LINEAR AND SEMIDEFINITE PROGRAMMING USING A TECHNIQUE FROM INTERIOR POINTS AND FEASIBLE DIRECTIONS

Angélica Miluzca Victorio Celis

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Nelson Maculan Filho
Grigori Chapiro

Rio de Janeiro
Novembro de 2022

LINEAR AND SEMIDEFINITE PROGRAMMING USING A TECHNIQUE FROM INTERIOR POINTS AND FEASIBLE DIRECTIONS

Angélica Miluzca Victorio Celis

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Nelson Maculan Filho, D.Sc.

Prof. Grigori Chapiro, D.Sc.

Prof. Luidi Gelabert Simonetti, D.Sc.

Prof. Laura Sílvia Bahiense da Silva Leite, D.Sc.

Prof. Luiz Mariano Paes de Carvalho Filho, D.Sc.

Prof. Luiz Satoru Ochi, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
NOVEMBRO DE 2022

Celis, Angélica Miluzca Victorio

Linear and semidefinite programming using a technique from interior points and feasible directions/Angélica Miluzca Victorio Celis. – Rio de Janeiro: UFRJ/COPPE, 2022.

X, 58 p. 29, 7cm.

Orientadores: Nelson Maculan Filho
Grigori Chapiro

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2022.

Referências Bibliográficas: p. 54 – 58.

1. Otimização Linear. 2. Algoritmo de Direções Viável. 3. Algoritmo de Ponto Interior. I. , Nelson Maculan Filho *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Dedicado a meus pais Aurelia e Juan, e à minha filha Avril.

Agradecimentos

Quero muito destacar e agradecer:

Ao Professor José Herskovits (in memoriam), ter um orientador como o senhor foi um privilégio que a vida me ofertou. Foram anos de orientações, conversas, reuniões presenciais e virtuais para concluirmos este trabalho.

Ao Professor Nelson Maculan, pelas orientações, direções, paciência e sobre tudo pela confiança depositada.

Ao Professor Grigori Chapiro por aceitar ser meu orientador, por seus comentários e por sua gentileza em colaborar na realização deste trabalho.

Aos professores membros da banca por aceitaram o convite.

Aos meus pais, Aurelia e Juan, por todo o carinho que, mesmo à distância, me ajudou a prosseguir.

A meu noivo Paul por todo o carinho, por seu companheirismo e por compartilhar seu tempo em especial por estar distante de minha terra natal, mas que pude encontrar o conforto em meus sentimentos.

A meus colegas do PPGI, PESC e IM, com os quais dividi almoços, ideias, angústias, disciplinas e tantas outras coisas que tornaram a vida acadêmica possível.

Ao PESC/COPPE e PEM/COPPE pelos profissionais que contribuíram em todos os processos administrativos, os quais mesmo em situações de desconforto por tantas outras demandas, jamais deixavam de oferecer o melhor aos alunos e seus pesquisadores.

À Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e COPPETEC pelo apoio financeiro, e ao povo do Brasil por sua hospitalidade.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

LINEAR AND SEMIDEFINITE PROGRAMMING USING A TECHNIQUE FROM INTERIOR POINTS AND FEASIBLE DIRECTIONS

Angélica Miluzca Victorio Celis

Novembro/2022

Orientadores: Nelson Maculan Filho
Grigori Chapiro

Programa: Engenharia de Sistemas e Computação

Propomos uma adaptação do Algoritmo de Pontos Interiores de Direção Viável (FDIPA) de J. Herskovits, para a resolução de programas lineares de grande porte. Este algoritmo requer um ponto inicial no interior de uma região viável definida pelas restrições e gera uma sequência de pontos no interior desta região que converge para a solução do problema. A cada iteração, a solução de dois sistemas lineares com a mesma matriz de coeficientes é determinada, o que envolve um esforço computacional significativo. Reduzir o tempo de solução de sistemas lineares é, portanto, uma forma de melhorar o desempenho do método. Os sistemas lineares a serem resolvidos estão associados a matrizes simétricas definidas positivas. Portanto, usamos o método do Gradiente Conjugado Pré-condicionado *Split* (SPCG) para resolvê-los, juntamente com um pré-condicionador Cholesky incompleto usando a função ICHOL do Matlab. Também propomos usar a primeira iteração do gradiente conjugado e pré-solucionador antes de aplicar o algoritmo, a fim de reduzir o custo computacional. A seguir, fornecemos demonstrações matemáticas de que as iterações se aproximam aos pontos de Karush-Kuhn-Tucker do problema sob suposições razoáveis. Finalmente, evidências numéricas mostram que o método não só funciona na teoria, mas também é competitivo com métodos mais avançados.

Consideramos também a extensão de FDIPA, para programação linear sujeitos às restrições nas quais matrizes $m \times m$ devem ser semidefinidas.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

LINEAR AND SEMIDEFINITE PROGRAMMING USING A TECHNIQUE FROM INTERIOR POINTS AND FEASIBLE DIRECTIONS

Angélica Miluzca Victorio Celis

November/2022

Advisors: Nelson Maculan Filho
Grigori Chapiro

Department: Systems Engineering and Computer Science

We propose an adaptation of the Feasible Direction Interior Points Algorithm (FDIPA) by J. Herskovits, for solving large-scale programs. This algorithm requires a starting point inside a feasible region defined by the constraints and generates a sequence of points inside this region that converges to the solution of the problem. At each iteration, the solution of two linear systems with the same coefficient matrix is determined. This step involves a significant computational effort. Reducing the solution time of linear systems is, therefore, a way to improve the performance of the method. The linear systems to be solved are associated with definite positive symmetric matrices. Therefore, we use Split Preconditioned Conjugate Gradient(SPCG) method to solve them, together with an Incomplete Cholesky preconditioner using Matlab's ICHOL function. We also propose to use the first iteration of the conjugate gradient, and to presolve before applying the algorithm, in order to reduce the computational cost. Following, we then provide mathematica proof that show that the iterations approach Karush-Kuhn-Tucker points of the problem under reasonable assumptions. Finally, numerical evidence show that the method not only works in theory but is also competitive with more advanced methods.

We also consider the extension of FDIPA, for linear programming subject to constraints in which matrices $m \times m$ must be semi-defined.

Sumário

Lista de Tabelas

Capítulo 1

Introdução

Neste trabalho, estamos interessados em resolver o problema de Programação Linear (LP), isto é, encontrar um mínimo de uma função objetivo linear sujeito a um conjunto de restrições lineares de igualdade e desigualdade, todas definidas em termos de alguma variável de decisão. A formulação primal do problema é

$$\begin{array}{ll} \underset{x}{\text{maximização}} & c^T x \\ \text{sujeito a} & Ax = b, \\ & x \geq 0, \end{array} \tag{1.1}$$

onde $A \in \mathbb{R}^{m \times n}$, $n \geq m$. A é a matriz de restrições do problema com $posto(A) = m$, $b \in \mathbb{R}^m$ é o vetor de termos independentes, $c \in \mathbb{R}^n$ é o vetor de custo, $x \in \mathbb{R}^n$ é o vetor de variáveis. Todos os pontos que satisfazem as restrições formam o conjunto de soluções viáveis.

O problema linear primal (1.1) possui outro problema linear associado chamado dual, da seguinte maneira:

$$\begin{array}{ll} \underset{\lambda}{\text{minimização}} & b^T \lambda \\ \text{sujeito a} & A^T \lambda \leq c. \end{array} \tag{1.2}$$

A teoria de dualidade relaciona os dois problemas. Se x^* e λ^* são soluções de (1.1) e (1.2) respectivamente, então $c^T x^* = b^T \lambda^*$.

A programação linear hoje é amplamente utilizada na tomada de decisões em finanças, economia, engenharia. Além disso, é utilizada como uma ferramenta para resolver problemas de programação inteira e não linear.

Existem dois tipos de métodos para resolver problemas de LP: o método Simplex

e o método de ponto interior. O primeiro foi proposto por Dantzig [13], [14], segue um caminho ao longo das bordas do conjunto de soluções viáveis, logo o número de iterações necessárias pelo método Simplex pode crescer exponencialmente na dimensão do problema [34], [13], [15], [38]. Mais tarde, para evitar a complexidade exponencial no pior caso, vários pesquisadores como Von Neumann [45], e Hoffman [32] propuseram algoritmos que caminham no interior da região viável, mas na prática eles viram que não era competitivo com o método Simplex por causa da possibilidade de instabilidade numérica nos cálculos e das etapas computacionais caras que eles exigem.

O segundo método surge com o algoritmo de Karmarkar [33], que parte de um ponto interior estrito e aplica transformações projetivas repetidas em uma esfera inscrita para criar uma sequência de pontos que converge para a solução ótima com complexidade polinomial. A partir do algoritmo de Karmarkar foram gerados diferentes tipos de algoritmos de pontos interiores, dependendo do tipo de transformadas e direções de busca a serem utilizadas. Por exemplo, o método projetivo, que foi estudado por Anstreicher [3, 4], Lustig [37] e Gonzaga [23], o método de escala afim estudado por Dikin [17], Barnes [8], Cavalier & Soyster [10], o método de *path-following* por Gonzaga [21], [22] e Monteiro & Adler [42, 43], e método de redução potencial Ye [55]. Para uma melhor compreensão da teoria dos pontos interiores, sugere-se a leitura dos livros [54] e [48].

Esses métodos de pontos interiores seguem um caminho através do interior do conjunto de soluções viáveis e convergem para a solução ótima em tempo polinomial. Na prática, o número de iterações para problemas de grande porte é muito menor que o do método Simplex, porém, cada um desses passos é mais complexo e trabalhoso para determinar a solução de sistemas lineares.

Neste trabalho, o Algoritmo FDIPA para problemas de programação não linear proposto por Herskovits [31] será adaptado para resolver problemas de programação linear de larga escala (FDIPA-LP) de forma eficiente. A partir de um ponto inicial estritamente viável, FDIPA executa uma iteração do tipo Newton para resolver a condição de otimalidade de Karush-Kuhn-Tucker (KKT). Durante este procedimento é obtida uma direção de descida para a função objetivo. Uma pertubação é introduzida na condição de complementaridade para que uma direção de descida viável seja obtida. Esse algoritmo garante a viabilidade primal e a atualização garante da viabilidade dual.

FDIPA já foi adaptado para vários tipos de problemas, por exemplo, [25], [49],

[6], [30], [28], [9] com diversas aplicações, especialmente em engenharia mecânica, aerodinâmica e eletromagnetismo; provando ser um código fácil de implementar e eficiente.

FDIPA possui uma primeira adaptação para resolver LP em Tits [52], onde se assemelha ao método de escala afim, demostra a convergência usando como direção de busca (d_α), a direção obtida ao resolver o primeiro sistema do FDIPA (ou seja, é um FDIPA sem perturbação), com um passo que garante a viabilidade estrita e o hessiano exato. Logo, em Celis [12] aplica FDIPA sem perturbação com hessiano aproximado a zero para manter as propriedades do algoritmo. Depois, em [11] usa os dois sistemas do FDIPA no qual o segundo sistema contém uma perturbação evitando ficar perto da fronteira.

Neste trabalho vamos detalhar FDIPA-LP estudado em [11]. FDIPA-LP resolve dois sistemas lineares, o que implica um esforço computacional significativo. Para reduzir o tamanho desses sistemas, antes de iniciar o algoritmo, usaremos um pré-solucionador [47] e técnicas de escalonamento. Também usaremos o método de gradiente conjugado pré-condicionado *Split* (SPCG) para resolvê-los, juntamente com um pré-condicionador Cholesky incompleto usando a função ICHOL do Matlab. Observou-se que as iterações geradas pelo SPCG são candidatas a direção de busca. Portanto, usamos apenas a primeira iteração do SPCG.

O algoritmo é implementado em MATLAB, com o objetivo de realizar testes numéricos de grande porte encontrados na literatura, como por exemplo, na biblioteca NETLIB [19].

É fácil ver que problemas de programação linear é uma instância especial de problemas de programação semidefinida linear, onde as inequações vetoriais são substituídas por inequações matriciais. Muitos elementos da teoria de algoritmos de pontos interiores para LP são estendidos de uma forma bastante natural a programação semidefinida (SDP). Tais extensões foram inicialmente sugeridas por Alizadeh [1], Monteiro [41], e Nesterov e Nemirovsltii [44]. [1] utiliza um algoritmo projetivo de redução potencial, enquanto a teoria geral de [44] permite a análise de uma variedade de algoritmos de redução potencial e de trajetória central. A convergência polinomial dos algoritmos para resolver SDP foi estudada e comprovada por vários pesquisadores [26], [27], [40], [46]. O grande interesse recente em problemas SDP está ligado a aplicações não-triviais de otimização combinatória, teoria de controle, otimização estrutural e outras áreas. Veja [53] e [2] para um excelente apanhado sobre a teoria e as aplicações de SDP.

Neste trabalho, estamos interessados em uma extensão FDIPA-LP [11] à programação semidefinida linear com o objetivo de obter um algoritmo mais eficiente.

1.1 Organização do Trabalho

Este trabalho está organizado da seguinte forma:

No Capítulo 2, apresentaremos algumas definições básicas sobre programação linear e FDIPA, necessários para a formulação do problema em estudo, em particular, as condições de otimalidade.

No Capítulo 3, fazemos a descrição do algoritmo de pontos interiores e direções viáveis (FDIPA) para programação linear.

No Capítulo 4, apresentamos detalhes da implementação e os resultados de alguns problemas teste da biblioteca NETLIB, mostrando seu desempenho.

No Capítulo 5, mostramos nossos avanços para generalizar FDIPA linear para a programação semidefinida linear.

Finalmente, no Capítulo 6, apresentamos as considerações finais sobre o trabalho com algumas perspectivas para trabalhos futuros.

Capítulo 2

Preliminares

Neste capítulo abordamos conceitos e resultados essenciais para o desenvolvimento dos capítulos subsequentes deste trabalho.

A continuação nosso resumo da notação usada neste trabalho:

- O conjunto de matrizes simétricas reais $m \times m$ é denotado por S^m.
- O conjunto de matrizes simétricas semidefinidas positivas (ou semidefinidas negativas) é denotado por S^m_+ (ou S^m_-) e o conjunto de matrizes simétricas definidas positivas (ou definidas negativas) é denotado por S^m_{++} (ou S^m_{--}).
- Para as matrizes $A, B \in S^m$, a notação $A \succeq B$ (ou $A \preceq B$) significa que a matriz $A - B$ é semidefinida positiva (ou semidefinida negativa). Da mesma forma, a notação $A \succ B$ (ou $A \prec B$) significa que a matriz $A - B$ é definida positiva (ou definida negativa).
- Seja $A \in \mathbb{R}^{m\times m}$ matriz simétrica, consideramos os mapeamentos $vec : \mathbb{R}^{m\times m} \to \mathbb{R}^{m^2}$ e $mat : \mathbb{R}^{m^2} \to \mathbb{R}^{m\times m}$ definidos como:

  $$vec(A) = [a_{11}, ..., a_{n1}, a_{12}, ..., a_{m2}, ..., a_{1m}, ..., a_{mm}]^T,$$

  e mat a inversa do mapeamento vec.
- O produto interno de duas matrizes $A, B \in S^m$ é denotado por $\langle A, B\rangle$, e definido como

  $$\langle A, B\rangle = Tr(A^T B) = vec(A)^T vec(B).$$

- Denota-se por $A \otimes B$ o produto de Kronecker de duas matrizes $A \in \mathbb{R}^{p\times m}$ e

$B \in \mathbb{R}^{r\times s}$ definido como

$$A \otimes B = [a_{ij}B] = \begin{bmatrix} a_{11}B & \cdots & a_{1m}B \\ \vdots & \ddots & \vdots \\ a_{p1}B & \cdots & a_{pm}B \end{bmatrix}, \ i = 1, ...p, \ \ j = 1, ..., m.$$

- Seja $\mathcal{A} : \mathbb{R}^n \to S^m$ é um mapeamento de $\mathbb{R}^n$ no espaço S^m definida como:

$$\mathcal{A}(x) = A_0 + \sum_{i=1}^{n} A_i x_i,$$

onde $A_i \in S^m$.

- Denota-se por $A_i(x) = \partial\mathcal{A}(x)/\partial x_i \in \mathbb{R}^{m\times m}$ as matrizes das derivadas parciais de $\mathcal{A}$, com $i = 1, ..., n$, definida como:

$$A_i = \frac{\partial \mathcal{A}}{\partial x_i}(x) = \begin{bmatrix} \frac{\partial a_{11}}{\partial x_i}(x) & \ldots & \frac{\partial a_{1m}}{\partial x_i}(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial a_{m1}}{\partial x_i}(x) & \ldots & \frac{\partial a_{mm}}{\partial x_i}(x) \end{bmatrix}.$$

- Denota-se por $\nabla\mathcal{A}(x) \in \mathbb{R}^{n\times m^2}$ a derivada do mapeamento $\mathcal{A}(.)$ em x, isto é, $\nabla\mathcal{A}(x)$ é uma transformação linear de $\mathbb{R}^n$ em $\mathcal{S}^m$ definida por:

$$\nabla\mathcal{A}(x) = \frac{\partial \mathcal{A}}{\partial x}(x) = \begin{bmatrix} \frac{\partial \mathcal{A}}{\partial x_1}(x) \\ \vdots \\ \frac{\partial \mathcal{A}}{\partial x_n}(x) \end{bmatrix},$$

ou seja

$$\nabla\mathcal{A}(x) = \frac{\partial \mathcal{A}}{\partial x}(x) = \begin{bmatrix} vec(A_1(x))^T \\ \vdots \\ vec(A_n(x))^T \end{bmatrix}.$$

- A derivada direcional do mapeamento $\mathcal{A}(.)$ na direção $d \in \mathbb{R}^n$, denotada por $\nabla\mathcal{A}(x)^T d \in \mathbb{R}^{n\times m^2}$, é definida com

$$\nabla\mathcal{A}(x)^T d = vec(D\mathcal{A}(x)d), \tag{2.1}$$

onde,

$$D\mathcal{A}(x)d = \sum_{p=1}^{n} d_p \frac{\partial \mathcal{A}}{\partial x_p}. \tag{2.2}$$

2.1 Conceitos básicos de Programação Linear

As relações entre os problemas (1.1) e (1.2) são dadas pelos teoremas da dualidade fraca, dualidade forte e folgas complementares, exposto abaixo [38].

Teorema 2.1. *(Dualidade Fraca) Se x é uma solução factível para o problema primal* (1.1)*, e λ é uma solução factível para o problema dual* (1.2)*, então*

$$c^T x \geq b^T \lambda.$$

Teorema 2.2. *(Dualidade Forte) Se o problema primal* (1.1) *possui uma solução ótima x^* então seu dual* (1.2) *também possui uma solução ótima λ^* e os valores ótimos dos dois problemas são iguais, ou seja,*

$$c^T x^* = b^T \lambda^*.$$

Teorema 2.3. *(Folgas complementares) Seja x^* solução ótima do problema primal* (1.1)*, e λ^* solução ótima do problema dual* (1.2) *se, e somente se,*

$$x_i^*(A^T \lambda^* - c)_i = 0 \text{ para todo } i = 1, ..., n.$$

A distância entre a função objetivo primal e dual será chamada de *gap* dual $c^T x^* - b^T \lambda^*$. No ótimo de ambos problemas, o *gap* dual é 0.

O método Simplex prevê qual x_i é zero, realiza partições em variáveis básicas $x_B > 0$ e não básicas $x_N = 0$, ou seja vai testando combinações diferentes. Se alguma variável básica é zero, ou alguma restrição é paralela na função objetivo, estaríamos enfrentando um problema degenerado, como consequência de ter múltiplas soluções ou ciclagens fazendo muitas iterações.

Um dos resultados da teoria de dualidade é a adaptação das condições de KKT para os dois problemas Primal (1.1) e Dual (1.2).

Teorema 2.4. *(Condições de KKT para o Primal e Dual) (x, y, z) são soluções ótimas para o par primal-dual se, e somente se, as seguintes condições são satisfeitas*

$$\begin{aligned} Ax &= b, && (2.3)\\ A^T y + z &= c, && (2.4)\\ x_i z_i &= 0 \quad i = 1, ..., n, && (2.5)\\ (x, z) &\geq 0. && (2.6) \end{aligned}$$

Demonstração. Em [54]. □

O teorema acima e o método de Newton [38] são a sustentação do método de pontos interiores primal-dual para programação linear.

2.1.1 Método de Pontos Interiores

Estes métodos de pontos interiores seguem um caminho através do interior do conjunto de soluções viáveis.

Se existe a solução do primal e dual do problema de programação linear então satisfaz as condições de otimalidade dadas pelo Teorema 2.4

$$F(x,y,z) \;=\; \begin{bmatrix} Ax-b \\ A^Ty+z-c \\ XZe \end{bmatrix} \;=\; 0, \quad (x,z)\geq 0 \tag{2.7}$$

onde $X = diag(x)$, $Z = diag(z)$ e $e = (1,1,...,1)^T \in R^n$.

Este método é um processo iterativo, por isso precisamos uma direção e um tamanho do passo, para o calculo da direção $(dx,dy,dz)^T$ aplicamos o método de Newton ao sistema (2.7) e resolvemos o sistema linear seguinte:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ Z & 0 & X \end{bmatrix} \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} \;=\; \begin{bmatrix} b-Ax \\ c-z-A^Ty \\ -XZe \end{bmatrix} \tag{2.8}$$

A direção calculada pelo sistema (2.8) é denominada direção afim escala (método de pontos interiores dual-primal afim escala). Uma desvantagem do método afim-escala é que a direções de busca estão muito próximas da fronteira, tendendo a ser distorcidas devido a erros de arredondamentos. Para evitar essa aproximação se realiza uma perturbação (μ) nas condições de complementariedade (2.5).

Então do sistema (2.7) substituímos $XZe = 0$ pelo equação parametrizada $XZe = \mu e$, μ é um parâmetro positivo. Então aplicando o método de Newton ao sistema perturbado, obtemos

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ Z & 0 & X \end{bmatrix} \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} \;=\; \begin{bmatrix} b-Ax \\ c-z-A^Ty \\ \mu e-XZe \end{bmatrix} \tag{2.9}$$

Esta alteração nas condições de otimalidade gera outro método chamado o

método de pontos interiores primal-dual *path following*.

Ele mantém seus iterados próximos a um caminho central C parametrizado pelo escalar $\mu > 0$, Tomar passos próximos a C faz com que a direção de busca aponte para o interior da região determinada por $(x, z) > 0$. Isso porque, o parâmetro μ pode ser visto como o parâmetro originado do problema de barreira logarítmica definido a partir de (1.1). Para mais detalhes [54].

2.2 (FDIPA) Algoritmo de Pontos Interiores e Direções Viáveis

Nesta seção faremos uma abordagem sobre o algoritmo de pontos interiores e direções viáveis FDIPA para programação não linear, apresentado por Herkovits em [31]. Primeiramente apresentamos o problema e algumas definições.

Considere o problema de otimização

$$\begin{array}{ll} \underset{x}{\text{minimização}} & f(x) \\ \text{sujeito a} & g(x) \leq 0, \end{array} \tag{2.10}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$ e $g : \mathbb{R}^n \to \mathbb{R}^m$ são contínuas até suas segundas derivadas, e considerando a função Lagrange definida como

$$L(x, y) = f(x) - y^T g(x),$$

onde $y \in \mathbb{R}^n$ é chamado de vetor de multiplicadores de Lagrange.

O gradiente de $L(x, y)$ com respeito a x é dado por

$$\nabla_x L(\lambda, x) \;\; = \;\; \nabla f(x) + y^T \nabla g(x), \tag{2.11}$$

e denote $G(x)$ uma matriz diagonal tal que $G_{ii}(x) = g_i(x)$.

Definição 2.1. *O conjunto Ω será chamado de conjunto viável do problema* (2.10)

$$\Omega = \{x \in \mathbb{R}^n \mid g(x) \leq 0\},$$

os pontos de Ω serão chamados de pontos viáveis.

Definição 2.2. *Dizemos que um ponto $\bar{x} \in \Omega$ é*

1. *minimizador global de* (2.10)*, se*

$$f(\bar{x}) \leq f(x) \quad \text{para todo } x \in \Omega;$$

2. *minimizador local de* (2.10)*, se existe uma vizinhança* U *de* $\bar{x}$ *tal que*

$$f(\bar{x}) \leq f(x) \quad \text{para todo } x \in \Omega \cap U.$$

Definição 2.3. *O conjunto dos índices das restrições ativas no ponto* $\bar{x} \in \Omega$ *será denotado por*

$$I(\bar{x}) = \{i = 1, \ \ldots, m \mid g_i(\bar{x}) = 0\}.$$

Definição 2.4. *(Condição de regularidade) Dado um ponto* $\bar{x} \in \Omega$*, a condição de regularidade de independência linear das restrições é satisfeita em* $\bar{x}$ *se o conjunto* $\{\nabla g_i(\bar{x}), \ i \in I(\bar{x})\}$ *é linearmente independente.*

Os pontos que satisfazem a condição de regularidade de independência linear das restrições são chamados pontos regulares.

Definição 2.5. *Um ponto regular* x *é um ponto estacionário do problema* (2.10) *se existe* $y \in \mathbb{R}^m$ *tal que as seguintes condições são verificadas simultaneamente:*

$$\nabla f(x) + y^T \nabla g(x) = 0, \tag{2.12}$$

$$y^T G(x) = 0, \tag{2.13}$$

$$g(x) \leq 0. \tag{2.14}$$

Definição 2.6. *Um ponto de KKT do problema* (2.10) *é um ponto estacionário* x *associado a* $y \in \mathbb{R}^m$ *tal que* $y \geq 0$*.*

Definição 2.7. *Dizemos que* $d \in \mathbb{R}^n$ *é uma direção de descida de* $f : \mathbb{R}^n \to \mathbb{R}$ *no ponto* $\bar{x} \in \mathbb{R}^n$*, se existe* $\epsilon > 0$ *tal que*

$$f(\bar{x} + td) < f(\bar{x}) \quad \text{para todo } t \in (0, \epsilon].$$

Definição 2.8. *Dizemos que* $d \in \mathbb{R}^n$ *é uma direção viável em relação ao conjunto* Ω *no ponto* $\bar{x} \in \Omega$*, quando existe* $\epsilon > 0$ *tal que*

$$\bar{x} + td \in \Omega \quad \text{para todo } t \in [0, \epsilon].$$

FDIPA é um método iterativo, dada pela regra $x^{k+1} = x^k + t^k d^k$, onde x^k é o ponto da iteração atual, t^k é o passo e d^k a direção de busca na iteração k. Ele

gera uma sequência de pontos interiores $\{x^k\}_{k\in N}$ totalmente contida ao interior da região viável Ω, a partir de um dado ponto inicial x^0, e que converge globalmente para um ponto de KKT do problema (2.10), reduzindo a função objetivo $f(x)$ em cada iteração.

A seguir, apresentaremos as seguintes primeiras e naturais condições para o desenvolvimento do algoritmo FDIPA:

Hipótese 2.1. *Existe um número real a, tal que o conjunto $\Omega_a = \{x \in \Omega | \, f(x) \leq a\}$ é compacto e tem interior denotado por $int(\Omega_a)$.*

Hipótese 2.2. *Cada $x \in int(\Omega_a)$ satisfaz $g(x) < 0$.*

Hipótese 2.3. *As funções $f(x)$ e $g(x)$ são continuamente diferenciáveis em Ω_a e suas derivadas satisfazem a condição de Lipschitz em $\mathbb{R}^n$.*

Hipótese 2.4. *(Condição de Regularidade). Para todo $x \in \Omega_a$ o conjunto $\{\nabla g_i(x) | \, g_i(x) = 0, \, i = 1, 2, \ldots, m\}$ é linearmente independente.*

Herskovits [31] demonstrou a convergência do algoritmo FDIPA, provando todos os passos seguintes:

1. A direção d_α é uma direção de descida da função objetivo.
2. A direção d é uma direção de descida da função objetivo e direção viável do problema (2.10).
3. O algoritmo produz uma sequência de pontos convergindo a um ponto estacionário do problema.
4. Se um ponto estacionário é um ponto de convergência do algoritmo, esse deve verificar as condições de KKT.

Algoritmo 2.1. *(Algoritmo FDIPA-LP)*
Parâmetros: $\xi \in (0,1)$, $\varphi > 0$, $\eta > 0$, $v > 0$.
Dados: $x^0 \in int(\Omega)$, $\lambda^0 \in \mathbb{R}^m_{++}$, $B \in \mathbb{R}^{n\times n}$ simétrica e definida positiva.

Passo 1 *Cálculo da direção de busca.*

(i) Resolver o sistema linear em $(d_\alpha, \lambda_\alpha)$:

$$\begin{bmatrix} B & \nabla g(x) \\ \Lambda \nabla g(x)^T & G(x) \end{bmatrix} \begin{bmatrix} d_\alpha \\ \lambda_\alpha \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}$$

Se $d_\alpha = 0$ parar.

(ii) Resolver o sistema linear em $(d_\beta,\ \lambda_\beta)$*:*

$$\begin{bmatrix} B & \nabla g(x) \\ \Lambda \nabla g(x)^T & G(x) \end{bmatrix} \begin{bmatrix} d_\beta \\ \lambda_\beta \end{bmatrix} = \begin{bmatrix} 0 \\ -\lambda \end{bmatrix}$$

(iii) Calculamos o valor positivo ρ
Se $(d_\beta)^T \nabla f(x) > 0$*, calcule:*

$$\rho = \min \left\{ \varphi \|d_\alpha\|^2;\ \ (\xi - 1) \frac{(d_\alpha)^T \nabla f(x)}{(d_\beta)^T \nabla f(x)} \right\} \tag{2.15}$$

caso contrário, calcule

$$\rho = \varphi \|d_\alpha\|^2$$

(iv) Calculamos a direção d

$$d = d_\alpha + \rho d_\beta, \tag{2.16}$$

$$\bar{\lambda} = \lambda_\alpha + \rho \lambda_\beta. \tag{2.17}$$

Passo 2: *Busca Linear*
Calcula t*, o primeiro número da sequência* $\{1, v, v^2, v^3, ...\}$ *satisfaz*

$$\begin{aligned} &f(x + td) \leq f(x) + t\eta d^T \nabla f(x), \\ &g_i(x + td) < 0 \quad \textit{se } \bar{\lambda} \geq 0, \ \textit{ou} \\ &g_i(x + td) \leq g_i(x) \quad \textit{caso contrario.} \end{aligned}$$

Passo 3: *Atualização.*

(i) *Assumir*

$$x = x + td$$

e definir um novo valor para: $\lambda > 0$ *e* B *simétrica positiva definida.*

(ii) *Vá ao* **Passo 1**.

A definição da direção de busca do algoritmo FDIPA é inspirada na aplicação do método de Newton no sistema não linear de equações extraída das condições de KKT. Herskovits provou que d_α é uma direção de descida para f. No entanto, d_α pode não ser uma direção viável pois, desenvolvendo o sistema do **Passo 1 (i)** temos que se $g_i(x) \to 0$, $\nabla g_i^T(x) d_\alpha \to 0$, ou seja, quando alguma restrição se aproxima de zero, d_α tende a uma direção tangente ao conjunto viável. Para evitar este efeito, produz o efeito de deflexão de d_α para o interior do conjunto viável gerando o segundo sistema auxiliar. No **Passo 1 (iii)** ρ é escolhido convenientemente para

garantir que d também seja uma direção de descida.

B é uma matriz simétrica definida positiva de ordem $n \times n$, ela pode ser a matriz identidade, a hessiana $H(x, \lambda, \mu)$ ou uma aproximação dela obtida por alguma técnica de quase-Newton. No entanto, B deve ser sempre definida positiva para que se garanta a convergência global do problema. A descrição mais detalhada deste algoritmo pode ser encontrada em [31].

Capítulo 3

Adaptação FDIPA para Programação Linear

Neste capítulo, da adaptação do FDIPA para programação linear já foram apresentados de forma detalhada em Celis & Herskovits [11].

A formulação dual do problema (1.2), de maneira geral se escreve como

$$\begin{array}{ll} \underset{\lambda}{\text{minimização}} & f(\lambda) \\ \text{sujeito a} & g_i(\lambda) \leq 0, \quad \text{para todo } i \in I, \end{array} \tag{3.1}$$

onde $I = 1, 2, ..., n$; a_i coluna de A que foi definida no problema (1.1), c_i i-ésimo termino de c, $f(\lambda) = b^T\lambda$, $g_i(\lambda) = a_i^T\lambda - c_i$.

Observação 3.1. *Se $b = 0$, o problema* (3.1) *é resolvido para qualquer λ e o primal ótimo seria $x = 0$, por isso considera-se $b \neq 0$.*

O conjunto viável do problema (3.1) é

$$\Omega = \{\lambda \in \mathbb{R}^m : g_i(\lambda) \leq 0 \text{ para todo } i \in I\},$$

o conjunto estritamente viável é

$$int(\Omega) = \{\lambda \in \mathbb{R}^m : g_i(\lambda) < 0 \text{ para todo } i \in I\}.$$

Denotemos os conjuntos de índices de restrições ativas e inativas respetivamente

$$I(\lambda) = \{i : g_i(\lambda) = 0\},$$

$$J(\lambda) = \{i : g_i(\lambda) < 0\}.$$

A função de Lagrange do problema (3.1) é

$$L(\lambda, x) \quad = \quad f(\lambda) + \sum_{i=1}^{n} x_i g_i(\lambda), \tag{3.2}$$

e o gradiente é dado por

$$\nabla_\lambda L(\lambda, x) \quad = \quad b + Ax. \tag{3.3}$$

FDIPA é construído baseado nas condições de otimalidade de primeira ordem KKT. Assim, o Teorema 2.4 para os problemas (1.1) e (3.1) pode ser escrito como: $\lambda^* \in \Omega$ é solução de (3.1) se somente se existe $x^* \in \mathbb{R}^n$ para o qual valem as seguintes condições:

$$\begin{aligned} Ax^* + b &= 0, && (3.4)\\ G(\lambda^*)x^* &= 0, && (3.5)\\ x^* &\geq 0, && (3.6) \end{aligned}$$

onde $G(\lambda^*) = \text{diag}(g_1(\lambda^*),\, g_2(\lambda^*),\, \ldots, g_n(\lambda^*))$.

A dificuldade de resolver as condições de KKT é baseada na Equação (3.5) chamada condição de complementariedade.

FDIPA é o algoritmo de ponto interior para programação não linear que adaptaremos para programação linear. Começa a partir de um ponto estritamente viável e procura uma direção de busca para λ perturbando a condição de complementariedade (3.5), garantindo a viabilidade estrita em cada iteração até convergir para ponto KKT.

Neste trabalho consideramos o problema de minimização (3.1) não degenerado. Seja

$$\Omega_{\mathtt{p}} = \{\lambda \in \Omega |\;\; f(\lambda) \leq \mathtt{p}\},$$

onde $\mathtt{p}$ é um número real. O conjunto interior de $\Omega_{\mathtt{p}}$ é definido por $int(\Omega_{\mathtt{p}}) = \{\lambda \in int(\Omega) |\;\; f(\lambda) < \mathtt{p}\}$.

Assumem-se as seguintes hipóteses.

Hipótese 3.1. *Existe um número real $\mathtt{p}$ de tal forma que $\Omega_{\mathtt{p}}$ é compacto e possui $int(\Omega_{\mathtt{p}}) \neq 0$.*

Hipótese 3.2. *Para todo $\lambda \in \Omega$ e $i \in I(\lambda)$, os vetores $\nabla g_i(\lambda) = a_i$ são linearmente*

independentes.

3.1 Direção de Descida e Viável

Vamos reafirmar as condições de otimalidade (3.4) e(3.5), por meio de um mapeamento $F : \mathbb{R}^{2n+m} \to \mathbb{R}^{2n+m}$ e definindo o vetor $(\lambda, x)^T$.

$$F(\lambda, x) = \begin{bmatrix} Ax + b \\ G(\lambda)x \end{bmatrix} = 0, \tag{3.7}$$

$$g_i(\lambda) \leq 0, \qquad i \in I, \tag{3.8}$$

$$x \geq 0. \tag{3.9}$$

Podemos encontrar a solução das condições de KKT aplicando o método de Newton ao sistema (3.7). Seja o ponto inicial (λ^0, x^0), então as iterações do método de Newton são dadas por

$$\nabla F(\lambda^k, x^k) \begin{bmatrix} \lambda^{k+1} - \lambda^k \\ x^{k+1} - x^k \end{bmatrix} = -F(\lambda^k, x^k). \tag{3.10}$$

Define-se a direção dual $d_\alpha^k = \lambda^{k+1} - \lambda^k$, $x_\alpha^k = x^{k+1}$ e $X = \text{diag}(x_1, x_2, \ldots, x_n)$. Assim, o sistema linear (3.10) nas variáveis d_α^k e x_α^k é escrito como

$$\begin{bmatrix} 0 & A \\ X^k A^T & G(\lambda^k) \end{bmatrix} \begin{bmatrix} d_\alpha^k \\ x_\alpha^k \end{bmatrix} = \begin{bmatrix} -b \\ 0 \end{bmatrix}. \tag{3.11}$$

Observação 3.2. *As matrizes diagonais X^k e $G(\lambda^k)$ devem ser calculadas em cada iteração.*

FDIPA perturba a condição de complementariedade do sistema KKT, colocando um valor $-\rho x$ do lado direito, iterativamente o valor ρ é estritamente reduzido, assim $\rho \to 0$. Não vamos a resolver o KKT uma vez por causa do custo computacional, em vez de isso, utilizamos passos cada vez menor tendendo a zero.

$$\begin{bmatrix} 0 & A \\ X^k A^T & G(\lambda^k) \end{bmatrix} \begin{bmatrix} d^k \\ \bar{x}^k \end{bmatrix} = \begin{bmatrix} -b \\ -\rho^k x^k \end{bmatrix}. \tag{3.12}$$

O calculo de d^k e $\bar{x}^k$ pode ser realizado resolvendo um sistema auxiliar

$$\begin{bmatrix} 0 & A \\ X^k A^T & G(\lambda^k) \end{bmatrix} \begin{bmatrix} d_\beta^k \\ x_\beta^k \end{bmatrix} = \begin{bmatrix} 0 \\ -x^k \end{bmatrix}, \tag{3.13}$$

tomando as seguintes relações

$$\begin{aligned} d^k &= d^k_\alpha + \rho^k d^k_\beta, \\ \bar{x}^k &= x^k_\alpha + \rho^k x^k_\beta, \end{aligned} \tag{3.14}$$

onde, d^k é uma direção viável e de descida.

3.1.1 Busca Linear

É possível procurar um passo $t^k > 0$ na direção d^k, tal que o novo ponto $\lambda^{k+1} = \lambda^k + t^k d^k$ mantenha as condições $f(\lambda^{k+1}) < f(\lambda^k)$ e $g(\lambda^{k+1}) < 0$.

Avaliando f no ponto λ^{k+1},

$$f(\lambda^{k+1}) = f(\lambda^k) + t^k d^{kT} b^k, \tag{3.15}$$

se $d^{kT} b^k < 0$, tem-se

$$f(\lambda^{k+1}) < f(\lambda^k). \tag{3.16}$$

Na seguinte seção mostramos que $d^{kT} b^k < 0$. Portanto d é uma direção de descida para qualquer positivo t.

Avaliando g no ponto λ^{k+1}, para todo $i \in I$,

$$g_i(\lambda^{k+1}) = g_i(\lambda^k) + t^k d^{kT} a_i, \tag{3.17}$$

- se $d^{kT} a_i < 0$ então $g_i(\lambda^{k+1}) < g_i(\lambda^k)$ para todo $t^k \in \mathbb{R}_{++}$.

- se $d^{kT} a_i = 0$ então $g_i(\lambda^{k+1}) = g_i(\lambda^k)$ para todo $t^k \in \mathbb{R}_{++}$.

- se $d^{kT} a_i > 0$, da Equação (3.17) para garantir a viabilidade ($g_i(\lambda^{k+1}) \leq 0$), deve cumprir-se

$$t^k \leq \frac{-g_i(\lambda^k)}{d^{kT} a_i}. \tag{3.18}$$

O valor de t^k é

$$t^k = \min_i \left\{ \frac{-g_i(\lambda^k)}{d^{kT} a_i} \mid \ d^{kT} a_i > 0 \right\}. \tag{3.19}$$

3.2 Algoritmo FDIPA-LP

Algoritmo 3.1. *(Algoritmo FDIPA-LP)*
Parâmetros: $\xi \in (0,1)$, $\varphi > 0$, $\mu > 0$, $\gamma > 0$, $x^s > 0$.
Dados: $\lambda^0 \in int(\Omega)$, $x^0 \in \mathbb{R}^m_{++}$.

Passo 1 *Cálculo da direção de busca.*

(i) Resolver o sistema linear em $(d^k_\alpha,\ x^k_\alpha)$*:*

$$\begin{aligned} Ax^k_\alpha &= -b \qquad (3.20)\\ X^k A^T d^k_\alpha + G(\lambda^k)x^k_\alpha &= 0. \qquad (3.21)\end{aligned}$$

se $d^k_\alpha = 0$ *parar.*

(ii) Resolver o sistema linear em $(d^k_\beta,\ x^k_\beta)$*:*

$$\begin{aligned} Ax^k_\beta &= 0 \qquad (3.22)\\ X^k A^T d^k_\beta + G(\lambda^k)x^k_\beta &= -x^k. \qquad (3.23)\end{aligned}$$

(iii) Calculamos o valor positivo ρ^k
Se $(d^k_\beta)^T b > 0$*, calcule:*

$$\rho^k = \min\left\{\varphi\|d^k_\alpha\|^2;\ \ (\xi-1)\frac{(d^k_\alpha)^T b}{(d^k_\beta)^T b}\right\} \qquad (3.24)$$

caso contrário, calcule

$$\rho^k = \varphi\|d^k_\alpha\|^2$$

(iv) Calculamos a direção d^k

$$d^k = d^k_\alpha + \rho^k d^k_\beta \qquad (3.25)$$

Passo 2 *Busca linear*

(i) Se $d^{kT}a_i > 0;\ \ i = 1, ..., m$ *calcule*

$$\bar{t}^k = \min\left\{\frac{-g_i(\lambda^k)}{d^{kT}a_i}\right\}, \qquad (3.26)$$

calcule

$$t^k = \min\{\max\{\gamma\bar{t}^k, \bar{t}^k - \|d\|\}, 1\}. \qquad (3.27)$$

Passo 3 *Atualizações.*

(i) Obtenha o novo ponto

$$\lambda^{k+1} = \lambda^k + t^k d^k.$$

(ii) Calcule

$$x_i^{k+1} = min\{max\{x_{\alpha i}, \mu\|d_\alpha\|^2\}, x^s\}, \quad para\ todo\ i \in I. \tag{3.28}$$

(iii) Retorne ao **Passo 1**.

O algoritmo realiza iterações dentro da região viável. No **Passo 1 (ii)**, obtém-se o valor da perturbação ρ que garante uma direção de descida d. No **Passo 2**, t é o passo máximo que garante a viabilidade, e com a expressão (3.27) temos uma viabilidade estrita. No **Passo 3 (ii)**, usamos a mesma regra de programação não linear do FDIPA para atualizar x.

3.2.1 Convergência do Algoritmo FDIPA Linear

O Lema a seguir mostra que os sistemas (3.11) e (3.13) possuem soluções, portanto o **Passo 1** do algoritmo está bem definido.

Lema 3.1. *Seja $\lambda \in \Omega$ e $x \in \mathbb{R}^n$ tal que $x_i > 0$ para todo $i \in I$. Então*

$$N(\lambda, x) = \begin{bmatrix} 0 & A \\ XA^t & G(\lambda) \end{bmatrix} \tag{3.29}$$

não é singular.

Demonstração. Provaremos que, se para algum $(d, \bar{x}) \in \mathbb{R}^m \times \mathbb{R}^n$, a solução do sistema $N(\lambda, x)\begin{pmatrix} d \\ \bar{x} \end{pmatrix} = 0$ admite somente a solução trivial; ou seja, $(d, \bar{x}) = 0$. Assim, tem-se o seguinte sistema

$$A\bar{x} = 0, \tag{3.30}$$

$$XA^T d + G(\lambda)\bar{x} = 0. \tag{3.31}$$

Multiplicando na Equação (3.30) por d^T resulta

$$d^T A\bar{x} = 0. \tag{3.32}$$

Dado $X > 0$, multiplicando na Equação (3.31) por $\bar{x}^T X^{-1}$ resulta

$$\bar{x}^T A^T d + \bar{x}^T X^{-1} G(\lambda)\bar{x} = 0. \tag{3.33}$$

De (3.32) e (3.33) temos que

$$\bar{x}^T X^{-1} G(\lambda)\bar{x} \;=\; 0. \tag{3.34}$$

Portanto

$$G(\lambda)\bar{x} \;=\; 0. \tag{3.35}$$

Substituindo a Equação (3.35) em (3.31) temos que

$$A^T d \;=\; 0, \tag{3.36}$$

$$\begin{bmatrix} a_{11}d_1 + a_{21}d_2 + \cdots + a_{m1}d_m \\ a_{12}d_1 + a_{22}d_2 + \cdots + a_{m2}d_m \\ \vdots \\ a_{1n}d_1 + a_{2n}d_2 + \cdots + a_{mn}d_m \end{bmatrix} = 0,$$

$$(a_{11}, a_{12}, \cdots, a_{1n})^T d_1 + (a_{21}, a_{22}, \cdots, a_{2n})^T d_2 + ... + (a_{m1}, a_{m2}, \cdots, a_{mn})^T d_m = 0,$$

dado que A tem posto completo, os vetores filas de A são l.i., então $d_1 = d_2 = ... = d_m = 0$. Assim $d = 0$.

Por outro lado, particionando $\bar{x} = [\bar{x}_I, \bar{x}_J]^T$ e da Equação (3.35), quando $i \in J(\lambda)$ temos que $\bar{x}_J = 0$.

Da Equação (3.30) e do fato que $G(\lambda) \leq 0$,

$$A_I \bar{x}_I \;+\; A_J \bar{x}_J = 0, \tag{3.37}$$

$$A_I \bar{x}_I \;=\; 0. \tag{3.38}$$

Pela **Hipótese 3.2** temos que $\bar{x}_I = 0$. Assim, a solução do sistema é $(d, \bar{x}) = (0, 0)$. □

Consequentemente, d_α, x_α, d_β, x_β obtidos no **Passo 1** do algoritmo estão bem definidos e são únicos.

Note que, se o algoritmo termina na iteração $k \in \mathbb{N}$, $d_\alpha^k = 0$ então λ^k resolve na Equação (3.21). Agora, para a Equação (3.20), temos $x_\alpha^k = 0$ e, consequentemente, $b = 0$. Portanto, quando $b \neq 0$, o algoritmo gera uma sequência infinita de x_α^k. De agora em diante, estudaremos o caso $d_\alpha^k \neq 0$ em cada iteração.

As iterações do Algoritmo FDIPA-LP geram $\lambda^k \in int(\Omega)$ e $X^k > 0$.

O lema a seguir mostra que d é uma direção de descida.

Lema 3.2. *Seja* $\xi \in (0,1)$*, se* $d \neq 0$ *e* $d_\alpha \neq 0$ *, então*

$$d^T b \leq \xi d_\alpha^T b < 0. \tag{3.39}$$

Demonstração. Do primeiro sistema do Algoritmo FDIPA-LP, multiplica-se na Equação (3.20) por d_α^T

$$d_\alpha^T A x_\alpha^k = -d_\alpha^T b \tag{3.40}$$

Dado $X > 0$, multiplicando na Equação (3.21) por $x_\alpha^T X^{-1}$ resulta

$$x_\alpha^T A^T d_\alpha + x_\alpha^T X^{-1} G(\lambda) x_\alpha = 0. \tag{3.41}$$

substituir (3.40) na Equação (3.41),

$$d_\alpha^T b = x_\alpha^T X^{-1} G(\lambda) x_\alpha. \tag{3.42}$$

Assim, $X^{-1}G(\lambda)$ é uma matriz definida negativa, para qualquer $x_\alpha \neq 0$, temos que

$$d_\alpha^T b < 0. \tag{3.43}$$

Alem disso, multiplicamos na Equação (3.25) por b temos que

$$d^T b = d_\alpha^T b + \rho d_\beta^T b, \tag{3.44}$$

do algoritmo no **Passo 1 (iii)** é claro que $\rho > 0$ nos dois casos, ademais

- se $d_\beta^T b \leq 0$, das Equações (3.44) e (3.43) temos que

$$d^T b \leq d_\alpha^T b < 0. \tag{3.45}$$

- se $d_\beta^T b > 0$, da Equação (3.24) tem-se que

$$\begin{aligned} \rho &\leq (\xi - 1)\frac{d_\alpha^T b}{d_\beta^T b}, \\ d_\alpha^T b + \rho d_\beta^T b &\leq \xi d_\alpha^T b, \\ d^T b &\leq \xi d_\alpha^T b. \end{aligned} \tag{3.46}$$

Por tanto, das Equações(3.43) e (3.46),

$$d^T b \leq \xi d_\alpha^T b < 0. \tag{3.47}$$

□

Como consequência do lema anterior, f é monótona decrescente, d_α e d são direções de descida de f, sendo

$$f(\lambda + td) = f(\lambda) + td^T b,$$

portanto

$$f(\lambda + td) < f(\lambda),$$

por construção, $t \in (0, 1]$.

Por outro lado, com a escolha de ρ no **Passo 1** do Algoritmo FDIPA-LP, garantimos que

$$0 < \rho \leq \varphi \|d_\alpha\|^2. \tag{3.48}$$

Agora vamos mostrar viabilidade de d.

Lema 3.3. *Existe $\tau > 0$ tal que em qualquer Ω_p e d calculados no* **Passo 1** *do Algoritmo 3.1, temos $g(\lambda + td) \leq 0$ para todo $t \in [0, \tau]$.*

Demonstração. O comprimento do passo t é definido na Equação (3.27) do algoritmo. Como as restrições são lineares, para satisfazer a condição de busca linear, as seguintes desigualdades devem ser verdadeiras:

$$g_i(\lambda + td) = g_i(\lambda) + ta_i^T d \leq 0, \tag{3.49}$$

para todo $i \in I$. Se $a_i^T d \leq 0$, a desigualdade acima é satisfeita com qualquer $t > 0$. Caso contrário, segue de (3.12) que

$$g_i(\lambda)\left(1 - t\frac{\bar{x}_i}{x_i}\right) - t\rho \leq 0, \tag{3.50}$$

obviamente $\rho t > 0$ e $g(\lambda) < 0$. Assim, a desigualdade é satisfeita se

$$t\left(\frac{\bar{x}_i}{x_i}\right) \leq 1. \tag{3.51}$$

Agora, x é limitado por (3.28) e, como x_α, x_β e ρ são limitados por cima, também $\bar{x}$ é limitado por cima. Assim, existe $\tau > 0$ tal que $x_i/\bar{x}_i > \tau$. Portanto, para todo

$i \in I$ e para todo $t \in [0, \tau]$, temos $g_i(\lambda + td) \leq 0$. □

Considerando $f(\lambda^0) = \mathtt{p}$, notamos que, dado um ponto $\lambda^0 \in int(\Omega)$ e uma função decrescente f, temos qualquer sequência $\{\lambda^k\}$ gerada pelo Algoritmo 3.1 está contido em $\Omega_{\mathtt{p}}$. Então, como **Hipótese 1**, $\{\lambda^k\}$ tem ponto de acumulação em $\Omega_{\mathtt{p}}$.

Lema 3.4. *Cada ponto de acumulação da sequência $\{\lambda^k\}$, gerada pelo Algoritmo 3.1, é um ponto estacionário do problema. Além disso, (λ^*, x^*_α) é um par estacionário, onde x^*_α é o único vetor multiplicador associado a λ^*.*

Demonstração. Considere uma sequência $\{\lambda^k\}_{k\in K}$, onde $K \subset \mathbb{N}$, convergindo para λ^*.

Pela construção $x^k_i \in [0, x^s]$ e (3.48), segue que $\{x^k\}$ e $\{\rho^k\}$ são limitados, então existe $K_1 \subset K$, tal que

$$\{\lambda^k, x^k, \rho^k\}_{k\in K_1}$$

convergem para (λ^*, x^*, ρ^*).

Como d^k depende continuamente de λ^k, x^k e ρ^k, temos que $\{d^k\}_{k\in K_1} \to d^*$, com $d^* = d(\lambda^*, x^*, \rho^*)$.

Como $t^k \in (0, 1]$ para todo k, segue do Lema 3.2 que existe $\hat{t} > 0$ tal que

$$f(\lambda^{k+1}) \leq f(\lambda^k) + \hat{t} d^{kT} b, \quad \text{para todo } k \in K_1$$

onde assumimos que $k \in K_1$ e $k + 1$ não necessariamente pertencem a K_1.

$$f(\lambda^{k'}) < f(\lambda^k) + \hat{t} d^{kT} b \quad \text{para todo } k' \geq k + 1,$$

e tomando limites em ambos os lados para $k \to \infty$,

$$f(\lambda^*) \leq f(\lambda^*) + \hat{t} d^{*T} b,$$

logo, segue que

$$0 \leq d^{*T} b. \tag{3.52}$$

Considerando agora o resultado do Lema 3.2 no limite de $k \to \infty$ temos

$$d^{*T} b = 0. \tag{3.53}$$

Também segue do Lema 3.2, $d^{*T}_\alpha = 0$.

Tomando limites em (3.20) e (3.21), temos

$$A^T x^*_\alpha = -b, \tag{3.54}$$
$$G(\lambda^*)x^*_\alpha = 0. \tag{3.55}$$

Assim, λ^* é estacionário, com vetor multiplicador x^*_α. Da Equação (3.54) e da **Hipótese 2** temos a unicidade de x^*.

□

Lema 3.5. *Suponha que $\{\lambda^k\}$ seja limitado. Seja L o conjunto de pontos limite de $\{\lambda^k\}$, então L são conjuntos compactos conexos com as mesmas restrições ativas.*

Demonstração. É análoga à prova do Lema A.5 e Lema 3.6 em Tits [52]. O Lema A.5 prova que L é um conjunto compacto conexo, e o Lema 3.6 prova que L possui um único $\bar{x}^*$ associado a ele. □

Lema 3.5 aparece como hipótese no FDIPA para Programação Não-linear [31] permitindo demonstrar a convergência da sequência de pontos do algoritmo.

A seguir, um lema técnico para entender a convergência.

Lema 3.6. *Seja $i \in I$, se $d \neq 0$ então*

$$d^T a_i < 0, \ \textit{para cada } \bar{x}_i < 0. \tag{3.56}$$

Demonstração. Reescrevendo as Equações (3.21) e (3.23)

$$XA^T d_\alpha + G(\lambda)x_\alpha = 0,$$
$$XA^T d_\beta + G(\lambda)x_\beta = -x.$$

Obtém-se para cada $i \in I$,

$$a_i^T d_\alpha = -\frac{x_{\alpha i}}{x_i} g_i(\lambda), \tag{3.57}$$
$$a_i^T d_\beta = -\frac{x_{\beta i}}{x_i} g_i(\lambda) - 1. \tag{3.58}$$

Multiplique-se a_i^T na Equação (3.25)

$$a_i^T d = a_i^T d_\alpha + \rho a_i^T d_\beta. \tag{3.59}$$

Substituindo (3.57) e (3.58) em (3.59) resulta

$$\begin{aligned} a_i^T d &= -\frac{x_{\alpha i}}{x_i} g_i(\lambda) - \rho \frac{x_{\beta i}}{x_i} g_i(\lambda) - \rho, \\ &= -\frac{g_i(\lambda)}{x_i}(x_{\alpha i} + \rho x_{\beta i}) - \rho \\ &= -\frac{g_i(\lambda)}{x_i} \bar{x}_i - \rho, \end{aligned} \tag{3.60}$$

na Equação (3.60) depende do valor de $\bar{x}_i$, então

- Se $\bar{x}_i < 0$ resulta

$$a_i^T d < 0. \tag{3.61}$$

As funções das restrições associadas com multiplicadores negativo estimados, são funções decrescentes ($g_i(\lambda + td) < g_i(\lambda)$). □

As funções de restrições associadas aos multiplicadores negativos estimados são funções decrescentes ($g_i(\lambda + td) < g_i(\lambda)$). Isso garante que pontos estacionários, que não sejam pontos KKT, sejam evitados.

Finalmente, o Teorema 3.1 demostra que o algoritmo apresentado converge para a solução do problema.

Teorema 3.1. *Qualquer ponto de acumulação λ^* de qualquer sequência gerada pelo algoritmo $\{\lambda^k\}$ é um ponto de Karush-Kuhn-Tucker do problema.*

Demonstração. Como λ^* é um ponto estacionário do problema (3.1), basta provar que os multiplicadores de Lagrange x_α^* são não negativos.

Temos que $d_\alpha^* = 0$ e $d^* = 0$ em um ponto estacionário. Então, segue de (3.11) que

$$G(\lambda^*) x_\alpha^* = 0,$$

onde $x_{\alpha i}^* = 0$ no caso de $g_i(\lambda^*) < 0$.

Considere agora uma restrição $g_j(\lambda^*) = 0$, produzindo as seguintes consequências. Como o método é estritamente viável,

$$g_j(\lambda^k) < 0 \text{ para todo } k \in \mathbb{N}.$$

Então, podemos definir uma sequência $\{\lambda^k\}_{k\in K}$, $K \subset \mathbb{N}$, convergindo para λ^* tal que

$$g_j(\lambda^k) > g_j(\lambda^{k-1}) \text{ para qualquer } k \in K.$$

Observe que $k-1$ pode não pertencer a K, e do Lema 3.6 obtemos $\bar{x}_j^{k-1} \geq 0$.

Agora, procedemos por contradição e assumimos que $x_{\alpha j}^* < 0$. Segue de (3.48) que $\rho^* = 0$ e de (3.28)

$$\bar{x}^* = x_\alpha^* + \rho^* x_\beta^*,$$

temos $\bar{x}_j^* < 0$. Logo, do Lema 3.5 existe uma bola aberta $B(\gamma_j) \equiv \{\lambda \mid \ \|\lambda - \lambda^*\| < \gamma_j\}$, tal que $\bar{x}_j < 0$ para qualquer $\lambda \in \{int(\Omega_{\mathtt{p}}) \cap B(\gamma_j)\}$, e qualquer x e ρ gerados pelo algoritmo.

Considere agora a sequência $\{\lambda^{k-1}\}_{k\in K}$. Como $\bar{x}_j^{k-1} \geq 0$, para qualquer $k \in K$, temos que $\lambda^{k-1} \notin B(\gamma_j)$. Assim, $\{\lambda^{k-1}\}_{k\in K}$ está no conjunto compacto $\{\Omega_{\mathtt{p}} - B(\gamma_j)\}$, portanto este possui um ponto de acumulação. Em consequência do Lema 3.4, este é um ponto estacionário do problema (1.2).

Seja $\bar{\lambda}$ um desses pontos e $\{\lambda^{k-1}\}_{k\in \bar{K}}$, $\bar{K} \subset K$, uma sequência convergente para $\bar{\lambda}$ portanto temos que $\|d^{k-1}\| \to 0$. Então, para I grande o suficiente, $\|d^{k-1}\| < \gamma_j$ para qualquer $k > I$, $k \in \bar{K}$, e em consequência da busca linear,

$$\begin{aligned} \|\lambda^k - \lambda^{k-1}\| &= |t^{k-1}|\|d^{k-1}\| \\ \|\lambda^* - \bar{\lambda}\| &< \gamma_j. \end{aligned}$$

□

Capítulo 4

Implementação de Programação Linear com FDIPA

Neste capítulo, os sistemas internos do FDIPA para programação linear já foram apresentados de forma detalhada em Celis & Herskovits [11].

4.1 Resolução dos Sistemas Lineares Internos do FDIPA-LP

Em cada iteração do algoritmo FDIPA-LP é calculada uma direção de descida e viável. Para isso, emprega-se uma iteração de Newton as condições de KKT que gera os sistemas lineares.

As soluções dos sistemas lineares são obtidos por o método direto ou método iterativo. Em muitos casos, o método iterativo tem um custo computacional muito menor do que o necessário para obter a solução exata por meio de um método direto.

Os métodos iterativos partem de uma estimativa inicial da solução e geram uma sequência de pontos que convergem para a solução do sistema. Nós usamos o método de gradiente conjugado pré-condicionado *Split*, na prática, qualquer dos pontos da sequencia é candidato a ser direção de busca.

Primeiramente, apresentaremos o método de gradiente conjugado pré-condicionado *Split*, necessário para a resolução dos sistemas lineares.

4.1.1 Método do Gradiente Conjugado Pré-condicionado *Split*

Na solução de sistemas com matrizes simétricas e definidas positivas, é utilizado o método do gradiente conjugado pré-condicionado *Split*. Quando trabalhamos com matrizes mal condicionadas, é necessário o uso de pré-condicionadores para garantir a convergência do método [50].

O algoritmo para resolver $Qx = B$ é detalhado abaixo, com a decomposição incompleta de Cholesky L onde $M = LL^T$ é simétrica definida positiva, sendo L o pré-condicionador de Q.

Algoritmo 4.1. *Algoritmo do Método do Gradiente Conjugado Pré-condicionado Split*

Dados: $Q \in \mathbb{R}^{n\times n}$ simétrica definida positiva, $B \in \mathbb{R}^n$ e $x_0 \in \mathbb{R}^n$.

Passo 1: Inicialização

$$\begin{aligned} r_0 &= B - Qx_0, \\ \hat{r}_0 &= L^{-1}r_0, \\ p_0 &= L^{-T}\hat{r}_0. \end{aligned}$$

Passo 2: *Loop* principal

Para $j = 0, 1, ...,$ até convergir, fazer

$$\begin{aligned} \alpha_j &= (\hat{r}_j, \hat{r}_j)/(Qp_j, p_j), \\ x_{j+1} &= x_j + \alpha_j p_j, \\ \beta_j &= (\hat{r}_{j+1}, \hat{r}_{j+1})/(\hat{r}_j, \hat{r}_j), \\ p_{j+1} &= L^{-T}\hat{r}_{j+1} + \beta_j p_j, \end{aligned}$$

fim.

O critério de convergência usado aqui foi $\frac{\|\hat{r}_j\|}{\hat{r}_0} < \epsilon$. Também calculamos L, um pré-condicionador Cholesky Incompleto, usando a função ICHOL do Matlab.

A escolha de um bom pré-condicionador é muito importante, Golub & Van Loan [20] e Greenbaum [24] consideram o pré-condicionador do tipo decomposição incompleta de Cholesky um eficiente pré-condicionador para sistemas com matriz de coeficientes esparsa.

4.1.2 Sistema Linear Interno do FDIPA para Problemas Lineares com Desigualdades

Os sistemas lineares internos do FDIPA-LP podem ser resolvidos por vários métodos numéricos, que dependem da forma e das características que obtêm quando manipulados.

Da Equação (3.20)-(3.21) obtemos

$$\begin{aligned} -AG^{-1}(\lambda)XA^T d_\alpha &= -b, && (4.1)\\ x_\alpha &= -G^{-1}(\lambda)XA^T d_\alpha. && (4.2) \end{aligned}$$

Do sistema (3.22)-(3.23) obtemos

$$\begin{aligned} -AG^{-1}(\lambda)XA^T d_\beta &= A^T G^{-1}(\lambda)X, && (4.3)\\ x_\beta &= -G^{-1}(\lambda)(x + XA^T d_\beta). && (4.4) \end{aligned}$$

A matriz $Q = -AG^{-1}(\lambda)XA^T$ é garantida simétrica e definida positiva para todo $\lambda \in int(\Omega_{\mathtt{p}})$. Portanto, o método SPCG pode ser usado para resolver (4.1) e (4.3).

Observamos, no desenvolvimento do algoritmo FDIPA-LP, que as iterações geradas pelo SPCG são candidatas a uma direção de descida viável. Portanto, paramos o método SPCG em uma iteração inicial (SPCG-T). O que mostra-se ser uma alternativa atrativa e econômica para problemas de grande escala.

Na Tabela 4.1, comparamos o número de iterações e tempo de execução, do SPCG (Iter SPCG) versus SPCG com iteração truncada (Iter SPCG-T), necessários para resolver os sistemas internos FDIPA-LP para problemas testados da coleção NETLIB [19]. A primeira coluna contém o nome do problema.

Observando que em ambos os casos, obtivemos o mesmo número de iterações do FDIPA para atingir o valor objetivo. Note que com SPCG-T o número de iterações para resolver os sistemas internos é reduzido, pois paramos o processo na primeira iteração, o que significa que menos operações numéricas são realizadas para chegar ao valor objetivo, reduzindo o custo computacional.

Tabela 4.1: Comparação de iterações e tempo SPCG vs SPCG-T

Problema	Iter SPCG	Tempo SPCG	Iter SPCG-T	Tempo SPCG-T
Adlittle	92	0.168s	46	0.124s
Afiro	38	0.083s	20	0.061s
Agg	259	2.058s	92	2.024s
Agg2	141	2.950ss	54	2.742s
Agg3	164	3.970s	86	3.689s

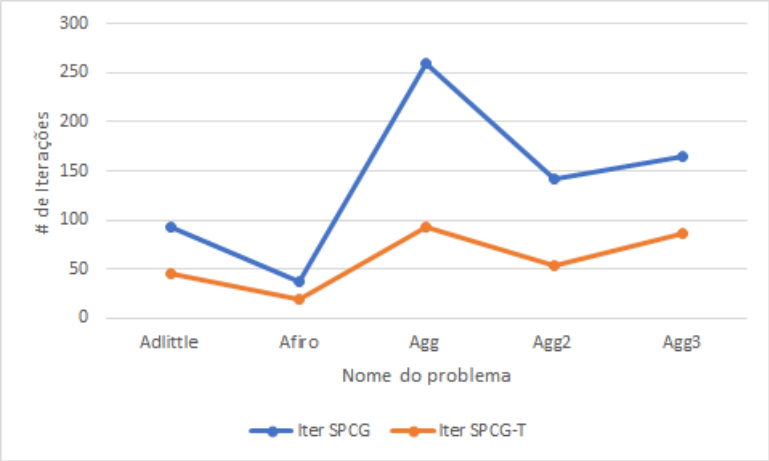


Figura 4.1: Número de iterações SPCG e SPCG-T.

4.2 Detalhes da Implementação do Algoritmo

4.2.1 Pré-Solucionador

Antes de iniciar o algoritmo tentamos simplificar o problema, eliminando redundâncias e simplificando as restrições seguindo [47]. Com isso conseguimos reduzir o tamanho do problema e o tempo de execução.

Na Tabela 4.2 apresentamos alguns resultados computacionais que defendem o uso de uma análise com pré-solucionador. As colunas variáveis e restrições mostram o número de linhas e colunas em A, respectivamente. As seguintes colunas variáveis e restrições mostram os mesmos números, depois do pré-solucionador. Os resultados coletados na Tabela 4.2 também mostram que existem problemas quase irredutíveis, por exemplo, Afiro, Degen 2, Degen 3, Israel, Qap 8 e Qap 12.

Também usamos técnicas de escalonamento, pois elas melhoram as propriedades computacionais do problema de PL. O dimensionamento é usado antes da aplicação

Tabela 4.2: Número de variáveis e restrições antes da pré-solucionador vs após a pré-solucionador

	Problema		Problema com Pré-solucionador	
Nome	Variáveis	Restrições	Variáveis	Restrições
Adlittle	138	56	137	55
Afiro	51	27	51	27
Agg	615	488	478	390
Agg2	758	516	755	514
Agg3	758	516	755	514
Blend	114	74	111	71
Degen2	757	444	757	444
Degen3	2604	1503	2604	1503
Israel	316	174	316	174
Qap8	1632	912	1632	912
Qap12	8856	3192	8856	3192
Sc205	317	205	315	203
Sc50A	78	50	77	49
Sc50B	78	50	76	48
Scagr7	185	129	183	127
Scagr25	671	471	669	469
Scorpion	466	388	412	346
Sctap1	660	300	644	284
Sctap2	2500	1090	2443	1033
Sctap3	3340	1480	3268	1408
Share1b	253	117	248	112
Stocfor1	165	117	161	113

do algoritmo FDIPA-LP para reduzir o número de condicionamento da matriz de restrição, melhorar o desempenho dos algoritmos, reduzir o número de iterações necessárias para resolver LP e simplificar a configuração de tolerâncias.

Para mais detalhes de sua formulação matemática, exemplo numérico ilustrativo e implementação em MATLAB de cada tipo de técnica de escalonamento veja [47], apresentam onze técnicas de escalonamento onde é realizado um estudo computacional, comparando o tempo de execução das técnicas de escalonamento,

e investigando o impacto do escalonamento antes da aplicação de algoritmos LP. Usaremos 6 tipos de técnicas de escalonamento apresentadas na Tabela 4.3.

Tabela 4.3: Técnicas de escalonamento.

Número	técnicas de escalonamento.
1	Média aritmética
2	Buchet1
3	Buchet2
4	BuchetInf
5	Entropia
6	Equilíbrio

4.2.2 Ponto Inicial

No algoritmo, assumimos que partimos de um ponto inicial λ^0 viável e interior. Este ponto será encontrado resolvendo o seguinte problema:

$$\begin{aligned} &\underset{\lambda,z}{\text{minimização}} \quad z \\ &\text{sujeito a} \qquad A^T\lambda - c \leq ze_m, \end{aligned} \tag{4.5}$$

obtemos um ponto viável quando $z < 0$.

4.2.3 Critério de Parada

Devemos estabelecer um critério para determinar quando o ponto atual λ^k está próximo o suficiente para o ponto ótimo λ^*.

Nosso critério é parar quando a melhoria relativa na função objetivo for pequena, isto é

$$\frac{|f(\lambda^k) - f(\lambda^{k+1})|}{1 + |f(\lambda^k)|} \leq \epsilon,$$

e $d_\alpha = 0$. Nós usamos $\epsilon = 10^{-8}$.

4.3 Resultados Numéricos

Nesta seção, relatamos resultados experimentais da coleção NETLIB [19]. Esta biblioteca reúne 98 problemas de LP de diferentes áreas entre eles tem 44 problemas com restrições de caixa.

Todo problema de programação linear primal tem outro problema associado chamado dual, onde no ponto ótimo cada um tem o mesmo valor objetivo. A teoria da dualidade relaciona ambos os problemas [54] e [38]. O problema dual se encaixa bem para ser resolvido com o FDIPA-LP proposto neste trabalho.

Nossa implementação foi feita em ambiente MATLAB. Todos os experimentos computacionais foram realizados em um microcomputador com processador Intel CORE i5, 8GB de RAM e 2,40GHz de frequência rodando na plataforma Windows 10.

O primeiro passo para resolver um problema de teste foi realizar pré-solucionador descrita na Seção 4.2.1, logo aplicar um tipo de técnica de escalonamento da Tabela 4.3. Em seguida, FDIPA-LP precisa como entrada um ponto λ^0 dentro da região viável, mas a biblioteca NETLIB não fornece pontos viáveis para seus problemas. Portanto, busca-se uma solução viável $\lambda^0 \in int(\Omega)$ resolvendo o problema auxiliar (4.5) com FDIPA-LP, o processo de otimização é interrompido quando a função objetivo atinge um valor negativo z^0.

Nas tabelas 4.4, 4.5, apresentamos 51 problemas de programação linear resolvidos com FDIPA-LP. As seis colunas fornecem, respectivamente, o nome do problema, o valor objetivo primal, o valor objetivo dual com FDIPA, o número de iterações necessárias para convergência, a soma do número de iterações SPCG truncado necessárias para resolver dois sistemas, o número de iterações necessárias para calcular o ponto inicial.

Os problemas Adlittle até Stocfor2 são problemas sem restrições de caixa, que convergiu com nosso FDIPA-LP. Os problemas restantes sem restrição de caixa não conseguimos encontrar o ponto inicial com nosso problema auxiliar (4.5). Além disso, conseguimos resolver 18 problemas com variáveis limitadas (Bore3d até Pilotnov).

O problema Standgub na biblioteca Netlib não convergiu, e com FDIPA-LP implementado aqui convergiu ao valor reportado na literatura [47].

Note que uma boa comparação de algoritmos de otimização requer testes para vários pontos iniciais distintos. Os softwares comerciares de LP, encontram o ponto inicial estritamente viável resolvendo um problema de programação linear ou algum problema de viabilidade ao qual nem sempre temos aceso. Esta comparação não foi feita neste trabalho.

Tabela 4.4: Testes numéricos do FDIPA

Problemas	Valor objetivo	Valor objetivo(dual) FDIPA	Iter FDIPA	Iter CG-T	Iter x^0
Adlittle	2.2549496316E+05	2.2549496316E+05	20	40	5
Afiro	-4.6475314286E+02	-4.6475314286E+02	11	22	1
Agg	-3.5991767287E+07	-3.5991767287E+07	39	78	4
Agg2	-2.0239252356E+07	-2.0239252356E+07	30	60	2
Agg3	1.0312115935E+07	1.0312115935E+07	37	54	1
Bandm	-1.5862801845E+02	-1.5862801845E+02	31	62	5
Beaconfd	3.3592485807E+04	3.3592485807E+04	24	48	2
Blend	-3.0812149846E+01	-3.0812149846E+01	15	30	5
Degen2	-1.4351780000E+03	-1.4351780000E+03	19	38	2
Degen3	-9.8729400000E+02	-9.8729400004E+02	27	100	7
Fffff800	5.5567961165E+05	5.5567956482E+05	79	242	22
Israel	-8.9664482186E+05	-8.9664482186E+05	23	46	6
Qap8	2.0350000000E+02	2.0349999989E+02	18	61	3
Qap12	5.2289435056E+02	5.2289435056E+02	34	68	2
Sc105	-5.2202061212E+01	-5.2202061212E+01	16	32	4
Sc205	-5.2202061212E+01	-5.2202061212E+01	18	36	9
Sc50A	-6.4575077059E+01	-6.4575077059E+01	13	26	3
Sc50B	-7.0000000000E+01	-7.0000000000E+01	11	22	2
Scagr7	-2.3313892548E+06	-2.3313898243E+06	21	42	7
Scagr25	-1.4753433061E+07	-1.4753433061E+07	27	54	10
Scorpion	1.8781248227E+03	1.8781248227E+03	21	42	4
Scsd1	8.6666666743E+00	8.6666666743E+00	11	22	2
Scsd6	5.0500000078E+01	5.0500000075E+01	18	36	2
Scsd8	9.0499999993E+02	9.0499999993E+02	16	32	2

Tabela 4.5: Testes numéricos do FDIPA

Problemas	Valor objetivo	Valor objetivo(dual) FDIPA	Iter FDIPA	Iter CG-T	Iter x^0
Sctap1	1.4122500000E+03	1.4122500000E+03	17	34	6
Sctap2	1.7248071429E+03	1.7248071429E+03	15	30	4
Sctap3	1.4240000000E+03	1.4240000000E+03	19	38	4
Share1b	-7.6589318579E+04	-7.6589318579E+04	33	66	9
Share2b	-4.1573224074E+02	-4.1573224074E+02	21	42	2
Ship04l	1.7933245380E+06	1.7933245380E+06	38	76	5
Ship04s	-1.7987147004E+06	1.7987147004E+06	24	48	5
Stocfor1	-4.1131976219E+04	-4.1131976220E+04	14	28	17
Stocfor2	-3.9024408538E+04	-3.9024408538E+04	32	64	39
Bore3d	1.3730803942E+03	1.3730803942E+03	21	42	22
Ganges	-1.0958636356E+05	-1.0958638221E+05	20	40	7
Gfrd-pnc	6.9022359995E+06	6.9022359957e+06	25	50	13
Grow7	-4.7787811815E+07	-4.7787811815E+07	21	42	1
Grow22	-1.6083433648E+08	-1.6083433648E+08	25	50	2
Grow15	-1.0687094129E+08	-1.0687094129E+08	25	50	4
Capri	2.6900129138E+03	2.6900129127E+03	51	102	21
Etamacro	-7.5571521774E+02	-7.5571541174E+02	55	110	8
Fit1d	-9.1463780924E+03	-9.1463780924E+03	21	42	7
Fit1p	9.1463780924E+03	9.1463780920E+03	45	90	8
Kb2	-1.7499001299E+03	-1.7499001299E+03	20	40	2
Standgub	-	1.2576995000E+03	20	40	9
Standmps	1.4060175000E+03	1.4060175000E+03	45	90	9
Standata	1.2576995000E+03	1.2576994988E+03	21	42	9
Czprob	2.1851966989E+06	2.1851966989E+06	118	236	4
D6cube	3.1549166667E+02	3.1549166404E+02	28	56	2
Pilot4	-2.5811392641E+03	-2.5811393097E+03	70	140	9
Pilotnov	-4.4972761882E+03	-4.4972772003E+03	46	92	36

4.3.1 Resultados Computacionais FDIPA LP vs FDIPA sem Perturbação.

Herskovits [31] comentou que o FDIPA original resolve problemas de LP, usando apenas o sistema (3.5). Por outro lado, Tits [52], desenvolve FDIPA sem pertubação (onde a direção é d_α) para LP, mostrando a convergência global e a velocidade de convergência. Já no Celis e Herskovits [11] mostra que alguns resultados computacionais usando a direção com perturbação d tem melhor desempenho. Em teoria, ambos os algoritmos funcionam, vamos comparar a eficiência de ambos.

A Tabela 4.6 mostra os resultados computacionais obtidos. Seguindo [31], [29] foram considerados os seguintes parâmetros $\xi = 0,7$, $\varphi = 1$, $\mu = 10^{-3}$, $\gamma = 0,9995$, $x^s = 10^{15}$. Em [18] foi mostrado que o algoritmo FDIPA funcionou bem para outros parâmetros.

A primeira coluna da Tabela 4.6 contém o nome do problema. As próximas duas colunas mostram o número de iterações que o FDIPA-LP realiza para obter a solução, com seu respectivo *gap*. As colunas 4 e 5 contêm o número de iteração do FDIPA resolvendo o primeiro sistema não perturbado com sua respectiva *gap*.

Tabela 4.6: Testes numéricos iterações FDIPA

Problem	Iter d	Gap d	Iter d_α	Gap d_α
Adlittle	23	0,00E+00	45	1,93E+01
Afiro	13	0,00E+00	88	2,27E-04
Agg2	31	0,00E+00	58	1,50E-02
Agg3	36	0,00E+00	721	4,71E+00
Blend	21	0,00E+00	63	8,06E-06
Degen2	19	0,00E+00	18	4,83E-01
Israel	32	8,29E-03	34	1,59E-02
Sc205	30	0,00E+00	235	9,35E-05
Sc50A	13	0,00E+00	40	1,34E-05
Sc50B	11	6,30E+02	42	1,50E-05
Scagr7	22	5,78E-01	111	6,10E-01
Scagr25	32	9,00E-03	55	9,00E-03
Scorpion	34	3,00E-07	82	2,72E-02

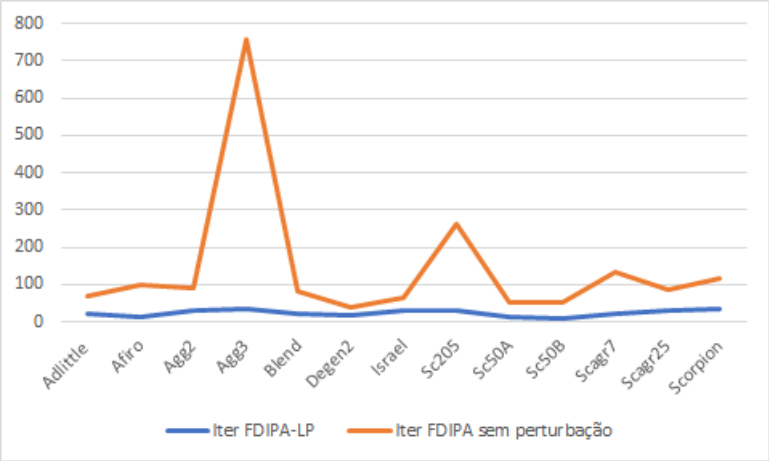


Figura 4.2: Número de iterações FDIPA-LP vs FDIPA sem Perturbação.

Capítulo 5

Programação Semidefinida

Neste capítulo apresentaremos a extensão de um algoritmo de pontos interiores para programação linear (FDIPA-LP) ao problema de programação semidefinida (SDP) linear.

5.1 Introdução

A otimização semidefinida é um ramo da otimização convexa que é de grande interesse teórico e prático. A idéia principal é generalizar a programação linear e os conjuntos viáveis associados (poliedros) para o caso em que as variáveis de decisão são matrizes simétricas e as desigualdades devem ser entendidas como matrizes semidefinidas positivas.

Como é atualmente sabido, muitos elementos da teoria de algoritmos de pontos interiores para LP são estendidos de uma forma bastante natural a programação semidefinida (SDP). Muitos autores desenvolveram algoritmos para resolver problemas de programação semidefinida linear entre eles temos os trabalhos de Babynine Zhadan [7]. Tem aplicações em otimização combinatória, teoria de controle e otimização estrutural. Em 2012, apareceu um manual sobre otimização semidefinida, cônica e polinomial, editado por Anjos e Lasserre em [2].

Em 2014, Aroztegui [6] descreve e analisa o algoritmo de pontos interiores e direções viáveis para o problema de programação semidefinida não linear (FDIPA-SDP), onde os sistemas de equações lineares internos que tem que resolver o algoritmo são reformulados com o produto simétrico de Kronecker. Logo, com a finalidade de preservar a estrutura das restrições matriciais semidefinidas, Roche [49] (FDIPA-SDP-NS) faz a mudança do produto simétrico de Kronecker pelo produto não simétrico tornado-se, desta maneira, computacionalmente mais eficiente.

Nós queremos estender o algoritmo FDIPA LP para o problema de programação semidefinida usando as ideas de FDIPA-SDP-NS [49], formulado da seguinte maneira:

$$\begin{array}{ll} \underset{x}{\text{minimização}} & f(x) \\ \text{sujeito a} & \mathcal{A}(x) \preceq 0, \end{array} \tag{5.1}$$

onde $x \in \mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}$ é a função objetivo e $\mathcal{A} : \mathbb{R}^n \to S^m$ é um mapeamento de $\mathbb{R}^n$ no espaço S^m das matrizes simétricas de ordem $m \times m$. A função $f(x) = c^T x$ com $c \in \mathbb{R}^n$.

O conjunto viável do problema (5.1) é

$$\begin{aligned} \Omega &= \{x \in R^n : \mathcal{A}(x) \leq 0\} \\ &= \{x \in R^n : A_0 + \sum_{i=1}^{n} A_i x_i \leq 0 \text{ para todo } i \in I\}, \end{aligned}$$

o conjunto estritamente viável é

$$int(\Omega) = \{x \in R^n : A_0 + \sum_{i=1}^{n} A_i x_i < 0 \text{ para todo } i \in I\}.$$

A função de Lagrange $L : R^n \times \mathcal{S}^m \to R$ do problema (5.1) é

$$L(x, \Lambda) = c^t x + \langle \Lambda, \mathcal{A}(x) \rangle,$$

onde $\Lambda \in \mathcal{S}^m$ e $\Lambda \succeq 0$.

O gradiente da função de Lagrange em relação a x é denotado por $\nabla L : R^n \times \mathcal{S}^m \to R^n$ e definido como:

$$\nabla_x L(x, \Lambda) = c + \nabla \mathcal{A}(x)\lambda,$$

onde $\lambda = vec(\Lambda)$.

Lema 5.1. *As seguintes propriedades do produto de Kronecker são verdadeiras:*

1. *Qualquer $A \in \mathbb{R}^{p \times q}$, $B \in \mathbb{R}^{r \times s}$ e $C \in \mathbb{R}^{s \times q}$,*

$$(A \otimes B)vec(C) = vec(BCA^T).$$

2. *Qualquer* $A \in \mathbb{R}^{p\times q}$ *e* $B \in \mathbb{R}^{r\times s}$

$$(A \otimes B)^T = (A^T \otimes B^T).$$

3. *Qualquer* $A \in \mathbb{R}^{p\times q}$, $B \in \mathbb{R}^{r\times s}$, $C \in \mathbb{R}^{q\times k}$ *e* $D \in \mathbb{R}^{s\times l}$,

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD),$$

4. *Qualquer* $A \in \mathbb{R}^{p\times p}$ *e* $B \in \mathbb{R}^{r\times r}$,

$$(A \otimes B)^{-1} = (A^{-1} \otimes B^{-1})$$

para A *e* B *invertíveis.*

Demonstração. Em [16]. □

Lema 5.2. *Sejam* $A \in S^m_{++}$ *e* $B \in S^m_-$. *Se elas comutam, então*

1. $y^T ABy = 0$ *se, e somente se,* $By = 0$.

2. $(A \otimes I)^{-1}(B \otimes I) \in S^{m^2}_-$.

Demonstração. Em [6] e [49]. □

A seguinte proposição é uma caracterização das direções viáveis para o problema 5.1.

Proposição 5.1. *Seja a função* $\mathcal{A}$ *diferenciável no ponto* $x \in \Omega$ *e* $\{b_1, ..., b_r\}$ *uma base ortonormal do* $ker(\mathcal{A}(x))$. *Seja também* $E_0 = [b_1, ..., b_r] \in R^{q\times r}$.

Se $E_0^t \Pi E_0 \prec 0$ *então,* $d \in R^n$ *é uma direção viável do problema* (5.1), *onde* $\Pi = \sum_{p=1}^n d_p \frac{\partial \mathcal{A}}{\partial x_p}$.

Demonstração. Em [5]. □

Definição 5.1. *(Ponto regular). Um ponto* x *é um ponto regular do problema* (5.1) *se os vetores* $\nabla\mathcal{A}(x)b_i(x)$, *para* $i = 1, ..., r$, *são linearmente independentes. Neste problema,* $b_1(x), ..., b_r(x) \subset R^{q^2}$ *representa uma base ortonormal do* $ker(A(x) \otimes I_q)$.

A continuação apresentamos uma caracterização das condições de otimalidade KKT proposto em [51], para o problema (5.1) que no futuro será utilizada no calculo na direção do FDIPA para SDP linear.

Ponto $\lambda^* \in \Omega$ é solução de (5.1) se somente se existe $x^* \in \mathbb{R}^n$ para o qual valem as seguintes condições de otimalidade de primeira ordem de KKT:

$$\begin{aligned} c + \nabla\mathcal{A}(x^*)\lambda^* &= 0 && (5.2)\\ \Lambda^*\mathcal{A}(x^*) &= 0 && (5.3)\\ \mathcal{A}(x^*) &\preceq 0 && (5.4)\\ \Lambda^* &\succeq 0, && (5.5) \end{aligned}$$

de (5.4) e (5.5) a condição de complementariedade (5.3) pode ser reescrita como:

$$\Lambda^*\mathcal{A}(x^*) = tr(\Lambda^*\mathcal{A}(x^*)) = 0, \tag{5.6}$$

ademais, na condição (5.3) implica que $\Lambda^* \succeq 0$ e $\mathcal{A}(x^*) \preceq 0$ possui os mesmos autovetores [51].

Seja

$$\Omega_{\mathtt{p}} = \{x \in \Omega |\ \ f(x) \leq \mathtt{p}\},$$

onde $\mathtt{p}$ é um número real. O conjunto interior de $\Omega_{\mathtt{p}}$ é definido por $int(\Omega_{\mathtt{p}}) = \{x \in int(\Omega) |\ \ f(x) < \mathtt{p}\}$.

Assumem-se as seguintes hipóteses.

Hipótese 5.1. *As matrizes A_i $(i = 1, ..., n)$ são linearmente independentes.*

Hipótese 5.2. *Existe um número real $\mathtt{p}$ de tal forma que $\Omega_{\mathtt{p}}$ é compacto e possui $int(\Omega_{\mathtt{p}}) \neq 0$.*

Hipótese 5.3. *Para qualquer ponto $x \in int(\Omega_{\mathtt{p}})$ verifica-se que $\mathcal{A}(x) \prec 0$.*

5.2 Direção de Descida e Viável

Vamos reafirmar as condições de otimalidade (5.2) e (5.3), por meio de um mapeamento $F : \mathbb{R}^{n+m^2} \to \mathbb{R}^{n+m^2}$ e definindo o vetor $(x, \lambda)^T$.

$$\begin{aligned} F(x,\lambda) = \begin{bmatrix} c + \nabla\mathcal{A}(x)\lambda \\ vec(\Lambda\mathcal{A}(x)) \end{bmatrix} &= 0, && (5.7)\\ \mathcal{A}(x) &\preceq 0, && (5.8)\\ \Lambda &\succeq 0. && (5.9) \end{aligned}$$

Podemos encontrar a solução das condições de KKT aplicando o método de Newton ao sistema (5.7). Primeiro, calcula-se o gradiente da função vetorial $F(x, \lambda)$

em relação às variáveis x e λ denotado por $\nabla F(x,\lambda) \in \mathbb{R}^{(n+m^2)\times(n+m^2)}$ e definido como

$$\nabla F(x,\lambda) = \begin{bmatrix} \nabla_x F_1(x,\lambda) & \nabla_\lambda F_1(x,\lambda) \\ \nabla_x F_2(x,\lambda) & \nabla_\lambda F_2(x,\lambda) \end{bmatrix},$$

sendo

$$F_1(x,\lambda) = c + \nabla\mathcal{A}(x)\lambda, \tag{5.10}$$
$$F_2(x,\lambda) = \Lambda\mathcal{A}(x). \tag{5.11}$$

Derivando a Equação (5.10) em relação às variáveis x e λ, resulta

$$\nabla_x F_1(x,\lambda) = 0, \tag{5.12}$$
$$\nabla_\lambda F_1(x,\lambda) = \nabla\mathcal{A}(x). \tag{5.13}$$

Para derivar a Equação (5.11) em relação às variáveis x e λ é necessário estabelecer a seguinte relação empregando a definição do mapeamento vec:

$$vec(\Lambda\mathcal{A}(x)I_m) = vec(I_m\Lambda\mathcal{A}(x)).$$

Aplicando as propriedades do Produto de Kronecker do Lema 5.1 na equação acima temos as seguintes expressões:

$$F_2(x,\lambda) = vec(\Lambda\mathcal{A}(x)I_m) = [I_m \otimes \Lambda]vec(\mathcal{A}(x)),$$
$$F_2(x,\lambda) = vec(I_m\Lambda\mathcal{A}(x)) = [\mathcal{A}(x) \otimes I_m]vec(\Lambda).$$

Derivando as equações em relação às variáveis x e λ, obtém-se as seguintes expressões

$$\nabla_x F_2(x,\lambda) = [I_m \otimes \Lambda]\nabla\mathcal{A}(x)^T,$$
$$\nabla_\lambda F_2(x,\lambda) = \mathcal{A}(x) \otimes I_m.$$

Desta maneira, obtém-se

$$\nabla F(x,\Lambda) = \begin{bmatrix} 0 & \nabla\mathcal{A}(x) \\ [I_m \otimes \Lambda]\nabla\mathcal{A}(x)^t & \mathcal{A}(x) \otimes I_m \end{bmatrix},$$

logo, provaremos que $\nabla F(x,\Lambda)$ é uma matriz inversível.

Seja o ponto inicial (λ^0, x^0), então as iterações do método de Newton são dadas por

$$\nabla F(x^k, \lambda^k) \begin{bmatrix} x^{k+1} - x^k \\ \lambda^{k+1} - \lambda^k \end{bmatrix} = -F(x^k, \lambda^k). \tag{5.14}$$

Define-se a direção dual $d_\alpha^k = x^{k+1} - x^k$, $\lambda_\alpha^k = \lambda^{k+1}$. Assim, o sistema linear (5.14) nas variáveis d_α^k e λ_α^k é escrito como

$$\begin{bmatrix} 0 & \nabla \mathcal{A}(x^k) \\ [I_m \otimes \Lambda^k] \nabla \mathcal{A}(x^k)^T & \mathcal{A}(x^k) \otimes I_m \end{bmatrix} \begin{bmatrix} d_\alpha^k \\ \lambda_\alpha^k \end{bmatrix} = \begin{bmatrix} -c \\ 0 \end{bmatrix}, \tag{5.15}$$

sendo $\lambda^k = vec(\Lambda^k)$ e $\lambda_\alpha^k = vec(\Lambda_\alpha^k)$.

A solução do sistema (5.15) fornece uma direção d_α^k no espaço primal e uma nova estimativa dos multiplicadores de Lagrange λ_α^k.

Se provará neste trabalho no caso $d_\alpha \neq 0$, que a direção d_α^k obtida do sistema de equações (5.15) é uma direção de descida da função $f(x)$. Porém, d_α^k pode não ser uma direção viável, se um ponto está na fronteira da região viável Ω.

Observação 5.1. *Para verificar que d_α^k não é viável, partimos de que existe uma base ortonormal do $ker A(x^k)$ definida pelo conjunto $\{b_1, ..., b_r\}$.*

Seja a matriz $E_0 = [b_1, ..., b_r] \in \mathbb{R}^{m \times r}$, logo, empregando a segunda equação do sistema (5.15) *e aplicando a definição do mapeamento vec, obtêm-se*

$$[I_m \otimes \Lambda^k] vec\left(D\mathcal{A}(x^k) d_\alpha^k\right) + [\mathcal{A}(x^k) \otimes I_m] vec(\Lambda_\alpha^k) = 0, \tag{5.16}$$

$$vec\left(D\mathcal{A}(x^k) d_\alpha^k + \Lambda_\alpha^k A(x^k)\right) = 0. \tag{5.17}$$

Aplicando a definição de smat à Equação (5.17)*, obtém-se*

$$\Lambda^k D\mathcal{A}(x^k) d_\alpha^k + \Lambda_\alpha^k A(x^k) = 0 \tag{5.18}$$

multiplicando ambos os termos da Equação (5.18) *pela esquerda E_0^T e pela direita E_0 resulta em*

$$E_0^T \Lambda^k D\mathcal{A}(x^k) d_\alpha^k E_0 + E_0^T \Lambda_\alpha^k A(x^k) E_0 = 0. \tag{5.19}$$

Como $E_0^T \Lambda_\alpha^k A(x^k) E_0 = 0$, então

$$E_0^T \Lambda^k D\mathcal{A}(x^k) d_\alpha^k E_0 = 0. \tag{5.20}$$

Assumindo que $\Lambda^k = \mu I_m \succ 0$*, obtém-se*

$$E_0^T D\mathcal{A}(x^k) d_\alpha^k E_0 = 0. \tag{5.21}$$

Portanto, pela Proposição (5.1)*, pode-se concluir que a Equação* (5.21) *não garante que a direção* d_α^k *seja viável.*

Com a finalidade de obter uma direção viável, Herskovits propõe uma pequena deflexão na direção d_α^k de tal maneira que esta nova direção fique sempre dentro da região viável. Esta deflexão é feita adicionando um termo auxiliar no lado direito da segunda equação do sistema (5.15), conservando a mesma matriz $\nabla F(x^k, \lambda^k)$ deste sistema

$$\begin{bmatrix} 0 & \nabla\mathcal{A}(x^k) \\ [I_m \otimes \Lambda^k]\nabla\mathcal{A}(x^k)^T & \mathcal{A}(x^k) \otimes I_m \end{bmatrix} \begin{bmatrix} d^k \\ \bar{\lambda}^k \end{bmatrix} = \begin{bmatrix} -c \\ -\rho^k \lambda^k \end{bmatrix}. \tag{5.22}$$

O calculo de d^k e $\bar{\lambda}^k$ pode ser realizado resolvendo um sistema auxiliar nas variáveis d_β^k e λ_β^k:

$$\begin{bmatrix} 0 & \nabla\mathcal{A}(x^k) \\ [I_m \otimes \Lambda^k]\nabla\mathcal{A}(x^k)^T & \mathcal{A}(x^k) \otimes I_m \end{bmatrix} \begin{bmatrix} d_\beta^k \\ \lambda_\beta^k \end{bmatrix} = \begin{bmatrix} 0 \\ -\rho^k \end{bmatrix}. \tag{5.23}$$

Neste caso

$$d^k = d_\alpha^k + \rho^k d_\beta^k, \tag{5.24}$$

$$\bar{\lambda}^k = \lambda_\alpha^k + \rho^k \lambda_\beta^k, \tag{5.25}$$

onde, d_α^k é uma direção viável e de descida da função objetivo no ponto x^k, então é necessário estabelecer os limites deste parâmetro que assegurem que a direção d^k seja também uma direção de descida. Isto é possível definindo um novo parâmetro $\xi \in (0, 1)$ e impondo que

$${d^k}^T \nabla f(x^k) \leq \xi {d_\alpha^k}^T \nabla f(x^k). \tag{5.26}$$

Como ${d_\alpha^k}^T \nabla f(x^k) < 0$ isto implica que ${d^k}^T \nabla f(x^k) < 0$ e, portanto, garante que ${d^k}^T$ é uma direção de descida.

Substituindo a Equação (5.24) na Equação (5.26), pode-se escrever

$${d_\alpha^k}^T \nabla f(x^k) + \rho^k {d_\beta^k}^T \nabla f(x^k) \leq \xi {d_\alpha^k}^T \nabla f(x^k). \tag{5.27}$$

Pode-se definir a cota superior de ρ da forma

$$\rho^k \leq (\xi - 1)\frac{{d_\alpha^k}^T \nabla f(x^k)}{{d_\beta^k}^T \nabla f(x^k)}, \tag{5.28}$$

se ${d_\beta^k}^T \nabla f(x^k) > 0$. Caso contrário, assume-se que

$$\rho^k \leq \varphi \|d_\alpha^k\|^2, \tag{5.29}$$

para algum parâmetro fixo $\varphi > 0$.

5.3 Busca Linear

É possível procurar um passo $t^k > 0$ na direção d^k, tal que o novo ponto $x^{k+1} = x^k + t^k d^k$ seja viável. Realiza-se uma minimização inexata de Armijo [36] ao longo da direção d^k de maneira que $x^{k+1} \in int(\Omega)$ e proporcione um decréscimo da função objetivo.

A busca linear de Armijo consiste em encontrar um comprimento de passo t, ou seja, o primeiro elemento da sequência $\{1, \nu, \nu^2, \nu^3, ...\}$, que reduza o valor da função objetivo e mantenha a viabilidade das restrições, tal que

$$f(x^k + t^k d^k) \leq f(x^k) + t^k \eta \nabla f(x^k)^T d^k \tag{5.30}$$

e

$$\mathcal{A}(x^k + t^k d^k) \prec 0, \tag{5.31}$$

onde $\eta \in (0, 1)$ e $\nu \in (0, 1)$ são parâmetros fixos.

5.4 Algoritmo FDIPA-SDP-NS-L

Parâmetros de controle: $\xi \in (0, 1)$, $\eta \in (0, 1)$, $\varphi > 0$, e $\nu \in (0, 1)$.

Dados Inicias: $x \in int(\Omega)$ e $\Lambda \in S_{++}^m$ comutando com $\mathcal{A}(x)$.

Passo 1: Cálculo da direção de busca.

(i) Resolva o sistema linear para obter $d_\alpha \in \mathbb{R}^n$ e $\lambda_\alpha \in \mathbb{R}^{m^2}$.

$$\nabla \mathcal{A}(x^k)\lambda_\alpha^k = -c, \tag{5.32}$$

$$[I_m \otimes \Lambda^k]\nabla \mathcal{A}(x^k)^T d_\alpha^k + [\mathcal{A}(x^k) \otimes I_m]\lambda_\alpha^k = 0. \tag{5.33}$$

Se $d_\alpha = 0$, parar.

(ii) Resolva o sistema linear para obter $d_\beta \in \mathbb{R}^n$ e $\lambda_\beta \in \mathbb{R}^m$ tal que

$$\nabla \mathcal{A}(x^k)\lambda_\beta^k = 0, \tag{5.34}$$

$$[I_m \otimes \Lambda^k]\nabla \mathcal{A}(x^k)^T d_\beta^k + [\mathcal{A}(x^k) \otimes I_m]\lambda_\beta^k = -\lambda. \tag{5.35}$$

(iii) Se ${d_\beta}^T c > 0$, calcule

$$\rho = min\left\{\varphi\|d_\alpha\|^2, (\xi - 1)\frac{{d_\alpha}^T c}{{d_\beta}^T c}\right\}. \tag{5.36}$$

Caso contrário, calcule

$$\rho = \varphi\|d_\alpha\|^2. \tag{5.37}$$

(iv) Calcule a direção de busca d e $\bar{\lambda}$

$$d = d_\alpha + \rho d_\beta, \tag{5.38}$$

$$\bar{\lambda} = \lambda_\alpha + \rho\lambda_\beta. \tag{5.39}$$

Passo 2: Busca Linear Calcule o comprimento do passo t, como o primeiro número da série $\{1, \nu, \nu^2, \nu^3, ...\}$ tal que:

$$f(x + td) \leq f(x) + t\eta\nabla f(x)^T d \tag{5.40}$$

e

$$\mathcal{A}(x + td) \prec 0. \tag{5.41}$$

Passo 3: Atualizações

(i) Obtenha o novo ponto, como: $x = x + td$;

(ii) Defina novos valores para $\Lambda \in S_{++}^m$ comutando com $\mathcal{A}(x)$;

(iii) Retorne ao **Passo 1**.

O Lema a seguir mostra que os sistemas (5.15) e (5.23) possuem soluções, portanto o **Passo 1** do algoritmo está bem definido.

Lema 5.3. *Assume-se que $x \in \Omega_p$ é um ponto regular do problema* (5.1)*, $\Lambda \in S^m_{++}$, tal que Λ e $\mathcal{A}(x)$ comutam. Então, a seguinte matriz*

$$N(x,\Lambda) = \begin{bmatrix} 0 & \nabla\mathcal{A}(x) \\ [I_m \otimes \Lambda]\nabla\mathcal{A}(x)^T & \mathcal{A}(x) \otimes I_m \end{bmatrix} \tag{5.42}$$

é não singular.

Demonstração. Demostraremos que, se para algum $(d, \bar{\lambda}) \in \mathbb{R}^n \times \mathbb{R}^{m^2}$, a solução do sistema $N(x,\Lambda)\begin{pmatrix} d \\ \bar{\lambda} \end{pmatrix} = 0$ admite somente a solução trivial; ou seja, $(d, \bar{\lambda}) = 0$. Assim, tem-se o seguinte sistema

$$\nabla\mathcal{A}(x)\bar{\lambda} = 0, \tag{5.43}$$

$$[I_m \otimes \Lambda]\nabla\mathcal{A}(x)^T d + [\mathcal{A}(x) \otimes I_m]\bar{\lambda} = 0. \tag{5.44}$$

Multiplicando na Equação (5.43) por d^T

$$d^T\nabla\mathcal{A}(x)\bar{\lambda} = 0. \tag{5.45}$$

Dado $\Lambda \succ 0$, multiplicando na Equação (5.44) por $\bar{\lambda}^k[I_m \otimes \Lambda]^{-1}$, resulta

$$[d^T\nabla\mathcal{A}(x)\bar{\lambda}]^T + \bar{\lambda}^T[I_m \otimes \Lambda]^{-1}[\mathcal{A}(x) \otimes I_m]\bar{\lambda} = 0. \tag{5.46}$$

De (5.45) e (5.46) temos que

$$\bar{\lambda}^T[I_m \otimes \Lambda]^{-1}[\mathcal{A}(x) \otimes I_m]\bar{\lambda} = 0. \tag{5.47}$$

Do Lema 5.2 e da Equação (5.47), temos

$$[\mathcal{A}(x) \otimes I_m]\bar{\lambda} = 0. \tag{5.48}$$

Dado que x é um ponto regular, então $\bar{\lambda} \in ker[\mathcal{A}(x) \otimes I_m]$ e $b_1, ..., b_r$ é uma base ortonormal de $ker[\mathcal{A}(x) \otimes I_m]$ portanto

$$\bar{\lambda} = \sum_{i=1}^{r} \alpha_i b_i. \tag{5.49}$$

Substituindo a Equação (5.49) em (5.43) temos que

$$0 = \nabla \mathcal{A}(x)\bar{\lambda} \tag{5.50}$$

$$= \nabla \mathcal{A}(x) \sum_{i=1}^{r} \alpha_i b_i \tag{5.51}$$

$$= \sum_{i=1}^{r} \alpha_i \nabla \mathcal{A}(x) b_i, \tag{5.52}$$

onde os vetores $\nabla \mathcal{A}(x)b_i$ são linearmente independentes, então $\alpha_i = 0$ para $i = 1, ..., r$. Desta maneira em (5.49) tem-se que $\bar{\lambda} = 0$.

Finalmente, substituindo $\bar{\lambda}$ a Equação (5.44)

$$\nabla \mathcal{A}(x)^T d = 0, \tag{5.53}$$

ou seja

$$\Big[vec(A_1) \quad \dots \quad vec(A_n) \Big] \, d = 0. \tag{5.54}$$

Da **Hipótese 5.1** observe que a independência linear de A_i é equivalente à independência linear de $vec(A_i)$.

Assim, tem-se que $d = 0$. Portanto, a solução do sistema é $(d, \bar{\lambda}) = (0, 0)$. □

Consequentemente, d_α, λ_α, d_β, λ_β obtidos no **Passo 1** do Algoritmo 5.4 estão bem definidos e são únicos.

Note que, se o Algoritmo 5.4 termina na iteração $k \in \mathbb{N}$, $d_\alpha^k = 0$ então x^k é solução da Equação (5.33). Da **Hipótese 5.3**, como o algoritmo é viável em cada iteração, então $\mathcal{A}(x^k) \prec 0$, sendo a matriz $[\mathcal{A}(x^k) \otimes I_m]$ não singular o que traz $\lambda_\alpha^k = 0$. Consequentemente, $\Lambda_\alpha^k = smat(\lambda_\alpha^k) = 0$ e $c = 0$. Portanto, quando $c \neq 0$, o algoritmo gera uma sequência infinita de λ_α^k. De agora em diante, estudamos o caso $d_\alpha^k \neq 0$ em cada iteração. As iterações do algoritmo FDIPA-LP geram $x^k \in int(\Omega)$ e $\Lambda^k \succ 0$. O lema a seguir mostra que d é uma direção de descida.

Lema 5.4. *Seja $\xi \in (0, 1)$, se $d \neq 0$ e $d_\alpha \neq 0$, então*

$$d^T c \leq \xi d_\alpha^T c < 0. \tag{5.55}$$

Demonstração. Do primeiro sistema do algoritmo, multiplica-se na Equação (5.32)

por d_α^T

$$d_\alpha^T \nabla \mathcal{A}(x)\lambda_\alpha = -d_\alpha^T c. \tag{5.56}$$

Dado $\Lambda \in S_{++}^m$, multiplicando na Equação (5.33) por $\lambda_\alpha^T[I_m \otimes \Lambda]^{-1}$ resulta

$$[d_\alpha^T \nabla \mathcal{A}(x)\lambda_\alpha]^T + \lambda_\alpha^T[I_m \otimes \Lambda]^{-1}[\mathcal{A}(x) \otimes I_m]\lambda_\alpha = 0. \tag{5.57}$$

Substituindo (5.56) em (5.57)

$$-c^T d_\alpha + \lambda_\alpha^T[I_m \otimes \Lambda]^{-1}[\mathcal{A}(x) \otimes I_m]\lambda_\alpha = 0 \tag{5.58}$$

$$\lambda_\alpha^T[I_m \otimes \Lambda]^{-1}[\mathcal{A}(x) \otimes I_m]\lambda_\alpha = c^T d_\alpha. \tag{5.59}$$

Assim, $\mathcal{A}(x) \prec 0$ e pelo Lema 2.2 tem-se $[I_m \otimes \Lambda]^{-1}[\mathcal{A}(x) \otimes I_m] \prec 0$. Portanto

$$d_\alpha^T c < 0. \tag{5.60}$$

Alem disso, multiplicamos na Equação (5.38) por c temos que

$$d^T c = d_\alpha^T c + \rho d_\beta^T c, \tag{5.61}$$

do **Passo 1 (iii)** no algoritmo é claro que $\rho > 0$ nos dois casos. Alem disso:

- Se $d_\beta^T c \leq 0$, das equações (5.61) e (5.60) temos que

$$d^T c \leq d_\alpha^T c < 0. \tag{5.62}$$

- Se $d_\beta^T c > 0$, da Equação (5.36) tem-se que

$$\begin{aligned} \rho &\leq (\xi - 1)\frac{d_\alpha^T c}{d_\beta^T c}, \\ d_\alpha^T c + \rho d_\beta^T c &\leq \xi d_\alpha^T c, \\ d^T c &\leq \xi d_\alpha^T c. \end{aligned} \tag{5.63}$$

Portanto, das Equações (5.61) e (5.63),

$$d^T c \leq \xi d_\alpha^T c < 0. \tag{5.64}$$

□

5.5 Resultados Numéricos

Nesta seção, relatamos instâncias de resultados experimentais da coleção SDPLIB. É o conjunto de dados de teste padrão para programação semidefinida linear.

FDIPA-SDP-NS-L precisa como entrada um ponto x^0 dentro da região viável, mas a biblioteca SDPLIB não fornece pontos viáveis para seus problemas. Portanto, busca-se uma solução viável $x^0 \in int(\Omega)$ resolvendo com FDIPA-SDP-NS-L o problema auxiliar

$$\begin{array}{ll} \underset{x,z}{\text{minimização}} & z \\ \text{sujeito a} & \mathcal{A}(x) - zI \preceq 0, \end{array} \tag{5.65}$$

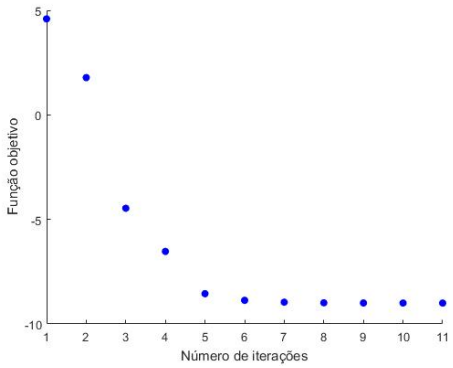
o processo de otimização é interrompido quando a função objetivo atinge um valor negativo z^0.

Na Tabela 5.1, as sete colunas fornecem, respectivamente, o nome do problema, o tamanho da matriz simétrica, o número da variáveis, o valor objetivo, o valor objetivo FDIPA, o número de iterações necessárias para a convergência, o número de iterações necessárias para calcular o ponto inicial.

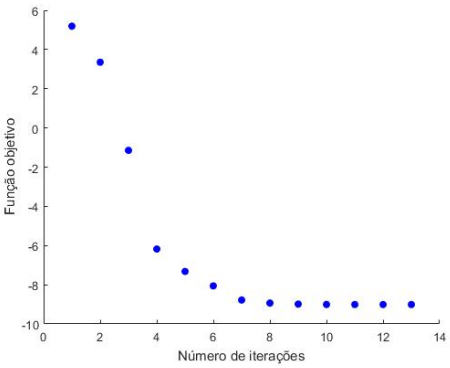
Na Figura 5.1 mostra a evolução da redução do valor objetivo. Note que a função objetivo nas primeiras iterações decresce rápido, e nas proximidades da solução ótima faz varias iterações pra chegar no ótimo.

Tabela 5.1: Testes numéricos do FDIPA SDP

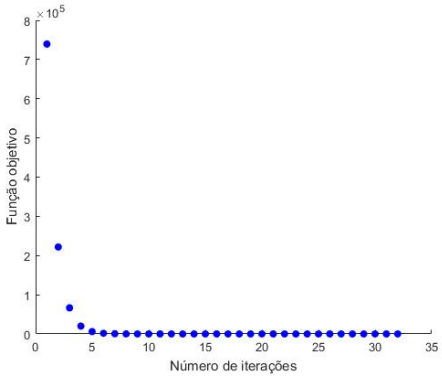
Problemas	m	n	Valor objetivo	Valor objetivo(dual) FDIPA	Iter FDIPA	Iter x^0
Truss1	6	13	-8.999996E+00	-8.999987E+00	11	4
Truss4	12	19	-9.009996E+00	-9.009995E+00	13	5
Theta1	104	50	2.300000E+01	2.300000E+01	32	1
Qap5	136	26	-4.360E+02	-4.359974E+02	19	1
Qap6	229	37	-3.8144E+02	-3.814315E+02	24	1
Qap7	358	50	-4.25E+02	-4.248043E+02	26	1
Qap8	529	65	-7.57E+02	-7.569420E+02	19	1



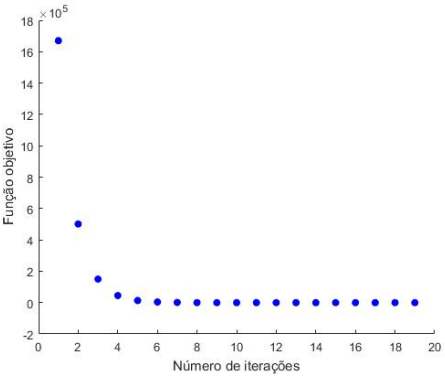
(a) Truss1.



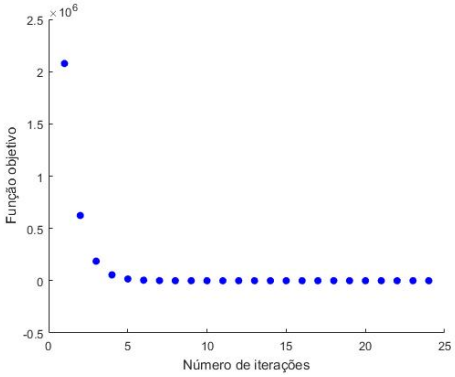
(b) Truss4.



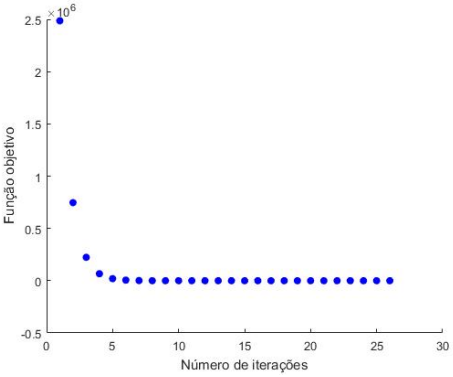
(c) Theta1.



(d) Qap5.



(e) Qap6.



(f) Qap7.

Figura 5.1: Evolução da redução do valor objetivo.

Capítulo 6

Conclusões

Neste trabalho adaptamos um algoritmo de direção viável para programação linear FDIPA-LP, inspirado na iterações de FDIPA, e fornecemos demonstrações matemáticas de que as iterações se aproximam aos pontos de KKT do problema sob suposições razoáveis.

FDIPA-LP começa realizando uma iteração do método de Newton nas condições KKT, gerando um sistema, então perturba-se as condições de complementaridade para que a direções de busca não esteja muito próximas da fronteira e não tenda a ser distorcidas devido a erros de arredondamentos. O sistema linear resolvido em cada iteração é idêntico ao do método de barreira logarítmica primal-dual [39], [35]. No entanto, o FDIPA-LP não é um método de penalidade ou barreira, a estratégia de perturbações do FDIPA-LP é drasticamente diferente.

Resolver os sistemas lineares internos do FDIPA-LP envolve um esforço computacional significativo. Os sistemas lineares a serem resolvidos estão associados a matrizes simétricas definidas positivas. Portanto, usamos o método do Gradiente Conjugado Pré-condicionado *Split* (SPCG) para resolvê-los, juntamente com um pré-condicionador Cholesky incompleto usando a função ICHOL do Matlab. Propomos usar a primeira iteração do gradiente conjugado, pré-solucionador antes de aplicar o algoritmo e algum tipo de escalonamento, que apresenta uma vantagem para reduzir o custo computacional.

Netlib reúne 98 problemas de LP incluindo alguns com restrições de caixa, onde as variáveis pertencem a um intervalo. Por outro lado, para 21 problemas dessa coletânea sem restrições de caixa não foi possível encontrar um ponto inicial com o problema auxiliar (4.5). Portanto, conseguimos testar, com sucesso, 51 problemas com o algoritmo proposto nesta tese. Dentre estes testes, para 37 problemas conseguimos soluções com a precisão na décima casa decimal; para 3 na nona casa

decimal; para 3 na oitava casa decimal; 1 na sétima casa decimal, 4 na sexta casa decimal; 2 na quinta casa decimal e 1 na segunda casa decimal. O fato de todos os testes serem resolvidos com os mesmos parâmetros mostra que o algoritmo é robusto e eficiente, mas também é competitivo quando comparado aos métodos mais avançados.

A segunda parte do trabalho é apresentada da extensão do algoritmo FDIPA-LP a programação semidefinida linear. Em particular, fazemos a formulação dos novos sistemas de Newton e foi resolvido do mesmo jeito que os sistemas internos do FDIPA-LP, assim conseguimos uma boa direção de descida e viável.

Referências Bibliográficas

[1] ALIZADEH, F., HAEBERLY, J.-P. A., OVERTON, M. L., 1998, "Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results", *SIAM Journal on Optimization*, v. 8, n. 3, pp. 746–768.

[2] ANJOS, M. F., LASSERRE, J. B., 2012, "Handbook on semidefinite, conic and polynomial optimization", .

[3] ANSTREICHER, K. M., 1989, "A combined phase I-phase II projective algorithm for linear programming", *Mathematical Programming*, v. 43, n. 1-3, pp. 209–223.

[4] ANSTREICHER, K. M., 1986, "A monotonic projective algorithm for fractional linear programming", *Algorithmica*, v. 1, n. 1-4, pp. 483–498.

[5] AROZTEGUI, M., 2010, *Técnicas de programação semidefinida e aplicações emotimização de material*. Tese de Doutorado, COPPE/UFRJ, Rio de Janeiro,RJ, Brasil,.

[6] AROZTEGUI, M., HERSKOVITS, J., ROCHE, J. R., et al., 2014, "A feasible direction interior point algorithm for nonlinear semidefinite programming", *Structural and Multidisciplinary Optimization*, v. 50, n. 6, pp. 1019–1035.

[7] BABYNIN, M., ZHADAN, V. G., 2008, "A primal interior point method for the linear semidefinite programming problem", *Computational Mathematics and Mathematical Physics*, v. 48, n. 10, pp. 1746.

[8] BARNES, E. R., 1986, "A variation on Karmarkar's algorithm for solving linear programming problems", *Mathematical programming*, v. 36, n. 2, pp. 174–182.

[9] BYRD, R. H., GILBERT, J. C., NOCEDAL, J., 2000, "A trust region method based on interior point techniques for nonlinear programming", *Mathematical programming*, v. 89, n. 1, pp. 149–185.

[10] CAVALIER, T., SOYSTER, A., 1985, *Some computational experience and a modification of the Karmarkar algorithm*. Pennsylvania State Univ., College of Engineering, Department of Industrial and Management Systems Engineering.

[11] CELIS, A. M. V., 2022, "Linear Programming with a Feasible Direction Interior Point Technique for Smooth Optimization", *RAIRO - Operations Research*.

[12] CELIS, A. M. V., 2018, *Algoritmo de ponto interior para programação linear baseado no FDIPA*. Tese de Mestrado, Universidade Federal do Rio de Janeiro.

[13] DANTZIG, G. B., 1951, "Maximization of a linear function of variables subject to linear inequalities", *New York*.

[14] DANTZIG, G. B., 1963, "Linear programming and extensions", *Princeton University*.

[15] DANTZIG, G. B., ORDEN, A., WOLFE, P., et al., 1955, "The generalized simplex method for minimizing a linear form under linear inequality restraints", *Pacific Journal of Mathematics*, v. 5, n. 2, pp. 183–195.

[16] DE KLERK, E., 2006, *Aspects of semidefinite programming: interior point algorithms and selected applications*, v. 65. Springer Science & Business Media.

[17] DIKIN, I., 1967, "Iterative solution of problems of linear and quadratic programming". In: *Soviet Mathematics Doklady*, v. 8, pp. 674–675.

[18] FONSECA, E. T., 2015, *FDIPA - Algoritmo de Pontos Interiores e Direções Viáveis para OtimizaçãoNão-Linear Diferenciável: Um Estudo de Parâmetros*. Tese de Mestrado, Universidad Federal Juiz de Fora, Instituto de Ciências Exatas.

[19] GAY, D. M., 1985, "Electronic mail distribution of linear programming test problems", *Mathematical Programming Society COAL Newsletter*, v. 13, pp. 10–12.

[20] GOLUB, G. H., VAN LOAN, C. F., 2012, *Matrix computations*, v. 3. JHU Press.

[21] GONZAGA, C. C., 1989, "An algorithm for solving linear programming problems in $O(n^3L)$ operations". In: *Progress in mathematical programming*, Springer, pp. 1–28, .

[22] GONZAGA, C. C., 1992, "Path-following methods for linear programming", *SIAM review*, v. 34, n. 2, pp. 167–224.

[23] GONZAGA, C. C., 1989, "Conical projection algorithms for linear programming", *Mathematical Programming*, v. 43, n. 1-3, pp. 151–173.

[24] GREENBAUM, A., 1997, *Iterative methods for solving linear systems*, v. 17. Siam.

[25] GUTIERREZ, A. E., MAZORCHE, S. R., HERSKOVITS, J., et al., 2017, "An interior point algorithm for mixed complementarity nonlinear problems", *Journal of Optimization Theory and Applications*, v. 175, n. 2, pp. 432–449.

[26] HALICKÁ, M., DE KLERK, E., ROOS, C., 2002, "On the convergence of the central path in semidefinite optimization", *SIAM Journal on Optimization*, v. 12, n. 4, pp. 1090–1099.

[27] HELMBERG, C., RENDL, F., VANDERBEI, R. J., et al., 1996, "An interior-point method for semidefinite programming", *SIAM Journal on optimization*, v. 6, n. 2, pp. 342–361.

[28] HERSKOVITS, J., LEONTIEV, A., DIAS, G., et al., 2000, "Contact shape optimization: a bilevel programming approach", *Structural and multidisciplinary optimization*, v. 20, n. 3, pp. 214–221.

[29] HERSKOVITS, J., SANTOS, G., 1997, "On the computer implementation of feasible direction interior point algorithms for nonlinear optimization", *Structural optimization*, v. 14, n. 2, pp. 165–172.

[30] HERSKOVITS, J., FREIRE, W. P., TANAKA FO, M., et al., 2011, "A feasible directions method for nonsmooth convex optimization", *Structural and Multidisciplinary Optimization*, v. 44, n. 3, pp. 363–377.

[31] HERSKOVITS, J., 1998, "Feasible direction interior-point technique for nonlinear optimization", *Journal of optimization theory and applications*, v. 99, n. 1, pp. 121–146.

[32] HOFFMAN, A., MANNOS, M., SOKOLOWSKY, D., et al., 1953, "Computational experience in solving linear programs", *Journal of the Society for Industrial and Applied Mathematics*, v. 1, n. 1, pp. 17–33.

[33] KARMARKAR, N., 1984, "A new polynomial-time algorithm for linear programming". In: *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pp. 302–311. ACM.

[34] KLEE, V., 1972, "How good is the simplex method", *Inequalities*, pp. 159–175.

[35] KOJIMA, M., MIZUNO, S., YOSHISE, A., 2012, "Algorithm for linear programming", *Progress in Mathematical Programming: Interior-Point and Related Methods*, p. 29.

[36] LUENBERGER, D., YE, Y., 2008, *Linear and nonlinear programming*. International Series in Operations Research & Management Science. Springer US.

[37] LUSTIG, I. J., 1985, *A practical approach to Karmarkar's algorithm.* Relatório técnico, STANFORD UNIV CA SYSTEMS OPTIMIZATION LAB.

[38] MACULAN, N., FAMPA, M. H. C., 2006, *Otimização linear*. Brasília: EdUnB.

[39] MEGIDDO, N., 1989, "Pathways to the optimal set in linear programming". In: *Progress in mathematical programming*, Springer, pp. 131–158.

[40] MONTEIRO, R. D., 1998, "Polynomial convergence of primal-dual algorithms for semidefinite programming based on the Monteiro and Zhang family of directions", *SIAM Journal on Optimization*, v. 8, n. 3, pp. 797–812.

[41] MONTEIRO, R. D., 1997, "Primal–dual path-following algorithms for semidefinite programming", *SIAM Journal on Optimization*, v. 7, n. 3, pp. 663–678.

[42] MONTEIRO, R. D., ADLER, I., 1990, "An extension of Karmarkar type algorithm to a class of convex separable programming problems with global linear rate of convergence", *Mathematics of Operations Research*, v. 15, n. 3, pp. 408–422.

[43] MONTEIRO, R. D., ADLER, I., 1989, "Interior path following primal-dual algorithms. Part II: Convex quadratic programming", *Mathematical Programming*, v. 44, n. 1-3, pp. 43–66.

[44] NESTEROV, Y., 1994, "Interior point polynomial methods in convex programming", *Theory and Applications*.

[45] NEUMANN, J. V., 1947, "On a maximization problem", *Manuscript, Institute for Advanced Studies, Princeton University, Princeton, NJ*, v. 8544.

[46] NUNEZ, M. A., FREUND, R. M., 2001, "Condition-measure bounds on the behavior of the central trajectory of a semidefinite program", *SIAM Journal on Optimization*, v. 11, n. 3, pp. 818–836.

[47] PLOSKAS, N., SAMARAS, N., 2017, *Linear programming using MATLAB®*, v. 127.

[48] ROBERT, J. V., 2001, *Linear programming: Foundations and extensions*. Boston, Kluwer Academic.

[49] ROCHE, J. R., HERSKOVITS, J., BAZÁN, E., et al., 2017, "A feasible direction algorithm for general nonlinear semidefinite programming", *Structural and Multidisciplinary Optimization*, v. 55, n. 4, pp. 1261–1279.

[50] SAAD, Y., 2003, *Iterative methods for sparse linear systems*, v. 82. SIAM.

[51] SHAPIRO, A., 1997, "First and second order analysis of nonlinear semidefinite programs", *Mathematical programming*, v. 77, n. 1, pp. 301–320.

[52] TITS, A. L., ZHOU, J. L., 1994, "A simple, quadratically convergent interior point algorithm for linear programming and convex quadratic programming". In: Hager, W. W., Hearn, D. W., Pardalos, P. M. (Eds.), *Large Scale Optimization: State of the Art*, pp. 411–427, Boston, MA, Springer US. ISBN: 978-1-4613-3632-7.

[53] WOLKOWICZ, H., SAIGAL, R., VANDENBERGHE, L., 2012, *Handbook of semidefinite programming: theory, algorithms, and applications*, v. 27. Springer Science & Business Media.

[54] WRIGHT, S. J., 1997, *Primal-dual interior-point methods*, v. 54. SIAM.

[55] YE, Y., 1991, "An $O(n^3L)$ potential reduction algorithm for linear programming", *Mathematical programming*, v. 50, n. 1-3, pp. 239–258.