UNIVERSIDADE ESTADUAL DE MARINGÁ

CENTRO DE CIÊNCIAS EXATAS

DEPARTAMENTO DE MATEMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM MATEMÁTICA

(Mestrado)

Application of Newton's and Broyden's methods in linear programming [1]

Aplicação dos métodos de Newton e de Broyden em programação linear

Lucas Moschen

Orientador: Prof. Dr. Francisco Nogueira Calmon Sobral

Maringá - PR

2022

[1] O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Application of Newton's and Broyden's methods in linear programming

Aplicação dos métodos de Newton e Broyden em programação linear

Lucas Moschen

Dissertação apresentada ao Programa de Pós-Graduação em Matemática do Departamento de Matemática, Centro de Ciências Exatas da Universidade Estadual de Maringá, como requisito parcial para a obtenção do título de Mestre em Matemática.
Área de concentração: Matemática Aplicada

Orientador: Prof. Dr. Francisco Nogueira Calmon Sobral

Maringá - PR

2022

Dados Internacionais de Catalogação na Publicação (CIP)
(Biblioteca Setorial BSE-DMA-UEM, Maringá, PR, Brasil)

Moschen, Lucas
M895a Application of Newton´s and Broyden´s methods in linear programming = Aplicação dos métodos de Newton e Broyden em programação linear / Lucas Moschen. -- Maringá, 2022.
67 f. : il.

Orientador: Prof. Dr. Francisco Nogueira Calmon Sobral
Dissertação (Mestrado) - Universidade Estadual de Maringá, Centro de Ciências Exatas, Programa de Pós-Graduação em Matemática - Área de Concentração: Matemática aplicada, 2022.

1. Método de Newton. 2. Método de Broyden. 3. Métodos de pontos interiores. 4. Equações não-suaves. 5. Newton´s method. 6. Broyden´s method. 7. Interior point methods. 8. Nons-mooth equations. I. Sobral, Francisco Nogueira Calmon, orient. II. Universidade Estadual de Maringá. Centro de Ciências Exatas. Programa de Pós-Graduação em Matemática - Área de Concentração: Matemática aplicada. III. Título.

CDD 22.ed. 519.72

Edilson Damasio CRB9-1.123

LUCAS MOSCHEN

APPLICATION OF NEWTON'S AND BROYDEN'S METHODS IN LINEAR PROGRAMMING

Dissertação apresentada ao Programa de Pós-Graduação em Matemática do Departamento de Matemática, Centro de Ciências Exatas da Universidade Estadual de Maringá, como parte dos requisitos necessários para a obtenção do título de Mestre em Matemática tendo a Comissão Julgadora composta pelos membros:

COMISSÃO JULGADORA:

Prof. Dr. Francisco Nogueira Calmon Sobral - UEM (Presidente)
Prof. Dr. Luiz Rafael dos Santos - UFSC
Prof. Dr. Emerson Vitor Castelani - UEM

Aprovado em: 22 de fevereiro de 2022.
Local de defesa: Videoconferência – Google Meet (https://meet.google.com/yrt-bgwq-xoz)

Agradecimentos

Agradeço primeiramente a Deus, por me dar todas as condições necessárias para perseguir meus objetivos.

A todos os meus professores, por suas aulas, explicações e por toda a ajuda, a qual muitas vezes foi além da universidade.

Ao meu orientador, Prof. Dr. Francisco Nogueira Calmon Sobral, por sua intensa disponibilidade, seu auxílio com meus estudos e toda a orientação que me deu em diversos aspectos da vida acadêmica.

À minha família, por todo o suporte, amor, e por estarem sempre inteiramente disponíveis para me ajudar no que fosse necessário.

A todos os meus colegas e amigos, pelo companheirismo e amizade que me concederam ao longo de toda a minha vida.

À minha querida esposa, Mariaclara Rauber, por todo o amor, compreensão e amparo, especialmente nos momentos mais difíceis.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pelo apoio financeiro que me permitiu dedicação integral a esse trabalho.

Resumo

Nessa dissertação, são apresentados os métodos de Newton e de Broyden com o objetivo de estudar a sua utilização na resolução de problemas de programação linear. Nesse sentido, primeiramente a versão padrão de um método de pontos interiores (IPM) primal-dual é introduzida, a qual usa o método de Newton como base para a sua operação. Em seguida, é estudado um IPM com aproximação quasi-Newton proposto recentemente, e é apresentado um resultado original desta pesquisa: a convergência local linear desse método sob algumas fortes hipóteses. Posteriormente, é estudado um sistema de equações não-suaves equivalente às condições de Karush-Kuhn-Tucker (KKT) de um problema de programação linear. São aplicadas variantes não-suaves dos métodos de Newton e de Broyden para resolver tal sistema e são observados os resultados sobre sua convergência. Com a finalidade de obter propriedades de convergência global em relação à resolução desse sistema, é estudado um algoritmo que utiliza métodos locais em sua operação. Considerando um problema de programação linear específico, são aplicadas duas versões desse algoritmo: uma utilizando o método de Newton e a outra utilizando o método de Broyden. A análise dos experimentos numéricos realizados sugere resultados teóricos relacionados à operação desse algoritmo e resulta em uma modificação com melhor desempenho de convergência global no problema selecionado.

Palavras-chave: Método de Newton; Método de Broyden; Métodos de pontos iteriores; Equações não-suaves.

Abstract

In this dissertation, Newton's and Broyden's methods are presented with the aim of studying their use in solving linear programming problems. To this end, first the standard version of a primal-dual interior point method (IPM) is introduced, which uses Newton's method as the basis for its operation. Next, a recently proposed IPM with quasi-Newton approach is studied, and an original result of this research is presented: the linear local convergence of this method under some strong assumptions. Subsequently, a system of nonsmooth equations equivalent to the Karush-Kuhn-Tucker (KKT) conditions of a linear programming problem is studied. Nonsmooth variants of Newton's and Broyden's methods are applied to solve such system and results on its convergence are observed. In order to obtain global convergence properties with respect to solving this system, it is studied an algorithm that uses local methods in its operation. By considering a specific linear programming problem, two versions of this algorithm are applied: one using Newton's method and the other using Broyden's method. The analysis of the numerical experiments suggests theoretical results related to the operation of this algorithm and results in a modification with better global convergence performance on the selected problem.

Keywords: Newton's method; Broyden's method; Interior point methods; Nonsmooth equations.

Contents

Introduction

Consider the task of solving a linear programming problem in the form

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0, \end{aligned} \tag{1}$$

where $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$ and $m < n$. In [17, Section 12.6 and 13.1] it is shown that this task is equivalent to solving the Karush-Kuhn-Tucker (KKT) conditions of this problem, which can be written as

$$\begin{aligned} A^T\lambda + z &= c \\ Ax &= b \\ XZe &= 0 \\ (x, z) &\geq 0, \end{aligned}$$

where $\lambda \in \mathbb{R}^m, z \in \mathbb{R}^n$, $e \in \mathbb{R}^n$ is a vector with all coordinates equal to 1 and $X, Z \in \mathbb{R}^{n \times n}$ are diagonal matrices with the main diagonal containing the vectors x and z, respectively. Therefore, it is possible to formulate a function $F : \mathbb{R}^N \to \mathbb{R}^N$ such that solving problem (1) can be visualized as the task of finding $x^* \in \mathbb{R}^N$ such that $F(x^*) = 0$ while the condition $(x, z) \geq 0$ is guaranteed. Newton's and Broyden's methods are well-known methods to solve a problem in this form when F is continuously differentiable [17, Chapter 11]. Therefore, there are applications of these methods in this sense, such as the creation of Newton-type methods in [5].

A well-known class of methods to solve problem (1) is the interior point methods (IPMs). As it is discussed in [7, 13, 22], it is characterized by using one iteration of Newton's method in order to approximate a solution of the system

$$\begin{aligned} A^T\lambda + z &= c \\ Ax &= b \\ XZe &= \mu e \\ (x, z) &\geq 0 \end{aligned} \tag{2}$$

at each iteration, where μ is a positive constant. Throughout the iterations, these methods brings μ to 0, consequently system (2) becomes more and more similar to the KKT

conditions of problem (1), and then the iterates of the IPMs approach a solution of these conditions, which provide a solution of the problem. At each iteration k, these methods need to solve the linear system

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z_k & 0 & X_k \end{bmatrix} \begin{bmatrix} \Delta x_k \\ \Delta \lambda_k \\ \Delta z_k \end{bmatrix} = \begin{bmatrix} c - A^T\lambda_k - z_k \\ b - Ax_k \\ \sigma_k \mu_k e - X_k Z_k e \end{bmatrix}$$

to obtain the Newton step $(\Delta x_k, \Delta \lambda_k, \Delta z_k)$, where I_n is the identity matrix and $\sigma_k \in (0, 1)$ is a chosen parameter value. However, this can be an expensive task if, for example, the system dimension is large and the Cholesky decomposition, normally used for this type of computation, destroys the sparsity structure of the coefficient matrix. Therefore, Gondzio and Sobral [8] developed an IPM with quasi-Newton approach, which have a strategy to reduce the cost of obtaining the step at each iteration. In this sense, one goal of this work is to study this method and obtain a result about its local convergence.

Throughout the iterations of a primal-dual IPM, a step length is used to guarantee that the variables x and z of the iterates are kept positive at every iteration, while Newton's method approaches to the solution of the three equalities of system (2). This operation can generate difficulties in the proof of some general results. It is possible to study the application of Newton's and Broyden's methods to solve linear programming problems through another angle, without the necessity of dealing with nonnegativity constraints. This is possible through a system of nonsmooth equations equivalent to the KKT conditions of problem (1), as showed in [19, 20], which in this case can be described as

$$\begin{aligned} A^T\lambda + z &= c \\ Ax &= b \\ \min\{x_1, z_1\} &= 0 \\ &\vdots \\ \min\{x_n, z_n\} &= 0. \end{aligned} \tag{3}$$

More details about this system can be seen in Section 3.1 of Chapter 3 of this work. In the sequence, also in Chapter 3, computational experiments and theoretical studies are performed to analyze whether convergence results are valid. Several situations are considered: by varying the method used, the conditions to which the initial iterate was subjected, and the conditions related to the initial approximation B_0 of the Jacobian matrix applied to the initial iterate, when the Broyden method is used.

The application of nonsmooth variants of Newton's and Broyden's methods in order to solve the system of nonsmooth equations (3) does not have good global convergence results. When developing this dissertation, two algorithms were found that had global convergence properties: the one proposed by Ito and Kunisch [9] and the one developed by

Gomes-Ruggiero, Martínez and Santos [6]. The algorithm presented in [9], however, have its global convergence results conditioned to satisfying a certain set of conditions that a general linear programming problem does not satisfy, what prevented the algorithm of being applied. On the other hand, the global convergence properties of the algorithm developed in [6] did not depend on rigorous assumptions, and therefore it can be used.

The algorithm developed by Gomes-Ruggiero, Martínez and Santos [6] matches a local method (such as Newton's or Broyden's methods), which have good performance near to a solution of the objective function, with another algorithm developed in [6], which has the property that, if it does not stop (in general at some stationary point of the merit function considered) then it converges to the solution of (3).

In this work, two versions of this algorithm are studied: one using Newton's method and the other using Broyden's method. The possibility of extending known convergence results of such methods to the algorithm [6] was observed. Numerical experiments in a single linear programming problem were executed, which suggested theoretical results capable of describing the behavior of the algorithm in certain situations. Moreover, it was also possible to note that the sequence of iterates generated in the experiments always approached some stationary point of the merit function considered, which was in general found close to the solution of (3). By considering this fact, it was possible to elaborate a proposal for modifying the algorithm, which basically consists of applying the pure Broyden's method in the final iterate obtained from the algorithm developed by [6]. The results obtaining through this modification indicate an improvement in the performance of the algorithm with respect to global convergence, which possibly will also be effective in any linear programming problem where most of the stationary points of the merit function are close to the solution of system (3).

This work is organized as follows. An introduction about Newton's and Broyden's methods is given in Chapter 1. The IPM with quasi-Newton approach and its respective result about local convergence is showed in Chapter 2. In Chapter 3, the nonsmooth equations equivalent to the KKT conditions of problem (1) are formulated and it is presented some results about local and global convergence of Newton's and Broyden's methods applied in this system. Finally, the algorithm developed in [6] was applied in the nonsmooth equations and a proposal for its modification was elaborated with the objective of improve the convergence to the optimal point while taking advantage of its global convergence properties for non-optimal points.

Chapter 1

Newton's and Broyden's methods for nonlinear equations

In this chapter, we introduce Newton's and Broyden's methods for systems of nonlinear equations, as well as results about the local convergence of both. The main references of this part are Dennis and Schnabel [3] and Nocedal and Wright [17]. Throughout this chapter, we consider $\|\cdot\|$ as any vector norm, unless we say otherwise.

1.1 Newton's method

Solving a system of nonlinear equations can be described as the problem of, given $F : \mathbb{R}^n \to \mathbb{R}^n$ continuously differentiable in $\mathbb{R}^n$, finding $x^* \in \mathbb{R}^n$ such that

$$F(x^*) = 0. \tag{1.1}$$

Newton's method is one of the most used methods to solve this problem. It is characterized by use the Jacobian matrix at each iteration, which is defined in Definition 1.1.

Definition 1.1. *Given $f : \mathbb{R}^n \to \mathbb{R}^m$ and defining, for each $j \in \{1, ..., m\}$, the function $f_j : \mathbb{R}^n \to \mathbb{R}$ such that $f(x) = (f_1(x), ..., f_m(x))$ for all $x \in \mathbb{R}^n$, the Jacobian matrix of f at the point $a \in \mathbb{R}^n$ is defined as*

$$J(a) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(a) & \frac{\partial f_1}{\partial x_2}(a) & \dots & \frac{\partial f_1}{\partial x_n}(a) \\ \frac{\partial f_2}{\partial x_1}(a) & \frac{\partial f_2}{\partial x_2}(a) & \dots & \frac{\partial f_2}{\partial x_n}(a) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(a) & \frac{\partial f_m}{\partial x_2}(a) & \dots & \frac{\partial f_m}{\partial x_n}(a) \end{bmatrix},$$

where $\frac{\partial f_j}{\partial x_i}(a)$ is the partial derivative of function f_j with respect to x_i at the point $a \in \mathbb{R}^n$.

The operation of Newton's method is based on Lemmas 1.3 and 1.4. The second one, in particular, uses the concept of Lipschitz continuous function, presented in Definition 1.2.

Definition 1.2. *Suppose that $\|\cdot\|_1$ and $\|\cdot\|_2$ are norms in $\mathbb{R}^m$ and $\mathbb{R}^n$, respectively. A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be Lispchitz continuous at $x \in \mathbb{R}^n$ if there are $L > 0$ and an open set $D \subset \mathbb{R}^n$ containing x such that*

$$\|f(x) - f(y)\|_1 \leq L\|x - y\|_2 \tag{1.2}$$

for all $y \in D$. If (1.2) holds for each $x \in D$, then f is Lipschitz continuous in D.

We can observe that, since it is possible define matrix norms, Definition 1.2 can be extended to functions f which have its domain or counter-domain being a matrix space, such as for example $\mathbb{R}^{n\times n}$.

Lemma 1.3. *Let $F : \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable in some open convex set $D \subset \mathbb{R}^n$. Given $x, x + p \in D$*

$$F(x + p) - F(x) = \int_0^1 J(x + tp)p \; dt \tag{1.3}$$

Proof. The proof can be found in Lemma 4.1.9 of [3]. □

Lemma 1.4. *Suppose that $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable in some open convex set $D \subset \mathbb{R}^n$, $x \in D$ and the Jacobian $J : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is Lipschitz continuous at x in the neighborhood D with constant $L > 0$. Then, for any $x + p \in D$,*

$$\|F(x + p) - F(x) - J(x)p\| \leq \frac{L}{2}\|p\|^2.$$

Proof. The proof can be found in Lemma 4.1.12 of [3]. □

If the hypotheses are satisfied, lemmas 1.3 and 1.4 allow us to describe $J(x)p$ as an approximation for the right-hand side of equation (1.3) for p with sufficiently small value of its norm. Therefore, we can describe a linear model $M : \mathbb{R}^n \to \mathbb{R}^n$ as

$$M(x + p) = F(x) + J(x)p \tag{1.4}$$

that will approximate the function F at point x, that is, the closer p is to null vector, the better will be the approximation of function F by linear model M.

Letting x_k be the current iterate, Newton's method is characterized by approximating function F at x_k by a linear model $M_k : \mathbb{R}^n \to \mathbb{R}^n$ in the form of (1.4) and select

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

where p_k is such that $M_k(x_k + p_k) = 0$, which implies

$$p_k = -J(x_k)^{-1}F(x_k) \tag{1.5}$$

whenever $J(x_k)^{-1}$ exists, by equation (1.4).

We describe Newton's method for nonlinear equations by Algorithm 1.5.

Algorithm 1.5. *(Newton's method for nonlinear equations).*
Given x_0;
for $k = 0, 1, 2, ...$
Step 1: *Compute the vector p_k with the equation (1.5);*
Step 2: $x_{k+1} \leftarrow x_k + p_k$;
end(for)

As we show below, in Theorem 1.8, Newton's method have quadratic local convergence to the solution x^* of problem (1.1) under some hypotheses. The concept of local convergence is related to how close the first iterate x_0 should be to x^* to guarantee that the sequence of iterates $\{x_k\}$ generated by the algorithm will approach this solution. On the other hand, the quadratic convergence is a concept related to how fast the iterates of the algorithm will approach the solution x^*. In the following, we will formally define these terms.

Definition 1.6. *Suppose that $x^* \in \mathbb{R}^n$ is a solution of problem (1.1). An algorithm is said to be local convergent to x^* if there is $\delta > 0$ such that the occurrence of $\|x_0 - x^*\| < \delta$ implies that the sequence $\{x_k\}$ generated by the algorithm converges to x^*.*

Definition 1.7. *Suppose that $x^* \in \mathbb{R}^n$ is a solution of problem (1.1). An algorithm is said to be quadratically convergent to x^* when, being $\{x_k\}$ the sequence generated by the algorithm, there are $k_0 \in \mathbb{N}$ and $c > 0$ such that, if $k > k_0$ then*

$$\|x_{k+1} - x^*\| \leq c\|x_k - x^*\|^2.$$

Theorem 1.8. *Suppose that $J(x^*)$ is nonsingular, $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable and $J : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is Lipschitz continuous, both in an open convex set $D \subset \mathbb{R}^n$ that contains x^*. There exists $\epsilon > 0$ such that, if $\|x_0 - x^*\| \leq \epsilon$, then the sequence defined by*

$$x_{k+1} = x_k - J(x_k)^{-1}F(x_k),$$

which is generated by Algorithm 1.5, is well defined and converges quadratically to x^*.

Proof. This result is a particular case of Theorem 5.2.1 of [3]. □

1.2 Broyden's method

Newton's method is known to be good, but can be expensive. In general, this occurs because of the necessity of computing matrix $J(x_k)$ and solve (1.5) to obtain p_k at each

iteration k. By this reason, there are the quasi-Newton methods, which are characterized by computing an approximation B_k for $J(x_k)$ that makes less expensive the task of obtaining p_k at each iteration k. This class of methods are described by Algorithm 1.9.

Algorithm 1.9. *(Quasi-Newton method for nonlinear equations).*
Given x_0 and B_0;
for $k = 0, 1, 2, ...$
Step 1: *Compute vector* $p_k = -B_k^{-1}F(x_k)$;
Step 2: $x_{k+1} \leftarrow x_k + p_k$;
Step 3: *Obtain* B_{k+1};
end(for)

As we can see, at each iteration k, quasi-Newton and Newton's methods use the same strategy to find the next iterate. The only difference is that quasi-Newton methods build a linear model $M_k : \mathbb{R}^n \to \mathbb{R}^n$ defined by

$$M_k(x_k + p) = F(x_k) + B_k p, \tag{1.6}$$

which uses B_k as the coefficient matrix, rather than $J(x_k)$.

There is a particular class of quasi-Newton methods called secant methods. This group is characterized by satisfying the **secant equation** at each iteration k, which is defined by

$$B_{k+1}s_k = y_k, \tag{1.7}$$

where $s_k = x_{k+1} - x_k$ and $y_k = F(x_{k+1}) - F(x_k)$. In this case, using the definition of M_k,

$$M_k(x_k) = B_k 0 + F(x_k) = F(x_k).$$

Moreover, by the secant equation we have

$$x_{k-1} = x_k + B_k^{-1}(F(x_{k-1}) - F(x_k)),$$

which implies

$$M_k(x_{k-1}) = M_k(x_k + B_k^{-1}(F(x_{k-1}) - F(x_k))) = F(x_k) + B_k B_k^{-1}(F(x_{k-1}) - F(x_k)) = F(x_{k-1}).$$

Therefore, we conclude that M_k interpolates F at points x_k and x_{k-1}, that is, the graph of M_k is secant to the graph of F at theses points.

When we consider the problem of finding $x^* \in \mathbb{R}$ such that $F(x^*) = 0$ for scalar functions $F : \mathbb{R} \to \mathbb{R}$, an interesting comparison between Newton's and secant methods can be made in a geometric point of view. In Newton's method, given x_k, the intersection between x axis and the tangent to the graph of F at the point x_k will be the next iterate x_{k+1}, as we can see in Figure 1.1.

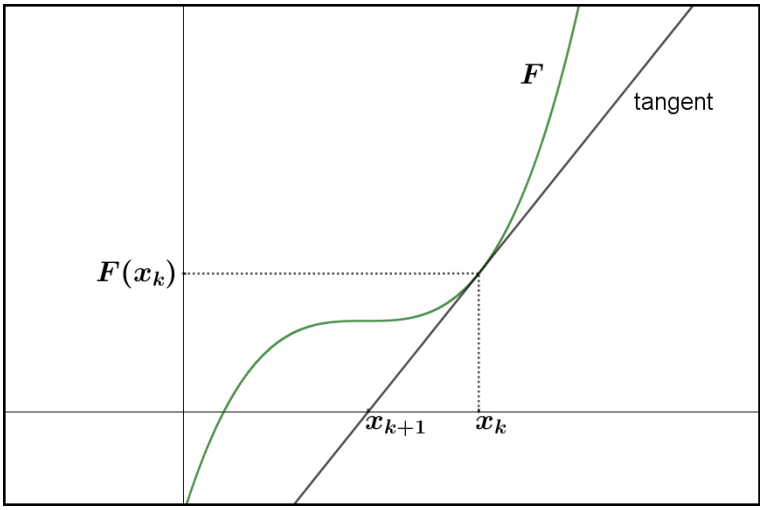

Figure 1.1: One step of Newton's method.

In secant methods, on the other hand, at each iteration k the next iterate x_{k+1} will be obtained from the intersection between x axis and the linear model that interpolates the graph of F at points $(x_{k-1},\ F(x_{k-1}))$ and $(x_k,\ F(x_k))$. This fact is illustrated in Figure 1.2.

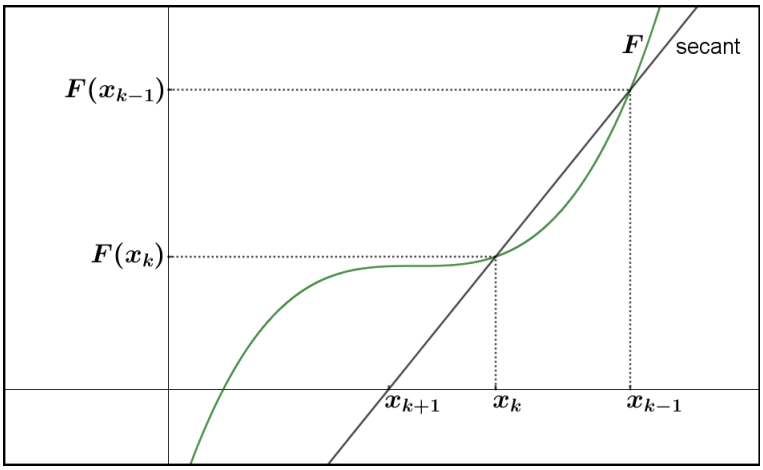

Figure 1.2: One step of secant method.

Going further, there is still a particular class of secant methods which, for all $k \in \mathbb{N}$, uses B_k to obtain B_{k+1} through the equation

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k) w_k^T}{w_k^T s_k}, \tag{1.8}$$

where $w_k \in \mathbb{R}^n$ and $w_k^T s_k \neq 0$. These methods are guaranteed secant methods because,

for any $k \in \mathbb{N}$,

$$B_{k+1}s_k = \left[B_k + \frac{(y_k - B_k s_k)w_k^T}{w_k^T s_k}\right] s_k = B_k s_k + (y_k - B_k s_k) = y_k,$$

and then all the matrices of the sequence $\{B_k\}$ generated by (1.8) satisfy the secant equation.

In this work, we use extensively a particular method that satisfies (1.8) with $w_k = s_k$. It is called Broyden's method and is described by Algorithm 1.10.

Algorithm 1.10. *(Broyden's method).*
Given x_0 and B_0;
for $k = 0, 1, 2, ...$
Step 1: *Compute the vector* $p_k = -B_k^{-1}F(x_k)$;
Step 2: $x_{k+1} \leftarrow x_k + p_k$;
Step 3: $s_k \leftarrow x_{k+1} - x_k$;
Step 4: $y_k \leftarrow F(x_{k+1}) - F(x_k)$;
Step 5: $B_{k+1} \leftarrow B_k + \frac{(y_k - B_k s_k)s_k^T}{s_k^T s_k}$;
end(for)

In the sequence, it is showed Definition 1.11 and 1.12. The first one concerns a characterization of how fast the sequence $\{x_k\}$ generated by an algorithm converges to the point x^*. On the other side, given some matrix $C \in \mathbb{R}^{n\times n}$ and norm $\|\cdot\|$ of $\mathbb{R}^n$, Definition 1.12 shows a measure related to C that represents the largest distortion that C can cause in a vector $x \neq 0$ with respect to the norm $\|\cdot\|$. These definitions are necessary for the Theorem 1.13, which guarantees the superlinear local convergence of Broyden's method.

Definition 1.11. *Suppose that $x^* \in \mathbb{R}^n$ is a solution of problem (1.1). An algorithm is said to be superlinearly convergent to x^* when, being $\{x_k\}$ the sequence generated by the algorithm, there is a sequence $\{r_k\}$ of positive scalars converging to 0 such that*

$$\|x_{k+1} - x^*\| \leq r_k \|x_k - x^*\|$$

for all $k \in \mathbb{N}$.

Definition 1.12. *A matrix norm $\|\cdot\|$ in $\mathbb{R}^{n\times n}$ is induced by a vector norm $\|\cdot\|$ of $\mathbb{R}^n$ when, given any matrix $C \in \mathbb{R}^{n\times n}$,*

$$\|C\| = \sup_{x\neq 0, x\in\mathbb{R}^n} \left\{\frac{\|Cx\|}{\|x\|}\right\}.$$

Theorem 1.13. *Let us denote $\|\cdot\|_2$ as the Euclidian vector norm and its induced matrix norm. Suppose that $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable and $J : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is Lipschitz continuous, both in a open convex set $D \subset \mathbb{R}^n$ that contains x^*, where $J(x^*)$ is nonsingular. There are $\epsilon, \delta > 0$ such that, if*

$$\|x_0 - x^*\|_2 \leq \epsilon \text{ and } \|B_0 - J(x^*)\|_2 \leq \delta$$

then the sequence $\{x_k\}$ generated by Broyden's method is well defined and converges superlinearly to x^*.

Proof. The proof follows by Theorem 8.2.2 of [3]. □

Chapter 2

Interior point methods

As described by Gondzio [7], since the 1940s, linear programming problems have attracted a lot of attention in the field of Optimization. One of the first methods that emerged aiming to solve this type of problem was the Simplex method (see Chapter 13 of [17]). A negative characteristic of this method is that it is not polynomial, that is, it does not occur that the number of iterations of the algorithm will always be related to a polynomial whose dimension is the domain of the objective function. This fact can be observed in Klee and Minty [12], which shows a problem of dimension n such that the Simplex method requires 2^n iterations to finds the solution. Thus, it is possible to find problems where Simplex would be a very expensive method to solve. However, in practice, the Simplex generally proved to be a very effective method, then was widely used for many years and still remains in use.

Considering this Simplex drawback, researchers searched for a long time to elaborate a polynomial method to solve linear programming problems. In this sense, the first two polynomial algorithms for solving these problems was developed by Dikin [4], in 1967, and Khachiyan [11], in the late 1970s. In particular, the second one sought to build a sequence of ellipsoids, so that the respective centers of each one constituted a sequence which approach a solution of the problem. Despite being polynomial, this method did not prove to be efficient in practice. Because of this, in the middle of 1980s, Karmarkar [10] devised a refinement of this algorithm, which added some geometric strategies in order to retain its polynomiality while improving its performance in practice.

After obtaining these promising results in the sense of obtaining an efficient and polynomial algorithm for solving linear programming problems, a large part of the field of Optimization started to seek for developing strategies with the purpose of elaborating algorithms that fit these characteristics. This gave rise to the class of methods currently known as interior point methods.

In this chapter, we present the interior point methods, which are popular methods to solve linear programming problems, since it is polynomial and generally quite efficient in practice. In Section 2.1, we show a standard primal-dual version of the method, which

uses as a basis the Newton's method, and prove some related properties. In Section 2.2, we present an alternative version of the method, which uses a quasi-Newton approach, and obtain a result about its local convergence. The main references of this chapter are [3, 8, 17, 22].

2.1 Primal-dual IPM for linear programming

In order to define the general problem used in this work, we first need Definition 2.1.

Definition 2.1. *The column rank of a matrix $A \in \mathbb{R}^{m\times n}$ is the maximum number of linearly independent columns in the matrix A. Analogously, the row rank is the maximum number of linearly independent rows in this matrix. The matrix A have full column rank if all its columns are linearly independent vectors. In the same way, A is said to be a matrix with full row rank if all its rows are linearly independent vectors.*

The linear programming problem considered in this work is

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0, \end{aligned} \tag{2.1}$$

where $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $m < n$ and $A \in \mathbb{R}^{m\times n}$ is a matrix with full row rank. The KKT conditions of this problem can be written as

$$\begin{aligned} A^T\lambda + z &= c \\ Ax &= b \\ XZe &= 0 \\ (x, z) &\geq 0 \end{aligned} \tag{2.2}$$

where $\lambda \in \mathbb{R}^m, z \in \mathbb{R}^n$ are the Lagrange multipliers, $e \in \mathbb{R}^n$ is a vector with all coordinates equal to 1 and $X, Z \in \mathbb{R}^{n\times n}$ are diagonal matrices with the main diagonal containing the vectors x and z, respectively.

From now on, let us denote the solution of (2.1) as x^*, the vector $(x, \lambda, z) \in \mathbb{R}^N$ as w, where $N = 2n + m$, the solution of (2.2) as $w^* = (x^*, \lambda^*, z^*)$, and the Euclidian vector norm and its induced matrix norm as $\|\cdot\|$, unless we say otherwise. In the sequence, we present some necessary definitions for the next results.

Definition 2.2. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be convex if, given any $x, y \in \mathbb{R}^n$ and $t \in [0, 1]$, it occurs*

$$f((1-t)x + ty) \leq (1-t)f(x) + tf(y).$$

If this inequality holds strictly, f is said a strict convex function.

Definition 2.3. *Given a function $f : \mathbb{R}^n \to \mathbb{R}^m$, the problem*

$$\begin{array}{ll} \min & f(x) \\ s.t. & x \in K \end{array} \tag{2.3}$$

is said to be a convex programming problem if f is a convex function and K is a convex set. When f is a strict convex function, the problem can also be called a strict convex programming problem.

Lemma 2.4 relates problem (2.1) with conditions (2.2).

Lemma 2.4. *A vector $x^* \in \mathbb{R}^n$ solves (2.1) if and only if there are $z^* \in \mathbb{R}^n$ and $\lambda^* \in \mathbb{R}^m$ such that x^*, λ^* and z^* satisfy (2.2).*

Proof. As we can see in [17, Section 12.6], the first order necessary optimality conditions hold in problem (2.1), since the constraints are linear.

Let us prove that problem (2.1) is a convex programming problem. In fact, we can see that, given any $y, y' \in \mathbb{R}^n$ and $t \in [0, 1]$,

$$c^T((1-t)y + ty') = (1-t)c^T y + tc^T y',$$

then the function $c^T x$ is convex. Moreover, given $y = (y_1, ..., y_n)$ and $y' = (y'_1, ..., y'_n)$ vectors contained in the feasible set of problem (2.1), we have $Ay = b$ and $Ay' = b$, while $y_j, y'_j \geq 0$ for all $j \in \{1, ..., n\}$. Given any vector z contained in the segment between y and y', there is $t \in [0, 1]$ such that $z = ty + (1-t)y'$. Therefore, for any $j \in \{1, ..., n\}$ occurs $z_j = ty_j + (1-t)y'_j \geq 0$, while

$$Az = A(ty + (1-t)y') = tAy + (1-t)Ay' = tb + (1-t)b = b,$$

which implies that z is in the feasible set of the (2.1) also. From this, we have that all the segment between y and y' is contained in this feasible set, and then it is convex, which guarantees that (2.1) is a convex programming problem.

In [17, Section 13.1] we can observe that the first order sufficient optimality conditions hold if problem (2.1) is a convex programming problem, which finishes the proof. □

Given a constant $\mu > 0$, consider the problem

$$\begin{array}{ll} \min & c^T x - \mu \displaystyle\sum_{j=1}^{n} \ln\ x_j \\ \text{s.t.} & Ax = b \end{array} . \tag{2.4}$$

By considering a new variable $z = \mu X^{-1}e$, the KKT conditions of (2.4) can be described as

$$\begin{aligned} A^T\lambda + z &= c \\ Ax &= b \\ XZe &= \mu e \\ (x, z) &\geq 0. \end{aligned} \tag{2.5}$$

In order to associate problem (2.4) with conditions (2.5), we have Lemma 2.5.

Lemma 2.5. *A vector $x^* \in \mathbb{R}^n$ solves problem (2.4) if and only if there are $\lambda^* \in \mathbb{R}^m$ and $z^* \in \mathbb{R}^n$ such that x^*, λ^* and z^* satisfy (2.5).*

Proof. The first order necessary optimality conditions hold in problem (2.4) because the constraints are linear, as showed in [17, Section 12.6].

On the other hand, for the first order sufficient optimality conditions, by defining the Lagrangian function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ of problem (2.4) as

$$\mathcal{L}(x, \lambda) = c^T x - \mu \sum_{j=1}^{n} \ln x_j - \sum_{i=1}^{m} [\lambda_i (A_{i\cdot}^T x - b_i)],$$

where $A_{i\cdot}$ is the i-th row vector of A, we obtain its Hessian with respect the x variables at vector $(x^*, \lambda^*) \in \mathbb{R}^{n+m}$, $\nabla^2_{xx}\mathcal{L}(x^*, \lambda^*)$, as

$$\nabla^2_{xx}\mathcal{L}(x^*, \lambda^*) = \mu \begin{bmatrix} \frac{1}{x_1^{*2}} & 0 & \cdots & 0 \\ 0 & \frac{1}{x_2^{*2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{x_n^{*2}} \end{bmatrix}.$$

As $x_i^* > 0$ for all $i \in \{1, ..., n\}$, $\nabla^2_{xx}\mathcal{L}(x^*, \lambda^*)$ is positive definite, and then the statement follows by Theorem 12.6 of [17]. □

Let us define the primal-dual strictly feasible set $\mathcal{F}^0$ as

$$\mathcal{F}^0 = \{w \in \mathbb{R}^N; Ax = b, A^T\lambda + z = c, (x, z) > 0\}. \tag{2.6}$$

The system (2.5) is frequently used by IPMs. Therefore, Lemma 2.6, which is a specification of Theorem 17.2 of [21] for our case, implies that a necessary condition to the application of IPMs is $\mathcal{F}^0 \neq \varnothing$, since this guarantees the existence of solution for (2.5).

Lemma 2.6. *Given any $\mu > 0$, system (2.5) has a unique solution if and only if $\mathcal{F}^0 \neq \varnothing$.*

Proof. If system (2.5) have a unique solution $(x^*, \lambda^*, z^*) \in \mathbb{R}^N$, then $Ax^* = b, A^T\lambda^* + z^* = c, (x^*, z^*) \geq 0$ and $x_i^* z_i^* = \mu > 0$ for all $i \in \{1, ..., n\}$. The last two conditions imply $(x^*, z^*) > 0$, therefore $(x^*, \lambda^*, z^*) \in \mathcal{F}^0$ and then we have $\mathcal{F}^0 \neq \varnothing$.

Suppose now that $\mathcal{F}^0 \neq \varnothing$, then there is $(\bar{x}, \bar{\lambda}, \bar{z}) \in \mathcal{F}^0$. Let us consider any $\mu > 0$ and the function $f_\mu : W \to \mathbb{R}$, where $W = \{x = (x_1, ..., x_n) \in \mathbb{R}^n; x_j > 0 \ \forall \ j \in \{1, ..., n\}\}$ defined by

$$f_\mu(x) = c^T x - \mu \sum_{j=1}^{n} \ln \ x_j.$$

Observe that this function is a sum between a strictly convex function and a linear function, which we saw is convex in the proof of Lemma 2.4. This fact implies that f_μ is also strictly convex, and therefore the set $\{x \in \mathbb{R}^n; f_\mu(x) \leq d\}$ is bounded for any $d \in \mathbb{R}$. By this fact, the inverse image of $(-\infty, f_\mu(\bar{x})]$ under f_μ, which is

$$f_\mu^{-1}((-\infty, f_\mu(\bar{x})]) = \{x \in \mathbb{R}^n; f_\mu(x) \leq f_\mu(\bar{x})\},$$

is a bounded set. Let us prove that it is also closed.

The function f_μ is a sum between continuous functions, thus is continuous. Therefore, since the set $(-\infty, f_\mu(\bar{x})]$ is closed in $\mathbb{R}$, it occurs that $f_\mu^{-1}((-\infty, f_\mu(\bar{x})])$ is closed in W, which is the domain of this function. If there is $x' = (x'_1, ..., x'_n)$ in the closure of $f_\mu^{-1}((-\infty, f_\mu(\bar{x})])$ such that $x' \notin W$ then there would be $i \in \{1, ..., n\}$ where $x'_i = 0$. In this case, there would be a sequence $\{y_j\}$ of vectors $y_j = (y_{j1}, ..., y_{jn}) \in f_\mu^{-1}((-\infty, f_\mu(\bar{x})])$ convergent to x', which implies that $\{y_{ji}\} \to 0$ as $j \to \infty$ and then $\{\ln y_{ji}\} \to -\infty$ while the sequence $\{\ln y_{jk}\}$ is bounded above for all $k \in \{1, ..., n\}$ such that $k \neq i$, since $f_\mu^{-1}((-\infty, f_\mu(\bar{x})])$ is bounded. Therefore, $\{f_\mu(y_j)\} \to \infty$ and then the sequence $\{y_j\}$ is not contained in $f_\mu^{-1}((-\infty, f_\mu(\bar{x})])$, which is a contradiction. This fact implies that the closure of $f_\mu^{-1}((-\infty, f_\mu(\bar{x})])$ is contained in W, and therefore this set is closed in $\mathbb{R}^n$.

Consider the set

$$\mathcal{A} = \{x \in \mathbb{R}^n; Ax = b, x \geq 0, f_\mu(x) \leq f_\mu(\bar{x})\},$$

then $\mathcal{A} \neq \varnothing$, since $\bar{x} \in \mathcal{A}$. Let us prove that $\mathcal{A}$ is a compact set.

We can write

$$\mathcal{A} = \{x \in \mathbb{R}^n; Ax = b\} \ \cap \ \{x \in \mathbb{R}^n; x \geq 0\} \ \cap \ \{x \in \mathbb{R}^n; f_\mu(x) \leq f_\mu(\bar{x})\}. \tag{2.7}$$

This set is closed because it is a finite intersection between closed sets: a linear space, a half space and a closed set by the previous paragraph. It is bounded, since $\mathcal{A} \subset \{x \in \mathbb{R}^n; f_\mu(x) \leq f_\mu(\bar{x})\}$ also by the previous paragraph. Therefore, $\mathcal{A}$ is compact.

Let us consider $f_\mu|_{\mathcal{A}} : \mathcal{A} \to \mathbb{R}$ as the restriction of the function f_μ to the compact set $\mathcal{A}$. As $f_\mu|_{\mathcal{A}}$ is a continuous function with compact domain, there is $x^* \in \mathcal{A}$ such that $f_\mu(x^*) \leq f_\mu(x)$ for all $x \in \mathcal{A}$. We can see that, as $x^* \in \mathcal{A}$, it occurs $Ax^* = b$, which implies that x^* is feasible for problem (2.4). Given any $x \in \mathbb{R}^n$ feasible for (2.4), we have two possibilities: $x \in \mathcal{A}$ or not. If $x \in \mathcal{A}$ then $f_\mu(x) \geq f_\mu(x^*)$ by the fact that x^* is a

minimizer of $f_\mu|_{\mathcal{A}}$. If not, then either the function f_μ is not well defined at point x, which means that $x_i \leq 0$ for some $i \in \{1, ..., n\}$, or it is well defined and $f_\mu(x) > f_\mu(\bar{x}) \geq f_\mu(x^*)$. Therefore, x^* is a solution for problem (2.4) and then, by Lemma 2.5, there are $\lambda^* \in \mathbb{R}^m$ and $z^* \in \mathbb{R}^n$ such that x^*, λ^* and z^* satisfy system (2.5), which finishes the proof of the solution's existence.

For the proof of uniqueness, assume that x^*, λ^* and z^* satisfy system (2.5). By Lemma 2.5, x^* is a solution of problem (2.4), which is a strict convex programming problem. Therefore, from [15, Theorem 3.1.17], this problem have at most one solution, which implies that x^* is unique and so is z^* as well, since $z^* = \mu X^{*-1} e$ is satisfied. Finally, by system (2.5), we have $A^T\lambda^* + z^* = c$. As A have full row rank, then A^T is a injective function, which guarantees the uniqueness of λ^* and finishes the proof. □

From Lemma 2.6, being $\mathbb{R}^+$ the set of positive scalars, we can define a function $h : \mathbb{R}^+ \to \mathbb{R}^N$ which relates each $\mu > 0$ with the vector $(x_\mu, \lambda_\mu, z_\mu)$ that satisfies system (2.5) for this respective scalar. The image of this function is called **central path**. In Lemma 2.8 will be showed an important property about this function. However, for its proof, we need of Lemma 2.7.

Lemma 2.7. *Suppose that*

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z & 0 & X \end{bmatrix}$$

is a matrix where $A \in \mathbb{R}^{m\times n}$ have full row rank, $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix and $Z, X \in \mathbb{R}^{n\times n}$ are diagonal matrices where the elements of its main diagonals, denoted by z_i and x_i respectively, are all strictly positive. Then, this matrix is nonsingular.

Proof. To prove the nonsingularity of this matrix, it is sufficient we guarantee that, given any vector $(u, v, w) \in \mathbb{R}^N$ such that $u, w \in \mathbb{R}^n$ and $v \in \mathbb{R}^m$, if

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z & 0 & X \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.8}$$

then u, v and w are null vectors.

In fact, by (2.8) we have

$$\begin{aligned} A^T v + w &= 0 \\ Au &= 0 \\ Zu + Xw &= 0. \end{aligned} \tag{2.9}$$

Through the last equation of system (2.9),

$$z_i u_i + x_i w_i = 0 \tag{2.10}$$

for all $i \in \{1, ..., n\}$. Observe that, by the first equation of (2.9),

$$u^T(A^T v + w) = u^T 0 = 0,$$

which implies

$$(Au)^T v + u^T w = 0$$

and then

$$u^T w = 0, \tag{2.11}$$

through the second equation of (2.9).

We can see that (2.11) is equivalent to

$$\sum_{i=1}^{n} u_i w_i = 0, \tag{2.12}$$

which through (2.10) implies that

$$-\sum_{i=1}^{n} \left(\frac{z_i}{x_i}\right) u_i^2 = 0. \tag{2.13}$$

By (2.13), since $x_i, z_i > 0$ for all $i \in \{1, ..., n\}$, it must occur $u = 0$. Analogously, by applying (2.10) in (2.12), we have

$$-\sum_{i=1}^{n} \left(\frac{x_i}{z_i}\right) w_i^2 = 0,$$

which implies that $w = 0$.

Finally, $A^T v + w = 0$ implies $A^T v = 0$. Since A have full row rank, then A^T have full column rank, which implies $v = 0$ and finishes the proof. □

Lemma 2.8. *The function $h : \mathbb{R}^+ \to \mathbb{R}^N$ defined such that, for each $\mu > 0$, $h(\mu) = (x_\mu, \lambda_\mu, z_\mu)$ which satisfy system (2.5), is continuous.*

Proof. Let us define $G : \Omega \subset \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N$ where $\Omega = \{(\mu, (x, \lambda, z)) \in \mathbb{R} \times \mathbb{R}^N; \mu, x, z > 0\}$ and

$$G(\mu, (x, \lambda, z)) = \begin{bmatrix} A^T\lambda + z - c \\ Ax - b \\ XZe - \mu e \end{bmatrix}.$$

Observe that G is continuously differentiable in Ω and that Ω is an open set, since it is a finite product between open sets. Let us consider any $\mu_0 > 0$. By Lemma 2.6, there

is $(x_{\mu_0}, \lambda_{\mu_0}, z_{\mu_0}) \in \mathbb{R}^N$ which satisfies system (2.5) for this scalar, which implies that $(\mu_0, (x_{\mu_0}, \lambda_{\mu_0}, z_{\mu_0})) \in \Omega$ and $G(\mu_0, (x_{\mu_0}, \lambda_{\mu_0}, z_{\mu_0})) = 0$. The Jacobian of G at this point is

$$JG(\mu_0, (x_{\mu_0}, \lambda_{\mu_0}, z_{\mu_0})) = \begin{bmatrix} 0 & 0 & A^T & I_n \\ 0 & A & 0 & 0 \\ -e & Z_{\mu_0} & 0 & X_{\mu_0} \end{bmatrix},$$

where $I_n \in \mathbb{R}^{n \times n}$ is the identity matrix. As A have full row rank, by Lemma 2.7 the matrix

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z_{\mu_0} & 0 & X_{\mu_0} \end{bmatrix}$$

is nonsingular. Therefore, by the Implicit Function Theorem, which can be found at [14, Theorem 6, Chapter 6], there are an open set $V \subset \mathbb{R}$ containing μ_0 and an open set $U \subset \Omega$ containing $(\mu_0, (x_{\mu_0}, \lambda_{\mu_0}, z_{\mu_0}))$ such that there is a differentiable function $g : V \to \mathbb{R}^N$ where g(μ_0)=$(x_{\mu_0}, \lambda_{\mu_0}, z_{\mu_0})$ and for all $\mu \in V$ occurs $(\mu, g(\mu)) \in U$ and $G(\mu, g(\mu)) = 0$. Therefore, given any $\mu \in V$, $g(\mu)$ satisfies system (2.5) for this scalar. However, by Lemma 2.6, this solution is unique, which implies $g(\mu) = h(\mu)$ for all $\mu \in V$, and then $g = h|_V$.

As $\mu_0 \in V$, $h|_V$ is continuous and V is an open set, then h is continuous at μ_0. Therefore, by the arbitrary choice of μ_0, we obtain that h is continuous. □

By the continuity of the central path, as μ tends to zero, the solution of problem (2.4) becomes each time more similar to the solution of problem (2.1). The standard version of the IPMs works through this strategy. At each iteration, given $\mu > 0$, one iteration of Newton's method is executed to solve system (2.5) in order to approximate the iterate to the solution of subproblem (2.4) using $\sigma\mu$ instead of μ, where $\sigma \in (0, 1)$. Therefore, throughout the iterations, the IPMs brings μ to 0 and leads the iterates to a solution of (2.1).

As showed in [17, Chapter 14], a standard primal-dual IPM for solving (2.1) is described by Algorithm 2.9.

Algorithm 2.9. *(Standard primal-dual IPM for solving (2.1)).*

Given (x_0, λ_0, z_0) *such that* $(x_0, z_0) > 0$*;*

for $k = 0, 1, 2, ...$

Step 1: *Compute*

$$\mu_k = \frac{x_k^T z_k}{n} \tag{2.14}$$

and choose $\sigma_k \in (0, 1)$;

Step 2: *Compute the Newton's direction by solving the linear system*

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z_k & 0 & X_k \end{bmatrix} \begin{bmatrix} \Delta x_k \\ \Delta \lambda_k \\ \Delta z_k \end{bmatrix} = \begin{bmatrix} c - A^T \lambda_k - z_k \\ b - Ax_k \\ \sigma_k \mu_k e - X_k Z_k e \end{bmatrix}; \tag{2.15}$$

Step 3: $w_{k+1} \leftarrow w_k + \alpha_k(\Delta x_k, \Delta \lambda_k, \Delta z_k)$, *where* $\alpha_k \in (0, 1]$ *is such that* $(x_{k+1}, z_{k+1}) > 0$*;*

end(for)

The information that Lemma 2.8 sends to us provides an important geometric notion about the way that this primal-dual IPM works, by considering the strategy that we commented above. At each iteration k, we have a value $\mu_k > 0$, which is computed through (2.14), and then it is a good measure of how close the current iterate w_k is to satisfying the condition $X_k Z_k e = 0$. Through the iteration of Newton's method that is realized, the new iterate approximates a solution of subproblem (2.4) using $\sigma_k \mu_k$ as the constant μ of its general form. By the fact that $\sigma_k \in (0, 1)$, the solution of this subproblem is a point of the central path closer to the solution w^* than the point of central path correspondent to μ_k. Thus, w_{k+1} will be closer to the solution w^* than w_k and then, through (2.14), μ_{k+1} will be closer to 0 than μ_k. Therefore, throughout the iterations, the method generates iterates that follows the central path and, as the sequence $\{\mu_k\}$ tends to zero, go towards the solution of problem (2.1).

In the literature about IPMs, there are two variables which can be called as "centrality parameter": the σ and the μ. Some authors, like for example Gondzio [7], call σ as being the centrality parameter because, by the way that the standard primal-dual IPM works, we can control the proximity between the next iterate w_{k+1} and the central path through this value. If σ is close to 0, then the Newton step of the IPM will place more importance on generating a new iterate w_{k+1} with a relevant proximity to the solution w^* than generating one close to the central path, which is a risky step in relation to the necessity of $(x_{k+1}, z_{k+1}) > 0$. Analogously, if σ is close to 1, then the objective will be to generate w_{k+1} close to the central path, even if no relevant approximation to the solution is obtained in this iteration, which promotes security in relation to the condition $(x_{k+1}, z_{k+1}) > 0$. On the other hand, authors like Wright [22] consider μ as the centrality parameter by the fact that this variable parametrizes the central path.

2.2 IPM with quasi-Newton approach

Let us define the function $F : \mathbb{R}^N \to \mathbb{R}^N$ by

$$F(w) = \begin{bmatrix} A^T\lambda + z - c \\ Ax - b \\ XZe \end{bmatrix}. \tag{2.16}$$

Finding a solution for (2.2), and therefore for (2.1), by Lemma 2.4 is equivalent to find $w^* \in \mathbb{R}^N$ such that $F(w^*) = 0$ and $(x^*, z^*) \geq 0$. Therefore, we can see the standard version of primal-dual IPM as a method with this objective. In this case, at each iteration k, solving equation (2.15) is equivalent to computing the Newton's direction for the system

$$F(w_k) = \begin{bmatrix} 0 \\ 0 \\ \sigma_k\mu_k e \end{bmatrix}$$

with $\mu_k > 0$ and $\sigma_k \in (0, 1)$. The coefficient matrix of (2.15) can be denoted as $J(w_k)$ for all $k \in \mathbb{N}$, since it coincides with the Jacobian of the function F at w_k.

One of the most expensive steps of the standard primal-dual IPM is solving system (2.15). Therefore, in [8] it was elaborated a method that, at each iteration k, uses a Broyden approach B_k of the matrix $J(w_k)$, with the goal that the computation of the direction through equation

$$B_k \begin{bmatrix} \Delta x_k \\ \Delta\lambda_k \\ \Delta z_k \end{bmatrix} = \begin{bmatrix} c - A^T\lambda_k - z_k \\ b - Ax_k \\ \sigma_k\mu_k e - X_k Z_k e \end{bmatrix} \tag{2.17}$$

is less expensive than with (2.15).

Our goal in this section is to understand some local properties of Newton and quasi-Newton IPMs under the framework of algorithms for systems of nonlinear equations [3, 15]. To do this, we first need to define some concepts.

Definition 2.10. *An algorithm is said to be linearly convergent to w^* when, being $\{w_k\}$ the sequence generated by the algorithm, there are $k_0 \in \mathbb{N}$ and $r \in (0, 1)$ such that, if $k > k_0$ then*

$$\|w_{k+1} - w^*\| \leq r\|w_k - w^*\|.$$

An IPM is said to be **feasible** if all the iterates w_k are in the set $\mathcal{F}^0$ defined in (2.6). In Lemma 2.11, Lemma 2.12 and Theorem 2.15 we built a result about the linear local convergence of the feasible IPM with quasi-Newton approach, which solves at each iteration k the equation (2.17) instead of (2.15), where B_k is the Broyden approach of

matrix $J(w_k)$. In this analysis, the function F considered is defined by (2.16) and J is its respective Jacobian, which coincides with the coefficient matrix of (2.15) when applied at w_k.

Lemma 2.11. *If $x = (x_1, ..., x_n)$ and $z = (z_1, ..., z_n)$ are vectors in $\mathbb{R}^n$ satisfying $x, z > 0$ and $\mu = \frac{x^T z}{n}$, then $\|\mu e\| \leq \|XZe\|$.*

Proof. Firstly, let us prove that

$$n\sum_{i=1}^{n} x_i^2 z_i^2 \geq \left(\sum_{i=1}^{n} x_i z_i\right)^2. \tag{2.18}$$

Observe that

$$0 \leq \left[\sum_{i=2}^{n}(x_1 z_1 - x_i z_i)^2\right] + \left[\sum_{i=3}^{n}(x_2 z_2 - x_i z_i)^2\right] + ... + [(x_{n-1}z_{n-1} - x_n z_n)^2] =$$
$$\left[\sum_{i=2}^{n} x_1^2 z_1^2 + x_i^2 z_i^2 - 2x_1 z_1 x_i z_i\right] + \left[\sum_{i=3}^{n} x_2^2 z_2^2 + x_i^2 z_i^2 - 2x_2 z_2 x_i z_i\right] + ...+$$
$$\left[x_{n-1}^2 z_{n-1}^2 + x_n^2 z_n^2 - 2x_{n-1} z_{n-1} x_n z_n\right] =$$
$$(n-1)\left(\sum_{i=1}^{n} x_i^2 z_i^2\right) - 2\left[\left(\sum_{i=2}^{n} x_1 z_1 x_i z_i\right) + \left(\sum_{i=3}^{n} x_2 z_2 x_i z_i\right) + ... + (x_{n-1}z_{n-1}x_n z_n)\right].$$

Therefore,

$$(n-1)\left(\sum_{i=1}^{n} x_i^2 z_i^2\right) \geq 2\left[\left(\sum_{i=2}^{n} x_1 z_1 x_i z_i\right) + \left(\sum_{i=3}^{n} x_2 z_2 x_i z_i\right) + ... + (x_{n-1}z_{n-1}x_n z_n)\right],$$

which implies that

$$n\sum_{i=1}^{n} x_i^2 z_i^2 \geq$$
$$2\left[\left(\sum_{i=2}^{n} x_1 z_1 x_i z_i\right) + \left(\sum_{i=3}^{n} x_2 z_2 x_i z_i\right) + ... + (x_{n-1}z_{n-1}x_n z_n)\right] + \left(\sum_{i=1}^{n} x_i^2 z_i^2\right) =$$
$$\left(\sum_{i=1}^{n} x_i z_i\right)^2,$$

as we wanted.

By (2.18), it occurs that

$$\sum_{i=1}^{n} x_i^2 z_i^2 \geq \frac{\left(\sum_{i=1}^{n} x_i z_i\right)^2}{n} \geq 0,$$

therefore

$$\sqrt{\sum_{i=1}^{n} x_i^2 z_i^2} \geq \frac{(\sum_{i=1}^{n} x_i z_i)}{\sqrt{n}} = \frac{(\sum_{i=1}^{n} x_i z_i)}{n}\sqrt{n}. \tag{2.19}$$

Since

$$\|XZe\| = \sqrt{\sum_{i=1}^{n} x_i^2 z_i^2}$$

and

$$\|\mu e\| = \mu\sqrt{n} = \frac{(\sum_{i=1}^{n} x_i z_i)}{n}\sqrt{n},$$

by (2.19) we have that $\|XZe\| \geq \|\mu e\|$, which finishes the proof. □

Lemma 2.12. *Consider a matrix $B \in \mathbb{R}^{N\times N}$. Given $r \in (0,1)$, there exist ϵ_1, $\delta_1 > 0$ and $\bar{\alpha}$, $\bar{\sigma} \in (0,1)$ such that, if $\|w - w^*\| \leq \epsilon_1$, $\|B - J(w^*)\| \leq \delta_1$, $J(w^*)$ is nonsingular, $\mu = \frac{x^T z}{n}$ and $\alpha \in (\bar{\alpha}, 1)$, then*

$$\bar{w} = w - \alpha B^{-1}\left(F(w) - \begin{bmatrix} 0 \\ 0 \\ \bar{\sigma}\mu e \end{bmatrix}\right)$$

is well defined and

$$\|\bar{w} - w^*\| \leq r\|w - w^*\|.$$

Proof. Consider $r \in (0,1)$ and $\beta : \mathbb{R}^N \to \mathbb{R}$ defined by $\beta(w) = \|F(w) - J(w^*)(w - w^*)\|$. Observe that we have

$$\lim_{w\to w^*} \frac{\beta(w)}{\|w - w^*\|} = 0$$

by the fact that F is differentiable. Consider also δ_1, $\bar{\sigma}$ and ϵ_1 sufficiently close to 0 where $\delta_1 \leq \frac{1}{2\|J(w^*)^{-1}\|}$ and $\bar{\alpha}$ sufficiently close to 1 such that

$$\delta_1 + \bar{\alpha}(1+\bar{\sigma}) \sup_{\|w-w^*\|\leq\epsilon_1} \frac{\beta(w)}{\|w - w^*\|} + (1 + (\bar{\sigma} - 1)\bar{\alpha})\|J(w^*)\| \leq \frac{r}{2\|J(w^*)^{-1}\|} \tag{2.20}$$

and

$$\sup_{\|w-w^*\|\leq\epsilon_1} \frac{\beta(w)}{\|w - w^*\|} \leq -\frac{\|J(w^*)\|(\bar{\sigma} - 1)}{1 + \bar{\sigma}} \tag{2.21}$$

hold.

If all the hypotheses are satisfied, then $\|B - J(w^*)\| \leq \frac{1}{2\|J(w^*)^{-1}\|}$, which implies

$$\|J(w^*)^{-1}(B - J(w^*))\| \leq \|J(w^*)^{-1}\|\|(B - J(w^*))\| \leq \|J(w^*)^{-1}\|\frac{1}{2\|J(w^*)^{-1}\|} \leq \frac{1}{2} < 1.$$

From this, by Theorem 3.1.4 of [3], B is nonsingular, therefore $\bar{w}$ is well defined and

$$\|B^{-1}\| \leq \frac{\|J(w^*)^{-1}\|}{1 - \|J(w^*)^{-1}(B - J(w^*))\|} \leq 2\|J(w^*)^{-1}\|. \tag{2.22}$$

Let us define $\Sigma = \begin{bmatrix} 0 \\ 0 \\ \bar{\sigma}\mu e \end{bmatrix}$, then

$$\begin{gathered} \|\bar{w} - w^*\| = \|w - \alpha B^{-1}(F(w) - \Sigma) - w^*\| = \\ \|w - \alpha B^{-1}F(w) + \alpha B^{-1}\Sigma - w^* - B^{-1}J(w^*)(w - w^*) + B^{-1}J(w^*)(w - w^*)\| \leq \\ \|w - w^* - B^{-1}J(w^*)(w - w^*)\| + \|B^{-1}(-\alpha F(w) + J(w^*)(w - w^*) + \alpha\Sigma)\|. \end{gathered} \tag{2.23}$$

For the first term on the right-hand side,

$$\begin{gathered} \|w - w^* - B^{-1}J(w^*)(w - w^*)\| = \|B^{-1}(B(w - w^*) - J(w^*)(w - w^*))\| \leq \\ \|B^{-1}\|\|B - J(w^*)\|\|w - w^*\| \leq 2\|J(w^*)^{-1}\|\delta_1\|w - w^*\|. \end{gathered} \tag{2.24}$$

By Lemma 2.11,

$$\|\Sigma\| = \|\bar{\sigma}\mu e\| = \bar{\sigma}\|\mu e\| \leq \bar{\sigma}\|XZe\| = \bar{\sigma}\|XZe - [J(w^*)]_3(w - w^*) + [J(w^*)]_3(w - w^*)\|$$

where $[J(w^*)]_3$ is the third row block of $J(w^*)$. From this,

$$\begin{gathered} \|\Sigma\| \leq \\ \bar{\sigma}(\|XZe - [J(w^*)]_3(w - w^*)\| + \|[J(w^*)]_3(w - w^*)\|) \leq \\ \bar{\sigma}(\beta(w) + \|J(w^*)\|\|w - w^*\|). \end{gathered} \tag{2.25}$$

By considering the second term of (2.23), through the consistency of the norm and some manipulation of the terms we have

$$\begin{gathered} \|B^{-1}(-\alpha F(w) + J(w^*)(w - w^*) + \alpha\Sigma)\| \leq \\ \|B^{-1}\|\| - \alpha F(w) + J(w^*)(w - w^*) + \alpha\Sigma - \alpha J(w^*)(w - w^*) + \alpha J(w^*)(w - w^*)\| = \\ \|B^{-1}\|\| - \alpha(F(w) - J(w^*)(w - w^*)) + (1 - \alpha)J(w^*)(w - w^*) + \alpha\Sigma\|. \end{gathered}$$

Through (2.22) and the triangular inequality it occurs

$$\begin{gathered} \|B^{-1}\|\| - \alpha(F(w) - J(w^*)(w - w^*)) + (1 - \alpha)J(w^*)(w - w^*) + \alpha\Sigma\| \leq \\ 2\|J(w^*)^{-1}\|[\alpha\|F(w) - J(w^*)(w - w^*)\| + (1 - \alpha)\|J(w^*)(w - w^*)\| + \alpha\|\Sigma\|]. \end{gathered}$$

By the definition of β, the consistency of the norm and (2.25) we have

$$\begin{gathered} 2\|J(w^*)^{-1}\|[\alpha\|F(w) - J(w^*)(w - w^*)\| + (1 - \alpha)\|J(w^*)(w - w^*)\| + \alpha\|\Sigma\|] \leq \\ 2\|J(w^*)^{-1}\|[\alpha\beta(w) + (1 - \alpha)\|J(w^*)\|\|w - w^*\| + \alpha\bar{\sigma}(\beta(w) + \|J(w^*)\|\|w - w^*\|)]. \end{gathered}$$

Finally, with some manipulation, we obtain

$$\begin{gathered}2\|J(w^*)^{-1}\|[\alpha\beta(w)+(1-\alpha)\|J(w^*)\|\|w-w^*\|+\alpha\bar{\sigma}(\beta(w)+\|J(w^*)\|\|w-w^*\|)]= \\ 2\|J(w^*)^{-1}\|[\alpha(1+\bar{\sigma})\beta(w)+(1+(\bar{\sigma}-1)\alpha)\|J(w^*)\|\|w-w^*\|]= \\ 2\|J(w^*)^{-1}\|[\alpha(1+\bar{\sigma})\tfrac{\beta(w)}{\|w-w^*\|}+(1+(\bar{\sigma}-1)\alpha)\|J(w^*)\|]\|w-w^*\|.\end{gathered}$$

From this, we have that

$$\begin{gathered}\|B^{-1}(-\alpha F(w)+J(w^*)(w-w^*)+\alpha\Sigma)\| \leq \\ 2\|J(w^*)^{-1}\|[\alpha(1+\bar{\sigma})\tfrac{\beta(w)}{\|w-w^*\|}+(1+(\bar{\sigma}-1)\alpha)\|J(w^*)\|]\|w-w^*\|.\end{gathered} \tag{2.26}$$

Thus, by (2.23), (2.24) and (2.26),

$$\begin{gathered}\|\bar{w}-w^*\| \leq \\ 2\|J(w^*)^{-1}\|\left(\delta_1+\alpha(1+\bar{\sigma})\tfrac{\beta(w)}{\|w-w^*\|}+(1+(\bar{\sigma}-1)\alpha)\|J(w^*)\|\right)\|w-w^*\|.\end{gathered} \tag{2.27}$$

From (2.21), if $\alpha \in (\bar{\alpha}, 1)$ then

$$\begin{gathered}\alpha[(1+\bar{\sigma})\sup_{\|w-w^*\|\leq\epsilon_1}\frac{\beta(w)}{\|w-w^*\|}+\|J(w^*)\|(\bar{\sigma}-1)] \leq \\ \bar{\alpha}[(1+\bar{\sigma})\sup_{\|w-w^*\|\leq\epsilon_1}\frac{\beta(w)}{\|w-w^*\|}+\|J(w^*)\|(\bar{\sigma}-1)].\end{gathered}$$

With some manipulation, this implies

$$\begin{gathered}\alpha(1+\bar{\sigma})\sup_{\|w-w^*\|\leq\epsilon_1}\frac{\beta(w)}{\|w-w^*\|}+(1+(\bar{\sigma}-1)\alpha)\|J(w^*)\| \leq \\ \bar{\alpha}(1+\bar{\sigma})\sup_{\|w-w^*\|\leq\epsilon_1}\frac{\beta(w)}{\|w-w^*\|}+(1+(\bar{\sigma}-1)\bar{\alpha})\|J(w^*)\|.\end{gathered} \tag{2.28}$$

Thus, using (2.20), (2.27) and (2.28),

$$\begin{gathered}\|\bar{w}-w^*\| \leq \\ 2\|J(w^*)^{-1}\|\left(\delta_1+\alpha(1+\bar{\sigma})\sup_{\|w-w^*\|\leq\epsilon_1}\frac{\beta(w)}{\|w-w^*\|}+(1+(\bar{\sigma}-1)\alpha)\|J(w^*)\|\right)\|w-w^*\| \leq \\ 2\|J(w^*)^{-1}\|\left(\delta_1+\bar{\alpha}(1+\bar{\sigma})\sup_{\|w-w^*\|\leq\epsilon_1}\frac{\beta(w)}{\|w-w^*\|}+(1+(\bar{\sigma}-1)\bar{\alpha})\|J(w^*)\|\right)\|w-w^*\| \leq \\ 2\|J(w^*)^{-1}\|\tfrac{r}{2\|J(w^*)^{-1}\|}\|w-w^*\|= \\ r\|w-w^*\|,\end{gathered}$$

which finishes the proof. □

Before we show the last result about the linear local convergence of the feasible IPM with quasi-Newton approach, we need a well-known result.

Definition 2.13. *Given a matrix $C \in \mathbb{R}^{m\times n}$, the Frobenius norm $\|\cdot\|_F$ is defined as*

$$\|C\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} C[i,j]^2},$$

where $C[i,j]$ is the element contained in i-th row and j-th column of matrix C.

Lemma 2.14. *Suppose that $\|\cdot\|$ is the matrix norm induced by the Euclidian vector norm and $\|\cdot\|_F$ is the Frobenius norm, both defined in $\mathbb{R}^{n\times n}$. Given any matrix $C \in \mathbb{R}^{n\times n}$, it occurs*

$$\|C\|_2 \leq \|C\|_F.$$

Proof. Let us consider the sets

$$D = \left\{ \frac{\|Cx\|_2}{\|x\|_2}; x \neq 0 \right\} \text{ and } E = \{\|Cx\|_2; \|x\|_2 = 1\}.$$

It is clear that $E \subset D$. On the other hand, given any scalar $y \in D$,

$$y = \frac{\|Cx\|_2}{\|x\|_2} = \frac{1}{\|x\|_2}\|Cx\|_2 = \left\| C\left(\frac{x}{\|x\|_2}\right)\right\|_2,$$

which is an element of E. Therefore, we have $D \subset E$ and then $D = E$. This fact implies that, through Definition 1.12,

$$\|C\|_2 = \sup_{\|x\|_2=1} \{\|Cx\|_2\}. \tag{2.29}$$

Let any $x \in \mathbb{R}^n$ such that $\|x\|_2 = 1$. By the definition of $\|\cdot\|_2$ it occurs

$$\|Cx\|_2 = \sqrt{\sum_{i=1}^{n} (C_{i\cdot}^T x)^2},$$

where $C_{i\cdot}$ is the i-th row vector of C. The Cauchy-Schwartz inequality affirms that, given any vectors u and v in a space with an inner product, it occurs that

$$|u^T v| \leq \|u\|\|v\|.$$

From this, we have

$$(C_{i\cdot}^T x)^2 \leq (\|C_{i\cdot}\|_2\|x\|_2)^2 = \|C_{i\cdot}\|_2^2,$$

which implies

$$\|Cx\|_2 \leq \sqrt{\sum_{i=1}^{n} \|C_{i\cdot}\|_2^2} = \|C\|_F.$$

Therefore,

$$\sup_{\|x\|_2=1} \{\|Cx\|_2\} \leq \|C\|_F,$$

which implies by (2.29) that

$$\|C\|_2 \leq \|C\|_F.$$

□

Let us denote as **Hypothesis H** the assumption that all the iterates w_k obtained from an IPM with quasi-Newton approach satisfy $(x_k, z_k) > 0$. We can note that, usually, this hypothesis is not necessary, since given any $k \in \mathbb{N}$ where $(x_k, z_k) > 0$ we can choose the step length α_k sufficiently close to 0 such that $(x_{k+1}, z_{k+1}) > 0$ holds. However, in this case, we need to use Lemma 2.12, and then it must occurs $\alpha_k \in (\bar{\alpha}, 1)$ for all $k \in \mathbb{N}$, which does not allow us to select the value of α_k as close to 0 as we want. Under this hypothesis, Theorem 2.15 guarantees the linear local convergence of this feasible method. About this result, it is interesting to note that it is valid for the Broyden method using $B_0 = J(w_0)$, and not for any previously selected B_0 matrix. This is because, in its proof, it is necessary to use Lemma 1 of [8], and also to use the fact that this Jacobian J is a continuous function in order to start an induction process involving the matrices B_k.

Theorem 2.15. *Let $r \in (0,1)$. There exists $\epsilon > 0$ such that, if w_0 is feasible, $\|w_0 - w^*\| \leq \epsilon$, $B_0 = J(w_0)$, B_k is the Broyden approach of matrix $J(w_k)$ for all $k \in \mathbb{N}$ and Hypothesis H is satisfied, then the sequence given by*

$$w_{k+1} = w_k - \alpha_k B_k^{-1} \left(F(w_k) - \begin{bmatrix} 0 \\ 0 \\ \bar{\sigma}\mu_k e \end{bmatrix} \right), \tag{2.30}$$

where $\alpha_k \in (\bar{\alpha}, 1)$ for all $k \in \mathbb{N}$ considering $\bar{\alpha}$ and $\bar{\sigma}$ provided by Lemma 2.12, is well defined and converges linearly to w^*.

Proof. Given $r \in (0,1)$, let us consider $\bar{\alpha}, \bar{\sigma}, \epsilon_1, \delta_1 > 0$ obtained from Lemma 2.12 and $\delta > 0$ such that $\delta < \delta_1$. Observe that the Jacobian $J : \mathbb{R}^N \to \mathbb{R}^{N \times N}$ is Lipschitz continuous in all $\mathbb{R}^N$. In fact, given any $w = (x, \lambda, z)$ and $w' = (x', \lambda', z')$ in $\mathbb{R}^N$ we have that

$$\|J(w) - J(w')\| = \left\| \begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z & 0 & X \end{bmatrix} - \begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z' & 0 & X' \end{bmatrix} \right\| =$$

$$\left\| \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ Z - Z' & 0 & X - X' \end{bmatrix} \right\| \leq \left\| \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ Z - Z' & 0 & X - X' \end{bmatrix} \right\|_F =$$

$$\|(x - x', 0, z - z')\| \leq \|(x - x', \lambda - \lambda', z - z')\| = \|w - w'\|,$$

where $\|\cdot\|_F$ is the Frobenius norm.

As J is continuous, it is possible to obtain $\epsilon_2 > 0$ sufficiently close to 0 such that $\|w_0 - w^*\| < \epsilon_2$ implies $\|B_0 - J(w^*)\| = \|J(w_0) - J(x^*)\| \leq \delta$ and, at the same time, the ϵ defined as $\epsilon = \min\{\epsilon_1, \epsilon_2\}$ is close enough to 0 so that $\delta + \frac{1}{2}\epsilon\frac{1+r}{1-r} \leq \delta_1$ is satisfied.

Let us prove by induction that

$$\|w_k - w^*\| \leq \epsilon \quad \text{and} \quad \|B_k - J(w^*)\| \leq \delta_1 \; \forall \, k \in \mathbb{N}, \tag{2.31}$$

where the sequence $\{w_k\}$ is generated by (2.30).

From the hypotheses, there are w_0 feasible and $B_0 = J(w_0)$ such that

$$\|w_0 - w^*\| \leq \epsilon \quad \text{and} \quad \|B_0 - J(w^*)\| \leq \delta.$$

Suppose, as induction hypothesis, that we have $k \in \mathbb{N}$ such that

$$\|w_p - w^*\| \leq \epsilon \quad \text{and} \quad \|B_p - J(w^*)\| \leq \delta_1$$

for all $p \leq k$. By Lemma 2.12,

$$\|w_{k+1} - w^*\| \leq r\|w_k - w^*\| \leq r\epsilon \leq \epsilon.$$

Let us prove with another induction process that

$$\|B_{k+1} - J(w^*)\| \leq \|B_0 - J(w^*)\| + \frac{1}{2}(1 + r)\|w_0 - w^*\| \left(\sum_{j=0}^{k} r^j \right). \tag{2.32}$$

In fact, by Lemma 8.2.1 of [3] we have

$$\|B_1 - J(w^*)\| \leq \|B_0 - J(w^*)\| + \frac{1}{2}(1 + r)\|w_0 - w^*\|,$$

which completes the first step of the process. Suppose, as induction hypothesis, that given p such that $1 \leq p \leq k$ it occurs

$$\|B_p - J(w^*)\| \leq \|B_0 - J(w^*)\| + \frac{1}{2}(1 + r)\|w_0 - w^*\| \left(\sum_{j=0}^{p-1} r^j \right).$$

Therefore, by Lemma 8.2.1 of [3]

$$\|B_{p+1} - J(w^*)\| \leq \|B_p - J(w^*)\| + \frac{1}{2}(\|w_{p+1} - w^*\| + \|w_p - w^*\|),$$

which implies, by the induction hypothesis, that

$$\|B_{p+1} - J(w^*)\| \leq$$
$$\|B_0 - J(w^*)\| + \tfrac{1}{2}(1+r)\|w_0 - w^*\| \left(\sum_{j=0}^{p-1} r^j\right) + \tfrac{1}{2}(\|w_{p+1} - w^*\| + \|w_p - w^*\|) \leq$$
$$\|B_0 - J(w^*)\| + \tfrac{1}{2}(1+r)\|w_0 - w^*\| \left(\sum_{j=0}^{p-1} r^j\right) + \tfrac{1}{2}(1+r)\|w_p - w^*\| \leq$$
$$\|B_0 - J(w^*)\| + \tfrac{1}{2}(1+r)\|w_0 - w^*\| \left(\sum_{j=0}^{p-1} r^j\right) + \tfrac{1}{2}(1+r)r^p\|w_0 - w^*\| =$$
$$\|B_0 - J(w^*)\| + \tfrac{1}{2}(1+r)\|w_0 - w^*\| \left(\sum_{j=0}^{p} r^j\right),$$

which completes the last step of the process and then proves the veracity of (2.32). With this, it occurs

$$\|B_{k+1} - J(w^*)\| \leq \|B_0 - J(w^*)\| + \frac{1}{2}(1+r)\|w_0 - w^*\| \left(\sum_{j=0}^{\infty} r^j\right) \leq \delta + \frac{1}{2}\epsilon\frac{1+r}{1-r} \leq \delta_1,$$

which finishes the induction and proves (2.31). Thus, by Lemma 2.12 we have

$$\|w_{k+1} - w^*\| \leq r\|w_k - w^*\| \; \forall \, k \in \mathbb{N}.$$

Let us prove that if $w_k = (x_k, \lambda_k, z_k)$ is feasible, then $w_{k+1} = (x_{k+1}, \lambda_{k+1}, z_{k+1})$ is also feasible. In fact, since w_k is feasible, it occurs

$$A^T\lambda_k + z_k = c$$

and

$$Ax_k = b.$$

Since the Jacobian matrix at some point w_k is the coefficient matrix of (2.15), that is

$$J(w_k) = \begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ Z_k & 0 & X_k \end{bmatrix}, \tag{2.33}$$

we can see that the first two row blocks of this matrix are constant for all $k \in \mathbb{N}$, and then it coincides with these blocks in matrix B_0, which is equals to $J(w_0)$ by hypothesis. One of the results provided by Lemma 1 of [8] is that, if B_{k+1} is obtained through a Broyden's update (1.8) of B_k and the first two row blocks of B_k coincide with those present in (2.33), then the first two row blocks of B_{k+1} will also coincide. Therefore, since $B_0 = J(w_0)$, we

have that all the matrices B_k obtained from Broyden's method in this case have the first two row blocks equal to those present in (2.33).

If we denote

$$\begin{bmatrix} \Delta x_k \\ \Delta \lambda_k \\ \Delta z_k \end{bmatrix} = -B_k^{-1} \left(F(w_k) - \begin{bmatrix} 0 \\ 0 \\ \bar{\sigma}\mu_k e \end{bmatrix} \right),$$

by (2.30) we have $w_{k+1} = w_k + \alpha_k(\Delta x_k, \Delta \lambda_k, \Delta z_k)$, in the same way as Step 3 of Algorithm 2.9. By considering the fact that B_k have the first two row blocks equal to those of Jacobian matrix (2.33), it occurs

$$A^T \Delta \lambda_k + \Delta z_k = c - A^T \lambda_k - z_k = 0$$

and

$$A \Delta x_k = b - A x_k = 0.$$

Therefore,

$$A^T \lambda_{k+1} + z_{k+1} = A^T(\lambda_k + \Delta \lambda_k) + (z_k + \Delta z_k) = (A^T \lambda_k + z_k) + (A^T \Delta \lambda_k + \Delta z_k) = c$$

and

$$A x_{k+1} = A(x_k + \Delta x_k) = A x_k + A \Delta x_k = b.$$

Through Hypothesis H, we have $(x_{k+1}, z_{k+1}) > 0$, which implies that w_{k+1} is feasible, as we wanted.

By considering the last affirmation proved, we obtain that the feasibility of w_0 guarantee this condition for all the sequence $\{w_k\}$, which finishes the proof. □

We can observe that the result built through Lemma 2.12 and Theorem 2.15 requires α_k sufficiently close to 1 for all $k \in \mathbb{N}$ and $\bar{\sigma}$ sufficiently close to 0. This goes against the conditions normally used in analogous results about interior point methods, which normally require σ_k sufficiently close to 1 and α_k sufficiently close to 0 as $k \to \infty$. This first hypothesis aims to approximate the central path of the iterates along the iterations, while the second one aims precisely to use this measure to guarantee the positivity of vectors x_k and z_k at each iteration k, which avoids the use of Hypothesis H.

It is clear that the necessity of using Hypothesis H to build this convergence result restricts its relevance with respect to applicability. Being necessary the use of this hypothesis, this result may be relevant in an application situation where, for some reason, we can guarantee that positivity will always hold for all iterates, and therefore we don't need to worry about it.

One possible way to construct a similar result while avoiding the use of the Hypothesis H might be to use strategies related to the concept of central path neighborhoods, which is a common assumption in interior points methods (see [17, Chapter 14]), which was not considered in the present work.

Chapter 3

Nonsmooth equations in linear programming

In this chapter, we apply nonsmooth variants of Newton's and Broyden's methods to solve a system of nonsmooth equations equivalent to the KKT conditions of a linear programming problem, in order to study their local and global convergence properties. The main references of this chapter are Gomes-Ruggiero, Martínez and Santos [6] and Qi and Jiang [20].

The work developed by Gomes-Ruggiero, Martínez and Santos [6] presents an algorithm that aims to solve a system of nonlinear equations without hypotheses about its differentiability. This algorithm uses local methods in its operation and has attractive properties about global convergence. On the other hand, Qi and Jiang [20] show definitions of systems of nonsmooth equations equivalent to the KKT conditions of nonlinearly constrained programming problems and promote a very interesting study, relating characteristics of the problem with properties of these systems, such as differentiability, for example.

3.1 Nonsmooth equations

Qi and Jiang [20] considered the nonlinearly constrained programming problem (3.1)

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & h(x) = 0 \\ & g(x) \leq 0, \end{array} \tag{3.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^p$ and $h : \mathbb{R}^n \to \mathbb{R}^q$ are continuously differentiable functions. The KKT conditions of this problem can be written as

$$\begin{aligned} \nabla f(x) + \sum_{j=1}^{p} u_j \nabla g_j(x) + \sum_{j=1}^{q} v_j \nabla h_j(x) = 0 \\ u \geq 0 \\ g(x) \leq 0 \\ u^T g(x) = 0 \\ h(x) = 0. \end{aligned} \tag{3.2}$$

In their work, it is showed some alternative formulations for conditions (3.2), like for example the well-known Burmeister-Fischer formulation. The system of nonsmooth equations which will be used in this work is the one developed in Pang [18], which is described in (3.3):

$$\begin{aligned} \nabla f(x) + \sum_{j=1}^{p} u_j \nabla g_j(x) + \sum_{j=1}^{q} v_j \nabla h_j(x) = 0 \\ h(x) = 0 \\ \min\{u_1, -g_1(x)\} = 0 \\ \vdots \\ \min\{u_p, -g_p(x)\} = 0. \end{aligned} \tag{3.3}$$

If we define in (3.1) the function $f : \mathbb{R}^n \to \mathbb{R}$ as $f(x) = c^T x$, $g : \mathbb{R}^n \to \mathbb{R}^n$ as $g(x) = -x$ and $h : \mathbb{R}^n \to \mathbb{R}^m$ as $h(x) = Ax - b$, given any $x \in \mathbb{R}^n$, the gradient of f at x is $\nabla f(x) = c$, the Jacobian of g at this point is $Jg(x) = -I_n$, where $I_n \in \mathbb{R}^{n \times n}$ is the identity matrix, and the Jacobian of h at x is $Jh(x) = A$. Therefore, the derivatives f', g' and h' of all these functions are constant, and then continuous. This fact implies that f, g and h are continuously differentiable functions, and then (2.1) is a problem with form (3.1). Moreover, by defining $u = z$ and $v = -\lambda$, we obtain that the KKT conditions (3.2) coincide with those shown in (2.2). Therefore, the KKT conditions (2.2) of a linear programming problem (2.1) can be write equivalently as the following system of nonsmooth equations:

$$\begin{aligned} A^T \lambda + z = c \\ Ax = b \\ \min\{x_1, z_1\} = 0 \\ \vdots \\ \min\{x_n, z_n\} = 0. \end{aligned} \tag{3.4}$$

Thus, from Lemma 2.4, $x^* \in \mathbb{R}^n$ solves the linear programming problem (2.1) if and only if there are $\lambda^* \in R^m$ and $z^* \in \mathbb{R}^n$ such that x^*, λ^* and z^* satisfy conditions (3.4).

Let us now consider function $F : \mathbb{R}^N \to \mathbb{R}^N$, where $N = 2n + m$ with n and m from (2.1), defined by

$$F(w) = \begin{bmatrix} A^T\lambda + z - c \\ Ax - b \\ \min\{x_1, z_1\} \\ \vdots \\ \min\{x_n, z_n\} \end{bmatrix}. \tag{3.5}$$

Finding a solution of (3.4), and therefore a solution of problem (2.1), is equivalent to find $w^* \in \mathbb{R}^N$ that satisfies equation

$$F(w) = 0. \tag{3.6}$$

A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be nonsmooth if there is $x \in \mathbb{R}^n$ such that the derivative of f at this point does not exist, and then neither the Jacobian. However, sometimes when methods are applied in order to find a $x \in \mathbb{R}^n$ such that $f(x) = 0$, as for example in Newton-based algorithms, it can be helpful to use all the available information about the derivative of this function on a neighborhood of this point. To that end, there exists the concept of generalized derivatives in the sense of Clarke [2], which is defined as

$$\partial f(x) = \text{ the convex hull of } \partial_B f(x)$$

where

$$\partial_B f(x) = \left\{ \lim_{x_i \to x,\ x_i \in D_f} Jf(x_i) \right\} \tag{3.7}$$

with $Jf(x_i)$ being the Jacobian matrix of f at x_i and D_f being the set of points at which f is differentiable. In this sense, for example, if we consider $f : \mathbb{R}^2 \to \mathbb{R}$ defined by

$$f(x_1, x_2) = \min\{x_1, x_2\},$$

we obtain that this function is nonsmooth, since the derivative of f does not exist at every point $(x_1, x_2) \in \mathbb{R}^2$ such that $x_1 = x_2$. Moreover, by considering the concept of generalized derivatives at the point $(0, 0) \in \mathbb{R}^2$, we have

$$\partial_B f(0,0) = \left\{ \begin{bmatrix} 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \end{bmatrix} \right\},$$

which implies

$$\partial f(0,0) = \left\{ (1-t) \begin{bmatrix} 0 & 1 \end{bmatrix} + t \begin{bmatrix} 1 & 0 \end{bmatrix};\ t \in [0,1] \right\}$$

and then

$$\partial f(0,0) = \left\{ \begin{bmatrix} t & 1-t \end{bmatrix};\ t \in [0,1] \right\}.$$

Through the definition of generalized derivative, it is possible also to define semismooth functions. However, before we show this concept, we need to define what is a locally Lipschitzian function.

Definition 3.1. *A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be locally Lipschitzian if, given any $x \in \mathbb{R}^n$, there is $\delta > 0$ such that the restriction of function f to the open ball centered in x with radius equals to δ is a Lipschitz continuous function.*

Definition 3.2. *A locally Lipschitzian function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be semismooth at $x \in \mathbb{R}^n$ if*

$$\lim_{\substack{V \in \partial f(x+td') \\ d' \to d, t \to 0^+}} \{Vd'\} \tag{3.8}$$

exists for any $d \in \mathbb{R}^n$.

If f is a semismooth function at x, then f is directionally differentiable at x, being the directional derivative $f'(x;d)$ equals to the limit (3.8). Going further, a semismooth function f can also be strongly semismooth, if it obeys Definition 3.3.

Definition 3.3. *Suppose $f : \mathbb{R}^n \to \mathbb{R}^m$ is semismooth at $x \in \mathbb{R}^n$. f is called strongly semismooth at x if given any $V \in \partial f(x+d)$ with $d \to 0$,*

$$Vd - f'(x;d) = o(\|d\|^2),$$

where $f'(x;d)$ is the directional derivative of f at x in relation to the direcion d.

The linear programming problem (2.1) can be seen as a particular case of some well-known classes of optimization problems. The properties of these problems can help us identify important characteristics about the function F defined in (3.5). Let us look at some of them.

Definition 3.4. *Given a function $f : \mathbb{R}^n \to \mathbb{R}^n$ and a closed convex set $\mathcal{C}$ in $\mathbb{R}^n$, a variational inequality problem is defined as the task of*

$$\begin{aligned} &\text{find } x \in \mathcal{C} \text{ such that} \\ &f(x)^T(y-x) \geq 0 \text{ for all } y \in \mathcal{C}. \end{aligned} \tag{3.9}$$

Note that, defining $f(x) = c$ for all $x \in \mathbb{R}^n$ and $\mathcal{C} = \{x \in \mathbb{R}^n; x \geq 0, Ax = b\}$, we obtain that problem (2.1) is a variational inequality problem.

Definition 3.5. *Consider the problem (3.1). If f, g and h are twice continuously differentiable and $\nabla^2 f, \nabla^2 g$ and $\nabla^2 h$ are locally Lipschitzian, then (3.1) is called an LC^2 problem. Moreover, if f, g and h are not necessarily twice continuously differentiable but their derivatives are semismooth, then (3.1) is called an SC^1 problem.*

We have already seen that if we define in (3.1) the function $f : \mathbb{R}^n \to \mathbb{R}$ as $f(x) = c^T x$, $g : \mathbb{R}^n \to \mathbb{R}^n$ as $g(x) = -x$ and $h : \mathbb{R}^n \to \mathbb{R}^m$ as $h(x) = Ax - b$, then (2.1) is a problem

with form (3.1). But note also that, since f', g' and h' are constant in this case, they are smooth, and then problem (2.1) can be classified as an SC^1 problem. Moreover, $\nabla^2 f, \nabla^2 g$ and $\nabla^2 h$ are null, then they are locally Lipschitzians, and therefore (2.1) is also an LC^2 problem.

Theorem 3.6 relates the concepts of SC^1 problem and LC^2 problem with semismoothness and strongly semismoothness of its system of nonsmooth equations.

Theorem 3.6. *If (3.1) is an SC^1 problem, then the function F defined as the nonsmooth version (3.3) of its KKT conditions is a semismooth function. If (3.1) is an LC^2 problem, then F is a strongly semismooth function.*

Proof. The proof is a direct application of Theorem 3.2 of [20] to (3.1). □

Through Theorem 3.6, we guarantee that function F defined in (3.5) is strongly semismooth and then, in particular, is also locally Lipschitzian. We can also obtain a result about the differentiability of this function F in some points of its domain. This result involves the concept of strong Fréchet differentiability, which is presented in Definition 3.7.

Definition 3.7. *A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is strongly Fréchet differentiable at $x \in \mathbb{R}^n$ if f is differentiable at this point and*

$$\lim_{\substack{(x_1, x_2) \to (x, x) \\ x_1 \neq x_2}} \frac{f(x_1) - f(x_2) - f'(x)(x_1 - x_2)}{\|x_1 - x_2\|} = 0$$

holds.

By adapting item (a) of Theorem 3.3 in [20] and considering problem (2.1), we obtain a result about the differentiability of function F defined by (3.5), which is showed in Theorem 3.8.

Theorem 3.8. *The function $F : \mathbb{R}^N \to \mathbb{R}^N$ given by (3.5) is differentiable, strongly Fréchet differentiable and continuously differentiable at $w = (x, \lambda, z) \in \mathbb{R}^N$, where $x = (x_1, ..., x_n)$ and $z = (z_1, ..., z_n)$, if and only if $x_i \neq z_i$ for all $i \in \{1, ..., n\}$.*

Proof. The proof is a direct application of Theorem 3.3 of [20] to (3.1). □

Observe that the function F defined in (3.5) is continuous, but as we see in Theorem 3.8, it is only differentiable in the set $\mathcal{G} = \{w \in \mathbb{R}^N; x_i \neq z_i \; \forall \; i \in \{1, ..., n\}\}$. The Jacobian of this function, which we denote as $J : \mathcal{G} \to \mathbb{R}^{N \times N}$, is defined as

$$J(w) = \begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ B & 0 & C \end{bmatrix}$$

where $B \in \mathbb{R}^{n\times n}$ is a diagonal matrix such that each diagonal element $B[i,i]$ is defined by

$$B[i,i] = \begin{cases} 0, \text{ if } \min\{x_i, z_i\} = z_i \\ 1, \text{ if } \min\{x_i, z_i\} = x_i \end{cases}, \tag{3.10}$$

and the elements of diagonal matrix $C \in \mathbb{R}^{n\times n}$, analogously, are defined by

$$C[i,i] = \begin{cases} 1, \text{ if } \min\{x_i, z_i\} = z_i \\ 0, \text{ if } \min\{x_i, z_i\} = x_i \end{cases}.$$

Given $w \in \mathcal{G}$, for each $i \in \{1,...,n\}$ there are two possibilities: $\min\{x_i, z_i\} = x_i$ or $\min\{x_i, z_i\} = z_i$. Therefore, if we consider the combination between these possibilities for each index i, we conclude that each vector of $\mathcal{G}$ satisfies one of 2^n possible combinations. From this, let $\mathcal{D}_1, ..., \mathcal{D}_{2^n} \subset \mathcal{G}$ subsets such that each one contains the vectors that satisfies one specific combination of the 2^n possible. With this definition, we observe the following properties:

1. $\mathcal{D}_j \neq \varnothing$ for all $j \in \{1,...,2^n\}$, since we can find a vector of $\mathcal{G}$ that satisfy any possible combination;
2. $\mathcal{D}_i \cap \mathcal{D}_j = \varnothing$ if $i \neq j$, because is not possible that $w \in \mathcal{G}$ to satisfy two different combinations;
3. $\mathcal{G} = \bigcup_{j=1}^{2^n} \mathcal{D}_j$, since every element of $\mathcal{G}$ satisfy some possible combination.

Lemma 3.9 shows two more relevant properties satisfied by sets $\mathcal{D}_j$ for all $j \in \{1,...,2^n\}$.

Lemma 3.9. *$\mathcal{D}_j \subset \mathcal{G}$ is an open convex set for any $j \in \{1,...,2^n\}$.*

Proof. Firstly, let us prove that $\mathcal{D}_j$ is an open set. Consider any $w = (x, \lambda, z) \in \mathcal{D}_j$ and $\epsilon > 0$ such that $\epsilon < \min\left\{\frac{|x_i - z_i|}{2}; i \in \{1,...,n\}\right\}$, which is possible since $w \in \mathcal{G}$. Suppose that $w' = (x', \lambda', z')$ is a vector contained in the open ball $B(w, \epsilon)$, with center being w and radius equal to ϵ. Consider any $i \in \{1,...,n\}$, observe that $|x'_i - x_i| \leq \|w' - w\| < \epsilon < \frac{|x_i - z_i|}{2}$, which implies that

$$x_i - \frac{|x_i - z_i|}{2} < x'_i < x_i + \frac{|x_i - z_i|}{2}. \tag{3.11}$$

Analogously, we have

$$z_i - \frac{|x_i - z_i|}{2} < z'_i < z_i + \frac{|x_i - z_i|}{2}. \tag{3.12}$$

From this, if $\min\{x_i, z_i\} = x_i$ then $z_i > x_i$, and therefore

$$z_i - \frac{|x_i - z_i|}{2} = \frac{x_i + z_i}{2} = x_i + \frac{|x_i - z_i|}{2},$$

which implies by (3.11) and (3.12) that $z_i' > x_i'$ and thus, $\min\{x_i', z_i'\} = x_i'$. On the other hand, if $\min\{x_i, z_i\} = z_i$ then $x_i > z_i$, therefore

$$x_i - \frac{|x_i - z_i|}{2} = \frac{x_i + z_i}{2} = z_i + \frac{|x_i - z_i|}{2}$$

and thus, by (3.11) and (3.12), we have $x_i' > z_i'$, which implies $\min\{x_i', z_i'\} = z_i'$. With this, we have $w' \in \mathcal{D}_j$, consequently $B(w, \epsilon) \subset \mathcal{D}_j$, and then $\mathcal{D}_j$ is an open set.

Let us prove that $\mathcal{D}_j$ is a convex set. Consider $w' = (x', \lambda', z')$, $w'' = (x'', \lambda'', z'') \in \mathcal{D}_j$ and the set

$$[w', w''] = \{tw' + (1-t)w''; t \in [0,1]\}.$$

If $w = (x, \lambda, z) \in [w', w'']$ then

$$x_i = tx_i' + (1-t)x_i'' \text{ and } z_i = tz_i' + (1-t)z_i'' \tag{3.13}$$

for all $i \in \{1, ..., n\}$ and some $t \in [0, 1]$. From this, given an index i, we have two possibilities:

$$\min\{x_i', z_i'\} = x_i' \text{ and } \min\{x_i'', z_i''\} = x_i'',$$

or

$$\min\{x_i', z_i'\} = z_i' \text{ and } \min\{x_i'', z_i''\} = z_i''.$$

If the first one occurs, then $z_i' > x_i'$ and $z_i'' > x_i''$, which implies by (3.13) that $z_i > x_i$, and then $\min\{x_i, z_i\} = x_i$. Analogously, if the second one occurs, then $x_i' > z_i'$ and $x_i'' > z_i''$, so by (3.13) we have $x_i > z_i$, and thus $\min\{x_i, z_i\} = z_i$. Therefore, $w \in \mathcal{D}_j$, which implies $[w', w''] \subset \mathcal{D}_j$ and finishes the proof. □

Observe that at any given set $\mathcal{D}_j$ it occurs that $\min\{x_i, z_i\}$ is equal to x_i or is equal to z_i throughout all the set and for all $i \in \{1, .., n\}$, therefore the Jacobian J is constant at $\mathcal{D}_j$, which is denoted as J_j. Moreover, as the function F is defined by (3.5), we obtain that the restriction of the function F to the set $\mathcal{D}_j$ is linear, then defining a function $F_j : \mathbb{R}^N \to \mathbb{R}^N$ by

$$F_j(w) = J_j w + \begin{bmatrix} -c \\ -b \\ 0 \end{bmatrix}, \tag{3.14}$$

we have that the restriction of functions F and F_j to the set $\mathcal{D}_j$ coincides.

3.2 Convergence of Newton's and Broyden's methods

During our research, no studies were found in the literature regarding the application of Newton's and Broyden's methods to solve linear programming problems through systems of nonsmooth equations. Therefore, this study was conducted and, in this section, results related to the application of these methods are presented in order to solve the system

$$\begin{aligned} A^T\lambda + z &= c \\ Ax &= b \\ \min\{x_1, z_1\} &= 0 \\ &\vdots \\ \min\{x_n, z_n\} &= 0, \end{aligned} \tag{3.15}$$

and therefore the linear programming problem (2.1). Although this study was not complex, it was possible to obtain interesting results through it, such as an instantaneous local convergence result for Newton's method, for example. Throughout this section, the function F considered is the one obtained through (3.5). If the derivatives of F are needed at w_k where F is not differentiable, we take some $V \in \partial_B F(w_k)$, which is defined in (3.7).

Theorem 3.10. *If $J(w^*)$ is nonsingular and there is $j \in \{1, ..., 2^n\}$ such that $w_0, w^* \in \mathcal{D}_j$, then Newton's method finds w^* in one iteration.*

Proof. By hypothesis, $w_0, w^* \in \mathcal{D}_j$, then $J_j = J(w^*) = J(w_0)$ and therefore, by (3.14), we have

$$F(w_0) = F_j(w_0) = J(w^*)w_0 + \begin{bmatrix} -c \\ -b \\ 0 \end{bmatrix}.$$

Through the application of Newton's method, we have that

$$w_1 = w_0 - J(w_0)^{-1}F(w_0) = w_0 - J(w^*)^{-1}\left(J(w^*)w_0 + \begin{bmatrix} -c \\ -b \\ 0 \end{bmatrix}\right),$$

which implies

$$w_1 + J(w^*)^{-1}\begin{bmatrix} -c \\ -b \\ 0 \end{bmatrix} = 0.$$

By considering the function F_j defined in (3.14), it occurs $F_j(w_1) = 0$, but $F_j(w^*) = 0$ and, since $J(w^*)$ is nonsingular, F_j is injective. Therefore, $w_1 = w^*$, which finishes the proof. □

As a consequence of Theorem 3.10, we obtain the local convergence of Newton's method to solve the nonsmooth equations (3.4) in Corollary 3.11.

Corollary 3.11. *If $J(w^*)$ is nonsingular, then there is $\epsilon > 0$ such that, if $\|w_0 - w^*\| < \epsilon$, Newton's method finds w^* in one iteration.*

Proof. By hypothesis $J(w^*)$ is nonsingular, then $w^* \in \mathcal{G}$ and therefore there is $j \in \{1, ..., 2^n\}$ such that $w^* \in \mathcal{D}_j$. By Lemma 3.9, $\mathcal{D}_j$ is an open set, then there is $\epsilon > 0$ such that $B(w^*, \epsilon) \subset \mathcal{D}_j$. From this, if $\|w_0 - w^*\| < \epsilon$ then $w_0 \in \mathcal{D}_j$, and therefore by Theorem 3.10, Newton's method converges in one iteration. □

We observe that if we use Broyden's method with $B_0 = J(w_0)$ in order to find $w^* \in \mathbb{R}^N$ such that $F(w^*) = 0$, the results of Theorem 3.10 and Corollary 3.11 remain valid, since the first iteration of this method is equal to the first iteration of Newton's method. Theorem 3.12 shows a result about the convergence of Broyden's method when one does not necessarily defines $B_0 = J(w_0)$.

Theorem 3.12. *Suppose that $J(w^*)$ is nonsingular. There are $\epsilon, \delta > 0$ such that, if*

$$\|w_0 - w^*\| \leq \epsilon \text{ and } \|B_0 - J(w^*)\| \leq \delta$$

then the sequence $\{w_k\}$ generated by Broyden's method is well defined and converges superlinearly to w^*.

Proof. By hypothesis $J(w^*)$ is nonsingular, then $w^* \in \mathcal{G}$, and therefore there is $j \in \{1, ..., 2^n\}$ such that $w^* \in \mathcal{D}_j$. We already show that F is linear and the Jacobian J is constant in $\mathcal{D}_j$, therefore this functions are continuously differentiable and Lipschitz continuous in this set, respectively. From Lemma 3.9, we have that the set $\mathcal{D}_j$ is open and convex. The result follows by Theorem 1.13. □

The best property that an algorithm can have about its convergence guarantee is global convergence. This concept is defined in the Definition 3.13.

Definition 3.13. *An algorithm is said to be globally convergent to w^* if, given any $w_0 \in \mathbb{R}^N$, the sequence $\{w_k\}$ generated converges to w^*.*

By considering just the hypothesis of $J(w^*)$ being nonsingular, unfortunately we can show that there is no result about the global convergence of Newton's method and Broyden's method with $B_0 = J(w_0)$ to solve the nonsmooth equations (3.4). This occurs because, given any problem in form (2.1), the matrix A is not full column rank. Therefore, if we consider a starting point for these methods $w_0 = (x, \lambda, z) \in \mathcal{G}$ such that $\min\{x_i, z_i\} = z_i$ for all $i \in \{1, ..., n\}$, then the matrix B defined by (3.10) is null, which implies that the matrix $J(w_0)$ is singular, and then these methods are not well defined.

In order to verify the veracity of another results about convergence of Newton's and Broyden's methods to solve the nonsmooth equations (3.4), we perform numerical experiments using the Julia language [1]. In these experiments, many linear programming problems were considered, however, we will show results about a specific one. This problem is

$$\begin{aligned} \min \quad & x_1 + x_2 \\ \text{s.t.} \quad & x_1 + x_2 \leq 10 \\ & x_1 \geq 2 \\ & x_1, x_2 \geq 0, \end{aligned} \tag{3.16}$$

which had as feasible set a rectangular triangle with vertices at points $(2,0), (2,8)$ and $(10,0)$, being the point $(2,0)$ the solution of the problem, as shows Figure 3.1.

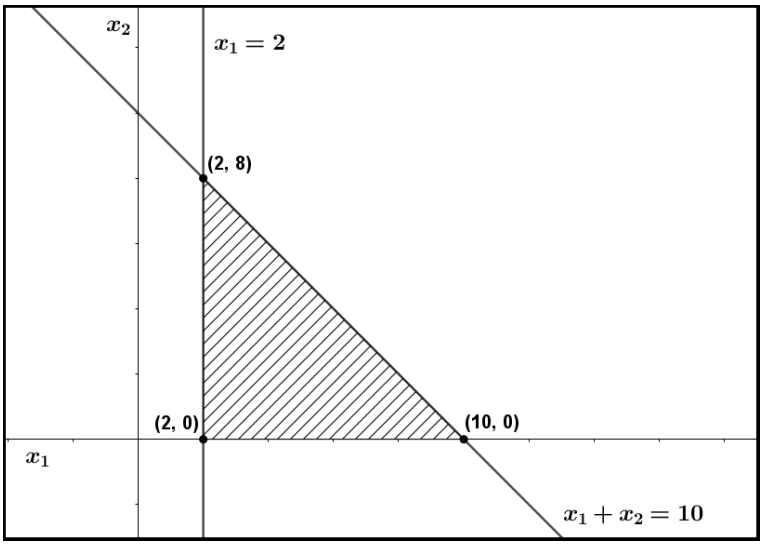

Figure 3.1: Feasible set of problem 3.16.

Through the addition of slack variables x_3 and x_4, this problem can be transformed to the form (2.1) with

$$c = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \; A = \begin{bmatrix} 1 & 1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}, \text{ and } b = \begin{bmatrix} 10 \\ -2 \end{bmatrix}.$$

and then the solution becomes the vector $w^* = (x^*, \lambda^*, z^*) \in \mathbb{R}^{10}$ with

$$x^* = (2, 0, 8, 0), \; \lambda^* = (0, -1) \text{ and } z^* = (0, 1, 0, 1). \tag{3.17}$$

Firstly, we applied Broyden's method with $B_0 = I_{10}$ and a starting point $w_0 = (x_0, \lambda_0, z_0) \in \mathbb{R}^{10}$ such that

$$x_0 = (1000, 300, 1000, 300), \; \lambda_0 = (300, 600) \text{ and } z_0 = (300, 1000, 300, 1000).$$

Therefore, we have that w_0 is far from w^*, however there is $j \in \{1, ..., 16\}$ such that $w_0, w^* \in \mathcal{D}_j$. The algorithm makes 5928 iterations and then stops due to finding a singular matrix B_k. The last $w_k = (x_k, \lambda_k, z_k)$ generated by the method was

$$x_k = (-49.111, 21.779, 35.307, -45.726),\ \lambda_k = (-6.014e19, -1.336e19) \text{ and}$$

$$z_k = (4.677e19, 6.014e19, 6.014e19, 1.336e19),$$

which indicates that the method is diverging. The failure of this experiment proves that is not true the following two results:

- If $J(w^*)$ is nonsingular, $w_0, w^* \in \mathcal{D}_j$ for some $j \in \{1, ..., 2^n\}$ and B_0 is any nonsingular matrix of $\mathbb{R}^{N\times N}$, then Broyden's method generates a sequence $\{w_k\}$ convergent to w^*;
- If $J(w^*)$ is nonsingular, w_0 is any vector of $\mathbb{R}^N$ and B_0 is any nonsingular matrix of $\mathbb{R}^{N\times N}$, then Broyden's method generates a sequence $\{w_k\}$ convergent to w^*.

We already see, through Theorem 3.10, that if $J(w^*)$ is nonsingular and there is $j \in \{1, ..., 2^n\}$ such that $w_0, w^* \in \mathcal{D}_j$, then Broyden's method with $B_0 = J(w_0)$ finds w^* in one iteration. Therefore, is valid think about if at least the occurrence of convergence remains if we apply this method in a linear programming problem having the matrix B_0 close to $J(w^*)$, but not necessarily equals to it. In order to verify this, we use Broyden's Method in problem (3.16) with a starting point $w_0 = (x_0, \lambda_0, z_0) \in \mathbb{R}^{10}$ such that

$$x_0 = (1000, 500, 1000, 500),\ \lambda_0 = (10, 100) \text{ and } z_0 = (500, 1000, 500, 1000),$$

which is in the same set $\mathcal{D}_j$ than the solution w^*. We know from (3.17) that

$$J(w^*) = \begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ B & 0 & C \end{bmatrix}$$

with

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Therefore, we make numerical experiments considering

$$B_0 = \begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ B' & 0 & C' \end{bmatrix}$$

where $B' \in \mathbb{R}^{4\times4}$ is different of B only in the elements $B'[1,1]$ and $B'[3,3]$, and analogously, $C' \in \mathbb{R}^{4\times4}$ is different of C only in $C'[2,2]$ and $C'[4,4]$.

First, we execute experiments considering $B'[1,1]$, $B'[3,3]$, $C'[2,2]$ and $C'[4,4]$ with values around 10^{-1} and 10^{-2}. In such cases, the algorithm either stopped because it found a matrix B_k singular, or worked for 10000 iterations without finding the solution, which is the maximum number of iterations allowed, and therefore did not converge to w^*. In the sequence, we selected $B'[1,1]$, $B'[3,3]$, $C'[2,2]$ and $C'[4,4]$ taking values around 10^{-3} and 10^{-4}. In this situation, the algorithm converged to w^* and more: we observed that the smaller is the values of the elements of matrices B' and C' which we selected, the smaller is the number of iterations used for obtain the convergence. Thus, these numerical experiments make us believe that the result we are checking is true. Unfortunately, we could not obtain a proof for this result yet, but it is formalized next as Conjecture 3.14.

Conjecture 3.14. *Suppose that $J(w^*)$ is nonsingular and $w_0, w^* \in \mathcal{D}_j$ for some $j \in \{1, ..., 16\}$. There exists $\delta > 0$ such that, if $\|B_0 - J(w^*)\| < \delta$ then Broyden's method converges to w^*. The closer B_0 is to $J(w^*)$, the smaller is the number of iterations used to obtain the convergence.*

We also applied Broyden's method in problem 3.16 with $B_0 = J(w^*)$ and the starting point $w_0 = (x_0, \lambda_0, z_0) \in \mathbb{R}^{10}$ being

$$x_0 = (1000, 1000, 1000, 1000),\ \lambda_0 = (1000, 1000) \text{ and } z_0 = (500, 500, 500, 500).$$

The method worked for 450 iterations and then stopped because a singular matrix B_k was found. Therefore, in this experiment the algorithm did not converge to w^*, and the failure of this test proves that is not true the following result:

Suppose that $J(w^*)$ is nonsingular. Given any $w_0 \in \mathbb{R}^N$, there is $\delta > 0$ such that, if $\|B_0 - J(w^*)\| < \delta$, then Broyden's method converges to w^*.

Finally, we wanted to verify the validity of the result:

Suppose that $J(w^*)$ is nonsingular. Given any matrix $B_0 \in \mathbb{R}^{N\times N}$ nonsingular, there is $\delta > 0$ such that, if $\|w_0 - w^*\| < \delta$, then Broyden's method converges to w^*.

Note that we can not prove the non-validity of this result through a counter-example, since if we consider $w_0 = w^*$ the algorithm will converges to w^*, and if we obtain the failure of some numerical experiment using a starting point w_0 different of the solution w^*, it is always possible to make another test with the starting point w_0 closer to the solution. Unfortunately, we have not obtained a proof about the validity of this result. Therefore, we executed numerical experiments in order to observe if it seems to be valid or not.

Broyden's method was applied in problem (3.16) with $B_0 \in \mathbb{R}^{10\times 10}$ being a diagonal matrix where, for all $j \in \{1, ..., 10\}$, $B[j, j] = 40 \cdot j$. In all experiments, we considered $w_0 = (x_0, \lambda_0, z_0) \in \mathbb{R}^{10}$ with

$$x_0 = (2, 0, \psi, 0),\ \lambda_0 = (0, -1) \text{ and } z_0 = (0, 1, 0, 1),$$

where $\psi > 0$, and then the closer ψ is to 8, the closer w_0 is to w^*. It was made four tests, which have ψ equals to $7, 7.9, 7.99$ and 7.999999, respectively. In all these experiments, the algorithm stops at the first iteration by the same reason: the matrix $B_1 \in \mathbb{R}^{10\times 10}$ is singular. Therefore, with these results, the statement considered does not seem to be true.

Table 3.1 summarizes the results obtained in this section. In it, "T" means that the considered statement is true, "F" means that it is false and "?" means that we do not obtained a conclusion about the validity of the claim.

	Newton's Method	Broyden's Method, $B_0 = J(w_0)$	Broyden's Method, B_0 close enough of $J(w^*)$	Broyden's Method, any B_0 nonsingular
w_0 close enough of w^*	T	T	T	?
$\exists\ j \in \{1, ..., 2^n\}$ such that $w_0, w^* \in \mathcal{D}_j$	T	T	?	F
any $w_0 \in \mathbb{R}^N$	F	F	F	F

Table 3.1: Results obtained from the application of Newton's and Broyden's methods in linear programming problems through nonsmooth equations.

3.3 Global convergence

As we can see in Table 3.1, the pure application of Newton's and Broyden's method in order to solve problem (3.6) does not have global convergence good results. Therefore,

in order to obtain an algorithm with a behavior at least closer to the global convergence when applied to this situation, we studied the algorithms presented by Gomes-Ruggiero, Martínez and Santos [6] and Ito and Kunisch [9].

Given a function $f : \mathbb{R}^n \to \mathbb{R}^n$, Ito and Kunisch [9] showed a semismooth Newton method which has a global convergence result to a point $x^* \in \mathbb{R}^n$ such that $f(x^*) = 0$ if this function f satisfies some hypotheses. One of the conditions is that the set

$$S = \{x \in \mathbb{R}^n; \|f(x)\| \leq \|f(x_0)\|\} \tag{3.18}$$

needs to be bounded, where x_0 is the first iterate of the semismooth Newton method.

For our purposes, we are interested in the application of this semismooth Newton method for solve problem (3.6). However, we can obtain an example of problem like (3.6) where condition (3.18) does not holds. This fact proves that the method considered by [9] does not have its global convergence guaranteed when it is applied to a general problem with the form (3.6). In fact, consider the problem (2.1) where

$$c = (1, 1),\ A = \begin{bmatrix} 1 & 1 \end{bmatrix} \text{ and } b = 2.$$

In this case, $n = 2$ and $m = 1$, which implies $N = 5$ and then the domain and counter-domain of F is the set $\mathbb{R}^5$, which have vectors with the form $w = (x, \lambda, z)$ where $x, z \in \mathbb{R}^2$ and $\lambda \in \mathbb{R}$. Consider the sequence $\{w'_k\}$ in $\mathbb{R}^5$ where, for each $k \in \mathbb{N}$, $w'_k = (x'_k, \lambda'_k, z'_k)$ with

$$x'_k = (1, 1),\ \lambda'_k = 1 - k \text{ and } z'_k = (k, k)$$

and select, as starting point of the method, the vector $w_0 = (x_0, \lambda_0, z_0)$ with $x_0 = z_0 = (1, 1)$ and $\lambda_0 = 1$. Therefore, given any $k \in \mathbb{N}$,

$$\|F(w'_k)\| = \left\| \begin{bmatrix} A^T\lambda'_k + z'_k - c \\ Ax'_k - b \\ \min\{1, k\} \\ \min\{1, k\} \end{bmatrix} \right\|. \tag{3.19}$$

Observe that

$$A^T\lambda'_k + z'_k - c = \begin{bmatrix} 1 \\ 1 \end{bmatrix}(1 - k) + \begin{bmatrix} k \\ k \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$
$$Ax'_k - b = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} - b = Ax_0 - b$$

and

$$\min\{1, k\} = 1 = \min\{1, 1\}.$$

Thus, since $\|\cdot\|$ is the Euclidian norm, by (3.19) it occurs

$$\|F(w_k')\| = \left\| \begin{bmatrix} 0 \\ 0 \\ Ax_0 - b \\ \min\{1,k\} \\ \min\{1,k\} \end{bmatrix} \right\| \leq \left\| \begin{bmatrix} A^T\lambda_0 + z_0 - c \\ Ax_0 - b \\ \min\{1,1\} \\ \min\{1,1\} \end{bmatrix} \right\| = \|F(w_0)\|,$$

which implies that, in this case, $w_k' \in S$ for all $k \in \mathbb{N}$. Therefore, since $\{w_k'\}$ is not a bounded sequence, the set S is not bounded also, as we wanted.

Differently from Ito and Kunisch [9], in Gomes-Ruggiero, Martínez and Santos [6] it is presented a general algorithm for global convergence, which has the objective of, given a function $f : \Omega \subset \mathbb{R}^n \to \mathbb{R}^n$, find a point x^* which satisfies

$$f(x) = 0 \tag{3.20}$$

without making smoothness assumptions about this function. The algorithm developed in [6] has relevant characteristics about global convergence, and the problem (3.20) is quite general. These facts motivated us to study the algorithm in depth and perform numerical experiments in order to observe how it works.

The algorithm developed in [6] is based on a monotone reduction of the merit function $g : \mathbb{R}^n \to \mathbb{R}$ defined by

$$g(x) = \frac{1}{2}\|f(x)\|^2.$$

It is described by Algorithm 3.15.

Algorithm 3.15.

Consider $\alpha_0 = 1$, $m_0 = 1$ *and select* $\sigma \in (0,1)$, $a \in (0,1)$, $\eta \in (0, \frac{1}{2}]$ *and* $x_0 \in \mathbb{R}^n$;

for $k = 0, 1, 2, ...$

Step 1: *Choose* $d_k \in \mathbb{R}^n$ *such that*

$$\frac{g(x_k + \alpha_k d_k) - g(x_k)}{\alpha_k} \leq -\sigma g(x_k) \tag{3.21}$$

whenever

$$\alpha_k \leq \frac{a}{m_k}. \tag{3.22}$$

If this choice is not possible, stop (in this case, we say that the algorithm breaks down);

Step 2: *If*

$$g(x_k + \alpha_k d_k) < g(x_k)$$

holds, define $x_{k+1} = x_k + \alpha_k d_k$. *Otherwise, define* $x_{k+1} = x_k$;

Step 3: *If*

$$g(x_{k+1}) \leq (1 - \sigma\alpha_k)g(x_k) \tag{3.23}$$

holds, define $\alpha_{k+1} = 1$ *and* $m_{k+1} = m_k + 1$. *Otherwise, choose*

$$\alpha_{k+1} \in [\eta\alpha_k, (1 - \eta)\alpha_k]$$

and define $m_{k+1} = m_k$*;*

end(for)

We can observe that the closer constant σ is to 1, the more difficult it will be for the inequalities (3.21) and (3.23) to be satisfied. Moreover, the smaller the a we previously chose, the more difficult it is to satisfy (3.22) and, consequently, we have to check (3.21) less frequently.

Through Step 2, the basic idea of Algorithm 3.15 is only accept as a new iterate the point $x_k + \alpha_k d_k$ if it produces a decrease in merit function g. This fact guarantee the monotone decreasing of this function throughout the iterations. We can see also that the inequality (3.22) works like a tolerance for the sufficient decrease of g, through inequality (3.21). If for several consecutive iterations (3.21) is false, then in all these iterations (3.23) did not occur, which made α_k becomes smaller, while m_k remains the same value. Therefore, since the value a is previously defined, (3.22) gets closer and closer to being satisfied, until it is, and then in this iteration either (3.21) is valid or the algorithm breaks down. However, it is valid to mention that in this situation, since α_k tends to 0, inequality (3.21) becomes easier to be satisfied. Then, if at some iteration the algorithm breaks down, probably the strategy used for the search direction d_k at Step 1 was not able to choose a vector with $\|d_k\|$ large enough for $x_k + \alpha_k d_k$ to be in a neighborhood of the solution, where there are points such that the image in relation to g is smaller than $g(x_k)$. This implies that, in all the directions possible to choose, this method was unable to find d_k such that $g(x_k + \alpha_k d_k)$ is not even slightly smaller than $g(x_k)$. Therefore, the algorithm breaking down strongly indicates that the current iterate x_k is a local minimum of g.

It is developed in [6] a theoretical study about Algorithm 3.15 in order to guarantee its convergence results. As the main result in this sense, we have Theorem 3.16.

Theorem 3.16. *If Algorithm 3.15 does not break down, the sequence* $\{x_k\}$ *generated by this algorithm is such that*

$$\lim_{k\to\infty} g(x_k) = 0.$$

Proof. The proof of this result can be found in Theorem 2.4 of [6]. □

There are several forms to implement Algorithm 3.15, which depend directly of the strategy used for choose the direction d_k at Step 1. In this work, analogously to [6], we will use a point s_k replacing $\alpha_k d_k$, which is an approximate minimizer of function

$$\begin{array}{ll} \min & \frac{1}{2}\|V_k s + f(x_k)\|^2 \\ s.t. & \|s\|_\infty \leq \Delta \end{array}, \tag{3.24}$$

where x_k is the current iterate of the algorithm and $V_k \in \partial_B f(x_k)$. We will apply this method in function F defined in (3.5), which we have already seen that is semismooth, and then it is differentiable "almost everywhere". Therefore, through this choice of V_k, we can guarantee that the s_k computed at each iteration k will be the point which minimizes the merit function associated with the linear model of f around x_k in relation to V_k subject to a square region with measure of side equals to Δ. Consequently, the task of solving problem (3.24) brings a "Newton's method strategy" for the algorithm, which can improve its operation.

In the implementation of Algorithm 3.15, since s_k is a solution of problem (3.24), we opt for use it instead of $\alpha_k s_k$, after all, since $\alpha_k \in (0,1)$, $\|\alpha_k s_k\| \leq \|s_k\| \leq \Delta$, and then $\alpha_k s_k$ might not be a solution of (3.24). From this, if we use $\alpha_k s_k$, it will not be the best point in relation with the "Newton's method strategy". Moreover, we can observe that, at each iteration k of Algorithm 3.15, α_k is previously computed, and then have no relation with the selection of direction d_k. Therefore, the use of $\alpha_k d_k$ in Algorithm 3.15 is to, in the case where (3.21) is not satisfied in successive iterations, bring direction $\alpha_k d_k$ closer and closer to the current iterate x_k, in order to identify an eventual local minimum of g and then breaks down the algorithm on it. In this sense, to ensure this characteristic in the implementation of the algorithm, we will use a progressive reduction of Δ in such a situation. The implementation is described in Algorithm 3.17, and was also taken from [6].

Algorithm 3.17.

Consider $m_0 = 1$, $\alpha_0 = 1$ *and given* $\sigma \in (0,1)$, $\eta \in (0, \frac{1}{2}]$, $a \in (0,1)$, $x_0 \in \mathbb{R}^n$ *and* $M > 0$ *such that* $\Delta_0 = M$*;*

for $k = 0, 1, 2, ...$

Step 1: *Compute* s_k *as an approximate solution of*

$$\begin{array}{ll} \min & \frac{1}{2}\|V_k s + f(x_k)\|^2 \\ s.t. & \|s\|_\infty \leq \Delta_k \end{array} \tag{3.25}$$

where $V_k \in \partial_B f(x_k)$*;*

Step 2: *If*

$$\alpha_k \leq \frac{a}{m_k}$$

but

$$\frac{g(x_k + s_k) - g(x_k)}{\alpha_k} \leq -\sigma g(x_k) \tag{3.26}$$

does not hold, stop (the algorithm breaks down);
Step 3: *If*

$$g(x_k + s_k) < g(x_k)$$

holds, define $x_{k+1} = x_k + s_k$. *Otherwise, define* $x_{k+1} = x_k$;
Step 4: *If*

$$g(x_{k+1}) \leq (1 - \sigma\alpha_k)g(x_k)$$

holds, define $\alpha_{k+1} = 1$ *and* $m_{k+1} = m_k + 1$. *Otherwise, choose*

$$\alpha_{k+1} \in [\eta\alpha_k, (1 - \eta)\alpha_k]$$

and define $m_{k+1} = m_k$;
Step 5: *If* $\alpha_{k+1} = 1$, *define* $\Delta_{k+1} = M$. *Otherwise, define*

$$\Delta_{k+1} = \frac{\|s_k\|_\infty}{2};$$

end(for)

The application of local methods, such as Newton's and Broyden's methods, has differences in relation to the application of Algorithm 3.15 in order to solve problem (3.20). Although the iterations of local methods do not usually generate a monotone decreasing of merit function g, when the starting point x_0 is close to the solution x^* of the problem (3.20), these methods in general work very well. On the other hand, Algorithm 3.15 have no restrictions about the starting point x_0, but it usually only converges to a stationary point of the merit function g. Therefore, by considering the characteristics of each type of method, Gomes-Ruggiero, Martínez and Santos [6] built an algorithm which matches iterations of local methods with iterations of global Algorithm 3.17, in order to try enjoy the benefits of each one. This algorithm is described in Algorithm 3.18. To facilitate its presentation, let us define, for any $k \in \mathbb{N}$,

$$\pi_k = \text{Argmin}\{g(x_0), ..., g(x_k)\}$$

and $g(\pi_k) = g(x_0)$ if $k < 0$. About the nomenclature adopted, let us call of "ordinary iteration" the iterations executed by local methods, and of "special iteration" the one executed by Algorithm 3.17.

Algorithm 3.18.

Consider $FLAG = 1$, $m_0 = 1$, $\alpha_0 = 1$ *and given* $\sigma \in (0,1)$, $\eta \in (0, \frac{1}{2}]$, $a \in (0,1)$, $M > 0$ *such that* $\Delta_0 = M$, $q \in \mathbb{N} \cup \{0\}$, $\gamma \in (0,1)$, $x_0 \in \mathbb{R}^n$ *and* B_0 *(if the local method chosen is Broyden's method);*

for $k = 0, 1, 2, ...$

Step 1: *If* $FLAG = 1$, *obtain* x_{k+1} *through an ordinary iteration. Else, obtain* x_{k+1} *using the special iteration;*

Step 2: *If*

$$g(x_{k+1}) \leq \gamma g(\pi_{k-q}) \tag{3.27}$$

set $FLAG \leftarrow 1$. *Else, re-define* $x_{k+1} \leftarrow \pi_{k+1}$ *and* $FLAG \leftarrow -1$;

end(for)

In this algorithm, we can note that M works as a standard trust region for problem (3.24). While it is produced special iterations where (3.26) is valid, and therefore the algorithm is succeeding in decreasing function g, the trust region remains a box with side length equals to M. However, if the algorithm begins to produce successive special iterations where (3.26) does not occur, the box is reduced until it breaks down, probably close to a stationary point of g, or it makes a iteration with sufficiently decrease of g.

We can observe that, in this algorithm, the variable $FLAG$ indicates a satisfactory decrease of g at the iteration. If it occurs, we make an ordinary iteration, if it does not, we make a special iteration. In the operation, q and γ work as "tolerance variables". The greater is q, the greater is the number of iterations that we allow to not produce sufficient decrease of g. Moreover, the more γ is close to 1, the more relaxed is condition (3.27), the sufficient decrease condition.

For Algorithm 3.18, we have also a result analogous to Theorem 3.16. If this algorithm does not break down, there are two possibilities: it makes infinitely many ordinary iterations, or infinitely many special iterations. If the first one occurs, (3.27) was valid infinitely many times, which implies that there is a subsequence $\{x_{k_r}\}$ of $\{x_k\}$ such that

$$\lim_{r \to \infty} g(x_{k_r}) = 0.$$

On the other hand, if the second one occurs, then this algorithm enjoy the convergence results proved in [6] for Algorithm 3.15, in particular Theorem 3.16, showed in this work.

Our interest in this work is to apply Algorithm 3.18 in function $F : \mathbb{R}^N \to \mathbb{R}^N$ defined in (3.5), in order to solve problem (3.6) and, consequently, the linear programming problem (2.1). We will use 2 different versions of this algorithm: the first one using Newton's method and the second one using Broyden's method. From now on, the merit function considered is $g : \mathbb{R}^N \to \mathbb{R}$ where

$$g(w) = \frac{1}{2}\|F(w)\|^2.$$

In first place, it is important to emphasize that, in this case, there are 2 convergence result for Algorithm 3.18 with Newton's method, which can be described as corollaries of Theorem 3.10.

Corollary 3.19. *If $J(w^*)$ is nonsingular and there is $j \in \{1, ..., 2^n\}$ such that $w_0, w^* \in \mathcal{D}_j$, then Algorithm 3.18 with Newton's method finds w^* in one iteration.*

Proof. Since Algorithm 3.18 starts with $FLAG = 1$, the first iteration will provide w_1 through the Newton's method. However, from Theorem 3.10, $w_1 = w^*$. □

Corollary 3.20. *If $J(w^*)$ is nonsingular, there is $\epsilon > 0$ such that, if $\|w_0 - w^*\| < \epsilon$, Algorithm 3.18 with Newton's method finds w^* in one iteration.*

Proof. Since Algorithm 3.18 starts with $FLAG = 1$, the first iteration will provide w_1 through the Newton's method. Therefore, by considering the $\epsilon > 0$ provided by Corollary 3.11, the result follows. □

On the other hand, Theorem 3.12, which guarantees a local convergence result for Broyden's method when it is applied to solve problem (3.6), can not be generalized for Algorithm 3.18 with Broyden's method. This occurs because, in a situation where the hypotheses of Theorem 3.12 are satisfied, we can not guarantee that this convergence will occur in only one iteration, neither that inequality (3.27) will be satisfied at each iteration. Therefore, it is possible that Algorithm 3.18 makes a special iteration at some moment, acquiring a different behavior in relation to the pure Broyden's method.

For the application of Algorithm 3.18, it was considered the same parameters than in [6], which are $\sigma = 10^{-4}$, $\eta = 0.5$, $a = 10^{-5}$, $q = 5$ and $\gamma = 0.9$, except the M, which in [6] is 10^3 and we consider it to be 10^4 in order to verify the operation of the algorithm with initial points w_0 far from the solution w^*. We consider problem (3.16) in order to observe the convergence characteristics of this algorithm in this case.

Since this problem has $n = 4$, its set $\mathcal{G}$ can be written as a union between sets $\mathcal{D}_1, ..., \mathcal{D}_{16}$. By considering $x = (x_1, x_2, x_3, x_4)$ and $z = (z_1, z_2, z_3, z_4)$, let us define each of these sets with the form $[a_1, a_2, a_3, a_4]$, where for each $i \in \{1, 2, 3, 4\}$, if $x_i < z_i$ then $a_i = 1$, otherwise, if $z_i < x_i$ then $a_i = 0$. The problem considered has a small drawback: most of the sets $\mathcal{D}_j$ have singular associated Jacobian, which makes the version of Algorithm 3.18 with the Newton's method frequently stop. Therefore, for the experiments it was made a small modification on the algorithm. The ordinary iterations are executed only if $FLAG = 1$ and $J(w_k)$ (or the matrix B_k, for the version with Broyden's method) is nonsingular. Otherwise, a special iteration is made. With this, we guarantee that the algorithm never stops by singularity, and then will always converges or breaks down.

Algorithm 3.18 with Broyden's method used in [6] have 2 characteristics. The first one is that the matrix B_0 is selected as being $J(w_0)$, and the second one is that every

cycle of ordinary iterations starts with a Newton's iteration. That is, if iteration $k-1$ is special and iteration k is ordinary, then matrix B_k used on it is equal to $J(w_k)$. In our implementation, considering the fact mentioned above, where most of sets $\mathcal{D}_j$ have its respective Jacobian matrix being singular, it was opted to make a modification in the first characteristic: instead of use $B_0 = J(w_0)$, it was opted to consider $B_0 = I_N$, which is the identity matrix. That choice comes from the fact that, through previous experiments applying the pure Broyden's method in order to solve problem (3.16) through (3.6), the choice of $B_0 = I_N$ implies a good behavior for this method, mainly when the starting point w_0 is near the solution w^*, with convergence occurring in most experiments.

For each version of Algorithm 3.18, 16 numerical experiments were run through Julia language, each one with a starting point in a different set $\mathcal{D}_j$. The subproblem (3.25) has always been solved using the package *Optim.jl* [16]. Table 3.2 shows the starting points considered for both algorithms (denoted by w_0) and the respective sets $\mathcal{D}_j$ which contains each one.

Test Nº	w_0	$\mathcal{D}_j$
1	$x_0 = (1000, 1200, 1400, 1600)$ $\lambda_0 = (1000, 500)$ $z_0 = (2000, 2100, 3000, 2500)$	[1,1,1,1]
2	$x_0 = (2.35, 3.4, 2.2, 1.1)$ $\lambda_0 = (0.75, 2)$ $z_0 = (2.5, 3.5, 2.7, 1)$	[1,1,1,0]
3	$x_0 = (3.1, 2.22, 1.3, 6.2)$ $\lambda_0 = (1.3, 0.7)$ $z_0 = (4.2, 3, 1.2, 6.5)$	[1,1,0,1]
4	$x_0 = (1300, 6400, 5500, 4200)$ $\lambda_0 = (3000, 3000)$ $z_0 = (2000, 3000, 6100, 4800)$	[1,0,1,1]
5	$x_0 = (1235, 1700, 500, 950)$ $\lambda_0 = (300, 3000)$ $z_0 = (1000, 2300, 700, 1150)$	[0,1,1,1]
6	$x_0 = (1333, 5300, 4200, 2100)$ $\lambda_0 = (720, 640)$ $z_0 = (1500, 6000, 2200, 1800)$	[1,1,0,0]
7	$x_0 = (2.3, 2.7, 2.5, 5)$ $\lambda_0 = (1.3, 1.2)$ $z_0 = (3.5, 2.5, 4, 3.2)$	[1,0,1,0]

Test Nº	w_0	$\mathcal{D}_j$
8	$x_0 = (720, 980, 1040, 3129)$ $\lambda_0 = (720, 730)$ $z_0 = (700, 1000, 1200, 3000)$	[0,1,1,0]
9	$x_0 = (1200, 700, 850, 1500)$ $\lambda_0 = (500, 600)$ $z_0 = (1500, 600, 800, 1700)$	[1,0,0,1]
10	$x_0 = (1.2, 4, 3.6, 2)$ $\lambda_0 = (0.5, 0.8)$ $z_0 = (1, 5, 3, 3)$	[0,1,0,1]
11	$x_0 = (14, 22, 37, 20)$ $\lambda_0 = (10, 20)$ $z_0 = (10, 20, 40, 22)$	[0,0,1,1]
12	$x_0 = (3.3, 2, 2.9, 4.1)$ $\lambda_0 = (2.2, 2.2)$ $z_0 = (3.5, 1.9, 1.8, 3.7)$	[1,0,0,0]
13	$x_0 = (2300, 2717, 3520, 1861)$ $\lambda_0 = (3274, 1000)$ $z_0 = (1718, 3251, 3111, 1223)$	[0,1,0,0]
14	$x_0 = (0.42, 0.57, 0.88, 1.2)$ $\lambda_0 = (1.5, 3)$ $z_0 = (0.4, 0.52, 1, 1.18)$	[0,0,1,0]
15	$x_0 = (32, 37, 35, 25)$ $\lambda_0 = (27, 11)$ $z_0 = (15, 18, 21, 31)$	[0,0,0,1]
16	$x_0 = (3000, 4215, 3817, 2914)$ $\lambda_0 = (501, 502)$ $z_0 = (1500, 1718, 1936, 2100)$	[0,0,0,0]

Table 3.2: Starting information for Algorithm 3.18.

In Table 3.3 and Table 3.4 we show the results obtained through the application of Algorithm 3.18 with Newton's and Broyden's methods, respectively. In these tables, "its(o,s)" represent the number of iterations, followed by a specification on how many iterations were ordinary and how many were special. The column "t" indicates the time used by the experiment in seconds. The number of times the new iterate w_{k+1} generated by the test was in a different set $\mathcal{D}_j$ than the previous iterate w_k is denoted by "Nº c". The column "Final $\mathcal{D}_j$" shows the set $\mathcal{D}_j$ which contains the last iterate obtained through the experiment. The symbol "$g(w_f)$" represents the image of g applied in the last iterate obtained. In this same sense, "$\|\nabla g(w_f)\|_\infty$" is the norm $\|\cdot\|_\infty$ applied in gradient vector of g in relation to the final iterate obtained. Finally, the column "Reason" contains the reason for stopping the algorithm, where "B" means that the algorithm breaks down and "C" means that occurs the convergence to the solution w^* of the nonsmooth equations related to problem (3.16).

Test Nº	its (o,s)	t	Nº c	Final $\mathcal{D}_j$	$g(w_f)$	$\|\nabla g(w_f)\|_\infty$	Reason
1	20 (0,20)	0.37	0	[1,1,1,1]	12.54	8.41e-9	B
2	22 (0,22)	0.31	1	[0,0,0,0]	0.15	1.21e-10	B
3	2 (1,1)	0.21	1	[0,1,0,1]	0	0	C
4	25 (1,24)	0.39	4	[0,0,0,1]	0.12	8.41e-10	B
5	20 (0,20)	0.26	0	[0,1,1,1]	6.4	1.02e-9	B
6	21 (1,20)	0.29	1	[1,1,0,1]	0.66	2.5e-10	B
7	24 (2,22)	0.32	3	[0,0,0,1]	0.12	7.49e-9	B
8	21 (1,20)	0.26	1	[0,1,0,0]	0.08	1.77e-15	B
9	23 (1,22)	0.28	2	[1,1,0,1]	0.66	2.5e-10	B
10	1 (1,0)	0.01	0	[0,1,0,1]	9.86e-32	4.44e-16	C
11	21 (1,20)	0.35	1	[0,0,0,1]	0.12	7.49e-9	B
12	23 (0,23)	0.27	2	[0,0,0,1]	0.12	7.5e-9	B
13	23 (1,22)	0.4	2	[0,1,0,0]	0.08	1.77e-15	B
14	22 (0,22)	0.26	1	[0,0,0,0]	0.15	9.73e-9	B
15	20 (0,20)	0.25	0	[0,0,0,1]	0.12	2.09e-10	B
16	23 (1,22)	0.4	2	[0,1,0,0]	0.08	1.77e-15	B

Table 3.3: Results of Algorithm 3.18 with Newton's method.

Test Nº	its (o,s)	t	Nº c	Final $\mathcal{D}_j$	$g(w_f)$	$\|\nabla g(w_f)\|_\infty$	Reason
1	21 (1,20)	0.36	0	[1,1,1,1]	12.54	5.37e-9	B
2	23 (1,22)	0.35	1	[0,0,0,0]	0.15	1.21e-10	B
3	3 (2,1)	0.03	1	[0,1,0,1]	0	0	C
4	24 (2,22)	0.72	1	[1,1,1,0]	12.5	1.47e-9	B
5	21 (1,20)	0.52	0	[0,1,1,1]	6.4	1.02e-9	B
6	23 (1,22)	0.29	1	[1,1,0,1]	0.66	2.5e-10	B
7	26 (3,23)	0.37	3	[0,0,0,1]	0.12	7.49e-9	B
8	31 (8,23)	0.43	9	[0,1,0,0]	0.08	2.66e-15	B
9	22 (2,20)	0.27	1	[1,1,1,1]	12.54	1.22e-9	B
10	2 (1,1)	0.02	0	[0,1,0,1]	1.72e-17	1.79e-9	C
11	23 (1,22)	0.29	1	[0,0,0,1]	0.12	7.49e-9	B
12	24 (1,23)	0.29	2	[0,0,0,1]	0.12	7.5e-9	B
13	21 (1,20)	0.26	1	[1,1,1,1]	12.54	3.5e-9	B
14	23 (1,22)	0.31	1	[0,0,0,0]	0.15	9.73e-9	B
15	21 (1,20)	0.26	0	[0,0,0,1]	0.12	2.09e-10	B
16	25 (3,22)	0.3	2	[0,1,0,0]	0.08	8.32e-17	B

Table 3.4: Results of Algorithm 3.18 with Broyden's method.

Through the above tables, basically we observe that occurs a small difference between the results obtained from Algorithm 3.18 with Newton's method and with Broyden's method, and the convergence occurs at exactly the same 2 starting points for both algorithms. Doing a little deeper analysis, it is possible to perceive a small advantage for the algorithm with Newton's method in the sense of number of iterations required to obtain the result, since in all tests except 4, 9 and 13 the number of iterations used by this algorithm was smaller then the number of iterations used by the algorithm with Broyden's method. This is expected, since the use of the true Jacobian is more expensive than low rank updates.

We observe that in all experiments the algorithm converges to a stationary point of function g. Actually, the usual behavior of both algorithms in numerical experiments is to, at the beginning of the test, generate iterates w_k that are not all contained in the same set $\mathcal{D}_j$. Then, one iterate w_k falls in a certain set $\mathcal{D}_i$, and then only promotes special iterations on it, generating iterates contained in this set $\mathcal{D}_i$ until it finds the stationary point of this set, which makes the algorithm breaks down. However, there are tests where all the iterates generated by the algorithm are contained in the same set $\mathcal{D}_j$, and then it converges to a local minimum on this set. This occurs in tests 1, 5, 10 and 15 for both algorithms. This fact indicates that if some point is favorable for the not occurrence of

a change in the set $\mathcal{D}_j$ which contains the iterates, the local method used in Algorithm 3.18 will not be effective in the sense of modify this characteristic.

The next results can be useful in order to guarantee situations where the Algorithm 3.18 will promote at least one change in the sets $\mathcal{D}_j$ which contains the iterates throughout its operation. The main idea is that stationarity can not occur in sets $\mathcal{D}_j$ where J_j is nonsingular, unless we have the solution of the original problem.

Lemma 3.21. *Suppose that $J(w^*)$ is nonsingular, $\mathcal{D}_j$ is a set such that its respective Jacobian matrix J_j is nonsingular, and $w^* \notin \mathcal{D}_j$. There is no $w \in \mathcal{D}_j$ such that $\nabla g(w) = 0$. In particular, this set have no local minimums of g.*

Proof. Suppose that there is $w \in \mathcal{D}_j$ such that $\nabla g(w) = 0$. By the definition of g,

$$\nabla g(w) = F(w)^T J(w) = 0,$$

which implies

$$J(w)^T F(w) = 0. \tag{3.28}$$

Since $w \in \mathcal{D}_j$, $J(w) = J_j$. By hypothesis, J_j is nonsingular, which implies J_j^T nonsingular, and then $J(w)^T$ also is. From this, through (3.28) we have $F(w) = 0$. Therefore, $w = w^*$, which contradicts the fact that $w^* \notin \mathcal{D}_j$. □

Lemma 3.22. *Suppose that $J(w^*)$ is nonsingular, $\mathcal{D}_j$ is a set such that its respective Jacobian matrix J_j is nonsingular, and $w^* \notin \mathcal{D}_j$. If $w_k \in \mathcal{D}_j$ and w_{k+1} is generated through Newton's iteration, then $w_{k+1} \notin \mathcal{D}_j$.*

Proof. By Newton's method,

$$w_{k+1} = w_k - J(w_k)^{-1} F(w_k). \tag{3.29}$$

Since $w_k \in \mathcal{D}_j$, $J(w_k) = J_j$. We have already seen in the end of Section 3.1 of Chapter 3 that, being $F_j : \mathbb{R}^N \to \mathbb{R}^N$ a function defined as in (3.14), F and F_j coincide in set $\mathcal{D}_j$. Therefore, from (3.29),

$$w_{k+1} = w_k - J_j^{-1} F_j(w_k) = w_k - J_j^{-1} \left(J_j w_k + \begin{bmatrix} -c \\ -b \\ 0 \end{bmatrix} \right),$$

which implies that

$$w_{k+1} + J_j^{-1} \begin{bmatrix} -c \\ -b \\ 0 \end{bmatrix} = 0,$$

and then

$$F_j(w_{k+1}) = J_j w_{k+1} + \begin{bmatrix} -c \\ -b \\ 0 \end{bmatrix} = 0.$$

If occurs $w_{k+1} \in \mathcal{D}_j$, then $F(w_{k+1}) = F_j(w_{k+1}) = 0$, which implies $w_{k+1} = w^*$ and then $w^* \in \mathcal{D}_j$, which is a contradiction. Therefore, $w_{k+1} \notin \mathcal{D}_j$, as we wanted. □

Since all the numerical experiments realized converges to a stationary point of g, by Lemma 3.21 occurs that, if the current iterate w_k of the Algorithm 3.18 is in a set $\mathcal{D}_j$ which its respective Jacobian matrix J_j is nonsingular and $w^* \notin \mathcal{D}_j$, then, in the sequence of its operation, the algorithm will make at least one change in the sets $\mathcal{D}_j$ which contains its iterates. On the other hand, the application of Lemma 3.22 in this sense is clear: if, in some iteration of the algorithm, we have w_k in a set $\mathcal{D}_j$ which satisfy the hypotheses of this lemma and the next iterate w_{k+1} is obtained through a Newton's iteration, then will occurs more one change in the sets $\mathcal{D}_j$ which contains its iterates.

The convergence occurs in tests 3 and 10. In test number 10, convergence was expected, mainly in Algorithm 3.18 with Newton's method, since in this experiment the starting point w_0 is in the same set $\mathcal{D}_j$ than the solution w^*. However, it is not clear the reason of convergence in test number 3. It was made an extra test, for both algorithms, with the starting point $w_0 = (x_0, \lambda_0, z_0)$ where

$$x_0 = (1000, 1000, 2000, 1000),\ \lambda_0 = (1500, 1000)$$

and

$$z_0 = (2000, 2000, 1000, 2000),$$

which is a point in the same set $\mathcal{D}_j$ than the one considered in test 3, but with highest values in its coordinates. In this case, the test of Algorithm 3.18 with Newton's method used 23 iterations, while the one of Algorithm 3.18 with Broyden's method used 21 iterations. Both algorithms break down in a stationary point and do not converge to the solution. The experiment shows that convergence does not occur by some characteristic associated with the set $\mathcal{D}_j$ that contains the starting point w_0.

As we commented before, the pure Broyden's method with $B_0 = I_N$ have a good behavior to find the solution w^* of problem (3.6), at least when considering problem (3.16), mainly when w_0 is near the solution. In Algorithm 3.18 with Broyden's method we used $B_0 = I_N$ and tested some starting points near the solution of the problem, in tests 2, 3, 7, 10, 12 and 14. In these tests, the convergence to the solution did not occur. This is possible because, as commented in [6], quasi-Newton methods usually have a good behavior to find the solution when the iterates is near to it, but this methods are not characterized by a monotone decrease of the merit function g throughout the iterations.

Therefore, when Algorithm 3.18 with Broyden's method is used, it is very possible that inequality (3.27) does not occur at some iteration, which implies that a special iteration is performed. That can make the algorithm lose the good performance that Broyden's method usually have in this situation.

By considering that Algorithm 3.18 with Newton's and Broyden's methods, in general do not have a good performance to solve problem (3.16), which is not a problem relationed with a high dimensional objective function, it is reasonable to believe that both algorithms will not behave well when applied to large scale problems.

Algorithm 3.18 with Newton's method and with Broyden's method generate together 16 different final points which are not equal to the solution w^*. One interesting characteristic of these points is that, in general, they are close to the solution and their image by g is small. Therefore, by considering the fact that the pure Broyden's method with $B_0 = I_N$ had good performance in order to solve equation (3.6) for problem (3.16) near the solution w^*, this method was applied in each one of these 16 final points. In 8 of this 16 tests the method finds the solution w^*, always performing around 40 iterations, which took less than 1 second.

By considering the numerical experiments of Algorithm 3.18, we propose a new algorithm. The algorithm first apply Algorithm 3.18 with Newton's or Broyden's method using $B_0 = I_N$. Then, if Algorithm 3.18 breaks down, we use the final point as the starting one for a pure version of Broyden's method with $B_0 = I_N$ to solve problem (3.6). This algorithm is described in Algorithm 3.23.

Algorithm 3.23.

Consider $FLAG = 1$, $m_0 = 1$, $\alpha_0 = 1$ *and given* $\sigma \in (0,1)$, $\eta \in (0, \frac{1}{2}]$, $a \in (0,1)$, M *such that* $\Delta_0 = M$, $q \in \mathbb{N} \cup \{0\}$, $\gamma \in (0,1)$, w_0 *and* B_0 *(if the local method chosen is Broyden's method);*

for $k = 0, 1, 2, ...$

Step 1: *If* $FLAG = 1$, *obtain* w_{k+1} *through an ordinary iteration. Else, obtain* w_{k+1} *using a special iteration;*

Step 2: *If the algorithm breaks down at Step 1, apply Broyden's method with* $B_0 = I_N$ *at the last iterate* w_k *obtained and then stop;*

Step 3: *If*

$$g(w_{k+1}) \leq \gamma g(\pi_{k-q})$$

set $FLAG \leftarrow 1$. *Else, re-define* $w_{k+1} \leftarrow \pi_{k+1}$ *and* $FLAG \leftarrow -1$;

end(for)

With the possible execution of Broyden's method in the final iterations, Algorithm 3.23 will not necessarily converges or breaks down. Now, there are two more possibilities:

it runs until it reaches the maximum of 10000 iterations allowed, or it stops because it has found a singular matrix B_k in Broyden's method. Tables 3.5 and 3.6 contain the results obtained by applying Algorithm 3.23 using Newton's and Broyden's methods, respectively, in the initial points showed in Table 3.2. In it, "S" means that the experiment stopped because Broyden's method found a singular B_k matrix, while "IT" means that it reached the limit of 10000 iterations allowed.

Test Nº	its (o,s)	t	Nº c	Final $\mathcal{D}_j$	$g(w_f)$	$\|\nabla g(w_f)\|_\infty$	Reason
1	10000 (0,20)	78.93	52	[1,1,1,1]	17.98	3.53	IT
2	52 (0,22)	0.55	11	[0,1,0,1]	1.59e-19	5.62e-10	C
3	2 (1,1)	0.05	1	[0,1,0,1]	0	0	C
4	82 (1,24)	0.9	28	[0,1,0,1]	1.33e-20	1.32e-10	C
5	24 (0,20)	0.75	1	[1,1,1,1]	7.28e17	1.49e9	S
6	52 (1,20)	0.57	9	[0,1,0,1]	2.39e-18	1.49e-9	C
7	85 (2,22)	0.91	30	[0,1,0,1]	6.37e-19	8.53e-10	C
8	1911 (1,20)	15.31	92	[1,1,1,1]	4.26e7	8706	S
9	54 (1,22)	0.58	10	[0,1,0,1]	2.39e-18	1.49e-9	C
10	1 (1,0)	0.05	0	[0,1,0,1]	9.86e-32	4.44e-16	C
11	82 (1,20)	0.74	28	[0,1,0,1]	6.37e-19	8.53e-10	C
12	64 (0,23)	0.63	17	[0,1,0,1]	2.36e-20	1.15e-10	C
13	1913 (1,22)	14.93	93	[1,1,1,1]	4.26e7	8706	S
14	76 (0,22)	0.8	33	[0,1,0,1]	4.36e-18	2.49e-9	C
15	59 (0,20)	0.69	20	[0,1,0,1]	5e-18	4.13e-9	C
16	1913 (1,22)	15.03	93	[1,1,1,1]	4.26e7	8706	S

Table 3.5: Results of Algorithm 3.23 with Newton's method.

Test Nº	its (o,s)	t	Nº c	Final $\mathcal{D}_j$	$g(w_f)$	$\|\nabla g(w_f)\|_\infty$	Reason
1	616 (1,20)	4.42	6	[1,1,1,1]	18.77	4.07	S
2	53 (1,22)	0.45	11	[0,1,0,1]	1.59e-19	5.62e-10	C
3	3 (2,1)	0.02	1	[0,1,0,1]	0	0	C
4	10000 (2,22)	67.64	121	[1,1,1,1]	523283.86	1023	IT
5	25 (1,20)	6.01	1	[1,1,1,1]	7.28e17	1.49e9	S
6	54 (1,22)	0.43	9	[0,1,0,1]	2.39e-18	1.49e-9	C
7	72 (3,23)	0.57	23	[0,1,0,1]	4.32e-17	7.25e-9	C
8	78 (8,23)	0.61	31	[0,1,0,1]	4.33e-19	7.65e-10	C
9	10000 (2,20)	69.13	71	[1,1,1,1]	524305.16	1025	IT
10	2 (1,1)	0.04	0	[0,1,0,1]	1.72e-17	1.79e-9	C
11	84 (1,22)	0.63	28	[0,1,0,1]	6.13e-19	8.27e-10	C
12	65 (1,23)	0.54	17	[0,1,0,1]	2.36e-20	1.15e-10	C
13	10000 (1,20)	67.99	31	[1,1,1,1]	17.76	3.48	IT
14	77 (1,22)	0.58	33	[0,1,0,1]	4.36e-18	2.49e-9	C
15	60 (1,20)	0.45	20	[0,1,0,1]	5e-18	4.13e-9	C
16	999 (3,22)	6.21	77	[1,1,1,1]	48808.45	256	S

Table 3.6: Results of Algorithm 3.23 with Broyden's method.

Through the analysis of these results, it is visible that we no longer have the guarantee of convergence to the stationary point, since the last iterations of this algorithm are usually done by the Broyden's method. However, Algorithm 3.23 converges in 11 of 16 numerical experiments with Newton's method, and in 10 of 16 tests with Broyden's method, always in less than 1 second, which indicates a very good performance of this algorithm. Therefore, by considering this results, it is reasonable to believe that, if the linear programming problem considered has the characteristic of having a good part of the stationary points of merit function g close to the solution w^* of equation (3.6), as problem (3.16) have, then Algorithm 3.23 has good performance in relation to the global convergence.

Conclusion

This work was devoted to the study of Newton's and Broyden's methods for solving linear programming problems. Under some strong hypotheses, it was obtained a result that guarantees the linear local convergence of the IPM with quasi-Newton approach [8].

With the goal of further understanding the behavior of Newton's and Broyden's methods in linear programming problems, a nonsmooth version of the KKT conditions was considered. It was used nonsmooth versions of such methods and interesting local convergence results were obtained. In particular, it was proved one-step convergence of Newton's and Broyden's methods, the later with $B_0 = J(w_0)$. Moreover, through computational experiments using Julia language, it was possible to find counter-examples for results about some types of convergence for these methods.

Seeking to obtain global convergence results about the application of Newton's and Broyden's methods on nonsmooth equations, the algorithm [6] was studied. General convergence results were presented. After numerical experiments, a study about its performance was possible, and then a modification of this algorithm was proposed, seeking to improve its performance regarding global convergence when it is applied to a system of nonsmooth equations relative to linear problem problems. The obtained results were promising, but more tests are necessary.

References

[1] BEZANSON, J.; EDELMAN, A.; KARPINSKI, S.; SHAH, V. B. Julia: A Fresh Approach to Numerical Computing. SIAM Review 59, 65-98 (2017).

[2] CLARKE, F. H. Optimization and Nonsmooth Analysis. John Wiley & Sons, Inc., New York, 1983.

[3] DENNIS JR., J. E.; SCHNABEL, R. B. Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Society for Industrial and Applied Mathematics, 1996.

[4] DIKIN, I. I. Iterative solution of problems of linear and quadratic programming. Doklady Akademii Nauk 174, 747-748 (1967).

[5] FACCHINEI, F.; FISCHER, A.; HERRICH M. An LP-Newton method: nonsmooth equations, KKT systems, and nonisolated solutions. Mathematical Programming 146, 1-36 (2014).

[6] GOMES-RUGGIERO, M. A.; MARTÍNEZ, J. M.; SANTOS, S. A. Solving Nonsmooth Equations by Means of Quasi-Newton Methods with Globalization. Recent Advances in Nonsmooth Optimization, 121-140 (1995).

[7] GONDZIO, J. Interior point methods 25 years later. European Journal of Operational Research 218, 587-601 (2012).

[8] GONDZIO, J.; SOBRAL, F. N. C. Quasi-Newton approaches to interior point methods for quadratic problems. Computational Optimization and Applications 74, 93-120 (2019).

[9] ITO, K.; KUNISCH, K. On a semi-smooth Newton method and its globalization. Mathematical Programming 118, 347-370 (2009).

[10] KARMARKAR, N. K. A new polynomial-time algorithm for linear programming. Combinatorica 4, 373-395 (1984).

[11] KHACHIYAN, L. G. A polynomial algorithm in linear programming. Soviet Mathematics Doklady 20, 191-194 (1979).

[12] KLEE, V.; MINTY, G. How good is the simplex algorithm. Inequalities 3, 159-175 (1972).

[13] LESAJA, G. Introducing Interior-Point Methods for Introductory Operations Research Courses and/or Linear Programming Courses. The Open Operational Research Journal 3, 1-12 (2009).

[14] LIMA, E. L. Análise Real, Vol. 2 - Funções de n Variáveis. 1st ed. IMPA, Rio de Janeiro, 2004.

[15] MARTÍNEZ, J. M.; SANTOS, S. A. Métodos de Otimização Computacional. Publicações do 20o Colóquio Brasileiro de Matemática, 1995.

[16] MOGENSEN, P. K.; RISETH, A. N. Optim: A mathematical optimization package for Julia. Journal of Open Source Software 24 (2018).

[17] NOCEDAL, J.; WRIGHT, S. J. Numerical Optimization. 2nd ed. Springer, New York, 2006.

[18] PANG, J. S. Newton's method for B-differentiable equations. Mathematics of Operations Research 15, 311-341 (1990).

[19] QI, L. Convergence Analysis of Some Algorithms for Solving Nonsmooth Equations. Mathematics of Operations Research 18, 227-244 (1993).

[20] QI, L.; JIANG, H. Semismooth Karush-Kuhn-Tucker Equations and Convergence Analysis of Newton and Quasi-Newton Methods for Solving These Equations. Mathematics of Operations Research 22, 301-325 (1997).

[21] VANDERBEI, R. J. Linear Programming: Foundations and Extensions. 4th ed. Springer, 2014.

[22] WRIGHT, S. J. Primal-Dual Interior-Point Methods. Society for Industrial and Applied Mathematics, Philadelphia, 1997.