
Università degli Studi di Padova
Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea Magistrale in Matematica

# New Active-Set Frank-Wolfe Variants for Saddle Point Problems

**Relatore:**
Professor Francesco Rinaldi

**Candidato:**
Francesco Bortolon
Numero di matricola:
1132466

15 Dicembre 2017 - A. A. 2016/2017

# Contents

# Summary

Let us give a brief summary of the contents of this thesis.

First of all, in Chapter 1, we define the saddle point problem

$$\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathscr{L}(x,y) \tag{1}$$

where $\mathscr{L}$ is a convex-concave function on the convex sets $\mathcal{X}$ and $\mathcal{Y}$ and we underline how these problems are very widespread in many fields of application (such as game theory, robust optimization, ...)

Then, we give a brief description of the KKT necessary conditions for a minimization problem and we extend them to our saddle point framework, resulting in particular in the following condition

**Theorem 0.1.** *Let $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ be bounded closed, convex sets and $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a convex-concave continuously differentiable function. Thus, a point $(x^*, y^*)$ such that*

$$\begin{pmatrix} \nabla_x \mathscr{L}(x^*,y^*) \\ -\nabla_y \mathscr{L}(x^*,y^*) \end{pmatrix} = 0 \tag{2}$$

*is a saddle point for problem* (1).

After that, we describe the Frank-Wolfe algorithm and its variants (away-step FW and pairwise FW) for the generic problem of minimizing a convex function over a convex set and we analyze the extensions of these algorithms to the saddle point problems given by G. Gidel, T. Jebara and S. Lacoste-Juliene, in their quite recent paper [7].

In Chapter 2, we describe the main ideas of the new active-set strategy proposed by A. Cristofari, M. De Santis, S. Lucidi and F. Rinaldi in their very recent paper [4]. Briefly, they address to the problem

$$\min_{x\in\Delta^1} f(x) \tag{3}$$

where $f$ is a convex function and $\Delta^1$ denotes the unit simplex. In particular, they are interested in the case when the solution of this problem is very sparse. Thus, the aim of the active-set strategy is to estimate as soon as

possible the zero components (called *active* components) of the solution, in order to put them to zero and so reduce the feasible set dimension and consequently the CPU-time of a FW algorithm on that set.

Then, we focus on problem (1), where $\mathcal{X} = \Delta_x^1$ and $\mathcal{Y} = \Delta_y^1$ and where we assume that the saddle point is very sparse. Combining the active-set strategy to the saddle point extension of Frank-Wolfe methodology, we describe a new personal algorithm to solve this problem and we prove its convergence under some strong assumptions.

Finally, in Chapter 3, we implement our algorithm in Python language and we test it on the Toy Problem

$$\mathscr{L}(x, y) = \frac{\mu}{2}\|x - x^*\|_2^2 + (x - x^*)^\top M(y - y^*) - \frac{\mu}{2}\|y - y^*\|_2^2 \qquad (4)$$

in which $(x^*, y^*)$ is the explicit saddle point that we consider very sparse. We show that, as we expected, the proposed active-set strategy allows us to significantly improve the convergence performance of the Frank-Wolfe methodology to the saddle point.

# Chapter 1

# Saddle Point Problems

In this first chapter, we write an introduction about Saddle Point problems. First, we give some reasons for this study with examples of applications in different applied fields. Then, we give some fundamental definitions about saddle point problems and we analyze some methodologies to solve them. In particular, we describe in detail the methods deriving from extensions and variants of the Frank-Wolfe algorithm. Finally, we show briefly some other possible approaches.

## 1.1 Motivations

Saddle point problems arise in a wide variety of applications in many different fields and we may find many different approaches to their solution in literature.

To highlight the vastness and variety of fields where these problems arise, we show some examples.

#### Game Theory

Let us define a two-player non-cooperative game as a triple $(\mathcal{X}, \mathcal{Y}, \mathscr{L})$, where $\mathcal{X}, \mathcal{Y}$ are the spaces of strategies for the two players respectively and $\mathscr{L}$ is a real-valued pay-off function of $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. We recall that a strategy is a Nash equilibrium if no player can do better by unilaterally changing his or her strategy.

The optimal strategy in the Nash equilibrium sense, is given by the solution of the following saddle point problem

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) = \max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \mathscr{L}(x, y) \tag{1.1}$$

**Robust Optimization**

The saddle point problems arise also in worst-case scenario framework. These problems involve looking for a configuration that minimize an objective function under the hypothesis that the worst-case scenario occurs.

For example, consider the following engineering design problem. We have to produce electronic parts within certain tolerances and we are interested in minimizing the error in manufacture. More precisely, suppose that when we specify a state $x \in \mathcal{X}$, the process actually gives a state $x + y$ for some $y \in \mathcal{Y}$. Now, assume that the function $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ measures the resulting distortion in the final product. Then, from the worst-case distorsion minimization, we derive the following problem

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) \tag{1.2}$$

**Chebyshev approximation**

Given a function $f : \mathcal{X} \subseteq \mathbb{R}^m \to \mathbb{R}$ and a function space $P_n$ of functions $p : \mathbb{R}^m \to \mathbb{R}$, we define the Chebyshev approximation $\bar{p}$ of $f$ in the space $P_n$ as the solution of the following minimax problem

$$\min_{p \in P_n} \max_{x \in \mathcal{X}} (f(x) - p(x))^2 \tag{1.3}$$

**Standard Convex Programming**

The standard convex programming problem can also be characterized in terms of saddle point problem.

For example, consider the problem

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \end{aligned} \tag{1.4}$$

where $f, g$ are convex functions. We can define the Lagrangian function $\mathscr{L}$ of the above convex problem as

$$\mathscr{L}(x, \lambda) := f(x) + \lambda g(x) \tag{1.5}$$

Then, $\mathscr{L}$ is a convex-concave function on the feasible set $\mathcal{X} \times \mathcal{Y}$, where

$$\mathcal{X} = \{x : g(x) \leq 0\} \tag{1.6}$$

$$\mathcal{Y} = \{\lambda \geq 0\} \tag{1.7}$$

Under some further hypothesis, by Karush-Kuhn-Tucker theory (that we will develop later in our work), we have that solving (1.4) is equivalent to find $(x^*, \lambda^*)$, which is the saddle point of the problem

$$\min_{x \in \mathcal{X}} \max_{\lambda \in \mathcal{Y}} \mathscr{L}(x, \lambda) \tag{1.8}$$

## 1.2 Problem Analysis

First, let us give some formal definitions.

**Definition 1.1** (Saddle Point Problem)**.** Let $\mathscr{L}$ be a smooth (with $L-$Lipschitz continuous gradient) convex-concave function, that is

- $\mathscr{L}(\cdot, y)$ is convex for all $y \in \mathcal{Y}$
- $\mathscr{L}(x, \cdot)$ is concave for all $x \in \mathcal{X}$

Furthermore, let $\mathcal{X} \times \mathcal{Y}$ be a convex compact set.

Then, we define the *saddle point problem* as

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) \tag{1.9}$$

Moreover, any point $(x^*, y^*) \in \mathcal{X} \times \mathcal{Y}$ that solves (1.9) is said to be a *saddle point*. Equivalently, $(x^*, y^*) \in \mathcal{X} \times \mathcal{Y}$ is a saddle point if and only if

$$\mathscr{L}(x^*, y) \leq \mathscr{L}(x^*, y^*) \leq \mathscr{L}(x, y^*) \tag{1.10}$$

for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

The smoothness of the objective function $\mathscr{L}$ and its convex-concave behaviour on the compact feasible set $\mathcal{X} \times \mathcal{Y}$ guarantee that there exists at least one saddle point solution for problem (1.9).

To find such a solution, we introduce some optimal conditions for constrained problems.

Let us consider the non-linear problem

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0 \\ & g(x) \leq 0 \end{aligned} \tag{1.11}$$

### 1.2.1 Fritz-John Conditions

Fritz-John conditions (FJ) represent one of the first optimal conditions for non-linear programming and were derived by Lagrangian multipliers theory.

**Definition 1.2** (Lagrangian Function)**.** We define the *Lagrangian function* relative to the non-linear problem above as

$$L(x, \lambda_0, \lambda, \mu) = \lambda_0 f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{i=1}^{p} \mu_i h_i(x)$$

where we have considered $\lambda = (\lambda_1, \ldots, \lambda_m)$ and $\mu = (\mu_1, \ldots, \mu_p)$. The variables $\lambda_0$, $\lambda$ and $\mu$ are called *generalized Lagrangian multipliers*.

Now, we are able to give the Fritz-John conditions.

**Theorem 1.1** (Fritz-John Necessary Conditions)**.** *Let $x^*$ be the local minimum of the non-linear constrained problem* (1.11) *and suppose that the functions $f$, $g$ and $h$ are continuously differentiable in some neighborhood of the point $x^*$. Then, there exist multipliers ${\lambda_0}^* \in \mathbb{R}$, $\lambda^* \in \mathbb{R}^m$ and $\mu^* \in \mathbb{R}^p$ such that*

*(a) the following conditions hold*

$$\begin{aligned} {\lambda_0}^* \nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) + \sum_{i=1}^{p} \mu_i^* \nabla h_i(x^*) = 0 \\ \lambda_i^* g_i(x^*) = 0 \qquad i = 1, \dots, m \\ ({\lambda_0}^*, \lambda^*) \geq 0 \qquad ({\lambda_0}^*, \lambda^*, \mu^*) \neq 0 \\ g(x^*) \leq 0 \qquad h(x^*) = 0 \end{aligned}$$

*(b) in each neighborhood of $x^*$ there exists a point $x$ such that*

$$\begin{aligned} \lambda_i^* g_i(x) > 0 \quad \textit{for all } i \textit{ such that } \lambda_i^* > 0 \\ \mu_i^* h_i(x) > 0 \quad \textit{for all } i \textit{ such that } \mu_i^* \neq 0 \end{aligned}$$

**Remark 1.2.** If we put $\nabla g = (\nabla g_1 \dots \nabla g_m)$ and $\nabla h = (\nabla h_1 \dots \nabla h_p)$ Fritz-John Conditions can be formulated in the following matrix notation

$$\begin{aligned} {\lambda_0}^* \nabla f(x^*) + \nabla g(x^*) \lambda^* + \nabla h(x^*) \mu^* = 0 \\ (\lambda^*)^\top g(x^*) = 0 \\ ({\lambda_0}^*, \lambda^*) \geq 0 \qquad ({\lambda_0}^*, \lambda^*, \mu^*) \neq 0 \\ g(x^*) \leq 0 \qquad h(x^*) = 0 \end{aligned}$$

**Remark 1.3.** Moreover, if $\nabla_x L(x, \lambda_0, \lambda, \mu)$ denotes the gradient of the Lagrangian function with respect to the variable $x$, Fritz-John Conditions can be rewritten in the equivalent form

$$\begin{aligned} \nabla_x L(x^*, {\lambda_0}^*, \lambda^*, \mu^*) = 0 \\ (\lambda^*)^\top g(x^*) = 0 \\ ({\lambda_0}^*, \lambda^*) \geq 0 \qquad ({\lambda_0}^*, \lambda^*, \mu^*) \neq 0 \\ g(x^*) \leq 0 \qquad h(x^*) = 0 \end{aligned}$$

### 1.2.2 Constraint Qualifications

When we consider the Fritz-John first condition, the gradient of the objective function is weighted with the coefficient $\lambda_0 \geq 0$.

Clearly, if $\lambda_0 = 0$, the objective function has no role in the optimality conditions and this make them meaningless. For this reason, we are interested in identifying when $\lambda_0 > 0$.

The assumptions that we take over constraints to ensure $\lambda_0 > 0$ are called *constraint qualification conditions*.

To describe them, we need some definitions that extend the properties of convexity in a local context.

**Definition 1.3.** Let $\bar{x} \in \mathbb{R}^m$ and let $g$ be a function which is continuously differentiable in a neighborhood $B(\bar{x}, \rho)$ of $\bar{x}$. We say that

(i) $g$ is convex in $\bar{x}$ if $g(x) \geq g(\bar{x}) + \nabla g(\bar{x})^\top (x - \bar{x})$ for all $x \in B(\bar{x}, \rho)$;

(ii) $g$ is strictly convex if
$g(x) > g(\bar{x}) + \nabla g(\bar{x})^\top (x - \bar{x})$ for all $x \in B(\bar{x}, \rho)$, $x \neq \bar{x}$

(iii) $g$ is concave in $\bar{x}$ if $g(x) \leq g(\bar{x}) + \nabla g(\bar{x})^\top (x - \bar{x})$ for all $x \in B(\bar{x}, \rho)$;

(iv) $g$ is strictly concave if
$g(x) < g(\bar{x}) + \nabla g(\bar{x})^\top (x - \bar{x})$ for all $x \in B(\bar{x}, \rho)$, $x \neq \bar{x}$

In the following, we assume that $x^*$ is a point that satisfy the Fritz-John conditions and we denote with $I(x^*)$ the set of inequalities indexes that are active in $x^*$, that is

$$I(x^*) = \{i : g_i(x^*) = 0\}$$

Now, we give four different conditions that ensure $\lambda_0 > 0$.

### (a) Linear independence constraint qualification

The gradients of the active inequality constraints and the gradients of the equality constraints are linearly independent at $x^*$, that is the set

$$\{\nabla h_i(x^*),\ i = 1, \ldots, p \qquad \nabla g_i(x^*),\ i \in I(x^*)\}$$

is linear independent.

### (b) Linearity and Concavity constraint qualification

The equality constraints are linear and the inequality active constraints are concave at $x^*$.

### (c) Mangasarian-Fromovitz Condition

The gradients of the equality constraints are linearly independent at $x^*$ and there exists a vector $d \in \mathbb{R}^n$ such that $\nabla g_i(x^*)^\top d < 0$ for all active inequality constraints and $\nabla h_j(x^*)^\top d = 0$ for all equality constraints.

**(d) Slater Condition**

Suppose that we have only convex inequalities constraints. Then, there exists a point $x$ such that $g_i(x) < 0$.

### 1.2.3 KKT Conditions

Provided that one of the above conditions hold, we get $\lambda_0 > 0$. The Karush-Kuhn-Tucker necessary conditions for optimality are derived from Fritz-John ones simply dividing the first equation by $\lambda_0$.

**Theorem 1.4** (KKT Necessary Conditions)**.** *Let $x^*$ be the local minimum of the non-linear constrained problem* (1.11) *and suppose that the functions $f$, $g$ and $h$ are continuously differentiable in some neighborhood of the point $x^*$.*

*Moreover, suppose that at least one of the four conditions (a), (b), (c), (d) above is satisfied in $x^*$.*

*Then, there exist multipliers $\lambda^* \in \mathbb{R}^m$ and $\mu^* \in \mathbb{R}^p$ such that*

$$\begin{aligned} \nabla f(x^*) + \nabla g(x^*)\lambda^* + \nabla h(x^*)\mu^* &= 0 \\ g(x^*) \leq 0 \qquad h(x^*) &= 0 \\ (\lambda^*)^\top g(x^*) &= 0 \\ \lambda^* &\geq 0 \end{aligned}$$

**Remark 1.5.** Let $L(x, \lambda, \mu) = f(x) + \lambda^\top g(x) + \mu^\top h(x)$ be the Lagrangian function for the problem (1.11). Then, we get

$$\nabla_x L(x, \lambda, \mu) = \nabla f(x) + \nabla g(x)\lambda + \nabla h(x)\mu$$

Then, KKT conditions can be reformulated in this way:

$$\begin{aligned} \nabla_x L(x^*, \lambda^*, \mu^*) &= 0 \\ g(x^*) \leq 0 \qquad h(x^*) &= 0 \\ (\lambda^*)^\top g(x^*) &= 0 \\ \lambda^* &\geq 0 \end{aligned}$$

KKT conditions for optimality are only necessary, but under some further hypothesis, they become also sufficient.

**Theorem 1.6** (KKT Sufficient Conditions)**.** *Suppose that $f$ and $g_i$, $i = 1, \ldots, m$, are convex functions and that the equality constraints are linear, that is $h(x) = Ax - b$. Moreover, suppose that $f, g, h$ are continuously differentiable in an open set that contains the feasible set.*

*Suppose that there exist multipliers $\lambda^*$ and $\mu^*$ such that the following conditions hold*

$$\begin{aligned} \nabla_x L(x^*, \lambda^*, \mu^*) &= 0 \\ g(x^*) \leq 0 \qquad h(x^*) &= 0 \\ (\lambda^*)^\top g(x^*) &= 0 \\ \lambda^* &\geq 0 \end{aligned}$$

*Thus, $x^*$ is a global constrained minimum.*

*Furthermore, if $f$ is strictly convex, then $x^*$ is the unique solution of the problem* (1.11).

We refer to [8] for a more detailed study of KKT conditions and, in particular, for the proofs of the above results.

### 1.2.4 KKT Conditions for Saddle Point Problems

Now, we extend the KKT conditions in the saddle point problems context. For this analysis, we follow in particular the ideas in [15].

First, we need a couple of classical results (see [15] for the first one and [16] for the second one).

**Lemma 1.7.** Let $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ be bounded closed sets and $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a continuous function. Then, $\mathscr{L}$ has a saddle point on $\mathcal{X} \times \mathcal{Y}$ if and only if

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) = \max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \mathscr{L}(x, y)$$

**Theorem 1.8.** *Let $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ be bounded closed sets and $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a convex-concave continuous function. Then*

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) = \max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \mathscr{L}(x, y)$$

Under the assumption of convex-concavity we have that $\mathscr{L}$ has a minimum in the first variable $x \in \mathcal{X}$ and a maximum in the second variable $y \in \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex sets. This simple remark leads us to the following sufficient condition for a saddle point solution.

**Theorem 1.9.** *Let $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ be bounded closed, convex sets and $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a convex-concave continuously differentiable function. Thus, a point $(x^*, y^*)$ such that*

$$\begin{pmatrix} \nabla_x \mathscr{L}(x^*, y^*) \\ -\nabla_y \mathscr{L}(x^*, y^*) \end{pmatrix} = 0 \tag{1.12}$$

*is a saddle point for problem*

$$\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathscr{L}(x,y)$$

To prove the theorem above, we need to recall some classical results (see for example [1]).

**Theorem 1.10.** *If $\mathcal{X}$ is a convex set and $f : \mathcal{X} \to \mathbb{R}$ is a convex function, then a local minimum over $\mathcal{X}$ is also a global minimum of $f$ over $\mathcal{X}$.*

**Proposition 1.11.** Let $\mathcal{X}$ be a convex set and $f : \mathcal{X} \to \mathbb{R}$ be a convex function. If $x^* \in \mathcal{X}$ is such that $\nabla f(x^*) = 0$, $x^*$ is a local minimum for $f$ in $\mathcal{X}$.

*Proof.* Now, let us prove Theorem 1.9. Let condition (1.12) holds.

First, from Lemma (1.7) and (1.8) we get that there exists a saddle point and we can exchange the order of minimization and maximization at will.

Now, consider the problem

$$\min_{x\in\mathcal{X}} \mathscr{L}(x,y^*) \tag{1.13}$$

The condition $\nabla_x \mathscr{L}(x^*,y^*) = 0$ and the convexity of $\mathscr{L}(\cdot,y^*)$ guarantee that $(x^*,y^*)$ is the solution of problem (1.13). In particular, it follows that

$$\mathscr{L}(x^*,y^*) \le \mathscr{L}(x,y^*) \quad \text{for all } x \in \mathcal{X} \tag{1.14}$$

On the other hand, consider the following problem

$$\max_{y\in\mathcal{Y}} \mathscr{L}(x^*,y) = -\min_{y\in\mathcal{Y}} -\mathscr{L}(x^*,y) \tag{1.15}$$

The condition $-\nabla_y \mathscr{L}(x^*,y^*) = 0$ and the convexity of $-\mathscr{L}(x^*,\cdot)$ guarantee that $(x^*,y^*)$ is the solution of problem (1.15). In particular, we get

$$\mathscr{L}(x^*,y) \le \mathscr{L}(x^*,y^*) \quad \text{for all } y \in \mathcal{Y} \tag{1.16}$$

If we put (1.14) and (1.16) together, we get

$$\mathscr{L}(x^*,y) \le \mathscr{L}(x^*,y^*) \le \mathscr{L}(x,y^*) \quad \text{for all } x \in \mathcal{X}, y \in \mathcal{Y} \tag{1.17}$$

that is $(x^*,y^*)$ is a saddle point. □

Now, consider the generic constrained min-max problem

$$\begin{aligned} \min_x \max_y \ & \mathscr{L}(x,y) \\ \text{s.t. } & g_1(x) = 0 \\ & h_1(x) \le 0 \\ & g_2(y) = 0 \\ & h_2(y) \le 0 \end{aligned}$$

We can associate to this problem the following Lagrangian function

$$L(x, y, \lambda_1, \mu_1, \lambda_2, \mu_2) = \mathscr{L}(x, y) + \lambda_1^\top g_1(x) + \mu_1^\top h_1(x) - \lambda_2^\top g_2(y) - \mu_2^\top h_2(y)$$

The KKT necessary conditions for optimality for this problem are the following:

$$\nabla_x L(x^*, y^*, \mu_1^*, \lambda_1^*, \mu_2^*, \lambda_2^*) = 0 \tag{1.18}$$
$$\nabla_y L(x^*, y^*, \mu_1^*, \lambda_1^*, \mu_2^*, \lambda_2^*) = 0 \tag{1.19}$$
$$h_1(x^*) \leq 0 \quad h_2(y) \leq 0$$
$$(\mu_1^*)^\top h_1(x^*) = 0 \quad (\mu_2^*)^\top h_2(y^*) = 0$$
$$\mu_1^* \geq 0 \quad \mu_2^* \geq 0$$
$$g_1(x^*) = 0 \quad g_2(y^*) = 0$$

where (1.18) is given by

$$\nabla_x \mathscr{L}(x^*, y^*) + \nabla g_1(x^*)^\top \lambda_1^* + \nabla h_1(x^*)^\top \mu_1^* = 0$$

and (1.19) is given by

$$\nabla_y \mathscr{L}(x^*, y^*) - \nabla g_2(y^*)^\top \lambda_2^* - \nabla h_2(y^*)^\top \mu_2^* = 0$$

In the next chapter, we describe how to solve this system in the case

$$\mathcal{X} = \Delta_x^1 \quad \text{and} \quad \mathcal{Y} = \Delta_y^1 \tag{1.20}$$

where $\Delta^1$ denotes the unit simplex.

## 1.3 Problem Methodologies

In this section, we describe some methodologies for the study of saddle point problems. In particular, we describe in detail the Frank Wolfe method and its variants (away step and pairwise). In order not to weigh the reading, we preferred to include most of the definitions and the results that prove the convergence of the algorithms in the Appendix A. Finally, we briefly describe some other possible approaches.

### 1.3.1 Frank-Wolfe algorithm

The Frank-Wolfe (FW) algorithm, also called *conditional gradient method*, is a first-order method for smooth constrained optimization over a compact set.

Consider the generic problem

$$\min_{x \in \mathcal{X}} f(x) \tag{1.21}$$

where $\mathcal{X}$ is a convex compact set and $f : \mathcal{X} \to \mathbb{R}$ is a convex differentiable function.

Now, we give a result that will be useful to understand the behaviour of Frank-Wolfe algorithm.

**Proposition 1.12.** A point $x^* \in \mathcal{X}$ is a local minimum for problem (1.21) if and only if

$$\nabla f(x)^\top (x - x^*) \geq 0 \tag{1.22}$$

for all $x \in \mathcal{X}$.

This proposition makes a fundamental relation between local minimum and absence of descent directions.

The FW algorithm, given a feasible point $x^k$, looks for a descent direction computing

$$\min_{x \in \mathcal{X}} \nabla f(x^k)^\top (x - x^k) \tag{1.23}$$

which always has a solution $\tilde{x}^k$ since we minimize a continuous function over the compact set $\mathcal{X}$.

Clearly, $\nabla f(x^k)^\top (\tilde{x}^k - x^k)$ can not be positive, since we have

$$\nabla f(x^k)^\top (x - x^k) \leq \nabla f(x^k)^\top (x^k - x^k) = 0 \tag{1.24}$$

for all $x \in \mathcal{X}$.

Thus, we may have two possible cases

- if we have
$$\nabla f(x^k)^\top (\tilde{x}^k - x^k) = 0 \tag{1.25}$$
then
$$0 = \nabla f(x^k)^\top (\tilde{x}^k - x^k) \leq \nabla f(x^k)^\top (x - x^k) \tag{1.26}$$
for all $x \in \mathcal{X}$. Hence, by Proposition (1.12), $x^k$ is a local minimum.

- if we have
$$\nabla f(x^k)^\top (\tilde{x}^k - x^k) < 0 \tag{1.27}$$
then we can choose (see Figure 1.1) the descent direction
$$d^k_{\mathrm{FW}} = \tilde{x}^k - x^k \tag{1.28}$$
and we can find a new feasible point
$$x^{k+1} = x^k + \gamma^k d^k_{\mathrm{FW}} \tag{1.29}$$
such that
$$f(x^{k+1}) < f(x^k) \tag{1.30}$$

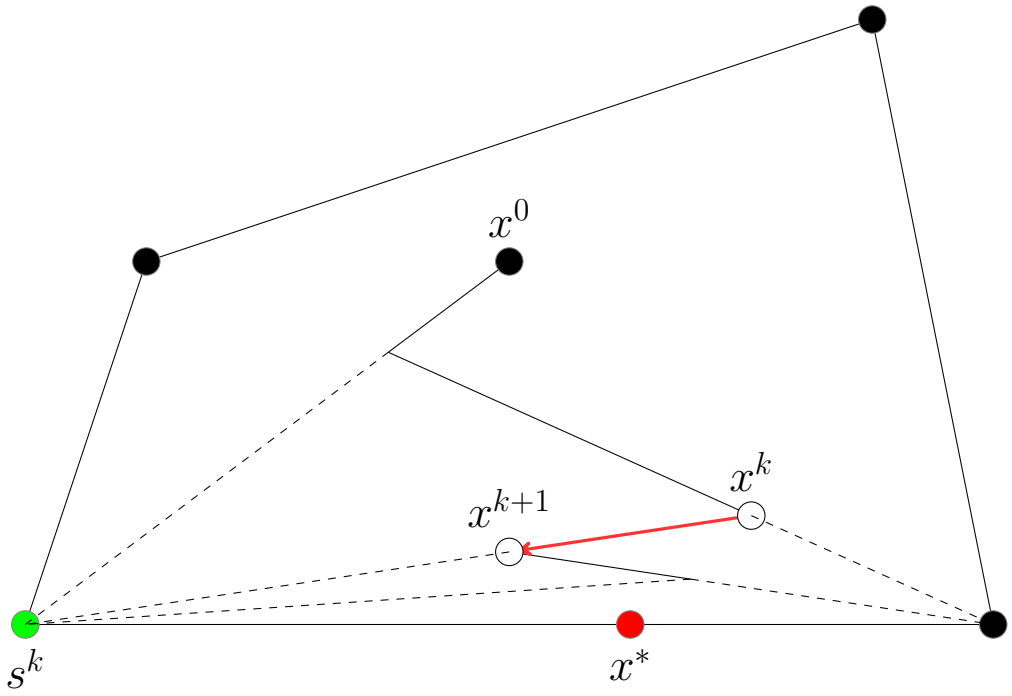



Figure 1.1: FW direction

Hence, each FW iteration consists of two parts: a first one in which we solve the simpler problem of linearizing the objective function near the current approximation of the solution and finding a descent direction; a second phase in which we find a new approximation of the solution of the original problem along the descent direction.
Clearly, FW methodology is useful when solving the linearized problem (1.23) is much easier then solving the original problem (1.21).

**Algorithm 1** Frank-Wolfe algorithm

1: Let $x^0 \in \mathcal{X}$
2: **for** $k = 0, \dots, T$ **do**
3: Compute $r^k = \nabla f(x^k)$
4: Compute $s^k := \arg\min_{s \in \mathcal{X}} < s, r^k >$
5: Compute $g^k_{\text{FW}} :=< x^k - s^k, r^k >$
6: **if** $g^k_{\text{FW}} \leq \epsilon$ **then return** $x^k$
7: **end if**
8: Let $\gamma^k = \frac{2}{2+k}$ or do line-search
9: Update $x^{k+1} := (1 - \gamma^k)x^k + \gamma^k s^k$
10: **end for**

Finally, we recall the main result about the convergence of Algorithm 1.

**Proposition 1.13.** Consider the problem (1.21) where $f \in \mathcal{C}^1(\mathbb{R}^n)$ and

$\mathcal{X} \in \mathbb{R}^n$ is a compact and convex subset. Assume that $\{x^k\}_k$ is a sequence generated by Algorithm 1 with a line search that satisfies

(i) $x^{k+1} \in \mathcal{X}$

(ii) $f(x^{k+1}) < f(x^k)$

(iii) $\lim_{k\to\infty} \nabla f(x^k)^\top d^k = 0$

Hence, we get one of the two following cases

- there exists an index $T$ such that $x^T$ is a stationary point (and so a minimum);
- the sequence $\{x^k\}_k$ is infinity and each of its limit points is stationary.

In [7], Frank-Wolfe algorithm is extended to deal with saddle point problem (1.9).
If we assume that the two sets $\mathcal{X}$ and $\mathcal{Y}$ are convex, we can extend the Algorithm 1 by simultaneously taking a FW update on both convex functions $\mathscr{L}(\cdot, y^k)$ and $-\mathscr{L}(x^k, \cdot)$.

**Algorithm 2** Saddle Point Frank-Wolfe algorithm

Let $z^0 = (x^0, y^0) \in \mathcal{X} \times \mathcal{Y}$
**for** $k = 0, \dots, T$ **do**
  Compute $r^k = \begin{pmatrix} \nabla_x \mathscr{L}(x^k, y^k) \\ -\nabla_y \mathscr{L}(x^k, y^k) \end{pmatrix}$
  Compute $s^k := \arg\min_{z \in \mathcal{X}\times\mathcal{Y}} < z, r^k >$
  Compute $g^k :=< z^k - s^k, r^k >$
  **if** $g^k \leq \epsilon$ **then return** $z^k$
  **end if**
  Let $\gamma^k = \frac{2}{2+k}$
  Update $z^{k+1} := (1 - \gamma^k) z^k + \gamma s^k$
**end for**

In [7], it is proved that Algorithm 2 converges to the saddle point solution of problem (1.9) under the following assumptions:

- $\mathcal{X}$ and $\mathcal{Y}$ are convex sets
- $\mathscr{L}$ is strongly convex-concave
- $\nabla\mathscr{L}$ is $L$-Lipschitz continuous
- $\mathscr{L}$ has a finite curvature constant $\mathcal{C}_{\mathscr{L}}$
- $\mathscr{L}$ has a positive interior strong convex-concavity constant $\mu^{\text{int}}_{\mathscr{L}}$
- the saddle point $(x^*, y^*)$ lies in the relative interior of the feasible set.

We refer to the Appendix for the details on the convergence of Algorithm 2.

### 1.3.2 Away-step Frank-Wolfe algorithm

It is well known that Frank-Wolfe algorithm may take a long time to converge when the solution lies on the boundary of the feasible set. The reason for this behaviour is due to the zig-zagging phenomenon that occurs when the algorithm is approaching the solution.
The away-step Frank-Wolfe algorithm was born to avoid this situation.
First of all, let us give some definitions.

**Definition 1.4.** Let $\mathcal{X} = \text{conv}(\mathscr{A})$ be the convex hull of a finite set $\mathscr{A}$. Then, a feasible point $x \in \mathcal{X}$ can be represented as

$$x = \sum_{v \in \mathcal{S}} \alpha_v v \tag{1.31}$$

where $\mathcal{S}$ is the set of all the vertices in $\mathscr{A}$ such that $\alpha_v > 0$.

**Definition 1.5.** Given a feasible point $x^k \in \mathcal{X} = \text{conv}(\mathscr{A})$, we shall compute the atom $v^k \in \mathcal{S}^k$ that maximizes the potential descent, given by

$$g_A^k :=< -\nabla f(x^k), x^k - v^k > \tag{1.32}$$

Hence, we can compute $v^k$ as

$$v^k \in \arg\max_{v \in \mathcal{S}^k} < \nabla f(x^k), v > \tag{1.33}$$

Then, we define the away direction as

$$d_A^k := x^k - v^k \tag{1.34}$$

Intuitively, the away direction allows to move away from a *bad* atom that keeps the gradient far from zero.
The away-step Frank-Wolfe variant chooses at each iteration if it is more useful to select the FW standerd direction or the away direction. In particular, we compute $g_A^k$ and

$$g_{\text{FW}}^k =< -\nabla f(x^k), d_{\text{FW}} > \tag{1.35}$$

and then we follow this scheme:

- if $g_{\text{FW}}^k > g_A^k$, choose the FW direction
- else, choose the away direction (see Figure 1.2).

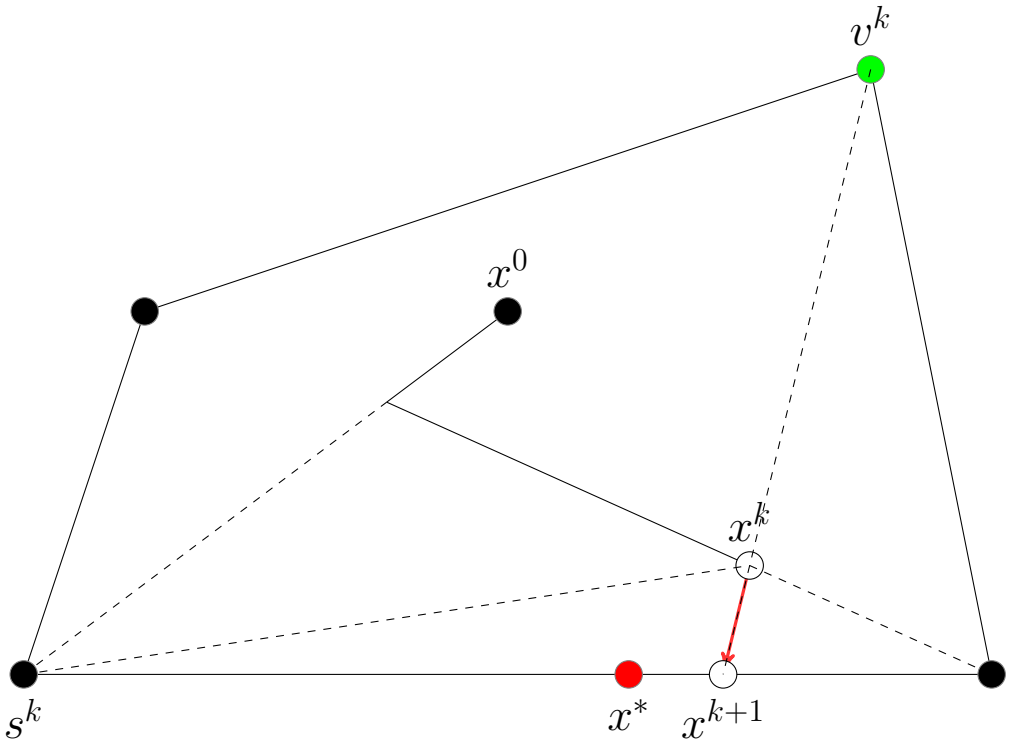



Figure 1.2: AFW direction

**Algorithm 3** Away-step Frank-Wolfe algorithm

Let $x^0 \in \mathscr{A}$, $S^0 := x^0$
**for** $k = 0, \ldots, T$ **do**
  Let $s^k = \arg\min_{s \in \mathscr{A}} < \nabla f(x^k), s >$ and $d_{\mathrm{FW}} := s^k - x^k$
  Let $v^k \in \arg\max_{v \in S^k} < \nabla f(x^k), v >$ and $d_{\mathrm{A}} := x^k - v^k$ ▷ (the away direction)
  **if** $g^k_{\mathrm{FW}} := \left\langle -\nabla f(x^k), d_{\mathrm{FW}} \right\rangle \leq \epsilon$ **then return** $z^k$
  **end if**
  **if** $\left\langle -\nabla f(x^k), d^k_{\mathrm{FW}} \right\rangle \geq \left\langle -\nabla f(x^k), d^k_{\mathrm{A}} \right\rangle$ **then** $d^k := d^k_{\mathrm{FW}}$ ▷ (choose the FW direction)
  **else**: $d^k := d^k_{\mathrm{A}}$ ▷ (choose the away-step direction)
  **end if**
  Choose $\gamma^k$
  Update $x^{k+1} := x^k + \gamma^k d^k$
  Update $S^{k+1} := \{v \in \mathscr{A} \text{ s.t. } \alpha_v^{k+1} > 0\}$
**end for**

In [13], it has been proved that Algorithm 3 is still convergent, when the objective function is strongly convex.

**Remark 1.14.** The choice of $\gamma^k$ at step 10 of Algorithm 3 has to guarantee the feasibility of the new point $x^{k+1}$. For this reason, we need to choose $\gamma^k \in (o, \gamma_{\max}]$, where $\gamma_{\max}$ is the greatest value that keeps $x^k + \gamma_{\max} d^k \in \mathcal{X}$. In particular, we have

- $\gamma_{\max} = 1$ if $d^k = d^k_{\text{FW}}$
- $\gamma_{\max} = \frac{\alpha_{v^k}}{1-\alpha_{v^k}}$ if $d^k = d^k_A$

In [7], the away-step Frank-Wolfe variant is also extended to deal with saddle point problem (1.9) when the feasible set is given by

$$\mathcal{X} \times \mathcal{Y} := \text{conv}(\mathscr{A}) \times \text{conv}(\mathscr{B}), \tag{1.36}$$

where $\mathscr{A}$ and $\mathscr{B}$ are two finite sets.

**Algorithm 4** Saddle Point away-step Frank-Wolfe algorithm

1: Let $z^0 = (x^0, y^0) \in \mathscr{A} \times \mathscr{B}$, $S^0_x := x^0$ and $S^0_y := y^0$
2: **for** $k = 0, \dots, T$ **do**
3: Compute $r^k = \begin{pmatrix} \nabla_x \mathscr{L}(x^k, y^k) \\ -\nabla_y \mathscr{L}(x^k, y^k) \end{pmatrix}$
4: Let $s^k := \arg\min_{s \in \mathscr{A} \times \mathscr{B}} < r^k, s >$ and $d_{\text{FW}} := s^k - z^k$
5: Let $v^k \in \arg\max_{v=(v_x, v_y) \in S^k_x \times S^k_y} < r^k, v >$ and $d_{\text{A}} := z^k - v^k$ ▷ (the away direction)
6: **if** $g^k_{\text{FW}} := \left\langle -r^k, d_{\text{FW}} \right\rangle \leq \epsilon$ **then return** $z^k$
7: **end if**
8: **if** $\left\langle -r^k, d^k_{\text{FW}} \right\rangle \geq \left\langle -r^k, d^k_{\text{A}} \right\rangle$ **then** $d^k := d^k_{\text{FW}}$ ▷ (choose the FW direction)
9: **else**: $d^k := d^k_{\text{A}}$ ▷ (choose the away-step direction)
10: **end if**
11: Put $\gamma^k = \max\{\gamma_{\max}, \frac{2}{2+k}\}$
12: Update $z^{k+1} := z^k + \gamma^k d^k$
13: Update $S^{k+1}_x := \{v_x \in \mathscr{A} \text{ s.t. } \alpha^{k+1}_{v_x} > 0\}$ and
14: $S^{k+1}_y := \{v_y \in \mathscr{B} \text{ s.t. } \alpha^{k+1}_{v_y} > 0\}$
15: **end for**

**Remark 1.15.** Also in this context, we have to choose $\gamma_{\max}$ in such a way that $z^{k+1}$ stays feasible. In order to keep this property, we need to choose

- $\gamma_{\max} = 1$ if $d^k := d^k_{\text{FW}}$
- $\gamma_{\max} = \min\{\frac{\alpha_{v_x}}{1-\alpha_{v_x}}, \frac{\alpha_{v_y}}{1-\alpha_{v_y}}\}$ if $d^k := d^k_{\text{A}}$

As for the standard case, this different approach reduces the zig-zagging phenomenon.
Moreover, it has been proved ([7]) that Algorithm 4 converges to the saddle point solution of problem (1.9), provided that

- $\mathcal{X} \times \mathcal{Y} := \text{conv}(\mathscr{A}) \times \text{conv}(\mathscr{B})$, that is $\mathcal{X}$ and $\mathcal{Y}$ are polytopes
- $\mathscr{L}$ is strongly convex-concave
- $\nabla\mathscr{L}$ is $L$-Lipschitz continuous
- $\mathscr{L}$ has a finite curvature constant $\mathcal{C}_{\mathscr{L}}$
- $\mathscr{L}$ has a positive geometric strong convex-concavity constant $\mu^A_{\mathscr{L}}$

where the quantity $\mu^A_{\mathscr{L}}$ has a quite complicated definition, but gives in this polytopes context the same informations that the interior strong convex-concavity constant described in Appendix A gives for the case in which the saddle point is in the relative interior of the feasible set.
We refer to [7] for formal definitions and proofs.

### 1.3.3 Pairwise Frank-Wolfe algorithm

The last variant of Frank-Wolfe algorithm that we analyze is the so called *pairwise Frank-Wolfe variant.*
As in the away-step FW variant, also in this case we consider the problem of minimizing a convex function over a polytope.
The main idea behind this different approach is to only move weight mass between two atoms at each step. In particular, we move mass from the *bad* away atom to the *good* Frank-Wolfe atom.
More precisely, let $x^k$ be the feasible point that we have found in the last iteration. Then, in the same way we do in Algorithm 3, we compute the Frank-Wolfe atom

$$s^k \in \arg\min_{s \in \mathscr{A}} < \nabla f(x^k), s > \tag{1.37}$$

and the away atom

$$v^k \in \arg\max_{v \in S^k} < \nabla f(x^k), v > \tag{1.38}$$

Hence, we define the Pairwise direction (see Figure 1.3) as

$$d^k_{\text{PFW}} := s^k - v^k \tag{1.39}$$

e then we take

$$x^{k+1} = x^k + \gamma^k d^k_{\text{PFW}} \tag{1.40}$$

In practise, this algorithm seems to perform quite well, often outperforming away-step FW, especially in the important case of sparse solutions.
Indeed, while Algorithm 3 moves the mass onto all other active atoms (atoms $v$ such that $\alpha_v > 0$ in representation (1.31)) and might require more corrections later, the PFW step only moves the mass onto one direction (the FW one).

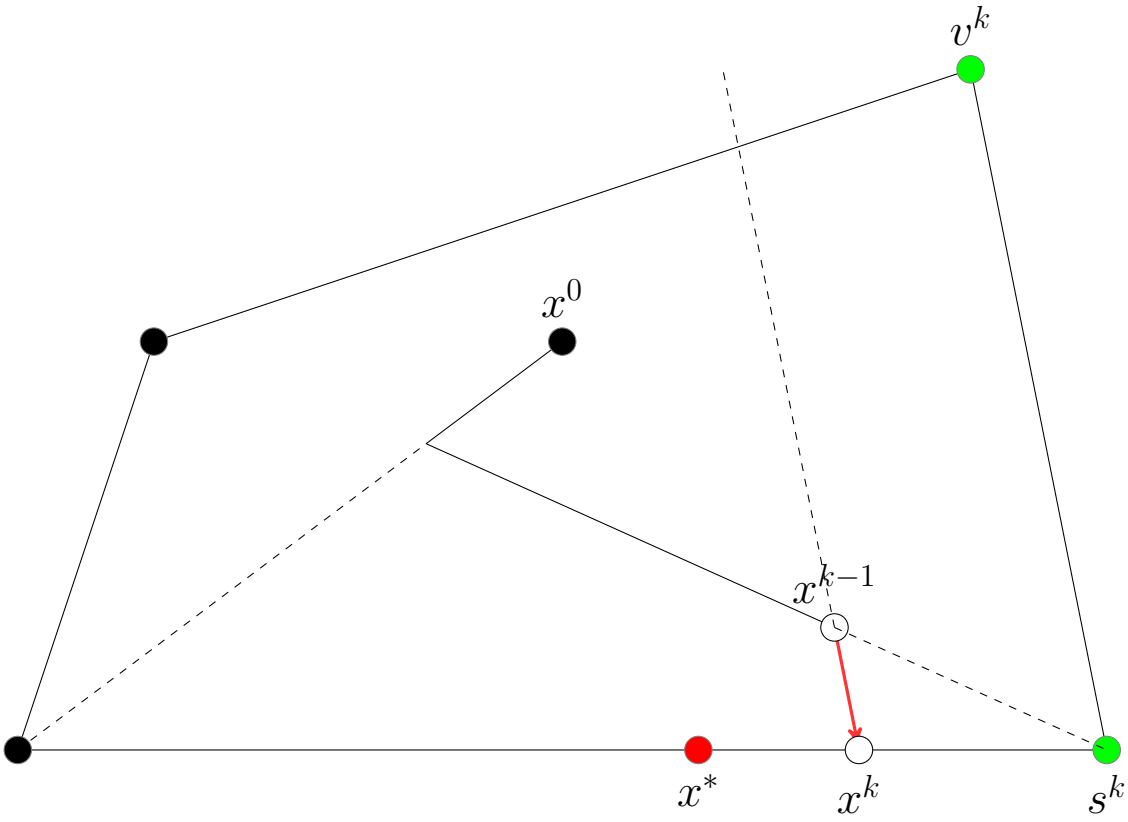



Figure 1.3: PFW direction

**Algorithm 5** Pairwise Frank-Wolfe algorithm

1: Let $x^0 \in \mathscr{A}$ and $S_x^0 := x^0$
2: **for** $k = 0, \ldots, T$ **do**
3: Let $s^k := \arg\min_{\mathscr{A}} < \nabla f(x^k), s >$ and $d_{\text{FW}} := s^k - x^k$
4: Let $v^k \in \arg\max_{v \in S^k} < \nabla f(x^k), v >$
5: **if** $g_{\text{FW}}^k := \left< -\nabla f(x^k), d_{\text{FW}} \right> \leq \epsilon$ **then return** $x^k$
6: **end if**
7: Let $d^k := d_{\text{PFW}}^k := s^k - v^k$
8: Choose $\gamma^k$
9: Update $x^{k+1} := x^k + \gamma^k d^k$
10: Update $S^{k+1} := \{v \in \mathscr{A} \text{ s.t. } \alpha_v^{k+1} > 0\}$
11: **end for**

**Remark 1.16.** In step 8 of Algorithm 5, we have to choose $\gamma^k \in (0, \gamma_{\max}]$ where

$$\gamma_{\max} = \alpha_v \tag{1.41}$$

is the largest value that keeps $x^k + \gamma_{\max} d_{\text{PFW}}^k$ still feasible.

Also in this case, in [7] is given an extension of this algorithm to solve the saddle point-problem (1.9).

**Algorithm 6** Saddle Point pairwise Frank-Wolfe algorithm

Let $z^0 = (x^0, y^0) \in \mathscr{A} \times \mathscr{B}$, $S_x^0 := x^0$ and $S_y^0 := y^0$
**for** $k = 0, \dots, T$ **do**
  Compute $r^k = \begin{pmatrix} \nabla_x \mathscr{L}(x^k, y^k) \\ -\nabla_y \mathscr{L}(x^k, y^k) \end{pmatrix}$
  Let $s^k := \arg\min_{s \in \mathscr{A} \times \mathscr{B}} < r^k, s >$ and $d_{\text{FW}} := s^k - z^k$
  Let $v^k \in \arg\max_{v=(v_x, v_y) \in S_x^k \times S_y^k} < r^k, v >$
  **if** $g_{\text{FW}}^k := \left\langle -r^k, d_{\text{FW}} \right\rangle \leq \epsilon$ **then return** $z^k$
  **end if**
  Let $d^k := d_{\text{PFW}}^k := s^k - v^k$
  Choose $\gamma^k$
  Update $z^{k+1} := z^k + \gamma^k d^k$
  Update $S_x^{k+1} := \{v_x \in \mathscr{A} \text{ s.t. } \alpha_{v_x}^{k+1} > 0\}$ and $S_y^{k+1} := \{v_y \in \mathscr{B} \text{ s.t. } \alpha_{v_y}^{k+1} > 0\}$
**end for**

Now, $\gamma^k$ has to be chosen in the interval $(0, \gamma_{\max}]$, where

$$\gamma_{\max} := \min\{\alpha_{v_x}, \alpha_{v_y}\} \tag{1.42}$$

### 1.3.4 Other approaches

For completeness, we also describe some other approaches that are used to study the saddle point problem (1.9).

#### Extragradient Method

First, this problem can be considered as a special case of the more general *variational inequality problem* (VIP), that is the problem of finding

$$z^* \in \mathcal{Z} \quad \text{such that} \quad < r(z^*), z - z^* > \geq 0 \quad \text{for all } z \in Z \tag{1.43}$$

where $r$ is a Lipschitz mapping from $\mathbb{R}^p$ to itself and $Z \subseteq \mathbb{R}^p$.
Clearly, if we take $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ and $r(z) =< \nabla_x \mathscr{L}(z), -\nabla_y \mathscr{L}(z) >$, then VIP reduces to the equivalent optimality conditions for the saddle point problem (1.9).

**Definition 1.6.** Let $P_{\mathcal{Z}}(\cdot)$ denote the orthogonal projection from $\mathbb{R}^p$ into $\mathcal{Z}$, that is

$$P_{\mathcal{Z}}(w) := \arg\min\{\|z - w\| \text{ such that } z \in \mathcal{Z}\}, \ w \in \mathbb{R}^p \tag{1.44}$$

Then, the *extragradient algorithm* generates the following iteration:

$$\tilde{z}^k = P_{\mathcal{Z}}(z^k - \alpha r(z^k)) \tag{1.45}$$

$$z^{k+1} = P_{\mathcal{Z}}(z^k - \alpha r(\tilde{z}^k)) \tag{1.46}$$

where $\alpha > 0$ is a fixed constant. Briefly, taking the projection two times makes this algorithm more efficient and it can be proved that such scheme is convergent if $r$ is monotone and Lipschitz continuous on $\mathcal{Z}$, provided the number $\alpha$ is sufficiently small.
However, this algorithm can be interesting only if the computation of the projection into $\mathcal{Z}$ is rather simple.
We refer to [17] for more details.

### Hammond's Generalized Frank-Wolfe Method

J.H. Hammond, in her PhD Thesis ([9]), describes a generalized Frank-Wolfe method to solve the following variational inequality problem directly.
Consider the problem of finding $x^* \in \mathcal{X}$ satisfying

$$(x - x^*)^\top f(x^*) \quad \text{for all } x \in \mathcal{X} \tag{1.47}$$

where $f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable and strictly monotone and $\mathcal{X}$ is a bounded polyhedron.
Then, Hammond proves that if $f$ is a gradient mapping, that is $f(x) = \nabla F(x)^\top$ for some strictly convex functional $F$, the following scheme converges to the solution of problem (1.47).

**Algorithm 7** Hammond's Generalized Frank-Wolfe algorithm

Let $x^0 \in \mathcal{X}$
**for** $k = 0, \dots, T$ **do**
  Compute $v^k := \arg\min_{x \in \mathcal{X}} x^\top f(x^k)$
  **if** $x^\top f(x^k) = v^\top f(x^k)$ **then return** $x^k$
  **end if**
  Find $\omega^k \in [0, 1]$ s.t.
$[(1-\omega)x^k + \omega v^k] - [(1-\omega^k)x^k + \omega^k v^k]^\top f[(1-\omega^k)x^k + \omega^k v^k] \geq 0$
    for every $\omega \in [0, 1]$
  Update $x^{k+1} := (1 - \omega^k)x^k + \omega^k v^k$
**end for**

## 1.3.5 Bilinear Saddle Point Problems on Domanins Given by Linear Minimization Oracles

In [3] and [11], a further approach is proposed. They study convex-concave saddle point problems on a convex domain $\mathcal{X}$ represented by Linear Minimization Oracle(LMO), that is a function that allows to minimize over $\mathcal{X}$ any

linear function. The property of LMO-representability of a convex domain is a weaker assumption than *proximal friendliness*, that is the possibility of minimize over such domain any linear perturbation of a strongly convex function, that fits very well with first order algorithms, such as FW one.
Their method consists in using Fenchel representation of the objective function, in order to pass from the original problem to its special dual. Indeed, in many important cases, such a dual problem has a proximal friendly domain. Thus, we can solve the dual problem and then build from its solution an approximate solution for the original problem.
In particular, they deal with the convex-concave saddle point problem

$$\min_{w \in W} \max_{z \in Z} \psi(w, z) \tag{1.48}$$

where $\psi : W \times Z \to \mathbb{R}$,

$$\psi(w, z) = < w, p > + < z, q > + < z, Sw > \tag{1.49}$$

is a bilinear function over the two convex compact sets $W$ and $Z$.

# Chapter 2

# Results

In this chapter, we describe the theoretical results of our work.
In the first section, we describe the active set strategy proposed in [4], that is the approach of estimating the zero components of the solution of the problem

$$\min_{x \in \Delta^1} f(x) \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable convex function with Lipschitz continuous gradient over the unit simplex $\Delta^1 \subseteq \mathbb{R}^n$.
Then, we focus on the saddle point problem context.
In the second section, we combine the theoretical results related to saddle point problems with the ones of active set strategy. In particular, we define a specific active-set strategy for the saddle point problems over two simplices and we prove that under some assumptions this strategy allows us to get closer to the saddle point.
In the third section, we describe our algorithm.
Finally, in the fourth section, we give the proof that our algorithm converges to the saddle point of our problem.

## 2.1 Active set strategy

In many applications, the solution of our problem is usually very sparse (with a great number of zero components). If we are able to predict in few steps which of the components will be zero at the solution, we can reduce the dimension of the problem and get significant savings in terms of CPU-TIME. Moreover, we focus our study on problems that have unit simplices as feasible sets because they can model many real-world situations deriving from different fields. For example, every optimization problem over a polytope can be traced back to this context. More precisely, consider the problem

$$\min_{x \in \mathscr{P}} f(x) \tag{2.2}$$

where $\mathscr{P}$ is a polytope. Clearly, each point $x \in \mathscr{P}$ can be written as a convex combination of its vertices $v_1, \ldots, v_m$. Let $V = [v_1, \ldots v_m]$ be the matrix that has the vertices of $\mathscr{P}$ in its columns. Hence, problem (2.2) can be reformulated in the equivalent form

$$\min_{e^\top y=1, y\geq 0} f(Vy) \tag{2.3}$$

where each variable $y_i$ represents the weight of the $i$-th vertex in the convex combination.
After this introductory comment, we describe the results presented in [4] and then we extend the methodology to the saddle point problems context. We consider the minimization of a smooth function over the unit simplex:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & e^\top x = 1 \\ & x \geq 0 \end{aligned} \tag{2.4}$$

where $e = (1, \ldots, 1)^\top$, $f : \mathbb{R}^m \to \mathbb{R}$ is continuously differentiable and its gradient $\nabla f(x)$ is Lipschitz continuous over the simplex.
By the KKT-conditions described above, we get this result:

**Theorem 2.1.** *A feasible point $x^*$ of problem (2.4) is a stationary point if and only if it satisfies the following first order necessary optimality conditions:*

$$\begin{aligned} \nabla f(x^*) - \lambda^* e - \mu^* &= 0 \\ (\mu^*)^\top x^* &= 0 \\ \mu^* &\geq 0 \end{aligned}$$

*where $\lambda^* \in \mathbb{R}$ and $\mu^* \in \mathbb{R}^n$ are the KKT multipliers.*

We shall now define the active set and its approximations.

**Definition 2.1.** Let $x^* \in \mathbb{R}^n$ be a stationary point of problem (2.4). We define the *active set* as the subset of zero components of this stationary point, that is

$$\bar{A}(x^*) = \{i \in \{1, \ldots, n\} : x_i^* = 0\} \tag{2.5}$$

Moreover, we define the *nonactive set* as the complementary of the active set, that is

$$\bar{N}(x^*) = \{1, \ldots, n\} \setminus \bar{A}(x^*) = \{i \in \{1, \ldots, n\} : x_i^* > 0\} \tag{2.6}$$

By simple computations, we obtain these equations for the KKT multipliers

$$\lambda^* = \nabla f(x^*)^\top x^* \tag{2.7}$$

$$\mu^* = \nabla f(x^*) - \lambda^* e \tag{2.8}$$

which suggest to take the following as multiplier functions

$$\lambda(x) = \nabla f(x)^\top x \tag{2.9}$$

$$\mu_i(x) = \nabla_i f(x) - \lambda(x) \quad i = 1, \ldots, n \tag{2.10}$$

From this functions, we define the estimates of the active and non-active sets.

**Definition 2.2.** Let $x \in \mathbb{R}^n$ be a feasible point of problem (2.4). We define the active-set estimate $A(x)$ and the nonactive-set estimate $N(x)$ as

$$A(x) := \{i : x_i \leq \epsilon \mu_i(x)\} = \{i : x_i \leq \epsilon \nabla f(x)^\top (e_i - x)\} \tag{2.11}$$

$$N(x) := \{i : x_i > \epsilon \mu_i(x)\} = \{i : x_i > \epsilon \nabla f(x)^\top (e_i - x)\} \tag{2.12}$$

where $\epsilon > 0$.

The nomenclature derives from the following result.

**Theorem 2.2.** *If $(x^*, \lambda^*, \mu^*)$ satisfies KKT conditions for problem (2.4), then there exists a neighborhood $B(x^*, \rho)$ such that, for each $x$ in this neighborhood, we have*

$$\{i : x_i^* = 0,\ \mu_i(x^*) > 0\} \subseteq A(x) \subseteq \bar{A}(x^*) \tag{2.13}$$

*Furthermore, if strict complementary holds, then*

$$\{i : x_i^* = 0,\ \mu_i(x^*) > 0\} = A(x) = \bar{A}(x^*) \tag{2.14}$$

*for each $x \in B(x^*, \rho)$.*

**Proposition 2.3.** Let $J(x)$ be the set

$$J(x) = \left\{ j : j \in \arg \min_{i=1,\ldots,n} \{\nabla_i f(x)\} \right\} \tag{2.15}$$

Let $x \in \mathbb{R}^n$ be a feasible point of problem (2.4). Then

$$N(x) \cap J(x) \neq \emptyset \tag{2.16}$$

In the following, we need to take an assumption on the constant $\epsilon$.

**Assumption 2.4.** Assume that parameter $\epsilon$ appearing in the estimates (2.11) - (2.12) satisfies the following condition

$$0 < \epsilon < \frac{1}{2Ln} \tag{2.17}$$

where $L$ is the Lipschitz constant of $\nabla f(x)$ over the unit simplex.

Given a feasible point, the active set strategy consists in putting to zero the components in the active set, in order to reduce the value of the objective function. To do this operation, we need to keep the feasibility, so we have to move the positive weight from the active components to some other component. The following proposition suggests how to choose an appropriate index.

**Proposition 2.5.** Let Assumption (2.4) hold. Given a feasible point $x$ of problem (2.4), let $j \in N(x) \cap J(x)$ and $I = \{1, \dots, n\} \setminus \{j\}$. Let $\hat{A}(x)$ be a set of indices such that

$$\hat{A}(x) \subset A(x) \tag{2.18}$$

Let $\tilde{x}$ be the feasible point defined as follows:

$$\tilde{x}_{\hat{A}(x)} = 0 \tag{2.19}$$

$$\tilde{x}_{I \setminus \hat{A}(x)} = x_{I \setminus \hat{A}(x)} \tag{2.20}$$

$$\tilde{x}_j = x_j + \sum_{h \in \hat{A}(x)} x_h \tag{2.21}$$

Then,

$$f(\tilde{x}) - f(x) \leq -L\|\tilde{x} - x\| \tag{2.22}$$

Now, we are ready to describe the Active-Set algorithm for problem (2.4)

**Algorithm 8** Active-Set algorithm for minimization over the simplex

1: Let $x^{(0)}$ be a feasible point
2: **for** $k = 0, 1, \dots$ **do**
3: **if** $x^k$ is a stationary point **then return**
4: **end if**
5: Compute $A^k := A(x^k)$ and $N^k := N(x^k)$
6: Compute $J^k := J(x^k)$, choose $j \in N^k \cap J^k$ and define $\tilde{N}^k = N^k \setminus \{j\}$
7: Set $\tilde{x}^k_{A^k} = 0$, $\tilde{x}^k_{\tilde{N}^k} = x^k_{\tilde{N}^k}$ and $\tilde{x}^k_j = x^k_j + \sum_{h \in A^k} x^k_h$
8: Set $d^k_{A^k} = 0$
9: Compute a feasible direction $d^k_{N^k}$ in $\tilde{x}_k$ and a maximum stepsize $\alpha^k_{max}$
10: **if** $\nabla f(\tilde{x}^k)^\top d^k < 0$ **then** Compute a stepsize $\alpha^k \in (0, \alpha^k_{max}]$ by means of a line search
11: **else** Set $\alpha^k = 0$
12: **end if**
13: Set $x^{k+1} = \tilde{x}^k + \alpha^k d^k$
14: **end for**

In particular, as described above, we can clearly divide each iteration of the algorithm in two phases:

- in steps 5-8, the active-set strategy reduces the feasible region and the objective function
- in steps 9-13, we find a descent direction in the reduced space and we update the current feasible point

## 2.2 Results

Now, we can move on to the saddle point context.
First of all, we specify the KKT Conditions described in Chapter 1 in the case in which the feasible set is given by

$$\mathcal{X} = \Delta_x^1 \subseteq \mathbb{R}^n \tag{2.23}$$
$$\mathcal{Y} = \Delta_y^1 \subseteq \mathbb{R}^m \tag{2.24}$$

where $\Delta^1$ denote the unit simplex. To this end, we have to choose the following constraints

$$g_1(x) = -e^\top x + 1 \tag{2.25}$$
$$h_1(x) = -x \tag{2.26}$$
$$g_2(y) = -e^\top y + 1 \tag{2.27}$$
$$h_2(y) = -y \tag{2.28}$$

and we get that the KKT conditions can be formulated as

$$\nabla_x \mathscr{L}(x^*, y^*) - \lambda_1^* e - \mu_1^* = 0 \tag{2.29}$$
$$\nabla_y \mathscr{L}(x^*, y^*) + \lambda_2^* e + \mu_2^* = 0 \tag{2.30}$$
$$-x^* \leq 0 \quad -y^* \leq 0$$
$$(\mu_1^*)^\top(-x^*) = 0 \quad (\mu_2^*)^\top(-y^*) = 0 \tag{2.31}$$
$$\mu_1^* \geq 0 \quad \mu_2^* \geq 0$$
$$-e^\top x^* + 1 = 0 \quad -e^\top y^* + 1 = 0 \tag{2.32}$$

From (2.29) and (2.30) we get

$$\mu_1^* = \nabla_x \mathscr{L}(x^*, y^*) - \lambda_1^* e \tag{2.33}$$
$$\mu_2^* = -\nabla_y \mathscr{L}(x^*, y^*) - \lambda_2^* e \tag{2.34}$$

If we multiply the first equation by $x^*$ and the second equation by $y^*$, using (2.31), we obtain

$$0 = (\mu_1^*)^\top x^* = \nabla_x \mathscr{L}(x^*, y^*)^\top x^* - \lambda_1^* e^\top x^* \tag{2.35}$$
$$0 = (\mu_2^*)^\top y^* = -\nabla_y \mathscr{L}(x^*, y^*)^\top y^* - \lambda_2^* e^\top y^* \tag{2.36}$$

Then, using (2.32) yields

$$\lambda_1^* = \nabla_x \mathscr{L}(x^*, y^*)^\top x^* \tag{2.37}$$
$$\lambda_2^* = -\nabla_y \mathscr{L}(x^*, y^*)^\top y^* \tag{2.38}$$

Hence, we introduce the following continuous multiplier functions

$$\lambda_1(x, y) = \nabla_x \mathscr{L}(x, y)^\top x \tag{2.39}$$
$$\lambda_2(x, y) = -\nabla_y \mathscr{L}(x, y)^\top y \tag{2.40}$$
$$\mu_{1,i}(x, y) = \nabla_{x,i} \mathscr{L}(x, y) - \lambda_1(x, y) \quad i = 1, \ldots, n \tag{2.41}$$
$$\mu_{2,j}(x, y) = -\nabla_{y,j} \mathscr{L}(x, y) - \lambda_2(x, y) \quad j = 1. \ldots, m \tag{2.42}$$

**Definition 2.3** (Active-Set and Non-Active-Set)**.** We define the active-sets as the subsets of indices that refer to non-zero components of the saddle point solution $(x^*, y^*)$, that are

$$\bar{\mathscr{A}_1}((x^*, y^*)) := \{i : x_i^* = 0\} \tag{2.43}$$

$$\bar{\mathscr{A}_2}((x^*, y^*)) := \{j : y_j^* = 0\} \tag{2.44}$$

Moreover, we define the non-active-sets as the complementary sets of the active-sets, that are

$$\bar{\mathscr{N}_1}((x^*, y^*)) := \{1, \ldots, n\} \setminus \bar{\mathscr{A}_1}((x^*, y^*)) \tag{2.45}$$
$$\bar{\mathscr{N}_2}((x^*, y^*)) := \{1, \ldots, m\} \setminus \bar{\mathscr{A}_2}((x^*, y^*)) \tag{2.46}$$

**Definition 2.4** (Active-Set and Non-Active-Set Estimates)**.** Let $(x, y)$ be a feasible point. We define the active-sets estimates at $(x, y)$ as

$$\mathscr{A}_1((x, y)) := \{i : x_i \leq \epsilon \mu_{1,i}(x, y)\} \tag{2.47}$$
$$\mathscr{A}_2((x, y)) := \{j : y_j \leq \epsilon \mu_{2,j}(x, y)\} \tag{2.48}$$

and the non-active-sets estimates as their complementary sets

$$\mathscr{N}_1((x, y)) := \{1, \ldots, n\} \setminus \mathscr{A}_1((x, y)) \tag{2.49}$$
$$\mathscr{N}_2((x, y)) := \{1, \ldots, m\} \setminus \mathscr{A}_2((x, y)) \tag{2.50}$$

where $\epsilon > 0$.

**Definition 2.5.** We need to define also these four subsets:

$$I_0 := \{i \in \{1, \ldots, n\} : x_i^* = 0\} \tag{2.51}$$
$$J_0 := \{j \in \{1, \ldots, m\} : y_j^* = 0\} \tag{2.52}$$
$$I_+ := \{i \in I_0 : \mu_{1,i}^* > 0\} \tag{2.53}$$
$$J_+ := \{j \in J_0 : \mu_{2,j}^* > 0\} \tag{2.54}$$

**Definition 2.6.** We say that strict complementary holds at $(x^*, y^*)$ if we have the following equalities

$$I_0 = I_+ \tag{2.55}$$
$$J_0 = J_+ \tag{2.56}$$

that is

$$x_i^* = 0 \quad \text{if and only if} \quad \mu_{1,i}^* > 0 \tag{2.57}$$
$$y_j^* = 0 \quad \text{if and only if} \quad \mu_{2,j}^* > 0 \tag{2.58}$$

Then, we can state and prove the following theorem.

**Theorem 2.6.** *If $(x^*, y^*, \lambda_1^*, \lambda_2^*, \mu_1^*, \mu_2^*)$ satisfies the KKT conditions for the problem*

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) \tag{2.59}$$

*then there exists a neighborhood $B((x^*, y^*), \rho)$ such that for all $(x, y) \in B((x^*, y^*), \rho)$ we have*

$$I_+ \subseteq \mathscr{A}_1((x, y)) \subseteq I_0 \tag{2.60}$$
$$J_+ \subseteq \mathscr{A}_2((x, y)) \subseteq J_0 \tag{2.61}$$

*Moreover, if strict complementary holds at $(x^*, y^*)$ we have that*

$$I_+ = \mathscr{A}_1((x, y)) = I_0 \tag{2.62}$$
$$J_+ = \mathscr{A}_2((x, y)) = J_0 \tag{2.63}$$

*for all $(x, y) \in B((x^*, y^*), \rho)$*

*Proof.*

- First, let us prove (2.60).

  Let $i \in I_+$, then $x_i^* = 0$ and $\mu_{1,i}^* > 0$. Now, since $\mu_1^* = \mu_1(x^*, y^*)$, we have $x_i^* \leq \epsilon \mu_{1,i}(x^*, y^*)$ and by the continuity of the multiplier functions we get that $i \in \mathscr{A}_1((x, y))$ for $(x, y) \in B((x^*, y^*), \rho)$ for some $\rho > 0$. Then, we get the left inclusion.

  To prove the right inclusion, let us show that if $i \notin I_0$ then $i \notin \mathscr{A}_1((x, y))$ for any $\rho > 0$. Now, if $i \notin I_0$, then $x_i^* \neq 0$. Since $x^*$ is feasible, we must have $\mu_{1,i}^* = \mu_{1,i}(x^*, y^*) = 0$. Then, $x_i^* > \mu_{1,i}(x^*, y^*)$ and so $i \notin \mathscr{A}_1(x^*, y^*)$.

- Now, let us prove (2.61).

  Let $j \in J_+$, then $y_j^* = 0$ and $\mu_{2,j}^* > 0$. Now, since $\mu_2^* = \mu_2(x^*, y^*)$, we have $y_j^* \leq \epsilon \mu_{2,j}(x^*, y^*)$ and by the continuity of the multiplier functions we get that $j \in \mathscr{A}_2((x, y))$ for $(x, y) \in B((x^*, y^*), \rho)$ for some $\rho > 0$. Then, we get the left inclusion.

To prove the right inclusion, let us show that if $j \notin J_0$ then $j \notin \mathscr{A}_2((x,y))$ for any $\rho > 0$. Now, if $j \notin J_0$, then $y_j^* \neq 0$. Since $y^*$ is feasible, we must have $\mu_{2,j}^* = \mu_{2,j}(x^*, y^*) = 0$. Then, $y_j^* > \mu_{2,j}(x^*, y^*)$ and so $j \notin \mathscr{A}_2(x^*, y^*)$.

- If strict complementary holds, we can prove (2.60) as above. Then, (2.62) follows by definition of strict complementary property, that is $I_0 = I_+$.
- Finally, if strict complementary holds, (2.61) can still be proved as above and (2.63) still follows from definition of strict complementary property, that is $J_0 = J_+$.

□

Before analyzing the algorithm that exploits the active-set estimate, we give some definitions and a proposition that will be useful later.

**Definition 2.7.** Given a feasible point $(x, y)$ we define the following two subsets

$$\mathscr{H}_1((x,y)) := \left\{ i \in \{1, \dots, n\} : i \in \arg\min\{\nabla_{x,i}\mathscr{L}(x,y)\} \right\} \tag{2.64}$$

$$\mathscr{H}_2((x,y)) := \left\{ j \in \{1, \dots, m\} : j \in \arg\max\{\nabla_{y,j}\mathscr{L}(x,y)\} \right\} \tag{2.65}$$

**Proposition 2.7.** Let $(x, y)$ be a feasible point for the problem

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x,y) \tag{2.66}$$

Then,

$$\mathscr{N}_1((x,y)) \cap \mathscr{H}_1((x,y)) \neq \emptyset \tag{2.67}$$

and

$$\mathscr{N}_2((x,y)) \cap \mathscr{H}_2((x,y)) \neq \emptyset \tag{2.68}$$

*Proof.* • First, let us prove (2.67). We consider two possible cases:

(i) If $|\mathscr{H}_1((x,y))| = n$, for every $i \in \mathscr{H}_1((x,y))$ we get

$$\nabla_x \mathscr{L}(x,y)^\top x = \nabla_{x,i}\mathscr{L}(x,y) e^\top x = \nabla_{x,i}\mathscr{L}(x,y) \tag{2.69}$$

Since $x$ is feasible ($e^\top x = 1, x \geq 0$), there exists $\bar{i}$ such that $x_{\bar{i}} > 0$. Then,

$$\begin{aligned} \mu_{1,\bar{i}}(x,y) &= \nabla_{x,\bar{i}}\mathscr{L}(x,y) - \lambda_1(x,y) && (2.70)\\ &= \nabla_{x,\bar{i}}\mathscr{L}(x,y) - \nabla_x \mathscr{L}(x,y)^\top x && (2.71)\\ &= \nabla_{x,\bar{i}}\mathscr{L}(x,y) - \nabla_{x,\bar{i}}\mathscr{L}(x,y) && (2.72)\\ &= 0 < x_{\bar{i}} && (2.73) \end{aligned}$$

Hence, from $x_{\bar{i}} > 0$ and $\mu_{1,\bar{i}}(x,y) = 0$, we get $x_{\bar{i}} > \epsilon \mu_{1,\bar{i}}(x,y)$, which yields to $\bar{i} \in \mathscr{N}_1((x,y)) \cap \mathscr{H}_1((x,y))$.

(ii) If $|\mathscr{H}_1((x,y))| < n$ we may have two further cases:

(ii-a) Assume that for each couple $h \in (\{1,\dots,n\}$ such that

$$\nabla_{x,h}\mathscr{L}(x,y) > \nabla_{x,i}\mathscr{L}(x,y) \quad \text{for } i \in \mathscr{H}_1((x,y)) \tag{2.74}$$

we have $x_h = 0$. In this case, we have

$$\sum_{i\in\mathscr{H}_1((x,y))} x_i = 1 \tag{2.75}$$

and (2.69) still hold. Then, as above, we can choose $\bar{i} \in \mathscr{H}_1((x,y))$ such that $\mu_{1,\bar{i}}(x,y) = 0$, $x_{\bar{i}} > 0$ and we get $\bar{i} \in \mathscr{N}_1((x,y)) \cap \mathscr{H}_1((x,y))$.

(ii-b) Suppose now that there exists $h \in \{1,\dots,n\}$ such that

$$\nabla_{x,h}\mathscr{L}(x,y) > \nabla_{x,i}\mathscr{L}(x,y) \quad \text{and} \quad x_h > 0 \tag{2.76}$$

where $i \in \mathscr{H}_1((x,y))$. Then, we can choose and index $\bar{i} \in \mathscr{H}_1((x,y))$ such that $x_{\bar{i}} \geq 0$ and we get

$$\begin{aligned}
\mu_{1,\bar{i}}(x,y) &= \nabla_{x,\bar{i}}\mathscr{L}(x,y) - \lambda_1(x,y) && (2.77)\\
&= \nabla_{x,\bar{i}}\mathscr{L}(x,y) - \nabla_x\mathscr{L}(x,y)^\top x && (2.78)\\
&< \nabla_{x,\bar{i}}\mathscr{L}(x,y) - \nabla_{x,\bar{i}}\mathscr{L}(x,y)e^\top x && (2.79)\\
&= \nabla_{x,\bar{i}}\mathscr{L}(x,y) - \nabla_{x,\bar{i}}\mathscr{L}(x,y) = 0 && (2.80)
\end{aligned}$$

so $x_{\bar{i}} > \epsilon\mu_{1,\bar{i}}(x,y)$ In this way, we have $\bar{i} \in \mathscr{N}_1((x,y)) \cap \mathscr{H}_1((x,y))$.

- In a similar way, we prove (2.68).

(i) If $|\mathscr{H}_2((x,y))| = m$, for every $j \in \mathscr{H}_2((x,y))$ we get

$$\nabla_y\mathscr{L}(x,y)^\top y = \nabla_{y,j}\mathscr{L}(x,y)e^\top y = \nabla_{y,j}\mathscr{L}(x,y) \tag{2.81}$$

Since $y$ is feasible ($e^\top y = 1, y \geq 0$), there exists $\bar{j}$ such that $y_{\bar{j}} > 0$. Then,

$$\begin{aligned}
\mu_{2,\bar{j}}(x,y) &= -\nabla_{y,\bar{j}}\mathscr{L}(x,y) - \lambda_2(x,y) && (2.82)\\
&= -\nabla_{y,\bar{i}}\mathscr{L}(x,y) + \nabla_y\mathscr{L}(x,y)^\top y && (2.83)\\
&= -\nabla_{y,\bar{j}}\mathscr{L}(x,y) + \nabla_{y,\bar{j}}\mathscr{L}(x,y) && (2.84)\\
&= 0 < y_{\bar{j}} && (2.85)
\end{aligned}$$

Hence, from $y_{\bar{j}} > 0$ and $\mu_{2,\bar{j}}(x,y) = 0$, we get $y_{\bar{j}} > \epsilon\mu_{2,\bar{j}}(x,y)$, which yields to $\bar{j} \in \mathscr{N}_2((x,y)) \cap \mathscr{H}_2((x,y))$.

(ii) If $|\mathscr{H}_2((x,y))| < m$ we may have two further cases:

(ii-a) Assume that for each couple $k \in (\{1, \dots, m\}$ such that

$$\nabla_{y,k}\mathscr{L}(x,y) < \nabla_{y,j}\mathscr{L}(x,y) \quad \text{for } j \in \mathscr{H}_2((x,y)) \tag{2.86}$$

we have $y_k = 0$. In this case, we have

$$\sum_{j \in \mathscr{H}_2((x,y))} y_j = 1 \tag{2.87}$$

and (2.81) still hold. Then, as above, we can choose $\bar{j} \in \mathscr{H}_2((x,y))$ such that $\mu_{2,\bar{j}}(x,y) = 0$, $y_{\bar{j}} > 0$ and we get $\bar{j} \in \mathscr{N}_2((x,y)) \cap \mathscr{H}_2((x,y))$.

(ii-b) Suppose now that there exists $k \in \{1, \dots, m\}$ such that

$$\nabla_{y,k}\mathscr{L}(x,y) < \nabla_{y,j}\mathscr{L}(x,y) \quad \text{and} \quad y_k > 0 \tag{2.88}$$

where $j \in \mathscr{H}_2((x,y))$. Then, we can choose and index $\bar{j} \in \mathscr{H}_2((x,y))$ such that $y_{\bar{j}} \geq 0$ and we get

$$\begin{aligned}
\mu_{2,\bar{j}}(x,y) &= -\nabla_{y,\bar{j}}\mathscr{L}(x,y) - \lambda_2(x,y) && (2.89)\\
&= -\nabla_{y,\bar{j}}\mathscr{L}(x,y) + \nabla_y\mathscr{L}(x,y)^\top y && (2.90)\\
&< -\nabla_{y,\bar{j}}\mathscr{L}(x,y) + \nabla_{y,\bar{j}}\mathscr{L}(x,y)e^\top y && (2.91)\\
&= -\nabla_{y,\bar{j}}\mathscr{L}(x,y) + \nabla_{y,\bar{j}}\mathscr{L}(x,y) = 0 && (2.92)
\end{aligned}$$

so $y_{\bar{j}} > \epsilon\mu_{2,\bar{j}}(x,y)$ In this way, we have $\bar{j} \in \mathscr{N}_2((x,y)) \cap \mathscr{H}_2((x,y))$.

□

**Assumption 2.8.** In the following we need to add a hypothesis on $\epsilon$ in order to prove the following proposition. We assume

$$\epsilon < \min\left\{\frac{1}{2L(n+1)}, \frac{1}{2L(m+1)}\right\} \tag{2.93}$$

**Proposition 2.9.** Let Assumption (2.8) hold. Given a feasible point $(x,y)$, let $i \in \mathscr{N}_1((x,y)) \cap \mathscr{H}_1((x,y))$, $j \in \mathscr{N}_2((x,y)) \cap \mathscr{H}_2((x,y))$, $\mathcal{I} := \{1, \dots, n\} \setminus \{i\}$ and $\mathcal{J} := \{1, \dots, m\} \setminus \{j\}$. Let $\hat{A}_1((x,y)) \subseteq A_1((x,y))$ and $\hat{A}_2((x,y)) \subseteq A_2((x,y))$ be two subsets of indices. Let $(\tilde{x}, \tilde{y})$ be the following feasible point:

$$\tilde{x}_{\hat{A}_1} = 0 \tag{2.94}$$

$$\tilde{y}_{\hat{A}_2} = 0 \tag{2.95}$$

$$\tilde{x}_{\mathcal{I}\setminus\hat{A}_1} = x_{\mathcal{I}\setminus\hat{A}_1} \tag{2.96}$$

$$\tilde{y}_{\mathcal{J}\setminus\hat{A}_2} = y_{\mathcal{J}\setminus\hat{A}_2} \tag{2.97}$$

$$\tilde{x}_i = x_i + \sum_{h\in\hat{A}_1} x_h \tag{2.98}$$

$$\tilde{y}_j = y_j + \sum_{k\in\hat{A}_2} y_k \tag{2.99}$$

where we write $\hat{A}_1 := \hat{A}_1((x,y))$ and $\hat{A}_2 := \hat{A}_2((x,y))$ for simplicity. Then,

$$\mathscr{L}(\tilde{x},y) - \mathscr{L}(x,y) \leq -L\|\tilde{x}-x\|^2 \quad \text{and} \quad \mathscr{L}(x,\tilde{y}) - \mathscr{L}(x,y) \geq L\|\tilde{y}-y\|^2 \tag{2.100}$$

*Proof.* • By the mean value theorem, we have

$$\mathscr{L}(\tilde{x},y) = \mathscr{L}(x,y) + \nabla_x\mathscr{L}(\omega,y)^\top(\tilde{x}-x) \tag{2.101}$$

$$= \mathscr{L}(x,y) + \nabla_x\mathscr{L}(x,y)^\top(\tilde{x}-x) + [\nabla_x\mathscr{L}(\omega,y) - \nabla_x\mathscr{L}(x,y)]^\top(\tilde{x}-x) \tag{2.102}$$

where $\omega = x + \xi_1(\tilde{x}-x)$, $\xi_1 \in (0,1)$, which is equivalent to

$$\mathscr{L}(\tilde{x},y) - \mathscr{L}(x,y) = \nabla_x\mathscr{L}(x,y)^\top(\tilde{x}-x) + [\nabla_x\mathscr{L}(\omega,y) - \nabla_x\mathscr{L}(x,y)]^\top(\tilde{x}-x) \tag{2.103}$$

Then,

$$\mathscr{L}(\tilde{x},y) - \mathscr{L}(x,y) \leq \nabla_x\mathscr{L}(x,y)^\top(\tilde{x}-x) + L\|\tilde{x}-x\|^2 \tag{2.104}$$

Now, if we add and remove $L\|\tilde{x}-x\|^2$, we get

$$\mathscr{L}(\tilde{x},y) - \mathscr{L}(x,y) \leq \nabla_x\mathscr{L}(x,y)^\top(\tilde{x}-x) - L\|\tilde{x}-x\|^2 + 2L\|\tilde{x}-x\|^2 \tag{2.105}$$

and we just need to show that

$$\nabla_x\mathscr{L}(x,y)^\top(\tilde{x}-x) + 2L\|\tilde{x}-x\|^2 \leq 0 \tag{2.106}$$

By definition of $\tilde{x}$ we have

$$\nabla_x\mathscr{L}(x,y)^\top(\tilde{x}-x) = -\nabla_{x,\hat{A}_1}\mathscr{L}(x,y)^\top x_{\hat{A}_1} + \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)^\top\left(\sum_{h\in\hat{A}_1} x_h\right) \tag{2.107}$$

$$= x_{\hat{A}_1}^\top\left(\nabla_{x,\bar{\imath}}\mathscr{L}(x,y)e_{\hat{A}_1} - \nabla_{x,\hat{A}_1}\mathscr{L}(x,y)\right) \tag{2.108}$$

Moreover,

$$\|\tilde{x} - x\|^2 = \sum_{h\in\hat{A}_1} x_h^2 + \Big(\sum_{h\in\hat{A}_1} x_h\Big)^2 \tag{2.109}$$

$$\leq \sum_{h\in\hat{A}_1} x_h^2 + |\hat{A}_1| \sum_{h\in\hat{A}_1} x_h^2 \tag{2.110}$$

$$= \big(1 + |\hat{A}_1|\big) x_{\hat{A}_1}^\top x_{\hat{A}_1} \tag{2.111}$$

Furthermore, by definition of $\bar{\imath}$, it yields

$$\nabla_{x,\bar{\imath}}\mathscr{L}(x,y) \leq \nabla_{x,h}\mathscr{L}(x,y) \quad \text{for all } h \in \{1,\dots,n\} \tag{2.112}$$

that leads to

$$\sum_{h=1}^{n} \nabla_{x,h}\mathscr{L}(x,y)x_h \geq \sum_{h=1}^{n} \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)x_h = \tag{2.113}$$

$$= \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)\sum_{h=1}^{n} x_h = \nabla_{x,\bar{\imath}}\mathscr{L}(x,y) \tag{2.114}$$

since $\sum_{h=1}^{n} x_h = 1$. Now, recalling the active-set estimate, we have that for any $h \in \hat{A}_1$

$$x_h \leq \epsilon\mu_{1,h}(x,y) \tag{2.115}$$

$$= \epsilon\bigg(\nabla_{x,h}\mathscr{L}(x,y) - \sum_{h=1}^{n} \nabla_{x,h}\mathscr{L}(x,y)x_h\bigg) \tag{2.116}$$

$$\leq \epsilon\bigg(\nabla_{x,h}\mathscr{L}(x,y) - \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)\bigg) \tag{2.117}$$

Combining this last inequality with (2.111), we get

$$\|\tilde{x} - x\|^2 \leq \epsilon\big(1 + |\hat{A}_1|\big) x_{\hat{A}_1}^\top \bigg(\nabla_{x,\hat{A}_1}\mathscr{L}(x,y) - \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)e_{\hat{A}_1}\bigg) \tag{2.118}$$

Hence, if we put everything together we reach

$$\begin{aligned}
&\nabla_x\mathscr{L}(x,y)^\top(\tilde{x} - x) + 2L\|\tilde{x} - x\|^2 \leq \\
&\qquad \leq x_{\hat{A}_1}^\top\bigg(\nabla_{x,\bar{\imath}}\mathscr{L}(x,y)e_{\hat{A}_1} - \nabla_{x,\hat{A}_1}\mathscr{L}(x,y)\bigg) + \\
&\quad + 2L\epsilon\big(1 + |\hat{A}_1|\big) x_{\hat{A}_1}^\top\bigg(\nabla_{x,\hat{A}_1}\mathscr{L}(x,y) - \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)e_{\hat{A}_1}\bigg) = \\
&= (2L\epsilon(1 + |\hat{A}_1|) - 1) x_{\hat{A}_1}^\top\bigg(\nabla_{x,\hat{A}_1}\mathscr{L}(x,y) - \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)e_{\hat{A}_1}\bigg) \leq \\
&\qquad \leq (2L\epsilon(n+1) - 1) x_{\hat{A}_1}^\top\bigg(\nabla_{x,\hat{A}_1}\mathscr{L}(x,y) - \nabla_{x,\bar{\imath}}\mathscr{L}(x,y)e_{\hat{A}_1}\bigg)
\end{aligned}$$

since $|\hat{A}_1| \leq n$. Now, by (2.112), the last term of the product is non-negative. Therefore, taking in account the Assumption (2.8), we get $2L\epsilon(n+1) - 1 < 0$. Thus

$$\nabla_x \mathscr{L}(x,y)^\top(\tilde{x} - x) + 2L\|\tilde{x} - x\|^2 \leq 0 \tag{2.119}$$

- By the mean value theorem, we have

$$\begin{aligned}&\mathscr{L}(x,\tilde{y}) = \mathscr{L}(x,y) + \nabla_y \mathscr{L}(x,\nu)^\top(\tilde{y} - y) = \\ &= \mathscr{L}(x,y) + \nabla_y \mathscr{L}(x,y)^\top(\tilde{y} - y) + [\nabla_y \mathscr{L}(x,\nu) - \nabla_y \mathscr{L}(x,y)]^\top(\tilde{y} - y)\end{aligned}$$

where $\nu = y + \xi_2(\tilde{y} - y)$, $\xi_2 \in (0,1)$, which is equivalent to

$$-\mathscr{L}(x,\tilde{y}) + \mathscr{L}(x,y) = -\nabla_y \mathscr{L}(x,y)^\top(\tilde{y} - y) + [\nabla_y \mathscr{L}(x,\nu) - \nabla_y \mathscr{L}(x,y)]^\top(y - \tilde{y}) \tag{2.120}$$

Then,

$$\mathscr{L}(x,y) - \mathscr{L}(x,\tilde{y}) \leq -\nabla_y \mathscr{L}(x,y)^\top(\tilde{y} - y) + L\|\tilde{y} - y\|^2 \tag{2.121}$$

Now, if we add and remove $L\|\tilde{y} - y\|^2$, we get

$$\mathscr{L}(x,y) - \mathscr{L}(x,\tilde{y}) \leq -\nabla_y \mathscr{L}(x,y)^\top(\tilde{y} - y) - L\|\tilde{y} - y\|^2 + 2L\|\tilde{y} - y\|^2 \tag{2.122}$$

and we just need to show that

$$-\nabla_y \mathscr{L}(x,y)^\top(\tilde{y} - y) + 2L\|\tilde{y} - y\|^2 \leq 0 \tag{2.123}$$

By definition of $\tilde{y}$ we have

$$-\nabla_y \mathscr{L}(x,y)^\top(\tilde{y} - y) = \nabla_{y,\hat{A}_2} \mathscr{L}(x,y)^\top y_{\hat{A}_2} - \nabla_{y,\bar{j}} \mathscr{L}(x,y)^\top \Big( \sum_{k \in \hat{A}_2} y_k \Big) \tag{2.124}$$

$$= y_{\hat{A}_2}^\top \Big( \nabla_{y,\hat{A}_2} \mathscr{L}(x,y) - \nabla_{y,\bar{j}} \mathscr{L}(x,y) e_{\hat{A}_2} \Big) \tag{2.125}$$

Moreover,

$$\|\tilde{y} - y\|^2 = \sum_{k \in \hat{A}_2} y_k^2 + \Big( \sum_{k \in \hat{A}_2} y_k \Big)^2 \tag{2.126}$$

$$\leq \sum_{k \in \hat{A}_2} y_k^2 + |\hat{A}_2| \sum_{k \in \hat{A}_2} y_k^2 \tag{2.127}$$

$$= \big(1 + |\hat{A}_2|\big) y_{\hat{A}_2}^\top y_{\hat{A}_2} \tag{2.128}$$

Furthermore, by definition of $\bar{j}$, we can write

$$\nabla_{y,\bar{j}}\mathscr{L}(x,y) \geq \nabla_{y,k}\mathscr{L}(x,y) \quad \text{for all } k \in \{1,\dots,m\} \tag{2.129}$$

that leads to

$$\sum_{k=1}^{m} \nabla_{y,k}\mathscr{L}(x,y)y_k \leq \sum_{k=1}^{m} \nabla_{y,\bar{j}}\mathscr{L}(x,y)y_k = \tag{2.130}$$

$$= \nabla_{y,\bar{j}}\mathscr{L}(x,y)\sum_{k=1}^{m} y_k = \nabla_{y,\bar{j}}\mathscr{L}(x,y) \tag{2.131}$$

since $\sum_{k=1}^{m} y_k = 1$. Now, recalling the active-set estimate, for any $k \in \hat{A}_2$ we have

$$y_k \leq \epsilon\mu_{2,k}(x,y) \tag{2.132}$$

$$= \epsilon\bigg( -\nabla_{y,k}\mathscr{L}(x,y) + \sum_{k=1}^{m} \nabla_{y,k}\mathscr{L}(x,y)y_k \bigg) \tag{2.133}$$

$$\leq \epsilon\bigg( -\nabla_{y,k}\mathscr{L}(x,y) + \nabla_{y,\bar{j}}\mathscr{L}(x,y) \bigg) \tag{2.134}$$

Combining this last inequality with (2.128), we get

$$\|\tilde{y}-y\|^2 \leq \epsilon\big(1+|\hat{A}_2|\big)y_{\hat{A}_2}^{\top}\bigg(\nabla_{y,\bar{j}}\mathscr{L}(x,y)e_{\hat{A}_2} - \nabla_{y,\hat{A}_2}\mathscr{L}(x,y)\bigg) \tag{2.135}$$

Hence, if we put everything together we reach

$$\begin{aligned}
&-\nabla_y\mathscr{L}(x,y)^{\top}(\tilde{y}-y) + 2L\|\tilde{y}-y\|^2 \leq \\
&\qquad \leq y_{\hat{A}_2}^{\top}\bigg(\nabla_{y,\hat{A}_2}\mathscr{L}(x,y) - \nabla_{y,\bar{j}}\mathscr{L}(x,y)e_{\hat{A}_2}\bigg) + \\
&\quad + 2L\epsilon\big(1+|\hat{A}_2|\big)y_{\hat{A}_2}^{\top}\bigg(\nabla_{y,\bar{j}}\mathscr{L}(x,y)e_{\hat{A}_2} - \nabla_{y,\hat{A}_2}\mathscr{L}(x,y)\bigg) = \\
&= (2L\epsilon(1+|\hat{A}_2|)-1)y_{\hat{A}_2}^{\top}\bigg(\nabla_{y,\bar{j}}\mathscr{L}(x,y)e_{\hat{A}_2} - \nabla_{y,\hat{A}_2}\mathscr{L}(x,y)\bigg) \leq \\
&\qquad \leq (2L\epsilon(m+1)-1)y_{\hat{A}_2}^{\top}\bigg(\nabla_{y,\bar{j}}\mathscr{L}(x,y)e_{\hat{A}_2} - \nabla_{y,\hat{A}_2}\mathscr{L}(x,y)\bigg)
\end{aligned}$$

since $|\hat{A}_2| \leq m$. Now, by (2.129), the last term of the product is non-negative. Therefore, taking in account the Assumption (2.8), we have in particular that $2L\epsilon(m+1)-1 < 0$. Thus

$$-\nabla_y\mathscr{L}(x,y)^{\top}(\tilde{y}-y) + 2L\|\tilde{y}-y\|^2 \leq 0 \tag{2.136}$$

□

## 2.3 Algorithm

In this section we combine the active-set approach described above with the Frank-Wolfe methodology proposed in [7]. In particular, each step of the following algorithm is composed by two different steps.

1. In the first phase, we estimate the non-active variables, that is, given $(x^k, y^k)$, we compute the set of indices $A_1((x^k, y^k))$, $A_2((x^k, y^k))$, $N_1((x^k, y^k))$ and $N_2((x^k, y^k))$. Then, we generate the new iterate $(\tilde{x}^k, \tilde{y}^k)$ following the instructions given in Proposition (2.9);

2. in the second one, we find the new feasible direction over the non-active-sets $N_1(\tilde{x}^k, \tilde{y}^k)$ and $N_2(\tilde{x}^k, \tilde{y}^k)$, using the Frank-Wolfe method.

**Algorithm 9** Active-Set algorithm framework for saddle point problems over the Cartesian product of two simplices (AS-SP-SIMPLICES)

Let $(x^{(0)}, y^{(0)})$ be a feasible point
**for** $k = 0, 1, \ldots$ **do**
  **if** $(x^k, y^k)$ is a stationary point **then return**
  **end if**
  Compute $A_1^k := A_1((x^k, y^k))$, $A_2^k := A_2((x^k, y^k))$, $N_1^k := N_1((x^k, y^k))$ and $N_2^k := N_2((x^k, y^k))$
  Compute $\mathscr{H}_1^k := \mathscr{H}_1((x^k, y^k))$ and $\mathscr{H}_2^k := \mathscr{H}_2((x^k, y^k))$,
   choose $i \in N_1^k \cap \mathscr{H}_1^k$, $j \in N_2^k \cap \mathscr{H}_2^k$ and
   define $\tilde{N}_1^k = N_1^k \setminus \{i\}$, $\tilde{N}_2^k = N_2^k \setminus \{j\}$
  Set $\tilde{x}^k_{A^k} = 0 = \tilde{y}^k_{A^k}$,
   $\tilde{x}^k_{\tilde{N}_1^k \setminus \{i\}} = x^k_{\tilde{N}_1^k \setminus \{i\}}$,
   $\tilde{y}^k_{\tilde{N}_2^k \setminus \{j\}} = y^k_{\tilde{N}_2^k \setminus \{j\}}$,
   $\tilde{x}^k_i = x^k_i + \sum_{h \in A_1^k} x^k_h$ and
   $\tilde{y}^k_j = y^k_l + \sum_{l \in A_2^k} y^k_l$
  Set $d^k_{A^k} = 0$
  Compute a feasible direction $d^k_{N^k}$ in $(\tilde{x}^k, \tilde{y}^k)$ and a maximum stepsize $\alpha^k_{max}$
  **if** $\begin{pmatrix} \nabla_x \mathscr{L}(\tilde{x}^k, \tilde{y}^k) \\ -\nabla_y \mathscr{L}(\tilde{x}^k, \tilde{y}^k) \end{pmatrix}^\top d^k < 0$ **then**
    Compute a stepsize $\alpha^k = \frac{2}{2+(k+1)}$
  **else** Set $\alpha^k = 0$
  **end if**
  Set $(x^{k+1}, y^{k+1}) = (\tilde{x}^k, \tilde{y}^k) + \alpha^k d^k$
**end for**

where we recall that the Frank-Wolfe direction for the saddle point problem

is given by:

$$d^{FW}_{A^k_1} := 0 \tag{2.137}$$

$$d^{FW}_{A^k_2} := 0 \tag{2.138}$$

$$d^{FW}_{N^k_1} := e_{\bar{i}} - \tilde{x}^k_{N^k_1} \tag{2.139}$$

$$d^{FW}_{N^k_2} := e_{\bar{j}} - \tilde{y}^k_{N^k_2} \tag{2.140}$$

where

$$\bar{i} \in \arg\min_{i \in N^k_1} \{\nabla_{x,i} \mathscr{L}(\tilde{x}^k, \tilde{y}^k)\} \tag{2.141}$$

$$\bar{j} \in \arg\max_{j \in N^k_2} \{\nabla_{y,j} \mathscr{L}(\tilde{x}^k, \tilde{y}^k)\} \tag{2.142}$$

In many applications, it can be very difficult to find the Lipschitz constant of our objective function gradient. As a consequence, it may not be easy to choose an appropriate value for the constant $\epsilon$ in the active set strategy.
To overcome this possible problem, we can follow an iterative scheme to estimate, at each iteration of the Algorithm 9, an appropriate $\epsilon$ that satisfies conditions in Proposition (2.9).
In particular, we need to introduce, for each step of the Algorithm 9 the further phase described in Algorithm 10.

**Algorithm 10** Choice of $\epsilon$

Let $(x^k, y^k)$ be the point computed at iteration k by Algorithm (9)
Let $\epsilon_x = 1$ and $\epsilon_y = 1$.
Let $L_x = \frac{1}{4\epsilon_x(n+1)}$ and $L_y = \frac{1}{4\epsilon_y(m+1)}$
**repeat**
    $\epsilon_x = \epsilon_x * 0.1$
    Compute $\tilde{x}^k$
    Update $L_x = L_x * 10$
**until** $\mathscr{L}(\tilde{x}^k, y^k) - \mathscr{L}(x^k, y^k) \leq -L_x \|\tilde{x}^k - x^k\|^2$
**repeat**
    $\epsilon_x = \epsilon_x * 0.1$
    Compute $\tilde{y}^k$
    Update $L_y = L_y * 10$
**until** $\mathscr{L}(x^k, \tilde{y}^k) - \mathscr{L}(x^k, y^k) \geq L_y \|\tilde{y}^k - y^k\|^2$

Clearly, the two cycles in Algorithm 10 end in a finite number of iterations. Indeed, whatever is the Lipschitz constant $L$, reducing $\epsilon$ of an order of magnitude at each iteration, we get $\epsilon < \min\left\{\frac{1}{2L(n+1)}, \frac{1}{2L(m+1)}\right\}$ after a finite number of iterations. Hence we are under the hypothesis of Proposition (2.9) and the conditions to stop the cycles are satisfied.

On the other hand, this phase of the algorithm can be very expensive, so, having an estimate of the Lipschitz constant, can be very useful.

## 2.4 Convergence

In this section, we want to prove the convergence of Algorithm 9 under some strong assumptions.
All the details and the formal definitions can be found in the Appendix A or in [7], but we give here the interpretation of some constants to make the following results understandable:

- the curvature constant $C_{\mathscr{L}}$ is related to the Lipschitz constant of $\nabla\mathscr{L}$ and ensures that the deviation of $\mathscr{L}$ from its linearization is bounded
- the interior strong convex-concavity constant $\mu_{\mathscr{L}}^{\text{int}}$ is related to the strong convex-concavity of $\mathscr{L}$
- the bilinearity coefficient $M_{\mathscr{L}}$ relates $\nabla\mathscr{L}$ computed at the saddle point and at the other feasible points
- the suboptimality sequence $\tilde{w}^k := \mathscr{L}(\tilde{x}^k, y^*) - \mathscr{L}(x^*, \tilde{y}^k)$ measures the distance to the saddle point and its convergence to 0 will show the convergence of Algorithm 9 to the saddle point itself.

First of all, we have the following result.

**Proposition 2.10.** Let Assumption 2.8 hold. Let $\{(x^k, y^k)\}_k$, $\{(\tilde{x}^k, \tilde{y}^k)\}_k$ and $\{d^k\}_k$ be the sequences produced by the Algorithm 9. If that algorithm does not terminate in a finite number of iterations, then

$$\lim_{k\to\infty} \|(\tilde{x}^k, \tilde{y}^k) - (x^k, y^k)\| = 0 \tag{2.143}$$

*Proof.* Using the thesis of Proposition (2.9), we get

$$\mathscr{L}(x^{k+1}, y) \leq \mathscr{L}(\tilde{x}, y) \leq \mathscr{L}(x^k, y) - L\|\tilde{x}^k - x^k\|^2 \quad \text{for all } y \in \mathcal{Y} \tag{2.144}$$

From the compactness of $\mathcal{X}$, the sequence $\{x^k\}_k$ is convergent to some point $\bar{x} \in \mathcal{X}$. Moreover, from the continuity of $\mathscr{L}$, we get

$$\lim_{k\to\infty} (\mathscr{L}(x^{k+1}, y) - \mathscr{L}(x^k, y)) = 0 \tag{2.145}$$

Putting together (2.144) and (2.145), we finally reach

$$\lim_{k\to\infty} \|\tilde{x}^k - x^k\|^2 = 0 \tag{2.146}$$

On the other hand, we have

$$\mathscr{L}(x, y^{k+1}) \geq \mathscr{L}(x, \tilde{y}^k) \geq \mathscr{L}(x, y^k) + L\|\tilde{y}^k - y^k\|^2 \quad \text{for all} \quad x \in \mathcal{X} \tag{2.147}$$

From the compactness of $\mathcal{Y}$, the sequence $\{y^k\}_k$ is convergent to some point $\bar{y} \in \mathcal{Y}$. Then, by the continuity of $\mathscr{L}$, it yields

$$\lim_{k\to\infty} (\mathscr{L}(x, y^{k+1}) - \mathscr{L}(x, y^k)) = 0 \tag{2.148}$$

Putting together (2.147) and (2.148) we get

$$\lim_{k\to\infty} \|\tilde{y}^k - y^k\|^2 = 0 \tag{2.149}$$

Finally, we have

$$\|(\tilde{x}^k, \tilde{y}^k) - (x^k, y^k)\|^2 \leq \|\tilde{x}^k - x^k\|^2 + \|\tilde{y}^k - y^k\|^2 \tag{2.150}$$

which leads to the thesis. □

In the following, we assume that

- $\mathscr{L}$ is strongly convex-concave
- $\nabla\mathscr{L}$ is $L$-Lipschitz continuous
- strict complementary holds at $(x^*, y^*)$
- $\mathscr{L}$ has a finite curvature constant $\mathcal{C}_{\mathscr{L}}$
- $\mathscr{L}$ has a positive interior strong convex-concavity constant $\mu^{\text{int}}_{\mathscr{L}}$
- Assumption 2.8 holds

Now, we restrict to the simpler case in which we compute the descent direction $d^k$ over the whole feasible set $\mathcal{X} \times \mathcal{Y}$ instead of the non-active estimates $N_1((x^k, y^k))$ and $N_2((x^k, y^k))$. Since the search of the descent direction is linked to the minimization of a linear function, this simplification does not involve a considerable increase in CPU-time.
The proof of convergence in the case where the descent direction is restricted to non-active estimates goes beyond the scope of this thesis, but the experimental results and some of our preliminary attempts allow us to assume the correctness of this approach.
Then, we can state the following theorem.

**Theorem 2.11.** *Let $\mathscr{L}$ be a strongly convex-concave function with a finite curvature constant $C_{\mathscr{L}}$, a positive interior strong convex-concavity constant $\mu^{int}_{\mathscr{L}}$ and a L- Lipschitz continuous gradient. Assume that strict complementary holds at the unique saddle point $(x^*, y^*)$. Let us also define the rate multiplier $\nu = 1 - \frac{M_{\mathscr{L}}}{\sqrt{\mu^{int}_{\mathscr{L}}}}$.*

*If $\nu > 0$, the suboptimality $\tilde{w}^k := \mathscr{L}(\tilde{x}^k, y^*) - \mathscr{L}(x^*, \tilde{y}^k)$ of the iterates of the algorithm with step size $\gamma^k = \min\left(\gamma_{max}, \frac{2}{2+k}\right)$ has the following decreasing upper bound*

$$\tilde{w}^k \leq \frac{C}{2+k} \tag{2.151}$$

*for $k$ sufficiently large, where $C = 2\max\{w^0, \frac{2C_{\mathscr{L}}}{2\nu-1}\}$*

In [7] it is proved that the suboptimality $w^k = \mathscr{L}(x^k, y^*) - \mathscr{L}(x^*, y^k)$ has the following asymptotic behaviour:

$$w^k = \mathcal{O}\left(\frac{1}{k}\right) \tag{2.152}$$

Moreover, we have the following result.

**Proposition 2.12.** Consider the two suboptimalities error sequences

$$w^k := \mathscr{L}(x^k, y^*) - \mathscr{L}(x^*, y^k) \tag{2.153}$$
$$\tilde{w}^k := \mathscr{L}(\tilde{x}^k, y^*) - \mathscr{L}(x^*, \tilde{y}^k) \tag{2.154}$$

where $\{(x^k, y^k)\}_k$ and $\{(\tilde{x}^k, \tilde{y}^k)\}_k$ are the sequences generated by Algorithm 9.
Then,

$$\tilde{w}^k \leq w^k \quad \text{for all } k \tag{2.155}$$

*Proof.* By Proposition (2.9) we get

$$\mathscr{L}(\tilde{x}^k, y^*) - \mathscr{L}(x^k, y^*) \leq -L\|\tilde{x}^k - x^k\|^2 \tag{2.156}$$
$$\mathscr{L}(x^*, \tilde{y}^k) - \mathscr{L}(x^*, y^k) \geq L\|\tilde{y}^k - y^k\|^2 \tag{2.157}$$

In particular, it yields

$$\mathscr{L}(\tilde{x}^k, y^*) \leq \mathscr{L}(x^k, y^*) \tag{2.158}$$
$$\mathscr{L}(x^*, \tilde{y}^k) \geq \mathscr{L}(x^*, y^k) \tag{2.159}$$

Combining the above inequalities, we have

$$\mathscr{L}(\tilde{x}^k, y^*) - \mathscr{L}(x^*, \tilde{y}^k) \leq \mathscr{L}(x^k, y^*) - \mathscr{L}(x^*, y^k) \tag{2.160}$$

that is

$$\tilde{w}^k \leq w^k \tag{2.161}$$

□

Then, the proof of Theorem 2.11 immediately follows from (2.152) and from the Proposition 2.12.

# Chapter 3

# Numerical Experiments

In this chapter, we describe the experiment we use to test our algorithm.

## 3.1 Toy Problem and Python codes

We consider the *Toy Problem* described in [7] .
Let $\mathscr{L} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$ such that

$$\mathscr{L}(x, y) = \frac{\mu}{2}\|x - x^*\|_2^2 + (x - x^*)^\top M(y - y^*) - \frac{\mu}{2}\|y - y^*\|_2^2 \tag{3.1}$$

where $\mu > 0$ is a scalar and $M \in \mathcal{M}_{n_1 \times n_2}(\mathbb{R})$ is a matrix.
This function is $\mu-$strongly convex-concave and $(x^*, y^*)$ is a saddle point for the problem

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) \tag{3.2}$$

where $\mathcal{X} = \Delta_x \subseteq \mathbb{R}^{n_1}$ and $\mathcal{Y} = \Delta_y \subseteq \mathbb{R}^{n_2}$ are unit simplices.
Since our algorithm should work better with problems where the saddle point is sparse, we introduce two more variables $m_1$ and $m_2$ such that $x^*$ has $m_1$ non-zero components and $y^*$ has $m_2$ non-zero components.
Then, we want to generate $x^*$ and $y^*$ randomly over the two simplices. According to [5], we use the following results.

**Proposition 3.1.** Let $U_1, \ldots, U_n$ be a sequence of i.i.d. random variables such that $U_i \sim \mathcal{U}(0, 1)$ for all $i$, with order statistics $U_{(1)}, \ldots, U_{(n)}$. Define $U_{(0)} = 0$ and $U_{(n+1)} = 1$. Moreover, let $S_i := U_{(i)} - U_{(i-1)}$ for $i = 1, \ldots, n+1$ and let $(Y_1, \ldots, Y_{n+1})$ be the sequence of i.i.d. random variables such that $Y_i \sim \text{Exp}(1)$. Then,

$$(S_1, \ldots, S_{n+1}) \sim \left( \frac{Y_1}{\sum_{i=1}^{n+1} Y_i}, \ldots, \frac{Y_{n+1}}{\sum_{i=1}^{n+1} Y_i} \right) \tag{3.3}$$

**Proposition 3.2.** Let $(S_1, \ldots, S_{n+1})$ be the same sequence defined in Proposition (3.1). Then, $(S_1, \ldots, S_{n+1})$ is uniformly distributed over the simplex

$$\Delta_{n+1} := \left\{ (x_1, \ldots, x_{n+1}) \text{ s.t. } x_i \geq 0, \sum_{i=1}^{n+1} x_i = 1 \right\} \tag{3.4}$$

Hence, we begin generating $\sigma_1, \ldots, \sigma_{m_1}$ from an exponential distribution with parameter 1. Then, we normalize the sequence, obtaining

$$(s_1, \ldots, s_{m_1}) = \left( \frac{\sigma_1}{\sum_{i=1}^{m_1} \sigma_i}, \ldots, \frac{\sigma_{m_1}}{\sum_{i=1}^{m_1} \sigma_i} \right) \tag{3.5}$$

and we randomly insert such components in a zero vector in $\mathbb{R}^{n_1}$. In such a way we generate $x^*$. In the same way, replacing $n_1$ with $n_2$ and $m_1$ with $m_2$, we generate $y^*$.
Moreover, as proposed in [7], we create the matrix $M$ generating
$n_1 \times n_2$ i.i.d. random samples from a uniform distribution over the interval $[-0.1, 0.1]$ (that is $\mathcal{U}(-0.1, 0.1)$).
Thus, we have written a Python function that takes as input $n_1, n_2, m_1$ and $m_2$ and returns $x^*, y^*$ and $M$.

Listing 3.1: Toy Problem

```
def Toy_Problem(n1,n2,m1,m2):
'''

:param n1: dimension of X
:param n2: dimension of Y
:param m1: non-zero components of x_star
:param m2: non-zero components of y_star

:return x_star: x component of the saddle point
        y_star: y component of the saddle point
        M: random n1*n2 matrix

'''

        x_star_simplex = np.random.exponential(1,size = m1)
        x_star_simplex = x_star_simplex / np.sum(x_star_simplex)

        y_star_simplex = np.random.exponential(1, size=m2)
        y_star_simplex = y_star_simplex / np.sum(y_star_simplex)

        x_star = np.zeros(n1)
        indexes_1 = np.random.choice(n1,m1,replace = False)
        x_star [indexes_1] = x_star_simplex

        y_star = np.zeros(n2)
        indexes_2 = np.random.choice(n2, m2, replace=False)
        y_star[indexes_2] = y_star_simplex
```

```
    x_star = np.matrix(x_star).T
    y_star = np.matrix(y_star).T

    M = np.random.uniform(0,1  , n1*n2)
    M = np.matrix(np.reshape(M,(n1,n2)))

return (x_star, y_star, M)
```

To use our algorithm, we need to compute the gradients of the objective function:

$$\nabla_x \mathscr{L}(x,y) = \mu(x-x^*)+M(y-y^*) \quad \text{and} \quad \nabla_y \mathscr{L}(x,y) = -\mu(y-y^*)+M^\top(x-x^*) \tag{3.6}$$

Moreover, we have

$$\begin{aligned}
&\begin{pmatrix} \nabla_x \mathscr{L}(\tilde{x}^k,\tilde{y}^k) \\ -\nabla_y \mathscr{L}(\tilde{x}^k,\tilde{y}^k) \end{pmatrix}^\top \begin{pmatrix} x \\ y \end{pmatrix} = \\
&= \big(\mu(\tilde{x}^k - x^*) + M(\tilde{y}^k - y^*)\big)^\top x + \big(\mu(\tilde{y}^k - y^*) - M^\top(\tilde{x}^k - x^*)\big)^\top y = \\
&= x^\top\big(\mu(\tilde{x}^k - x^*) + M(\tilde{y}^k - y^*)\big) + \big((\tilde{y}^k - y^*)^\top\mu - (\tilde{x}^k - x^*)^\top M\big)y = \\
&= a(x) + b(y)
\end{aligned}$$

Thus, at each step of the algorithm, to compute the feasible FW direction, we have to find

$$\operatorname*{arg\,min}_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \begin{pmatrix} \nabla_x \mathscr{L}(\tilde{x}^k,\tilde{y}^k) \\ -\nabla_y \mathscr{L}(\tilde{x}^k,\tilde{y}^k) \end{pmatrix}^\top \begin{pmatrix} x \\ y \end{pmatrix} = \operatorname*{arg\,min}_{x\in\mathcal{X}} a(x) + \operatorname*{arg\,min}_{y\in\mathcal{Y}} b(y) \tag{3.7}$$

Since $a(x)$ and $b(y)$ are linear functions over two convex sets, we can easily solve the two minimization problems above. The solution is $(\bar{x},\bar{y})$ such that

$$\bar{x}_i = \begin{cases} 1 & \text{if } i \in \arg\min_i \mu(\tilde{x}_i^k - x_i^*) + (M(\tilde{y}^k - y^*))_i \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

$$\bar{y}_j = \begin{cases} 1 & \text{if } j \in \arg\min_j (\tilde{y}_j^k - y_j^*)^\top\mu - ((\tilde{x}^k - x^*)^\top M)_j \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

In particular, if such minima are achieved for more than one index, we choose the lower index.
Furthermore, we need to show that the gradient of the objective function is Lipschitz continuous. In particular, as described in the previous chapter, having an upper bound of the Lipschitz constant guarantees significant saving in terms of CPU-TIME. Indeed, in this case, we can avoid to use Algorithm 10, that is, concretely, two *for* cycles less at each iteration of the algorithm.
This reason justifies the effort to estimate a constant $L > 0$ such that

$$\|\nabla f(x'',y'') - \nabla f(x',y')\|_* \leq L\|(x'',y'') - (x',y')\| \tag{3.10}$$

for all $x'', x' \in \mathcal{X}, y'', y' \in \mathcal{Y}$, where

$$\|(x,y)\|_* := \max\left\{\|x\|_{\mathcal{X}^*}, \|y\|_{\mathcal{Y}^*}\right\} \tag{3.11}$$

and

$$\|x\|_{\mathcal{X}^*} := \sup_{\substack{s\in\mathbb{R}^n \\ \|s\|\leq 1}} x^\top s \quad , \quad \|y\|_{\mathcal{Y}^*} := \sup_{\substack{r\in\mathbb{R}^m \\ \|r\|\leq 1}} y^\top r \tag{3.12}$$

are the dual norms. Finally, we define

$$\|(x,y)\| := \|x\| + \|y\| \tag{3.13}$$

Such a constant $L$, is a an estimate of the full Lipschitz constant of $\nabla\mathscr{L}$ and can be used to bound also the partial Lipschitz constants $L_{XX}, L_{XY}$, $L_{YX}, L_{YY}$ that are defined in Appendix A.
Now, we have

$$\|\nabla_x\mathscr{L}(x'',y'')-\nabla_x\mathscr{L}(x',y')\|_{\mathcal{X}^*} = \sup_{\substack{s\in\mathbb{R}^n \\ \|s\|\leq 1}} (\nabla_x f(x'',y'') - \nabla_x f(x',y'))^\top s \tag{3.14}$$

$$= \sup_{\substack{s\in\mathbb{R}^n \\ \|s\|\leq 1}} \left\{(\mu(x''-x^*) - \mu(x'-x^*) + My'' - My')^\top s\right\} \tag{3.15}$$

$$= \sup_{\substack{s\in\mathbb{R}^n \\ \|s\|\leq 1}} \left\{(\mu(x''-x') + M(y''-y'))^\top s\right\} \tag{3.16}$$

$$\leq \sup_{\substack{s\in\mathbb{R}^n \\ \|s\|\leq 1}} \left\{\|\mu(x''-x') + M(y''-y')\|\|s\|\right\} \tag{3.17}$$

$$\leq \mu\|x''-x'\| + \|M\|\|y''-y'\| \tag{3.18}$$

$$\leq \mu\|x''-x'\| + 0.1 * \sqrt{n_2}\|y''-y'\| \tag{3.19}$$

$$\leq \max\{\mu, 0.1*\sqrt{n_2}\}(\|x''-x'\| + \|y''-y'\|) \tag{3.20}$$

$$= \max\{\mu, 0.1*\sqrt{n_2}\}\|(x'',y'')-(x',y')\| \tag{3.21}$$

since $\|s\| \leq 1$ and

$$\|M\| := \max_{x,\|x\|=1} \|Mx\| = \|(0.1\cdots 0.1)^\top\| = 0.1*\sqrt{n_2} \tag{3.22}$$

where $\|\cdot\|$ denotes here the Euclidean norm in $\mathbb{R}^{n_2}$.

In the same way, we can write

$$\|\nabla_y \mathscr{L}(x'', y'') - \nabla_y \mathscr{L}(x', y')\|_{\mathcal{Y}^*} = \sup_{\substack{r\in\mathbb{R}^m \\ \|r\|\leq 1}} (\nabla_y f(x'', y'') - \nabla_y f(x', y'))^\top r \tag{3.23}$$

$$= \sup_{\substack{r\in\mathbb{R}^m \\ \|r\|\leq 1}} \left\{(-\mu(y'' - y^*) + \mu(y' - y*) + M^\top x'' - M^\top x')^\top r\right\} \tag{3.24}$$

$$= \sup_{\substack{r\in\mathbb{R}^m \\ \|r\|\leq 1}} \left\{(-\mu(y'' - y') + M^\top (x'' - x'))^\top r\right\} \tag{3.25}$$

$$\leq \sup_{\substack{r\in\mathbb{R}^m \\ \|r\|\leq 1}} \left\{(\mu\|y'' - y'\| + \|M^\top\|\|x'' - x'\|)\|r\|\right\} \tag{3.26}$$

$$\leq \mu\|y'' - y'\| + 0.1 * \sqrt{n_1}\|x'' - x'\| \tag{3.27}$$

$$\leq \max\{\mu, 0.1 * \sqrt{n_1}\}(\|x'' - x'\| + \|y'' - y'\|) \tag{3.28}$$

$$= \max\{\mu, 0.1 * \sqrt{n_1}\}\|(x'', y'') - (x', y')\| \tag{3.29}$$

since $\|r\| \leq 1$ and

$$\|M^\top\| := \max_{y, \|y\|=1} \|M^\top y\| = \|(0.1 \cdots 0.1)^\top\| = 0.1 * \sqrt{n_1} \tag{3.30}$$

where $\|\cdot\|$ denotes here the Euclidean norm in $\mathbb{R}^{n_1}$.
Thus, we have shown that $\nabla\mathscr{L}$ is Lipschitz-continuous with constant at least

$$L = \max\{\mu, 0.1 * \sqrt{n_1}, 0.1 * \sqrt{n_2}\} \tag{3.31}$$

Hence, we can choose

$$\epsilon = \frac{1}{4L\max\{n+1, m+1\}} \tag{3.32}$$

In this way, we can skip Algorithm 10 and save time in our computations.
For the active-set estimates, we define a Python function that takes in input the current point $(x_k, y_k)$, the input values generated by Function (3.1) and the constants $\mu$ and $\epsilon$. In output we get $(\tilde{x}_k, \tilde{y}_k)$, that is the new point where we have put to zero the components in the active-set estimates.

Listing 3.2: Active-Set Estimates

```
def active_set_estimate(x,y,x_star,y_star,M,mu,eps):
    '''

    :param x: x component of the current approximation
    :param y: y component of the current approximation
    :param x_star: x component of the saddle point
    :param y_star: y component of the saddle point
```

```
:param M: matrix of the toy problem
:param mu: strong convex-concavity constant
:param eps: active-set constant

:return xx: new x component according to active-set strategy
        yy: new y component according to active-set strategy
'''

        xx = x.copy()
        yy = y.copy()
        mu_1 = (mu * (x-x_star)  + M * (y-y_star))
        mu_1n = mu_1 - x.T*(mu_1)
        mu_2 = (mu * (y - y_star) - M.T * (x - x_star))
        mu_2n = mu_2 - y.T * (mu_2)
        k1 = xx - eps * mu_1n
        k2 = yy - eps * mu_2n

        x_count = np.sum(xx[k1 < 0])
        y_count = np.sum(yy[k2 < 0])

        xx[k1 < 0] = 0
        yy[k2 < 0] = 0

        ii = np.nonzero(mu_1)
        i_bar = np.argmin(ii)

        jj = np.nonzero(mu_2)
        j_bar = np.argmin(jj)

        xx[ii[0][i_bar]] += x_count
        yy[jj[0][j_bar]] += y_count

return xx,yy
```

To check the behaviour of our algorithm, we choose the centre of mass of the two simplices as starting point of the algorithm, because we know that Frank-Wolfe methodology suffers from the zig-zagging problem in this context. The following Python code generates such a starting point.

Listing 3.3: Starting Point

```
def Starting_Point(n1,n2):
'''

:param n1: dimension of X
:param n2: dimension of Y
:return:        x_0 = (1/n1, ... , 1/n1)
                y_0 = (1/n2, ... , 1/n2)
'''

        x_0 = np.matrix(np.ones(n1) / n1).T
        y_0 = np.matrix(np.ones(n2) / n2).T

return x_0, y_0
```

In our computations, we need to find at each iteration the Frank-Wolfe direction. The following function takes in input the current iteration and the usual other values to give in output the FW-direction described in Section 2.3. In particular, we take in input not the current iteration $(\tilde{x}_k, \tilde{y}_k)$, but only its components in the current non-active estimates. In this way, we can put $d_{A_1^k} = 0 = d_{A_2^k}$.

Listing 3.4: Frank-Wolfe directon

```
def FW_direction (x,y,x_star,y_star,M,mu):
'''

:param x: x component of the current approximation
:param y: y component of the current approximation
:param x_star: x component of the saddle point
:param y_star: y component of the saddle point
:param M: matrix of the toy problem
:param mu: strong convex-concavity constant

:return d_x: x component of the FW direction
        d_y: y component of the FW direction
'''

        x_new = np.matrix(np.zeros(len(x))).T
        y_new = np.matrix(np.zeros(len(y))).T

        mu_1 = (mu * (x - x_star) + M * (y - y_star))
        mu_1n = mu_1 - x.T * (mu_1)
        mu_2 = (mu * (y - y_star) - M.T * (x - x_star))
        mu_2n = mu_2 - y.T * (mu_2)

        i = np.argmin(mu_1n)
        j = np.argmin(mu_2n)

        x_new[i] = 1
        y_new[j] = 1

        d_x = x_new - x
        d_y = y_new - y

return d_x, d_y
```

Finally, we can give the AS-SIMPLICES-FW function, that implements the algorithm described in Chapter 2.

Listing 3.5: AS-SIMPLICES Function

```
def AS_SIMPLICES_FW (x_0,y_0,x_star,y_star,mu,M,eps,toll,max_it,
                     M_y_star,MT_x_star):

'''

:param x_0: x component of the starting point
:param y_0: y component of the starting point
```

```
:param x_star: x component of the saddle point
:param y_star: y component of the saddle point
:param mu: strong convex-concavity constant
:param M: matrix of the toy problem
:param eps: active-set constant
:param toll: tolerance for optimality
:param max_it: maximum number of iterations
:param M_y_star: product M* y_star
:param MT_x_star: product M^T * x_star

:return x_k: x oomponent of the last approximation
        y_k: y component of the last appoximation
        k_stop: number of the last iteration
        gaps: list of the values of the gap function
'''

x_k , y_k = x_0 , y_0
gaps = []

for k in range(0,max_it):
        i_nonzero = np.nonzero(x_k)[0]
        j_nonzero = np.nonzero(y_k)[0]

        xtilde_k = np.matrix(np.zeros(len(x_0))).T
        ytilde_k = np.matrix(np.zeros(len(y_0))).T

        xtilde_k[i_nonzero], ytilde_k[j_nonzero] =
                = active_set_estimate(x_k[i_nonzero], y_k[j_nonzero],
                        x_star[i_nonzero], y_star[j_nonzero],
                        (M[[i_nonzero]])[:,j_nonzero], mu, eps)

        d_x = np.matrix(np.zeros(len(x_0))).T
        d_y = np.matrix(np.zeros(len(y_0))).T

        d_x[i_nonzero], d_y[j_nonzero] =
                = FW_direction(xtilde_k[i_nonzero], ytilde_k[j_nonzero],
                        x_star[i_nonzero], y_star[j_nonzero],
                        (M[[i_nonzero]])[:, j_nonzero], mu)

        gap_x = - (d_x).T * (mu * xtilde_k - mu * x_star +
                + (M * ytilde_k) - M_y_star)

        gap_y = -(d_y).T * (mu*ytilde_k- mu*y_star -
                + (M.T *xtilde_k)+ MT_x_star)

        gap = gap_x + gap_y
        gaps.append(gap.item(0))

        if gap < toll :
                k_stop = k
                return xtilde_k , ytilde_k , k_stop , gaps

        else:
                x_k = xtilde_k + 2/(2+(k+1))* d_x
```

```
            y_k = ytilde_k + 2/(2+(k+1))* d_y

        k_stop = max_it

        return x_k,y_k, k_stop, gaps
```

To reduce the dimension of our problem, at each iteration of the for cycle, we compute the non-zero components of the current point. Then, we call Function 3.2 only for such values. After that, we compute the FW direction with Function 3.4.
We save the gaps and the total number of iterations because they will be useful to show the results of our tests.
Finally, we also give the script of the SIMPLICES-FW implementation, that we will use to have a comparison of the results of our algorithm.

Listing 3.6: SIMPLICES-FW Function

```
def SIMPLICES_FW(x_0,y_0,x_star,y_star,mu,M,eps,
        toll,max_it, M_y_star,MT_x_star):

'''
Same Parameters And Output Values that AS-SIMPLICES-FW Function
'''

x_k, y_k = x_0, y_0
gaps = []

for k in range(0, max_it):

        d_x, d_y = FW_direction(x_k, y_k, x_star, y_star, M, mu)

        gap = - d_x.T * (mu* x_k - mu*x_star + M* y_k - M_y_star) -
        + d_y.T * (mu*y_k - mu*y_star - M.T *x_k + MT_x_star)

        gaps.append(gap.item(0))

        if gap < toll:
                k_stop = k
                return x_k, y_k, k_stop, gaps

        else:
                x_k = x_k + 2 / (2 + (k + 1)) * d_x
                y_k = y_k + 2 / (2 + (k + 1)) * d_y

k_stop = max_it

return x_k, y_k,k_stop, gaps
```

## 3.2 Numerical Results

In this section, we describe the numerical results of our tests.
First of all, let us give the data of our simulations.
We have considered the Toy Problem described above with these data:

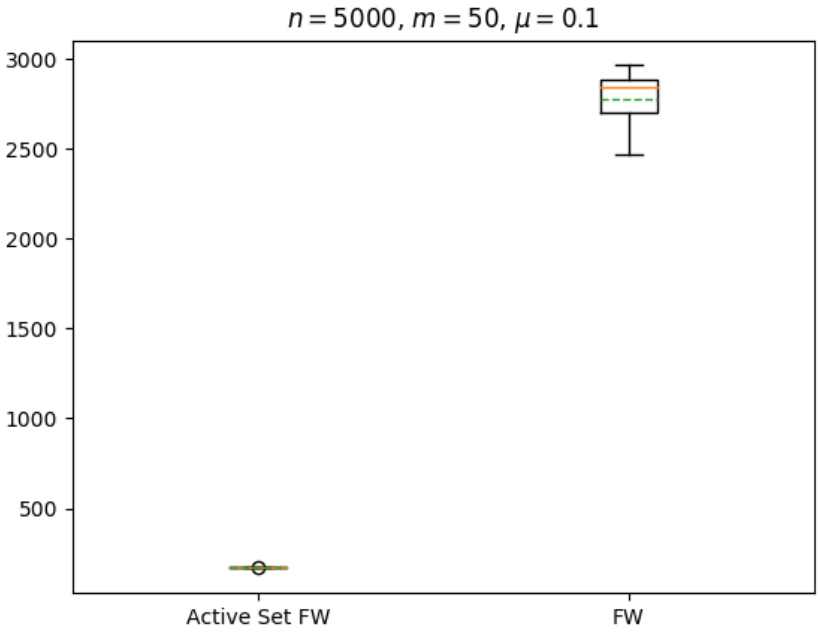
- $n = n_1 = n_2 \in \{5000, 10000\}$
- $m = m_1 = m_2 \in \{0.01 * n, 0.05 * n\}$, that means the active-set represents the 99 % or 95 % of all the components of the saddle point and so the solution is very sparse.
- $\mu \in \{0.1, 1\}$, that leads to different convergence times
- toll $= 0.001$
- a maximum number of iterations of 100000

We run FW algorithm for saddle point problems and our active set modification 20 times for each setting.
We report the results in the following table, where for each setting we give the means of CPU-TIME and number of iterations and also their standard deviations.

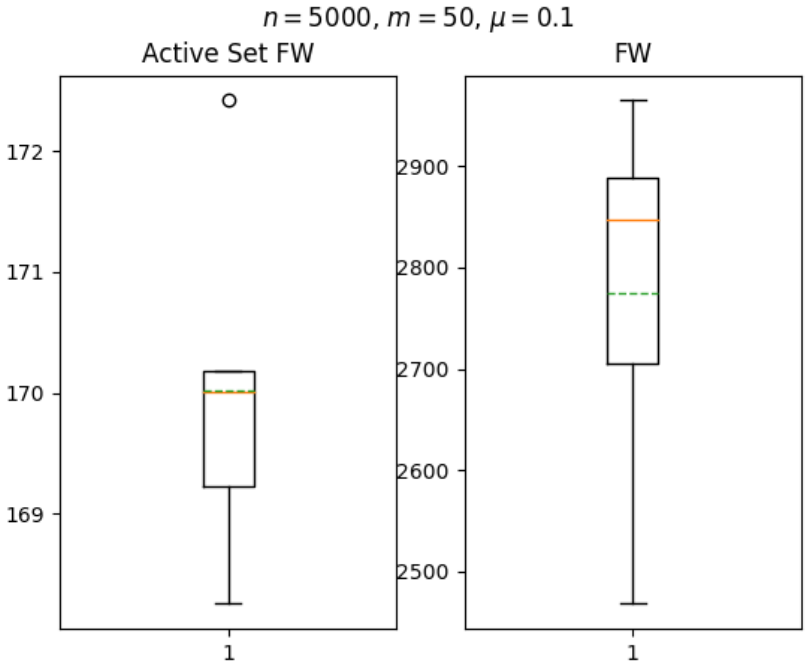
| | | | CPU-TIME | | ITERATIONS | |
|---|---|---|---|---|---|---|
| $n$ | $m$ | $\mu$ | Active-Set | FW | Active-Set | FW |
| 5000 | 50 (99%) | 0.1 | $170 \pm 1.4$ | $2774 \pm 175$ | $608 \pm 40$ | $53014 \pm 3422$ |
| | | 1 | $164 \pm 2.5$ | $267 \pm 9$ | $364 \pm 62$ | $5094 \pm 173$ |
| | 250 (95%) | 0.1 | $235 \pm 9$ | $5228 \pm 9$ | $1505 \pm 225$ | 100000 |
| | | 1 | $205 \pm 4$ | $835 \pm 17.5$ | $693 \pm 85$ | $15990 \pm 344$ |
| 10000 | 100 (99%) | 0.1 | $830 \pm 18$ | $20914 \pm 75$ | $1051 \pm 163$ | 100000 |
| | | 1 | $766 \pm 12$ | $1762 \pm 58$ | $510 \pm 53$ | $8424 \pm 292$ |
| | 500 (95%) | 0.1 | $1127 \pm 33$ | $20910 \pm 107$ | $1942 \pm 278$ | 100000 |
| | | 1 | $1006 \pm 10$ | $6109 \pm 133$ | $965 \pm 79$ | $29148 \pm 666$ |

As we can see in the table above, the active-set strategy leads to much better statistics, both on CPU-TIME and number of iterations. In particular, in some cases, the active-set algorithm reaches the tolerance in a few hundreds iterations, while FW can not do it in 100000 iterations and more then 5 hours of CPU-TIME (see also Figure 3.13 or Figure 3.15).
Moreover, we can see that active-set results are characterized by a lower standard deviation and so, they seem more stable (see for example Figure 3.5 or Figure 3.8).
To better visualize the results, we also give the box-plots for the CPU-time experiments (Figures 3.1 - 3.8) and the classical Cartesian plots for

the decreasing of the gap function with respect to the number of iterations (Figures 3.9 - 3.16).
In the box-plots, we have also underlined the mean (in green) and the median (in orange) of our samples.
In the Cartesian plots instead, we can see that the two algorithms behave in the same way for the very first iterations, but then the active-set gap drops drastically faster. To underline this difference, we use a logarithmic scale for the ordinates axis.

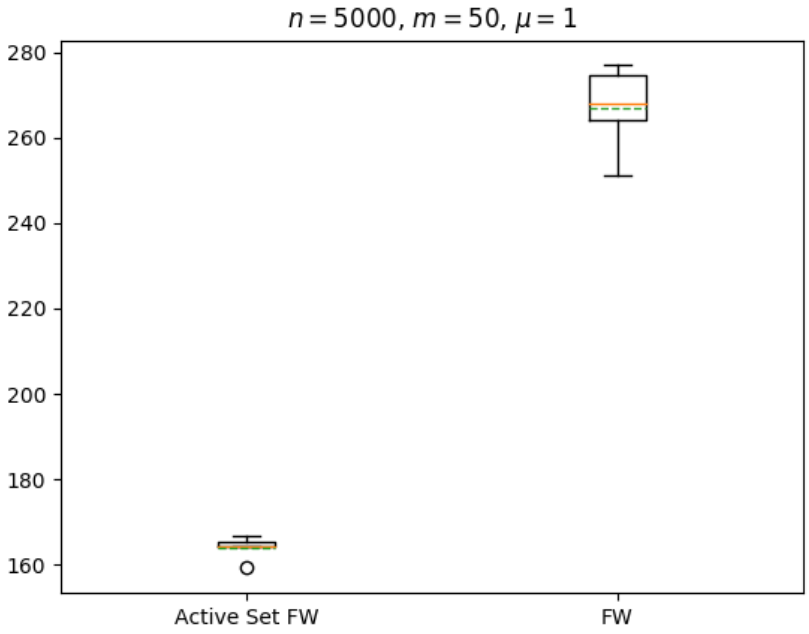



(a) Box-Plots

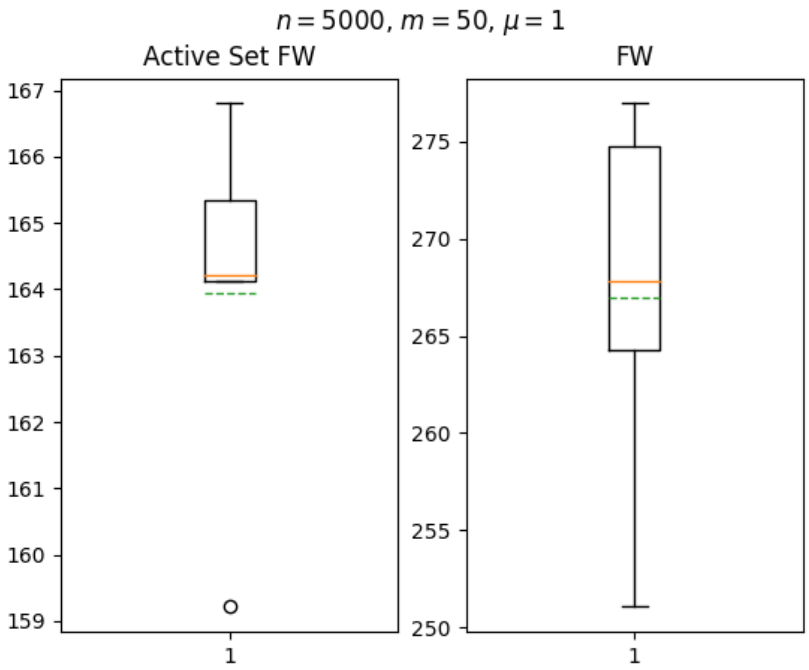



(b) Detailed Box-Plots

Figure 3.1: CPU-TIME Box-Plots $n = 5000,\ m = 50,\ \mu = 0.1$

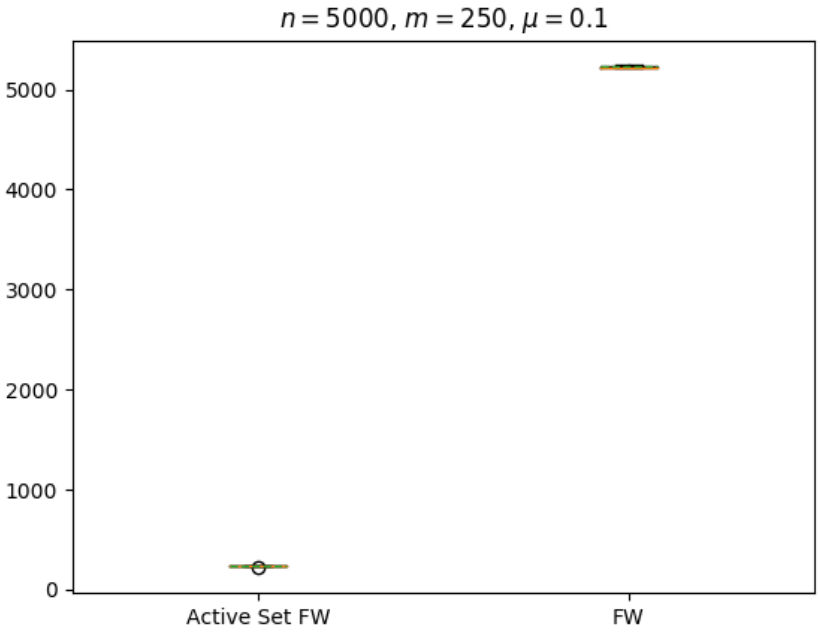



(a) Box-Plots

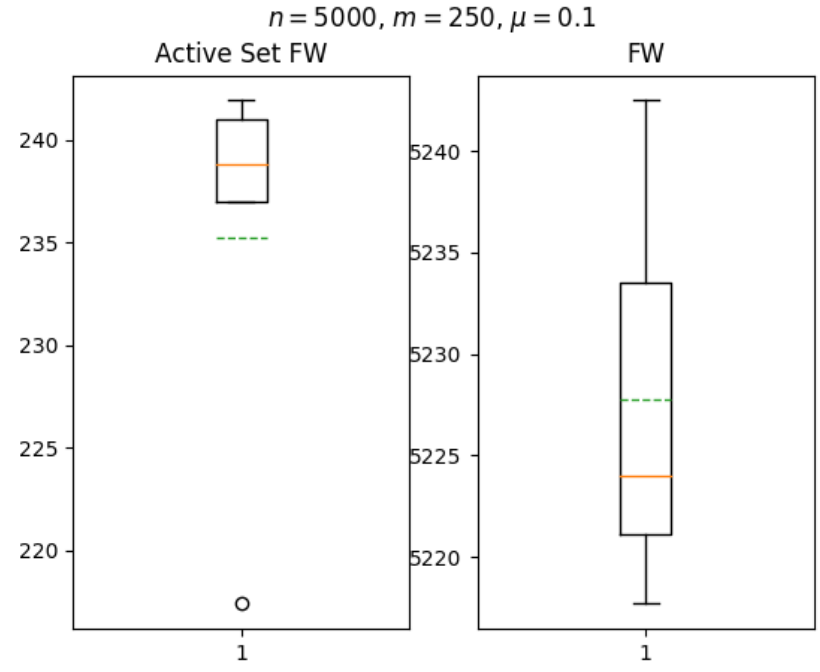



(b) Detailed Box-Plots

Figure 3.2: CPU-TIME Box-Plots $n = 5000,\ m = 50,\ \mu = 1$

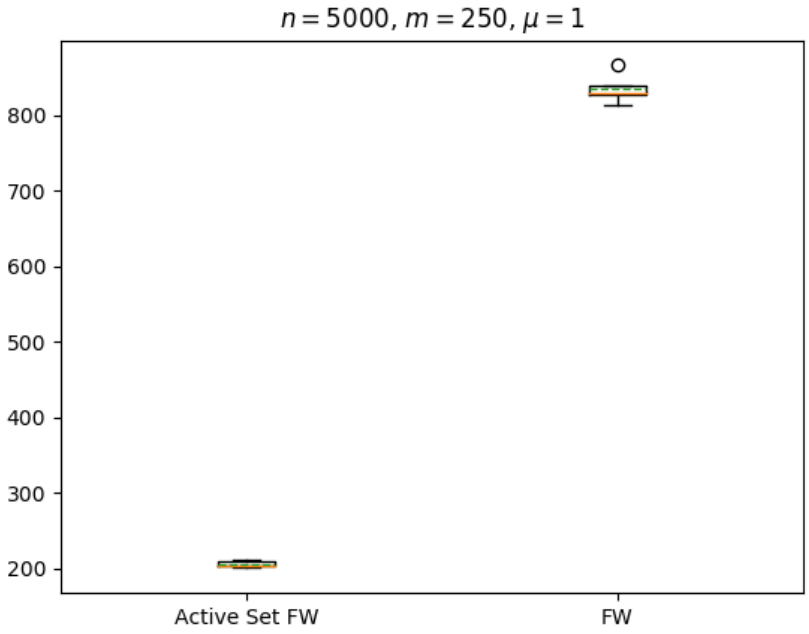



(a) Box-Plots

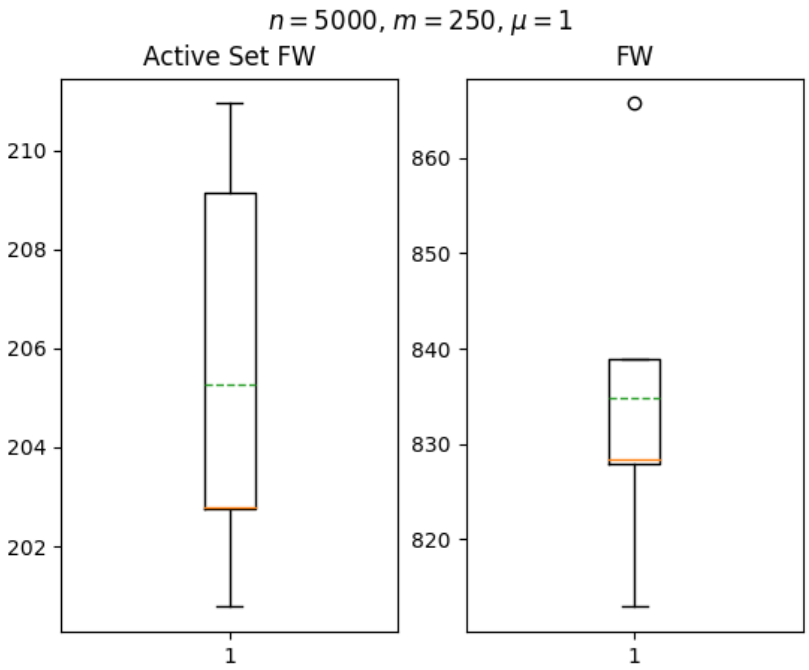



(b) Detailed Box-Plots

Figure 3.3: CPU-TIME Box-Plots $n = 5000,\ m = 250,\ \mu = 0.1$

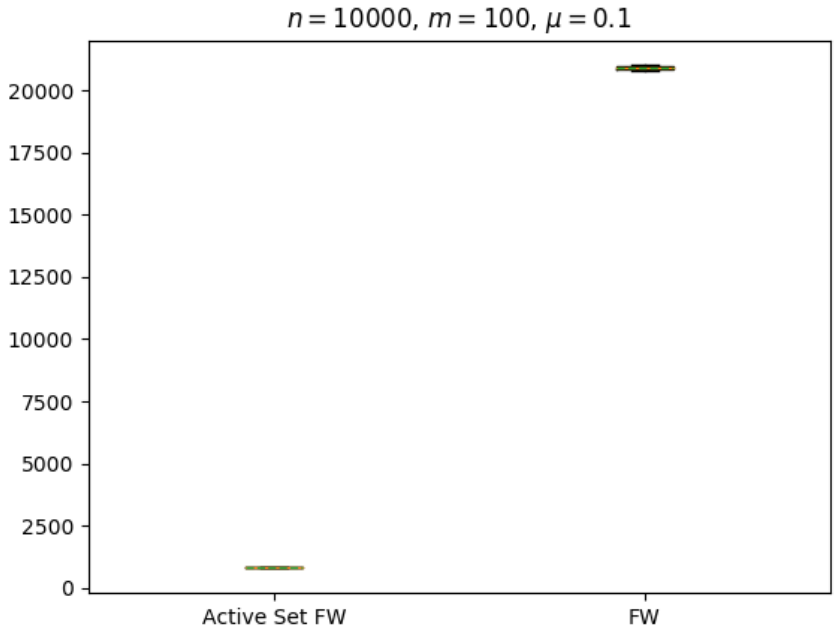



(a) Box-Plots

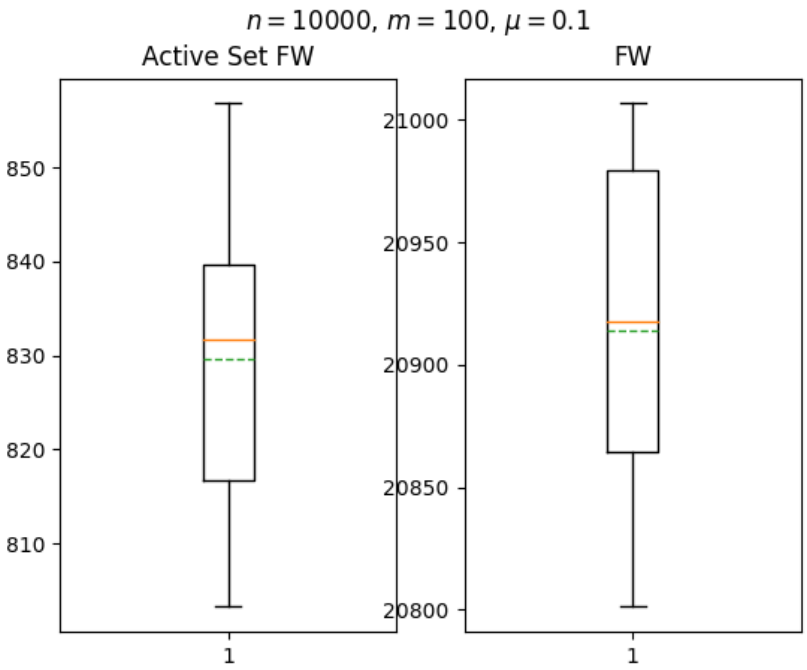



(b) Detailed Box-Plots

Figure 3.4: CPU-TIME Box-Plots $n = 5000,\ m = 250,\ \mu = 1$

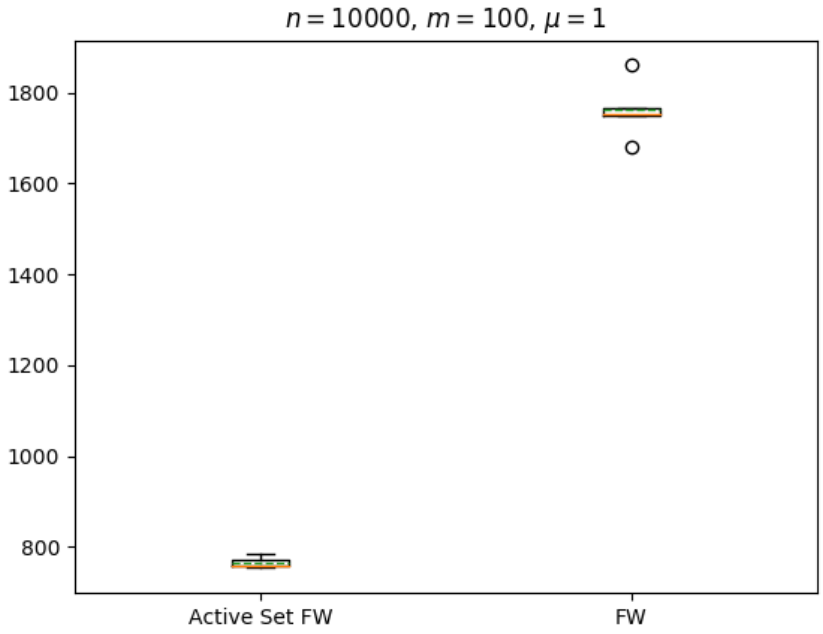



(a) Box-Plots

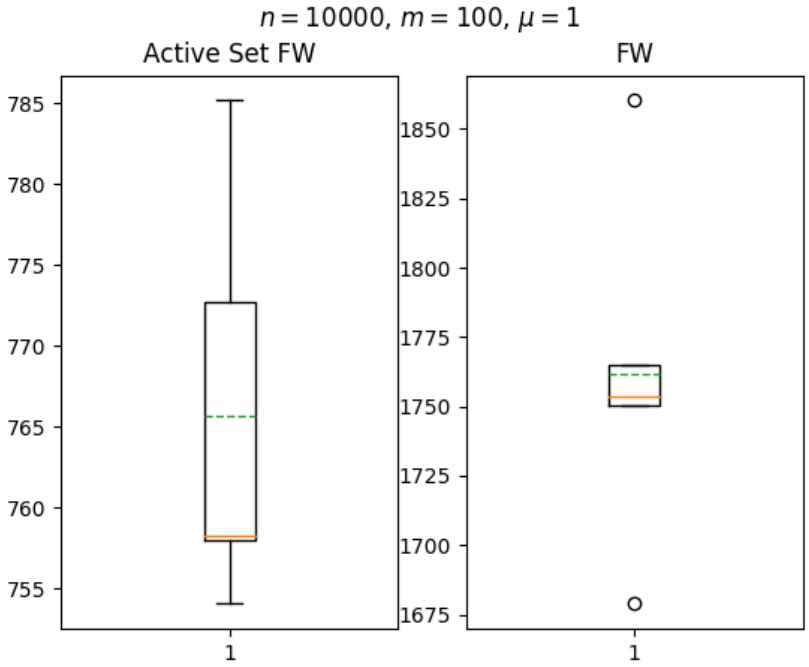



(b) Detailed Box-Plots

Figure 3.5: CPU-TIME Box-Plots $n = 10000,\ m = 100,\ \mu = 0.1$

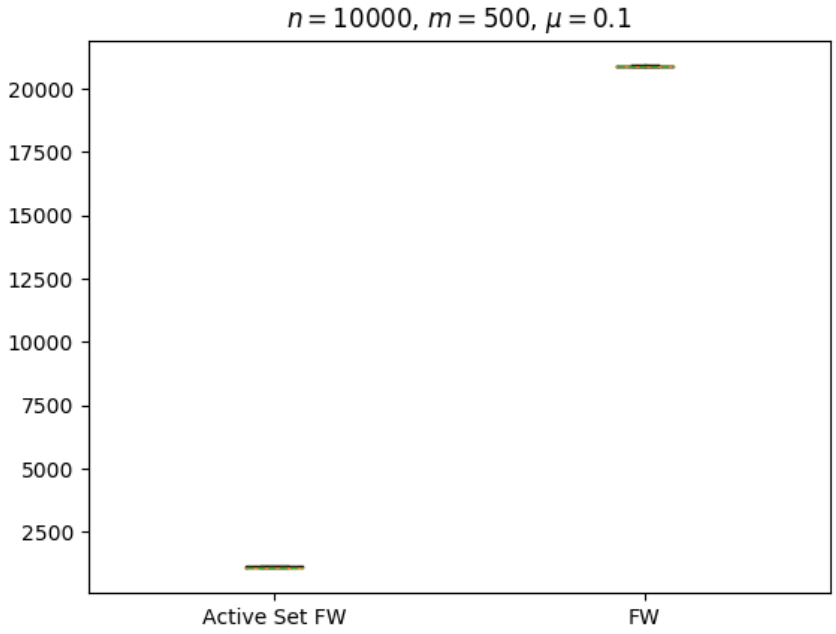



(a) Box-Plots

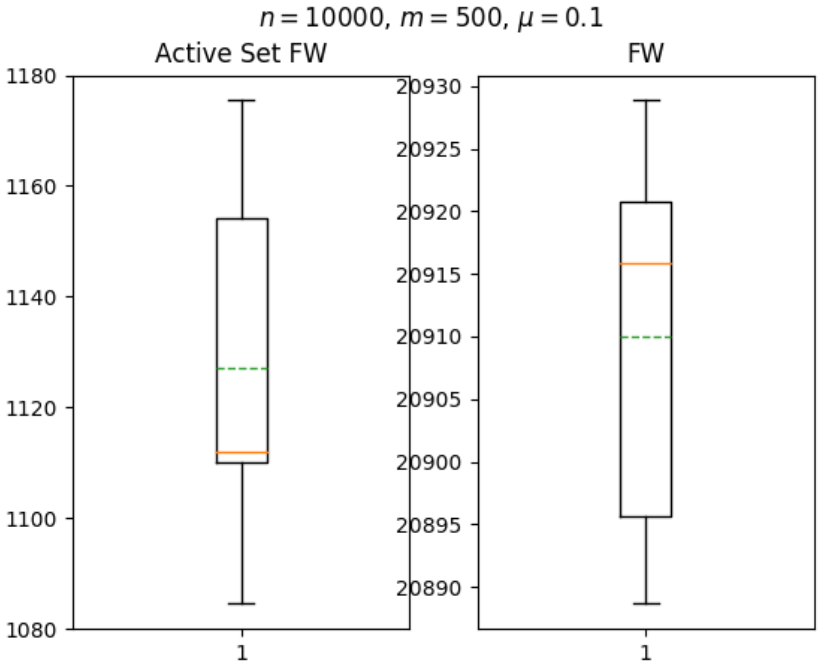



(b) Detailed Box-Plots

Figure 3.6: CPU-TIME Box-Plots $n = 10000,\ m = 100,\ \mu = 1$

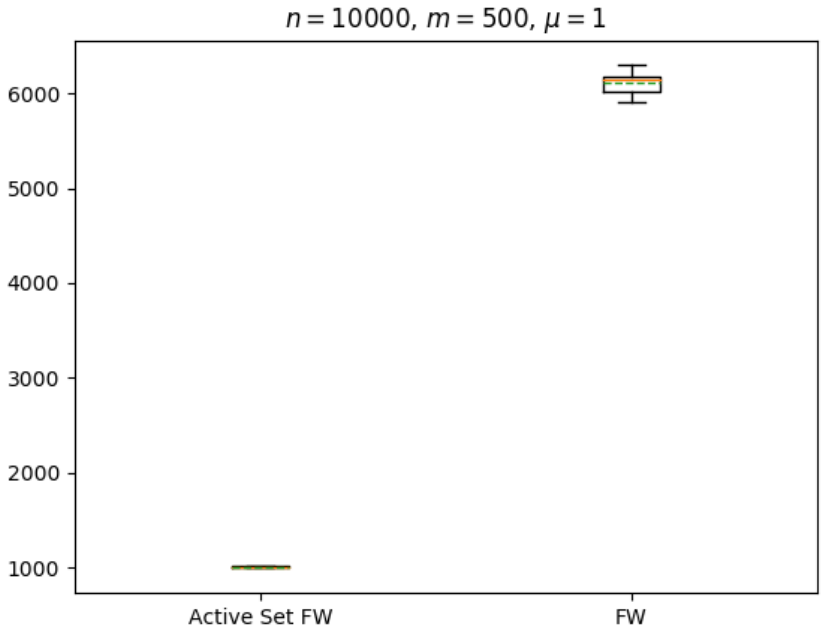



(a) Box-Plots

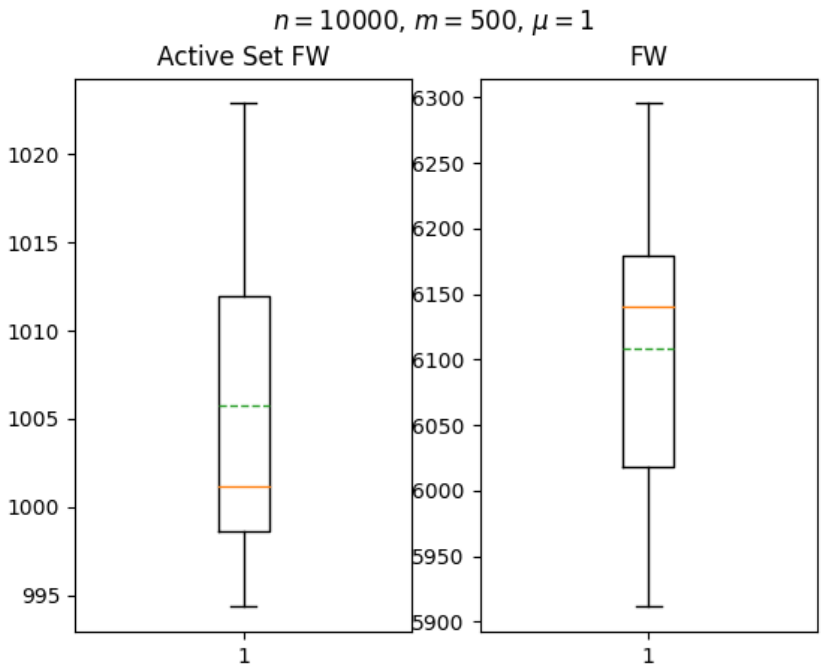



(b) Detailed Box-Plots

Figure 3.7: CPU-TIME Box-Plots $n = 10000,\ m = 500,\ \mu = 0.1$




(a) Box-Plots




(b) Detailed Box-Plots

Figure 3.8: CPU-TIME Box-Plots $n = 10000,\ m = 500,\ \mu = 1$

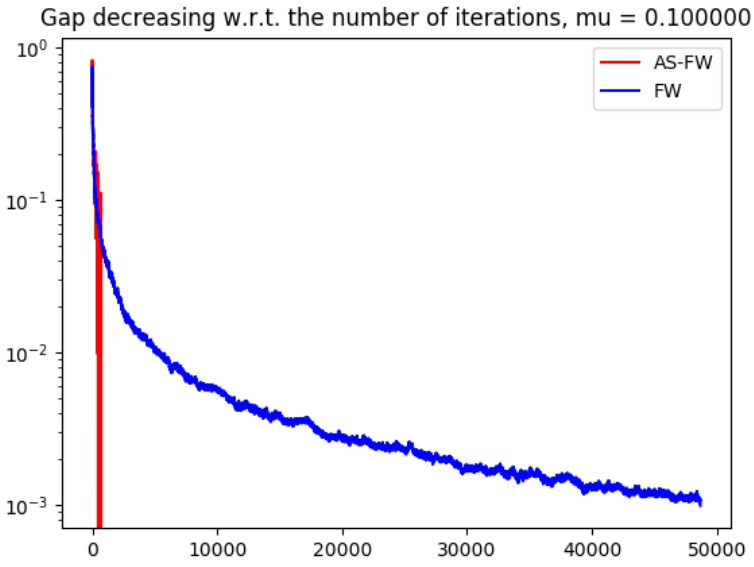



Figure 3.9: Iterations Plot $n = 5000, m = 50, \mu = 0.1$

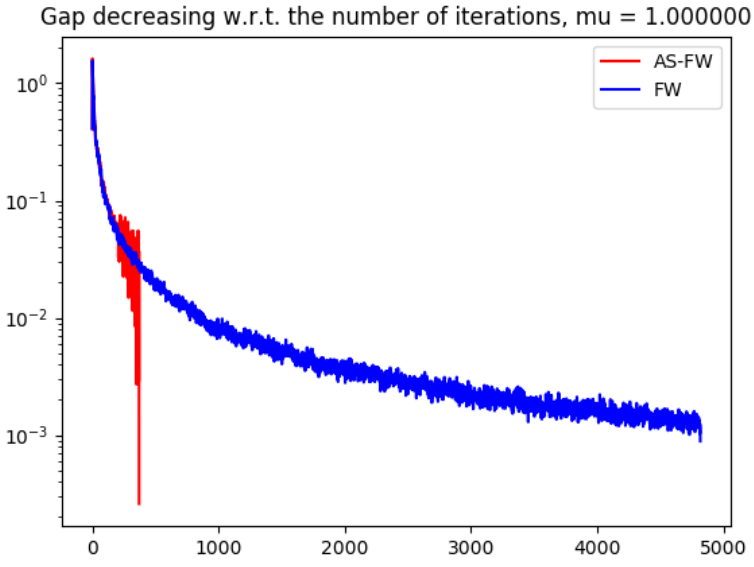



Figure 3.10: Iterations Plot $n = 5000, m = 50, \mu = 1$

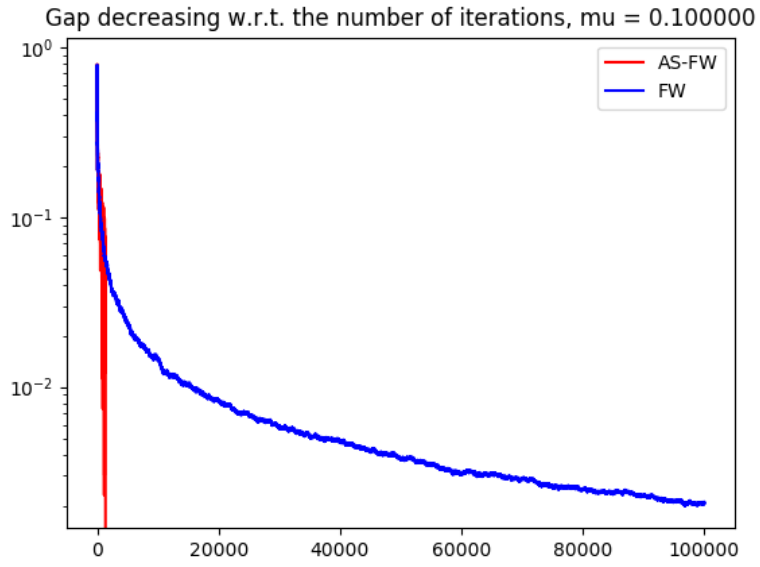



Figure 3.11: Iterations Plot $n = 5000, m = 250, \mu = 0.1$

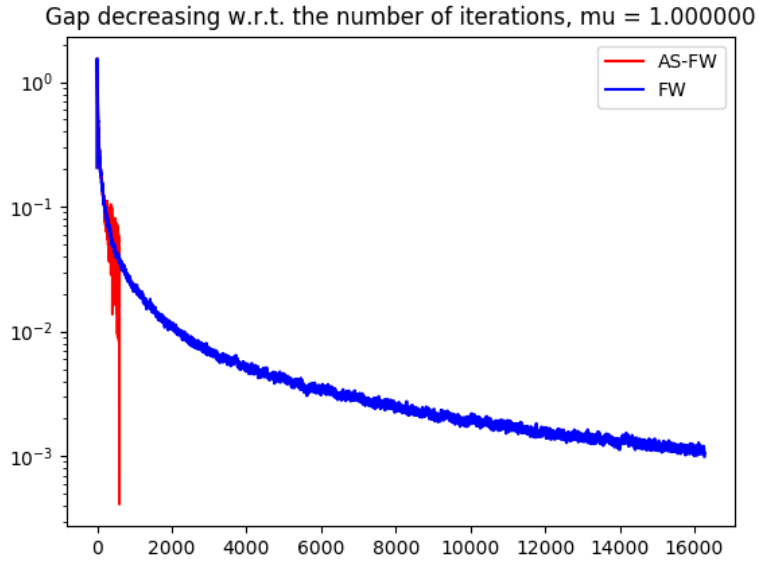



Figure 3.12: Iterations Plot $n = 5000, m = 250, \mu = 1$

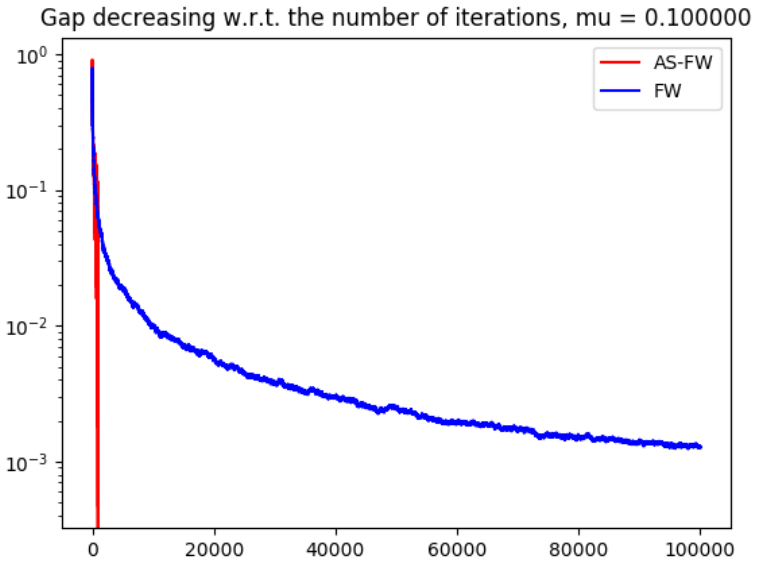



Figure 3.13: Iterations Plot $n = 10000, m = 100, \mu = 0.1$

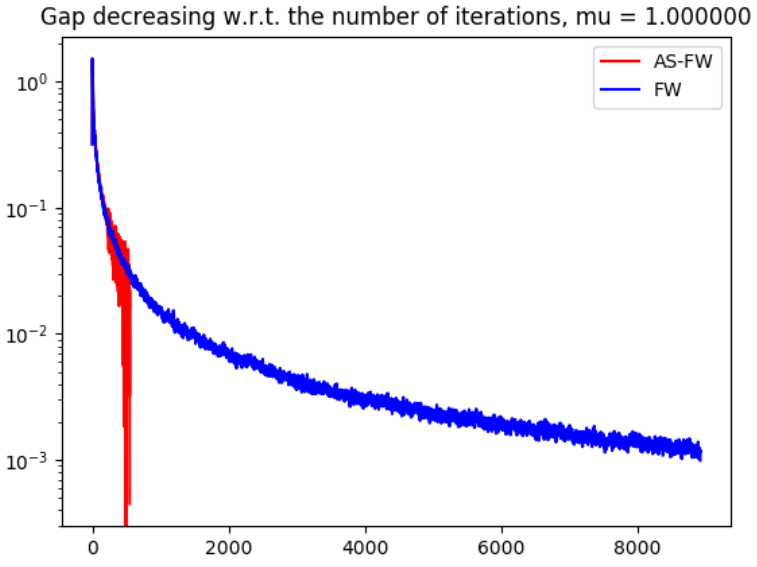



Figure 3.14: Iterations Plot $n = 10000, m = 100, \mu = 1$

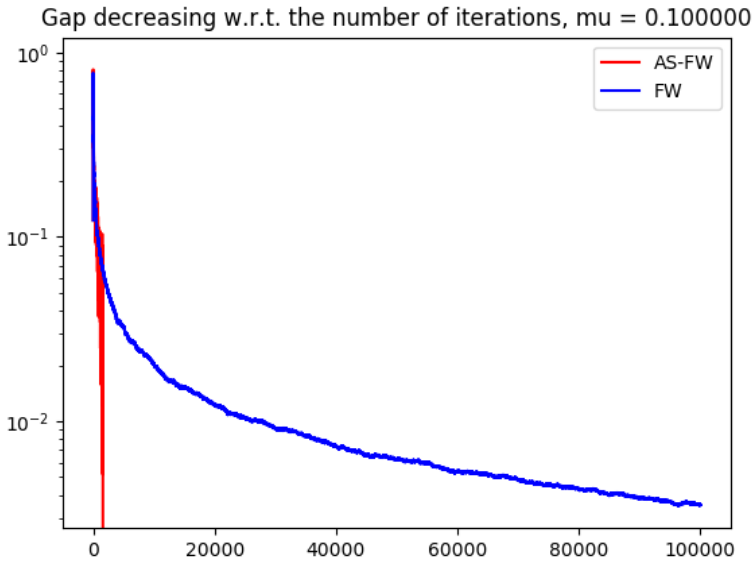



Figure 3.15: Iterations Plot $n = 10000, m = 500, \mu = 0.1$

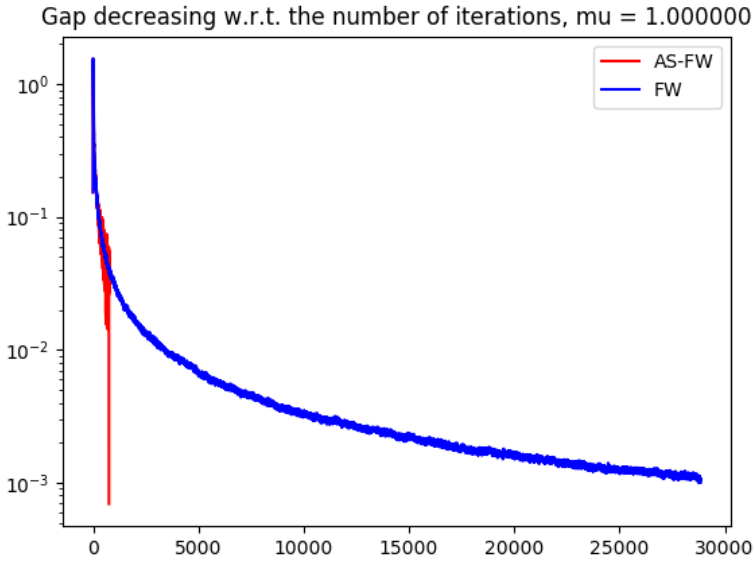



Figure 3.16: Iterations Plot $n = 10000, m = 500, \mu = 1$

# Chapter 4

# Conclusions

In this last chapter, we review the main points of this thesis.
First, we have underlined the importance and the frequency with which saddle point problems occur in applications. In particular, we have shown examples of application fields in which these problems arise, such as game theory and robust optimization.
Then, we have formally described the problem we have studied, that is the search for a saddle point of a convex-concave differentiable function with Lipschitz gradient, defined on the Cartesian product of two convex and compact sets.
After that, we have presented the KKT conditions for this problem and we have proved that a sufficient condition for a point to be a saddle point is that the gradient of our function vanishes there.
Whereupon we have studied the Frank-Wolfe algorithm for the minimization of a convex function over a convex set and we have also described its variants (away-step FW and pairwise FW) to deal with the case where the feasible set is a polytope. The purpose of this study was to better explain how these methodologies could then be adapted to the case of saddle point problems as presented by G. Gidel, T. Jebara and S. Lacoste-Juliene in their quite recent paper [7].
Then, we went on to study the particular case in which the feasible set is given by the Cartesian product of two unit simplices. In this context, we have supposed that the saddle point solution of our problem is very sparse. Hence, we extended the new active-set strategy proposed by A. Cristofari, M. De Santis, S. Lucidi and F. Rinaldi in their very recent paper [4] to our problem. In short, we were interested in finding as soon as possible the zero components (called *active* components) of the saddle point, in order to solve the problem in a smaller feasible set and save CPU-TIME. For this reason, we defined some estimates of the active sets and we have proved that in a neighborhood of the saddle point, these estimates identify exactly the active components.

After that, we have proposed a new personal algorithm that combines the active-set strategy to the saddle point extension of Frank-Wolfe methodology and we have given the proof that, under some appropriate assumptions, it converges to the saddle point.
Finally, we have implemented our algorithm in Python language and we have tested it on a simple bilinear Toy Problem, built so as to satisfy our assumptions and in such a way to make known the saddle point we wish to find. We have shown that this new algorithm allowed us to find the saddle point saving a lot of CPU-TIME.

# Appendices

# Appendix A

# Proofs of Convergence Theorems

In this Appendix we present the principal details of the paper [7] and the scheme that leads to the convergence proof of Algorithm 2 and, as a consequence, of our Algorithm 9.

## A.1 Definitions

First of all, we give all the definitions of the affine invariant constants of a convex function and their extension to the the convex-concave framework. In particular, we underline how they can be updated in our active set context.

### A.1.1 Convex Functions

In this first section, we recall some basic concepts of convex sets and functions (see also [1], [2] or [14]).

**Definition A.1** (Convex Set)**.** Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a subset. We say that $\mathcal{X}$ is a *convex set* if

$$\lambda x + (1-\lambda)y \in \mathcal{X} \quad \text{for all } x, y \in \mathcal{X}, \text{ for all } \lambda \in [0,1]$$

**Definition A.2** (Convex Function, Concave Function)**.** Let $\mathcal{X} \in \mathbb{R}^n$ be a convex set and let $f : \mathcal{X} \to \mathbb{R}$ a function. $f$ is said to be a *convex function* if

$$f(\lambda x + (1-\lambda)y) \quad \leq \quad \lambda f(x) + (1-\lambda) f(y)$$

for all $x, y \in \mathcal{X}$ and $\lambda \in [0,1]$ $f$ is said to be *concave* if $-f$ is convex. Moreover, $f$ is said to be *strictly convex* if for all $x, y \in \mathcal{X}$ and $\lambda \in [0,1]$

$$f(\lambda x + (1-\lambda)y) \quad < \quad \lambda f(x) + (1-\lambda) f(y)$$

and *strictly concave* if $-f$ is strictly convex.

We recall that when the objective function has some regularity, convexity can be characterized in the following way.

**Proposition A.1.** Let $\mathcal{X} \subseteq \mathbb{R}^n$, $\mathcal{X} \neq \emptyset$ a convex set and let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable over an open set that contains $\mathcal{X}$. Then

- $f$ is convex over $\mathcal{X}$ (that is $f|_{\mathcal{X}} : \mathcal{X} \to \mathbb{R}$ is convex) if and only if

$$f(z) \geq f(x) + \nabla f(x)^\top (z - x) \quad \text{for all} \quad x, z \in \mathcal{X}$$

- $f$ is strictly convex over $\mathcal{X}$ if and only if

$$f(z) > f(x) + \nabla f(x)^\top (z - x) \quad \text{for all} \quad x, z \in \mathcal{X},\ x \neq z$$

**Definition A.3** (Strongly Convex Function)**.** Let $\mathcal{X} \in \mathbb{R}^n$ be a convex set and let $f : \mathcal{X} \to \mathbb{R}$ a convex function. We say that $f$ is a $\mu$-strongly convex function if

$$x \mapsto f(x) - \frac{\mu}{2}\|x\|^2 \tag{A.1}$$

is convex.

**Definition A.4** (Convex-Concave Function)**.** Let $\mathcal{X}$ and $\mathcal{Y}$ be two convex sets. We say that a function

$$\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R} \tag{A.2}$$

is convex-concave if

$$x \mapsto \mathscr{L}(x, y) \quad \text{is convex for all } y \in \mathcal{Y} \tag{A.3}$$

and

$$y \mapsto \mathscr{L}(x, y) \quad \text{is concave for all } x \in \mathcal{X} \tag{A.4}$$

**Definition A.5** (Strongly Convex-Concave Function)**.** Let $\mathcal{X}$ and $\mathcal{Y}$ be two convex sets and let $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a convex-concave function. We say that $\mathscr{L}$ is uniformly $(\mu_x, \mu_y)$-strongly convex-concave if

$$(x, y) \mapsto \mathscr{L}(x, y) - \frac{\mu_x}{2}\|x\|^2 + \frac{\mu_y}{2}\|y\|^2 \tag{A.5}$$

is convex-concave.

### A.1.2 Relative Interior

In [7] it is proved that Algorithm 2 converges to the saddle point $(x^*, y^*)$ of the problem

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathscr{L}(x, y) \tag{A.6}$$

under the assumption that $(x^*, y^*)$ belongs to the relative interior of $\mathcal{X} \times \mathcal{Y}$. Then, let us give the following formal definitions (see also [1] or [2]).

**Definition A.6.** Let $\mathcal{X}$ be a subset of $\mathbb{R}^n$. The affine hull of $\mathcal{X}$, denoted $\text{aff}(\mathcal{X})$, is the intersection of all affine sets containing $\mathcal{X}$, where an affine set is defined as a translation of a vector subspace. Equivalently,

$$\text{aff}(\mathcal{X}) := \left\{ \sum_{i=1}^{k} \alpha_i x_i \text{ s.t. } k > 0, x_i \in \mathcal{X}, \alpha_i \in \mathbb{R}, \sum_{i=1}^{k} \alpha_i = 1 \right\} \tag{A.7}$$

**Definition A.7.** Let $\mathcal{X}$ be a non-empty convex set. We say that $x \in \mathcal{X}$ is a relative interior point of $\mathcal{X}$ if there exists an open ball $B(x, \rho)$ centered at $x$ such that

$$B(x, \rho) \cap \text{aff}(\mathcal{X}) \subset \mathcal{X} \tag{A.8}$$

The set of all the relative interior points of $\mathcal{X}$ is called the relative interior of $\mathcal{X}$.

**Remark A.2.** If we consider the feasible set $\mathcal{X} = \Delta^1 \subseteq \mathbb{R}^n$, the relative interior of $\mathcal{X}$ can be described as

$$\text{ri}(\Delta^1) := \left\{ x = (x_1, \ldots, x_n) \text{ s.t. } \sum_{i=1}^{n} x_i = 1, x_i > 0 \right\} \tag{A.9}$$

Hence, when we study our problem in the restricted space of the non-active components of the saddle point, we have that $(x^*, y^*)$ lies in the relative interior of that space.

### A.1.3 The Lipschitz Constants

One of the main assumptions in our work is that the objective function is differentiable with $L$-Lipschitz gradient. Let us give the formal definitions.

**Definition A.8.** We say that a function $f : \mathbb{R}^n \to \mathbb{R}$ has a Lipschitz continuous gradient with Lipschitz constant $L$ if

$$\|\nabla f(x) - \nabla f(x')\|_* \leq \|x - x'\| \tag{A.10}$$

for all $x, x' \in \mathcal{X}$, where

$$\|y\|_* := \sup_{x \in \mathbb{R}^n, \|x\| \leq 1} y^\top x \tag{A.11}$$

is the dual norm of $\|\cdot\|$.

Now, consider a convex-concave function $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. In this case, we consider the dual pairing of norms $(\|\cdot\|_{\mathcal{X}}, \|\cdot\|_{\mathcal{X}^*})$ on $\mathcal{X}$ and similarly $(\|\cdot\|_{\mathcal{Y}}, \|\cdot\|_{\mathcal{Y}^*})$ on $\mathcal{Y}$. We also define the norm on the product space $\mathcal{X} \times \mathcal{Y}$ as the $\ell_1$-norm on the components, that is

$$\|(x, y)\|_{\mathcal{X} \times \mathcal{Y}} := \|x\|_{\mathcal{X}} + \|y\|_{\mathcal{Y}}. \tag{A.12}$$

We thus have the dual norm of $\mathcal{X} \times \mathcal{Y}$ is the $\ell_\infty$-norm of the dual norms, that is

$$\|(x, y)\|_{(\mathcal{X} \times \mathcal{Y})^*} = \max(\|x\|_{\mathcal{X}^*}, \|y\|_{\mathcal{Y}^*}) \tag{A.13}$$

**Definition A.9.** Let $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a continuously differentiable function. We say that $\mathscr{L}$ has a $L$-Lipschitz continuous gradient if

$$\|\nabla\mathscr{L}(x,y) - \nabla\mathscr{L}(x',y')\|_{(\mathcal{X}\times\mathcal{Y})^*} \leq L\|(x,y) - (x',y')\|_{\mathcal{X}\times\mathcal{Y}} \tag{A.14}$$

Moreover, $L$ is called full Lipschitz constant.

**Definition A.10.** The *partial Lipschitz constants* $L_{XX}, L_{YY}, L_{XY}$ and $L_{YX}$ of the gradient of the function $\mathscr{L}$ are the constants such that for all $x, x' \in \mathcal{X}$ and $y, y' \in \mathcal{Y}$,

$$\|\nabla_x\mathscr{L}(x,y) - \nabla_x\mathscr{L}(x',y)\|_{\mathcal{X}^*} \leq L_{XX}\|x - x'\|_{\mathcal{X}} \tag{A.15}$$
$$\|\nabla_x\mathscr{L}(x,y) - \nabla_x\mathscr{L}(x,y')\|_{\mathcal{X}^*} \leq L_{XY}\|y - y'\|_{\mathcal{Y}} \tag{A.16}$$
$$\|\nabla_y\mathscr{L}(x,y) - \nabla_y\mathscr{L}(x',y)\|_{\mathcal{Y}^*} \leq L_{YX}\|x - x'\|_{\mathcal{X}} \tag{A.17}$$
$$\|\nabla_y\mathscr{L}(x,y) - \nabla_y\mathscr{L}(x,y')\|_{\mathcal{Y}^*} \leq L_{YY}\|y - y'\|_{\mathcal{Y}} \tag{A.18}$$

**Remark A.3.** Clearly, the partial Lipschitz constants can always be taken to be smaller than the full Lipschitz constant for the gradient of $\mathscr{L}$, that is

$$L \geq \max\{L_{XX}, L_{XY}, L_{YX}, L_{YY}\} \tag{A.19}$$

### A.1.4 Curvature constant

The convergence analysis of Frank-Wolfe type algorithms is usually based on a measure of non-linearity of the objective function over the feasible set.

**Definition A.11.** Let $f : \mathcal{X} \to \mathbb{R}$ be a convex function. We define the curvature $C_f$ of $f$ as

$$C_f := \sup_{\substack{x,s,v\in\mathcal{X} \\ \gamma>0 \text{ s.t.} \\ x_\gamma := x+\gamma d\in\mathcal{X} \\ \text{with } d:=s-v}} \frac{2}{\gamma^2}(f(x_\gamma) - f(x) - \gamma < d, \nabla f(x) >) \tag{A.20}$$

For linear functions $f$, it holds that $C_f = 0$.
The boundness of such constant guarantees that the deviation of $f$ at $x_\gamma$ from the linearization of $f$ given by $\nabla f(x)$ is also bounded.
We extend the definition above to the case of a convex-concave function.
First of all, let us define the sets $\mathcal{F}$ and $\mathcal{G}$ of the marginal convex functions.

$$\mathcal{F} := \{x' \mapsto \mathscr{L}(x', y)\}_{y\in\mathcal{Y}} \quad \text{and} \quad \mathcal{G} := \{y' \mapsto -\mathscr{L}(x, y')\}_{x\in\mathcal{X}} \tag{A.21}$$

Then,

**Definition A.12.** Let $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a convex-concave function. We define the curvature pair $(C_{\mathscr{L}_x}, C_{\mathscr{L}_y})$ of $\mathscr{L}$ as

$$(C_{\mathscr{L}_x}, C_{\mathscr{L}_y}) := \left(\sup_{f\in\mathcal{F}} C_f, \sup_{g\in\mathcal{G}} C_g\right) \tag{A.22}$$

and the curvature of $\mathscr{L}$ as

$$C_{\mathscr{L}} = \frac{C_{\mathscr{L}_x} + C_{\mathscr{L}_y}}{2} \tag{A.23}$$

In particular, in our framework, we reduce our study only over the subset of the non-active components of the saddle point of problem

$$\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathscr{L}(x,y) \tag{A.24}$$

Hence, we consider a quite different curvature constant. Indeed, if we define the two sets

$$\bar{\mathcal{X}} := \{x_i \text{ such that } x \in \mathcal{X} \text{ and } i \in \bar{N}_1(x^*, y^*)\} \subseteq \mathbb{R}^{|\bar{N}_1(x^*,y^*)|} \tag{A.25}$$

$$\bar{\mathcal{Y}} := \{y_j \text{ such that } y \in \mathcal{Y} \text{ and } j \in \bar{N}_2(x^*, y^*)\} \subseteq \mathbb{R}^{|\bar{N}_2(x^*,y^*)|} \tag{A.26}$$

we study the curvature constant of the function

$$\begin{aligned} \bar{\mathscr{L}}: \quad \bar{\mathcal{X}} \times \bar{\mathcal{Y}} &\to \quad \mathbb{R} \\ (\bar{x}, \bar{y}) &\mapsto \mathscr{L}(x,y) \end{aligned} \tag{A.27}$$

where

$$x_i = \begin{cases} \bar{x}_i & \text{if } i \in \bar{N}_1(x^*, y^*) \\ 0 & \text{if } i \in \bar{A}_1(x^*, y^*) \end{cases} \tag{A.28}$$

and

$$y_j = \begin{cases} \bar{y}_j & \text{if } j \in \bar{N}_2(x^*, y^*) \\ 0 & \text{if } j \in \bar{A}_2(x^*, y^*) \end{cases} \tag{A.29}$$

Clearly, when we compute the suprema in the definition of $C_{\bar{\mathscr{L}}}$, we are considering fewer cases than in the definition of $C_{\mathscr{L}}$. For this reason, it yields $C_{\bar{\mathscr{L}}} \leq C_{\mathscr{L}}$.
For a complete description of this constant we refer to [10].

### A.1.5 Interior strong convexity constant

Similarly, we define an affine invariant measure of strong convexity for the points in the relative interior of a convex set $\mathcal{X}$.

**Definition A.13.** Let $x_c$ be a point in the relative interior of $\mathcal{X}$. The *interior strong convexity constant* for $f$ with respect to the reference point $x_c$ is defined as

$$\mu_f^{x_c} := \inf_{\substack{x\in\mathcal{X}\setminus\{x_c\} \\ s=\bar{s}(x,x_c,\mathcal{X}) \\ \gamma\in(0,1] \\ z=x+\gamma(s-x)}} \frac{2}{\gamma^2}(f(z) - f(x) - < z-x, \nabla f(x) >) \tag{A.30}$$

where $s$ is the point where the ray from $x$ to the reference point $x_c$ pinches the boundary of the set $\mathcal{X}$, i.e. $\bar{s}(x, x_c, \mathcal{X}) := \text{ray}(x, x_c) \cap \partial\mathcal{X}$, where $\partial\mathcal{X}$ is the boundary of the convex set $\mathcal{X}$.

This new constant gives us some informations about the feasible set $\mathcal{X}$ and the convex function $f$. In particular, while the curvature constant provides an upper bound for the deviation of $f$ from its linearization, the interior strong convexity constant gives a lower bound. Indeed, by definition, we get

$$f(z) - f(x) - < \nabla f(x), z - x > \geq \frac{\gamma^2}{2} \mu_f^{x_c} \tag{A.31}$$

Also this constant can be extended to the convex-concave context.

**Definition A.14.** The SP-FW interior strong convex-concavity constants with respect to the reference point $(x_c, y_c)$ are defined as:

$$(\mu_{\mathscr{L}}^{x_c}, \mu_{\mathscr{L}}^{y_c}) := \left( \inf_{f \in \mathcal{F}} \mu_f^{x_c}, \inf_{g \in \mathcal{G}} \mu_g^{y_c} \right) \tag{A.32}$$

Moreover, we define the smallest quantity of both

$$\mu_{\mathscr{L}}^{\text{int}} = \min\{\mu_{\mathscr{L}}^{x_c}, \mu_{\mathscr{L}}^{y_c}\} \tag{A.33}$$

As described for the curvature constant, in the active-set framework, we refer to the strong interior constant of the function $\bar{\mathscr{L}}$ defined in (A.27). In this case, since we consider the infima in a smaller set, we have $\mu_{\mathscr{L}}^{\text{int}} \leq \mu_{\bar{\mathscr{L}}}^{\text{int}}$ We refer to [12] for a detailed description of this constant.

### A.1.6 The bilinearity coefficient

In the convergence proof, it is necessary to relate the gradient at the point $(x^k, y^k)$ with the one at the point $(x^k, y^*)$ and the one at the point $(x^*, y^k)$. For this reason, we define the following quantities.

**Definition A.15.** Let $\mathscr{L}$ be a strongly convex-concave function, and let $(x^*, y^*)$ be its unique saddle point. Let $\mathscr{L}^* := \mathscr{L}(x^*, y^*)$. We define the *bilinearity coefficients* $(M_{XY}, M_{YX})$ as

$$M_{XY} := \sup_{\substack{y \in \mathcal{Y} \\ x,s,v \in \mathcal{X} \\ d=s-v}} \left\langle d, \frac{\nabla_x \mathscr{L}(x, y^*) - \nabla_x \mathscr{L}(x, y)}{\sqrt{\mathscr{L}^* - \mathscr{L}(x^*, y)}} \right\rangle \tag{A.34}$$

and

$$M_{YX} := \sup_{\substack{x \in \mathcal{X} \\ y,s,v \in \mathcal{Y} \\ d=s-v}} \left\langle d, \frac{\nabla_y \mathscr{L}(x, y) - \nabla_y \mathscr{L}(x^*, y)}{\sqrt{\mathscr{L}(x, y^*) - \mathscr{L}^*}} \right\rangle \tag{A.35}$$

We also define the *global bilinearity coefficient as*

$$M_{\mathscr{L}} := \max\{M_{XY}, M_{YX}\} \tag{A.36}$$

### A.1.7 Suboptimality Functions

To establish convergence, we need to define some quantities of interest. In classical convex optimization, when we deal with the problem

$$\min_{x\in\mathcal{X}} f(x), \tag{A.37}$$

we usually define the suboptimality error as

$$f(x^k) - \min_{x\in\mathcal{X}} f(x) \tag{A.38}$$

and proving that this quantity goes to 0 is enough to establish convergence. However, when we study the saddle point problem

$$\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathscr{L}(x,y) \tag{A.39}$$

the quantity $\mathscr{L}(x^k,y^k) - \mathscr{L}(x^*,y^*)$ is no longer non-negative and it can be zero for an infinite number of points without them being saddle points.
For this reason, we need to introduce a new suboptimality error sequence.

**Definition A.16.** We define the *primal suboptimality* as

$$h^k := \mathscr{L}(x^k,\hat{y}^k) - \mathscr{L}(\hat{x}^k,y^k) \tag{A.40}$$

where $\hat{x}^k \in \arg\min_{x\in\mathcal{X}} \mathscr{L}(x.y^k)$ and $\hat{y}^k \in \arg\max_{y\in\mathcal{Y}} \mathscr{L}(x^k,y)$

Then, to get a convergence rate, we have to upper bound this primal suboptimality, but working with the moving quantities $\hat{x}_k$ and $\hat{y}_k$ is too hard. To overcome this problem, we define one more suboptimality error sequence.

**Definition A.17.** We define the *second primal suboptimality* as

$$w^k := \mathscr{L}(x^k,y^*) - \mathscr{L}(x^*,y^k) \tag{A.41}$$

**Remark A.4.** From the above definitions,

$$\mathscr{L}(x^k,\hat{y}^k) \geq \mathscr{L}(x^k,y^*) \quad \text{and} \quad \mathscr{L}(x^*,y^k) \leq \mathscr{L}(\hat{x}^k,y^k) \tag{A.42}$$

Then, we get $w^k \leq h^k$.

Clearly, we need to establish that there is some relation between these quantities.
To show this, we need to introduce two more constants.

**Definition A.18.** We define

$$P_{\mathcal{X}} := \sup_{x\in\mathcal{X}} \frac{< \nabla_x \mathscr{L}(x,\hat{y}(x)), x - x^* >}{\sqrt{\mathscr{L}(x,y^*) - \mathscr{L}(x^*,y^*)}} \tag{A.43}$$

and

$$P_{\mathcal{Y}} := \sup_{y \in \mathcal{Y}} \frac{< \nabla_y \mathscr{L}(\hat{x}(y), y), y - y^* >}{\sqrt{\mathscr{L}(x^*, y^*) - \mathscr{L}(x^*, y)}} \tag{A.44}$$

where

$$\hat{y}(x) := \arg\max_{y \in \mathcal{Y}} \mathscr{L}(x, y) \quad \text{and} \quad \hat{x}(y) := \arg\min_{x \in \mathcal{X}} \mathscr{L}(x, y) \tag{A.45}$$

Moreover, we define

$$P_{\mathscr{L}} := \max\{P_{\mathcal{X}}, P_{\mathcal{Y}}\} \tag{A.46}$$

Finally, we recall that Algorithm 2 stops as soon as the following gap function reaches a certain tolerance.

**Definition A.19.** Let

$$r^k := \begin{pmatrix} \nabla_x \mathscr{L}(x^k, y^k) \\ -\nabla_y \mathscr{L}(x^k, y^k) \end{pmatrix} \tag{A.47}$$

Then, we define the *gap function* as

$$g^k :=< d_x^k, -r_x^k > + < d_y^k, -r_y^k >= g_x^k + g_y^k \tag{A.48}$$

Hence, we need to show that there is also a strong relation between this gap function and the primal suboptimalities.

## A.2 Important bounds

In this section, we list some bounds that allow us to estimate the geometric quantities described above. All the proofs of this results can be found in [7]. First, we have that each convex-concave function defined on the product of two compact sets has a finite curvature constant if its gradient is Lipschitz continuous.

**Proposition A.5.** Let $\mathscr{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a differentiable convex-concave function. If $\mathcal{X}$ and $\mathcal{Y}$ are compact and $\nabla\mathscr{L}$ is Lipschitz continuous, then the curvature of $\mathscr{L}$ is bounded by

$$C_{\mathscr{L}} \leq \frac{L_{XX} D_{\mathcal{X}}^2 + L_{YY} D_{\mathcal{Y}}^2}{2} \tag{A.49}$$

where $D_{\mathcal{X}}$ and $D_{\mathcal{Y}}$ are the respective diameter of $\mathcal{X}$ and $\mathcal{Y}$, that is

$$D_{\mathcal{X}} := \sup_{x, x' \in \mathcal{X}} \|x - x'\| \quad \text{and} \quad D_{\mathcal{Y}} := \sup_{y, y' \in \mathcal{Y}} \|y - y'\| \tag{A.50}$$

Now, we give some results that relate a strong convex-concave objective function to the saddle point problem analysis.

**Proposition A.6.** Let $\mathscr{L}$ be a uniformly $(\mu_{\mathcal{X}}, \mu_{\mathcal{Y}})$-strongly convex-concave function and $(x^*, y^*)$ be the saddle point of $\mathscr{L}$. Then we have for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$

$$\sqrt{\mathscr{L}(x, y^*) - \mathscr{L}^*} \geq \|x - x^*\| \sqrt{\frac{\mu_{\mathcal{X}}}{2}} \tag{A.51}$$

and

$$\sqrt{\mathscr{L}^* - \mathscr{L}(x^*, y)} \geq \|y^* - y\| \sqrt{\frac{\mu_{\mathcal{Y}}}{2}} \tag{A.52}$$

**Proposition A.7.** Let $\mathscr{L}$ be a convex-concave function. If the reference point $(x_c, y_c)$ belongs to the relative interior of $\mathcal{X} \times \mathcal{Y}$ and if the function $\mathscr{L}$ is strongly convex-concave with a strong convex-concavity constant $\mu > 0$, then $\mu_{\mathscr{L}}^{\text{int}}$ is lower bounded away from zero. More precisely,

$$\mu_{\mathscr{L}}^{x_c} \geq \mu_{\mathcal{X}} \delta_x^2 \quad \text{and} \quad \mu_{\mathscr{L}}^{y_c} \geq \mu_{\mathcal{Y}} \delta_y^2 \tag{A.53}$$

where

$$\delta_x := \min_{s_x \in \partial \mathcal{X}} \|s_x - x_c\| \tag{A.54}$$

and

$$\delta_y := \min_{s_y \in \partial \mathcal{Y}} \|s_y - y_c\| \tag{A.55}$$

**Proposition A.8.** If $\mathcal{X}$ and $\mathcal{Y}$ are compact, $\nabla \mathscr{L}$ is Lipschitz continuous and $\mathscr{L}$ is uniformly strongly convex-concave with constant $(\mu_{\mathcal{X}}, \mu_{\mathcal{Y}})$, then

$$M_{XY} \leq \frac{2}{\mu_{\mathcal{Y}}} L_{XY} D_{\mathcal{X}} \tag{A.56}$$

and

$$M_{YX} \leq \frac{2}{\mu_{\mathcal{X}}} L_{YX} D_{\mathcal{Y}} \tag{A.57}$$

where $L_{XY}$ and $L_{YX}$ are the partial Lipschitz constants.

**Proposition A.9.** For any $(\mu_{\mathcal{X}}, \mu_{\mathcal{Y}})$-uniformly convex-concave function $\mathscr{L}$,

$$P_{\mathcal{X}} \leq \sqrt{\frac{2}{\mu_{\mathcal{X}}}} \sup_{z \in \mathcal{X} \times \mathcal{Y}} \|\nabla_x \mathscr{L}(z)\|_{\mathcal{X}^*} \tag{A.58}$$

and

$$P_{\mathcal{Y}} \leq \sqrt{\frac{2}{\mu_{\mathcal{Y}}}} \sup_{z \in \mathcal{X} \times \mathcal{Y}} \|\nabla_y \mathscr{L}(z)\|_{\mathcal{Y}^*} \tag{A.59}$$

Finally, we give the fundamental relation between the primal suboptimalities.

**Proposition A.10.** For any $(\mu_{\mathcal{X}}, \mu_{\mathcal{Y}})$-uniformly convex-concave function $\mathscr{L}$,

$$h^k \leq P_{\mathscr{L}} \sqrt{2w^k} \tag{A.60}$$

and

$$P_{\mathscr{L}} \leq \sqrt{2} \sup_{z \in \mathcal{X} \times \mathcal{Y}} \left\{ \frac{\|\nabla_x \mathscr{L}(z)\|_{\mathcal{X}^*}}{\sqrt{\mu_{\mathcal{X}}}}, \frac{\|\nabla_y \mathscr{L}(z)\|_{\mathcal{Y}^*}}{\sqrt{\mu_{\mathcal{Y}}}} \right\} \tag{A.61}$$

This result and Remark A.4 guarantee that the convergence of one of the primal suboptimaly implies also the convergence of the other one.

## A.3 Convergence Proof

In this last section we list the results that lead to the convergence of the algorithms.
The first lemmas guarantees that when the gap function stays under a certain tolerance, even the suboptimalities are bounded.

**Lemma A.11.** For all $k \in \mathbb{N}$, $x \in \mathcal{X}$, $y \in \mathcal{Y}$

$$g^k \geq h^k \geq w^k \tag{A.62}$$

Moreover,

**Lemma A.12.** If $\mathscr{L}$ is a strongly convex-concave function, then for any $(x^k, y^k) \in \mathcal{X} \times \mathcal{Y}$

$$w^k \leq \frac{(g^k)^2}{2\mu_{\mathscr{L}}^{\text{int}}} \tag{A.63}$$

where $\mu_{\mathscr{L}}^{\text{int}} = \min\{\mu_{\mathscr{L}}^{x^*}, \mu_{\mathscr{L}}^{y^*}\}$

On the other hand, also the gap function is bounded by a function of the second primal suboptimality.

**Theorem A.13.** *If $\mathscr{L}$ is strictly convex-concave and has a finite curvature constant, then for any $z^k \in \mathcal{X} \times \mathcal{Y}$*

$$g^k \leq \frac{2}{\nu^{FW}} \max \left\{ \sqrt{C_{\mathscr{L}} w^k}, w^k \right\} \tag{A.64}$$

Then, the convergence proof follows from the results above and the following lemma.

**Lemma A.14.** Let $\mathscr{L}$ be a strongly convex-concave function with a finite curvature constant $C_{\mathscr{L}}$ and a positive interior strong convex-concavity constant $\mu_{\mathscr{L}}^{\text{int}}$. Let us also define the rate multiplier $\nu = 1 - \frac{M_{\mathscr{L}}}{\sqrt{\mu_{\mathscr{L}}^{\text{int}}}}$. If $\nu > 0$, the

suboptimality $w^k$ of the algorithm with step size $\gamma^k = \min\left(\gamma_{\max}, \frac{\nu}{2C_{\mathscr{L}}} g^k\right)$ decreases geometrically as

$$w^{k+1} \leq (1 - \rho_{\mathscr{L}}) w^k \tag{A.65}$$

where

$$\rho_{\mathscr{L}} := \frac{\nu^2}{2} \frac{\mu_{\mathscr{L}}}{C_{\mathscr{L}}} \tag{A.66}$$

Finally, we can state the convergence theorem.

**Theorem A.15.** *Let $\mathscr{L}$ be a strongly convex-concave function with a finite curvature constant $C_{\mathscr{L}}$ and a positive interior strong convex-concavity constant $\mu_{\mathscr{L}}^{int}$. Let us also define the rate multiplier $\nu = 1 - \frac{M_{\mathscr{L}}}{\sqrt{\mu_{\mathscr{L}}^{int}}}$. If $\nu > 0$, the suboptimality $w^k$ of the iterates of the algorithm with step size $\gamma^k = \min\left(\gamma_{max}, \frac{2}{2+k}\right)$ has the following decreasing upper bound*

$$w^k \leq \frac{C}{2+k} \tag{A.67}$$

*where $C = 2\max\{w^0, \frac{2C_{\mathscr{L}}}{2\nu - 1}\}$ Moreover, we can also upper bound the minimum FW-gap observed for $T \geq 1$,*

$$\min_{k \leq T} g^k \leq \frac{5C}{\nu(T+1)} \tag{A.68}$$

# Bibliography


[1] D.P. BERTSEKAS, Convex Optimization Algorithms, Athena Scientific, 2015

[2] D.P. BERTSEKAS, Convex Optimization Theory, Athena Scientific, 2009

[3] B. COX, A. JUDITSKY AND A. NEMIROVSKI, Decomposition techniques for bilinear saddle point problems and variational inequalities with affine monotone operators on domains given by linear minimization oracles, J. Optim. Theory Appl., 172 (2017)

[4] A. CRISTOFARI, M. DE SANTIS, S. LUCIDI AND F. RINALDI, New Active-Set Frank-Wolfe variants for minimization over the simplex and the $\ell_1$-ball, arXiv:1703.07761, 2017

[5] L. DEVROYE, Non-Uniform Random Variate Generation, Springer, 1986

[6] F. FACCHINEI and S. LUCIDI, Quadratically and superlinearly convergent algorithms for the solution of inequality constrained minimization problems, J. Optim. Theory Appl., 85 (1995)

[7] G. GIDEL, T. JEBARA AND S. LACOSTE-JULIEN, Frank-Wolfe Algorithms for Saddle Point Problems, 2016

[8] L. GRIPPO AND M. SCIANDRONE, Metodi di ottimizzazione non vincolata, Springer, 2011

[9] J. H. HAMMOND, Solving asymmetric variational inequality problems and systems of equations with generalized nonlinear programming algorithms, PhD thesis, Massachusetts Institute of Technology, 1984

[10] M. JAGGI, Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization, in ICML, 2013

[11] A. JUDITSKY AND A. NEMIROVSKI, Solving variational inequalities with monotone operators on domains given by linear minimization oracles, Mathematical Programming, 2016

[12] S. LACOSTE-JULIEN AND M. JAGGI, An affine invariant linear convergence analysis for Frank-Wolfe algorithms, arXiv preprint arXiv:1312.7864, 2013

[13] S. LACOSTE-JULIEN AND M. JAGGI, On the Global Linear Convergence of Frank-Wolfe Optimization Variants, in NIPS, 2015

[14] R.T. ROCKAFELLAR, Convex Analysis, Princeton University Press, 1997

[15] B. RUSTEM AND M. HOWE, Algorithms for Worst-Case Design and Applications to Risk Management, Princeton University Press, 2002

[16] P. VIANNEY AND G. VIGERAL, A minmax theorem for concave-convex mappings with no regularity assumptions, Journal of Convex Analysis, Heldermann, 2015

[17] N. XIU AND J. ZHANG, Some recent advances in projection-type methods for variational inequalities, Journal of Computational and Applied Mathematics, 2003