ERRATA nos 2 exemplares da dissertação da aluna Denise dos Santos Trevisoli
Em folha ii:

Onde se lê:	
Área de concentração: Otimização	Desconsidere
Programa de Pós-Graduação: **Mestrado em Matemática Aplicada**	Leia-se: Programa de Pós-Graduação: **Matemática Aplicada**

Prof. Dr. Luiz Koodi Hotta
Coordenador CPG/IMECC
Matric. 042471 - UNICAMP

Universidade Estadual de Campinas

INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E COMPUTAÇÃO CIENTÍFICA

Departamento de Matemática Aplicada

Dissertação de Mestrado

Sobre problemas associados a cones de segunda ordem

por

Denise dos Santos Trevisoli

Mestrado em Matemática Aplicada - Campinas - SP

Orientadora: Profa. Dra. Maria Aparecida Diniz Ehrhardt

Sobre problemas associados a cones de segunda ordem

Este exemplar corresponde à redação
final da dissertação devidamente corrigida
e defendida por Denise dos Santos Trevisoli
e aprovada pela Comissão Julgadora.

Campinas, 27 de maio de 2011

Profa. Dra. Maria Aparecida Diniz Ehrhardt

Orientadora

Banca examinadora:

Profa. Dra. Maria Aparecida Diniz Ehrhardt
Prof. Dr. Aurelio Ribeiro Leite de Oliveira
Prof. Dr. Lucas Garcia Pedroso

Dissertação apresentada ao Instituto
de Matemática, Estatística e Computação
Científica, UNICAMP, como requisito
parcial para obtenção do Título de
MESTRE em Matemática Aplicada.

FICHA CATALOGRÁFICA ELABORADA POR
Maria Fabiana Bezerra Müller – CRB8/6162 – BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E COMPUTAÇÃO CIENTÍFICA - UNICAMP

Trevisoli, Denise dos Santos
T729s Sobre problemas associados a cones de segunda ordem / Denise dos Santos Trevisoli. -- Campinas, SP : [s.n.], 2011.

Orientador: Maria Aparecida Diniz Ehhardt.
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Cone (Matemática). 2. Programação convexa. 3. Programação não linear. I. Ehrhardt, Maria Aparecida Diniz, 1956-. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Informações para Biblioteca Digital

Título em inglês: About problems related to second order cones
Palavras-chave em inglês:
Cone
Convex programming
Nonlinear programming
Área de concentração: Otimização
Titulação: Mestre em Matemática Aplicada
Banca examinadora:
Maria Aparecida Diniz Ehrhardt [Orientador]
Aurelio Ribeiro Leite de Oliveira
Lucas Garcia Pedroso
Data da defesa: 27-05-2011
Programa de Pós-Graduação: Mestrado em Matemática Aplicada

Dissertação de Mestrado defendida em 27 de maio de 2011 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.

Prof.(a). Dr(a). MARIA APARECIDA DINIZ EHRHARDT

Prof.(a). Dr(a). AURELIO RIBEIRO LEITE DE OLIVEIRA

Prof.(a). Dr(a). LUCAS GARCIA PEDROSO

AGRADECIMENTOS

Agradeço:

À professora Cheti, pelo exemplo de caráter e pela orientação feita com muito carinho e atenção.

Ao professor Aurelio, pelo incentivo, pelos conselhos e pelas sugestões no texto.

Ao professor Lucas, pela correção criteriosa de meu texto.

A Deus, por me amar e iluminar permitindo que eu alcance mais do que almejo.

À Nossa Senhora, por interceder por mim diariamente.

Aos meus pais Elcio e Irma, pelo apoio, carinho, paciência e pelas orações. Às minhas irmãs Vanessa e Elaine e aos meus sobrinhos Nicole, Gabriel e Henrique, que alegraram meus dias estressantes.

Ao André que por amor e com amor gastou muitas horas me ajudando e apoiando.

Ao Dom Filipe, Ivete, José Evanil e demais amigos equipistas que acompanharam os sofrimentos e alegrias desta conquista.

Ao José Evangelista, Márcio, Kênia, Cíntia e demais amigos do Imecc por me ajudarem sempre que preciso e pelas risadas diante das dificuldades.

Às pessoas que colaboraram de forma direta ou indireta, por meio de incentivo, e troca de experiências.

À CAPES, pelo apoio financeiro.

"...aquilo que se tem diante dos olhos não é objeto de esperança: como pode alguém esperar o que está vendo? Mas, se esperamos o que não vemos, é porque o aguardamos com perseverança."
Rm 8, 24-25

RESUMO

Este trabalho teve como foco o estudo de problemas SOCP, tanto nos seus aspectos teóricos quanto nos seus aspectos práticos. Problemas SOCP são problemas convexos de otimização nos quais uma função linear é minimizada sobre restrições lineares e restrições de cone quadrático. Tivemos dois objetivos principais: estudar o conceito, as aplicações e os métodos de resolução de problemas SOCP, permitindo verificar a viabilidade de trabalhar com tais problemas; e verificar na prática o benefício de se utilizar uma ferramenta específica de SOCP para a resolução de problemas que se enquadram nessa classe. Para a avaliação prática utilizamos um software de otimização genérica (`fmincon`) e outro específico de SOCP (CVXOPT). A análise ficou concentrada nos requisitos robustez, número de iterações e variação do tempo com o aumento da dimensão dos problemas. Diante dos resultados obtidos com os testes numéricos, pudemos concluir que é interessante usar SOCP sempre que possível.

ABSTRACT

This dissertation focuses on the study of SOCP problems, both in its theoretical, and in its practical aspects. SOCP problems are convex optimization problems in which a linear function is minimized over linear constraints and second-order cone constraints. We had two main objectives: study the concept, applications and methods for solving the SOCP problem, making it possible to verify the feasibility of working with such problems; and to verify the practical benefits of using a SOCP specific tool for the resolution of problems of this class. The experimental evaluation used a generic optimization software (`fmincon`) and other SOCP specific software (CVXOPT). The analysis was concentrated on the robustness, number of iterations and time variation with the increasing scale of the problems. From results obtained with the numerical tests, we concluded that SOCP is worth to be used whenever possible.

SUMÁRIO

1 Introdução **1**

2 Conceitos Fundamentais **4**

2.1 Dualidade 6

2.2 Análise Convexa 9

3 Problemas Associados a Cones de Segunda Ordem **16**

3.1 Introdução 16

3.2 O problema de programação cônica de segunda ordem 17

3.2.1 Restrições SOC 17

3.3 Problemas que podem ser expressos como SOCP 18

3.3.1 Soma de normas 19

3.3.2 Máximo de normas 19

3.3.3 QCQP - *Quadratically Constrained Quadratic Programming* 20

3.3.4 Problemas com restrições hiperbólicas 21

3.3.5 Superfície mínima 23

3.3.6 Programação Linear Robusta 24

3.3.7 Quadrados mínimos robustos 27

4 Método de Pontos Interiores Primal-Dual **31**

4.1 Problema Geral de Programação Não Linear 31

4.1.1 Métodos de Pontos Interiores do tipo Primal-Dual 31

4.1.2 O Caminho Central 33

4.1.3 Método Seguidor de Caminho (*Path-Following Method*) 34

4.1.4 O Método de Barreira 35

4.1.5 Métodos de Redução Potencial 37

4.2 SOCP - Second-Order Cone Programming 38

4.2.1 Dual do SOCP 38

4.2.2 Barreira para Cone Quadrático 39

4.2.3 Função Potencial Primal-Dual 40

4.2.4 Algoritmo Primal-Dual de Redução Potencial 41

4.2.5 Encontrando pontos iniciais estritamente factíveis 42

4.2.6 Método Seguidor de Caminho para SOCP com Escalamento de Nesterov-Todd 43

5 Testes Numéricos **48**

5.1 Software utilizado 49

5.1.1 CVXOPT - *Python Software for Convex Optimization* 49

5.1.2 `fmincon` - *Find Minimum of Constrained Nonlinear Multivariable Function* 50

5.2 Problemas Preliminares 50

5.2.1 Exemplo 1 - Programação Linear 50

5.2.2 Exemplo 2 - Função Objetivo Linear e Restrições Quadráticas 51

5.2.3 Exemplo 3 - Soma de Normas 52

5.2.4 Exemplo 4 - QCQP (*Quadratically Constrained Quadratic Programming*) 53

5.3 Problemas Teóricos 55

5.3.1 QCQP 55

5.4 Problemas Reais 57

5.4.1 Arranjo de Antenas 57
5.4.2 Garras Múltiplos Dedos - *Grasp* 60

6 Considerações Finais **73**

NOTAÇÕES UTILIZADAS

$u^T \in \mathbb{R}^{1\times n}$ denota o transposto do vetor $u \in \mathbb{R}^{n\times 1}$.

$u^T v$ é o produtor escalar canônico em $\mathbb{R}^n$.

$u \times v$ é o produto vetorial entre u e v.

$I \in \mathbb{R}^{n\times n}$ é a matriz identidade de ordem n.

$\nabla f \in \mathbb{R}^n$ é o vetor gradiente da função $f : \mathbb{R}^n \longrightarrow \mathbb{R}$.

$\nabla Z \in \mathbb{R}^{n\times m}$ é a matriz jacobiana da função $Z : \mathbb{R}^n \longrightarrow \mathbb{R}^m$.

$\nabla^2 f \in \mathbb{R}^{n\times n}$ é a matriz hessiana da função $f : \mathbb{R}^n \longrightarrow \mathbb{R}$.

CAPÍTULO 1

INTRODUÇÃO

Programação não linear (PNL) é um campo de pesquisa bastante ativo, com aplicação direta em várias áreas do conhecimento humano. Em problemas de PNL minimizamos ou maximizamos uma função, conhecida por função objetivo, sujeita a restrições de igualdade e desigualdade. Tais problemas se caracterizam por envolverem ao emnos uma função não linear. Isto faz com que sua solução tenda a ser complexa, o que se traduz na prática em processos de otimização computacionalmente caros.

Dentro deste cenário, torna-se importante explorar as propriedades matemáticas de cada classe de problemas, permitindo direcionar o processo de otimização, e tornando-o mais eficiente. Este processo nem sempre é simples, mas pode representar a diferença entre uma otimização ser ou não computacionalmente viável.

Na década de 90, identificou-se que vários tipos de problemas podiam ser formulados como problemas de programação cônica de segunda ordem (SOCP - *Second-Order Cone Programming*). Problemas SOCP são problemas convexos de otimização nos quais uma função linear é minimizada sobre restrições lineares e restrições de cone quadrático.

Tratados como SOCP, muitos problemas, como por exemplo, programação linear robusta, programação quadrática, programação quadrática com restrições quadráticas, entre outros, podem ser otimizados com maior eficiência, estabelecendo um campo de pesquisa bastante

frutífero.

Exemplos específicos de otimização de problemas SOCP são abundantes na literatura científica. Por exemplo, o problema clássico de *Fermat-Weber* (descrito em 3.3.1), vide [32] e Witzgall apud [1]. Lobo et al [12], Boyd et al [5] e Srirangarajan et al [24] expõem diversas aplicações de problemas SOCP em engenharia. Os artigos de Nesterov e Nemirovski [19], Lobo et al [12], Nemirovski e Ben-Tal [18], Boyd e Vandenberghe [28] e Boyd et al [5] mostram que diversos tipos de problemas, entre eles programação linear robusta, quadrados mínimos robustos, problemas com restrições hiperbólicas, mínimo de soma de normas e problemas de controle também podem ser formulados como SOCP.

Devido à importância e aplicabilidade do problema SOCP, podem ser encontrados não apenas estudos científicos, mas também softwares específicos para a sua resolução. São exemplos destes softwares o CVXOPT [27], o SeDuMi [25] e o CVX [9], mantidos principalmente por pesquisadores e disponibilizados de forma livre.

Este trabalho tem como foco problemas SOCP, tanto nos seus aspectos teóricos quanto nos seus aspectos práticos. Como principais objetivos, temos: (i) estudar o conceito, as aplicações e os métodos de resolução de problemas SOCP, permitindo verificar a viabilidade de trabalhar com estes; e (ii) verificar na prática o benefício de se utilizar uma ferramenta específica de SOCP para a resolução de problemas que se enquadram nessa classe.

O estudo teórico foi conduzido tanto com base em material específico de SOCP, quanto em material relativo a áreas relevantes ao estudo destes problemas. As principais áreas estudadas foram programação não linear, análise convexa, problemas SOCP, métodos de pontos interiores, método de barreira (muito comum na resolução de problemas de PNL), método de redução potencial e método seguidor de caminho com escalamento de Nesterov-Todd [21](bastante utilizado na resolução de problemas SOCP).

A avaliação prática utilizou um software de otimização genérica (`fmincon` [15]) e outro específico de SOCP (CVXOPT). Durante o estudo, comparou-se a eficiência de cada ferramenta na otimização tanto de problemas teóricos (QCQP) quanto de problemas práticos (arranjo de antenas e garra com múltiplos dedos).

Esta dissertação está organizada da seguinte forma. No Capítulo 2, estudamos conceitos fundamentais de análise convexa, otimização não linear, dualidade e problemas convexos de

otimização. No Capítulo 3, descrevemos o problema SOCP e realizamos um estudo detalhado da reformulação dos problemas de otimização, programação linear, soma de normas, máximo de normas, programação quadrática, QCQP, problemas com restrições hiperbólicas, superfície mínima, programação linear robusta e quadrados mínimos robustos, em problemas SOCP. Em seguida, no Capítulo 4, é feito o estudo do método de barreira, do método de redução potencial e do método seguidor de caminho com escalamento de Nesterov-Todd. No Capítulo 5, apresentamos um estudo dos problemas práticos, arranjo de antenas e garras múltiplos dedos e diversos testes numéricos, com problemas teóricos e práticos, utilizando `fmincon` e CVXOPT. Por fim, no Capítulo 6 são resumidas as conclusões deste trabalho e são propostos trabalhos futuros dentro da linha de pesquisa.

CAPÍTULO 2

CONCEITOS FUNDAMENTAIS

Consideremos o seguinte problema de programação não linear:

$$\begin{array}{ll} \text{Minimizar} & f(x) \\ \text{sujeita a} & h(x) = 0, \\ & g(x) \leq 0, \end{array} \tag{2.1}$$

onde $f,\ g,\ h \in C^1; f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $h : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, com $m < n$, $g : \mathbb{R}^n \longrightarrow \mathbb{R}^p$. A função a ser minimizada, f, é a função objetivo.

Um ponto x é dito solução factível, ou ponto factível, se pertence ao conjunto $S = \{x \in \mathbb{R}^n \mid g(x) \leq 0, h(x) = 0\}$. Se x pertence ao conjunto $S = \{x \in \mathbb{R}^n \mid g(x) < 0, h(x) = 0\}$, então x é dito ponto estritamente factível.

Um ponto $x^* \in S$ é um minimizador global do problema (2.1) se $f(x^*) \leq f(x)$ para todo $x \in S$. Dizemos que x é minimizador global estrito se $f(x^*) < f(x)$ para todo $x \in S$.

Um ponto $x^* \in S$ é um minimizador local do problema (2.1) se existe uma ϵ-vizinhança em x^* tal que $f(x^*) \leq f(x)$ para todo $x \in S$ nesta vizinhança. E se $f(x^*) < f(x)$ para todo $x \in S$, $x \neq x^*$, nessa vizinhança, então x^* é dito minimizador local estrito.

Quando resolvemos o problema (2.1), nosso objetivo é encontrar seu mínimo global. Entretanto, calcular soluções globais é frequentemente muito difícil, e mesmo reconhecer se

um ponto x factível é um minimizador global pode ser impossível.

Assim, é importante analisar propriedades mais fracas que sejam satisfeitas pelos minimizadores locais do problema. Essas propriedades são denominadas *Condições Necessárias de Otimalidade* e os pontos que as satisfazem se dizem *estacionários*.

Um ponto estacionário, ou ponto KKT (Karush-Kuhn-Tucker) do problema (2.1), é um ponto factível para o qual existem vetores $\lambda \in \mathbb{R}^m$, $\mu \in \mathbb{R}^p$ tais que:

$$\begin{aligned} \nabla f(x) + \sum_{i=1}^{m} \lambda_i \nabla h_i(x) + \sum_{j=1}^{p} \mu_j \nabla g_j(x) = 0, \\ \mu_j g_j(x) = 0, \\ \mu_j \geq 0, \quad j = 1, .., p. \end{aligned} \tag{2.2}$$

Os escalares λ_i e μ_j são chamados de multiplicadores de Lagrange.

No entanto, nem todo minimizador local de um problema de programação não linear é um ponto KKT. Podemos afirmar que um minimizador local é um ponto KKT quando este satisfaz determinadas condições, conhecidas por condições de qualificação [14].

Definição 2.0.1. *Uma condição de qualificação é uma propriedade das restrições do problema* (2.1) *suficiente para garantir que um minimizador local x^* seja um ponto KKT.*

A condição de qualicação mais conhecida é a regularidade ou independência linear dos gradientes das restrições ativas.

Dizemos que uma restrição $g(x) \geq 0$ está ativa num ponto factível $\bar{x}$ se $g(\bar{x}) = 0$, e inativa se $g(\bar{x}) > 0$. Além disso, é importante ressaltar que as restrições de igualdade estão sempre ativas no conjunto factível.

Teorema 2.0.2. *Condições Necessárias de Otimalidade de Primeira Ordem.*

Seja x^* um minimizador local regular do problema (2.1)*. Então existem vetores $\lambda \in \mathbb{R}^m$ e $\mu \in \mathbb{R}^p$, tais que as condições (2.2) são satisfeitas.*

Demonstração. Ver Luenberger [13]. □

2.1 Dualidade

Dado um problema de programação não linear, existe outro problema, também de programação não linear, associado a ele. Estes dois problemas são chamados primal e dual, respectivamente. Existem várias formulações de problema dual, mas para esse trabalho usaremos o dual Lagrangiano, que chamaremos apenas de dual.

Para o estudo da dualidade, chamaremos o problema (2.1) de problema primal.

$$\text{Problema Primal} \quad \begin{array}{ll} \text{Minimizar} & f(x) \\ \text{sujeita a} & h(x) = 0, \\ & g(x) \leq 0. \end{array} \tag{2.3}$$

O dual do problema acima é dado por:

$$\text{Problema Dual} \quad \begin{array}{ll} \text{Maximizar} & \theta(\lambda, \mu) \\ \text{sujeita a} & \mu \geq 0, \end{array} \tag{2.4}$$

onde

$$\theta(\lambda, \mu) = \inf_x \left\{ f(x) + \sum_{i=1}^{m} \lambda_i h_i(x) + \sum_{j=1}^{p} \mu_j g_j(x);\ x \in S \right\}.$$

A função $L(x, \lambda, \mu) = f(x) + \sum_{i=1}^{m} \lambda_i h_i(x) + \sum_{j=1}^{p} \mu_j g_j(x)$ é conhecida por função Lagrangiana.

Teorema 2.1.1. *Teorema Fraco da Dualidade*

Seja x uma solução factível do problema (2.3) e seja (λ, μ) uma solução factível do problema (2.4)*. Então*

$$f(x) \geq \theta(\lambda, \mu).$$

Demonstração. Como x e μ são factíveis, temos que $h(x) = 0$, $g(x) \leq 0$ e $\mu \geq 0$. Logo:

$$\begin{aligned} \theta(\lambda, \mu) &= \inf_y \left\{ f(y) + \mu^T g(y) + \lambda^T h(y) \mid y \in \mathbb{R}^n \right\} \\ &\leq f(x) + \mu^T g(x) + \lambda^T h(x) \\ &\leq f(x). \end{aligned} \tag{2.5}$$

□

Sejam x^* e (λ^*, μ^*) soluções ótimas dos problemas primal e dual respectivamente. Segue do teorema acima que $f(x^*) \geq \theta(\lambda^*, \mu^*)$, isto é, o valor objetivo ótimo primal é maior ou igual ao valor objetivo ótimo dual. Se esses valores objetivos ótimos não forem iguais, dizemos que existe um **gap de dualidade**, que denotaremos por $\eta(x, \lambda, \mu)$, ou simplesmente η:

$$\eta(x, \lambda, \mu) = f(x) - \theta(\lambda, \mu) \tag{2.6}$$

Teorema 2.1.2. *Teorema Forte da Dualidade*

Se existem $\tilde{x}$ e $(\tilde{\lambda}, \tilde{\mu})$ pontos factíveis, respectivamente, do problema primal (2.3)e do problema dual (2.4), tais que

$$\theta(\tilde{\lambda}, \tilde{\mu}) \ = \ f(\tilde{x}),$$

então $\tilde{x}$ é ótimo global primal e $(\tilde{\lambda}, \tilde{\mu})$ é ótimo global dual.

Demonstração. Sejam $\tilde{x}$, solução factível do problema primal (2.3), e $(\tilde{\lambda}, \tilde{\mu})$ solução factível do problema dual (2.4), tais que

$$\theta(\tilde{\lambda}, \tilde{\mu}) \ = \ f(\tilde{x}),$$

Então, pelo Teorema 2.1.1, temos que:

$$\begin{aligned} &f(\tilde{x}) = \theta(\tilde{\lambda}, \tilde{\mu}) \leq f(x), \quad \forall\ x \text{ factível}, \\ &\theta(\tilde{\lambda}, \tilde{\mu}) = f(\tilde{x}) \geq \theta(\lambda, \mu), \quad \forall\ (\lambda, \mu) \text{ factível}. \end{aligned}$$

Logo, concluímos que $\tilde{x}$ é ótimo global primal e $(\tilde{\lambda}, \tilde{\mu})$ é ótimo global dual.

□

Ponto de Sela

Para simplificar o estudo dos pontos de sela, vamos assumir que o problema primal possui apenas restrições de desigualdade. Neste caso, a função Lagrangiana terá a forma:

$$L(x, \mu) = f(x) + \mu^T g(x).$$

Daí,

$$\sup_{\mu \geq 0} \ L(x, \mu) = \sup_{\mu \geq 0} \big(f(x) + \mu^T g(x) \big) = \begin{cases} f(x) & \text{se} \quad g(x) \leq 0, \\ \infty & \text{caso contrário.} \end{cases}$$

Ou seja, se x for factível teremos $\sup_{\mu \geq 0} L(x, \mu) = f(x)$. Isto significa que podemos expressar o valor ótimo do problema primal através da Lagrangiana, da seguinte forma:

$$f(x^*) = \inf_x \sup_{\mu \geq 0} L(x, \mu).$$

Mas, pela definição do problema dual temos:

$$\theta(\mu^*) = \sup_{\mu \geq 0} \inf_x L(x, \mu).$$

Então, a dualidade fraca pode ser expressa pela desigualdade:

$$\inf_x \sup_{\mu \geq 0} L(x, \mu) \geq \sup_{\mu \geq 0} \inf_x L(x, \mu). \tag{2.7}$$

E a dualidade forte pela igualdade:

$$\inf_x \sup_{\mu \geq 0} L(x, \mu) = \sup_{\mu \geq 0} \inf_x L(x, \mu).$$

Portanto, dualidade forte significa que a ordem de minimização sobre x e maximização sobre $\mu \geq 0$ pode ser mudada sem afetar o resultado.

Observe que a desigualdade (2.7) é independente de propriedades da Lagrangiana, já que,

$$\sup_{z \in Z} \inf_{w \in W} u(w, z) \leq \inf_{w \in W} \sup_{z \in Z} u(w, z),$$

para qualquer $u : \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}$, qualquer $W \subseteq \mathbb{R}^n$ e qualquer $Z \subseteq \mathbb{R}^m$. A desigualdade anterior é conhecida por desigualdade max-min e, quando a igualdade ocorre, ou seja, quando existem z e w tais que

$$\inf_{w \in W} \sup_{z \in Z} u(z, w) = \sup_{z \in Z} \inf_{w \in W} u(z, w),$$

dizemos que u, W e Z satisfazem o critério da propriedade forte min-max ou a propriedade do ponto de sela.

Definição 2.1.3. *Chamamos o par $(\tilde{w}, \tilde{z})$ de ponto de sela de u, W, Z se*

$$u(\tilde{w}, z) \leq u(\tilde{w}, \tilde{z}) \leq u(w, \tilde{z}),$$

para todo $w \in W$ e $z \in Z$ [3].

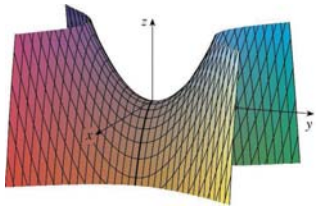


Figura 2.1: ponto de sela

Da definição acima segue que se $(\tilde{w}, \tilde{z})$ é ponto de sela de $u,\ W,\ Z$, então

$$\begin{aligned} u(\tilde{w}, \tilde{z}) &= \inf_{w \in W}\ u(w, \tilde{z}) \\ u(\tilde{w}, \tilde{z}) &= \sup_{z \in Z} u(\tilde{w}, z), \end{aligned}$$

e temos que a propriedade de ponto de sela ocorre, o que nos fornece o teorema:

Teorema 2.1.4. *O ponto (x^*, μ^*) é de sela da Lagrangiana se, e somente se, x^* é ótimo primal e μ^* é ótimo dual e o gap de dualidade é zero.*

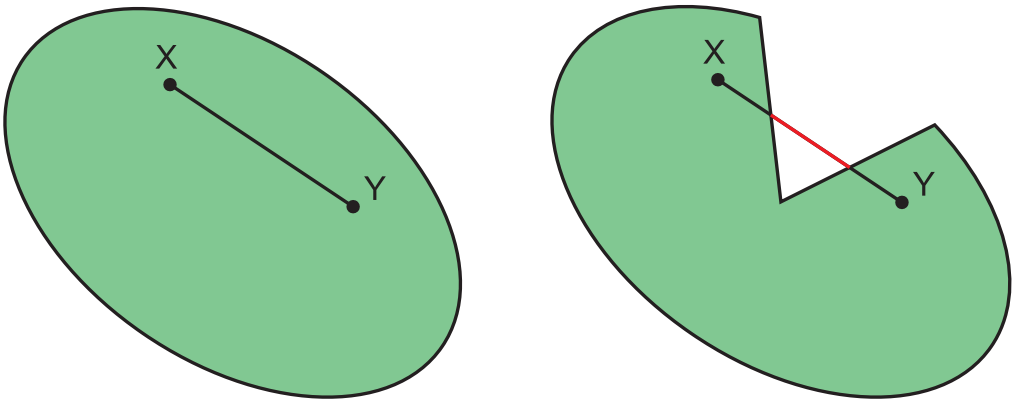
2.2 Análise Convexa

Definição 2.2.1. *Um conjunto não vazio $S \subset \mathbb{R}^n$ é convexo se, para quaisquer pontos $x, y \in S$, tivermos que $\alpha x + (1-\alpha)y \in S$, para qualquer $\alpha \in \mathbb{R}$ tal que $0 \leq \alpha \leq 1$ [3].*

Teorema 2.2.2. *Sejam A e B conjuntos convexos. Então $A \cap B$ é convexo.*

Demonstração. Dados $x,\ y \in A \cap B$ arbitrários, temos que $x,\ y \in A$ e x e $y \in B$. Como A e B são convexos, temos que dado $0 \leq \alpha \leq 1$, $\alpha x + (1-\alpha)y \in A$ e $\alpha x + (1-\alpha)y \in B$, logo $\alpha x + (1-\alpha)y \in A \cap B$. Portanto, $A \cap B$ é convexo. □

Definição 2.2.3. *Um cone é um conjunto não vazio $C \subset \mathbb{R}^n$ tal que para qualquer $x \in C$, temos $\alpha x \in C$, $\forall \alpha \geq 0$. Além disso, se C é convexo, então C é chamado cone convexo [3].*



(a) (b)

Figura 2.2: *A figura (a) apresenta um conjunto convexo. A figura (b) apresenta um conjunto não convexo.*

Definição 2.2.4. *Considere uma norma $\|.\|$ arbitrária em $\mathbb{R}^{k-1}$. Um cone normado associado à norma $\|.\|$ é o conjunto $C_k = \{(x,t) \mid x \in \mathbb{R}^{k-1}, t \in \mathbb{R}, \|x\| \leq t\} \subseteq \mathbb{R}^k$ [28]. Para $k = 1$ definimos $C_1 = \{t \mid 0 \leq \|t\|\}$.*

Observe que todo cone normado é convexo.

Sejam $(x_1, t_1), (x_2, t_2) \in C_k$ arbitrários e α um número real, $\alpha \in [0,\ 1]$. Então

$$\|x_1\| \leq t_1,\ \|x_2\| \leq t_2,$$
$$|\alpha| = \alpha,\ |1-\alpha| = 1-\alpha.$$

Logo

$$|\alpha|\,\|x_1\| \leq |\alpha|\,t_1 \ \Rightarrow\ \|\alpha x_1\| \leq \alpha t_1,$$

$$|(1-\alpha)|\,\|x_2\| \leq |(1-\alpha)|\,t_2 \ \Rightarrow\ \|(1-\alpha)x_2\| \leq (1-\alpha)t_2.$$

Somando as desigualdades acima temos que

$$\|\alpha x_1\| + \|(1-\alpha)x_2\| \leq \alpha t_1 + (1-\alpha)t_2.$$

Segue da desigualdade triangular que

$$\|\alpha x_1 + (1-\alpha)x_2\| \leq \alpha t_1 + (1-\alpha)t_2.$$

E concluímos que

$$(\alpha x_1 + (1-\alpha)x_2, \alpha t_1 + (1-\alpha)t_2) \in C_k.$$

Portanto C_k é convexo.

Definição 2.2.5. *Um cone quadrático é um cone normado, na norma Euclidiana [28], isto é,*

$$C_k = \left\{(x,t) \mid x \in \mathbb{R}^{k-1}, t \in \mathbb{R}, \|x\|_2 \leq t\right\} \subseteq \mathbb{R}^k.$$

O cone quadrático é também chamado de cone de segunda ordem e recebe a abreviação SOC (*second-order cone*).

Definição 2.2.6. *Seja $f\colon \boldsymbol{S} \longrightarrow \mathbb{R}$, onde $\boldsymbol{S}$ é um subconjunto convexo de $\mathbb{R}^n$. A função f é dita convexa em $\boldsymbol{S}$ se para quaisquer x, $y \in \boldsymbol{S}$ e todo $\alpha \in \mathbb{R}$, $0 \leq \alpha \leq 1$, ocorrer*

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

Geometricamente, uma função convexa é uma função f que, dados quaisquer dois pontos x, y no seu domínio, o segmento de reta entre os pontos $(x,\ f(x))$ e $(y,\ f(y))$ fica acima do gráfico da função.

Dizemos que uma função f é côncava se a função $-f$ for convexa.

Uma função $f\colon \mathbb{R} \longrightarrow \mathbb{R}$ definida por $f(x) = ax + b$, com $a, b \in \mathbb{R}$, é chamada função afim.

Por exemplo, a função $f\colon \mathbb{R} \longrightarrow \mathbb{R}$ definida por $f(x) = 2x - 4$ (Figura 2.2), é uma função afim.

Uma função afim é uma função côncava e convexa. Observe:

$$\begin{aligned} f(\alpha x + (1-\alpha)y) &= a(\alpha x + (1-\alpha)y) + b \\ &= \alpha a x + ay - \alpha a y + b \\ &= \alpha a x + \alpha b + ay + b - \alpha a y - \alpha b \\ &= \alpha(ax+b) + (1-\alpha)(ay+b) = \alpha f(x) + (1-\alpha)f(y). \end{aligned}$$

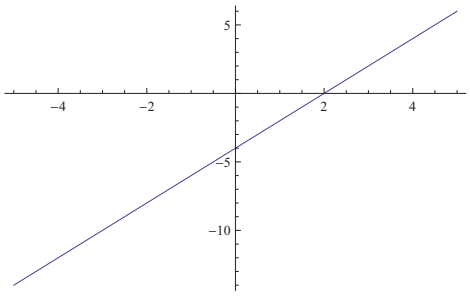


Figura 2.3: *Função afim*

Proposição 2.2.7. *Sejam f_1 e f_2 funções convexas sobre um conjunto convexo Ω. Então a função $f_1 + f_2$ é convexa em Ω.*

Demonstração. Sejam $x,\ y \in \Omega$ e $\alpha \in \mathbb{R},\ 0 \leq \alpha \leq 1$. Então,

$$\begin{aligned} f_1(\alpha x + (1-\alpha)y) &\leq \alpha f_1(x) + (1-\alpha)f_1(y), \\ f_2(\alpha x + (1-\alpha)y) &\leq \alpha f_2(x) + (1-\alpha)f_2(y). \end{aligned}$$

Somando as desigualdades acima, obtemos:

$$\begin{aligned} (f_1 + f_2)(\alpha x + (1-\alpha)y) &= f_1(\alpha x + (1-\alpha)y) + f_2(\alpha x + (1-\alpha)y) \\ &\leq \alpha f_1(x) + (1-\alpha)f_1(y) + \alpha f_2(x) + (1-\alpha)f_2(y) \\ &= \alpha(f_1 + f_2)(x) + (1-\alpha)(f_1 + f_2)(y). \end{aligned}$$

□

Proposição 2.2.8. *Seja f uma função convexa sobre um conjunto convexo Ω. Então a função αf é convexa para qualquer $\alpha \geq 0$.*

Demonstração. Imediata. □

Teorema 2.2.9. *Seja $f : \mathbb{R}^m \longrightarrow \mathbb{R}^n$ uma função afim e $S \subseteq \mathbb{R}^n$ um conjunto convexo. Então tanto a imagem de f:*

$$f(S) = \{f(x) \,|x \in S\},$$

como a imagem da inversa de f:

$$f^{-1}(S) = \{x \,|f(x) \in S\}$$

são convexas.

Demonstração. Dados y_1 e $y_2 \in f(S)$ arbitrários, então existem x_1 e $x_2 \in S$ tais que,

$$f(x_1) = y_1 \text{ e } f(x_2) = y_2 \in f(S).$$

Logo,

$$\alpha f(x_1) + (1-\alpha) f(x_2) = f(\alpha x_1 + (1-\alpha)x_2) \in S,$$

pois $\alpha x_1 + (1-\alpha)x_2 \in S$.

A demonstração da imagem inversa é similar, já que se f é afim, temos que f^{-1} é afim. □

Análise Convexa e Otimização

Consideremos, de acordo com [28], o problema convexo de otimização:

$$\begin{array}{ll} \text{Minimizar} & f(x) \\ \text{sujeita a} & Ax \ + \ b \ = \ 0, \\ & g_j(x) \leq 0, \quad j \ = \ 1, ..., p, \end{array} \tag{2.8}$$

onde, $A \ \in \ \mathbb{R}^{m \times n}$, $b \ \in \ \mathbb{R}^n$ e f, $g_1, .., g_p$ são funções convexas.

Iremos também nos referir ao problema (2.8) como problema convexo se f for côncava e $g_1, .., g_p$ forem funções convexas, já que esse problema pode ser resolvido minimizando a função $-f$, que é convexa.

Teorema 2.2.10. *Para qualquer problema primal, a função objetivo dual é côncava e o conjunto factível é convexo.*

Demonstração. Pela linearidade da Lagrangiana em relação aos multiplicadores, para todo $x \ \in \ \mathbb{R}^n$, λ_1, $\lambda_2 \ \in \ \mathbb{R}^m$ e μ_1, $\mu_2 \ \in \ \mathbb{R}^p$ e $\alpha \ \in \ [0, 1]$, temos que

$$\begin{array}{l} L(x, \alpha \ \lambda_1 \ + (1-\alpha)\lambda_2, \ \alpha \ \mu_1 \ + (1-\alpha)\mu_2) \\ = \ \alpha \ L(x, \lambda_1, \ \mu_1) \ + \ (1-\alpha) \ L(x, \lambda_2, \ \mu_2). \end{array} \tag{2.9}$$

Portanto, considerando x factível primal, (λ_1, μ_1) e (λ_2, μ_2) factíveis duais, temos:

$$\begin{aligned}
&\theta(\alpha\ \lambda_1\ + (1-\alpha)\lambda_2,\ \alpha\ \mu_1\ + (1-\alpha)\mu_2) \\
&= \inf_{x\in\Omega} L(x, \alpha\ \lambda_1\ + (1-\alpha)\lambda_2,\ \alpha\ \mu_1\ + (1-\alpha)\mu_2) \\
&= \inf_{x\in\Omega} \{\alpha\ L(x, \lambda_1,\ \mu_1)\ +\ (1-\alpha)\ L(x, \lambda_2,\ \mu_2)\} \\
&\geq\ \alpha \inf_{x\in\Omega} L(x,\ \lambda_1,\ \mu_1)\ +\ (1-\alpha) \inf_{x\in\Omega} L(x,\ \lambda_2, \mu_2) \\
&=\ \alpha\theta(\lambda_1,\ \mu_1)\ +\ (1-\alpha)\ \theta(\lambda_2,\ \mu_2).
\end{aligned}$$

Logo, a função objetivo dual é côncava.

É fácil perceber que o conjunto factível $\{\mu \mid \mu \geq 0\}$ é um conjunto convexo. □

Do teorema acima segue que todo problema dual é um problema convexo. Este fato independe do problema primal ser convexo [28].

Teorema 2.2.11. *Seja $\Omega \subseteq \mathbb{R}^n$ um conjunto convexo e $f : \Omega \longrightarrow \mathbb{R}$ uma função convexa. Então qualquer mimimizador local é também um minimizador global de f.*

Demonstração. Suponha que $x^* \in \Omega$ é um minimizador local de f e que existe

$$x \in \Omega \text{ tal que } f(x) < f(x^*). \text{ Então,}$$

$$f(\alpha x + (1-\alpha)x^*) \leq \alpha f(x) + (1-\alpha) f(x)^* < f(x^*), \text{ sempre que } 0 \leq \alpha \leq 1.$$

Mas, para α suficientemente pequeno, temos que $\alpha x + (1-\alpha)x^*$ pertence à ϵ-vizinhança de x^*, o que contradiz que x^* é um minimizador local. □

Outra propriedade interessante de problemas convexos está relacionada com as condições KKT (2.2). Em problemas convexos as condições KKT além de necessárias, são suficientes, como veremos no teorema a seguir.

Teorema 2.2.12. *Considerando que as funções f, g do problema (2.1) são convexas, h linear, e que $\tilde{x}$, $(\tilde{\lambda},\ \tilde{\mu})$, são pontos que satisfazem as condições KKT, ou seja,*

$$\begin{gathered}
h(\tilde{x}) = 0, \\
g(\tilde{x}) \geq 0, \\
\nabla f(\tilde{x})\ +\ \tilde{\lambda}\nabla h(\tilde{x})\ +\ \tilde{\mu}\nabla g(\tilde{x}) = 0, \\
\tilde{\mu} \geq 0, \\
\tilde{\mu} g(\tilde{x}) = 0.
\end{gathered}$$

então, $\tilde{x},\ (\tilde{\lambda},\ \tilde{\mu})$ *são ótimos primal e dual, respectivamente, e o gap de dualidade é zero.*

Demonstração. As duas primeiras condições garantem que $\tilde{x}$ é primal factível.

Como $\tilde{\mu}\ \geq\ 0$, pelas proposições 2.2.7 e 2.2.8, temos que a função Lagrangiana $L(x,\ \tilde{\lambda},\ \tilde{\mu})$ é convexa em x. Além disso, a terceira condição garante que o gradiente da Lagrangiana se anula em $x\ =\ \tilde{x}$. Portanto, pelo Teorema 2.2.11, $\tilde{x}$ é minimizador da função Lagrangiana. Além disso, utilizando a primeira e a última condição, temos

$$\begin{aligned}\theta(\tilde{\lambda},\ \tilde{\mu}) &= L(\tilde{x},\ \tilde{\lambda},\ \tilde{\mu}) \\ &= f(\tilde{x})\ +\ \tilde{\lambda}h(\tilde{x})\ +\ \tilde{\mu}g(\tilde{x}) \\ &= f(\tilde{x}).\end{aligned}$$

Portanto, pelo teorema forte da dualidade, temos que $\tilde{x}$ é minimizador primal e $(\tilde{\lambda},\ \tilde{\mu})$ é minimizador dual. □

CAPÍTULO 3

PROBLEMAS ASSOCIADOS A CONES DE SEGUNDA ORDEM

3.1 Introdução

Problemas de programação cônica de segunda ordem (SOCP - *Second-Order Cone Programming*) são problemas convexos de otimização nos quais uma função linear é minimizada sobre restrições lineares e restrições de cone quadrático. Estes problemas são um caso especial de programação semidefinida (SDP - *semidefinite programming*).

Problemas de programação linear, problemas de programação quadrática e problemas de programação quadrática com restrições quadráticas podem ser formulados como problemas SOCP, assim como outros problemas que não se encaixam em nenhuma destas três categorias [1].

A partir deste capítulo usaremos $\|.\|$ para nos referirmos à norma Euclidiana.

3.2 O problema de programação cônica de segunda ordem

Um problema SOCP é escrito na forma:

$$\begin{aligned} \text{Minimizar} \quad & f^T x \\ \text{sujeita a} \quad & \|A_i x + b_i\| \leq {c_i}^T x + d_i, \quad i = 1, ...N, \\ & Fx = g, \end{aligned} \tag{3.1}$$

onde $x \in \mathbb{R}^n$ é a variável de otimização, e os parâmetros do problema são $f \in \mathbb{R}^n$, $A_i \in \mathbb{R}^{(n_i-1)\times n}, b_i \in \mathbb{R}^{n_i-1}, c_i \in \mathbb{R}^n$, $d_i \in \mathbb{R}$, $F \in \mathbb{R}^{m\times n}$, $g \in \mathbb{R}^m$.

Chamamos a restrição $\|A_i x + b_i\| \leq {c_i}^T x + d_i$ de restrição de cone quadrático, ou restrição SOC, já que o conjunto de pontos que satisfazem essa desigualdade é a imagem inversa de um cone quadrático sobre a função afim $f(x) = (Ax + b, c^T x + d)$. Observe que, pelo teorema 2.2.9, essa restrição é convexa.

Temos então que o problema SOCP (3.1) é convexo, já que sua função objetivo é convexa e, de acordo com o teorema 2.2.2, suas restrições definem um conjunto convexo.

3.2.1 Restrições SOC

Restrições SOC podem ser usadas para representar diversas outras restrições convexas. Quando $n_i = 1$ para todo i, a restrição SOC representa uma restrição linear

$$0 \leq c_i^T x + d_i, \ i = 1, ..., N,$$

e o problema (3.1) torna-se um problema de programação linear (PL).

Da mesma forma, quando $c_i = 0$, para todo i, a restrição SOC reduz-se a

$$\|A_i x + b_i\| \leq d_i,$$

e considerando $d_i \geq 0$, temos $\|A_i x + b_i\|^2 \leq d_i^2$, que é uma restrição quadrática. Neste caso, temos um problema com função objetivo linear e restrições quadráticas.

Também é possível escrever diversas restrições como SOC. Como exemplo, considere as restrições hiperbólicas:

$$\begin{cases} x^t x \leq yz \\ y \geq 0 \\ z \geq 0. \end{cases}$$

Estas restrições são conhecidas por restrições hiperbólicas pois descrevem metade de um hiperbolóide.

Para mostrar que o conjunto de restrições acima pode ser escrito como uma restrição SOC, vamos provar que:

$$\left.\begin{array}{c} w^t w \leq yz \\ y \geq 0 \\ z \geq 0 \end{array}\right\} \Leftrightarrow \begin{cases} \left\| \begin{bmatrix} 2w \\ y-z \end{bmatrix} \right\| \leq y+z. \\ y \geq 0 \\ z \geq 0 \end{cases} \tag{3.2}$$

De fato, suponha que $w^T w \leq yz$, então:

$$\begin{aligned} \left\| \begin{bmatrix} 2w \\ y-z \end{bmatrix} \right\|^2 &= 4w^T w + (y-z)^2 \\ &\leq 4yz + (y-z)^2 \\ &= (y+z)^2. \end{aligned}$$

Suponha agora que

$$\left\| \begin{bmatrix} 2w \\ y-z \end{bmatrix} \right\|^2 \leq y+z.$$

Então $4w^T w + (y-z)^2 \leq (y+z)^2$. Logo, $w^T w \leq yz$.

3.3 Problemas que podem ser expressos como SOCP

Diversos problemas podem ser escritos como problemas SOCP. Nesta seção serão apresentados alguns exemplos, extraídos de [28], [12] e [18].

3.3.1 Soma de normas

Considere o problema irrestrito:

$$\text{Minimizar} \sum_{i=1}^{p} \|A_i x + b_i\|, \tag{3.3}$$

onde $A_i \in \mathbb{R}^{(n_i-1)\times n}$ e $b_i \in \mathbb{R}^{(n_i-1)},\ i = 1, ..., p$.

Introduzindo novas variáveis auxiliares, $t_1, ..., t_p$, podemos escrever o problema acima como um problema SOCP:

$$\begin{aligned} \text{Minimizar} \quad & \sum_{i=1}^{p} t_i \\ \text{sujeita a} \quad & \|A_i x + b_i\| \leq t_i, \qquad i = 1, ..., p, \end{aligned}$$

com variáveis de otimização $x \in \mathbb{R}^n$ e $t_i \in \mathbb{R}$.

Observe que não há alteração nos cálculos se inserirmos qualquer restrição SOC no problema (3.3.1).

O problema clássico de *Fermat-Weber* é um caso especial de soma de normas. Ele identifica onde posicionar uma instalação de modo a minimizar a soma das distâncias desta em relação a um conjunto de locais fixos. A formulação deste problema é:

$$\text{Minimizar} \sum_{i=1}^{p} \|d_i - x\|$$

onde $d_i,\ i = 1, ..., p$ são as posições dos locais fixos e x é a posição desconhecida da instalação.

3.3.2 Máximo de normas

De forma similar ao problema de soma de normas, podemos escrever como SOCP problemas cujo objetivo é minimizar o máximo dentre algumas normas.

Considere o problema:

$$\text{Minimizar} \max_{i=1,...,p} \|A_i x + b_i\|.$$

Introduzindo uma única variável t, escrevemos o problema acima como SOCP:

$$\begin{aligned} \text{Minimizar} \quad & t \\ \text{sujeita a} \quad & \|A_i x + b_i\| \leq t, \qquad i = 1, ..., p. \end{aligned}$$

As variáveis deste problema são: $x \in \mathbb{R}^n$ e $t \in \mathbb{R}$.

3.3.3 QCQP - *Quadratically Constrained Quadratic Programming*

Um problema de otimização convexa é chamado de problema de programação quadrática (PQ) se a função objetivo é quadrática e as funções de restrição são afins. Um problema de programação quadrática pode ser expresso na forma:

$$\begin{array}{ll} \text{Minimizar} & \frac{1}{2}x^tPx \ + \ q^tx + r \\ \text{sujeita a} & Bx \ \leq \ v \\ & Fx = g, \end{array} \tag{3.4}$$

onde $P \in \mathbb{R}^{n\times n}$ é simétrica semidefinida positiva, $B \in \mathbb{R}^{N\times n}$, $x,\ q \in \mathbb{R}^n,\ \ r \in \mathbb{R}$, $v \in \mathbb{R}^N$, $F \in \mathbb{R}^{m\times n}$ e $g \in \mathbb{R}^m$.

Problemas de PQ incluem problemas de PL como um caso especial, basta tomar $P = 0$.

Um problema de otimização convexa é chamado de QCQP (*Quadratically Constrained Quadratic Programming*) se a função objetivo e as restrições de desigualdade são quadráticas e as restrições de igualdade são lineares. Um problema QCQP pode ser expresso na forma:

$$\begin{array}{ll} \text{Minimizar} & x^tPx \ + \ 2q^tx + r \\ \text{sujeita a} & x^tA_ix \ + \ 2b_i^Tx + c_i \leq 0, \quad i = 1, ..., N \\ & Fx = g, \end{array} \tag{3.5}$$

onde $P \in \mathbb{R}^{n\times n},\ A_i \in \mathbb{R}^{n\times n},\ i = 1, ..., N$, são simétricas e semidefinidas positivas, $F \in \mathbb{R}^{m\times n}$, $x,\ q,\ b_i \in \mathbb{R}^n$, $g \in \mathbb{R}^m$ e $r,\ c_i \in \mathbb{R}$.

Tomando $A_i = 0$ para todo i, o problema QCQP acima transforma-se em um problema PQ. Logo, problemas QCQP incluem problemas PQ, e consequentemente PL, como casos especiais.

Para mostrar que problemas PL, PQ e QCQP podem ser expressos como SOCP, mostraremos apenas a transformação de QCQP em SOCP, já que PL e PQ são casos especiais de QCQP.

Para simplificar os cálculos trabalharemos apenas com problemas cujas matrizes P e A_i são definidas positivas. Assim, o problema (3.5) pode ser escrito da seguinte forma:

$$\begin{array}{ll} \text{Minimizar} & \left\| P^{\frac{1}{2}}x \ + \ P^{-\frac{1}{2}}q \right\|^2 + r - q^t P^{-1} q \\ \text{sujeita a} & \left\| A_i^{\frac{1}{2}}x \ + \ A_i^{-\frac{1}{2}}b_i \right\|^2 + c_i - b_i^t A_i^{-1} b_i \leq 0, \quad i = 1, ...N, \\ & Fx = g. \end{array}$$

Introduzindo uma nova variável t, podemos escrever o problema acima como:

$$\begin{array}{ll} \text{Minimizar} & t \\ \text{sujeita a} & \left\| P^{\frac{1}{2}}x \ + \ P^{-\frac{1}{2}}q \right\|^2 + r - q^t P^{-1} q \leq t \\ & \left\| A_i^{\frac{1}{2}}x \ + \ A_i^{-\frac{1}{2}}b_i \right\|^2 + c_i - b_i^t A_i^{-1} b_i \leq 0, \quad i = 1, ...N, \\ & Fx = g. \end{array}$$

Tomando $d_i = b_i^t A_i^{-1} b_i - c_i$, temos:

$$\begin{array}{ll} \text{Minimizar} & t \\ \text{sujeita a} & \left\| P^{\frac{1}{2}}x \ + \ P^{-\frac{1}{2}}q \right\| \leq t \\ & \left\| A_i^{\frac{1}{2}}x \ + \ A_i^{-\frac{1}{2}}b_i \right\| \leq d_i, \quad i = 1, ...N, \\ & Fx = g. \end{array}$$

O problema acima é um SOCP com variável de otimização $(t, \ x)$.

3.3.4 Problemas com restrições hiperbólicas

Como visto anteriormente, uma restrição hiperbólica pode ser expressa como restrição SOC. Segue deste fato que uma grande classe de problemas convexos podem ser escritos como problemas SOCP.

Como um exemplo, considere o problema de maximizar a média harmônica de algumas funções de x sobre um politopo:

$$\begin{array}{ll} \text{Maximizar} & \left(\sum_{i=1}^{p} \frac{1}{a_i^T x + b_i} \right)^{-1} \\ \text{sujeita a} & a_i^T x + b_i > 0, \quad i = 1, ..., p, \\ & c_i^T x + d_i \geq 0, \quad i = 1, ..., q. \end{array} \tag{3.6}$$

Podemos reescrever o problema de maximização como problema de minimização, como segue.

$$\begin{array}{rl} \text{Minimizar} & \sum_{i=1}^{p} \frac{1}{(a_i^T x + b_i)} \\ \text{sujeita a} & a_i^T x + b_i > 0, \quad i = 1, ..., p, \\ & c_i^T x + d_i \geq 0, \quad i = 1, ..., q. \end{array}$$

Observe que este problema é convexo, já que sua função objetivo é a soma das funções $1/(a_i^T x + b_i)$, que são convexas para $a_i^T x + b_i > 0$. De fato, considerando $f(x) = \frac{1}{ax+b}$ e utilizando a desigualdade das médias aritmética e geométrica,

$$\frac{\sum_{i=1}^{n} a_i x_i}{\sum_{i=1}^{n} a_i} \geq \left(\prod_{i=1}^{n} a_i x_i \right)^{\frac{1}{\sum_{i=1}^{n} a_i}},$$

temos:

$$\begin{aligned} f(\alpha x + (1-\alpha)y) &= (a(\alpha x + (1-\alpha)y) + b)^{-1} \\ &= (\alpha(ax+b) + (1-\alpha)(ay+b))^{-1} \\ &\leq ((ax+b)^{\alpha}(ay+b)^{1-\alpha})^{-1} \\ &= f(x)^{\alpha} f(y)^{1-\alpha} \\ &\leq \alpha f(x) + (1-\alpha) f(y). \end{aligned}$$

Assim, introduzindo novas variáveis $t_i,\ i = 1, .., p,\ t_i \in \mathbb{R}$, podemos escrever o problema (3.6) como um problema com restrições hiperbólicas:

$$\begin{array}{rl} \text{Minimizar} & \sum_{i=1}^{p} t_i \\ \text{sujeita a} & t_i(a_i^T x + b_i) \geq 1, \\ & t_i \geq 0, \quad i = 1, ..., p, \\ & c_i^T x + d_i \geq 0, \quad i = 1, ..., q. \end{array}$$

De acordo com (3.2), para $w = 1,\ y = t_i$ e $z = a_i^T x - b_i$, podemos escrever o problema

acima como um SOCP nas variáveis x e t.

$$\begin{aligned}
&\text{Minimizar} \quad \sum_{i=1}^{p} t_i \\
&\text{sujeita a} \quad \left\| \begin{bmatrix} 2 \\ t_i - a_i^T x + b_i \end{bmatrix} \right\| \leq t_i + a_i^T x - b_i, \\
&\qquad t_i \geq 0, \quad i = 1, ..., p, \\
&\qquad c_i^T x + d_i \geq 0, i = 1, ..., q.
\end{aligned}$$

3.3.5 Superfície mínima

Considere uma função diferenciável $f : \mathbb{R}^2 \longrightarrow \mathbb{R}$ com $domf = C$. A área da superfície do gráfico é dada por:

$$A = \int_C \|\nabla f(x), 1\| \, dx,$$

o qual é um funcional convexo de f.

O problema de superfície mínima é encontrar a função f que minimiza a área A sujeita a algumas restrições, por exemplo alguns valores dados de f no limite de C.

Resolveremos este problema por aproximação, discretizando a função f. Sejam $C = [0, 1] \times [0, 1]$ e f_{ij} o valor de f no pontos $\left[\frac{i}{k}, \frac{j}{k}\right]$. Aproximando a integral como uma soma, obtemos a área discretizada:

$$A \approx A_{disc} = \frac{1}{k^2} \sum_{i,j=0}^{k-1} \left\| \begin{array}{c} \nabla f_{i,j} \\ 1 \end{array} \right\|.$$

Utilizando diferenças finitas avançadas encontramos uma aproximação para o gradiente de f no ponto $\bar{x} = (\frac{i}{k}, \frac{j}{k})$:

$$\nabla f(\bar{x}) \approx k \begin{bmatrix} f_{i+1,j} - f_{i,j} \\ f_{i,j+1} - f_{i,j} \end{bmatrix}.$$

Substituindo a aproximação do gradiente na equação da área discretizada, encontramos:

$$A \approx A_{disc} = \frac{1}{k^2} \sum_{i,j=0}^{k-1} \left\| \begin{array}{c} k(f_{i+1,j} - f_{i,j}) \\ k(f_{i,j+1} - f_{i,j}) \\ 1 \end{array} \right\|.$$

A título de ilustração, considere o problema de encontrar a superficie de área mínima com os valores limites fixados nas margens esquerda e direita do quadrado C. Este problema pode ser expresso por:

$$\begin{array}{ll} \text{Minimizar} & A_{disc} \\ \text{sujeita a} & f_{0,j} = l_j, \\ & f_{k,j} = r_j, \quad j = 1, ..., k, \end{array} \tag{3.7}$$

onde $f_{i,j}$, para $i, j = 0, ..., k$, são variáveis e l_j, r_j são valores limites dados do lado direito e esquerdo de C.

Introduzindo novas variáveis $t_{i,j}$, com $i, j = 0, ..., k-1$, podemos transformar o problema (3.7) em um SOCP, como segue.

$$\begin{array}{ll} \text{Minimizar} & \frac{1}{k^2} \sum_{i,j=0}^{k-1} t_{i,j}, \\ \text{sujeita a} & \left\| \begin{array}{c} k(f_{i+1,j} - f_{i,j}) \\ k(f_{i,j+1} - f_{i,j}) \\ 1 \end{array} \right\| \leq t_{i,j}, \ i, \ j = 0, ..., k-1, \\ & f_{0,j} = l_j, \\ & f_{k,j} = r_j, \quad j = 1, ..., k. \end{array} \tag{3.8}$$

3.3.6 Programação Linear Robusta

Em muitos problemas práticos de otimização a presença de incertezas em determinados parâmetros é um fato. Porém é comum construir modelos de otimização admitindo que os dados são conhecidos exatamente e ignorar a possível influência de incertezas, associadas a esses parâmetros, sobre a otimalidade e factibilidade das soluções desses modelos [17].

Segundo [4], para casos reais de programação linear, é importante considerar a possibilidade de que uma pequena incerteza nos dados (intrínseca para a maioria dos problemas de casos reais de PL) pode fazer a solução ótima completamente sem sentido do ponto de vista prático.

Considere um problema de programação linear:

$$\begin{aligned} &\text{Minimizar} && c^T x \\ &\text{sujeita a} && Ax - b \leq 0. \end{aligned} \tag{3.9}$$

Como dito acima, em aplicações reais, os dados $c,\ A,\ b$ do problema (3.9) não são todos conhecidos exatamente, o que geralmente se conhece é um domínio U nos espaços dos dados, conhecido por ***conjunto de incertezas***, que certamente contém os dados reais. Há casos em que, apesar dessa incerteza dos dados, a escolha de x deve cumprir as restrições reais, quer as conheçamos ou não [18].

Se realmente tudo o que sabemos sobre os dados é que eles pertencem a um determinado conjunto U, mas ainda temos que satisfazer as restrições reais, a única maneira de satisfazer os requisitos é limitar-nos a soluções factíveis candidatas a soluções robustas. Soluções robustas são soluções que satisfazem todas as possíveis restrições incertas. Para este problema, soluções factíveis candidatas a robustas são os vetores x tais que

$$\begin{aligned} &Ax - b \leq 0 \ \forall\ [A; b] \\ &\text{para algum } c \text{ que satisfaça } (c, A, b) \in U. \end{aligned}$$

A fim de escolher entre essas soluções factíveis robustas a melhor possível, tomamos como função objetivo $f(x)$ o maior valor de todas as realizações possíveis de $c^t x$, ou seja

$$f(x) = \sup c^t x.$$

Desta forma, podemos associar ao nosso problema de programação linear incerto, o problema de minimizar o valor ótimo robusto sobre o conjunto de todas as soluções factíveis robustas, ou seja,

$$\begin{aligned} &\text{Minimizar} && t \\ &\text{sujeita a} && c^t x \leq t \\ & && Ax - b \leq 0 \qquad \forall (c, A, b) \in U. \end{aligned} \tag{3.10}$$

O problema acima é chamado contrapartida robusta de (3.9).

Note que (3.10) é um problema de otimização usual, sem incertezas, mas na maioria das vezes não é um problema de programação linear. A estrutura de (3.10) depende da geometria do conjunto U.

Segundo Nemirovski [18], é razoável especificar o conjunto incerteza como um elipsóide, pois

- conjuntos de incertezas elipsoidais representam uma grande família, incluindo politopos e podem ser usados como boa aproximação, em muitos casos, de conjuntos convexos complicados;
- um elipsóide é representado parametricamente por dados de tamanho moderado, portanto é útil utilizá-lo como entrada;
- em muitos casos de dados estocásticos incertos, existem argumentos probabilísticos que permitem substituir incerteza estocástica por incerteza elipsoidal determinística;
- por último, e mais importante, o problema de incertezas se tranforma em um problema SOCP.

Para mostrar que um problema de programação linear com elipsódes de incerteza pode ser representado como um problema SOCP, assumiremos, para simplificar, que c e b de (3.9) são fixos, e que as linhas a_i de A pertencem a elipsóides dados:

$$a_i \in \epsilon_i = \{\bar{a}_i + P_i u \mid \|u\| \leq 1\}, \tag{3.11}$$

onde P_i é simétrica definida positiva.

Observe que se P_i fosse singular, obteríamos elipsóides planos com dimensão igual ao posto de P_i e que se $P_i = 0$, então, a_i seria conhecido.

Para satisfazer a exigência de que as restrições sejam satisfeitas para todos os valores possíveis dos parâmetros a_i, associamos ao problema original sua contrapartida robusta, ou programa linear robusto

$$\begin{array}{lll} \text{Minimizar} & c^t x & \\ \text{sujeita a} & a_i^T x - b_i \leq 0 & \forall a_i \in \epsilon_i, i = 1, ..., m, \\ & (c, A, b) \in U. & \end{array} \tag{3.12}$$

A restrição linear robusta $a_i^T x - b_i \leq 0,\ \forall a_i \in \epsilon_i, i = 1, ..., m$ pode ser expressa como

$$sup\left\{a_i^T x \mid a_i \in \epsilon_i\right\} \leq b_i.$$

Utilizando (3.11), podemos reescrever o lado esquerdo da desigualdade acima da seguinte forma:

$$\sup\left\{a_i^T x \mid a_i \in \epsilon_i\right\} = \bar{a}_i^T x + \sup\left\{u^t P_i x \mid \|u\| \leq 1\right\}.$$

Pela desigualdade de Cauchy-Schwarz, temos:

$$\bar{a}_i^T x + \sup\left\{u^t P_i x \mid \|u\| \leq 1\right\} = \bar{a}_i^T x + \|P_i x\|.$$

Portanto, a restrição linear robusta pode ser expressa por:

$$\bar{a}_i^T x + \|P_i x\| \leq b_i,$$

a qual é uma restrição SOC.

Assim, o problema linear robusto (3.12) pode ser expresso por:

$$\begin{array}{ll} \text{Minimizar} & c^t x \\ \text{sujeita a} & \bar{a}_i x + \left\|P_i^T x\right\| \leq b_i, \quad i = 1, ..., m, \end{array} \tag{3.13}$$

que é um problema SOCP.

Note que os termos adicionais envolvendo normas atuam como regularização, previnem que x cresça em direções consideradas incertas em relação aos parâmetros a_i.

3.3.7 Quadrados mínimos robustos

Considere um conjunto de restrições $Ax \approx b$, onde $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ estão sujeitos a erros desconhecidos mas limitados δA e δb, com $\|\delta A\| < \rho$ e $\|\delta b\| < \xi$. Definimos a solução robusta de quadrados mínimos como a solução $\hat{x} \in \mathbb{R}^n$ que minimiza o maior valor residual, ou seja, é a solução do problema de quadrados mínimos robusto:

$$\text{Minimizar} \max_{\substack{\|\delta A\|<\rho \\ \|\delta b\|<\xi}} \|(A + \delta A)x - (b + \delta b)\|. \tag{3.14}$$

É possivel, como veremos a seguir, escrever a função objetivo do problema (3.14) sem incertezas.

Utilizando a desigualdade de Cauchy-Schwarz, temos:

$$\begin{aligned}\|y\| \, \|(A+\delta A)x - (b+\delta b)\| &= \|y\| \, \|Ax - b + \delta Ax - \delta b\| \\ &\geq y^t(Ax - b + \delta Ax - \delta b).\end{aligned}$$

Desta forma,

$$\max_{\substack{\|\delta A\|<\rho \\ \|\delta b\|<\xi}} \|(A+\delta A)x - (b+\delta b)\| = \max_{\substack{\|\delta A\|<\rho \\ \|\delta b\|<\xi}} \max_{\|y\|\leq 1} y^t(Ax-b) + y^t\delta Ax - y^t\delta b.$$

Tomando $y^t\delta A = z$, temos:

$$\|z\| = \left\|y^t\delta A\right\| \leq \|y\| \, \|\delta A\| \leq \|\delta A\| \leq \rho.$$

E observando que

$$y^t\delta b \leq \|y\| \, \|\delta b\| \leq \|\delta b\| \leq \xi,$$

temos:

$$\max_{\substack{\|\delta A\|<\rho \\ \|\delta b\|<\xi}} \max_{\|y\|\leq 1} y^t(Ax-b) + y^t\delta Ax - y^t\delta b = \max_{\|z\|\leq\rho} \max_{\|y\|\leq 1} y^t(Ax-b) + z^tx - \xi.$$

Novamente pela desigualdade de Cauchy,

$$y^t(Ax-b) \leq \|y\| \, \|Ax-b\| \leq \|Ax-b\|$$

e

$$z^tx \leq \|z\| \, \|x\| \leq \rho \, \|x\| \, .$$

Assim,

$$\max_{\|z\|\leq\rho} \max_{\|y\|\leq 1} y^t(Ax-b) + z^tx + \xi = \|Ax-b\| + \rho \, \|x\| - \xi.$$

Logo, o problema (3.14) é equivalente ao problema de minimizar a soma das normas:

$$\text{Minimizar } \|Ax-b\| + \rho \, \|x\| - \xi.$$

O problema acima é um SOCP e pode ser resolvido como tal, porém há uma solução mais simples através da decomposição em valor singular de A. A resolução como problema SOCP torna-se útil quando existirem restrições adicionais sobre x, por exemplo, restrições de não negatividade.

Vamos estudar uma aplicação semelhante à aplicação da seção anterior. Considere que as linhas a_i de A estão sujeitas a erros independentes, mas conhecidos por pertencerem a um dado elipsóide ϵ_i:

$$a_i \in \epsilon_i = \{\bar{a}_i + P_i u \,|\, \|u\| \leq 1\}, \tag{3.15}$$

onde P_i é simétrica definida positiva.

Obtemos a estimativa x de quadrados mínimos robustos minimizando o pior caso residual:

$$\text{Minimizar } \max_{a_i \in \epsilon_i} \left(\sum_{i=1}^{n} \left(a_i^T x - b_i \right)^2 \right)^{1/2}. \tag{3.16}$$

Da mesma forma que fizemos cálculos algébricos com o problema (3.14), faremos com este, com a finalidade de escrever a função objetivo sem incertezas. Assim:

$$\begin{aligned}
&\max_{a_i \in \epsilon_i} \left| a_i^T x - b_i \right| \\
&= \max_{\|u\| \leq 1} \left| \bar{a}_i^T x - b_i + u^t P_i x \right| \\
&= \max_{\|u\| \leq 1} \max \left\{ \bar{a}_i^T x - b_i + u^t P_i x, -\bar{a}_i^T x + b_i - u^t P_i x \right\} \\
&= \max \left\{ \bar{a}_i^T x - b_i + \|P_i x\|, -\bar{a}_i^T x + b_i + \|P_i x\| \right\} \\
&= |\bar{a}_i{}^t x - b_i| + \|P_i x\|.
\end{aligned}$$

Portanto, o problema de quadrados mínimos robusto (3.16) pode ser formulado da seguinte maneira:

$$\text{Minimizar } \left(\sum_{i=1}^{n} \left(\left| \bar{a}_i{}^t x - b_i \right| + \|P_i x\| \right)^2 \right)^{1/2},$$

o qual pode ser expresso como SOCP:

$$\begin{array}{ll} \text{Minimizar} & s \\ \text{sujeita a} & \|t\| \leq s \\ & \left|\bar{a}_i^T x - b_i\right| + \|P_i x\| \leq t_i, \quad i = 1, ..., n. \end{array} \tag{3.17}$$

Essas duas variações de quadrados mínimos robustos podem ser estendidas para permitir a incerteza sobre b. Para o problema (3.14), suponha que os erros δA e δb são limitados, sendo $\|[\delta A \ \delta b]\| \leq \rho$. Usando a mesma análise anterior, é possível mostrar que:

$$\max_{\|[\delta A \ \delta b]\| \leq \rho} \|(A + \delta A)x - (b + \delta b)\| = \|Ax - b\| + \rho \left\| \begin{bmatrix} x \\ 1 \end{bmatrix} \right\|.$$

Assim, temos o problema:

$$\text{Minimizar} \ \|Ax - b\| + \rho \left\| \begin{bmatrix} x \\ 1 \end{bmatrix} \right\|.$$

E concluímos que a solução de quadrados mínimos robustos pode ser encontrada resolvendo um problema SOCP, sem incertezas.

CAPÍTULO 4

MÉTODO DE PONTOS INTERIORES PRIMAL-DUAL

Neste capítulo, inicialmente, apresentaremos um estudo de métodos de pontos interiores do tipo primal-dual, seguido de um estudo do método de barreira e sobre o método de redução potencial.

Os métodos apresentados serão aplicados a problemas de programação não linear gerais e a problemas SOCP.

4.1 Problema Geral de Programação Não Linear

Nesta seção consideraremos inicialmente problemas de programação não linear na forma padrão.

4.1.1 Métodos de Pontos Interiores do tipo Primal-Dual

Os métodos do tipo primal-dual aplicados a problemas de programação não linear (2.1),

$$\begin{aligned} \text{Minimizar} \quad & f(x) \\ \text{sujeita a} \quad & h(x) = 0, \\ & g(x) \leq 0, \end{aligned} \tag{4.1}$$

geram iterações (x^k, λ^k, μ^k) que satisfazem as desigualdades $g(x) \geq 0,\ \mu \geq 0$ estritamente. Esta propriedade originou o termo *pontos interiores*. Respeitando essas desigualdades, os métodos evitam soluções falsas, pontos que satisfazem

$$\nabla L(x,\ \lambda,\ \mu) = 0,\ h(x) = 0,\ \mu g(x) = 0,$$

mas não satisfazem

$$g(x) \geq 0,\ \mu \geq 0.$$

Como soluções falsas surgem de forma abundante e nenhuma delas gera informação útil sobre o problema original, o melhor a se fazer é excluí-las da região de busca.

Da mesma maneira que a maioria dos algoritmos em otimização, métodos primal-dual de pontos interiores possuem dois componentes básicos: um procedimento para determinar o passo, ou direção de busca, e uma medida de conveniência de cada ponto no espaço de busca. O procedimento da direção de busca tem origem no Método de Newton.

Procedimento Básico do Método de Newton

Considere as condições KKT (2.2):

$$\nabla_x L(x, \lambda, \mu) = 0, \tag{4.2a}$$

$$h(x) = 0, \tag{4.2b}$$

$$g(x) \geq 0, \tag{4.2c}$$

$$\mu_i g_i(x) = 0, \quad i = 1, ..., p, \tag{4.2d}$$

$$\mu \geq 0. \tag{4.2e}$$

O Método de Newton fornece um modelo linear para as equações (4.2a, 4.2b, 4.2d) na vizinhança do ponto atual e obtém uma direção de busca $(\Delta x, \Delta\lambda, \Delta\mu)$.

Geralmente não é possível realizar um passo completo nessa direção sem violar as fronteiras $g(x) \geq 0$ e $\mu \geq 0$. Para evitar que isso ocorra, realizamos uma busca linear sobre a direção de Newton. Então a nova iteração é:

$$(x, \lambda, \mu) + \alpha(\Delta x, \Delta\lambda, \Delta\mu),$$

para algum parâmetro escalar $\alpha \in (0, 1]$. Infelizmente, na maioria dos casos, para não violar as fronteiras, realizamos passos pequenos nesta direção, portanto o Método de Newton não nos permite grandes progressos em direção a uma solução.

Como veremos a seguir, métodos do tipo primal-dual modificam o procedimento básico do Método de Newton de duas maneiras importantes:

1. fazem com que a direção de busca aponte para o interior da região não negativa ($g(x) \geq 0,\ \mu \geq 0$). Assim podemos realizar passos maiores na direção desejada, sem que as fronteiras sejam violados.

2. mantêm as componentes de $g(x)$ e μ estritamente positivas enquanto possível.

4.1.2 O Caminho Central

No processo iterativo de métodos de pontos interiores, são propostas duas modificações nas condições KKT: conversão das desigualdades em igualdades, usando variáveis de folga $s > 0$, e a perturbação das equações de complementariedade (4.2d) através do parâmetro τ [8]. Desta forma, o novo conjunto de equações é:

$$\nabla_x L(x, \lambda, \mu) = 0, \tag{4.3a}$$

$$h(x) = 0, \tag{4.3b}$$

$$g(x) + s = 0, \tag{4.3c}$$

$$diag(\mu)s - \tau e = 0, \tag{4.3d}$$

$$(s, \mu, \tau) > 0, \tag{4.3e}$$

onde $\tau \in \mathbb{R}$, $s \in \mathbb{R}^p$, $e \in \mathbb{R}^p$ e $e = [1, ..., 1]^T$.

O caminho central $\mathcal{C}$ é um arco de pontos estritamente factíveis que desempenha um papel vital na teoria dos algoritmos do tipo primal-dual. Cada ponto $(x, \lambda, \mu) \in \mathcal{C}$ é solução do sistema (4.3).

Para resolver o sistema (4.3), utiliza-se o Método de Newton, que fornece um modelo linear para as equações deste sistema em torno do ponto atual e obtém uma direção de busca $(\Delta x, \Delta\lambda, \Delta\mu)$ em dois passos.

Primeiro as direções Δx e $\Delta\lambda$ são encontradas através do sistema:

$$\begin{bmatrix} T & \nabla h(x) \\ \nabla^T h(x) & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} t \\ -h(x) \end{bmatrix}, \tag{4.4}$$

onde

$$T = \nabla^2_{xx} L + \nabla g \ \text{diag}\left(\tfrac{\mu}{s}\right) \nabla^T g$$
$$t = -\nabla_x L + \nabla g \ \text{diag}(s^{-1})(\tau e + \text{diag}(\mu) g).$$

No segundo passo, as direções Δs e $\Delta\mu$ são obtidas por:

$$\begin{aligned} \Delta s &= -g(x) - s - \nabla^T g(x) \Delta x, \\ \Delta\mu &= -\mu + \ \text{diag}(s^{-1})(\tau e - \ \text{diag}(\mu)\Delta s). \end{aligned} \tag{4.5}$$

Para assegurar a positividade de s e μ, são utilizados dois parâmentros, $\alpha_p \in (0, 1]$ e $\alpha_d \in (0, 1]$. Desta forma, as variáveis primal e dual para a iteração seguinte são:

$$\begin{aligned} x^{k+1} &= x + \alpha_p \Delta x, \\ s^{k+1} &= s + \alpha_p \Delta s, \\ \lambda^{k+1} &= \lambda + \alpha_d \Delta\lambda, \\ \mu^{k+1} &= \mu + \alpha_d \Delta\mu. \end{aligned}$$

A cada iteração, o parâmetro de perturbação τ é gradualmente reduzido a zero, de forma que as equações (4.3) se aproximam das condições KKT.

4.1.3 Método Seguidor de Caminho (*Path-Following Method*)

Um algoritmo *path-following* restringe as iterações para uma vizinhança do caminho central $\mathcal{C}$ e segue para uma solução do problema original. A vizinhança exclui pontos (x, λ, μ) que estão

muito próximos da fronteira da região não negativa. Assim, direções de busca calculadas a partir de qualquer ponto desta vizinhança fazem pelo menos progressos mínimos em direção ao conjunto solução [31].

Ao manter todas as iterações dentro de uma vizinhança, os métodos do tipo seguidor de caminho reduzem todos os produtos $\mu_i g_i(x)$ para zero aproximadamente na mesma taxa.

Alguns métodos do tipo seguidor de caminho escolhem valores conservadores para o parâmetro central σ (isto é, σ apenas ligeiramente inferior a 1). Estes métodos, que são conhecidos como métodos *short-step path-following*, fazem apenas progressos lentos em direção à solução porque uma vizinhança restritiva é necessária para fazê-lo funcionar.

Os métodos *long-step path-following* fazem escolhas menos conservadoras de σ do que os métodos *short-steps*. Como uma consequência, eles precisam realizar uma busca linear ao longo de $(\Delta x,\ \Delta\lambda,\ \Delta\mu)$ para evitar deixar a vizinhança escolhida. Fazendo escolhas convenientes para σ, todavia, métodos long-step podem fazer progressos muito mais rápido que métodos short-step.

4.1.4 O Método de Barreira

Um método seguidor de caminho muito utilizado é o método de barreira. Este é utilizado para a resolução de problemas com restrições de desigualdade, cujo interior é não vazio, pois trabalha no interior da região factível, utilizando uma função auxiliar que cresce indefinidamente próxima à fronteira e uma sequência decrescente de fatores de barreira.

Os métodos de barreira introduzem as restrições na função objetivo através de um parâmetro de barreira, que impede a aproximação de um ponto factível à fronteira da região factível. Trabalhando no interior dessa região, tais parâmetros geram barreiras que impedem as variáveis de violarem suas fronteiras. Logo, parte-se de um ponto factível e geram-se novos pontos factíveis.

O objetivo é transformar um problema com restrições de desigualdade (2.1),

$$\begin{aligned} \text{Minimizar} \quad & f(x) \\ \text{sujeita a} \quad & h(x) = 0 \\ & g(x) \leq 0, \end{aligned}$$

em um problema com restrições de igualdade, no qual o método de Newton pode ser aplicado [28].

O primeiro passo é reescrever o problema (2.1), colocando as restrições de desigualdade implicitamente na função objetivo:

$$\begin{array}{rl} \text{Minimizar} & f(x) + \sum_{i=1}^{p} -(\frac{1}{t}) \log(-g_i(x)) \\ \text{sujeita a} & h(x) = 0, \end{array} \tag{4.6}$$

onde $t > 0$ é um parâmetro que define a precisão da aproximação.

A função $\phi(x) = \sum_{i=1}^{p} -\log(-g_i(x))$ é chamada de *barreira logarítmica* do problema (2.1). Seu domínio

$$dom\phi = \{x \in \mathbb{R}^n \mid g_i(x) < 0, i = 1, ..., p\},$$

é o conjunto de pontos que satisfazem as restrições de desigualdade de (2.1) estritamente. Independentemente do valor de t, a função barreira cresce ilimitada se $g_i(x) \to 0$, para qualquer i, isto é,

$$\left(\frac{1}{t}\right) \phi(x) \stackrel{t\to\infty}{\longrightarrow} \begin{cases} 0, & \text{se } g(x) < 0, \\ \infty & \text{caso contrário.} \end{cases} \tag{4.7}$$

Quando aumentamos t, a solução $x(t)$ do subproblema apresentado em (4.6) aproxima-se da solução x^* do problema original.

Considerando $t = \frac{1}{\mu}$, temos a função Lagrangiana associada ao subproblema (4.6):

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

Desta forma, podemos escrever o sistema que define o caminho central (4.3) como [30]:

$$\begin{array}{c} f(x) + \sum_{i=1}^{p} -\mu \log(-g_i(x)) + \sum_{i=1}^{m} \lambda_i h_i(x) = 0, \\ h(x) = 0, \\ \text{diag}(\mu)s - \tau e = 0, \\ (s, \mu, \tau) > 0. \end{array} \tag{4.8}$$

Para resolver o sistema (4.8) utiliza-se o Método de Newton. O vetor direção de busca $[\Delta x, \Delta\lambda, \Delta\mu]$ é utilizado para atualizar as variáveis $x,\ \lambda,\ \mu$ [10] como segue:

$$(x, \lambda, \mu) + \alpha(\Delta x, \Delta\lambda, \Delta\mu),$$

em que o tamanho de passo $\alpha \in (0, 1]$ é escolhido para preservar a positividade de t.

Pra mais detalhes ver [7], [30].

4.1.5 Métodos de Redução Potencial

Métodos de redução potencial realizam passos semelhantes aos dos métodos do tipo seguidor de caminho, mas não explicitamente na direção de $\mathcal{C}$. Estes métodos usam uma função logarítmica potencial para medir o mérito de cada ponto estritamente factível e visam atingir uma certa redução fixa nesta função em cada iteração [31]. A função potencial tipo primal-dual, a qual denotamos por Φ, geralmente tem duas propriedades importantes:

$$\begin{aligned} &\Phi(x, \lambda, \mu) \rightarrow\ \infty \text{ se } \mu_i g_i(x) \rightarrow 0, \text{ mas } \mu \not\rightarrow 0, \\ &\Phi(x, \lambda, \mu) \rightarrow\ -\infty \text{ se, e só se, } (x,\ \lambda,\ \mu)\ \rightarrow\ (x^*,\ \lambda^*,\ \mu^*) \in \Omega, \end{aligned} \tag{4.9}$$

onde Ω é o conjunto das soluções primais-duais. A primeira propriedade de (4.9) impede que qualquer um dos produtos $\mu_i g_i(x)$ aproxime-se de zero independentemente dos outros e, portanto, mantém as iterações distantes da fronteira da região não negativa. A segunda propriedade, relaciona Φ com o conjunto solução Ω. Se o algoritmo forçar Φ para $-\infty$, então (4.9) assegura que a sequência se aproxima de um ponto do conjunto solução.

Como nos algoritmos do tipo primal-dual, os algoritmos de redução potencial obtêm suas direções de busca resolvendo os sistemas (4.4), (4.5). Os passos α_p e α_d são escolhidos minimizando Φ aproximadamente ao longo das direções calculadas. Há estratégias adequadas para garantir redução constante de Φ em toda iteração. Assim Φ tenderá a $-\infty$, forçando a convergência (ver [31]).

4.2 SOCP - Second-Order Cone Programming

Nesta seção, faremos uma breve descrição de métodos do tipo primal-dual aplicados a problemas SOCP.

4.2.1 Dual do SOCP

Para os cálculos desta seção, associaremos ao cone C a ordenação parcial em $\mathbb{R}^n$ definida por:

$$x \geq_C y \Leftrightarrow x - y \in C \tag{4.10}$$

Desta forma, temos que $x \geq_C 0$ representa $x \in C$.

Considere o problema SOCP (3.1) reescrito da seguinte forma:

$$\begin{array}{rl} \text{Minimizar} & f^T x \\ \text{sujeita a} & A_i x + b_i = u_i, \\ & c_i^t x + d_i = t_i, \\ & \|u_i\| \leq t_i \quad i = 1, ...N. \end{array} \tag{4.11}$$

Observe que, pela definição 2.2.4, as restrições de desigualdade $\|u_i\| \leq t_i$ implicam que, para todo i, o vetor $(u_i; t_i)$ pertence ao cone C_i. Desta forma, temos por (4.10) que $(u_i; t_i) \geq_{C_i} 0$ e podemos reescrever o problema primal por:

$$\begin{array}{rl} \text{Minimizar} & f^T x \\ \text{sujeita a} & A_i x + b_i = u_i, \\ & c_i^t x + d_i = t_i, \\ & (u_i; t_i) \geq_{C_i} 0 \quad i = 1, ...N. \end{array}$$

Definindo $y = (x, u, t)$, temos a Lagrangiana deste problema:

$$L(y, \lambda, \mu) = f^T x + \sum_{i=1}^{N} \lambda_{1_i}^T (A_i x + b_i - u_i) + \lambda_{2_i} (c_i^T x + d_i - t_i) - \mu_i^T (u_i; t_i),$$

com $\mu_i \geq_{C_i} 0$, ou seja, $\mu_i \in C_i$.

Considerando $\mu_i = (z_i; w_i)$ temos novamente pela definição 2.2.4 que $\|z_i\| \leq w_i$. Assim, podemos reescrever a Lagrangiana como,

$$\begin{aligned} L(y,\lambda,\mu) &= f^T x + \sum_{i=1}^{N} \lambda_{1_i}^T (A_i x + b_i - u_i) + \lambda_{2_i}(c_i^T x + d_i - t_i) - (z_i; w_i)^T (u_i; t_i) \\ &= f^T x + \sum_{i=1}^{N} \lambda_{1_i}^T (A_i x + b_i - u_i) + \lambda_{2_i}(c_i^T x + d_i - t_i) - z_i^T u_i - w_i t_i. \end{aligned}$$

Reorganizando, temos:

$$L(y,\lambda,\mu) = \left(f + \sum_{i=1}^{N} (A_i^T \lambda_{1_i} + c_i \lambda_{2_i}) \right)^T x - \sum_{i=1}^{N} (\lambda_{1_i} + z_i)^T u_i - (\lambda_{2_i} + w_i) t_i.$$

Desta forma, temos:

$$\theta(\lambda,\mu) = \inf_y \; L(y,\lambda,\mu) = \begin{cases} \sum_{i=1}^{N} b_i^T \lambda_{1_i} + d_i^T \lambda_{2_i} & \text{se} \begin{cases} f + \sum_{i=1}^{N} A_i^T \lambda_{1_i} + c_i \lambda_{2_i} = 0, \\ \sum_{i=1}^{N} \lambda_{1_i} + z_i = 0, \\ \sum_{i=1}^{N} \lambda_{2_i} + w_i = 0. \end{cases} \\ -\infty & \text{caso contrário.} \end{cases}$$

De acordo com a equação acima, utilizamos $\lambda_{1_i} = -z_i$ e $\lambda_{2_i} = -w_i$ temos o SOCP dual:

$$\begin{aligned} \text{Maximizar} \quad & -\sum_{i=1}^{N} (b_i^T z_i + d_i w_i) \\ \text{sujeita a} \quad & \sum_{i=1}^{N} (A_i^T z_i + c_i w_i) = f \\ & \|z_i\| \leq w_i \quad i = 1, ...N. \end{aligned} \tag{4.12}$$

As variáveis duais são os vetores $z_i \in \mathbb{R}^{n_i - 1}$ e $w \in \mathbb{R}^N$. Como visto anteriormente, o SOCP dual também é convexo. De fato, (4.12) e (4.11) têm a mesma forma. Além disso, se eliminarmos as restrições de igualdade, podemos reescrever o SOCP dual na forma do SOCP original (3.1).

4.2.2 Barreira para Cone Quadrático

Definimos a função barreira para cone quadrátido de forma semelhante à função barreira original (4.7):

$$\phi(u,t) = \begin{cases} -\log(t^2 - \|u\|^2), & \text{se } \|u\| < t, \\ \infty \quad \text{caso contrário.} \end{cases} \tag{4.13}$$

Observe que a função ϕ é a função barreira para cone de segunda ordem. De fato, sendo C um cone de segunda ordem, $\phi(u,t)$ é finito se e só se $(u,t) \in C$ (i.e., $\|u\| < t$), e $\phi(u,t)$ converge para ∞ quando (u,t) se aproxima do fronteira de C.

4.2.3 Função Potencial Primal-Dual

A função potencial primal-dual para um ponto (x,λ,μ) factível em um problema SOCP é definida, de acordo com [12], como:

$$\Phi(x,z,w) = (2N + \nu\sqrt{2N})\log\eta + \sum_{i=1}^{N}(\phi(u_i,t_i) + \phi(z_i;w_i)) - 2N\log N \tag{4.14}$$

onde $\nu \geq 1$ é um parâmetro do algoritmo e η é o gap (2.6) associado a (x,λ,μ). A propriedade mais importante da função potencial é a desigualdade

$$\eta(x,z,w) \leq exp(\Phi(x,z,w)/\nu\sqrt{2N}), \tag{4.15}$$

que ocorre para todo ponto factível (x,z,w).

Desta desigualdade concluímos que se o valor da função potencial é pequeno, então o gap de dualidade também é pequeno e o ponto está próximo do ótimo. Em particular, se $\Phi \to -\infty$, então $\eta \to 0$ e (x,z,w) aproxima-se da otimalidade.

A desigualdade da função potencial (4.15) pode ser facilmente verificada notando que a função $\psi(x,z,w)$, definida por:

$$\psi(x,z,w) = 2N\log\eta + \sum_{i=1}^{N}(\phi(u_i,t_i) + \phi(z_i;w_i)) - 2N\log N,$$

é positiva para todo ponto (x,z,w) factível. Daí segue que

$$\Phi(x,z,w) \geq \nu\sqrt{2N}\log(\eta(x,z,w)),$$

e concluímos a desigualdade (4.15).

4.2.4 Algoritmo Primal-Dual de Redução Potencial

Em um método primal-dual de redução potencial, começamos com pontos factíveis do primal e do dual x, z, w e os atualizamos de tal maneira que a função potencial $\Phi(x, z, w)$ é reduzida a cada iteração ao menos uma certa quantia garantida. Existem diversas variações desta idéia. Nesta seção apresentamos o algoritmo redução potencial primal-dual de Nesterov e Nemirovski [19].

A cada iteração do método de Nesterov e Nemirovski, as direções de busca primal e dual $\Delta x, \Delta\lambda$ e $\Delta\mu$ são calculadas resolvendo-se o conjunto de equações abaixo:

$$\begin{bmatrix} H^{-1} & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} \Delta Z \\ \Delta x \end{bmatrix} = \begin{bmatrix} H^{-1}(\gamma Z + g) \\ 0 \end{bmatrix} \tag{4.16}$$

nas variáveis Δx e ΔZ, onde $\gamma,\ H,\ g,\ Z$ e ΔZ são definidos por

$$\gamma = \frac{(2N + \nu\sqrt{2N})}{\eta},$$

$$H = \begin{bmatrix} \nabla^2\phi(u_1, t_1) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \nabla^2\phi(u_N, t_N) \end{bmatrix},$$

$$g = \begin{bmatrix} \nabla\phi(u_1, t_1) \\ \vdots \\ \nabla\phi(u_N, t_N) \end{bmatrix},$$

$$Z = [z_1^T w_1 \dots z_N^T w_N]^T,\ \Delta Z = [\Delta z_1^T \Delta w_1 \dots \Delta z_N^T \Delta w_N]^T.$$

Algoritmo Primal-Dual de Redução Potencial

Dados x, λ, μ estritamente factíveis, uma tolerância $\epsilon > 0$ e um parâmetro $\nu \geq 1$.

Repita

1. Encontre direções de busca primal e dual resolvendo a equação (4.16).

2. Encontre $p,\ q \in \mathbb{R}$ que minimizem $\Phi(x + p\delta x,\ z + q\delta z,\ w + q\delta w)$.

3. Atualize. $x := x + p\delta x,\ z := z + q\delta z,\ w := w + q\delta w$.

Até $\eta(x,\ z,\ w) \leq \epsilon$.

A função potencial decresce ao menos um valor fixo [12], i.e.,

$$\Phi(x^{(k+1)},\ z(k+1),\ w^{(k+1)}) \leq \Phi(x^{(k)},\ z^{(k)},\ w^{(k)}) - \delta \tag{4.17}$$

onde δ independe de qualquer dado do problema.

4.2.5 Encontrando pontos iniciais estritamente factíveis

O algoritmo da seção anterior requer pontos iniciais primal e dual estritamente factíveis. Nesta seção discutiremos duas técnicas que podem ser usadas quando não se conhece pontos primais e duais factíveis.

Limites nas variáveis primais

Geralmente é fácil encontrar pontos duais factíveis em SOCPs quando as restrições primais incluem limites explícitos nas variáveis. Por exemplo, suponha que modificamos o SOCP (3.1) adicionando um limite na norma de x,

$$\begin{array}{ll} \text{Minimizar} & f^T x \\ \text{sujeita a} & \|A_i x + b_i\| \leq {c_i}^t x + d_i, \quad i = 1, ...N, \\ & \|x\| \leq R. \end{array} \tag{4.18}$$

Se R é grande o suficiente, a restrição extra não muda a solução, nem o valor ótimo do SOCP. O dual do SOCP (4.18) é

$$\begin{array}{ll} \text{Maximizar} & -\left(\sum_{i=1}^{N} (b_i^T z_i + d_i w_i) + R w_{N+1}\right) \\ \text{sujeita a} & f + \sum_{i=1}^{N} (A_i^T z_i + c_i w_i) + z_{N+1} = 0 \\ & \|z_i\| \leq w_i \quad i = 1, ...N. \end{array} \tag{4.19}$$

Pontos estritamente factíveis para o problema (4.19) podem ser facilmente calculados. Para $i = 1, ..., N$, podemos escolher quaisquer z_i e tomar $w_i > \|z_i\|$. A variável z_{N+1} segue da restrição de igualdade do problema (4.19) e w_{N+1} pode ser escolhida entre qualquer número maior que $\|z_{N+1}\|$.

Método Fase 1

Um ponto primal factível pode ser encontrado resolvendo o SOCP:

$$\begin{array}{ll} \text{Minimizar} & t \\ \text{sujeita a} & \|A_i x + b_i\| \leq {c_i}^t x + d_i + t, \quad i = 1, ...N, \end{array} \tag{4.20}$$

nas variáveis x e t. Se $(x,\ t)$ é factível no problema (4.20), e $t < 0$, então x satisfaz $\|A_i x + b_i\| \leq {c_i}^t x + d_i$. Em outras palavras, x é estritamente factível para o problema original (3.1). Podemos então encontrar um ponto estritamente factível x, resolvendo o SOCP (4.20), e exigindo que o valor ótimo t^* seja negativo. É facil verificar que se $t^* > 0$, o SOCP original (3.1), é infactível.

Note que é fácil encontrar um ponto estritamente factível para o SOCP (4.20). Uma possibilidade de escolha é:

$$x = 0, \quad t > \max_i \|b_i\| - d_i.$$

Se temos um ponto factível para o dual do problema acima, podemos resolvê-lo pelo método primal-dual de redução potencial. Caso contrário, podemos adicionar limites explícitos nas variáveis primais como descrito anteriormente.

4.2.6 Método Seguidor de Caminho para SOCP com Escalamento de Nesterov-Todd

Vamos descrever agora um outro método específico para resolução de problemas SOCP. Para este método, inicialmente, consideraremos um problema SOCP simplificado:

$$\begin{array}{ll} \text{Minimizar} & f^T x \\ \text{sujeita a} & \|Ax\| \leq d, \\ & Fx = g. \end{array} \tag{4.21}$$

Então novamente associando ao cone C a ordenação parcial em $\mathbb{R}^n$ definida por $x \geq_C y \Leftrightarrow x - y \in C$, vamos reformular o problema (4.21),

$$\begin{array}{ll} \text{Minimizar} & f^T x \\ \text{sujeita a} & Ax \leq_C d, \\ & Fx = g. \end{array} \tag{4.22}$$

O dual do problema SOCP (4.22) é:

$$\begin{array}{ll} \text{Maximizar} & -d^T \mu - g^T \lambda \\ \text{sujeita a} & A^T \mu + F^T \lambda + f = 0, \\ & \mu \geq_C 0. \end{array} \tag{4.23}$$

As condições KKT (4.2) do problema (4.22) são:

$$\begin{array}{l} f + \lambda F + \mu A = 0, \\ Fx = g, \\ Ax \leq_C d, \\ diag(\mu)(Ax - d) = 0. \\ \mu \geq_C 0. \end{array} \tag{4.24}$$

O caminho central do problema (4.22) é definido pelo conjunto de pontos que satisfazem o sistema KKT modificado (4.3),

$$\begin{array}{c} f + \lambda F + \mu A = 0, \\ Fx = g, \\ Ax - d + s = 0, \\ diag(\mu)s - \tau e = 0, \\ (s, \mu) >_C 0, \\ \tau > 0. \end{array} \tag{4.25}$$

Reescrevendo as equações (4.25) na forma matricial, temos:

$$\begin{gathered}\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} + \begin{bmatrix} 0 & F^T & A^T \\ F & 0 & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \\ \mu \end{bmatrix} = \begin{bmatrix} -f \\ g \\ d \end{bmatrix}, \\ diag(\mu)s = \tau e, \\ (s,\mu) >_C 0, \\ \tau > 0. \end{gathered} \tag{4.26}$$

Definição 4.2.1. *Um escalamento primal-dual W é uma transformação linear*

$$\tilde{s} = W^{-T}s, \;\; \tilde{\mu} = W\mu,$$

que mantém o cone e o caminho central invariantes, isto é,

$$\begin{gathered} s >_C 0 \Leftrightarrow \tilde{s} >_C 0, \qquad \mu >_C 0 \Leftrightarrow \tilde{\mu} >_C 0, \\ diag(\mu)s = \tau e \Leftrightarrow diag(\tilde{\mu})\tilde{s} = \tau e. \end{gathered}$$

Se W é um escalamento, podemos escrever as equações do caminho central (4.26) equivalentemente como:

$$\begin{gathered}\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} + \begin{bmatrix} 0 & F^T & A^T \\ F & 0 & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \\ \mu \end{bmatrix} = \begin{bmatrix} -f \\ g \\ d \end{bmatrix}, \\ diag(w_\mu)w_s = \tau e, \\ (s,\mu) >_C 0, \\ \tau > 0. \end{gathered} \tag{4.27}$$

onde $w_\mu = W\mu$ e $w_s = W^{-T}s$.

Propriedades da função barreira logarítmica (4.13) fornecem um procedimento para construir um escalamento primal-dual .

O algoritmo de Nesterov-Todd, a seguir, é baseado nas equações de caminho central (4.27), depois de ser aplicado ao problema um escalamento com uma matriz W obtida a partir da Hessiana da função barreira (4.13). Para mais detalhes vide [21], [20]

Algoritmo Seguidor de Caminho para SOCP

Dados (s, x, λ, μ).

1. Compute
$$\begin{bmatrix} r_x \\ r_\lambda \\ r_\mu \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} - \begin{bmatrix} 0 & F^T & A^T \\ F & 0 & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \\ \mu \end{bmatrix}$$
Pare se $(s,\ x,\ \lambda,\ \mu)$ satisfizer (aproximadamente) as equações KKT (4.26).

2. Resolva o sistema linear
$$\begin{bmatrix} 0 \\ 0 \\ \Delta s_a \end{bmatrix} - \begin{bmatrix} 0 & F^T & A^T \\ F & 0 & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x_a \\ \Delta \lambda_a \\ \Delta \mu_a \end{bmatrix} = - \begin{bmatrix} r_x \\ r_\lambda \\ r_\mu \end{bmatrix}$$

3. Compute
$$\alpha = sup\,\{\alpha \in [0,\ 1] \mid (s, \mu) + \alpha(s_a, \mu_a)\}$$
$$\sigma = (1 - \alpha)^3.$$

4. Resolva o sistema linear:
$$\begin{bmatrix} 0 \\ 0 \\ \Delta s \end{bmatrix} - \begin{bmatrix} 0 & F^T & A^T \\ F & 0 & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta \mu \end{bmatrix} = -(1 - \sigma) - \begin{bmatrix} r_x \\ r_\lambda \\ r_\mu \end{bmatrix}$$

5. Atualize.
$$(s, x, \lambda, \mu) = (s, x, \lambda, \mu) + \alpha(\Delta s, \Delta x, \Delta \lambda, \Delta \mu)$$
onde $\alpha = sup\,\{\alpha \in [0,\ 1] \mid (s, \mu) + \alpha(s_a, \mu_a)\}$

Compute a matriz de escalameto W para s e μ.

Inicialização

Quando os pontos iniciais ($x_0,\ s_0,\ \lambda_0,\ \mu_0$) não são especificados pelo usuário, então eles são selecionados da seguinte forma. A variável primal incial será a solução do problema de quadrados mínimos:

$$\begin{array}{rl} \text{Minimizar} & \|Ax - b\| \\ \text{sujeita a} & c^T x = d. \end{array}$$

O valor inicial de s_0 é calculado do resíduo $\tilde{s} = Ax_0 - b$, como segue

$$s_0 = \begin{cases} \tilde{s} & \text{se } \alpha_p < 0, \\ \tilde{s} + (1 + \alpha_p)\, e & \text{caso contrário,} \end{cases}$$

onde $\alpha_p = \inf\{\alpha \mid \tilde{s} + \alpha e \geq 0\}$. Os valores x_0 e $\tilde{s}$ podem ser calculados através do sistema linear:

$$\begin{bmatrix} 0 & F^T & A^T \\ F & 0 & 0 \\ A & 0 & -I \end{bmatrix} \begin{bmatrix} x_0 \\ \lambda \\ -\tilde{s} \end{bmatrix} = \begin{bmatrix} 0 \\ d \\ b \end{bmatrix}.$$

As variáveis duais iniciais λ_0 e μ_0 são calculadas resolvendo o problema de norma mínima:

$$\begin{array}{rl} \text{Minimizar} & \|\mu\| \\ \text{sujeita a} & A^T\mu + c^T\lambda + f = 0. \end{array} \tag{4.28}$$

Se a solução é λ_0 e $\tilde{\mu}$, então usamos λ_0 como valor inicial de λ, e

$$\mu_0 = \begin{cases} \tilde{\mu} & \text{se } \alpha_p < 0, \\ \tilde{\mu} + (1 + \alpha_p)\, e & \text{caso contrário,} \end{cases}$$

onde $\alpha_p = \inf\{\alpha \mid \tilde{\mu} + \alpha e \geq 0\}$, como valor inicial de μ. O problema de norma mínima (4.28) é equivalente ao sistema linear

$$\begin{bmatrix} 0 & F^T & A^T \\ F & 0 & 0 \\ A & 0 & -I \end{bmatrix} \begin{bmatrix} x_0 \\ \lambda_0 \\ -\tilde{\mu} \end{bmatrix} = \begin{bmatrix} -c \\ 0 \\ 0 \end{bmatrix}.$$

Desta forma, podemos encontrar os valores λ_0 e $\tilde{\mu}$ resolvendo este sistema linear.

CAPÍTULO 5

TESTES NUMÉRICOS

Este capítulo apresenta experimentos numéricos realizados com o objetivo de avaliar a eficiência do uso de um software específico para problemas SOCP, quando comparado a um software genérico para resolução de problemas de otimização.

A avaliação teve como foco quatro problemas preliminares, de pequenas dimensões, um problema teórico, com dimensões variáveis, e duas aplicações reais. O software específico utilizado para execução de problemas SOCP foi o CVXOPT *Python Software for Convex Optimization* [27], enquanto a função `fmincon` *Find Minimum of Constrained Nonlinear Multivariable Function* [15], nativa do Matlab [16], foi utilizada como o software genérico.

Os problemas preliminares e o problema teórico são mostrados de duas formas diferentes, primeiro em seus formatos originais, em seguida no seu formato SOCP. Estes problemas são solucionados de três maneiras distintas: formato original em `fmincon`, formato SOCP em `fmincon` e formato SOCP em CVXOPT. As aplicações reais são executadas sempre em seu formato SOCP, tanto em `fmincon` quanto em CVXOPT.

Nestas simulações, avaliamos a quantidade de testes bem sucedidos, o valor da função objetivo e a quantidade de iterações. Como se tratam de dois softwares distintos - Matlab e Python, para avaliar o tempo utilizamos uma padronização. Todos os valores de tempo foram divididos pelo menor valor de tempo de cada software, sendo assim, o menor tempo,

tanto do `fmincon` como do CVXOPT, estão representados por 1.

Foram utilizados o mesmo ponto inicial, vetor nulo, e os mesmos critérios de parada nos dois softwares: gap relativo: 10^{-6}; tolerância das restrições não lineares: 10^{-6}; máximo de iterações: 5.000, no problema QCQP, 25.000 na aplicação de arranjo de antenas e 25.000 na aplicação garra de múltiplos dedos (GRASP). Em CVXOPT zeramos o critério de gap absoluto.

Os experimentos foram realizados em um computador com processador Intel Core i5 750 (2.67GHz) com 4GB de RAM.

5.1 Software utilizado

5.1.1 CVXOPT - *Python Software for Convex Optimization*

O estudo recente de problemas SOCP levou à criação de algumas ferramentas de software para resolução dos mesmos. Dentre elas, destacam-se SeDuMi [25], desenvolvido por Jos F. Sturm, mantido pelo Advanced Optimization Laboratory; CVX *Matlab Software for Disciplined Convex Programming* [9] e CVXOPT.

O CVXOPT é um pacote de software livre para otimização convexa, baseado na linguagem de programação Python [22]. Possui uma grande gama de funcionalidades, incluindo resolução de SOCP. Problemas são definidos em scripts escritos em Python, onde são especificados exatamente o método e os parâmetros a serem utilizados. Foi desenvolvido por Joachim Dahl e Lieven Vandenberghe. Ativamente mantido: última versão de Setembro de 2010.

O algoritmo de pontos interiores de CVXOPT utiliza o método seguidor de caminho com escalamento de Nesterov-Todd descrito na seção 4.2.6. Mais detalhes, vide os artigos [26] e [2].

O CVX é um sistema de modelagem para *programação convexa disciplinada*. Programas convexos disciplinados são problemas de otimização convexa descritos por um conjunto limitado de regras de construção, as quais permitem que eles sejam analisados e resolvidos de forma eficiente. O CVX é implementado em Matlab, pode resolver problemas padrão,

tais como programas lineares, programas quadráticos, SOCPs e programação semidefinida. Especificações do modelo são construídas utilizando operações e funções comuns de Matlab. O CVX converte o problema especificado em um LP, SOCP ou SDP, quando o conjunto de todas as funções do problema puder ser representado em uma dessas formas. Porém a escolha de como o problema será resolvido não pode ser feita pelo usuário. Também em ativo desenvolvimento, sua última versão é de Fevereiro de 2011.

Motivação para escolha do CVXOPT: é um software com uma opção específica para executar problemas no formato SOCP.

5.1.2 `fmincon` - *Find Minimum of Constrained Nonlinear Multivariable Function*

O `fmincon` é uma função genérica do Matlab para resolução de problemas de otimização. Ele busca encontrar um mínimo de um problema de programação não linear a partir de uma estimativa inicial.

Foi escolhido por se tratar de uma ferramenta de fácil utilização e bastante difundida.

Utilizamos o método de pontos interiores por mais se aproximar do método utilizado no software CVXOPT. O algoritmo de pontos interiores de `fmincon` utiliza o método de barreira descrito na seção 4.1.4. Mais detalhes, vide os artigos [6], [7] e [30]. Dentre as opções de cálculo de Hessiana, escolhemos a *user-supplied*, assim inserimos o gradiente da função objetivo e das restrições e a Hessiana.

5.2 Problemas Preliminares

5.2.1 Exemplo 1 - Programação Linear

Formato Original

$$\begin{aligned} \text{Minimizar} \quad & x_1 + x_2 \\ \text{sujeita a} \quad & -2x_1 - x_2 - 4 \leq 0 \\ & -x_2 \leq 0, \end{aligned} \tag{5.1}$$

Formato SOCP

$$\begin{array}{ll} \text{Minimizar} & x_1 + x_2 \\ \text{sujeita a} & \|0\| \leq 2x_1 + x_2 + 4, \\ & \|0\| \leq x_2. \end{array} \tag{5.2}$$

Resultados experimentais

Ao rodar este exemplo em `fmincon`, tanto no formato original quanto no formato SOCP, obtivemos como resultados $f(x^*) = -1,9999999599964757$ em 12 iterações. Já em CVXOPT obtivemos: $f(x^*) = -2$ e quantidade de iterações nula, pois o resultado encontrado na fase 1 foi o minimizador, não sendo necessárias iterações no algoritmo principal.

5.2.2 Exemplo 2 - Função Objetivo Linear e Restrições Quadráticas

Formato Original

$$\begin{array}{ll} \text{Minimizar} & -2x_1 + x_2 \\ \text{sujeita a} & (x_1 + x_2)^2 + (x_1 + 2)^2 \leq 0 \end{array} \tag{5.3}$$

Formato SOCP

$$\begin{array}{ll} \text{Minimizar} & -2x_1 + x_2 \\ \text{sujeita a} & \left\| \begin{array}{c} x_1 + x_2 \\ x_1 + 2 \end{array} \right\| \leq 0. \end{array} \tag{5.4}$$

Resultados experimentais

Ao rodar este exemplo obtivemos:

- `fmincon` com problema no formato original $\rightarrow f(x^*) = 5,916666766666666$ em 5 iterações,
- `fmincon` com problema no formato SOCP $\rightarrow f(x^*) = 6,000000000315058$ em 61 iterações,

- CVXOPT $\rightarrow f(x^*) = 6$ em 4 iterações.

Todos encontraram o valor ótimo, o melhor resultado foi do CVXOPT, por apresentar uma quantidade baixa de iterações, seguido pelo `fmincon` com problema no formato original, que também apresentou uma quantidade baixa de iterações.

5.2.3 Exemplo 3 - Soma de Normas

Formato Original

$$\text{Minimizar} \left\| \begin{array}{l} x_1 + 3x_3 \\ x_2 + 0,5x_3 + 0,5 \\ 2x_1 - x_2 - 0,5x_3 - 1 \end{array} \right\| + \|2x_1 + 1,5\| \tag{5.5}$$

Formato SOCP

$$\begin{array}{ll} \text{Minimizar} & t_1 + t_2 \\ \text{sujeita a} & \left\| \begin{array}{c} x_1 + 3x_3 \\ x_2 + 0,5x_3 + 0,5 \\ 2x_1 - x_2 - 0,5x_3 - 1 \end{array} \right\| \leq t_1 \\ & \|2x_1 + 1,5\| \leq t_2 \end{array} \tag{5.6}$$

Resultados experimentais

Os resultados obtidos neste exemplo foram:

- `fmincon` com problema no formato original$\rightarrow$ $f(x^*) = 1,4142135623891974$ em 30 iterações,
- `fmincon` com problema no formato SOCP $\rightarrow$ $f(x^*) = 1,4142143698236727$ em 40 iterações,
- CVXOPT $\rightarrow f(x^*) = 1,41421355688$ em 5 iterações.

Neste exemplo todas as abordagens encontraram o mínimo e CVXOPT teve um melhor resultado apresentando uma quantidade de iterações bem baixa em relação ao `fmincon`.

5.2.4 Exemplo 4 - QCQP (*Quadratically Constrained Quadratic Programming*)

Neste exemplo utilizamos as transformações algébricas descritas na seção 3.3, subseção 3.3.3.

Formato Original

$$\text{Minimizar} \quad [x_1 \;\; x_2 \;\; x_3 \;\; x_4] \begin{bmatrix} 10 & 10 & 0 & 0 \\ 10 & 50 & 0 & 0 \\ 0 & 0 & 26 & 25 \\ 0 & 0 & 25 & 25 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + 2x_1 - 6x_2 + 2x_3 - 2x_4$$

$$\text{sujeita a} \quad [x_1 \;\; x_2 \;\; x_3 \;\; x_4] \begin{bmatrix} 29 & 12 & 0 & 0 \\ 12 & 5 & 0 & 0 \\ 0 & 0 & 20 & 12 \\ 0 & 0 & 12 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + 2x_1 + 4x_2 - 10 \leq 0 \qquad (5.7)$$

$$[x_1 \;\; x_2 \;\; x_3 \;\; x_4] \begin{bmatrix} 10 & 10 & 0 & 0 \\ 10 & 50 & 0 & 0 \\ 0 & 0 & 29 & 12 \\ 0 & 0 & 12 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + 2x_4 - 5 \leq 0$$

Formato SOCP

$$\begin{aligned}
&\text{Minimizar} \quad x_5 \\
&\text{sujeita a} \quad \left\| \begin{array}{c} 3x_1 + x_2 + 0,5 \\ x_1 + 7x_2 - 0,5 \\ 3,9691x_3 + 3,2009x_4 + 1,4084 \\ 3,2009x_3 + 3,8411x_4 - 1,434 \end{array} \right\| \leq x_5 \\
&\qquad\qquad \left\| \begin{array}{c} 5x_1 + 2x_2 - 3 \\ 2x_1 + x_2 + 8 \\ 4x_3 + 2x_4 \\ 2x_3 + 2x_4 \end{array} \right\| \leq 9,11 \qquad (5.8) \\
&\qquad\qquad \left\| \begin{array}{c} 3x_1 + x_2 \\ x_1 + 7x_2 \\ 5x_3 + 2x_4 - 2 \\ 2x_3 + x_4 + 5 \end{array} \right\| \leq 5,83
\end{aligned}$$

Resultados experimentais

Neste problema há uma mudança no valor da função objetivo. Para comparar os valores ótimos, efetuamos $x_5^2 + 4,54$, onde 4,54 foi encontrado através do cálculo,

$$[1 \quad -3 \quad 1 \quad -1] \begin{bmatrix} 10 & 10 & 0 & 0 \\ 10 & 50 & 0 & 0 \\ 0 & 0 & 26 & 25 \\ 0 & 0 & 25 & 25 \end{bmatrix} \begin{bmatrix} 1 \\ -3 \\ 1 \\ -1 \end{bmatrix}.$$

Os resultados obtidos foram:

- `fmincon` com problema no formato original $\rightarrow f(x^*) = -2,6023779651644223$ em 6 iterações,
- `fmincon` com problema no formato SOCP $\rightarrow f(x^*) = -2,602378025194515$ em 8 iterações,

- CVXOPT $\rightarrow f(x^*) = -2,60251460936$ em 7 iterações.

Neste exemplo as três abordagens foram parecidas.

Conclusões

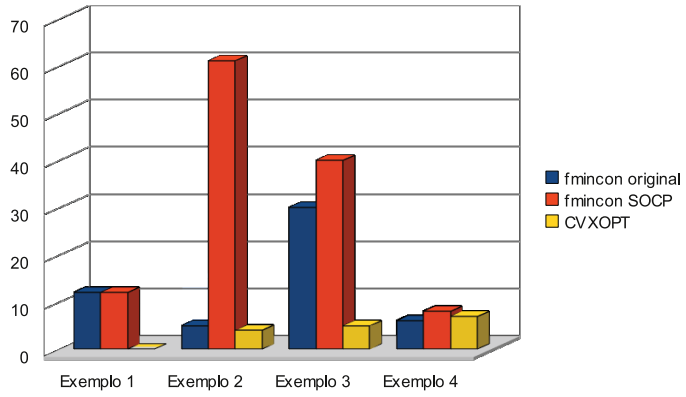


Figura 5.1: *Número de iterações dos problemas preliminares.*

Mostramos na Figura 5.1 a comparação da quantidade de iterações em cada exemplo. Ao analisar este gráfico, percebemos que o CVXOPT sempre apresenta uma quantidade baixa de iterações e apenas no exemplo 4 esta abordagem não é a melhor com relação às iterações. Este fato motivou um estudo mais detalhado de problemas QCQP no formato SOCP. Apresentamos este estudo na próxima seção.

5.3 Problemas Teóricos

5.3.1 QCQP

Nesta seção apresentamos problemas QCQP com variações de dimensão de matrizes e de quantidade de restrições. Geramos problemas QCQP teóricos, com objetivo de comparar os problemas em seus formatos originais e no formato SOCP.

Foram avaliados 1080 problemas, com 2, 4, 6, 8, 10 ou 12 restrições. Para cada restição variamos as dimensões de x em 2, 4, 6, 8, 10 e 12. Utilizamos 30 sementes diferentes para

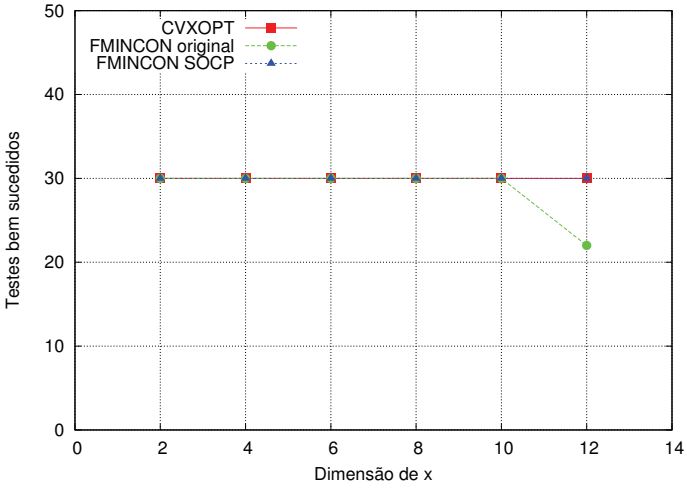


Figura 5.2: *Número de testes bem sucedidos X dimensão n de x para os problemas QCQP com número de iterações = n.*

gerar todos os dados aleatoriamente. Mostramos o desempenho dos softwares através de gráficos.

Convergência

Começamos com a análise da convergência. A Figura 5.2 nos mostra problemas que possuem quantidade de restrições igual a dimensão de x. Para estes casos, CVXOPT e `fmincon` com problema no formato SOCP realizaram todas as execuções com sucesso, já `fmincon` usando o formato original não obteve convergência em alguns problemas com 12 restrições. A Figura 5.3 apresenta o número de testes bem sucedidos de todos os problemas com variações de restrições e dimensões. Podemos observar pela Figura 5.3(b) que CVXOPT convergiu em todos os problemas; `fmincon` usando o formato SOCP falhou em alguns problemas com dimensões maiores ou iguais a 8 e `fmincon`, usando o formato original falhou em alguns problemas de todas as dimensões. Concluímos que CVXOPT foi o mais robusto.

Iterações

Para análise da quantidade de iterações, utilizamos apenas os dados dos problemas que obtiveram testes bem sucedidos. Primeiro analisamos problemas que possuem quantidade de restrições igual a dimensão de x. Através da Figura 5.4 verificamos que para estes casos `fmincon` usando o formato original, apresenta um desempenho muito abaixo dos demais, os quais, por sua vez, têm um comportamento parecido. Já analisando a Figura 5.5, que nos mostra todas as variações de restrições e dimensões, podemos perceber que existe uma diferença significativa na quantidade de iterações realizadas por `fmincon`, formato SOCP, e por CVXOPT. Este último tem o melhor desempenho.

Tempo

Para fazer a análise do tempo, usamos a padronização descrita no início deste capítulo e novamente apenas os dados dos problemas que obtiveram testes bem sucedidos. Através da Figura 5.6 analisamos os problemas que possuem quantidade de restrições igual a dimensão de x e assim verificamos que para estes problemas `fmincon` usando o formato original apresenta um desempenho muito abaixo dos demais, que apresentam um comportamento parecido. Já analisando o gráfico 5.7, podemos perceber que ao variarmos a dimensão nas diversas restrições, `fmincon` usando o formato SOCP apresenta uma diferença significativa na quantidade de iterações quando comparado a CVXOPT. Este último tem o melhor desempenho.

5.4 Problemas Reais

5.4.1 Arranjo de Antenas

Descrição do problema

Em um arranjo de antenas, as emissões de vários sensores são linearmente combinadas para produzir uma emissão mais apurada. A saída do arranjo tem um padrão direcional, que depende dos pesos relativos ou fatores escalares utilizados no processo combinado. O objetivo é estabalecer os pesos que satisfaçam os requisitos do feixe de ondas específico a ser emitido de maneira ótima.

Consideremos o modelo mais simples, um arranjo onidirecional de antenas em um plano, nas posições $(x_j, y_j),\ j = 1, ...n$. Uma única onda, de freqüência ω, incide com ângulo θ. Assumimos que o comprimento de onda é $\lambda = 2\pi$. Esta onda incidente induz no j-ésimo sensor um sinal $exp(i(x_j cos\theta + y_j sen\theta - \omega t))$, onde $i = \sqrt{-1}$. Este sinal é demodulado, isto é, multiplicado por $e^{i\omega t}$ e então multiplicado por um fator complexo $w_j \in \mathbb{C}$. Desta forma temos,

$$\begin{aligned} y_j &= w_j exp(i((x_j cos\theta + y_j sen\theta)) \\ &= (w_{re,j} cos\gamma_j(\theta) - w_{im,j} sen\gamma_j(\theta)) + i(w_{re,j} sen\gamma_j(\theta) + w_{im,j} cos\gamma_j(\theta)), \end{aligned}$$

onde $\gamma_j(\theta) = x_j cos\theta + y_j sen\theta$. A saída do arranjo de antenas é a soma das saídas ponderadas dos sensores:

$$y(\theta) = \sum_{j=1}^{n} y_j(\theta),$$

Para um dado conjunto de pesos, estas saídas combinadas são uma função do ângulo de chegada θ da onda. O objetivo do problema é selecionar os pesos w_i que alcancem uma direção padrão $y(\theta)$ desejável.

A propriedade fundamental é que, para qualquer θ, $y(\theta)$ é uma função linear do vetor peso w. Esta propriedade é verdadeira para uma classe muito ampla de problemas de arranjos e utilizaremos o caso $y(\theta) = a(\theta)w$ para algum vetor linha complexo $a(\theta)$.

Como exemplo de problema de um projeto simples, podemos utilizar a normalização $y(\theta_t) = 1$, onde θ_t é chamado de direção de destino. Se tivermos como objetivo fazer um arranjo relativamente insensível a ondas provenientes de outras direções, teremos $|\theta - \theta_t| \geq \Delta$, onde 2Δ é chamado de largura de feixe do padrão.

Para minimizar a maior sensibilidade de ondas fora do feixe, resolvemos o problema:

$$\begin{array}{ll} \text{Minimizar} & \max\limits_{|\theta-\theta_t|>\Delta} |y(\theta)| \\ \text{sujeita a} & y(\theta_t) = 1. \end{array} \tag{5.9}$$

Este problema pode ser aproximado por um problema SOCP discretizando o ângulo θ, por exemplo, considerando $\theta_1, ..., \theta_m$, onde $m >> n$. Assumimos que a direção de destino é

um dos ângulos, tomemos $\theta_t = \theta_k$. Podemos expressar o aranjo padrão por:

$$\tilde{y} = Aw$$

onde $\tilde{y} \in \mathbb{C}^{m \times n}$, e

$$\tilde{y} = \begin{bmatrix} y(\theta_1) \\ \vdots \\ y(\theta_m) \end{bmatrix}, \; A = \begin{bmatrix} a(\theta_1) \\ \vdots \\ a(\theta_m) \end{bmatrix}.$$

O problema (5.9) pode ser expresso da seguinte forma:

$$\begin{array}{ll} \text{Minimizar} & t \\ \text{sujeita a} & |y(\theta_i)| \leq t \\ & |\theta_i - \theta_k| > \Delta \\ & y(\theta_k) = 1, \end{array} \tag{5.10}$$

o qual torna-se um SOCP quando é expresso em termos das partes reais e imaginárias das variáveis e dos dados [12].

Resultados experimentais

Geramos 870 problemas, com $n = 4, ..., 32$ sensores e utilizamos 30 sementes diferentes para gerar aleatoriamente os dados. Mostramos o desempenho dos softwares através de gráficos.

Convergência

A Figura 5.8 nos mostra que CVXOPT convergiu em todos os problemas testados, mostrando robustez. Já `fmincon` convergiu para uma quantidade menor de problemas com mais de 15 sensores.

Iterações

Para análise da quantidade de iterações, utilizamos apenas os dados dos problemas que obtiveram testes bem sucedidos. Pela Figura 5.9, podemos perceber que o número de iterações de CVXOPT mantem-se aproximadamente constante e de `fmincon` varia de forma desordenada, sendo sempre maior que no primeiro caso.

Na Figura 5.10, podemos fazer uma leitura mais apurada do número de iterações de CVXOPT. Percebemos que o número de iterações aumenta, mas não ultrapassa 20 iterações.

Tempo

Para fazer a análise do tempo, usamos a padronização descrita no início deste capítulo e novamente apenas os dados dos problemas que obtiveram testes bem sucedidos. Através da Figura 5.11 podemos perceber que o tempo de CPU em CVXOPT mantem-se baixo, quase constante. Já `fmincon` apresenta variação no tempo de processamento a partir de 8 sensores, sendo este tempo sempre maior que o registrado por CVXOPT.

Na Figura 5.12, podemos fazer uma leitura mais apurada do tempo de CPU em CVXOPT. Percebemos que o tempo aumenta, mas não ultrapassa 25 vezes o tempo da primeira iteração ao passo que na Figura 5.11 ultrapassa 9.000 vezes o tempo da primeira iteração.

5.4.2 Garras Múltiplos Dedos - *Grasp*

Descrição do problema

Este problema trata de uma garra múltiplos dedos, também conhecida por mão mecânica, que deve agarrar um objeto de forma estável, o mais suave possível.

Consideramos um corpo rígido segurado por N dedos de um robô. Para simplificar as fórmulas assumimos que o centro de massa do corpo está na origem. Os dedos exercem forças de contato em pontos dados: $p_1, ..., p_N \in \mathbb{R}^3$. O vetor normal à superfície no i-ésimo ponto de contato é dado pelo vetor unitário $v_i \in \mathbb{R}^3$ e a força aplicada no ponto é dada por $F_i \in \mathbb{R}^3$.

Cada força de contato F_i é decomposta em duas partes: uma componente $v_i^T F_i v_i$ normal à superfície, e uma componente $(I - v_i v_i^T)F_i$, a qual é tangente à superfície. Assumiremos que a componente tangente ocorre pelo atrito estático e sua magnitude não pode exceder o produto da magnitude da componente normal com o coeficiente de atrito $\mu > 0$, isto é,

$$\left\| \left(I - v_i v_i^T\right) F_i \right\| \leq \mu v_i^T F_i, i = 1, ..., N. \tag{5.11}$$

Essas restrições são restrições de cone quadrático nas variáveis F_i [12].

Por fim, assumimos que forças e torques externos agem sobre o corpo. Isso é equivalente a uma única força externa F_{ext} agindo na origem (que é o centro de massa do corpo) e um torque externo T_{ext}. O equilíbrio estático do corpo é caracterizado pelas equações lineares

$$\begin{aligned} \sum_{i}^{N} F_i + F_{ext} &= 0, \\ \sum_{i}^{N} p_i \times F_i + T_{ext} &= 0. \end{aligned} \tag{5.12}$$

O objetivo do problema é encontrar forças de contato F_i que satisfaçam as restrições de atrito (5.11), as restrições de equilíbrio estático (5.12), e certos limites nas forças de contato, por exemplo, um limite superior $v_i^T F_i \leq F_{max}$ na componente normal.

É possível selecionar um conjunto particular de forças por um certo critério de otimização. Por exemplo, podemos calcular a pegada mais suave possível, isto é, o conjunto de forças F_i que realiza uma pegada estável e minimiza o máximo da força normal nos pontos de contato, resolvendo o SOCP

$$\begin{aligned} \text{Minimizar} \quad & t \\ \text{sujeita a} \quad & v_i^T F_i \leq t, \quad i = 1, ..., N, \\ & \left\| (I - v_i v_i^T F_i \right\| \leq \mu v_i^T F_i, i = 1, ..., N, \\ & \sum_{i}^{N} F_i + F_{ext} = 0, \\ & \sum_{i}^{N} p_i \times F_i + T_{ext} = 0. \end{aligned} \tag{5.13}$$

Para mais detalhes de otimização da força de uma mão mecânica, vide [29] e [11].

Resultados experimentais

Geramos 870 problemas, com $N = 4, ..., 32$ dedos e utilizamos 30 sementes diferentes para gerar aleatoriamente os dados. Mostramos o desempenho dos softwares através de gráficos.

Convergência

A Figura 5.13 nos mostra que CVXOPT convergiu em todos os problemas testados, mostrando robustez. Já `fmincon` falhou em alguns dos problemas.

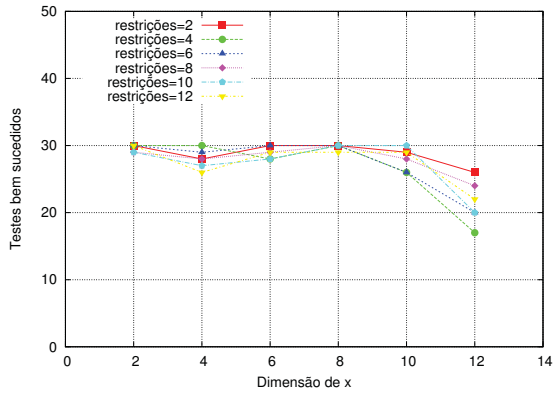
Iterações

Para análise da quantidade de iterações, utilizamos apenas os dados dos problemas que obtiveram testes bem sucedidos. Pela Figura 5.14, podemos perceber que o número de iterações de CVXOPT mantem-se aproximadamente constante, enquanto que em `fmincon` varia de forma desordenada. Outra informação deste gráfico é a diferença entre o número de iterações nos dois softwares: CVXOPT apresenta um número de iterações bem inferior a `fmincon`.

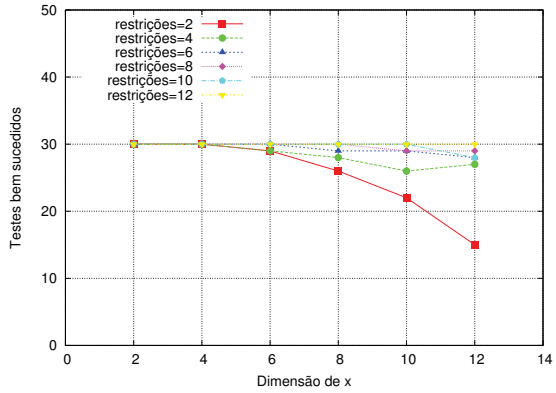
Na Figura 5.15 podemos fazer uma leitura mais apurada do número de iterações de CVXOPT. Percebemos que o número de iterações aumenta, mas não ultrapassa 14 iterações.

Tempo

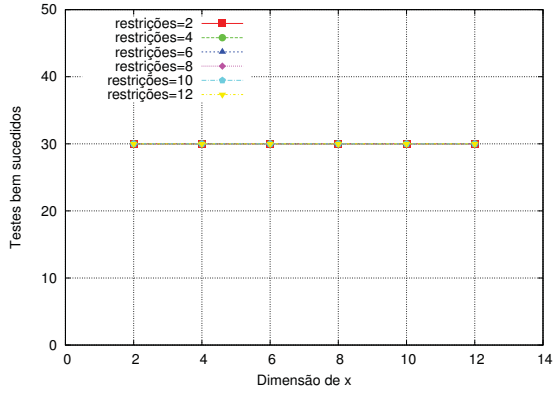
Através da Figura 5.16, podemos perceber que o tempo de CPU registrado por CVXOPT cresce de forma ordenada até 25 dedos, e o de `fmincon` apresenta-se de forma desordenada desde o princípio. Nestes primeiros problemas o tempo de CPU de CVXOPT é menor que o tempo de `fmincon`. No entanto, nos problemas com mais de 25 dedos, o tempo de processamento de CVXOPT tornou-se maior se comparado ao de `fmincon`.



(a)



(b)



(c)

Figura 5.3: *Número de testes bem sucedidos X dimensão n de x para os problemas QCQP com número de iterações = n. Em (a) problema original em* `fmincon`*, (b) problema SOCP em* `fmincon`*, (c) CVXOPT.*

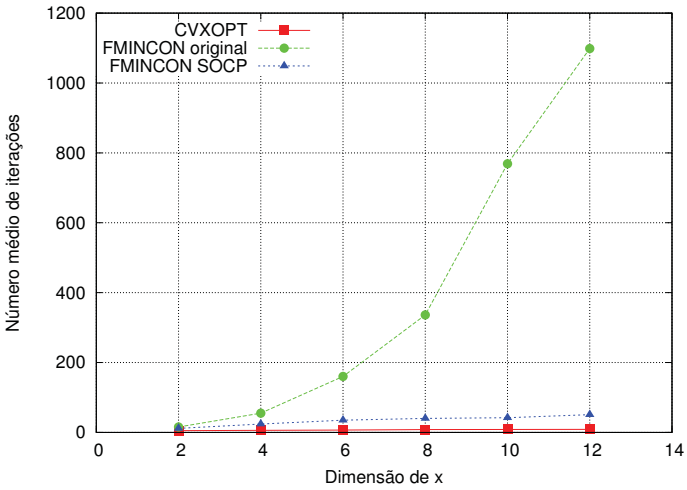
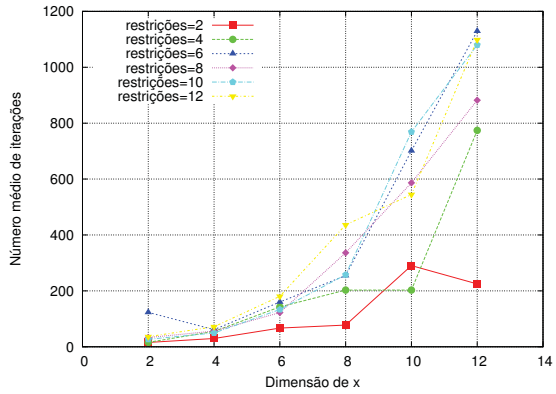
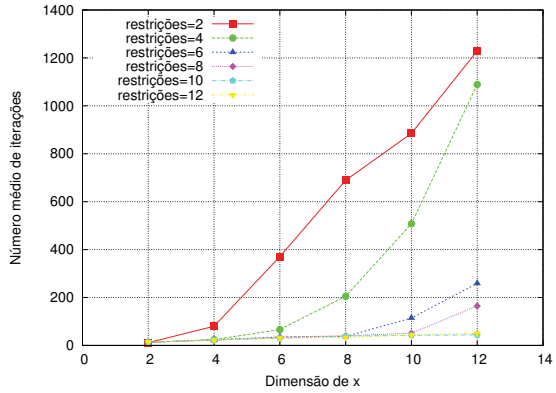


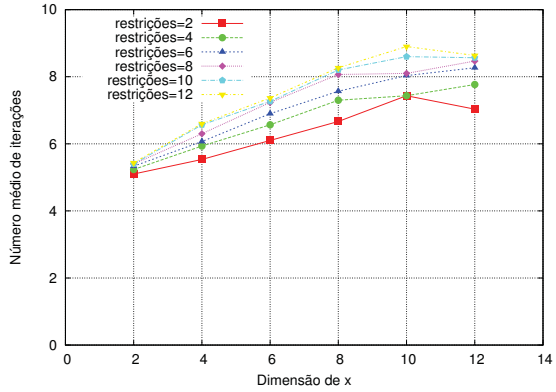
Figura 5.4: *Número de iterações X dimensão n de x para os problemas QCQP com número de iterações = n.*



(a)



(b)



(c)

Figura 5.5: *Número de iterações X dimensão n de x para os problemas QCQP com número de iterações = n. Em (a) problema original em* `fmincon`*, (b) problema SOCP em* `fmincon`*, (c) CVXOPT.*

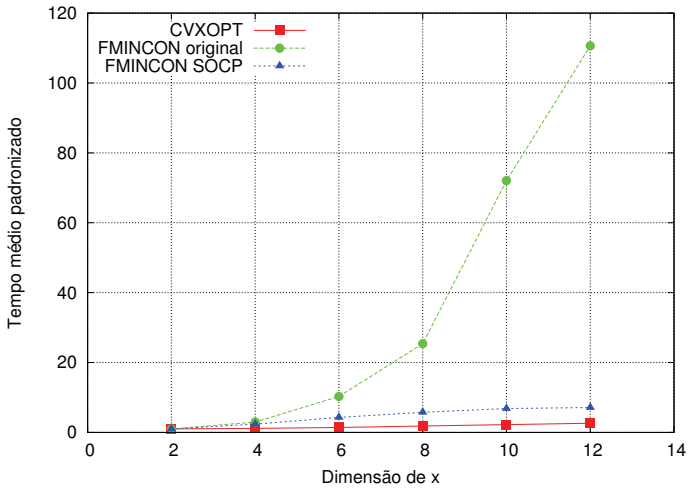
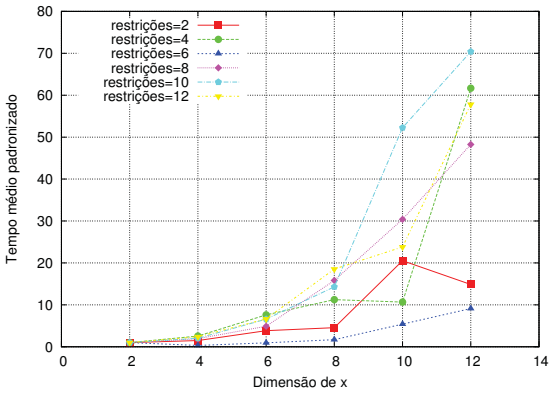
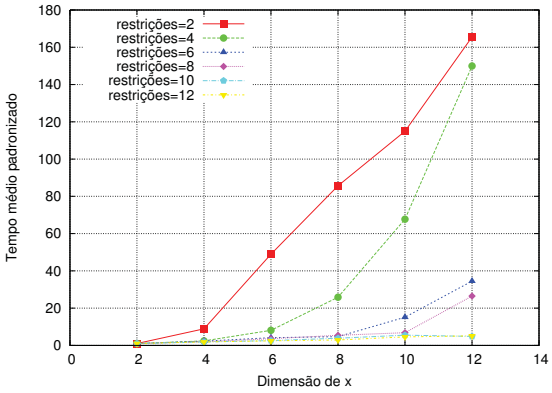


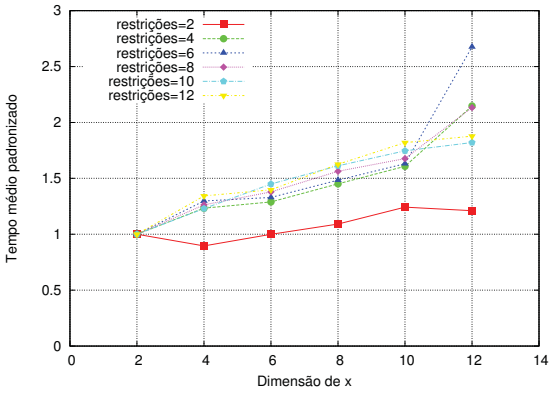
Figura 5.6: *Tempo de processamento X dimensão n de x para os problemas QCQP com número de iterações = n.*



(a)



(b)



(c)

Figura 5.7: *Tempo de processamento X dimensão n de x para os problemas QCQP com número de iterações = n. Em (a) problema original em* `fmincon`*, (b) problema SOCP em* `fmincon`*, (c) CVXOPT.*

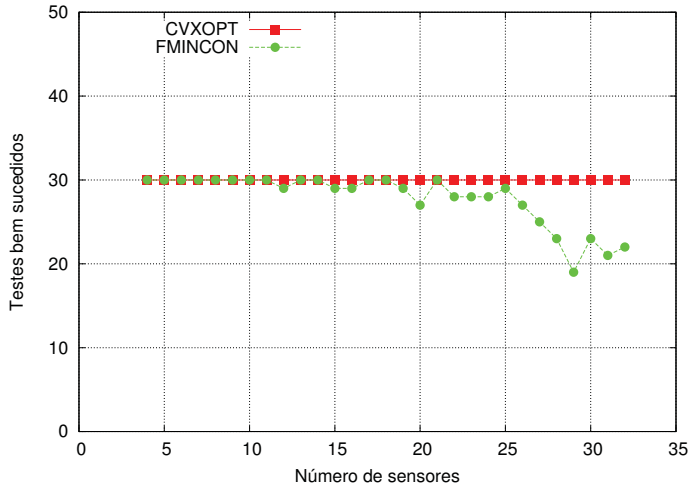


Figura 5.8: *Testes bem sucedidos para os problemas de arranjo de antenas, com variação do número de sensores.*

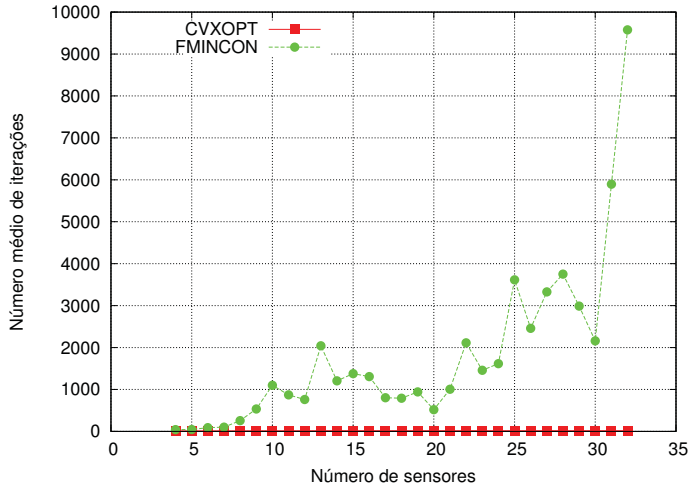


Figura 5.9: *Número de iterações para os problemas de arranjo de antenas com variação do número de sensores.*

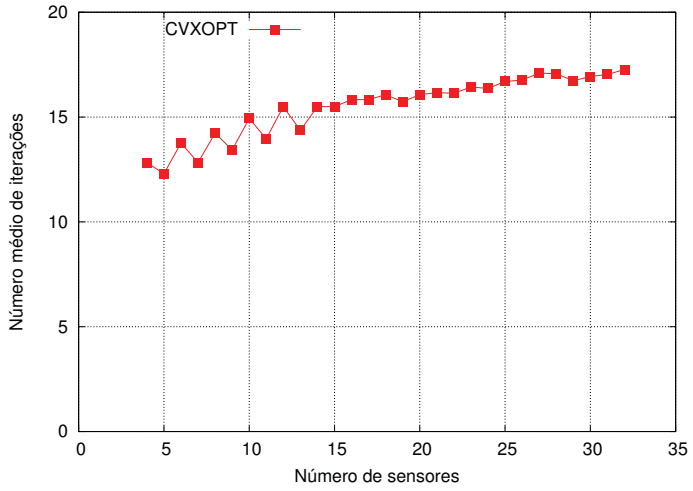


Figura 5.10: *Número de iterações para os problemas de arranjo de antenas com variação do número de sensores.*

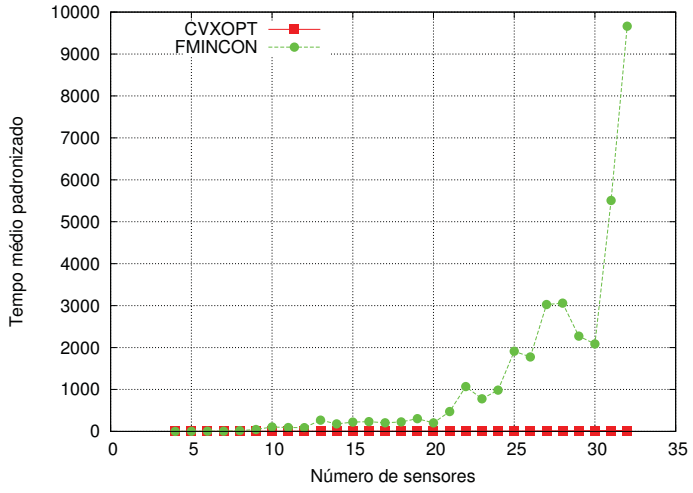


Figura 5.11: *Tempo de processamento para os problemas de arranjo de antenas com variação do número de sensores.*

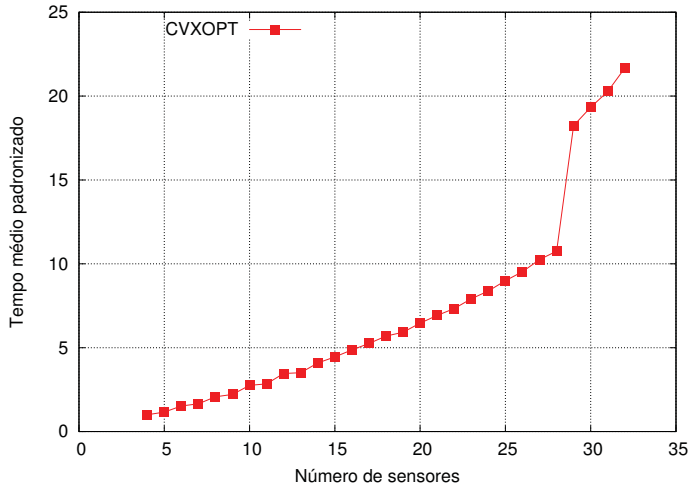


Figura 5.12: *Tempo de processamento para os problemas de arranjo de antenas com variação do número de sensores.*

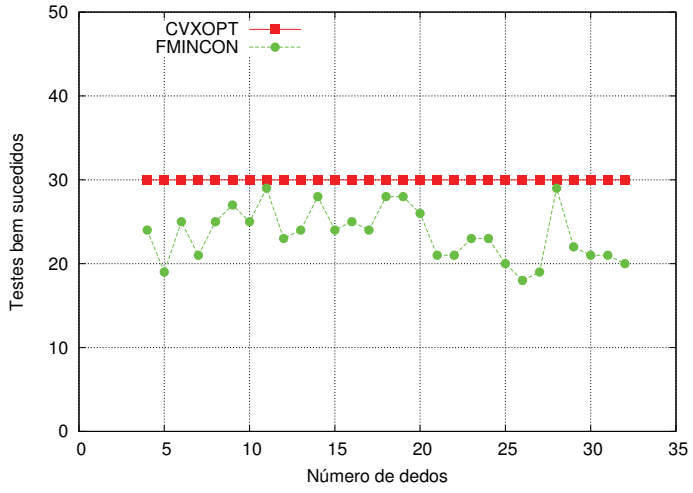


Figura 5.13: *Testes bem sucedidos para os problemas de garras múltiplos dedos, com variação do número de dedos.*

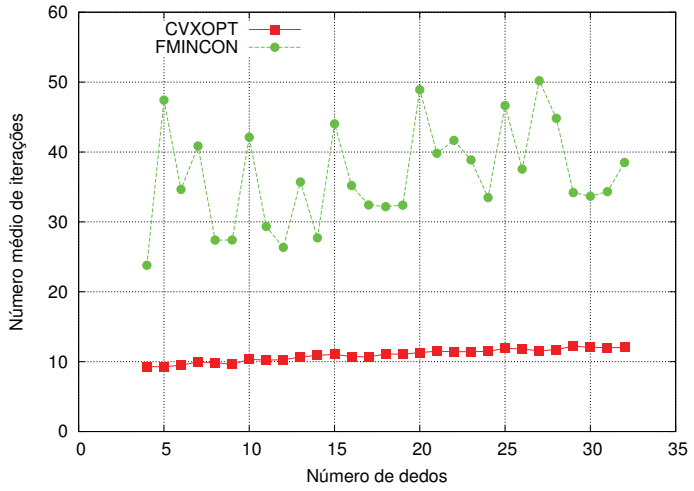


Figura 5.14: *Número de iterações para os problemas de garras múltiplos dedos com variação do número de dedos.*

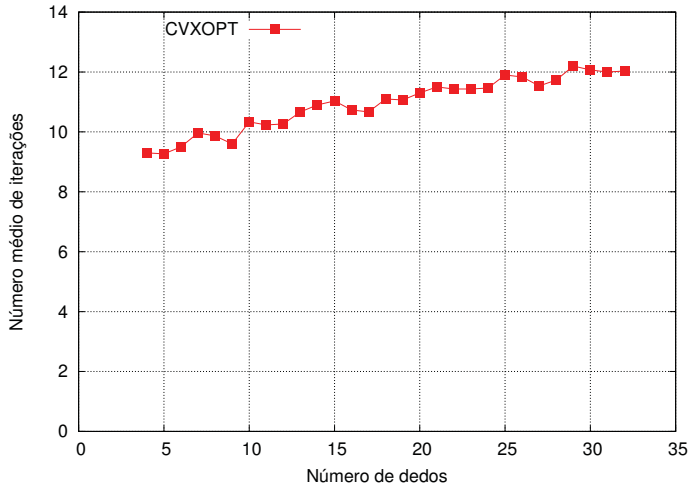


Figura 5.15: *Número de iterações para os problemas de garra múltiplos dedos com variação do número de dedos.*

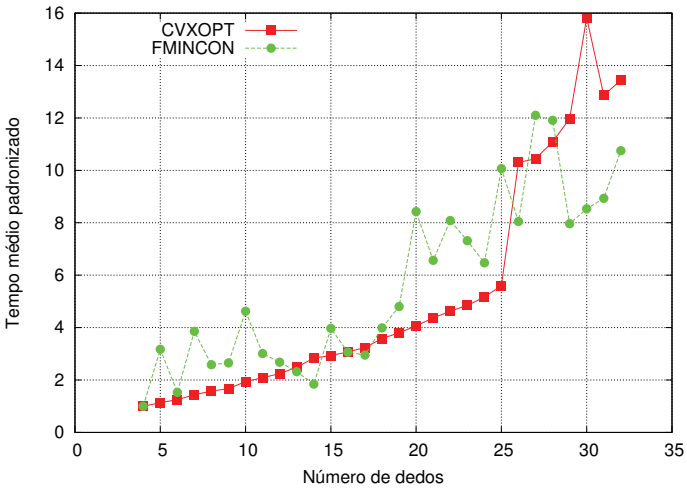


Figura 5.16: *Tempo de processamento para os problemas de garras múltiplos dedos com variação do número de dedos.*

CAPÍTULO 6

CONSIDERAÇÕES FINAIS

Problemas SOCP abrangem diversos problemas de programação matemática como casos especiais e possuem uma formulação simples. Essa dissertação apresentou um estudo teórico sobre problemas de programação não linear e problemas SOCP, além de testes numéricos que possibilitaram uma análise dos algoritmos para problemas SOCP baseando-se tanto em problemas teóricos como práticos.

Foi feito um estudo teórico que analisou a transformação de diversos problemas: soma de normas, máximo de normas, programação linear, QCQP (*Quadratically Constrained Quadratic Programming*), média harmônica, superfície mínima, programação linear robusta e quadrados mínimos robusto em problemas SOCP. Esta análise mostrou que a transformação é simples, o que implica que a escolha do formato pode se basear apenas na facilidade de resolução do problema.

Foi feito também um estudo teórico sobre métodos usados em problemas gerais de programação não linear e métodos usados especificamente em problemas SOCP.

Para os testes numéricos foram analisados 1080 problemas teóricos e 1740 problemas práticos. Os problemas teóricos analisados foram problemas QCQP, que incluem programação linear e programação quadrática como casos especiais. Os práticos foram problemas sobre arranjo de antenas e garras múltiplos dedos. Um arranjo de antena se caracteriza pela utilização

de vários sensores em conjunto para uma captação ou emissão mais apurada (se comparado a uma antena simples) de sinais vindos de diferentes direções. Uma garra múltiplos dedos, também conhecida por mão mecânica, deve agarrar um objeto de forma estável, satisfazendo alguns requisitos.

Nos problemas teóricos foi feita a comparação do formato original, QCQP, com o formato SOCP. Com isso pudemos avaliar experimentalmente a diferença entre otimizar um problema em seu formato original e em seu formato SOCP, utilizando uma ferramanta para problemas gerais de otimização, e a diferença na otimização de um problema SOCP usando uma ferramenta genérica e usando uma ferramenta específica para SOCP. Utilizamos os solvers `fmincon`, nativo do MatLab, como ferramenta genérica e o CVXOPT, biblioteca do Python, como ferramenta específica para SOCP. Os resultados obtidos foram analisados de uma maneira detalhada.

A análise ficou concentrada nos requisitos robustez, número de iterações e variação do tempo de CPU com o aumento da dimensão dos problemas. Observou-se que a ferramenta específica de SOCP obteve melhores resultados em todos os problemas e em todos os requisitos estudados.

Através da análise da convergência observamos que CVXOPT mostrou robustez, realizando todos os experimentos com sucesso e `fmincon` mostrou falhas de convergência em problemas dos três tipos.

Para análise da quantidade de iterações, utilizamos apenas os dados dos problemas que obtiveram testes bem sucedidos. Através de gráficos pode-se perceber que o número de iterações de CVXOPT mantém-se aproximadamente constante, enquanto que em `fmincon` varia de forma desordenada, sendo sempre maior que no primeiro caso.

Para fazer a análise do tempo de CPU, usamos a padronização descrita no início do capítulo 5 e novamente apenas os dados dos problemas que obtiveram testes bem sucedidos. Nos problemas QCQP e de arranjo de antenas o tempo de CPU em CVXOPT manteve-se baixo, quase constante, enquanto `fmincon` apresentou variação no tempo de processamento, sendo este tempo sempre maior que o registrado por CVXOPT. Nos problemas de garras múltiplos dedos o tempo padronizado de CVXOPT aumentou de forma semelhante ao tempo de `fmincon`.

Diante dos resultados obtidos com os testes numéricos, podemos concluir que é interessante usar SOCP sempre que possível.

Como trabalhos futuros planejamos avaliar outras ferramentas específicas para SOCP tais como CVX e SeDuMi e estudar outros problemas, especialmente problemas reais que tenham duas formulações, a formulação original e a formulação SOCP.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Alizadeh, F. ; Goldfarb, D. *Second-order cone programming*. Mathematical Programming, 95:351, 2001.

[2] Andersen, M.; Dahl, J.; Liu, Z.; Vandenberghe, L. *Interior-point methods for large-scale cone programming*. In Optimization for Machine Learning.

[3] Bazaraa, M. S. Sherali; H. D.; Shetty, C. M. Nonlinear Programming - *Theory and Algorithms*. Jonh Wiley and Sons, Inc., New Jersey, 3rd edition, 2006.

[4] Ben-tal, A.; Nemirovski, A. *Robust solutions of linear programming problems contaminated with uncertain data*. Mathematical Programming, 88:411424, 2000.

[5] Boyd, S. ; Crusius, C.; Hansson, A. *Control applications of nonlinear convex programming*. Journal of Process Control, pages 56, 1998.

[6] Byrd, R. H.; Gilbert, J. C. *A trust region method based on interior point techniques for nonlinear programming*. Mathematical Programming, 89:149185, 1996.

[7] Byrd, R. H.; Hribar, M. E.; Nocedal, J. *An interior point algorithm for large scale nonlinear programming*. SIAM Journal on Optimization, 1999.

[8] Castronuovo, E. D. ; Campagnolo, J. M. ; Salgado, R. *A largest-step central-path algorithm applied to the optimal power flow problem*. SBA Controle e Automação, 11:176-181, 2000.

[9] Grant, M.; Boyd, S. *Matlab software for disciplined convex programming*, version 1.21. URL: http://cvxr.com/cvx/, 2011. Acessado em: 11/04/2011.

[10] Guardia, L. E. T. *Método primal-dual para otimização não linear usando algoritmos de decomposição*. X Simpósio de Pesquisa Operacional e Logística da Marinha - SPOLM, 2007.

[11] Han, L. ; Trinkle, J. C.; Li, Z. *Grasp analysis as linear matrix inequality problems*. IEEE Transactions on Robotics and Automation, 16:12611268, 1998.

[12] Lobo, M. S. ; Vandenberghe, L. ; Boyd, S. ; Lebret, H. *Applications of second-order cone programming*. Linear Algebra and its Applications, 284:193228, 1998.

[13] Luenberger, D. G.; Ye, Y. *Linear and Nonlinear Programming*. Springer Science, New York, 3rd edition, 2008.

[14] Martínez J. M. *Otimização Prática Usando o Lagrangiano Aumentado*. Opúsculo, Departamento de Matemática Aplicada, Universidade Estadual de Campinas, Campinas, 2009.

[15] MathWorks. *Find minimum of constrained nonlinear multivariable function*. URL: http://www.mathworks.com/help/toolbox/optim/ug/fmincon.html, 2011. Acessado em: 11/04/2011.

[16] MATLAB. *version 7.11.0.584 (R2010b)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[17] Munhoz, J. R. *Otimização no Planejamento Agregado de Produção em Indústrias de Processamentode Suco Concentrado Congelado de Laranja*. Tese Doutorado, Departamento de engenharia de Produção - Universidade Federal de São Carlos, Brazil, 2009.

[18] Nemirovski, A. ; Ben-Tal, A. *Lectures on Modern Convex Optimization, Analysis, Algorithms, and Engeineering Applications*. Society for Industrial and Applied Mathematics, USA, 2001.

[19] Nesterov, Y. ; Nemirovski, A. *Interior-Point Polynomial Algorithms in Convex Programming*. Siam Studies in Applied Mathematics, Philadelphia, 1994.

[20] Nesterov,Y.E.; Todd, M.J. *Primal-dual interior-point methods for self-scaled cones*. SIAM Journal on Optimization, 8:324364, 1995.

[21] Nesterov,Y.E.; Todd, M.J. *Self-scaled cones and interior-point methods in nonlinear programming*. In Working Paper, CORE, Catholic University of Louvain, Louvain-laNeuve, 1997.

[22] Rossum, G. V. *Python programming language*. URL: http://www.python.org/, 2011. Acessado em: 11/04/2011.

[23] Solodov, M.; Izmailov, A. *Otimização - Condições de Otimalidade, Elementos de Análise Convexa e de Dualidade*. IMPA, Instituto de Matemática Pura e Aplicada, 2005.

[24] Srirangarajan, S.; Tewfik, A.; Luo, Z.-Q. *Distributed sensor network localization using socp relaxation*. IEEE Transaction on wireless Comunication, 7, 2008.

[25] Sturm, J. F. *Sedumi*. URL: http://sedumi.ie.lehigh.edu, 2010. Acessado em: 11/04/2011.

[26] Vandenberghe, L. *The CVXOPT linear and quadratic cone program solvers*. http://abel.ee.ucla.edu/cvxopt/documentation/coneprog.pdf, 2010.

[27] Vandenberghe, L. *Python software for convex optimization*. URL: http://abel.ee.ucla.edu/cvxopt/, 2010. Acessado em: 11/04/2011.

[28] Vandenberghe, L.; Boyd, S. *Convex Optimization*. Cambridge University Press, New York, 1st edition, 2004.

[29] Vanderberghe, L. ; Boyd, S. *Socp: Software for second-order cone programming*. Laboratory Stanford University, 1997.

[30] Waltz, R. A.; Morales, J. L.; Nocedal, J.; Orban, D. *An interior algorithm for nonlinear optimization that combines line search and trust region steps*. Mathematical Programming 107, 2006.

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

[32] Xue, G.; Ye, Y. *An efficient algorithm for minimizing a sum of euclidean norms with applications*. SIAM Journal on Optimization, 7:10171036, 1997.