 *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function that is twice continuously differentiable over a convex domain C.*

1. *f is a convex function on C if and only if the hessian matrix is positive semi-*

definite, i.e.,

$$\forall x \in C,\ y^T \mathbf{Hess}(x) y \geqslant 0,\ \forall y \in \mathbb{R}^n.$$

2. *f is a strictly convex function on C if and only if the hessian matrix is positive definite.*

$$\forall x \in C,\ y^T \mathbf{Hess}(x) y > 0,\ \forall y \in \mathbb{R}^n / \{0\}\ .$$

Definition 1.1.8 *A function f is said to be coercive on a convex set C if*

$$\lim_{\|x\| \to +\infty} \frac{f(x)}{\|x\|} = +\infty,\ \forall x \in C.$$

1.2 Optimizations Problems

Optimization problems are a fundamental concept in mathematics and are used to solve a wide range of problems in various fields, including economics, engineering, and computer science. The goal of optimization is to find the best solution among a set of possible solutions

1.2.1 Basic concepts

In optimization problems, we seek to find the value of the independent variable(s) that minimizes or maximizes a function, subject to certain constraints. The objective function is the function that we want to optimize, and the constraints are the limitations that the solution must satisfy. The objective function can be either a linear or nonlinear function, and the constraints can be either equality or inequality constraints.

Let $(x_1, x_2, ..., x_n)$ be a vector of n variables, and let $f(x)$ be the objective function. The optimization problem can be written as: Minimize/Maximize $f(x)$ Subject to: $g_i(x) \leq 0,\ i = 1, ..., m$ (inequality constraints) $h_j(x) = 0,\ j = 1, ..., p$ (equality

constraints) where $g_i(x)$ and $h_j(x)$ are functions that define the constraints.

1.2.2 Types of optimization problems

There are several types of optimization problems, each with its own unique characteristics and solution techniques.

Unconstrained Optimization Problems

Unconstrained optimization problems are optimization problems where there are no constraints. The goal is to find the minimum or maximum value of the objective function without any restrictions.
Example: Find the minimum value of the function $f(x) = x^2 + 2x + 1$.

Constrained Optimization Problems

Constrained optimization problems are optimization problems where there are constraints. The goal is to find the minimum or maximum value of the objective function subject to the constraints.
Example : Find the minimum value of the function $f(x) = x^2 + 2x + 1$. subjected to $x \leq 0$.

Linear Programming Problems

Linear programming problems are a type of constrained optimization problem where both the objective function and the constraints are linear functions.

- **Linear Programming (LP)**: Minimize or maximize a linear function subject to linear constraints. The linear programming problem can be written as:

Minimize/Maximize $c^T x$ Subject to: $Ax \leq b$, $x \geq 0$, where c is a vector of coefficients, A is a matrix of coefficients, b is a vector of constants, and x is a vector of variables.

Example: Maximize $z = 2x + 3y$ subject to $x + y \leq 10$, $x \geq 0$, $y \geq 0$

- **Quadratic Programming (QP):** Minimize or maximize a quadratic function subject to linear constraints. The quadratic programming problem can be written as: Minimize/Maximize $x^T Q x + c^T x$ Subject to: $Ax \leq b$, $x \geq 0$, where Q is a matrix of coefficients, c is a vector of coefficients, A is a matrix of coefficients, b is a vector of constants, and x is a vector of variables.

  Example: Minimize $z = x^2 + y^2$ subject to $x + y \leq 5$, $x \geq 0$, $y \geq 0$

Non-Linear Programming (NLP)

Minimize or maximize a non-linear function subject to non-linear constraints. The non-linear programming problem can be written as: Minimize/Maximize $f(x)$ Subject to: $g_i(x) \leq 0$, $i = 1, ..., m$ (inequality constraints), $h_j(x) = 0$, $j = 1, ..., p$ (equality constraints), where $f(x)$ is the objective function, and $g_i(x)$ and $h_j(x)$ are functions that define the constraints.

Example: Minimize $z = x^3 + y^2$ subject to $x^2 + y^2 \leq 4$, $x \geq 0$, $y \geq 0$

1.2.3 Optimilaty conditions

Optimality conditions are used to determine whether a given solution is optimal or not. There are several types of optimality conditions, including:

First-Order Optimality Conditions

The first-order optimality condition states that a necessary condition for a solution to be optimal is that the gradient of the objective function with respect to the

variable(s) being optimized is equal to zero at that point. Formally, let $f(x)$ be the objective function and let x^* be a candidate solution. Then, x^* is a local minimizer (or maximizer) if: $\nabla f(x^*) = 0$. where $\nabla f(x)$ is the gradient of $f(x)$.

Second-Order Optimality Conditions

The second-order optimality condition states that a necessary condition for a solution to be optimal is that the Hessian matrix of the objective function with respect to the variable(s) being optimized is positive semi-definite (PSD) at that point. Formally, let $f(x)$ be the objective function and let x^* be a candidate solution. Then, $x*$ is a local minimizer (or maximizer) if: $\mathbf{Hess}(f)(x^*) \geq 0$, where $\mathbf{Hess}(f)(x^*)$ is the Hessian matrix of $f(x)$ at x^*.

Chapter 2

Interior Point Methods for Linear Optimization

2.1 Linear Optimization

A linear program **(LP)** is an optimization problem that involves maximizing (or minimizing) a linear objective function of n decision variables subject to a set of constraints expressed as linear equations or inequalities.

Definition 2.1.1 *A linear program is defined as a mathematical program where the objective function is linear and the set of constraints is affine.*

Mathematical notation: There are three forms to express a linear program:

1. Canonical form:

$$\begin{cases} \min\limits_{x\in\mathbb{R}^n} c^t x, & 0 \neq c \in \mathbb{R}^n \text{ (given vector)}, \\ Ax \geq b, & A\ (m \times n) - matrix,\ \text{(given)}, \\ x \geq 0, & \text{variable} \end{cases}$$

2. Standard form:
$$\begin{cases} \min_x c^t x, \\ Ax = b, x \geq 0. \end{cases}$$

3. General form:
$$\begin{cases} \min_x c^t x, \\ Ax \leq b, \\ Bx \geq b', \\ B\ (p \times n) - matrix, b' \in \mathbb{R}^n. \end{cases}$$

2.1.1 Duality in Linear Optimization

Consider a linear program in the following standard form

$$\begin{cases} \min_x c^t x, \\ Ax = b, x \geq 0, \end{cases} \tag{P}$$

where A is an $(m \times n)$ matrix, $b \in \mathbb{R}^m, c \in \mathbb{R}^n$.

Obviously, any **(LP)** can be easily converted to this form.

The dual of (P) is defined by

$$\begin{cases} \max\limits_{(y,z)\in\mathbb{R}^m\times\mathbb{R}^n} b^t y, \\ A^t y + z = c, \\ z \geq 0, y \in \mathbb{R}^m. \end{cases} \tag{D}$$

We will denote

$$\mathcal{F}_{(P)} = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\},$$
$$\mathcal{F}^0_{(P)} = \{x \in \mathbb{R}^n : Ax = b, x > 0\},$$

and

$$\mathcal{F}_{(D)} = \{(y,z) \in \mathbb{R}^{m+n} : A^t y + z = c, z \geq 0\},$$
$$\mathcal{F}^0_{(D)} = \{(y,z) \in \mathbb{R}^{m+n} : A^t y + z = c, z > 0\},$$

the sets of feasible and strictly feasible solutions of the two problems of (P) and (D) respectively.

Theorem 2.1.1 (Weak duality) *Weak duality states that the optimal value of the primal problem is greater than or equal to the optimal value of the dual problem.*

$$c^t x \geq b^t y, \quad \forall \ x \ \text{ is a feasible for } ($$

$P), \quad \forall \ y \ \text{ is a feasible for } (D).$

Theorem 2.1.2 (Strong duality) *Strong duality states that the optimal value of the primal problem is equal to the optimal value of the dual problem.*

If $x \in \mathcal{F}_{(P)}$ and $(y,z) \in \mathcal{F}_{(D)}$, where $c^t x = b^t y$, so x, and (y,z) are optimal solutions for (P) and (D), respectively.

Theorem 2.1.3 (The complementary slackness theorem) *States that for feasible, primal, and dual solutions x and y, respectively $z = c - A^t y$, the vector of slack variables associated with y, so x and (y,z) are optimal if $x_i z_i = 0$, $\forall i = 1, ..., n$.*

Theorem 2.1.4 • *If one of the problems **(P)** and **(D)** has a finite optimal solution, the same is true for the other, and their corresponding optimal values are equal.*

• *If one of the problems has an unbounded optimal value, the other has no optimal solution*

If B is a basis of A i.e B is a sub-matrix formed by m linearly independent columns

of $A.$, then we introduce these definitions:

• The basic solution associated with B is the point $x = (x_B, x_N) \in \mathbb{R}^n$, such that: $x_B = B^{-1}b, x_N = 0.$

• A basic solution that satisfies $x_B \geq 0$ is considered a feasible basic solution.

• A point x is a vertex of $\mathcal{F}_{(P)}$ if and only if it is a feasible basic solution.

• A vertex is non-degenerate if $x_B > 0$, and degenerate otherwise (if at least one component of x_B is zero).

2.1.2 Simplex Method

It was developed in the late $1947s$ by **G. Dantzig**. It systematically takes into account previously established results. It evolves on the boundary of the feasible region from one adjacent vertex to another, reducing the objective value to the optimum. A simple optimality criterion allows for the recognition of the optimal vertex. Since the number of vertices is finite, the algorithm thus defined converges in a finite number of iterations not exceeding the number $C_n^m = \frac{n!}{m!(n-m)!}$, assuming that all visited vertices are non-degenerate.

In the degenerate case, the algorithm may cycle, but there are suitable techniques to avoid this phenomenon. In general, the simplex method exhibits a very satisfactory numerical behavior, confirmed by its numerous applications in solving a wide class of practical problems. In theory, the method is not as successful, as it is considered inefficient due to its exponential arithmetic complexity, which is of the order of $\mathcal{O}(2^n)$ operations. The Simplex Method is a popular algorithm for solving linear programming problems. It is based on the following steps:

Convert the linear programming problem into standard form by adding slack variables. Construct the initial tableau by rewriting the problem constraints in matrix

form. Use the simplex method to find the optimal solution.

Step 1: Converting the Problem to Standard Form

Given a linear programming problem:

$$\text{Maximize/Minimize} \quad z = c^T x \qquad \text{Subject to} \quad Ax \leq b \qquad x \geq 0$$

where A is a matrix of coefficients, b is a vector of constants, and c is a vector of coefficients. To convert the problem to standard form, we add slack variables to the constraints. Let x_0 be the slack variable for each constraint. The standard form is:

$$\text{Maximize/Minimize} \quad z = c^T x \qquad \text{Subject to} \quad Ax + x_0 = b \qquad x, x_0 \geq 0$$

where x_0 is the slack variable.

Step 2: Constructing the Initial Table

The initial table is constructed by rewriting the problem constraints in matrix form. Let A be a matrix of coefficients, b be a vector of constants, and x_0 be the slack variable. The initial tableau is:

	A	b
x_0	I	0
x	$-A$	c

where I is the identity matrix.

Step 3: Using the Simplex Method

The simplex method uses the following steps to find the optimal solution:

Find an initial basic feasible solution. Check if the current solution is optimal. If not, pivot and repeat step 2 until an optimal solution is found. An initial basic feasible solution is a solution that satisfies all constraints and has a non-zero value for at least one decision variable.

To check if the current solution is optimal, we use the following conditions:

If the current solution is feasible and satisfies all constraints, it is optimal. If the current solution is not feasible, it is not optimal. To pivot, we select a column with a positive reduced cost and pivot on that column.

Here is an illustration example:

Consider the following linear programming problem:

$$\begin{aligned} \text{Maximize} \quad & z = 2x_1 + 3x_2 \\ \text{Subject to} \quad & x_1 + x_2 \leq 4 \\ & 2x_1 + x_2 \leq 6 \\ & x_1, x_2 \geq 0 \end{aligned}$$

We convert the problem to standard form by adding slack variables:

$$\begin{aligned} \text{Maximize} \quad & z = 2x_1 + 3x_2 \\ \text{Subject to} \quad & x_1 + x_2 + x_3 = 4 \\ & 2x_1 + x_2 + x_4 = 6 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

We construct the initial tableau:

	A	b
x_3	1	-4
x_4	2	-6
x_1	1	0
x_2	1	0

We find an initial basic feasible solution by setting all non-basic variables to zero.

The current solution is: $x_1 = x_2 = x_3 = 0$. The reduced cost of each variable is calculated as follows: $r_j = c_j - c_B^T A_j$, where c_j is the coefficient of variable j, c_B^T is the transpose of the coefficient vector of the basic variables, and A_j is the column of matrix A corresponding to variable j.

The reduced cost of each variable is:

$$r_{x_1} = 2 - (-4) = -2$$

$$r_{x_2} = 3 - (-4) = 7$$

$$r_{x_3} = -4 - (-4) = 0$$

$$r_{x_4} = -6 - (-4) = -2$$

We select column x_2 with a positive reduced cost and pivot on that column.

The new tableau is:

	A	b
x_3	1	-4
x_4	2	-6
x_1	0	8
x_2	-7	-14

We repeat step 3 until an optimal solution is found. The final solution is: $x^\star = (2, 2)$

The optimal value of the objective function is:

$$z^\star = (2)(2) + (3)(2) = 8 + 6 = 14$$

This concludes our illustration example of the Simplex Method for Linear Programming.

2.2 Interior Point Method

The Interior Point Method (IPM) is a popular algorithm for solving linear programming problems. It is based on the concept of moving from an interior feasible solution to the optimal solution by iteratively improving the objective function. The IPM starts by solving a relaxed problem, which is a linear programming problem with a perturbation of the constraints. The solution to this relaxed problem is then used to generate a new iterate, which is closer to the optimal solution. This process is repeated until the optimal solution is reached.

The IPM has several advantages over other linear programming methods, such as the Simplex Method. It is more robust and can handle degenerate problems, and it can solve problems with a large number of variables and constraints. The IPM also has a faster convergence rate than other methods, making it more efficient.

2.2.1 Approach for Interior Point Method

Let us set once and for all the following assumptions

(H1) S_{int} is non-empty,

(H2) T_{int} is non-empty,

(H3) A is full rank $(rg(A) = m < n)$

with

$$S_{int} = \{x \in \mathbb{R}^n_+ : Ax = b, x > 0\},$$
$$T_{int} = \{(y, s) \in \mathbb{R}^m \times \mathbb{R}^n_+ : A^T y + s = c\}.$$

• These assumptions are necessary for the existence of feasible solutions.

2.2.2 Primal Method

the standard linear program **(P)** is formulated as:

$$(P)\begin{cases} \min c^T x, & 0 \neq c \in \mathbb{R}^n \text{ (given vector)}, \\ Ax = b, & A\ (m \times n) - matrix,\ \text{(given)}, \\ x \geq 0. \end{cases}$$

Many of the new interior point methods can be seen as variants of Frisch's logarithmic barrier methods (1955).

Indeed, to problem **(P)** we associate the following nonlinear problem:

$$(P_\mu)\begin{cases} \min c^T x - \mu \sum_{i=1}^{n} \ln x_i = f_\mu(x), \\ Ax = b, \\ x > 0. \end{cases}$$

The objective f_μ of P_μ is called the logarithmic barrier function, and $\mu > 0$ denotes the barrier parameter.

The solution of (P_μ) is equivalent to that of **(P)** in the sense that if x^*_μ is an optimal solution of (P_μ), then $x^* = \lim_{\mu \to 0} x^*(\mu)$ is an optimal solution of **(P)**. It is therefore sufficient to solve (P_μ).

Proposition 2.2.1 *Under assumption **(H1)**, for all $\mu > 0$, the set of optimal solutions of **(P)** is non-empty and bounded.*

Proof. The constraints of (P_μ) are linear, and the objective function f_μ is strictly convex since $f"(x) = \mu X^{-2}$ is positive definite. Therefore, if the solution exists, it is unique, global, and completely characterized by the **KKT,** i.e., x optimality if and only if $\exists y \in \mathbb{R}^m$ such that

$$\begin{cases} c - \mu X^{-1} e - A^T y = 0, \\ Ax = b, \\ x > 0. \end{cases} \tag{2.1}$$

The duality jump is defined by

$$c^T x - b^T y = \mu x^T X^{-1} e = n\mu$$

■

Therefore, as μ tends to zero, $c^T x$ and $b^T y$ converge to the same optimal value. Consequently, x and y are optimal solutions of **(P)** and **(D)** respectively.

Technical Description

Following the principle of feasible point methods (TC), the nonlinear system 2.1 will be solved using the Newton method. This can be justified by the magnificent results

obtained theoretically and by the numerical subtleties offered by this method.

Thus, we limit ourselves to seeking approximate solutions along the central trajectory by forming a decreasing sequence $\{\mu^k = \frac{(x^k)^T s^k}{n}\}$ converging to zero.

More precisely, it is a matter of solving the system $F(x, y) = 0$ such that

$$F(x,y) = \begin{pmatrix} c - \mu X^{-1} e - A^T y \\ Ax - b \end{pmatrix}.$$

Or, $x_+ = x + \alpha \Delta x, y_+ = y + \Delta y$, with $\alpha > 0$, is a displacement introduced to ensure $x_+ > 0$, and $\Delta w = (\Delta x, \Delta y)$ is a solution of the linear system

$$DF(x,y)\Delta w = -F(x,y), \tag{2.2}$$

With $DF(x, y)$ being the Jacobian matrix of F at the point (x, y), given by:

$$DF(x,y) = \begin{pmatrix} \mu X^{-2} & -A^T \\ A & 0 \end{pmatrix}.$$

According to 2.2, we obtain the system:

$$\begin{pmatrix} \mu X^{-2} & -A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} -(c - \mu X^{-1} e - A^T y) \\ 0 \end{pmatrix}.$$

We change μ ($\widehat{\mu} = \sigma\mu, 0 < \sigma < 1$) and iterate until optimality.

The algorithm corresponding to this version can be written as follows:

Algorithm of the primal (TC) method

In this part, we present the prototype algorithm of the primal (TC) method.

Algorithm start

- Find $X^0 \in S_{int}, y^0 \in \mathbb{R}^m, \mu^0 > 0, k = 0,$
- While the optimality test is not satisfied ($c^T x - b^T y > \epsilon$) do

1. Solve the following linear system

$$\begin{pmatrix} \mu^k (X^k)^{-2} & -A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \end{pmatrix} = \begin{pmatrix} -(c - \mu^k (X^k)^{-1} e - A^T y^k) \\ 0 \end{pmatrix}.$$

2. Find $0 < \alpha \leq 1 / x^{k+1} = x^k + \alpha \Delta x^k > 0;$
3. $y^{k+1} = y^k + \Delta y^k > 0;$
4. $\mu^{k+1} = \sigma \mu^k, 0 < \sigma < 1;$
5. $k = k + 1;$

End while

End Algorithm

2.2.3 Dual Method

Let's consider the standard linear program **(D)**

$$(D) \begin{cases} \max b^T y \\ A^T y + s = c, \\ s \geq 0 \end{cases}$$

to which we associate the following barrier problem

$$(D_\mu) \begin{cases} \max b^T y - \mu \sum_{i=1}^{n} \ln s_i \\ A^T y + s = c, \\ s > 0, \end{cases}$$

with, $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$ et $c \in \mathbb{R}^n$.

Technical Description

Applying the same techniques used in the primal version, we obtain the following nonlinear system:

$$\begin{cases} A^T y + s - c = 0, \\ Ax - b = 0, \\ \mu S^{-1} e - x = 0, \\ s > 0, x > 0. \end{cases} \tag{2.3}$$

We solve it using the Newton method starting from a point $x \in \mathbb{R}^n_{++}$ and $(y, s) \in T_{int}$. The new iterate is given by:

$$x_+ = x + \alpha \Delta x, y_+ = y + \Delta y, s_+ = s + \alpha \Delta s,$$

with, $\alpha > 0$ is the displacement step introduced to maintain the strict positivity of x_+ and s_+ (e.i, $x_+ = x + \alpha \Delta x > 0, s_+ = s + \alpha \Delta s > 0$).

Algorithm of the dual (TC) method

The prototype algorithm of this version is written as follows:

Algorithm start

• Choose $x^0 \in \mathbb{R}^n_{++}, \mu_0 > 0$ and find $(y^0, s^0) \in T_{int}, k = 0$;

While the optimality test is not satisfied, do

1. Solve the following linear system

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ -I & 0 & \mu(S^k)^{-2} \end{pmatrix} \begin{pmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{pmatrix} = \begin{pmatrix} -Ax^k + b \\ 0 \\ -\mu(S^k)^{-1}e + x^k \end{pmatrix}$$

2. Find $0 < \alpha \leq 1/x^{k+1} = x^k + \alpha\Delta x^k > 0$ and $s^{k+1} = s^k + \alpha\Delta s^k > 0$;

3. $x^{k+1} = x^+, s^{k+1} = s^+, y^{k+1} = y^k + \Delta y^k > 0$;

4. $\mu^{k+1} = \sigma\mu^k, 0 < \sigma < 1$;

5. $k = k + 1$;

2.2.4 Primal-Dual Method

The primal-dual path-following methods were introduced in the early 1990s. They have attracted a great deal of attention from researchers worldwide and have generally shown excellent practical and theoretical behavior (polynomial complexity and superlinear convergence).

Technical Description

The primal-dual path-following methods are based on solving the following nonlinear system:

$$\begin{cases} A^T y + s - c = 0, \\ Ax - b = 0 \\ XSe - \mu e = 0, \\ s > 0, x > 0, \end{cases} \tag{2.4}$$

which can be obtained by combining the primal nonlinear system 2.2 and the dual nonlinear system 2.3. The system has a unique solution if and only if assumptions **(H1)** and **(H2)** are simultaneously satisfied.

Clearly, the system 2.4 is more suitable for numerical treatment than the systems 2.1 and 2.3 (it is less nonlinear). Therefore, the primal-dual version is more interesting than the two previous versions for the following reasons:

- The theoretical results are more consistent.
- The algorithms are more efficient in practice.

Now, we will solve the system 2.4 using the Newton method, i.e., for each $\mu > 0$, we solve the following linear system:

$$DF(x, y, s)\Delta w = -F(x, y, s),$$

with

$$F(x, y, s) = \begin{pmatrix} A^T y + s - c \\ Ax - b \\ Xs - \widehat{\mu} e \end{pmatrix}, \widehat{\mu} = \sigma\mu, 0 < \sigma < 1.$$

$DF(x, y, s)$ denotes the Jacobian matrix of F at the point (x, y, s) and Δw denotes the Newton direction.

Through simple calculations, we can solve the following linear system:

$$\begin{cases} S\Delta x + X\Delta s = Xs - \widehat{\mu} e, \\ A\Delta x = 0, \\ A^T\Delta y + \Delta s = 0. \end{cases}$$

where its solution is

$$\Delta x = [S^{-1} - S^{-1}XA^T(AS^{-1}XA^T)AS^{-1}](Xs - \widehat{\mu}e),$$
$$\Delta y = [(AS^{-1}XA^T)^{-1}AS^{-1}](Xs - \widehat{\mu}e),$$
$$\Delta s = -A\Delta y.$$

The new iterate is expressed in the form

$$(x_+, y_+, s_+) = (x, y, s) + \alpha(\Delta x, \Delta y, \Delta s),$$

iterate until obtain a value $\widehat{\mu}$ sufficiently close to zero. Positivity procedure:

We know that the new iterate must be positive, i.e.,

$$x^+ = x + \alpha\Delta x > 0, \tag{2.5}$$

$$s^+ = s + \alpha\Delta s > 0, \tag{2.6}$$

from 2.5, we have

$$\alpha\Delta x > -x.$$

This is equivalent to

$$\alpha = \alpha_1 = \begin{cases} \min(\frac{-x_i}{(\Delta x)_i}), \text{ si } I_0 \neq \varnothing, \\ +\infty, \text{ si } I_0 = \varnothing, \end{cases}$$

such that

$$I_0 = \{i : \Delta x_i < 0, i = 1, ..., n\}.$$

Similarly for 2.6, we've

$$\alpha\Delta s > -s.$$

This is equivalent to

$$\alpha = \alpha_2 = \begin{cases} \min(\frac{-s_i}{(\Delta s)_i}), \text{ si } I_1 \neq \varnothing, \\ +\infty, \text{ si } I_1 = \varnothing, \end{cases}$$

such that

$$I_0 = \{i : \Delta s_i < 0, i = 1, ..., n\}.$$

Then, it suffices to take the following displacement step:

$$\alpha = \rho \min(\alpha_1, \alpha_2), 0 < \rho < 1.$$

Centrality Factor

To measure the quality of a found solution, we introduce a so-called "centrality" factor defined by the scalar$\|XSe - \mu e\|$

In this regard, a point is said to be a neighbor of the central trajectory if it belongs to the following set:

$$S_{cent}(\sigma) = \{(x, y, s) \in S_{int} \times T_{int} : \|XSe - \mu e\| \leq \sigma\mu, \mu > 0\}.$$

Assumptions: Let δ and σ be two constants such that

$$0 \leq \sigma < 1/2 \text{ et } 0 < \delta < \sqrt{n}, \tag{2.7}$$

$$\frac{\sigma^2\delta^2}{2(1-\sigma)} \leq \sigma(1 - \frac{\delta}{\sqrt{n}}). \tag{2.8}$$

These assumptions allow, in particular, to maintain $x > 0$ and $s > 0$ during the iterations while keeping the found solution always close to the trajectory.

Algorithm of the primal-dual (TC) method

In this section, we present the algorithm of the primal-dual (TC) method, and then we give some results that are used for the study of convergence and arithmetic complexity for this version.

Algorithm start

Data: let $\epsilon > 0$ be a precision parameter, δ and σ be two constants satisfying 2.7 and 2.8,

Initialization $(x^0, y^0, s^0) \in S_{cent}(\sigma)$, such that $\mu_0 = \frac{(x^0)^T s^0}{n}$ (major difficulty), $k = 0$;

While $(x^k)^T s^k \geq \epsilon$ do

1. Take $\mu_{k+1} = \mu_k(1 - \frac{\delta}{\sqrt{n}})$;

2. calculate

$$\begin{cases} \Delta x = [(S^k)^{-1} - (S^k)^{-1}X^k A^T (A(S^k)^{-1}X^k A^T)AS^{-1}](X^k S^k e - \mu_{k+1}e), \\ \Delta y = [-(A(S^k)^{-1}X^k A^T)^{-1}A(S^k)^{-1}](X^k S^k e - \mu_{k+1}e), \\ \Delta s = -A^T \Delta y^k; \end{cases}$$

3. Set $\alpha = \rho \min(\alpha_1^k, \alpha_2^k), 0 < \rho < 1$;

4. Find

$$\begin{aligned} x^{k+1} &= x^k + \Delta x^k, \\ s^{k+1} &= s^k + \Delta s^k, \\ y^{k+1} &= y^k + \Delta y^k; \end{aligned}$$

5. $k = k + 1$;

End while

End Algorithm

Chapter 3

Central Trajectory Method via Kernel Function

3.1 Primal-dual method based on kernel functions

Let's recall that the primal-dual central trajectory methods are based on the following principles:

• We associate the primal problem **(P)** with the perturbed problem (P_μ).

• By applying the Karush-Kuhn-Tucker **(KKT)** conditions to the latter, we obtain the following nonlinear system

$$\begin{cases} A^T y + s - c = 0, \\ Ax - b = 0 \\ XSe - \mu e = 0, \\ s > 0, x > 0, \end{cases} \tag{3.1}$$

Where we assume that the sets S_{int} and T_{int} are non-empty. This condition is also called the interior point condition (IPC). Under this condition (IPC), the system 3.1

has a unique solution $(x(\mu), y(\mu), s(\mu))$ for each $\mu > 0$.

• We solve the system 3.1 using the Newton method, which involves finding the direction $\Delta w = (\Delta x, \Delta y, \Delta s)$ by solving the following linear system.

$$\begin{cases} A\Delta x = 0, \\ A^T\Delta y + \Delta s = 0, \\ s\Delta x + x\Delta s = \mu e - xs. \end{cases} \tag{3.2}$$

The Newton iteration is written as follows:

$$x_+ = x + \alpha\Delta x,\ s_+ = s + \alpha\Delta s,\ y_+ = y + \alpha\Delta y, \tag{3.3}$$

where α is a positive parameter ($\alpha > 0$) introduced in such a way as to maintain the strict feasibility of the new iterates x_+ and s_+.

3.2 Kernel function and its qualification

In recent years, kernel functions have gained significant popularity. The utilization of kernel functions has proven to be crucial and highly advantageous in various domains of mathematical programming research. A kernel function based on IPMs is considered one of the most effective approaches for conducting interior point analysis and solving the LO linear optimization,

Definition 3.2.1 *Let $\Psi(t) : \mathbb{R}_{++} \to \mathbb{R}_+$ be a function twice continuously differentiable. Then Ψ is called a kernel function, if it satisfies the following conditions:*

1. $\Psi'(1) = \Psi(1) = 0$,

2. $\Psi"(t) > 0$,

3. $\lim_{t\to 0^+} \Psi(t) = \lim_{t\to+\infty} \Psi(t) = +\infty$.

The first two conditions show that Ψ *is strictly convex and minimal at* 1 *with* $\Psi(1) = 0$, *implying that* $\Psi(t)$ *can be written as follows*

$$\Psi(t) = \int_1^t (\int_1^x \Psi"(y)dy)dx. \tag{3.4}$$

Condition (3) indicates that Ψ *is a barrier function.*

Lemma 3.2.1 *Let* $\Psi(t)$ *be a twice-differentiable function, then the following properties are equivalent:*

(a) $\Psi(\sqrt{t_1 t_2}) \leq \frac{\Psi(t_1)+\Psi(t_2)}{2}$, *for all* $t_1, t_2 > 0$.

(b) $t\Psi"(t) + \Psi\prime(t) \geq 0, t > 0$.

(c) $\Psi(e^{\epsilon})$ *is convex.*

In the following table, we provide the different known kernel functions in the literature [1, 5, 2] and the complexity of their algorithm for small- and large-step interior point methods

	Kernel function	algorithmic complexity	algorithmic complexity
	$\Psi_i(t)$	Small step	large step
1	$\frac{1}{2}(t^2-1)-logt$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(n\log\frac{n}{\epsilon})$
2	$\frac{t^2-1}{2}+\frac{t^{1-q}-1}{q(q-1)}-\frac{q-1}{q(q-1)}(t-1), q>1$	$O(q\sqrt{n}\log\frac{n}{\epsilon})$	$O(qn^{\frac{q+1}{2q}}\log\frac{n}{\epsilon})$
3	$\frac{t^2-1}{2}+\frac{(e-1)^2}{e}\frac{1}{e-1}-\frac{e-1}{e}$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(n^{3/4}\log\frac{n}{\epsilon})$
4	$\frac{1}{2}(t^2-\frac{1}{t})^2$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(n^{2/4}\log\frac{n}{\epsilon})$
5	$\frac{t^2-1}{2}+e^{\frac{1}{t-1}}-1$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(\sqrt{n}(\log n)^2\log\frac{n}{\epsilon})$
6	$\frac{t^2-1}{2}-\int_1^t e^{\frac{1}{\xi}}d\xi$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(\sqrt{n}\log^2 n\log\frac{n}{\epsilon})$
7	$\frac{t^2-1}{2}+\frac{t^{1-q}-1}{q-1}, q>1$	$O(q^2\sqrt{n}\log\frac{n}{\epsilon})$	$O(qn^{\frac{q+1}{2q}}\log\frac{n}{\epsilon})$
8	$t-1+\frac{t^{1-q}-1}{q-1}, q>1$	$O(q^2\sqrt{n}\log\frac{n}{\epsilon})$	$O(q\log^2 n\log\frac{n}{\epsilon})$
9	$\frac{t^{1-p}-1}{p-1}-\log t, p\in[0,1]$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(n\log\frac{n}{\epsilon})$
10	$\frac{t^{1-p}-1}{p-1}+\frac{t^{1-q}-1}{q-1}, p\in[0,1], q>1$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(qn^{\frac{q+p}{q(1+p)}}\log\frac{n}{\epsilon})$
11	$t^2-1+\frac{t^{1-q}-1}{q-1}-\log t, p>1, q>1$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(qn^{\frac{q+1}{2q}}\log\frac{n}{\epsilon})$
12	$(m+1)t^2-(m+2)t+\frac{1}{t^m}, m>4$	$O(q^2\sqrt{n}\log\frac{n}{\epsilon})$	$O(m^{\frac{2m+1}{2m}}\log\frac{n}{\epsilon})$
13	$t^2-2t+\frac{1}{\sin(\frac{\pi t}{1+t})}, t>0$	$O(\sqrt{n}\log\frac{n}{\epsilon})$	$O(n^{3/4}\log\frac{n}{\epsilon})$
14	$\frac{t^2-1}{2}+\frac{4}{\pi p}[\tan^p h(t)-1], t>0$	$O(p^2\sqrt{n}\log\frac{n}{\epsilon})$	$O(pn^{\frac{p+2}{2(p+1)}}\log\frac{n}{\epsilon})$

Table 3.1: Different known kernel functions.

In the following, we use the kernel function proposed by Bouafia and others [2] as follows

$$\Psi(t)=\frac{t^2-1}{2}+\frac{4}{\pi p}[\tan^p h(t)-1], h(t)=\frac{\pi}{2t+2}, p\geq 2. \tag{3.5}$$

3.2.1 Properties of the Bouafia kernel function

In this section, we study properties of the Bouafia function with trigonometric barrier terms, which are essential for the analysis of algorithmic complexity.

Now, we give the first three derivatives of this kernel function

$$\Psi'(t) = t + \frac{4}{\pi}\sec^2 h(t)(\tan^{p-1} h(t))h'(t), \tag{3.6}$$

$$\Psi"(t) = 1 + \frac{4}{\pi}\sec^2 h(t)\left[\begin{array}{c}\left[\begin{array}{c}(p-1)\tan^{p-2} h(t)+\\ (p+1)\tan^{p} h(t)\end{array}\right](h'(t))^2+\\ [\tan^{p-1} h(t)]h"(t)\end{array}\right], \tag{3.7}$$

$$\Psi'''(t) = \frac{4}{\pi}\sec^2 h(t)\left[\begin{array}{c}\left[\begin{array}{c}(p-1)(p-2)\tan^{p-3} h(t)+\\ 2p^2\tan^{p-1} h(t)+\\ (p+1)(p+2)\tan^{p+1} h(t)\end{array}\right](h'(t))^3+\\ \left[\begin{array}{c}3(p-1)\tan^{p-2} h(t)+\\ 3(p+1)\tan^{p} h(t)\end{array}\right]h"(t)h'(t)+\\ [\tan^{p-1} h(t)]h'''(t)\end{array}\right], \tag{3.8}$$

with,

$$h'(t) = \frac{-\pi}{2(t+1)^2}, h"(t) = \frac{\pi}{(t+1)^3}, h'''(t) = \frac{-3\pi}{(t+1)^4}, \tag{3.9}$$

and

$$\sec h(t) = \frac{1}{\cos h(t)}.$$

Lemma 3.2.2 *For $h(t)$ defined in 3.5 and $p \geq 1$, we have the following properties:*

$$0 < h(t) < \frac{\pi}{2}, t > 0. \tag{3.10}$$

$$\tan h(t) > 0, t > 0. \tag{3.11}$$

$$\tan h(t) - \frac{2}{(p+1)\pi t} > 0, t > 0. \tag{3.12}$$

Proof. For 3.10, as long as $h(t)$ is a decreasing function in the interval $]0, +\infty[$ and

$\lim_{t\to 0+} h(t) = \frac{\pi}{2}$, $\lim_{t\to+\infty} h(t) = 0$, so we have

$$0 < h(t) < \frac{\pi}{2}, t > 0.$$

■

3.11 is an immediate consequence of 3.10.

To prove 3.12, we define the function g as follows:

$$g(t) = \tan h(t) - \frac{2}{(p+1)\pi t},$$

The first derivative of g is given by:

$$g'(t) = \frac{h'(t)}{\cos^2 h(t)} + \frac{2}{(p+1)\pi t} = \frac{(p+1)\pi t^2 h'(t) + 2\cos^2 h(t)}{(p+1)\pi t^2 \cos^2 h(t)}.$$

We know that $\sin(\frac{\pi}{2} - h(t)) = \cos h(t)$ and $\sin(\frac{\pi}{2} - h(t)) \leq \frac{\pi}{2} - h(t)$ and by using 3.10, it follows that:

$$\begin{aligned}
g'(t) &= \frac{(p+1)\pi t^2 h'(t) + 2\cos^2 \frac{\pi}{2}}{(p+1)\pi t^2 \cos^2 h(t)}, \\
&= \frac{(p+1)\pi t^2 h'(t) + 2\sin^2(\frac{\pi}{2} - h(t))}{(p+1)\pi t^2 \cos^2 h(t)}, \\
&\leq \frac{(p+1)\pi t^2 h'(t) + 2(\frac{\pi}{2} - h(t))^2}{(p+1)\pi t^2 \cos^2 h(t)}, \\
&= \frac{\frac{-(p+1)\pi^2 t^2}{2(t+1)^2} + 2(\frac{\pi^2}{4} - 2\frac{\pi}{2}\frac{\pi}{2t+2} + \frac{\pi^2}{4(t+1)^2})}{(p+1)\pi t^2 \cos^2 h(t)} \\
&= \frac{-2p\pi}{(p+1)\cos^2 h(t)(2t+2)^2} < 0
\end{aligned}$$

Therefore, g is a decreasing function in the interval $]0, +\infty[$ and since $\lim_{t\to+\infty} g(t) = 0$, we arrive at 3.12.

This completes the proof.

The following lemma is used to prove the efficiency of Bouafia kernel function 3.5.

Lemma 3.2.3 *Let $\Psi(t)$ be the kernel function defined in 3.5. Then, for all $t > 0$, we have:*

$$\Psi"(t) > 1. \tag{3.13}$$

$$\Psi'''(t) < 0. \tag{3.14}$$

$$t\Psi"(t) - \Psi'(t) > 0. \tag{3.15}$$

$$t\Psi"(t) + \Psi'(t) > 0. \tag{3.16}$$

Proof. For 3.13, using 3.9 and 3.11 which implies that all terms in the expression 3.7 are positive, we get 3.13.

To prove 3.14, using 3.8, 3.9, and 3.11, we obtain, for all $t < 0, \Psi'''(t) < 0$.

For 3.15, we use 3.6 and 3.7, we've for all $t < 0$,

$$t\Psi"(t) - \Psi'(t) = \frac{4}{\pi}\sec^2 h(t) \begin{bmatrix} (h'(t)^2[(p-1)\tan^{p-2} h(t) + (p+1)\tan^p h(t)]t+ \\ h"(t)(\tan^{p-1} h(t))t + (\tan^{p-1} h(t))(-h'(t)) \end{bmatrix}$$

Using 3.9 again for the last equality, all terms are positive, proving 3.15. For the last inequality 3.16, using 3.6 and 3.7, we've

$$t\Psi"(t) + \Psi'(t) = 2t + \frac{4}{\pi}\sec^2 h(t) \begin{bmatrix} (h'(t)^2[(p-1)\tan^{p-2} h(t)]t+ \\ (\tan^{p-1} h(t)) \begin{bmatrix} (p+1)(h'(t)^2 \tan h(t)t+ \\ h"(t)t + h'(t) \end{bmatrix} \end{bmatrix}$$

with,

$$\begin{bmatrix} (p+1)(h'(t)^2 \tan h(t)t+ \\ h"(t)t + h'(t) \end{bmatrix} = \frac{4\pi}{(2t+2)^4} \begin{bmatrix} 2t^2+ \\ \pi(p+1)t(\tanh(t) - \frac{2}{(p+1)\pi t} \end{bmatrix},$$

according to 3.12, the right-hand side of the above equality is positive, proving 3.16. This completes the proof of this Lemma. ■

Lemma 3.2.4 *Let $\Psi(t)$ be the kernel function defined in 3.5, if $\Psi(t)$ satisfies properties 3.13 and 3.15, then*

$$\Psi"(t)\Psi'(\beta t) - \beta\Psi'(t)\Psi"(\beta t) > 0, \forall t > 1, \forall \beta > 1.$$

Lemma 3.2.5 *for $\Psi(t)$, the kernel function defined in 3.5, we have*

$$\frac{1}{2}(t-1)^2 \leq \Psi(t) \leq \frac{1}{2}[\Psi'(t)]^2, t > 0. \tag{3.17}$$

$$\Psi(t) \leq (1 + \frac{\pi}{8}p)(t-1)^2, t > 1. \tag{3.18}$$

Proof. For 3.17, using 3.13 in 3.4, we obtain,

$$\begin{aligned} \Psi(t) = \int_1^t \int_1^x \Psi"(y)dydx &\geq \int_1^t \int_1^x dydx, \\ &= \int_1^t (x-1)dx, \\ &= \tfrac{1}{2}[(x-1)^2]_1^t \\ &= \tfrac{1}{2}(t-1)^2, \end{aligned}$$

where the left-hand side of inequality 3.17. Comes from. Similarly, for the right-hand side of this inequality, we find

$$\begin{aligned} \Psi(t) = \int_1^t \int_1^x \Psi"(y)dydx &\leq \int_1^t \int_1^x \Psi"(y)\Psi"(x)dydx, \\ &= \int_1^t \Psi"(x)[\Psi"(y)]_1^x dx, \\ &= \int_1^t \Psi"(x)\Psi"(x)dx, \\ &= \tfrac{1}{2}[\Psi'(t)]^2. \end{aligned}$$

For 3.18, we use the Taylor expansion at $t = 1$ with $\Psi(1) = \Psi'(1) = 0, \Psi"(1) =$

$(2 + \frac{\pi}{4}p)$ and $\Psi'''(t) < 0$ we obtain

$$\begin{aligned}\Psi(t) &= \Psi(1) + \Psi'(1)(t-1) + \frac{\Psi"(1)}{2}(t-1)^2 + \frac{\Psi'''(\xi)}{6}(t-1)^3, 1 \leq \xi \leq t \\ &= \frac{\Psi"(1)}{2}(t-1)^2 + \frac{\Psi'''(t)}{6}(\xi-1)^3 \\ &\leq \frac{(2+\frac{\pi}{4}p)(t-1)^2}{2} \\ &= (1+\frac{\pi}{8}p)(t-1)^2.\end{aligned}$$

■

Lemma 3.2.6 *Let $\sigma : [0, +\infty[\rightarrow [1, +\infty[$ be the inverse function of the function $\Psi(t)$ for $t \geq 1$ and $\rho : [0, +\infty[\rightarrow]0, 1]$ be the inverse function of the function $-\frac{1}{2}\Psi'(t)$ for $t \in]0, 1]$, then we have: 3.19*

$$1 + \sqrt{\frac{8}{8+\pi ps}} \leq \sigma(s) \leq 1 + \sqrt{2s}, s \geq 0, \tag{3.19}$$

$$\tan h(t) \leq (4z+2)^{\frac{1}{p+1}}, z \geq 0. \tag{3.20}$$

Proof. For 3.19, let $s = \Psi(t)$, according to 3.17, we have,

$$\Psi(t) \geq \frac{1}{2}(t-1)^2,$$

which implies,

$$2s \geq (t-1)^2, t \geq 1,$$

then, for all $t \geq 1$, we have,

$$t - 1 = |t-1| \leq \sqrt{2s},$$

where,

$$\sigma(s) = t \leq 1 + \sqrt{2s},$$

On the other hand, and according to 3.18, we have,

$$s = \Psi(t) \leq (1 + \frac{\pi}{8}p)(t-1)^2, t > 1,$$

which gives,

$$\frac{8s}{(8+\pi p)} \leq (t-1)^2,$$

which implies,

$$t - 1 = |t-1| \geq \sqrt{\frac{8}{8+\pi ps}},$$

hence

$$t = \sigma(s) \geq 1 + \sqrt{\frac{8}{8+\pi ps}}.$$

For 3.20, let $z = -\frac{1}{2}\Psi'(t), t \in]0,1]$ be equivalent to,

$$2z = -\Psi'(t),$$

from 3.6, we find,

$$2z = -(t + \frac{4}{\pi}[1 + \tan^2 h(t)][\tan^{p-1} h(t)]h'(t)),$$

which implies,

$$[1 + \tan^2 h(t)] \tan^{p-1} h(t) = \frac{-\pi(2z+t)}{4h'(t)} \leq 2(2z+1), \forall t \in]0,1],$$

and since,

$$\tan^{p+1} h(t) \leq [1 + \tan^2 h(t)] \tan^{p-1} h(t)$$

we find,

$$\tan^{p+1} h(t) \leq 2(2z+1),$$

hence,

$$\tan h(t) \leq (4z+2)^{\frac{1}{p+1}}, z \geq 0$$

which completes the proof. ■

Lemma 3.2.7 *Suppose* $\Psi(t_1) = \Psi(t_2)$ *with* $t_1 \leq 1 \leq t_2$ *and* $\beta \geq 1$, *then:*

$$\Psi(\beta t_1) \leq \Psi(\beta t_2).$$

The equality holds if and only if $\beta = 1$ or $t_1 = 1 = t_2$.

The following lemma is a very important lemma, which is valid for any kernel function satisfying conditions 3.13 and 3.14.

Lemma 3.2.8 *Let* $\sigma : [0, +\infty[\rightarrow [1, +\infty[$ *be the inverse function of the function* Ψ, $t \geq 1$, *then we have:*

$$\Phi(\beta v) \leq n\Psi(\beta\sigma(\frac{\Phi(v)}{n})), v \in \mathbb{R}_{++}, \beta \geq 1.$$

Proof. For $\beta > 1$, we consider the following maximization problem:

$$\max\{\Phi(\beta v) : \Phi(v) = z, \forall z > 0\}.$$

■

The first-order optimality conditions (or the Lagrangian) give:

$$\Phi(\beta v) + y(z - \Phi(v)) = 0, \tag{3.21}$$

where $y > 0$ is the Lagrange multiplier. By differentiating 3.21 with respect to v, we obtain:

$$\beta\Phi'(\beta v) - y\Phi'(v) = 0,$$

or equivalently:

$$\beta \sum_{i=1}^{n} \Psi'(\beta v_i) - y \sum_{i=1}^{n} \Psi'(v_i) = 0 \Leftrightarrow \sum_{i=1}^{n} \beta\Psi'(\beta v_i) - \sum_{i=1}^{n} y\Psi'(v_i) = 0,$$

which is equivalent to:

$$\beta\Psi'(\beta v_i) = y\Psi'(v_i), \forall i = 1, ..., n. \tag{3.22}$$

Since $\Psi'(t) = 0$ and $\Psi'(\beta) > 0$ for all $\beta > 1$, then $\Psi'(v) > 0$.

So $v_i \neq 1$, for v_i satisfies the constraints of problem 3.22, i.e., $\Psi(v) = z_i$, for all $i = 1, ..., n$, with z_i given, we distinguish two types of values of v_i

$$v_i = \begin{cases} v_i^+, \text{ si } v_i > 1, \\ v_i^-, \text{ si } v_i < 1, \end{cases}$$

so

$$v_i^- < 1 < v_i^+.$$

According to Lemma 3.2.7, we have:

$$\Psi(\beta v_i^-) \leq \Psi(\beta v_i^+)$$

so, we take:

$$v_i = v_i^+ > 1, \forall i = 1, ..., n.$$

Since we have a maximization problem, equation 3.22 gives:

$$\beta\Psi'(\beta v_i) > 0 \text{ et } y\Psi'(v_i) > 0, y > 0.$$

Now, we define the function g as follows:

$$g(t) = \frac{\Psi'(t)}{\Psi'(\beta t)}, t \geq 1,$$

from equation 3.22, we deduce that:

$$g(v_i) = \frac{\beta}{y}, \forall i = 1, ..., n.$$

So g is a constant function.

We derive g we obtain:

$$g'(t) = \frac{\Psi"(t)\Psi'(\beta t) - \beta\Psi'(t)\Psi"(\beta t)}{(\Psi'(\beta t))^2}, \forall \beta > 1, \forall t \geq 1.$$

According to Lemma 3.2.4, we find that $g\prime(t) > 0$. So, g is a strictly increasing function, hence all v_i are equal.

Let $v_i = t$, for all $i = 1, ..., n$, from the equality $\Psi(\mu) = z$, we deduce that:

$$\sum_{i=1}^{n} \Psi(t) = n\Psi(t) \Longleftrightarrow \Psi(t) = \frac{z}{n}.$$

Since σ is the inverse function of the function Ψ, then $t = \sigma(\frac{z}{n})$

$$\Phi(\beta v) \leq \max \Phi(\beta v) = \Phi(\beta t) = n\Psi(\beta t) = n\Psi(\beta\sigma(\frac{z}{n})) = n\Psi(\beta\sigma(\frac{\Phi(v)}{n})),$$

so

$$\Phi(\beta v) \leq n\Psi(\beta\sigma(\frac{\Phi(v)}{n})), \forall \beta > 1.$$

The case $\beta = 1$ is evident since:

$$\frac{\Phi(v)}{n} = \Psi(\sigma(\frac{\Phi(v)}{n})).$$

Which implies:

$$\Phi(v) = n\Psi(\sigma(\frac{\Phi(v)}{n})).$$

The following theorem allows us to obtain the estimation of Ψ after a μ update.

Theorem 3.2.1 *Let* $0 \leq \theta < 1$, $v+ = v/\sqrt{(1-\theta)}$, *if* $\Phi(v) \leq \tau$, *then we have*

$$\Phi(v_+) \leq \frac{\theta n + 2\tau + 2\sqrt{2\tau n}}{2(1-\theta)}.$$

3.3 Newton direction

To facilitate the study of these methods, we need to define the vector v as follows:

$$v = \sqrt{\frac{xs}{\mu}}, xs = (x_1 s_1, x_2 s_2, ..., x_n s_n), \tag{3.23}$$

where v is called "the scaling vector." Using 3.23, the system 3.1 can be rewritten as follows:

$$\begin{cases} \overline{A} d_x = 0, \\ \overline{A^T} \Delta y + d_s = 0, \\ d_x + d_s = v^{-1} - v, \end{cases} \tag{3.24}$$

with

$$\overline{A} = \frac{1}{\mu} A V^{-1} X, V = diag(v), X = diag(x),$$

and

$$d_x = \frac{v\Delta x}{x}, d_s = \frac{v\Delta s}{s}, \tag{3.25}$$

d_x and d_s are called "the scaled Newton directions".

Note that the right-hand side of the third equation in 3.24 is equal to the negative

gradient of the logarithmic barrier function $\Phi(v)$, i.e.,

$$d_x + d_s = -\nabla\Phi(v).$$

In this case, 3.24 becomes:

$$\begin{cases} \overline{A}d_x = 0, \\ \overline{A^T}\Delta y + d_s = 0, \\ d_x + d_s = -\nabla\Phi(v), \end{cases} \tag{3.26}$$

Where the barrier function $\Phi(v)$ is defined from $\mathbb{R}^n_{++}$ to $\mathbb{R}^n_+$ by

$$\Phi(v) = \Phi(x, s; \mu) = \sum_{i=1}^{n} \Psi_c(v_i), \tag{3.27}$$

where Ψ_c is called the classic kernel function of the logarithmic barrier function $\Phi(v)$, defined by

$$\Psi_c(t) = \frac{t_2 - 1}{2} - \log t.$$

Remark 3.3.1 *If $(d_x, \Delta y, d_s)$ is a solution of the system 3.26, then d_x and d_s are orthogonal.*

3.4 Generic algorithm

In this section, we replace the kernel function f of the logarithmic barrier function Φ with a new function u, which is defined in the following part. Let's assume that $\tau \geq 1$. Then the algorithm proceeds as follows:

Assume we are given a strictly feasible point (x, y, s) that is in a $\tau-$neighborhood of a $\mu-$center. Then, we decrease μ by a geometric sequence with ratio $(1 - \theta)$, for $0 < \theta < 1$, and then solve the Newton system to obtain the descent direction.

The positivity condition of a new iteration is ensured with the right choice of step size, which is defined in the following part. This procedure is repeated until we find a new iteration (x^+, y^+, s^+) that is in a $\tau-$neighborhood of the μ^+-center. This process is repeated until ,i.e., $n\mu \leq \varepsilon$.

The parameters τ, θ, and the step size α must be chosen such that the algorithm is optimized in the sense that the number of required iterations is minimized.

The choice of parameter θ plays a very important role in the theory and practice of **IPMs.** If θ is a constant independent of the problem's dimension n, then we call the algorithm a large step algorithm. And if θ depends on the problem's dimension, such as $\theta = \frac{1}{\sqrt{n}}$ then the algorithm is called a small step algorithm.

Algorithm The generic Primal-dual IPMs algorithm for linear optimization is given as follows:

Initialization:

A threshold parameter $0 < \tau < 1$;

A precision parameter $\varepsilon > 0$;

A barrier parameter θ $0 < \theta < 1$;

A proximity function $\Phi(v)$;

Begin:

Let (x^0, y^0, s^0) satisfy the **IPC** (an initial feasible solution) such as:

$k = 0; \mu^0 = 1; v^0 = \sqrt{\frac{x^0 s^0}{\mu^0}}; \Phi(x^0, y^0, s^0) \leq \tau;$

While: $n\mu > \varepsilon$ do:

$\mu^{k+1} = (1 - \theta)\mu^k$ (outer iteration)

While: $\Phi(v) > \tau$ do:

(inner iteration)

Solve the system 3.26 to determine $(d_x, \Delta y, \Delta s)$, then $(\Delta x, \Delta y, \Delta s)$;

Determine the feasible step size α;

Update:

$$x = x + \alpha\Delta x, y = y + \alpha\Delta y, s = s + \alpha\Delta s,$$
$$v = \sqrt{\frac{xs}{\mu}}.$$

End While (inner iteration)

End While (outer iteration)

End Algorithm.

Conclusion

Interior Point Methods for linear optimization problems have emerged as a powerful tool in the field of optimization, offering efficient solutions to complex linear programming problems. The historical development of IPMs showcases their evolution from theoretical concepts to practical applications, highlighting their significance in modern optimization theory.

This thesis has contributed to the understanding and advancement of IPMs for linear optimization problems by providing a comprehensive overview of their theoretical foundations and practical implementations. The exploration of convex analysis, optimization concepts, and the simplex method has laid a solid foundation for understanding IPMs. The detailed discussion of the primal, dual, and primal-dual methods has demonstrated the effectiveness of IPMs in solving linear optimization problems efficiently.

Bibliography

[1] Bai, Y.Q., El Ghami, M., Roos, C. (2004). A comparative study of kernel functions for primal-dual interior point algorithms in linear optimization. SIAM. J. Optim. 15(1), 101 128

[2] Bouaafia, M., Benterki, D., Yassine, A., (2016). An Efficient Primal-Dual Interior Point Methode For Linear Programing Based on a New Kernel Function with a Trigonome tric Barrier Term.

[3] Cho, G.M. (2011). An interior point algorithm for linear optimization based on a new barrier function. Appl. Math. Comput. 218, 386 395

[4] Gordon.G. Convexity - 10-725: Optimization Fall 2012 Lecture 3 - Studocu.

[5] Robert, M., Freund and Jorge V, (2014) Interior-Point Methods for Linear Optimization.

[6] Kenioua, I. and Menghour, I., (2018). La Méthode De Point Intérieur Pour La Programmation Linéaire Basée Sur Une Nouvelle Fonction Noyau. Master's thesis of University of Mohammed Seddik Ben Yahia-Jijel.

[7] TERLAKY, TamÃ¡s (ed.). Interior point methods of mathematical programming. Springer Science and Business Media, 2013.

Annexe: Abreviations et Notations

ε	:	A precision parameter.
θ	:	A barrier parameter.
$\Phi(v)$	:	A barrier function.
τ	:	A threshold parameter.
Ψ_c	:	The classic kernel function
d_x, d_s	:	The scaled Newton directions.
σ	:	The inverse function of the function Ψ.
KKT	:	The Karush-Kuhn-Tucker
IPC	:	Interior point condition
LP	:	Linear programming
IPMs	:	Interior point methods
$Hess$	:	The hessian matrix

ABSTRACT

The dissertation addresses the importance of interior point methods in improving the performance of algorithms for solving linear optimization problems, as it highlights their development from theoretical concepts to practical applications and provides a comprehensive overview of the theoretical and applied foundations of these methods, confirming their efficiency in complex linear optimization problems, as it establishes a strong understanding of interior point methods. By initially addressing the concepts of convex analysis and the concepts of optimization and its types, and in the end, it sheds light on their development through central paths and Newtonian trends using the kernel function to improve its performance.

ملخص

تتناول المذكرة أهمية طرق النقطة الداخلية في تحسين أداء خوارزميات حل مسائل التحسين الخطية، حيث تبرز تطورها من المفاهيم النظرية الى التطبيقات العملية و تقدم نظرة شاملة على الأسس النظرية و التطبيقية لهذه الطرق مؤكدة كفاءتها في مسائل التحسين الخطي المعقدة، حيث ترسخ فهما قويا لطرق النقطة الداخلية من خلال التطرق في البداية الى مفاهيم التحليل المحدب و مفاهيم التحسين و أنواعه و في الختام تسلط الضوء على تطورها عن طريق المسارات المركزية و اتجاهات نيوتن باستخدام الدالة النواة في تحسين أدائها

.

Résumé

Le mémoire aborde l'importance des méthodes de points intérieurs dans l'amélioration des performances des algorithmes de résolution de problèmes d'optimisation linéaire, car elle met en évidence leur développement des concepts théoriques aux applications pratiques et fournit un aperçu complet des fondements théoriques et appliqués de ces méthodes, confirmant leur efficacité dans des problèmes d'optimisation linéaire complexes, car il établit une solide compréhension des méthodes de points intérieurs. En abordant dans un premier temps les concepts d'analyse convexe et les concepts d'optimisation et ses types, et au final, il met en lumière leur développement à travers des chemins centraux et des tendances newtoniennes en utilisant la fonction noyau pour améliorer ses performances.