Rafaela Filippozzi

UM ALGORITMO GEOMÉTRICO PARA O PROBLEMA DE INCLUSÃO NO ENVOLTÓRIO CONVEXO

Florianópolis
Março/2019

Rafaela Filippozzi

UM ALGORITMO GEOMÉTRICO PARA O PROBLEMA DE INCLUSÃO NO ENVOLTÓRIO CONVEXO

Dissertação/Tese submetida ao Programa de Pós-Graduação em Matemática Pura e Aplicada para a obtenção do Grau de mestre em Matemática Pura e Aplicada.

Universidade Federal de Santa Catarina – UFSC
Centro de Ciências Físicas e Matemáticas
Departamento de Matemática

Orientador Prof. Dr. Douglas Soares Gonçalves
Coorientador Prof. Dr. Luiz Rafael dos Santos

Florianópolis
Março/2019

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Filippozzi, Rafaela
Um Algoritmo Geométrico para o problema de inclusão no envoltório convexo / Rafaela Filippozzi ; orientador, Douglas Soares Gonçalves, coorientador, Luiz Rafael dos Santos, 2019.
92 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro de Ciências Físicas e Matemáticas, Programa de Pós-Graduação em Matemática Pura e Aplicada, Florianópolis, 2019.

Inclui referências.

1. Matemática Pura e Aplicada. 2. Otimização. 3. Matemática Aplicada. I. Soares Gonçalves, Douglas . II. dos Santos, Luiz Rafael. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Matemática Pura e Aplicada. IV. Título.

Rafaela Filippozzi

UM ALGORITMO GEOMÉTRICO PARA O PROBLEMA DE INCLUSÃO NO ENVOLTÓRIO CONVEXO

Esta Dissertação foi julgada aprovada para a obtenção do Título de "mestre em Matemática Pura e Aplicada", e aprovada em sua forma final pelo Programa de Pós-Graduação em Matemática Pura e Aplicada.

Florianópolis, 08 de Março de 2019.

Prof. Dr. Marcelo Sobottka
Coordenador da Pós Graduação- UFSC

Banca Examinadora:

Prof. Dr. Douglas Soares Gonçalves
Orientador - UFSC

Prof(ª). Dr(ª). Márcia Aparecida Gomes Ruggiero
Universidade Estadual de Campinas - Unicamp.
(Participação em tempo real via videoconferência)

Prof. Dr. Juliano de Bem Francisco
Universidade Federal de Santa Catarina - UFSC

Prof. Dr. Maicon Marques Alves
Universidade Federal de Santa Catarina - UFSC

Prof. Dr. Marcelo Sobottka
Coordenador do Programa Pós-Graduação em Matemática Pura e Aplicada
Portaria nº 1099/2018/GR - UFSC

Este trabalho é dedicado a Pink.

AGRADECIMENTOS

Agradecer é simplesmente reconhecer tudo o que chega até você, seja algo bom ou algo ruim. Sendo assim começo agradecendo as coisas ruins que me aconteceram ao longo do mestrado, pois todas as dificuldades que encontrei nesses dois anos me fizeram estar escrevendo esses agradecimentos e ser uma pessoa melhor.

Agradeço a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro para o desenvolvimento dessa dissertação. Agradeço a matemática, por ser uma ciência tão completa, tão linda e com tanto para descobrir. Agradeço a todos os professores do Departamento de Matemática da Universidade Federal de Santa Catarina(UFSC) em especial aos que me deram aulas ou seminários, com certeza aprendi muito com eles. Agradeço ao meu Coorientador, professor Luiz Rafael dos Santos, que desde a minha graduação no Instituto Federal Catarinense campus Camboriú vem me inspirando a ser uma grande matemática e ainda me indicou meu orientador para o mestrado na UFSC. Agradeço ao meu orientador, professor Douglas Soares Gonçalves, principalmente pela paciência em me orientar, sei que vim para o mestrado com muitas deficiências de conteúdo. Agradeço também pelos puxões de orelha e pelas horas de aulas/ reuniões que me fizeram confirmar que escolhi a área e o orientador da maneira mais correta possível.

Não posso deixar de agradecer aos professores do IFC campus Camboriú que de alguma forma me mostraram um pouco da matemática superior. Agradeço a todos os amigos da turma LM13, em especial agradeço a Douglas Deitos, Aline de Oliveira Sant'Anna, Elis Regina Thiago, Suzana Thiago e Matheus Modesti que sempre me lembravam que eu era capaz e que acreditaram que eu viraria mestre em momentos que eu mesma não acreditei.

Meu primeiro contato com a UFSC foi no curso de verão em 2016, não havia acabado minha graduação mas quis aproveitar o verão da melhor maneira possível, afinal de contas, férias indo para praia é para fracos. Foi então que conheci um grupo de pessoas que posteriormente se denominaram "Bagunceiros" composto originalmente por Everton Boos, Francieli Triches, Jéssica Neckel, Josiane Hoffmann, Marduck Montoya, Mariana Ventureli da Veiga e Luiza Sorice. Reforço que a coisa mais bagunceira que eles faziam era estudar até tarde no departamento e pedir pizza ou ficar meia hora no EFI conversando sobre tudo e tendo muitas

piadas matemáticas. Tenho muito a agradecer a presença deles tanto no verão de 2016 quanto ao longo do ano de 2017, cada risada, cada pizza e cada aprendizado foi muito importante. Agradecimento especial ao Everton e a Mariana por me aturarem pelos dois anos do mestrado, por me ensinarem coisas sobre o Matlab e me acompanharem em algumas matérias dessa matemática tão bonita chamada de Aplicada.

Em 2017 fiz o curso de verão novamente com o objetivo de ingressar ao mestrado. Em meio a tantos rostos diferentes e com uma pressão de concorrentes achei que não encontraria amigos como em 2016. Isso foi um ledo engano, não me lembro bem como, nem o porquê, mas comecei a falar com um Chileno chamado Luis Rodrigo Ortiz Henriquez, logo começamos a nos identificar um com outro e ganhamos uma cumplicidade que fez dele o meu primeiro amigo daquele verão. Agradeço a ele por todos os momentos e palavras amigas ao decorrer desses dois anos. Ainda no curso de verão comecei a conversar com outro estrangeiro, um Colombiano chamado Ever Elias Álvares Vasquez, por mais que tenhamos nossas divergências sei que é uma pessoa que posso contar e que também tenho a agradecer.

Acabado o curso de verão comecei a frequentar a sala dos alunos de mestrado, nossa eterna salinha, que eu não sabia que viraria meu lar. Aos poucos a maioria dos que ingressaram naquele ano no mestrado iam ocupando o seu lugar e frequentando a salinha. Foi então que percebi que uma das meninas que fazia verão comigo e que ficou em primeiro lugar iria também ocupar a salinha. Não gostava dela, afinal ela tinha sido a melhor no curso de verão e confesso que tinha uma certa "invejinha ". Mas aos poucos ela não ocupou o lugar apenas na salinha ocupou também um lugar no meu coração, tenho muito a agradecer a Bruna Caveion. Foram tantas conversas, risos e choros que passamos nesses dois anos que me emociono ao lembrar, obrigada por sempre estar ali. Nessa sala tive o primeiro contato com o modelo da turma o Elemar Rapach, agradeço por todos os conselhos e pelas enumeráveis horas de companhia. Conheci e agradeço também ao aluno mais inteligente que entrou naquele ano, o Julio Eduardo Cáceres Gonzales que me ajudou em algumas listas de exercício sempre utilizando suas palavras sinceras e verdadeiras. Desses amigos que ganhei no começo do mestrado ainda tenho um último a agradecer, o José Guilherme Simion Antunes, agradeço principalmente pela enorme paciência em me motivar e repetir que eu era capaz de concluir essa Dissertação. No segundo ano do mestrado um dos calouros se aproximou da gente, obrigada João Paulo Silva por todas as mágicas, piadas e perguntas reflexivas feitas em 2018.

Todas as pessoas que falei acima carregam um pouquinho de

matemática consigo e sem elas o mestrado ficaria muito, mas muito mais difícil do que já foi e essa dissertação só foi concretizada por ter amigos como eles. Não posso terminar essa seção de agradecimento dos amigos sem agradecer a Luana Strapazzon, a Thaine dos Santos Abich e a Bruna Muniz que mesmo longe sempre me motivaram a seguir o meu sonho, seja me dando apoio e conselhos via Whatsapp ou pessoalmente, afinal foram algumas vezes que a Luana veio a Florianópolis para estudar medicina enquanto eu estudava matemática com a simples motivação de estar do lado de uma amiga.

Agradeço a minha família pelo apoio, principalmente aos meus pais, Silvana Figueiró Botta e Sérgio Luis Filippozzi, aos meus avós, Edith Figueiró Botta e Celso Ivan Botta e a minha tia, Adriana Figueiró Botta. Obrigada pela compreensão da minha ausência por motivos de estudos, por serem minha motivação para sempre me dedicar e aprender mais e por terem me criado, pois sou o que sou hoje graças a eles.

Ao Rafael Herger Pinto agradeço pelo abraço sempre que precisei, pela paciência em me esperar sair do Departamento de Matemática, pelas palavras de incentivo e por me mostrar que com o seu amor do lado tudo fica mais fácil.

All our dreams can come true if we have the courage to pursue them.
(Walt Disney, 1988)

RESUMO

O problema de inclusão no envoltório convexo consiste em determinar se um certo ponto pertence ao envoltório convexo de um conjunto de n pontos. Este problema encontra importantes aplicações em geometria computacional e programação linear. Apresentamos um estudo teórico e prático de um Algoritmo Geométrico proposto em (B. Kalantari, Ann Oper Res (2015) 226:301?349), que tem como base um teorema de separação chamado de Dualidade de Distâncias. Na análise teórica do algoritmo, fornecemos algumas demonstrações alternativas ao artigo original, sob um ponto de vista próprio, fundamentado em conceitos de otimização contínua. Este ponto de vista nos permitiu propor duas variações para o Algoritmo Geométrico, além de estabelecer a relação deste com o clássico algoritmo de Frank-Wolfe. Também estudamos o comportamento prático do algoritmo em instâncias artificiais do problema, comparando-o com algoritmos clássicos de otimização para reformulações lineares e quadráticas. Os resultados obtidos sugerem o Algoritmo Geométrico como uma alternativa promissora para o problema de inclusão no envoltório convexo.

Palavras-chave: Algoritmo Geométrico; Envoltório Convexo; Dualidade de Distâncias; Teoremas de Separação; Frank-Wolfe.

ABSTRACT

The convex hull membership problem consists in deciding whether a certain point belongs to the convex hull of a set of n points. This problem finds interesting applications in computational geometry and linear programming. We present a theoretical and practical study of a geometric algorithm proposed in (B. Kalantari, Ann Oper Res (2015) 226:301?349), which is based on a separation theorem known as Distance Duality. In the theoretical analysis, we provide alternative proofs for some results, under our own perspective, relying on concepts of continuous optimization. This new point of view allowed us to develop two variants of the geometric algorithm and also to establish its relation with the classical Frank-Wolfe method. We also assessed the practical behaviour of the algorithm in artificially generated instances and compare its performance with classical optimization algorihtms for linear and quadratic programming reformulations of the problem. The numerical results suggest the geometric algorithm as a promissing alternative for the convex hull membership problem.

Keywords: Geometric Algorithm; Convex Hull; Distance Duality; Separation theorems; Frank-Wolfe.

LISTA DE ILUSTRAÇÕES

0.1 Ilustração dos casos do lema de alternativas. 2
1.1 Conjunto S e seu envoltório convexo. 4
1.2 Ilustração de hiperplano separador e hiperplano suporte. 8
2.1 No PIEC precisamos decidir se $p \in \text{conv}(S)$, sem conhecer as desigualdades que definem $\text{conv}(S)$ nem seus pontos extremos. Na figura da esquerda $p \in \text{conv}(S)$ e na da direita $p \notin \text{conv}(S)$. 14
2.2 Triângulo $pp'v$ ilustrando os elementos envolvidos em uma iteração do Algoritmo 4. 15
2.3 Ilustração das iterações do Algoritmo Geométrico. . . . 15
2.4 Hiperplano bissetor ortogonal ao segmento $p'p$ que separa p de $\text{conv}(S)$. 18
3.1 A região em cinza mostra onde podem estar os p-pivôs para p'. 26
3.2 Pior caso para $\cos\theta$ quando v é pivô simples. 27
3.3 A região cinza mostra onde podemos encontrar pivôs estritos. 32
3.4 Pior caso para $\cos\theta$ quando v é pivô estrito. 34
3.5 Iteração típica do Algoritmo Geométrico para pivôs estritos. 36
3.6 Exemplo quando $\text{conv}(S)$ é um quadrado e p está na borda. 36
3.7 Ilustração do pior caso para o Algoritmo Geométrico, quando $1/c$ é muito grande. 38
3.8 Existência de um pivô estrito com constante de visibilidade limitada por $1/\sqrt{1+\rho^2/R^2}$. 38
3.9 Exemplo quando $\text{conv}(S)$ é um quadrado e $B(p,\rho) \subset \text{conv}(S)$ 40
4.1 Para $\kappa \geq 1/2$ temos que a condição do ângulo implica pivô simples e para $\kappa \geq \sqrt{2}/2$ a condição implica pivô estrito. 49
4.2 Tempo médio em 10 instâncias com $m = 100$, $p = 0 \in \text{conv}(S)$, para $n = 500, 1000, \ldots, 5000$, $\varepsilon = 10^{-2}$. 54
4.3 Tempo médio em 10 instâncias com $m = 100$, $p = 0 \in \text{conv}(S)$, para $n = 500, 1000, \ldots, 5000$, $\varepsilon = 10^{-2}$. 54
4.4 Tempo médio em 10 instâncias com $m = 100$, $p = 0 \in \text{conv}(S)$, para $n = 10000, \ldots, 100000$, $\varepsilon = 10^{-2}$. 56
4.5 Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 2$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$. 56

4.6 Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 2$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$. 57
4.7 Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 2$, para $n = 10000, \ldots, 100000$, $\varepsilon = 10^{-4}$. 58
4.8 Tempo médio em 10 instâncias com $m = 100$, $p = \frac{1}{2}v_\ell + \frac{1}{2}v_q$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-2}$. 59
4.9 Tempo médio em 10 instâncias com $m = 100$, $p = \frac{1}{2}v_\ell + \frac{1}{2}v_q$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-3}$. 59
4.10 Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 1.01$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$. 60
4.11 Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 1.01$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$. 61
4.12 Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 1.01$, para $n = 10000, \ldots, 100000$. 62

LISTA DE TABELAS

4.1 Algoritmos testados. 52
4.2 Custo por Iteração (pior caso) 52

SUMÁRIO

	Introdução	**1**
1	**CONCEITOS PRELIMINARES**	**3**
1.1	NOÇÕES DE CONVEXIDADE	3
1.2	OTIMIZAÇÃO SUAVE SOBRE UM CONJUNTO CONVEXO	5
1.3	PROJEÇÃO E HIPERPLANO SEPARADOR	7
1.4	ALGORITMOS DE BUSCA DIRECIONAL	8
1.4.1	**Gradiente Projetado**	**10**
1.4.2	**Gradiente Condicional**	**11**
2	**ALGORITMO GEOMÉTRICO**	**13**
2.1	PROBLEMA DE INCLUSÃO NO ENVOLTÓRIO CONVEXO	13
2.2	ALGORITMO GEOMÉTRICO	13
2.3	RELAÇÃO COM O MÉTODO DE FRANK-WOLFE	22
3	**ANÁLISE DE COMPLEXIDADE**	**25**
3.1	CARACTERIZAÇÃO GEOMÉTRICA DE PIVÔ SIMPLES	25
3.2	COMPLEXIDADE DE ITERAÇÃO	25
3.3	PIVÔS ESTRITOS	31
3.4	UMA ANÁLISE ALTERNATIVA	35
4	**FORMULAÇÕES ALTERNATIVAS E EXPERIMENTOS NUMÉRICOS**	**43**
4.1	FORMULAÇÕES LINEARES	43
4.2	FORMULAÇÃO QUADRÁTICA	45
4.2.1	**Projeção no simplex unitário**	**45**
4.2.2	**Critérios de parada**	**46**
4.3	VARIANTES DO ALGORITMO GEOMÉTRICO	47
4.3.1	**Algoritmo Geométrico com condição do ângulo**	**47**
4.3.2	**Algoritmo Geométrico Ganancioso**	**48**
4.4	EXPERIMENTOS NUMÉRICOS	50
5	**CONCLUSÕES E TRABALHOS FUTUROS**	**63**
	REFERÊNCIAS	**65**
A	**ALGUMAS DEMONSTRAÇÕES DO CAPÍTULO 1**	**67**

INTRODUÇÃO

Dado $S \subset \mathbb{R}^m$, um conjunto de n pontos em $\mathbb{R}^m$, o problema de inclusão no envoltório convexo (PIEC) consiste em determinar se um dado $p \in \mathbb{R}^m$ pertence ao conv(S)[1] (o envoltório convexo de S). Se as coordenadas dos n pontos de S são colocadas como colunas de uma matriz $S \in \mathbb{R}^{m \times n}$, e denotando por $e \in \mathbb{R}^n$ o vetor cujas componentes são todas iguais a um, podemos formalmente descrever o PIEC como o seguinte problema de decisão: existe $x \in \mathbb{R}^n$ tal que $Sx = p$, $e^T x = 1$, $x \geq 0$?

Este problema está relacionado a conceitos fundamentais em programação linear [1–4] e encontra importantes aplicações em geometria computacional [1, 5, 6].

A geometria computacional estuda algoritmos e estruturas de dados para a resolução computacional de problemas geométricos. Para confecção de máquinas automatizadas, por exemplo, muitas vezes é preciso saber qual o envoltório convexo de um conjunto de pontos [6]. Outro problema em geometria computacional, e um caso particular do PIEC, é o problema de irredundância, que consiste em determinar todos os pontos extremos do conv(S) [5].

Embora o PIEC possa ser formulado como um problema de otimização (mais especificamente um problema de programação quadrática) e então resolvido por métodos clássicos como Gradiente Projetado ou Frank-Wolfe [7], em [1], Kalantari apresenta um algoritmo simples, inspirado em ideias geométricas, ao qual chamaremos de *Algoritmo Geométrico*. Este algoritmo explora o seguinte lema de alternativas: ou para todo $p' \in \text{conv}(S)$, existe $v \in S$ tal que v está mais próximo de p do que de p'; ou então existe $p' \in \text{conv}(S)$ tal que o hiperplano bissetor ortogonal ao segmento $p'p$ separa p de conv(S), como ilustra a Figura 0.1.

Seja $d(x, y)$ a distância Euclidiana entre x e $y \in \mathbb{R}^m$. A cada iteração, o Algoritmo Geométrico busca $v \in S$ tal que $d(v, p) \leq d(v, p')$. Se tal v não existe, temos que $p \notin \text{conv}(S)$. Caso contrário, p' é atualizado para o ponto no segmento $p'v$ mais próximo de p. Isto leva a uma iteração de implementação simples e computacionalmente mais barata que a de algoritmos clássicos de otimização.

[1] O envoltório convexo também é conhecido por invólucro convexo ou fecho convexo.

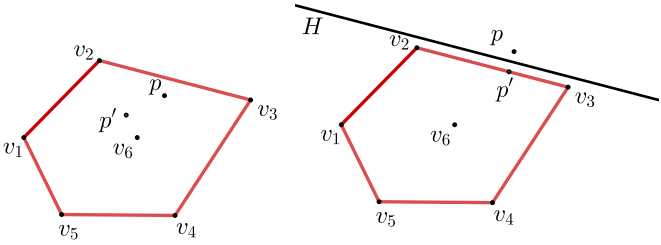

Figura 0.1 – Ilustração dos casos do lema de alternativas.

Neste trabalho, realizamos um estudo teórico e prático do Algoritmo Geométrico para o problema de inclusão no envoltório convexo. Na parte teórica, apresentamos os resultados que asseguram a corretude do algoritmo, os teoremas de *dualidade de distâncias*, e a análise de complexidade de iteração. Algumas demonstrações são diferentes do artigo original [1], sob um ponto de vista próprio, que nos possibilitou propor variantes do Algoritmo Geométrico e estabelecer a conexão entre este e o clássico algoritmo do Gradiente Condicional (Frank-Wolfe). No estudo numérico, apresentamos formulações alternativas para o PIEC e realizamos comparações entre variantes do Algoritmo Geométrico e algoritmos clássicos de otimização.

A dissertação está organizada da seguinte forma. Revisamos alguns conceitos de convexidade e otimização no Capítulo 1, bem como alguns algoritmos clássicos para minimizar uma função suave sobre um conjunto convexo compacto. No Capítulo 2, apresentamos o problema de inclusão no envoltório convexo, descrevemos o Algoritmo Geométrico e discutimos os resultados teóricos que sustentam o algoritmo. Também mostramos a relação entre o Algoritmo Geométrico e o clássico método de Frank-Wolfe. A análise de complexidade do Algoritmo Geométrico é detalhada no Capítulo 3. No Capítulo 4, são apresentadas as formulações lineares e quadráticas para o problema de inclusão no envoltório convexo, além de algumas variações do Algoritmo Geométrico e experimentos computacionais. Fechamos o trabalho com o Capítulo 5 trazendo conclusões e trabalhos futuros.

1 CONCEITOS PRELIMINARES

Com base nas referências [3,7], este capítulo traz algumas noções de convexidade, otimização de funções suaves sobre um conjunto convexo compacto e uma breve descrição de alguns algoritmos clássicos. Tais conceitos serão importantes posteriormente na discussão teórica sobre o Algoritmo Geométrico (Capítulo 2) e na proposta de métodos alternativos para o problema de inclusão no envoltório convexo (Capítulo 4). A maioria das demonstrações dos resultados presentes neste capítulo também podem ser encontradas no apêndice A.

Ao longo do texto, $\|.\|$ denotará a norma Euclidiana e a correspondente norma matricial induzida. Dados $x, y \in \mathbb{R}^m$, a distância Euclidiana entre eles será denotada por

$$d(x,y) = \|x-y\| = \left(\sum_{i=1}^{m} |x_i - y_i|^2\right)^{1/2},$$

e o produto interno usual de $\mathbb{R}^m$ por

$$x^T y = \sum_{i=1}^{m} x_i y_i,$$

onde $x \in \mathbb{R}^m$ representa um vetor coluna e x^T (o transposto de x) um vetor linha.

Denotaremos por $B(x,r) = \{y \in \mathbb{R}^n \ : \ d(y,x) \leq r\}$, a bola de centro em $x \in \mathbb{R}^n$ e raio $r > 0$.

1.1 Noções de convexidade

Definição 1.1. *Um subconjunto não-vazio $\Omega \subseteq \mathbb{R}^n$ é dito um conjunto convexo quando para quaisquer $x, y \in \Omega$ tem-se que:*

$$\alpha x + (1-\alpha)y \in \Omega, \quad \forall \alpha \in [0,1].$$

Definição 1.2. *Sejam $v_1, v_2, \ldots, v_n$ e v vetores de $\mathbb{R}^n$. Quando $v = \sum_{i=1}^{n} \alpha_i v_i$, com $\alpha_i \geq 0$, para $i = 1, \ldots, n$, e $\sum_{i=1}^{n} \alpha_i = 1$, dizemos que v é combinação convexa dos vetores $v_1, v_2, \ldots, v_n$.*

Definição 1.3. *Seja Ω um conjunto convexo. Dizemos que $v \in \Omega$ é um ponto extremo de Ω, se v não pode ser escrito como combinação convexa de outros elementos de Ω.*

Definição 1.4. *Seja $S \subseteq \mathbb{R}^n$ um conjunto finito de pontos. O envoltório convexo de S é o conjunto de todas as combinações convexas de qualquer número finito de elementos de S.*

Daqui em diante, denotaremos o envoltório convexo de S por $\text{conv}(S)$. É possível mostrar que $\text{conv}(S)$ é o menor conjunto convexo que contém S, como ilustrado na Figura 1.1, ou equivalente, a intersecção de todos os conjuntos convexos que contêm S. Para mais detalhes, veja [7, 8].

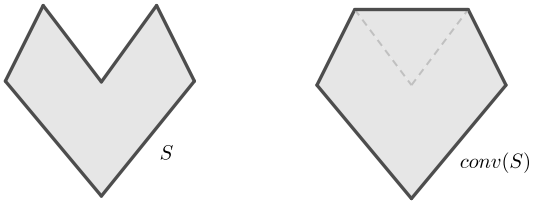

Figura 1.1 – Conjunto S e seu envoltório convexo.

Definição 1.5. *Seja $\Omega \subset \mathbb{R}^n$ um conjunto convexo não-vazio. Uma função $f : \Omega \to \mathbb{R}$ é dita convexa se para quaisquer $x, y \in \Omega$:*

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y), \quad \forall \alpha \in [0,1].$$

Definição 1.6. *Seja $\Omega \subset \mathbb{R}^n$ um conjunto convexo não-vazio. Uma função $f : \Omega \to \mathbb{R}$ é dita estritamente convexa se para quaisquer $x, y \in \Omega$, com $x \neq y$:*

$$f(\alpha x + (1-\alpha)y) < \alpha f(x) + (1-\alpha)f(y), \quad \forall \alpha \in (0,1).$$

Proposição 1.7. *Sejam $\Omega \subset \mathbb{R}^n$ um conjunto convexo não-vazio e $f : \mathbb{R}^n \to \mathbb{R}$ um função diferenciável em $\mathbb{R}^n$. A função f é convexa sobre Ω se, e somente se,*

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \quad \forall x, y \in \Omega. \tag{1.1}$$

1.2 Otimização suave sobre um conjunto convexo

Considere o problema

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{s.a} \quad & x \in \Omega \subset \mathbb{R}^n \end{aligned} \tag{1.2}$$

em que Ω é não-vazio, fechado e convexo, e $f : \mathbb{R}^n \to \mathbb{R}$ é continuamente diferenciável. Denominamos $f(x)$ de função objetivo, Ω de região factível(ou viável) e $x \in \Omega$ de ponto factível.

Definição 1.8. *Um vetor $x \in \Omega$ é dito mínimo local de f em Ω, se existe $\varepsilon > 0$ tal que $f(x) \leq f(y)$ para todo $y \in \Omega$ satisfazendo $\|x - y\| \leq \varepsilon$. Um vetor $x \in \Omega$ é chamado de mínimo global de f em Ω se $f(x) \leq f(y)$ para todo $y \in \Omega$.*

Com a hipótese de convexidade associada a função objetivo obtemos alguns resultados interessantes e úteis para solucionar o problema (1.2). Como vemos na proposição a seguir.

Proposição 1.9. *Seja $\Omega \subset \mathbb{R}^n$ não-vazio e convexo, e seja $f : \Omega \to \mathbb{R}$ uma função convexa.*

(a) Todo minimizador local é minimizador global.

(b) Se f é estritamente convexa e existe minimizador global, então o minimizador global é único.

(c) Se f é continuamente diferenciável e $x \in \Omega$ é tal que $\nabla f(x) = 0$, então x é minimizador global.

A definição a seguir é de estrema importância para motivação dos Algoritmos de Busca Direcional apresentados na seção 1.4 e para alguns resultados teóricos que veremos a seguir.

Definição 1.10. *Considere o problema (1.2) e $x \in \Omega$. Dizemos que $d \in \mathbb{R}^n$ é uma direção factível a partir de x, se existe $\bar{t} > 0$ tal que $x + td \in \Omega \quad \forall t \in [0, \bar{t})$. Dizemos que d é uma direção de descida para f a partir de x se existe $\varepsilon > 0$ tal que $f(x + td) < f(x), \forall t \in (0, \varepsilon]$.*

Proposição 1.11. *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função continuamente diferenciável. Se $\nabla f(x_k)^T d_k < 0$, então d_k é direção de descida para f a partir de x_k.*

Definição 1.12. *Considere o problema* (1.2)*. Dizemos que* x_* *é um ponto estacionário se* $\nabla f(x_*)^T(x - x_*) \geq 0, \quad \forall x \in \Omega$.

Esta definição de estacionariedade implica que a partir de $x_* \in \Omega$ não existe direção d factível tal que $\nabla f(x_*)^T d < 0$, isto é, não há direção factível e de descida (até primeira ordem).

Teorema 1.13. *Sejam* $\Omega \subset \mathbb{R}^n$*, não-vazio, convexo e fechado,* $f : \Omega \to \mathbb{R}$ *uma função continuamente diferenciável. Se* x_* *é minimizador local de* f *em* Ω*, então:*

$$\nabla f(x_*)^T(x - x_*) \geq 0, \quad \forall x \in \Omega.$$

Além disso, se f *é convexa, a condição acima assegura que* x_* *é um minimizador global de* f *em* Ω*.*

O Teorema 1.13 apresenta condições necessárias e suficientes para um problema de otimização convexa suave. Para o problema geral de otimização não-linear, minimizadores locais que satisfazem certas condições de regularidade devem cumprir as conhecidas condições de Karush-Kuhn-Tucker (KKT) que iremos apenas enunciar.

Considere o problema geral de otimização não-linear:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.a.} \quad & h(x) = 0, \\ & g(x) \leq 0, \end{aligned} \tag{1.3}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$, $h : \mathbb{R}^n \to \mathbb{R}^m$, $g : \mathbb{R}^n \to \mathbb{R}^p$, são funções de classe $\mathcal{C}^1$.

Definição 1.14. *Seja* $\bar{x}$ *um ponto factível para* (1.3)*. Uma restrição* $g_j(x) \leq 0$ *é dita restrição ativa em* $\bar{x}$ *quando* $g_j(\bar{x}) = 0$*. Se* $g_j(\bar{x}) < 0$*, dizemos que a restrição é inativa.*

Denotamos por $I(\overline{x}) \subset \{1, \ldots, p\}$ o conjunto de índices das restrições de desigualdade ativas em $\overline{x}$, i.e., $I(\bar{x}) = \{j \in \{1, \ldots, n\} \ : \ g_j(\bar{x}) = 0\}$.

Definição 1.15. *Considere o problema* (1.3)*. Dizemos que o ponto factível* $\overline{x}$ *é regular se* $\{\nabla h_i(\overline{x})\}_{i=1}^m \cup \{\nabla g_j(\overline{x})\}_{j \in I(\overline{x})}$ *é linearmente independente.*

Teorema 1.16 (Condições KKT)**.** *Seja $\overline{x}$ minimizador local de* (1.3)*. Se $\overline{x}$ é regular, então existem $\lambda \in \mathbb{R}^m$, $\mu \in \mathbb{R}^p$ tais que*

$$\begin{cases} \nabla f(\overline{x}) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\overline{x}) + \sum_{j=1}^{p} \mu_j \nabla g_j(\overline{x}) = 0 \\ \mu \geq 0 \\ \mu_j g_j(\overline{x}) = 0, \quad j = 1, \ldots, p \\ h(\overline{x}) = 0, \quad g(\overline{x}) \leq 0. \end{cases} \tag{1.4}$$

Uma discussão detalhada e demonstrações dos resultados dessa seção podem ser encontradas em [7] e [3].

1.3 Projeção e Hiperplano Separador

Os dois resultados que veremos nessa subseção podem ser encontrados em [7] e motivaram algumas demonstrações que serão apresentadas no próximo capítulo. Para facilitar a leitura, as demonstrações desta seção também foram deixadas para o Apêndice A, uma vez que já são bem conhecidas da literatura.

Teorema 1.17 (Teorema da Projeção)**.** *Seja $\Omega \subset \mathbb{R}^n$ um conjunto não-vazio, convexo e fechado.*

(a) Para todo $x \in \mathbb{R}^n$, existe um único elemento $P_\Omega(x) \in \Omega$ (chamado de projeção de x em Ω) tal que $\|P_\Omega(x) - x\| \leq \|z - x\|, \forall z \in \Omega$.

(b) Dado qualquer $x \in \mathbb{R}^n$, $z = P_\Omega(x)$ se, e somente se,

$$(y - z)^T (x - z) \leq 0, \quad \forall y \in \Omega.$$

(c) A função $f : \mathbb{R}^n \to \Omega$ definida por $f(x) = P_\Omega(x)$ é contínua e não expansiva, isto é:

$$\|P_\Omega(x) - P_\Omega(y)\| \leq \|x - y\|, \quad \forall x, y \in \mathbb{R}^n.$$

(d) Dado $x \in \mathbb{R}^n$, para todo y em Ω temos que $\|y - P_\Omega(x)\| \leq \|y - x\|$.

A caracterização da projeção dada pelo Teorema 1.17, nos permite expressar a condição de estacionariedade para (1.2) de uma maneira alternativa.

Teorema 1.18. *Seja f uma função continuamente diferenciável e Ω não vazio, fechado e convexo. Um ponto x_* é estacionário para* (1.2) *se, e somente se, $x_* = P_\Omega(x_* - \nabla f(x_*))$.*

Definição 1.19. *Um hiperplano em* $\mathbb{R}^n$ *é um conjunto da forma* $\{x \in \mathbb{R}^n : a^T x = b\}$*, onde* $a \in \mathbb{R}^n$*,* $a \neq 0$*, e* $b \in \mathbb{R}$*. Os conjuntos* $\{x \in \mathbb{R}^n : a^T x \geq b\}$*,* $\{x \in \mathbb{R}^n : a^T x \leq b\}$*, são chamados de semi-espaços associados ao hiperplano.*

Definição 1.20. *Um subconjunto de* $\mathbb{R}^n$ *que pode ser expresso como a interseção de um número finito de semi-espaços é chamado de poliedro.*

Teorema 1.21 (Teorema do Hiperplano Separador)**.** *Seja* $\Omega \subset \mathbb{R}^n$ *um conjunto não-vazio, convexo e fechado e* $x \notin \Omega$*. Então existe um vetor* $a \in \mathbb{R}^n \setminus \{0\}$ *tal que*

$$a^T y > a^T x, \quad \forall y \in \Omega.$$

O Teorema 1.21 nos diz que dado um conjunto convexo fechado Ω e um ponto $x \notin \Omega$, existe um hiperplano que separa estritamente x do conjunto Ω. Não é difícil mostrar que o hiperplano $\{y \in \mathbb{R}^n : a^T y = \beta\}$, onde $a = P_\Omega(x) - x$ e $\beta = (P_\Omega(x) - x)^T P_\Omega(x)$, além de separar x de Ω, tangencia Ω em $P_\Omega(x)$: a ele damos o nome de hiperplano suporte.

Os conceitos de hiperplano separador/suporte, ilustrados na Figura 1.2, bem como o teorema da projeção, são fundamentais na demonstração da *dualidade de distâncias* que fundamenta o Algoritmo Geométrico discutido no Capítulo 2.

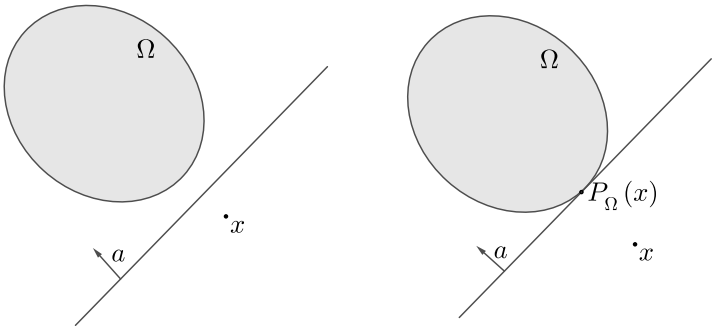

Figura 1.2 – Ilustração de hiperplano separador e hiperplano suporte.

1.4 Algoritmos de Busca Direcional

A ideia de algoritmos de busca direcional (factíveis) é, a cada iteração, reduzir o valor da função objetivo através de direções factíveis

e de descida. O Algoritmo 1 apresenta um protótipo desse tipo de algoritmo para o problema (1.2).

Algoritmo 1: BUSCA DIRECIONAL

Dados: $x_0 \in \Omega, \varepsilon > 0, k = 0$

1 Se x_k é um ponto estacionário de (1.2), pare.
2 Caso contrário, determine uma direção factível d_k tal que $\nabla f(x_k)^T d_k < 0$ e um tamanho máximo de passo $\bar{t} > 0$ tal que $x_k + \bar{t} d_k \in \Omega$.
3 Encontre $t_k \in (0, \bar{t}]$ tal que $f(x_k + t_k d_k) < f(x_k)$.
4 Atualize $x_{k+1} = x_k + t_k d_k$, faça $k = k + 1$, e retorne ao Passo 1.

Como mencionado, o Algoritmo 1 é apenas um protótipo de algoritmo factível de descida e mesmo quando $\Omega = \mathbb{R}^n$, é possível encontrar contra-exemplos onde pontos limite da sequência gerada pelo algoritmo não são estacionários [7,9]. Para garantir a convergência global a pontos estacionários, condições adicionais sobre d_k e t_k precisam ser incorporadas. Aqui discutiremos brevemente as condições apresentadas em [9] para o caso $\Omega = \mathbb{R}^n$.

A condição de *proporcionalidade* exige que:

$$\|d_k\| \geq \beta \|\nabla f(x_k)\|, \quad \forall k \geq 0, \tag{1.5}$$

para alguma constante $\beta > 0$. Tal condição evita direções relativamente pequenas em relação ao gradiente. Já a *condição do ângulo* pede que:

$$\nabla f(x_k)^T d_k \leq -\kappa \|\nabla f(x_k)\| \|d_k\|, \quad \forall k \geq 0, \tag{1.6}$$

para algum $\kappa \in (0, 1)$. Esta condição procura evitar que as direções de descida $\{d_k\}$ fiquem ortogonais ao gradiente de f. Por fim, a condição de Armijo pede um decréscimo suficiente (ao invés de decréscimo simples) no Passo 3 do Algoritmo 1:

$$f(x_k + t_k d_k) \leq f(x_k) + \eta t_k \nabla f(x_k)^T d_k, \tag{1.7}$$

com $\eta \in (0, 1)$.

Incorporando as condições de proporcionalidade, ângulo e o critério de Armijo ao Algoritmo 1, é possível mostrar que todo ponto limite da sequência gerada $\{x_k\}$ é ponto estacionário de f em $\mathbb{R}^n$. Para a demonstração deste resultado de convergência global, consulte [9] para o caso $\Omega = \mathbb{R}^n$. Para Ω não vazio, fechado e convexo qualquer é possível adaptar as condições acima e obter um resultado similar (veja [7]).

Dada d_k tal que $\nabla f(x_k)^T d_k < 0$, há vários métodos para escolher o tamanho de passo t_k satisfazendo (1.7), entre eles: busca linear exata, *backtracking* e estratégias de interpolação [7].

Nesta dissertação utilizaremos o primeiro, que consiste em tomar

$$t_k = \arg\min_{t>0} \phi(t), \tag{1.8}$$

onde $\phi(t) = f(x_k + td_k)$. Em geral, resolver (1.8) pode ser difícil, mas há casos tratáveis.

Considere $f(x) = \frac{1}{2}x^T Ax - b^T x$, onde $A \in \mathbb{R}^{n\times n}$ é uma matriz simétrica definida positiva[1] e $b \in \mathbb{R}^n$. Encontrar o minimizador global de $\phi(t)$ equivale a encontrar t_k tal que $\phi'(t) = \nabla f(x_k + td_k)^T d_k = 0$. Usando o fato de que $\nabla f(x) = Ax - b$ e que A é uma matriz definida positiva, temos que t_k que satisfaz (1.8) é dado por:

$$t_k = \frac{-\nabla f(x_k)^T d_k}{d_k^T A d_k}. \tag{1.9}$$

Nas próximas seções serão apresentados dois algoritmos de busca direcional para o problema (1.2): Gradiente Projetado [7] e Gradiente Condicional [10].

1.4.1 Gradiente Projetado

No método de Gradiente Projetado para o problema (1.2), dado um ponto factível $x_k \in \Omega$, é realizada uma busca linear ao longo da direção $d_k = \bar{x}_k - x_k$, onde $\bar{x}_k = P_\Omega(x_k - \nabla f(x_k))$. Não é difícil mostrar que, se x_k não é estacionário, então d_k é uma direção de descida, e que $x_k + t\,d_k \in \Omega$, para todo $t \in [0, 1]$. Note que se $x_k - \nabla f(x_k) \in \Omega$, a direção de busca é a mesma do tradicional algoritmo do gradiente para otimização sem restrições [3].

O Algoritmo 2 sintetiza o método de Gradiente Projetado com busca linear.

Para um dado $x_k \in \Omega$, se $d_k = 0$, temos então pelo Teorema 1.18 que x_k é estacionário. Isso justifica o critério de parada no Passo 2 do Algoritmo 2.

Observe que a dificuldade desse algoritmo está em projetar em Ω. O cálculo da projeção pode ser simples para alguns conjuntos como caixas, bolas, hiperplanos, semi-espaços, e arbitrariamente complicada em outros casos. Uma alternativa para quando a projeção em Ω é muito cara é o algoritmo de Gradiente Condicional apresentado na seção seguinte.

[1] $x^T Ax > 0$, para todos vetores não nulos $x \in \mathbb{R}^n$ [7].

Algoritmo 2: Gradiente Projetado

Dados: $x_0 \in \Omega, \varepsilon > 0, \eta \in (0,1), k = 0$.

1 Calcule $\bar{x}_k = P_\Omega(x_k - \nabla f(x_k))$ e defina $d_k = \bar{x}_k - x_k$.

2 Se $\|d_k\| < \varepsilon$, pare.

3 Encontre $t_k \in (0,1]$, tal que
$f(x_k + t_k d_k) \leq f(x_k) + \eta t_k \nabla f(x_k)^T d_k$.

4 Atualize $x_{k+1} = x_k + t_k d_k$, faça $k = k+1$ e retorne ao Passo 1.

1.4.2 Gradiente Condicional

O método de Gradiente Condicional, também conhecido como método de Frank-Wolfe [10], é um método iterativo de primeira ordem para a otimização convexa suave. O método tenta resolver o problema (1.2) através de uma sequência de subproblemas da forma:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \nabla f(x_k)^T (x - x_k) \\ \text{s.a} \quad & x \in \Omega. \end{aligned} \tag{1.10}$$

Perceba que este tipo de problema consiste em minimizar uma linearização da função objetivo sobre o conjunto factível.

Algoritmo 3: Gradiente Condicional

Dados: $x_0 \in \Omega, \varepsilon > 0, \eta \in (0,1), k = 0$

1 Encontre $\overline{x}_k$ solução de (1.10).

2 Se $\nabla f(x_k)^T (\bar{x}_k - x_k) \geq -\varepsilon$, pare.

3 Defina $d_k = \overline{x}_k - x_k$.

4 Encontre $t_k \in (0,1]$, tal que
$f(x_k + t_k d_k) \leq f(x_k) + \eta t_k \nabla f(x_k)^T d_k$.

5 Atualize $x_{k+1} = x_k + t_k d_k$, faça $k = k+1$ e retorne ao Passo 1.

Note que o Algoritmo 3 pode não estar bem-definido quando Ω é ilimitado, pois o subproblema (1.10) pode não ter solução (ser ilimitado). Assim, assumiremos que Ω além de convexo, é também compacto[2].

Seja $\bar{x}_k$ solução de (1.10), isto é

$$\nabla f(x_k)^T (\bar{x}_k - x_k) \leq \nabla f(x_k)^T (x - x_k), \quad \forall x \in \Omega.$$

[2] No problema de inclusão que estamos interessados, conv(S) é sempre convexo e compacto, já que S é formado por uma quantidade finita de pontos.

Se $\nabla f(x_k)^T(\bar{x}_k - x_k) \geq 0$, então pelo Teorema 1.13 temos que x_k é ponto estacionário para (1.2). Isto justifica o critério de parada do Passo 2:

$$\nabla f(x_k)^T(\bar{x}_k - x_k) \geq -\varepsilon. \tag{1.11}$$

Seja x_* um minimizador global de (1.2). Se f é convexa temos

$$f(x_*) \geq f(x_k) + \nabla f(x_k)^T(x_* - x_k),$$

e utilizando o fato que $\bar{x}_k$ é solução de (1.10)

$$f(x_*) \geq f(x_k) + \nabla f(x_k)^T(\bar{x}_k - x_k).$$

Portanto, quando f é convexa, o critério de parada (1.11) implica em

$$f(x_k) - f(x_*) \leq \varepsilon. \tag{1.12}$$

O lado esquerdo da desigualdade acima é conhecido como "gap" entre o valor atual $f(x_k)$ e o valor ótimo $f(x_*)$. Assim, a cada iteração do Algoritmo 3, $\nabla f(x_k)^T(x_k - \bar{x}_k)$ fornece uma cota superior para o gap, mesmo não conhecendo o valor ótimo $f(x_*)$ explicitamente.

Observe que o método do gradiente projetado exige uma etapa de projeção no conjunto factível em cada iteração, enquanto o método de Frank-Wolfe precisa da solução de (1.10). Em vários problemas, resolver (1.10) é computacionalmente mais barato que o cálculo da projeção em Ω. Um exemplo interessante é discutido na Seção 2.3, onde $\Omega = \Delta_n := \{x \in \mathbb{R}^n \ : \ e^T x = 1, \ x \geq 0\}$ é o Simplex Unitário (Simplex de probabilidades).

2 ALGORITMO GEOMÉTRICO

Neste capítulo discutiremos sobre o problema de inclusão no envoltório convexo [11] e apresentaremos o objeto de estudo desta dissertação: o Algoritmo Geométrico [1]. Com base nos resultados teóricos em [1], mostraremos a boa definição e corretude do algoritmo, bem como detalhes da implementação. Ao final deste capítulo, iremos interpretar o Algoritmo Geométrico como uma variante do método de Gradiente Condicional (Frank-Wolfe).

2.1 Problema de inclusão no envoltório convexo

Dado $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$, o problema de inclusão no envoltório convexo (PIEC) consiste em determinar se um ponto $p \in \mathbb{R}^m$ pertence ao envoltório convexo de S. Como discutido na Introdução, este problema possui aplicações em geometria computacional e em programação linear [1, 5, 6].

Se $p \in \text{conv}(S)$, pela Definição 1.4, temos que:

$$p = \sum_{i=1}^{n} \alpha_i v_i, \text{ com } \sum_{i=1}^{n} \alpha_i = 1, \text{ e } \alpha_i \geq 0, \text{ para } i = 1, \ldots, n. \tag{2.1}$$

Porém, se $p \notin \text{conv}(S)$, o Teorema 1.21 nos diz que existe um hiperplano que separa p do $\text{conv}(S)$. Destas observações, não é difícil formular o PIEC como um problema de programação quadrática (veja Seção 2.3) ou mesmo um problema de programação linear (veja Seção 4.1) como discutiremos mais adiante.

Como S possui um número finito de pontos de $\mathbb{R}^m$, é possível mostrar que $\text{conv}(S)$ é um poliedro limitado e, portanto, pode ser visto como a intersecção de um número finito de semi-espaços. No entanto, é importante ressaltar que no PIEC, não conhecemos as desigualdades lineares que definem este poliedro, tampouco sabemos quais elementos de S são pontos extremos de $\text{conv}(S)$[1].

2.2 Algoritmo Geométrico

Para abordar o problema de inclusão no envoltório convexo usaremos um algoritmo proposto por Kalantari [1], que chamaremos de

[1] Em verdade este é o problema de irredundância mencionado na Introdução.

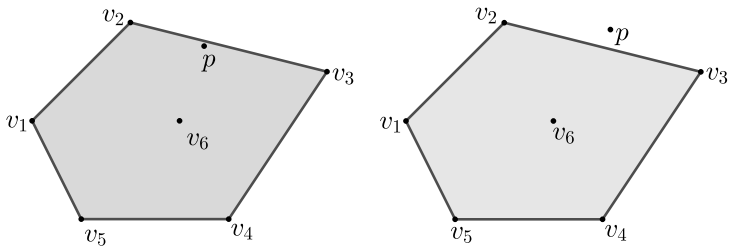

Figura 2.1 – No PIEC precisamos decidir se $p \in \text{conv}(S)$, sem conhecer as desigualdades que definem $\text{conv}(S)$ nem seus pontos extremos. Na figura da esquerda $p \in \text{conv}(S)$ e na da direita $p \notin \text{conv}(S)$.

Algoritmo Geométrico[2].

Algoritmo 4: Algoritmo Geométrico

Dados: $S = \{v_1, ..., v_n\}, p \in \mathbb{R}^m, \varepsilon \in (0, 1)$

1 Escolha $p' \in \arg\min\{d(v_j, p) \ : \ v_j \in S\}$.
2 Se $\forall v_j \neq p'$, $d(v_j, p) > d(v_j, p')$, pare.
3 Escolha $v_j \neq p'$, tal que $d(v_j, p) \leq d(v_j, p')$.
4 Calcule $\bar{\alpha} = \arg\min\{d(p, (1-\alpha)p' + \alpha v_j) : 0 \leq \alpha \leq 1\}$, defina $p'' = (1-\bar{\alpha})p' + \bar{\alpha} v_j$ e atualize $p' \leftarrow p''$.
5 Se $d(p', p) < \varepsilon$, pare.
6 Retorne ao passo 2

Se $\forall v_j \neq p'$, $d(v_j, p) > d(v_j, p')$ a resposta para o PIEC é que $p \notin \text{conv}(S)$ (isto ficará mais claro após o Teorema de Dualidade de Distâncias que será visto posteriormente). Caso contrário, se o Algoritmo Geométrico para no Passo 5, temos um ponto $p' \in \text{conv}(S)$ que está a uma distância menor que ε de p. Temos então que p_ε é uma ε-solução pertencente ao $\text{conv}(S)$, e classificamos p como elemento de $\text{conv}(S)$.

No Passo 1, poderíamos escolher como ponto inicial qualquer $p' \in \text{conv}(S)$. No entanto, a escolha sugerida $p' = \arg\min\{d(v_i, p) : v_i \in S\}$, de começar com um ponto de S mais próximo de p, tem razões técnicas que serão discutidas mais adiante.

[2] No trabalho de Kalantari, o algoritmo foi originalmente chamado de "Triangle Algorithm" (Algoritmo do Triângulo).

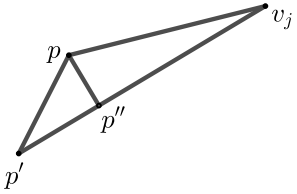

Figura 2.2 – Triângulo $pp'v$ ilustrando os elementos envolvidos em uma iteração do Algoritmo 4.

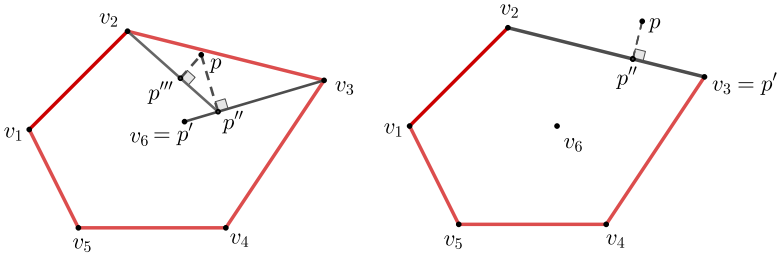

Figura 2.3 – Ilustração das iterações do Algoritmo Geométrico.

No Passo 3, buscamos $v_j \in S \setminus \{p'\}$ tal que $d(v_j, p) \leq d(v_j, p')$. Um elemento de S com essa propriedade será chamado de *p-pivô para* p', *pivô simples*, ou simplesmente *pivô*. Se existe um pivô v_j, então p' é atualizado para p'': o ponto no segmento $p'v$ mais próximo de p (Passo 4; veja também Figura 2.2). Se em uma dada iteração não existe pivô, então o algoritmo para, e retorna p' como uma *testemunha* de que $p \notin \text{conv}(S)$: o hiperplano que bisseta ortogonalmente o segmento $p'p$ separa p de $\text{conv}(S)$.

Exemplo 2.1. Na Figura 2.3 exemplificamos as iterações do Algoritmo Geométrico para dois casos, em que $S = \{v_1, v_2, v_3, v_4, v_5, v_6\} \subset \mathbb{R}^2$. No caso a esquerda o ponto pertence ao conv(S) e no a direita (b) não.

No primeiro caso, como sugerido anteriormente, começamos escolhendo o ponto de S mais próximo de p, denominado p' (v_6 no lado esquerdo da Figura 2.3). Posteriormente para $v_j \neq p'$, calculamos $d(v_j, p')$ e comparamos com $d(v_j, p)$. Neste exemplo, o pivô escolhido é v_3. Determinamos então $\bar{\alpha}$ que minimiza $d(p, \alpha v_3 + (1-\alpha)v_6)$.

Atualizamos $p'' = \bar{\alpha}v_3 + (1-\bar{\alpha})v_6$ como o próximo iterado. A partir de p'' o próximo pivô será o v_2 e atualizando temos $p''' = \bar{\alpha}v_2 + (1-\bar{\alpha})p''$.

Repetimos o processo até que $d(p_{r+1}, p) < \varepsilon$, onde r é o número de iterações realizadas. Sendo assim, por construção, temos que $p_{r+1} \in \text{conv}(S)$ e que $d(p_{r+1}, p) < \varepsilon$, ou seja, temos um ponto no $\text{conv}(S)$ suficientemente próximo de p.

Para o outro caso fazemos uma iteração completa, exatamente como descrito acima. Para p'' vemos que $\forall v_j, d(v_j, p) > d(v_j, p'')$, que é um critério de parada do algoritmo. Observe que usando p'' e p obtemos um vetor normal a um hiperplano que separa o ponto p do $\text{conv}(S)$. Logo, o ponto consultado p não pertence ao envoltório convexo de S.

A boa definição e corretude do Algoritmo 4 seguem de um teorema de separação denominado *dualidade de distâncias* que será apresentado a seguir. Antes, apresentamos um resultado auxiliar. Em [1] tal resultado é demonstrado usando o conceito de célula de Voronoi. Aqui apresentaremos uma demonstração alternativa que utiliza o Teorema 1.17.

Lema 2.2. *Sejam $S = \{v_1, v_2, \ldots, v_n\} \subset \mathbb{R}^m$ e $p \in \mathbb{R}^m$. Temos que $p \notin \text{conv}(S)$ se, e somente se, existe $p' \in \text{conv}(S)$ tal que $d(v_j, p) > d(v_j, p')$, $\forall v_j \in S$.*

Demonstração. Como $\text{conv}(S)$ é um conjunto não-vazio, fechado, e convexo, pelo Teorema 1.17, se $p \notin \text{conv}(S)$, existe um único $p^+ \in \text{conv}(S)$ tal que

$$\|v - p\| > \|v - p^+\|, \quad \forall v \in \text{conv}(S) \setminus \{p^+\}.$$

Como todo $v_j \in S$ pertence ao $\text{conv}(S)$, usando $p' = p^+$ provamos que $d(v_j, p) > d(v_j, p'), \forall v_j \in S$.

Por outro lado, considere agora que existe $p' \in \text{conv}(S) \setminus \{p\}$ tal que $d(v_j, p) > d(v_j, p'), \forall v_j \in S$, isto é, $\|v_j - p\| > \|v_j - p'\|, \forall v_j \in S$. Em particular,

$$\|v_j - p\| > \|v_j - p'\|, \forall v_j \in E,$$

em que $E \subset S$ é o subconjunto de S dos pontos extremos de $\text{conv}(S)$. Daí, de $\|v_j - p\|^2 > \|v_j - p'\|^2$, segue que:

$$\frac{\|p\|^2 - \|p'\|^2}{2} > (p - p')^T v_j, \quad \forall v_j \in E. \tag{2.2}$$

Além disso, $\forall v \in \text{conv}(S)$, v é combinação convexa dos v_j's em E, isto é,

$$v = \sum_{j=1}^{|E|} \alpha_j v_j, \text{ em que } \sum_{j=1}^{|E|} \alpha_j = 1, \quad \alpha_j \geq 0, \quad \forall j. \tag{2.3}$$

Então, de (2.2), temos que $\forall v \in \text{conv}(S)$

$$(p-p')^T v = (p-p')^T \left(\sum_{j=1}^{|E|} \alpha_j v_j \right) < \frac{\|p\|^2 - \|p'\|^2}{2}. \tag{2.4}$$

Em particular, para $v = p'$ temos

$$\frac{\|p\|^2 - \|p'\|^2}{2} > (p-p')^T p', \tag{2.5}$$

do que segue que

$$\begin{aligned}(p-p')^T p + \frac{\|p\|^2 - \|p'\|^2}{2} &> (p-p')^T p + (p-p')^T p' \\ &= (p-p')^T (p+p') \\ &= \|p\|^2 - \|p'\|^2,\end{aligned}$$

e portanto

$$(p-p')^T p > \frac{\|p\|^2 - \|p'\|^2}{2}. \tag{2.6}$$

De (2.4) e (2.6) concluímos que $p \notin \text{conv}(S)$. □

Assim, se em um dado $p' \in \text{conv}(S)$ não existir pivô, temos que o hiperplano bissetor ortogonal ao segmento $p'p$, dado por

$$H = \left\{ x \in \mathbb{R}^m \ : \ (p-p')^T x = \frac{\|p\|^2 - \|p'\|^2}{2} \right\}, \tag{2.7}$$

separa estritamente p de $\text{conv}(S)$, como representado na Figura 2.4.

Uma consequência imediata do resultado anterior é o seguinte teorema de alternativas.

Teorema 2.3 (Dualidade de Distâncias)**.** *Sejam $S = \{v_1, v_2, \ldots, v_n\} \subset \mathbb{R}^m$ e $p \in \mathbb{R}^m$. Exatamente uma das duas condições é satisfeita*

1. *Existe $p' \in \text{conv}(S)$ tal que $d(v_j, p) > d(v_j, p')$, $\forall v_j \in S$;*
2. *Para todo $p' \in \text{conv}(S)$, existe $v_j \in S$ tal que $d(v_j, p) \leq d(v_j, p')$.*

De acordo com o Lema 2.2 temos que a primeira condição do Teorema 2.3 é verdadeira se, e somente se, p não pertence ao $\text{conv}(S)$, enquanto a segunda é verdadeira se, e somente se, p pertence ao $\text{conv}(S)$.

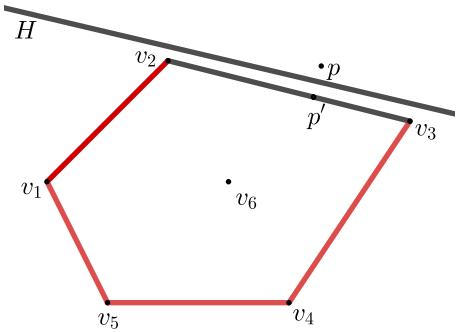

Figura 2.4 – Hiperplano bissetor ortogonal ao segmento $p'p$ que separa p de $\text{conv}(S)$.

Com isso temos a justificativa para o critério de parada do Passo 2 e a boa definição do Passo 3 do Algoritmo 4.

O lema abaixo será útil na análise do Algoritmo Geométrico. Ele apresenta caracterizações equivalentes para um pivô.

Lema 2.4. *Sejam $p' \in \text{conv}(S)$, $v_j \in S$ e $p \in \mathbb{R}^m$ o ponto a ser consultado. As seguintes afirmações são equivalentes:*

(a) $\|v_j - p\| \leq \|v_j - p'\|$;

(b) $2v_j^T(p' - p) \leq \|p'\|^2 - \|p\|^2$;

(c) $(p' - p)^T(v_j - p') \leq -\frac{1}{2}\|p' - p\|^2$;

(d) $(p' - p)^T(v_j - p) \leq \frac{1}{2}\|p' - p\|^2$.

Demonstração. (a) $\Rightarrow$ (b): De (a) temos que $\|v_j - p\|^2 \leq \|v_j - p'\|^2$. Expandindo

$$\|v_j\|^2 - 2v_j^T p + \|p\|^2 \leq \|v_j\|^2 - 2v_j^T p' + \|p'\|^2,$$

e em seguida simplificando, obtemos: $2v_j^T(p' - p) \leq \|p'\|^2 - \|p\|^2$.

(b) $\Rightarrow$ (c): De (b) temos que $2(v_j - p' + p')^T(p' - p) \leq \|p'\|^2 - \|p\|^2$. Isto implica em

$$2(v_j - p')^T(p' - p) + 2\|p'\|^2 - 2p'^T p \leq \|p'\|^2 - \|p\|^2,$$

ou ainda

$$2(v_j - p')^T(p' - p) \leq - \|p'\|^2 + 2p'^T p - \|p\|^2 ,$$

e portanto $(p' - p)^T(v_j - p') \leq -\frac{1}{2} \|p' - p\|^2$.

(c) $\Rightarrow$ (d): Da desigualdade acima, somando e subtraindo p em $(v_j - p')$ obtemos

$$(p' - p)^T(v_j - p) - \|p' - p\|^2 \leq -\frac{1}{2} \|p' - p\|^2 ,$$

logo $(p' - p)^T(v_j - p) \leq \frac{1}{2} \|p' - p\|^2$.

(d) $\Rightarrow$ (a): Da desigualdade anterior, temos que

$$(p' - v_j + v_j - p)^T(v_j - p) \leq \frac{1}{2} \|p' - p\|^2 ,$$

e então

$$(p' - v_j)^T(v_j - p) + \|v_j - p\|^2 \leq \frac{1}{2} \|p' - p\|^2 .$$

Somando e subtraindo p' em $v_j - p$ temos que

$$- \|v_j - p'\|^2 + (p' - v_j)^T(p' - p) + \|v_j - p\|^2 \leq \frac{1}{2} \|p' - p\|^2 .$$

Organizando, somando e subtraindo p em $(p' - v_j)^T$ ficamos com

$$\|v_j - p\|^2 + \|p' - p\|^2 + (p - v_j)^T(p' - p) \leq \frac{1}{2} \|p' - p\|^2 + \|v_j - p'\|^2 .$$

Agrupando os temos semelhantes, a inequação acima implica em

$$\frac{1}{2} \|v_j - p\|^2 + \frac{1}{2} \|v_j - p\|^2 - (v_j - p)^T(p' - p) + \frac{1}{2} \|p' - p\|^2 \leq \|v_j - p'\|^2 .$$

Utilizando produto notáveis e simplificado temos que $\|v_j - p\| \leq \|v_j - p'\|$. □

Proposição 2.5. *Considere o problema* (1.2)*, com* $f(x) = \frac{1}{2}\|x - p\|^2$ *e* $\Omega = \text{conv}(S)$*. Se* $v_j \in S$ *é um pivô para* $p' \in \text{conv}(S)$*, então* $d = v_j - p'$ *é direção factível e de descida para* f *a partir de* p'*.*

Demonstração. Como $p', v_j \in \text{conv}(S)$ e $\text{conv}(S)$ é convexo, segue que $p' + \alpha(v_j - p') = (1 - \alpha)p' + \alpha v_j \in \text{conv}(S)$, para todo $\alpha \in [0, 1]$. Logo, d é factível a partir de p' . Do item (c) do Lema 2.4, segue que d é direção de descida para f a partir de p'. □

Corolário 2.6. *Sejam p', v_j e p'' definidos pelo Algoritmo 4. Então $d(p'', p) < d(p', p)$.*

Demonstração. Segue diretamente do fato de $d = v_j - p'$ ser direção de descida e da escolha de $\bar{\alpha}$ no Passo 4. □

No Passo 4, dado um pivô v_j, buscamos $\bar{\alpha} \in [0, 1]$ tal que $p'' = (1 - \bar{\alpha})p' + \bar{\alpha}v_j$ seja o ponto no segmento $p'v_j$ mais próximo de p, ou seja, a projeção de p no segmento $p'v_j$. Vejamos que há uma fórmula explícita para $\bar{\alpha}$.

Proposição 2.7. *Seja $\phi(\alpha) = \frac{1}{2}\|p''(\alpha) - p\|^2$, em que $p''(\alpha) = (1 - \alpha)p' + \alpha v_j$. O minimizador global de $\phi(\alpha)$ em $\mathbb{R}$ é dado por*

$$\bar{\alpha} = -\frac{(p' - p)^T(v_j - p')}{\|v_j - p'\|^2}. \tag{2.8}$$

Demonstração. Basta aplicar a fórmula (1.9), considerando que $f(x) = \frac{1}{2}\|x-p\|^2$ é uma quadrática estritamente convexa, e usar $x_k = p''(0) = p'$ e $d_k = v_j - p'$. □

Proposição 2.8. *Seja v_j um pivô para $p' \neq p$ e assuma que $d(p', p) \leq d(v_j, p)$. Então $\bar{\alpha}$ de (2.8) está no intervalo $(0, 1]$.*

Demonstração. Por hipótese, v_j é um pivô para p'. Logo, do Lema 2.4, item (c), temos que $\bar{\alpha} > 0$. Basta provar então que $\bar{\alpha} \leq 1$. Note que

$$|\bar{\alpha}| = \frac{|(p' - p)^T(v_j - p')|}{\|v_j - p'\|^2} \leq \frac{\|p' - p\| \, \|v_j - p'\|}{\|v_j - p'\|^2}.$$

Simplificando e usando que $\|p' - p\| \leq \|v_j - p\| \leq \|v_j - p'\|$, temos

$$|\bar{\alpha}| \leq \frac{\|p' - p\|}{\|v_j - p'\|} \leq 1.$$

□

Como inicializamos o Algoritmo 4 com o ponto inicial $p' \in \arg\min\{d(v, p) : v \in S\}$, e em vista do Corolário 2.6, temos que a condição $d(p', p) \leq d(v_j, p)$ da proposição acima é satisfeita em toda iteração do Algoritmo Geométrico. Assim fica evidente que para $p' \in \text{conv}(S)$ e $v_j \in S$ temos que $p'' \in \text{conv}(S)$.

Por fim, se $p \in \text{conv}(S)$, temos que a cada iteração, a distância $d(p', p)$ é reduzida[3] e o algoritmo para com uma solução aproximada quando $d(p', p) < \varepsilon$ (Passo 5).

No entanto, este critério de parada pode ser problemático quando o diâmetro de $\text{conv}(S)$ é muito pequeno. Por exemplo, considere que $S \subset B(0, \varepsilon/2)$. Suponha $p \notin \text{conv}(S)$, mas $p \in B(0, \varepsilon/2)$. Assim, $d(p, \text{conv}(S)) < \varepsilon$ e para todo $p' \in \text{conv}(S)$ temos que $d(p', p) < \varepsilon$, isto é, o algoritmo pararia declarando $p \in \text{conv}(S)$, mas na verdade $p \notin \text{conv}(S)$.

Para evitar essa situação, consideramos $R = \max\{d(v_j, p) : v_j \in S\}$ e trocamos o critério de parada para

$$\frac{d(p', p)}{R} < \varepsilon$$

isto é, quando o *erro relativo* é menor que a tolerância ε.

No caso em que $p \notin \text{conv}(S)$, o Algoritmo 4 retornará uma testemunha p' com a qual podemos construir o hiperplano separador definido em (2.7). A distância $d(p', p)$ dessa testemunha para p aproxima a distância de p a $\text{conv}(S)$ por um fator de no máximo dois.

Teorema 2.9. *Sejam $p \notin \text{conv}(S)$ e p' uma p-testemunha e*

$$\Delta \coloneqq d(p, \text{conv}(S)) = \min\{d(p, v) : v \in \text{conv}(S)\}. \tag{2.9}$$

Então,

$$\frac{1}{2} d(p, p') \leq \Delta \leq d(p, p'). \tag{2.10}$$

Demonstração. A desigualdade da direita segue direto da definição de Δ. Para provar a desigualdade da esquerda, seja p' uma p-testemunha.

Já vimos no Lema 2.2 que o hiperplano H de (2.7) que bisseta ortogonalmente o segmento $p'p$ separa p do $\text{conv}(S)$. Assim, denotando por $\beta = \frac{1}{2}(\|p\|^2 - \|p'\|^2)$ temos que $H = \{x \in \mathbb{R}^m : (p - p')^T x = \beta\}$, $p \in H^+ = \{x \in \mathbb{R}^m : (p - p')^T x > \beta\}$ e $\text{conv}(S) \subset H_- = \{x \in \mathbb{R}^m : (p - p')^T x < \beta\}$.

De (2.4), temos que

$$(p - p')^T v < \frac{\|p\|^2 - \|p'\|^2}{2}, \quad \forall v \in \text{conv}(S),$$

[3] Quantificaremos a redução na distância $d(p', p)$ a cada iteração no Capítulo 3.

em particular, se p^+ é a projeção de p em $\text{conv}(S)$, então

$$(p - p')^T p^+ < \frac{\|p\|^2 - \|p'\|^2}{2}. \tag{2.11}$$

Além disso

$$\begin{aligned}
\|p - p^+\|^2 &= \|p - p' + p' - p^+\|^2 \\
&= \|p - p'\|^2 + 2(p - p')^T(p' - p^+) + \|p' - p^+\|^2 \\
&= \|p - p'\|^2 + 2(p - p')^T p' - 2(p - p')^T p^+ + \|p' - p^+\|^2 \\
&> \|p - p'\|^2 + 2(p - p')^T p' + \|p'\|^2 - \|p\|^2 + \|p' - p^+\|^2 \\
&= \|p - p'\|^2 - (\|p'\|^2 - 2p^T p' + \|p'\|^2) + \|p' - p^+\|^2 \\
&= \|p' - p^+\|^2,
\end{aligned}$$

em que a desigualdade vem de (2.11). Então temos que $\|p' - p^+\| < \|p - p^+\|$ e portanto

$$\|p - p'\| \leq \|p - p^+\| + \|p' - p^+\| \leq 2\|p - p^+\|.$$

□

Em relação ao custo computacional, cada iteração do Algoritmo 4 demanda, no pior caso, $O(nm)$ operações aritméticas: $O(m)$ é o custo de cada produto interno, o que pode ocorrer até n vezes caso a lista S precise ser percorrida por completo. No entanto, a expectativa é que na prática, o algoritmo encontre um pivô simples antes de percorrer toda a lista S, levando a custo por iteração (em média) menor que $O(nm)$.

2.3 Relação com o método de Frank-Wolfe

Uma alternativa para responder a pergunta do PIEC seria, por exemplo, resolver o problema de projeção:

$$\begin{aligned}
\min_{v \in \mathbb{R}^m} \quad & \frac{1}{2}\|v - p\|^2, \\
\text{s.a.} \quad & v \in \text{conv}(S),
\end{aligned} \tag{2.12}$$

em que $S = \{v_1, v_2, \ldots, v_n\} \subset \mathbb{R}^m$ e $p \in \mathbb{R}^m$, declarando $p \in \text{conv}(S)$ se valor ótimo for zero.

Note que não conhecemos uma descrição do $\text{conv}(S)$ em termos de desigualdades lineares, então reescreveremos esse problema utilizando combinações convexas de elementos de S. Do fato que $v \in \text{conv}(S)$ temos que v pode ser escrito como a seguinte combinação convexa:

$$v = \sum_{i=1}^{n} x_i v_i = Sx, \text{ com } x \geq 0, \text{ e } \sum_{i=1}^{n} x_i = 1,$$

em que, permitido certo abuso de notação, construímos a matriz $m \times n$: $S = [v_1 \ \ldots \ v_n]$. Assim, sendo $e \in \mathbb{R}^n$ o vetor cujas componentes são todas iguais a um, temos o seguinte problema de programação quadrática:

$$\begin{aligned} \min_{x} \quad & \frac{1}{2} \|Sx - p\|^2 \\ \text{s.a.} \quad & e^T x = 1 \\ & x \geq 0. \end{aligned} \tag{2.13}$$

Se aplicarmos o método de Frank-Wolfe ao problema (2.13), a cada iteração, devemos resolver o seguinte subproblema:

$$\begin{aligned} \min_{x} \quad & \nabla f(x_k)^T (x - x_k) \\ \text{s.a.} \quad & e^T x = 1 \\ & x \geq 0, \end{aligned} \tag{2.14}$$

em que $f(x) = \frac{1}{2}\|Sx - p\|^2$ e $\nabla f(x) = S^T(Sx - p)$.

Podemos encontrar a solução analítica do problema (2.14) utilizando as condições KKT (1.4). Seja $F(x) = \nabla f(x_k)^T (x - x_k)$, temos que $\nabla F(x) = \nabla f(x_k)$ e as condições KKT para o subproblema (2.14) são

$$\nabla f(x_k) + \lambda e - \mu = 0, \tag{2.15}$$

$$e^T x = 1, \quad x \geq 0, \quad \mu \geq 0, \tag{2.16}$$

$$\mu_i x_i = 0, \quad \forall i. \tag{2.17}$$

Tomando o produto interno de (2.15) com x e utilizando (2.17) e (2.16) temos que:

$$\lambda = -\nabla f(x_k)^T x. \tag{2.18}$$

Substituindo (2.18) em (2.15):

$$\nabla f(x_k) - (\nabla f(x_k)^T x)e - \mu = 0. \tag{2.19}$$

De (2.16), temos que $\mu_i \geq 0$, para $i = 1, \ldots, n$, e então de (2.19), obtemos

$$\nabla f(x_k)_i \geq \nabla f(x_k)^T x, \quad \text{para } i = 1, \ldots, n.$$

Sendo assim, seja

$$j \in \arg\min\{\nabla f(x)_i \ : \ 1 \leq i \leq n\}. \tag{2.20}$$

Não é difícil verificar que $x = e_j$ (j-ésimo vetor canônico de $\mathbb{R}^n$) e $\mu_i = 0, \forall i \neq j$ satisfazem as condições (2.15)–(2.17), com λ de (2.18). Como (2.14) é um problema de programação linear, as condições KKT são também suficientes e portanto $\bar{x}_k = e_j$ é solução de (2.14).

Como $\nabla f(x_k) = S^T(Sx_k - p)$, temos que

$$\nabla f(x_k)_i = e_i^T \nabla f(x_k) = e_i^T S^T(Sx_k - p) = (Se_i)^T(Sx_k - p) = v_i^T(p_k - p), \tag{2.21}$$

em que $p_k \coloneqq Sx_k$. Logo, para obter um índice j como em (2.20), devemos avaliar os produtos internos $v_i^T(p_k - p)$ e escolher o índice correspondente ao menor deles.

Se denotarmos o k-ésimo iterado do Algoritmo Geométrico por p_k e usarmos a caracterização do Lema 2.4(b), vemos que o Algoritmo 4 escolhe v_j tal que

$$v_j^T(p_k - p) \leq \frac{\|p_k\|^2 - \|p\|^2}{2},$$

enquanto Frank-Wolfe toma v_j tal que o produto interno $v_j^T(p_k - p)$ é mínimo.

Apesar de no pior caso ambos os algoritmos terem que avaliar toda a lista de produtos internos $v_i^T(p_k - p)$, é de se esperar que, em geral, a iteração do Algoritmo Geométrico demande menos avaliações, uma vez que ele busca apenas um pivô simples, enquanto Frank-Wolfe busca "o melhor pivô".

Além disso, vemos que o Algoritmo Geométrico pode ser visto como um método de Frank-Wolfe "inexato" [12], ou ainda, que o método de Frank-Wolfe aplicado a (2.13) pode ser visto como um "Algoritmo Geométrico Ganancioso".

A expressão (2.21) por sua vez, mostra que a iteração do gradiente condicional, quando aplicado ao problema (2.13), pode ser implementada de forma mais econômica, evitando produtos matriz-vetor da forma Sx e S^Ty que custam $O(mn)$ cada.

3 ANÁLISE DE COMPLEXIDADE

Neste capítulo apresentamos a análise de pior caso para Algoritmo Geométrico. Veremos que a redução na distância $d(p', p)$ de uma iteração para outra depende do ângulo $\angle pp'v$ e que em no máximo $O(1/\varepsilon^2)$ iterações o algoritmo encontra p' tal que $d(p', p) < \varepsilon R$. Em certos casos particulares, veremos que com uma escolha mais restrita dos pivôs é possível melhorar a complexidade para $O(\ln 1/\varepsilon)$. Toda a análise é baseada em [1], porém alguns lemas e resultados auxiliares são demonstrados de forma diferente, fazendo uso de interpretações alternativas para os pivôs.

3.1 Caracterização geométrica de pivô simples

Da definição de pivô simples da Seção 2.2, temos que $v \in S$ é pivô para $p' \in \text{conv}(S)$ quando $\|v - p\| \leq \|v - p'\|$. Por outro lado, devido a escolha do iterado inicial no Passo 1 do Algoritmo 4 e do Corolário 2.6, temos também que $\|p' - p\| \leq \|v - p\|$, que junto com a definição de pivô simples, implica em $\|p' - p\| \leq \|v - p'\|$. Dessas duas últimas desigualdades, temos que qualquer pivô v para p' deve estar fora da bola de centro em p e raio $\delta = \|p' - p\|$ e também fora da bola de centro em p' e δ.

Além disso, do Lema 2.4(b), temos que v é pivô para p' se, e somente se,

$$(p - p')^T v \geq \frac{\|p\|^2 - \|p'\|^2}{2}.$$

Assim, sendo $H = \{x \in \mathbb{R}^m \ : \ (p - p')^T x = (\|p\|^2 - \|p'\|^2)/2\}$ o hiperplano bissetor ortogonal ao segmento $p'p$, temos que v deve pertencer ao mesmo semi-espaço que p.

A Figura 3.1 ilustra a discussão acima em duas dimensões (no plano definido por p, p' e v).

3.2 Complexidade de iteração

Sejam $p' \in \text{conv}(S)$ o iterado corrente do Algoritmo Geométrico, $v \in S$ um pivô simples para p' e denote o novo iterado por p''. O lema a seguir trata da redução de $d(p'', p)$ em relação a $d(p', p)$.

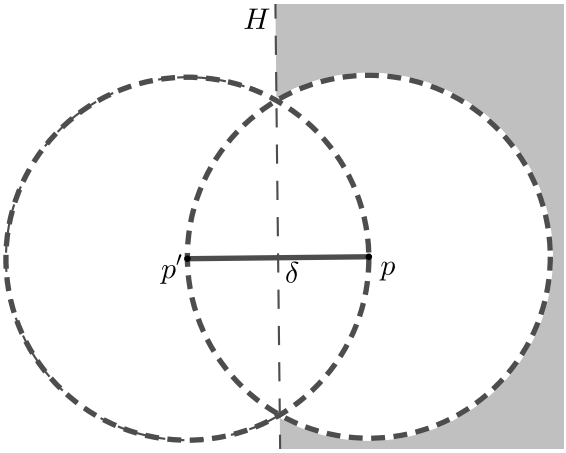

Figura 3.1 – A região em cinza mostra onde podem estar os p-pivôs para p'.

Lema 3.1. *Sejam p, p', v pontos distintos em $\mathbb{R}^m$. Suponha que $d(v,p) \leq d(v,p')$. Seja p'' o ponto no segmento $p'v$ que está mais próximo de p. Defina $\delta = d(p',p)$, $\delta' = d(p'',p)$, $r = d(v,p)$ e assuma $\delta \leq r$. Então,*

$$\delta' \leq \delta\sqrt{1 - \frac{\delta^2}{4r^2}}. \tag{3.1}$$

Demonstração. Lembrando do Passo 4 do Algoritmo 4, e usando v como pivô para p', temos que $p'' = (1-\bar{\alpha})p' + \bar{\alpha}v$ e então

$$\begin{aligned}
\|p'' - p\|^2 &= \|\alpha(v-p') + (p'-p)\|^2 \\
&= \alpha^2\|v-p'\|^2 + \alpha 2(v-p')^T(p'-p) + \|p'-p\|^2 \\
&= \frac{|(p'-p)^T(v-p')|^2}{\|v-p'\|^2} - 2\frac{|(p'-p)^T(v-p')|^2}{\|v-p'\|^2} + \|p'-p\|^2 \\
&= -\frac{\|p'-p\|^2\|v-p'\|^2\cos^2\theta}{\|v-p'\|^2} + \|p'-p\|^2 \\
&= \left(1-\cos^2\theta\right)\|p'-p\|^2, \qquad (3.2)
\end{aligned}$$

em que a terceira igualdade segue de $\alpha = \bar{\alpha}$ de (2.8) com $v_j = v$ e $\theta \coloneqq \angle pp'v$.

Agora, tendo em vista a caracterização de pivô simples da Seção 3.1, não é difícil notar que, para $\|v-p\| = r$, $\cos\theta$ será mínimo quando v estiver sobre o hiperplano $H = \{x \in \mathbb{R}^m \ : \ (p-p')^T x = (\|p\|^2 -$

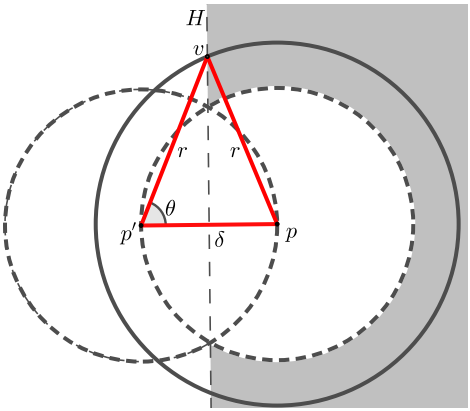

Figura 3.2 – Pior caso para $\cos\theta$ quando v é pivô simples.

$\|p'\|^2)/2\}$. Neste caso $\cos\theta = \delta/2r$, como ilustra a Figura 3.2. Desta observação e de (3.2) segue (3.1). □

Corolário 3.2. *Sejam $p, p', p'', v, r, \delta, \delta'$ como no Lema 3.1. Se $p'' \neq v$, então:*

$$\delta' \leq \delta\sqrt{1 - \frac{\delta^2}{4R^2}} \leq \delta \exp\left(-\frac{\delta^2}{8R^2}\right), \tag{3.3}$$

em que $R = \max\{d(v_j, p) : v_j \in S\}$.

Demonstração. A primeira desigualdade segue do Lema 3.1 e da definição de R. Para provar a desigualdade seguinte, utilizamos que para $x \neq 0$, $1 + x < \exp x$. Com isso, sendo $x = -\delta^2/(4R^2)$ note que:

$$\delta\sqrt{1 - \frac{\delta^2}{4R^2}} \leq \delta\sqrt{\exp\left(\frac{\delta^2}{4R^2}\right)} = \delta \exp\left(-\frac{\delta^2}{8R^2}\right).$$

□

Lema 3.3. *Assuma que $p \in \text{conv}(S)$. Escolha $p_0 \in \text{conv}(S)$, seja $\delta_0 = d(p_0, p)$. Assuma $\delta_0 \leq R_0 = \min\{d(v_i, p), \quad \forall v_i \in S\}$. Seja $k \equiv k(\delta_0)$ o número de iterações para o Algoritmo Geométrico obter $p_k \in \text{conv}(S)$ tal que*

$$\delta_k < \frac{\delta_0}{2} \leq \delta_j, \quad j = 1, \ldots, k-1,$$

em que $\delta_j = d(p_j, p)$. *Então,* k *satisfaz*

$$k = k(\delta_0) \leq \lceil N_0 \rceil,$$

com $N_0 = N(\delta_0) = (32 \ln 2)(R/\delta_0)^2 < 23(R/\delta_0)^2$.

Demonstração. Para cada $j = 1, \ldots, k-1$ o Corolário 3.2 é aplicável, e além disso $\dfrac{\delta_0}{2} \leq \delta_j$. Então, repetindo a aplicação de (3.3) do Corolário 3.2, e pela monotonicidade da função exponencial temos que:

$$\delta_j < \delta_{j-1} \exp\left(-\frac{\delta_{j-1}^2}{8R^2}\right) \leq \delta_{j-1} \exp\left(-\frac{\delta_0^2}{32R^2}\right) < \ldots \leq \delta_0 \exp\left(-\frac{\delta_0^2}{32R^2}\right)^j.$$

Segue que

$$\delta_{k-1} < \delta_0 \exp\left(-\frac{(k-1)\delta_0^2}{32R^2}\right). \tag{3.4}$$

De (3.3) e (3.4) chegamos a

$$\delta_k < \delta_{k-1} \exp\left(-\frac{\delta_{k-1}^2}{8R^2}\right) \leq \delta_0 \exp\left(-\frac{k\delta_0^2}{32R^2}\right). \tag{3.5}$$

Assim, para que $\delta_k < \delta_0/2$, é suficiente que

$$\exp\left(-\frac{k\delta_0^2}{32R^2}\right) \leq \frac{1}{2},$$

e para isso, basta que $k = \lceil N_0 \rceil$, em que

$$N_0 = (32 \ln 2)\left(\frac{R}{\delta_0}\right)^2.$$

Portanto, em no máximo $k = k(\delta_0) \leq \lceil N_0 \rceil$ iterações, o gap inicial δ_0 é reduzido pela metade. □

Teorema 3.4 (Complexidade do Algoritmo Geométrico)**.** *O Algoritmo Geométrico resolve corretamente o problema de inclusão no envoltório convexo do seguinte modo:*

(a) se $p \in \text{conv}(S)$, *dados* $\varepsilon > 0, p_0 \in \text{conv}(S)$, *com* $\delta_0 = d(p, p_0) \leq R_0 = \min\{d(v_i, p) : i = 1, \ldots, n\}$, *o número máximo de iterações* K_ε *para computar um ponto* $p_\varepsilon \in \text{conv}(S)$ *tal que* $d(p_\varepsilon, p) < \varepsilon R$ *é de*

$$K_\varepsilon \leq \frac{48}{\varepsilon^2} = O(\varepsilon^{-2}). \tag{3.6}$$

(b) se $p \notin \text{conv}(S)$ o número de iterações K_Δ para computar uma $p-$testemunha, é de

$$K_\Delta \leq \frac{48R^2}{\Delta^2} = O\left(\frac{R^2}{\Delta^2}\right), \tag{3.7}$$

em que $\Delta = \min\{d(x,p) : x \in \text{conv}(S)\}$.

Demonstração. (a) Pelo Lema 3.3, para reduzir δ_0 para $\delta_0/2$, no pior caso, o Algoritmo Geométrico requer $k(\delta_0)$ iterações. Então, para reduzir o gap de $\delta_0/2$ para $\delta_0/4$, o Algoritmo 4 requer no máximo $k(\delta_0/2)$ iterações e assim por diante. Logo, para um inteiro não-negativo r, o número de iterações para reduzir o gap de $\delta_0/2^r$ para $\delta_0/2^{r+1}$ é dado por

$$k\left(\frac{\delta_0}{2^r}\right) \leq \lceil N_0\left(\frac{\delta_0}{2^r}\right)\rceil = \lceil 2^{2r}N_0\rceil \leq 2^{2r}\lceil N_0\rceil. \tag{3.8}$$

Seja t o menor índice tal que $\delta_0/2^t < \varepsilon R$, isto é,

$$2^{t-1} \leq \frac{\delta}{R\varepsilon} < 2^t.$$

Então, o número total de iterações K_ε satisfaz

$$\begin{aligned} K_\varepsilon &\leq \lceil N_0\rceil\left(1 + 2^2 + \ldots + 2^{2(t-1)}\right) \\ &\leq \lceil N_0\rceil\frac{2^{2t}-1}{3} \leq \lceil N_0\rceil\frac{2^{2t}}{3} \leq \lceil N_0\rceil 2 \times 2^{2(t-1)} \\ &\leq \lceil N_0\rceil\frac{2\delta_0^2}{R^2\varepsilon^2} \leq (N_0+1)\frac{2\delta_0^2}{R^2\varepsilon^2}. \end{aligned}$$

Do Lema 3.3 temos que

$$K_\varepsilon \leq \left(23\frac{R^2}{\delta_0^2}+1\right)\frac{2\delta_0^2}{R^2\varepsilon^2} = \left(23 + \frac{\delta_0^2}{R^2}\right)\frac{2}{\varepsilon^2}.$$

Como $p \in \text{conv}(S)$ e da definição de R segue que $\delta_0 = d(p_0,p) \leq R$ e assim

$$K_\varepsilon \leq (23+1)\,\frac{2}{\varepsilon^2} = \frac{48}{\varepsilon^2}.$$

(b) Suponha agora que $p \notin \text{conv}(S)$. No item (a) encontramos $p_\varepsilon \in \text{conv}(S)$ tal que $d(p_\varepsilon, p) < \varepsilon R$. Mas agora, como $p \notin \text{conv}(S)$, o

mínimo de $d(p', p)$ para $p' \in \text{conv}(S)$ será Δ. Então, tomando $\varepsilon = \frac{\Delta}{R}$ e usando (3.6) obtemos

$$K_\Delta \leq \frac{48R^2}{\Delta^2} = O\left(\frac{R^2}{\Delta^2}\right).$$

□

Observação 3.5. De acordo com [12] as iterações do método de Frank-Wolfe para o problema (1.2) satisfazem:

$$f(x_k) - f(x_*) = O\left(\frac{1}{k}\right).$$

Em particular, para o problema de inclusão no envoltório convexo, temos que

$$f(x) = (1/2)\|Sx - p\|^2 \quad \text{e} \quad \Omega = \Delta_n = \{x \in \mathbb{R}^n \ : \ e^T x = 1, x \geq 0\}.$$

Para o PIEC, o Algoritmo Geométrico gera uma sequência de pontos $p_k \in \text{conv}(S)$ que se aproxima de p. Cada p_k pode ser associado a um $x_k \in \Delta_n$ tal que $p_k = Sx_k$. Desse modo

$$f(x_k) = \frac{1}{2}\|Sx_k - p\|^2 = \frac{1}{2}\|p_k - p\|^2.$$

Se $p \in \text{conv}(S)$, dado $\varepsilon \in (0, 1)$, de acordo com o Teorema 3.4, o Algoritmo Geométrico em $K_\varepsilon \leq 48\varepsilon^{-2}$ iterações obtém um ponto $p_\varepsilon \in \text{conv}(S)$ tal que $d(p_\varepsilon, p) < \varepsilon R$.

Dado um índice k, da desigualdade $\lceil 48\varepsilon^{-2} \rceil \geq k$, temos que $\varepsilon \leq \sqrt{48/k}$, e assim:

$$\|p_k - p\| < \sqrt{\frac{48}{k}} R.$$

Equivalentemente,

$$f(x_k) = f(x_k) - f(x_*) < \frac{24R^2}{k} = O\left(\frac{1}{k}\right).$$

Sendo assim, quando $p \in \text{conv}(S)$ temos que a complexidade do Algoritmo Geométrico é similar a de Frank-Wolfe para o PIEC. No entanto, como já discutimos na Seção 2.3, o Algoritmo Geométrico pode ter um desempenho prático superior pois necessita apenas de um pivô simples a cada iteração, em contraste ao método de Frank-Wolfe que necessita do "melhor pivô". Além disso, quando $p \notin \text{conv}(S)$, segundo (3.7), a complexidade de iteração do algoritmo geométrico depende apenas da "geometria do problema", ou seja, independe da tolerância ε.

3.3 Pivôs estritos

Como vimos no Lema 3.1, mais especificamente na equação (3.2), a redução na distância $d(p', p)$ a cada iteração do Algoritmo 4, depende do ângulo $\theta = \angle pp'v$:

$$\delta' = d(p'', p) = \|p'' - p\| = \sqrt{1 - \cos^2\theta}\|p' - p\| = \sqrt{1 - \cos^2\theta}\,\delta = \sin\theta\,\delta. \tag{3.9}$$

Assim, quanto mais próximo de zero estiver o ângulo θ, maior a redução. Em [1], uma definição mais restritiva de pivô é apresentada, visando melhorar a complexidade do algoritmo.

Definição 3.6. *Dado $p' \in \text{conv}(S)$, dizemos que $v_j \in S$ é um* pivô estrito *para p, se $\omega := \angle p'pv_j \geq \pi/2$.*

Analogamente, $v \in S$ é pivô estrito para p' quando

$$(p' - p)^T(v - p) \leq 0. \tag{3.10}$$

Evidentemente, pelo item (d) do Lema 2.4, temos que todo pivô estrito é pivô simples. Além disso, considerando o triângulo $pp'v$, se v é pivô estrito para p', *i.e.*, se $\omega = \angle p'pv \geq \pi/2$, então $\theta < \pi/2$ e $\sin\theta < 1$.

Como todo pivô estrito é pivô simples, i.e., $\|v - p\| \leq \|v - p'\|$, e assumindo $\|p' - p\| \leq \|v - p\|$, temos então que v deve estar fora das bolas de centros p e p', ambas de raio $\|p' - p\|$, e satisfazer

$$(p - p')^T v \geq (p - p')^T p.$$

A Figura 3.3 ilustra a região onde podemos encontrar pivôs estritos para p'.

O lema a seguir é análogo ao Lema 2.4, só que para pivôs estritos.

Lema 3.7. *Sejam $p' \in \text{conv}(S)$, $v_j \in S$ e $p \in \mathbb{R}^m$ o ponto a ser consultado. As seguintes afirmações são equivalentes:*

(a) $\|v_j - p\|^2 \leq \|v_j - p'\|^2 - \|p' - p\|^2$;

(b) $2v_j^T(p' - p) \leq 2p^T(p' - p)$;

(c) $(p' - p)^T(v_j - p') \leq -\|p' - p\|^2$;

(d) $(p' - p)^T(v_j - p) \leq 0$.

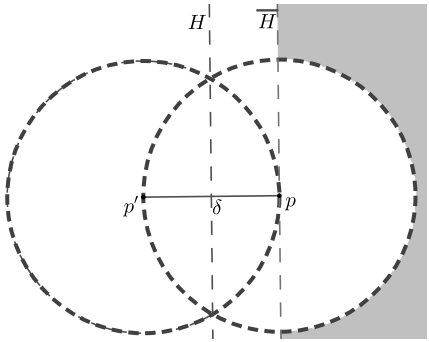

Figura 3.3 – A região cinza mostra onde podemos encontrar pivôs estritos.

Demonstração. Partimos de (d), que é a definição de pivô estrito e obtemos

$$(p' - p)^T v_j - (p' - p)^T p \leq 0.$$

Assim

$$2v_j^T(p' - p) \leq 2p^T(p' - p),$$

implicando em (b). De (b), segue que

$$(v_j - p' + p')^T(p' - p) \leq p^T(p' - p).$$

Separando os termos temos

$$(v_j - p')^T(p' - p) + p'^T(p' - p) \leq p^T(p' - p).$$

e então

$$(p' - p)^T(v_j - p') \leq - \|p' - p\|^2,$$

provando (c).

Como

$$\begin{aligned} \|v_j - p\|^2 &= \|v_j - p' + p' - p\|^2 \\ &= \|v_j - p'\|^2 + 2(p' - p)^T(v_j - p') + \|p' - p\|^2 \\ &\leq \|v_j - p'\|^2 - 2\|p' - p\|^2 + \|p' - p\|^2 \\ &= \|v_j - p'\|^2 - \|p' - p\|^2, \end{aligned}$$

em que a desigualdade segue de (c). Assim obtemos (a).

Por fim

$$\begin{aligned}\|v_j - p'\|^2 &= \|v_j - p + p - p'\|^2 \\ &= \|v_j - p\|^2 + 2(p - p')^T(v_j - p) + \|p' - p\|^2 \\ &\leq \|v_j - p'\|^2 - \|p' - p\|^2 + 2(p - p')^T(v_j - p) + \|p' - p\|^2,\end{aligned}$$

em que a desigualdade segue de (a). Portanto, cancelando os termos, obtemos

$$2(p - p')^T(v_j - p) \geq 0,$$

o que prova (d). □

O próximo teorema é uma especialização da dualidade de distâncias, mais especificamente do Lema 2.2, para pivôs estritos.

Teorema 3.8. *Sejam $S = \{v_1, v_2, \ldots, v_n\} \subset \mathbb{R}^m$ e $p \in \mathbb{R}^m$. Temos que $p \notin \text{conv}(S)$ se, e somente se, existe $p' \in \text{conv}(S) \setminus \{p\}$ tal que $(p' - p)^T(v_j - p) > 0$, para todo $v_j \in S$.*

Demonstração. A demonstração segue as mesmas linhas daquela do Lema 2.2, mas usando a caracterização de pivô estrito dada pelo Lema 3.7. Como $\text{conv}(S)$ é um conjunto não-vazio, fechado, e convexo, pelo Teorema 1.17, se $p \notin \text{conv}(S)$, existe um único $p^+ \in \text{conv}(S)$ tal que

$$\|v - p\| > \left\|v - p^+\right\|, \quad \forall v \in \text{conv}(S) \setminus \{p^+\}.$$

Como todo $v_j \in S$ pertence ao $\text{conv}(S)$, e usando $p' = p^+$ mostramos que

$$\|v_j - p\| > \|v_j - p'\|, \quad \forall v_j \in S,$$

que, pelo Lema 3.7(a), mostra que não há pivô estrito para p'.

Por outro lado, considere agora que existe $p' \in \text{conv}(S) \setminus \{p\}$ tal que

$$(p' - p)^T(v_j - p) > 0, \quad \forall v_j \in S.$$

Pela caracterização do Lema 3.7(b), temos que

$$(p' - p)^T v_j > (p' - p)^T p, \quad \forall v_j \in S.$$

Como qualquer elemento de $\text{conv}(S)$ é combinação convexa dos elementos de S, obtemos

$$(p' - p)^T v > (p' - p)^T p, \quad \forall v \in \text{conv}(S),$$

implicando que $p \notin \text{conv}(S)$. □

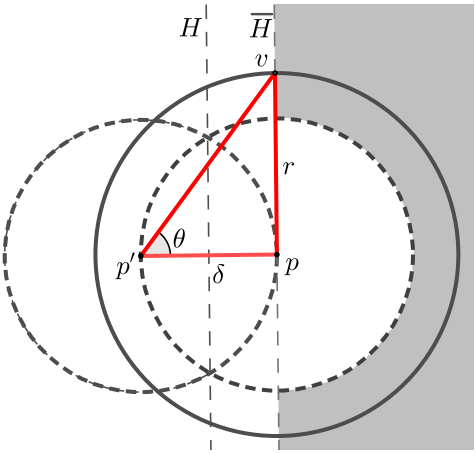

Figura 3.4 – Pior caso para $\cos\theta$ quando v é pivô estrito.

A complexidade utilizando pivôs estritos é deduzida do teorema a seguir.

Teorema 3.9. *Dado $p' \in \text{conv}(S)$, suponha que v_j é um pivô estrito para p'. Então se $\delta = d(p', p) \leq r = d(v_j, p)$ e $\delta' = d(p'', p)$, sendo p'' o ponto mais próximo de p no segmento $p'v_j$, temos:*

$$\delta' \leq \frac{\delta r}{\sqrt{r^2 + \delta^2}} \leq \delta\sqrt{1 - \frac{\delta^2}{2r^2}} \leq \delta \exp\left(-\frac{\delta^2}{4r^2}\right). \tag{3.11}$$

Demonstração. De (3.9), temos que $\delta' = \delta \sin\theta$. Lembrando que p, p', v_j e p'' estão no mesmo plano, não é difícil observar que para $\delta \leq r$ fixo e v_j tal que $\|v_j - p\| = r$, com v_j sendo pivô estrito, o maior valor para $\sin\theta$ ocorre quando $v_j \in \bar{H}$. A Figura 3.4 ilustra exatamente este fato. Neste pior caso, temos que $\sin\theta = r/\sqrt{r^2 + \delta^2}$ e isto implica que

$$\delta' \leq \frac{\delta r}{\sqrt{r^2 + \delta^2}}.$$

Para a segunda desigualdade utilizamos o fato de que para um número positivo, $x \leq 1$,

$$\frac{1}{1+x} \leq 1 - \frac{x}{2}.$$

Então, como $\delta \leq r$, para $x = \frac{\delta^2}{r^2}$ temos a segunda desigualdade

$$\frac{\delta r}{\sqrt{r^2+\delta^2}} = \delta \sqrt{\frac{1}{\frac{r^2+\delta^2}{r^2}}} \leq \delta \sqrt{1 - \frac{\delta^2}{2r^2}}.$$

A última desigualdade segue do fato que quando $x \neq 0$, $1 + x < \exp x$, sendo $x = \dfrac{-\delta^2}{2r^2}$:

$$\delta \sqrt{1 - \frac{\delta^2}{2r^2}} \leq \delta \exp\left(\frac{-\delta^2}{2r^2}\right)^{\frac{1}{2}} = \delta \exp\left(\frac{-\delta^2}{4r^2}\right).$$

□

Deste resultado segue que quando $p \in \text{conv}(S)$, usando pivôs estritos, conseguimos uma constante melhor na equação (3.6). Ainda assim, temos que o Algoritmo Geométrico demanda $O(1/\varepsilon^2)$ iterações para encontrar uma ε-solução.

3.4 Uma análise alternativa

Fica claro de (3.9), e também da lei dos senos (veja Figura 3.5), que

$$\frac{\delta'}{\delta} = \sin\theta, \tag{3.12}$$

e portanto a redução no gap $d(p', p)$, a cada iteração do Algoritmo Geométrico, depende do ângulo $\theta = \angle pp'v$, em que $v \in S$ é um pivô (simples ou estrito) para p'. Assim, se existem constantes $\nu \in [0, 1)$ e $c > 0$ tais que

$$\sin\theta \leq \nu = \frac{1}{\sqrt{1+c}}, \quad \forall p' \in \text{conv}(S) \setminus B(p, \varepsilon R), \tag{3.13}$$

podemos obter um resultado de complexidade alternativo. Em [1], a constante ν é chamada de *constante de visibilidade*, e a constante c de *fator de visibilidade*.

Note que ν depende diretamente dos pivôs disponíveis em cada $p' \in \text{conv}(S) \setminus B(p, \varepsilon R)$.

Exemplo 3.10. Considere o caso em que S são os vértices de um quadrado e p é o ponto médio de uma das arestas, como na Figura 3.6. No quadrado da esquerda $\theta = \frac{\pi}{4}$, $\sin\theta = \frac{\sqrt{2}}{2}$ e no outro, $\theta = \frac{\pi}{3}$,

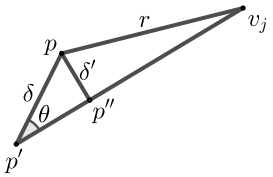

Figura 3.5 – Iteração típica do Algoritmo Geométrico para pivôs estritos.

$\sin\theta = \frac{\sqrt{3}}{2}$. Note que, a medida que p' fica mais próximo de p, o $\sin\theta$ se aproxima de 1. Assim, neste exemplo, a constante de visibilidade será muito próxima de 1 e o fator de visibilidade c próximo de zero. Note que $\sin\theta$ só não atinge seu supremo por conta da bola de centro p e raio εR.

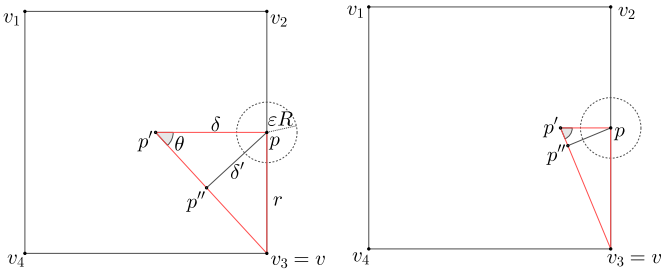

Figura 3.6 – Exemplo quando $\mathrm{conv}(S)$ é um quadrado e p está na borda.

Teorema 3.11. *Suponha $p \in \mathrm{conv}(S)$. Seja $\delta_0 = d(p_0, p)$, $p_0 \in \mathrm{conv}(S)$ e assuma que existem $\nu \in [0, 1)$ e $c > 0$ tais que (3.13) é satisfeita. Então, o número de iterações do Algoritmo Geométrico para obter $p_\varepsilon \in \mathrm{conv}(S)$ tal que $d(p_\varepsilon, p) \leq \varepsilon R$, $R = \max\{d(v_j, p) : v_j \in S\}$ é*

$$O\left(\frac{1}{c}\ln\frac{\delta_0}{\varepsilon R}\right).$$

Demonstração. Denotando os iterados do Algoritmo Geométrico por p_i, e $\delta_i = d(p_i, p)$, em vista de (3.12) e assumindo (3.13), após k iterações teremos

$$\delta_k \leq \nu^k \delta_0. \tag{3.14}$$

Para que $\delta_k < \varepsilon R$, é suficiente que o lado direto de (3.14) seja menor que ou igual a εR, logo

$$k \ln \nu = k \ln \frac{1}{\sqrt{1+c}} \leq \ln \frac{\varepsilon R}{\delta_0}.$$

Equivalentemente

$$\frac{k}{2} \ln (1+c) \geq \ln \frac{\delta_0}{\varepsilon R}.$$

Para $u \in (0,1)$, temos que $\ln(1+u) \geq \frac{u}{2}$. Assim é suficiente escolher k satisfazendo

$$k \geq 4 \frac{1}{c} \ln \frac{\delta_0}{\varepsilon R}.$$

□

Observação 3.12. Assim como no Teorema 3.4 se $p \notin \text{conv}(S)$, considerando $\varepsilon = \frac{\Delta}{R}$, onde $\Delta = \min\{d(x,p) : x \in \text{conv}(S)\}$ e $R = \max\{d(v_j,p) : v_j \in S\}$, temos que o número de iterações do Algoritmo Geométrico para obter uma p-testemunha é

$$O\left(\frac{1}{c} \ln \frac{\delta_0}{\Delta}\right).$$

Exemplo 3.13. Considere que o conv(S) é um quadrado de lado 1, de vértices v_1, v_2, v_3 e v_4. Sejam $v_5 = (1/2, 1/2)$, $S = \{v_1, v_2, v_3, v_4, v_5\}$ e $p = (1, 1/2)$. Na Figura 3.7 ilustramos as 100 primeiras iterações do Algoritmo Geométrico, começando de $p_0 = v_5$. Como observado no Exemplo 3.10, $\sin\theta$ se aproxima de 1 conforme p' se aproxima de p. Como ilustrado na figura, fica claro que o ângulo $\angle pp'v$ torna-se cada vez mais próximo de $\pi/2$ e com isso observamos o fenômeno de "zigzag". Neste caso, como o fator de visibilidade c é muito próxima de 0, pelo Teorema 3.11 um número elevado de iterações pode ser necessário para que o algoritmo encontre uma $\varepsilon-$solução, para $\varepsilon = 10^{-4}$.

Encerramos este capítulo com um resultado que mostra que se p está "suficientemente dentro" de conv(S), então o fator de visibilidade c fica afastado de zero. Mais especificamente, supondo que p pertence ao interior relativo de conv(S), e que $\rho > 0$ é o supremo dos raios das bolas centradas em p contidas no interior relativo, mostraremos que $c \geq \rho^2/R^2$.

Seja p' o iterado atual do Algoritmo Geométrico, tal que $d(p',p) \geq \varepsilon R$. Considere q o ponto na extensão do segmento $p'p$, tal que $d(p,q) = \rho$ (veja Figura 3.8).

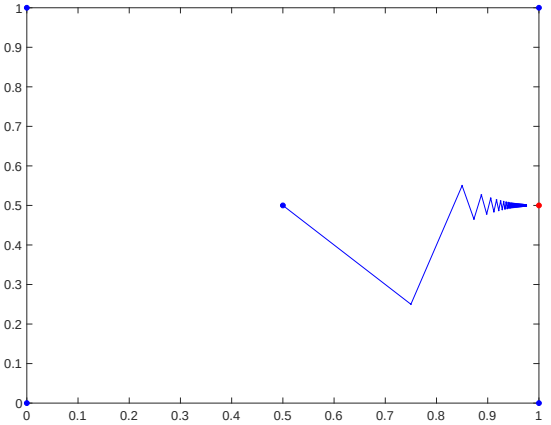

Figura 3.7 – Ilustração do pior caso para o Algoritmo Geométrico, quando $1/c$ é muito grande.

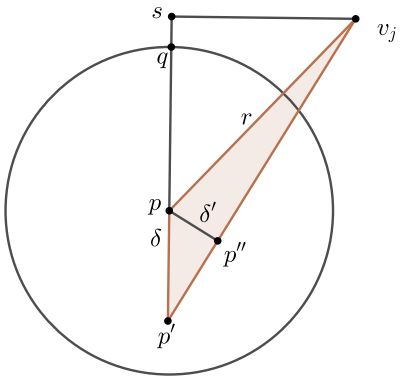

Figura 3.8 – Existência de um pivô estrito com constante de visibilidade limitada por $1/\sqrt{1+\rho^2/R^2}$.

Como $B(p,\rho) \subset \text{conv}(S)$, temos que $q \in \text{conv}(S)$. Logo pelo Teorema 3.8, para todo $q' \in \text{conv}(S)$ existe um q-pivô estrito para q'. Em particular, para $q' = p$, temos que o semi-espaço definido pelo hiperplano ortogonal ao segmento pq passando por q e que exclui p deve conter um q-pivô estrito v, para o qual o ângulo $\angle pqv$ é não-agudo (veja Figura 3.8). Note que v também é p-pivô estrito para p' e neste caso chamaremos este v de pivô estrito *forte*.

Por um lado, note que

$$\angle pp'v \leq \angle qpv. \tag{3.15}$$

Por outro lado, uma vez que $\angle pqv$ é não-agudo, podemos adicionar um ponto s tal que $\angle psv = \pi/2$, conforme Figura 3.8. Assim,

$$\sin\angle qpv = \sin\angle spv = \frac{\sqrt{r^2 - (d(s,q)+\rho)^2}}{r} \leq \frac{\sqrt{r^2-\rho^2}}{r} = \sqrt{1-\frac{\rho^2}{r^2}}. \tag{3.16}$$

De (3.15), (3.16), e algumas manipulações algébricas, temos que

$$\sin\theta = \sin\angle pp'v \leq \frac{1}{\sqrt{1+\rho^2/R^2}}.$$

Então, após k iterações do algoritmo, obtemos

$$\delta_k \leq \left(\frac{1}{\sqrt{1+\rho^2/R^2}}\right)^k \delta_0.$$

Para que $\delta_k < \varepsilon R$, é suficiente que k satisfaça

$$-\frac{k}{2}\ln\left(1+\frac{\rho^2}{R^2}\right) < \ln\frac{\varepsilon R}{\delta_0},$$

ou equivalentemente

$$k > \frac{4R^2}{\rho^2}\ln\frac{\delta_0}{\varepsilon R}.$$

Com isso, acabamos por provar o seguinte teorema.

Teorema 3.14. *Suponha que p esteja no interior relativo de* $\text{conv}(S)$. *Seja $\rho > 0$ o supremo dos raios das bolas centradas em p neste interior relativo. Sejam $\delta_0 = d(p_0,p)$, $p_0 \in \text{conv}(S)$, $R = \max\{d(v_i,p) : i = 1,\ldots,n\}$. Dado $\varepsilon \in (0,1)$, se o Algoritmo Geométrico usa um pivô estrito forte em cada iteração, então o número de iterações para obter $p' \in \text{conv}(S)$ tal que $d(p',p) < \varepsilon R$ é*

$$O\left(\frac{R^2}{\rho^2}\ln\frac{\delta_0}{\varepsilon R}\right).$$

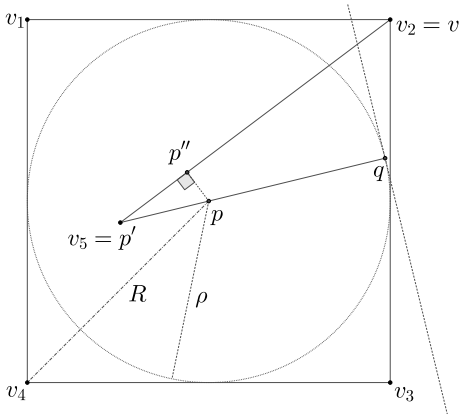

Figura 3.9 – Exemplo quando conv(S) é um quadrado e $B(p, \rho) \subset$ conv(S)

Observação 3.15. A demonstração do teorema acima é praticamente a mesma da do artigo [1]. No entanto, em [1, Teorema 14, p. 330] o enunciado requer apenas o uso de um pivô estrito em cada iteração. Como vimos na demonstração acima isto não é suficiente pois precisamos de um pivô estrito forte – e perceba que nem todo p-pivô estrito para p' é q-pivô estrito para p (pivô estrito forte).

Exemplo 3.16. Considere conv(S) como o quadrado de lado 1, p no centro desse quadrado e $S = \{v_1, v_2, v_3, v_4, v_5\}$ como na Figura 3.9 . Na iteração ilustrada, $v_5 = p'$ e v_2 é escolhido como pivô estrito forte, visto que está no semi-espaço definido pelo hiperplano ortogonal ao segmento $p'q$ passando por q e que exclui p.

Neste exemplo, as hipóteses do Teorema 3.14 são satisfeitas, com $R = \dfrac{1}{\sqrt{2}}$, $\rho = \dfrac{1}{2}$, $\delta_0 = \dfrac{1}{4}$, e então o número de iterações será da ordem de

$$O\left(2\ln\left(\frac{\sqrt{2}}{4\varepsilon}\right)\right).$$

Logo, para $\varepsilon = 10^{-2}$ encontraríamos a $\varepsilon-$solução em até 8 iterações, escolhendo $\varepsilon = 10^{-3}$ em até 12 e assim por diante.

Para este exemplo, além do caso em que p está no centro do quadrado, esperamos que a complexidade do Algoritmo Geométrico seja boa também para pontos no conv(S) contidos em uma bola de raio

suficientemente grande no interior relativo de $\text{conv}(S)$. Apenas para pontos próximos a borda a complexidade do Algoritmo Geométrico é prejudicada, como visto no Exemplo 3.13.

Observação 3.17. Uma maneira de garantir a escolha de um pivô estrito forte, por exemplo, é utilizar a estratégia do Algoritmo Geométrico Ganancioso (veja Seções 2.3 e 4.3.2) de buscar v_j tal que o produto interno $v_j^T(p' - p)$ seja mínimo.

Resumindo todos os resultados de complexidade deste capítulo, temos que, para $p \in \text{conv}(S)$, o número máximo de iterações para obter uma $\varepsilon-$solução é

$$O\left(\min\left\{\frac{1}{\varepsilon^2}, \frac{1}{c}\ln\frac{\delta_0}{\varepsilon R}\right\}\right).$$

Além disso, se $p \notin \text{conv}(S)$ o número máximo de iterações para o Algoritmo Geométrico obter uma p-testemunha $p' \in \text{conv}(S)$ é

$$O\left(\min\left\{\frac{R^2}{\Delta^2}, \frac{1}{c}\ln\frac{\delta_0}{\Delta}\right\}\right).$$

4 FORMULAÇÕES ALTERNATIVAS E EXPERIMENTOS NUMÉRICOS

A fim de verificar a performance prática do Algoritmo Geométrico, neste capítulo apresentamos formulações alternativas para o PIEC, modelando-o como problemas de programação linear e quadrática. Com isso, podemos comparar o desempenho de métodos clássicos para as respectivas formulações com o Algoritmo Geométrico.

Além disso, tendo em vista a análise de complexidade do Capítulo 3, também propomos variantes do Algoritmo Geométrico que podem apresentar um desempenho prático melhor que o original.

4.1 Formulações lineares

O problema de inclusão no envoltório convexo é um caso especial de problema de factibilidade em Programação Linear (PL). Decidir se $p \in \text{conv}(S)$ é equivalente a testar a factibilidade de $Sx = p, x \geq 0, e^T x = 1$, em que (abusando da notação) S é uma matriz na qual as colunas são os vetores $\{v_1, \ldots, v_n\}$, $e^T = (1, 1, \ldots, 1) \in \mathbb{R}^n$ e $p \in \mathbb{R}^m$ é o ponto que queremos consultar.

Em [13] foi mostrado que o PIEC pode ser formulado como o seguinte PL:

$$\begin{aligned} \min_{x} \quad & \sum_{i=1}^{m+1} z_i \\ \text{s.a.} \quad & Sx + z = p \\ & e^T x + z_{m+1} = 1 \\ & x \geq 0, z \geq 0, z_{m+1} \geq 0, \end{aligned} \tag{4.1}$$

em que $z \in \mathbb{R}^m$ e z_{m+1} são variáveis de folga para cada uma das $m+1$ restrições.

Com efeito, se assumirmos que p possui todas as componentes não-negativas[1] e definindo $z_i = p_i, i = 1, \ldots, m$, $z_{m+1} = 1$ e $x = 0$, temos uma solução básica factível para (4.1). Note que, se o valor ótimo de (4.1) for zero para uma solução ótima (x, z, z_{m+1}), temos que todas as variáveis de folga são nulas e então $p = Sx, x \geq 0$ e $e^T x = 1$, ou seja, p é uma combinação convexa dos elementos de S e portanto $p \in \text{conv}(S)$.

[1] Se houvesse alguma componente negativa, bastaria multiplicar a equação correspondente por (-1).

No entanto, se o valor ótimo da função objetivo for maior que zero, isso implica que algum $z_i \neq 0$, $i = 1, \ldots, m$, e assim $p \notin \text{conv}(S)$.

Neste trabalho, propomos uma outra formulação linear para o PIEC, usando como base o Teorema 1.21. Considere o seguinte problema de otimização:

$$\begin{aligned} \min \quad & -w_{m+1} \\ \text{s.a.} \quad & (x-p)^T w + w_{m+1} \leq 0, \quad \forall x \in \text{conv}(S), \\ & \|w\|_\infty \leq 1, w_{m+1} \geq 0, \end{aligned} \tag{4.2}$$

em que assumimos que $p \neq 0$ é o ponto a ser consultado.

Observe que se existe um hiperplano separador $w^T x = \beta$, com $w \neq 0$, entre p e $\text{conv}(S)$, tal que $w^T x \leq \beta$, $\forall x \in \text{conv}(S)$ e $w^T p > \beta$, então necessariamente $(x-p)^T w \leq 0$, o que leva a formulação (4.2).

Se na solução ótima de (4.2) tivermos $w_{m+1} = 0$, então não existe tal hiperplano e concluímos que $p \in \text{conv}(S)$. Note que a formulação (4.2) apresenta infinitas restrições. Para contornar esta dificuldade, consideramos o seguinte resultado.

Proposição 4.1. *Seja $w \in \mathbb{R}^m \setminus \{0\}$ e considere um conjunto finito de pontos $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$. Então $x^T w \geq 0, \forall x \in \text{conv}(S)$ se, e somente se, $v_i^T w \geq 0, \forall v_i \in S$.*

Demonstração. Considere que $x^T w \geq 0, \quad \forall x \in \text{conv}(S)$. Como $S \subset \text{conv}(S)$, em particular, temos que $v_i^T w \geq 0, \forall v_i \in S$.

Por outro lado, se $v_i^T w \geq 0, \forall v_i \in S$, então para qualquer $\alpha \in \mathbb{R}^n$ tal que $\alpha \geq 0$ e $e^T \alpha = 1$ temos

$$x^T w = \left(\sum_{i=1}^{n} \alpha_i v_i \right)^T w = \sum_{i=1}^{n} \alpha_i v_i^T w \geq 0,$$

e portanto $x^T w \geq 0$ para todo $x \in \text{conv}(S)$. □

Logo, da Proposição 4.1, podemos escrever (4.2) de forma equivalente como

$$\begin{aligned} \min \quad & -w_{m+1} \\ \text{s.a.} \quad & (v_j - p)^T w + w_{m+1} \leq 0, \quad j = 1, 2, \ldots, n, \\ & -1 \leq w_i \leq 1, \quad i = 1, 2, \ldots, m, \\ & w_{m+1} \geq 0. \end{aligned} \tag{4.3}$$

É possível mostrar que (4.3) está associado ao dual de (4.1).

Com a finalidade de comparar com os algoritmos propostos neste trabalho, nos experimentos numéricos consideramos o método Simplex [2–4] para resolução das formulações lineares (4.1) e (4.3).

4.2 Formulação quadrática

Como já discutimos na Seção 2.3, o PIEC pode ser tratado através do problema de projeção (2.13):

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \frac{1}{2}\|Sx-p\|^2 \\ \text{s.a} \quad & x \in \Delta_n, \end{aligned} \tag{4.4}$$

lembrando que $\Delta_n = \{x \in \mathbb{R}^n \ : \ e^T x = 1, x \geq 0\}$ denota o simplex unitário. O vetor $x \in \Delta_n$ corresponde aos coeficientes de uma combinação convexa.

Seja x_* a solução de (4.4). Se o valor ótimo for estritamente positivo, i.e., $\|Sx_* - p\| > 0$, então $p \notin \text{conv}(S)$ e Sx_* corresponde a projeção de p em $\text{conv}(S)$. Se o valor ótimo for zero então temos que $p \in \text{conv}(S)$.

Para resolver a formulação quadrática (4.4), vamos considerar os clássicos métodos do Gradiente Projetado (Algoritmo 2) e Gradiente Condicional (Algoritmo 3) descritos nas Seções 1.4.1 e 1.4.2, respectivamente.

4.2.1 Projeção no simplex unitário

No método do Gradiente Projetado, aplicado a (4.4), o cálculo da direção factível e de descida envolve a projeção de um vetor de $\mathbb{R}^n$ em Δ_n. Para obter a projeção no simplex unitário usaremos a abordagem descrita em [14, 15], que calcula a projeção sobre Δ_n em um número finito de iterações.

Sejam $I_n = \{1, \ldots, n\}$, $I \subset I_n$, $X_I = \{x \in \mathbb{R}^n : x_i = 0, \forall i \in I\}$, $n_I = \dim(X_I) = n - |I|$, $K_I = X_I \cap \Delta_n$ e $V_I = X_I \cap V$, em que $V = \{x \in \mathbb{R}^n : e^T x = 1\}$.

Algoritmo 5: Projeção no Simplex Unitário

Entrada: Dado $c \in \mathbb{R}^n$, faça $x = c$ e $I = \emptyset$.

1 Compute $\tilde{x} = P_{V_I}(x)$. Se $\tilde{x} \geq 0$, pare $(\tilde{x} = P_{\Delta_n}(c))$.

2 Atualize $I \leftarrow I \cup \{i \ : \ \tilde{x}_i < 0\}$ e substitua x por $P_{X_I}(\tilde{x})$. Volte ao Passo 1.

O Algoritmo 5 nada mais é que um algoritmo de projeções alternadas [16, 17]. Note que a projeção de x em V tem expressão analítica dada por:

$$P_V(x) = x + \left(\frac{1 - e^T x}{n}\right) e,$$

assim como a projeção de x em $\mathbb{R}^n_+$ que é $\max\{x, 0\}$. A dificuldade está exatamente em projetar na intersecção $V \cap \mathbb{R}^n_+$.

A observação chave que permite encontrar a projeção de x em Δ_n em um número finito de passos é que se $\tilde{x} = P_V(x)$ possuir componentes negativas, tais componentes serão nulas na solução $P_{\Delta_n}(x)$. Logo, tais componentes podem ser zeradas e passamos a trabalhar em um subespaço de dimensão menor. A convergência ocorre em no máximo n iterações porque a dimensão n_I do subespaço X_I decresce pelo menos uma unidade por iteração. Para mais detalhes, consulte [14].

As projeções requeridas pelo Algoritmo 5 admitem fórmulas fechadas. Para calcular $\tilde{x} = P_{V_I}(x)$:

$$\begin{aligned} \tilde{x}_i &= x_i + \frac{1 - \sum_j x_j}{n_I}, \quad \forall i \notin I, \\ \tilde{x}_i &= 0, \quad \forall i \in I, \end{aligned}$$

e $P_{X_I}(\tilde{x}) = \max\{\tilde{x}, 0\}$.

No pior caso o Algoritmo 5 gasta $(n+1)n/2$ operações para calcular a projeção em Δ_n.

4.2.2 Critérios de parada

O Algoritmo Geométrico para quando

$$d(p', p) = \|p' - p\| = \|Sx' - p\| \leq \varepsilon, \tag{4.5}$$

ou quando detecta uma testemunha p' tal que o hiperplano (2.7) separa p de $\text{conv}(S)$. Assim, um critério de parada natural para o Gradiente Projetado e Frank-Wolfe é

$$\|Sx_k - p\| \leq \varepsilon. \tag{4.6}$$

Outro critério de parada, relevante para quando $p \notin \text{conv}(S)$, é discutido a seguir.

Na Seção 1.4.2, vimos que o critério de parada para Frank-Wolfe, aplicado ao problema (1.2), dado por (1.11) implica em

$$f(x_k) - f(x_*) \leq \nabla f(x_k)^T (x_k - \bar{x}_k) \leq \varepsilon_f,$$

em que x_k é o iterado atual, $\bar{x}_k$ é a solução do subproblema (1.10) e ε_f é a tolerância escolhida. Vejamos como devemos escolher ε_f para garantir que $\|Sx_k - p\|$ já está perto o suficiente do valor ótimo $\|Sx_* - p\|$.

Note que a função objetivo em questão para a formulação (4.4) é $f(x) = \frac{1}{2}\|Sx - p\|^2$. Assim, as iterações de Frank-Wolfe param com:

$$\|Sx_k - p\|^2 - \|Sx_* - p\|^2 \leq 2\varepsilon_f,$$

que implica em

$$\|Sx_k - p\| - \|Sx_* - p\| \leq 2\frac{\varepsilon_f}{\|Sx_k - p\|}.$$

Logo, impondo $\varepsilon_f = \|Sx_k - p\|\varepsilon/2$, teremos

$$\|Sx_k - p\| \leq \|Sx_* - p\| + \varepsilon.$$

Se $p \in \text{conv}(S)$, x_* solução de (4.4) é tal que $\|Sx_* - p\| = 0$ e então a desigualdade acima coincide com (4.5). Por outro lado, se $\|Sx_k - p\| > \varepsilon$, então da desigualdade acima $\|Sx_* - p\| > 0$ e podemos concluir que $p \notin \text{conv}(S)$. Portanto, além da condição (4.6), também usamos como critério de parada para Frank-Wolfe (aplicado ao problema (4.4)):

$$\nabla f(x_k)^T(x_k - \bar{x}_k) \leq \frac{\|Sx_k - p\|}{2}\varepsilon. \tag{4.7}$$

4.3 Variantes do Algoritmo Geométrico

Com base na análise apresentada no Capítulo 3, em particular na Seção 3.4, vemos que o desempenho do Algoritmo Geométrico é ditado pela constante de visibilidade ν. Como na prática ν depende não apenas dos pivôs disponíveis para cada $p' \in \text{conv}(S) \setminus B(p, \varepsilon R)$ mas também da escolha de tais pivôs, nesta seção propomos estratégias que procuram melhorar a constante de visibilidade.

4.3.1 Algoritmo Geométrico com condição do ângulo

Vimos que os algoritmos de busca direcional (Seção 1.4) utilizam a condição do ângulo (1.6) para evitar que as direções de busca se tornem ortogonais ao gradiente negativo.

Por outro lado, também vimos que no Algoritmo Geométrico, se v_j é pivô para p', então $d = v_j - p'$ é uma direção de descida (veja Lema 2.4(c)) para $f(y) = \frac{1}{2}\|y - p\|^2$, e que $\sin\theta = \sin\angle pp'v_j$ determina a redução na distância em relação a p segundo (3.12).

Então, ao invés de simplesmente escolher um pivô, propomos, a cada iteração do Algoritmo Geométrico, tomar $v_j \in S$ tal que

$$(p' - p)^T (v_j - p') \leq -\kappa \|p' - p\| \|v_j - p'\|, \tag{4.8}$$

para $\kappa \in (0, 1]$.

Algoritmo 6: Algoritmo Geométrico com condição do ângulo

Dados: $S = \{v_1, ..., v_n\}, p \in \mathbb{R}^m, \varepsilon \in (0, 1), \kappa \in (0, 1]$

1 Escolha $p' \in \arg\min\{d(v_j, p) \ : \ v_j \in S\}$.

2 Determine

$S_\kappa = \{v_j \in S \setminus \{p'\} : (p' - p)^T (v_j - p') \leq -\kappa \, \|p' - p\| \, \|v_j - p'\|\}$.

3 Se $S_\kappa \neq \emptyset$, escolha $v_j \in S_\kappa$ e vá para o passo 6.

4 Se $\forall v_j \neq p'$, $d(v_j, p) > d(v_j, p')$, pare.

5 Escolha $v_j \neq p'$, tal que $d(v_j, p) \leq d(v_j, p')$.

6 Calcule $\bar{\alpha} = \arg\min\{d(p, (1 - \alpha)p' + \alpha v_j) : 0 \leq \alpha \leq 1\}$,

defina $p'' = (1 - \bar{\alpha})p' + \bar{\alpha} v_j$ e atualize $p' \leftarrow p''$.

7 Se $d(p', p) < \varepsilon$, pare.

8 Retorne ao passo 2

No Algoritmo 6, buscamos na lista S algum v_j que satisfaça a condição (4.8). Dependendo de p' e do valor de κ pode ser que não exista v_j em S satisfazendo este critério, e então simplesmente tomamos um pivô simples.

É claro que para concluir que $S_\kappa = \emptyset$ temos que percorrer toda a lista S e, neste processo, já podemos aproveitar para verificar quais v_j são pivôs.

É importante notar que a condição (4.8) só implica que v_j é pivô simples (ou estrito) para valores suficientemente grandes de $\kappa = \cos\theta$, como ilustra a Figura 4.1. Por outro lado, dependendo de κ, nem todo pivô cumpre a condição do ângulo.

4.3.2 Algoritmo Geométrico Ganancioso

Na Seção 2.3 vimos a relação entre o Algoritmo Geométrico e o Gradiente Condicional(Frank-Wolfe): o método de Frank-Wolfe, aplicado ao problema (4.4), equivale ao Algoritmo Geométrico com a escolha de um pivô $v_j \neq p'$ tal que

$$v_j^T (p' - p) = \min\{v_i^T (p' - p) : v_i \in S \setminus \{p'\}\}. \tag{4.9}$$

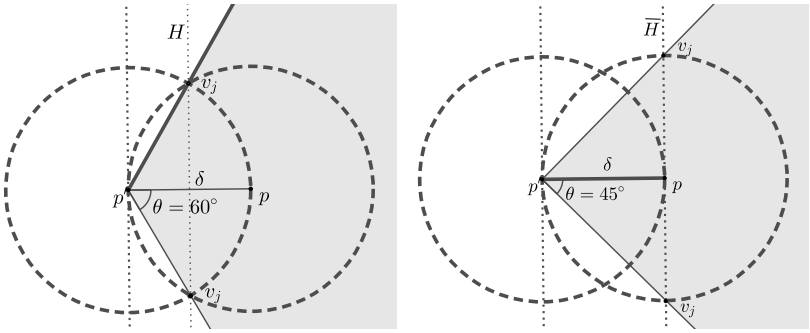

Figura 4.1 – Para $\kappa \geq 1/2$ temos que a condição do ângulo implica pivô simples e para $\kappa \geq \sqrt{2}/2$ a condição implica pivô estrito.

Com base na caracterização do item (b) do Lema 2.4, temos que o método de Frank-Wolfe (com as particularidades do PIEC) equivale a um Algoritmo Geométrico Ganancioso, em que ao invés de escolher v_j tal que

$$v_j^T(p' - p) \leq \frac{\|p'\|^2 - \|p\|^2}{2}, \tag{4.10}$$

escolhemos v_j tal que o lado esquerdo de (4.10) é mínimo.

Algoritmo 7: Algoritmo Geométrico Ganancioso

Dados: $S = \{v_1, ..., v_n\}, p \in \mathbb{R}^m, \varepsilon \in (0, 1)$

1 Escolha $v_j \in S \setminus \{p'\}$ tal que $v_j^T(p' - p) = \min\{v_i^T(p' - p) : v_i \in S \setminus \{p'\}\}$.

2 Se $v_j^T(p' - p) > (\|p'\|^2 - \|p\|^2)/2$, pare.

3 Calcule $\bar{\alpha} = \arg\min\{d(p, (1 - \alpha)p' + \alpha v_j) : 0 \leq \alpha \leq 1\}$, defina $p'' = (1 - \bar{\alpha})p' + \bar{\alpha}v_j$ e atualize $p' \leftarrow p''$.

4 Se $d(p', p) < \varepsilon$, pare.

5 Retorne ao passo 1.

Note ainda, do Lema 3.7(b), que se $v_j^T(p' - p) \leq p^T(p' - p)$, então v_j é pivô estrito. Assim, em vista de (4.9), o Algoritmo 7 escolherá um pivô estrito sempre que possível.

Além disso, como já comentado nas observações após o Teorema 3.14, se existem pivôs estritos fortes para p', o Algoritmo 7 selecionará um deles.

A desvatagem deste algoritmo (em relação ao Algoritmo Geométrico original) é que a cada iteração é necessário percorrer toda a lista S para assegurar o mínimo. Logo, o custo por iteração é sempre $O(nm)$.

4.4 Experimentos Numéricos

Para avaliar o desempenho do Algoritmo Geométrico e suas variantes frente a algoritmos clássicos de otimização para as formulações lineares e quadrática, nesta seção apresentamos alguns resultados computacionais.

As instâncias do PIEC foram geradas de forma artificial seguindo o mesmo esquema de [13]. Os pontos do conjunto $S = \{v_1, \dots v_n\}$, são gerados segundo uma distribuição uniforme na bola unitária em $\mathbb{R}^m$ [18]. Cada v_j é gerado da seguinte maneira: primeiro geramos um vetor $v \in \mathbb{R}^m$ com componentes amostradas independentemente de uma distribuição normal padrão; a seguir obtemos v_j pela expressão

$$v_j = \sqrt[m]{u}(v/\|v\|),$$

na qual u vem de uma distribuição uniforme no intervalo $[0, 1]$.

O ponto p foi gerado de quatro maneiras distintas para analisar diferentes cenários, como discutiremos a seguir.

Os algoritmos foram implementados e os experimentos realizados em ambiente Matlab R2018b em um computador com processador Intel Core i5 1.8Ghz, com 8GB de RAM.

Algumas particularidades dos experimentos:

- Para todos os problemas dos cenários descritos abaixo, definimos

  $$maxit = \min\{\max\{1000n, 10000\}, 10^6\},$$

  como o número máximo de iterações para os algoritmos geométricos, FW e GP.

- Para as formulações lineares empregamos o comando `linprog` do Matlab, usando como opção o método "dual-simplex[2]" [19].

- Ressaltamos que as formulações lineares foram resolvidas de maneira exata pelo Dual-Simplex, enquanto os outros algoritmos testados utilizam (direta ou indiretamente) a tolerância ε, encontrando soluções aproximadas. Embora o Dual-Simplex seja capaz

[2] Optamos por utilizar o Dual-Simplex, pois em testes preliminares este algoritmo foi mais eficiente (em relação ao tempo) do que o de Pontos Interiores.

de resolver instâncias com $n > 10000$, decidimos não considerar as formulações lineares para os valores mais elevados de n por apresentarem um tempo computacional muito superior aos demais. Por exemplo, o tempo médio em instâncias com $n = 50000$ pontos foi de aproximadamente 500 segundos.

- Para todos os problemas fixamos o tempo limite de execução de cada algoritmo em 20 segundos, pois, com exceção das formulações lineares (que em determinados casos para encontrar a solução exata precisavam de muito mais tempo), esse tempo, na maioria das vezes, foi suficiente para os demais algoritmos.

- Para a formulação quadrática implementamos o método de Frank-Wolfe (Seção 1.4.2) adicionando o critério de parada (4.7) e o método de Gradiente Projetado (Seção 1.4.1). Em ambos, utilizamos busca linear exata.

- No algoritmo de Gradiente Projetado, utilizamos o Algoritmo 5 para calcular a projeção no simplex unitário e consideramos o critério de parada usual $\|d_k\| < \varepsilon_g = 10^{-6}$.

- Para o Algoritmo 6, utilizamos $\kappa = 0.14$.

- A tolerância $\varepsilon > 0$ no critério de parada do Algoritmo Geométrico e variantes é discutida em detalhes a seguir, na descrição de cada cenário.

Como as variáveis $x \in \mathbb{R}^n$ na formulação quadrática (4.4) são os coeficientes da combinação convexa (das colunas de S), e não o ponto $p' \in \mathbb{R}^m$ como no Algoritmo Geométrico, consideramos como ponto inicial $x_0 = e_i \in \mathbb{R}^m$, em que o índice i é tal que

$$i = \arg\min_{1 \leq j \leq n} \{\|v_j - p\|\}, \tag{4.11}$$

pois assim $Sx_0 = p_0$, em que p_0 é o ponto inicial para o Algoritmo 4.

Os algoritmos considerados nestes experimentos estão descritos na Tabela 4.1. Ressaltamos que AG representa o Algoritmo Geométrico original que emprega pivô simples em cada iteração. Nas comparações, não consideramos o algoritmo que usa pivôs estritos pois o Algoritmo Geométrico Ganancioso já escolhe tais pivôs sempre que possível.

Outro detalhe importante é a diferença entre Frank-Wolfe (FW) e o Algoritmo Geométrico Ganancioso (AGG). No algoritmo FW avaliamos o gradiente sem tirar proveito da estrutura do problema e portanto são

Tabela 4.1 – Algoritmos testados.

Sigla	**Algoritmo**	**Referência**
AG	Algoritmo Geométrico (Alg. 4)	[1]
AGA	AG com condição do ângulo (Alg. 6)	Seção 4.3.1
AGG	AG Ganancioso (Alg. 7)	Seção 4.3.2
LP1	`linprog` para formulação (4.1)	[13]
LP2	`linprog` para formulação (4.3)	Seção 4.1
FW	Algoritmo de Frank-Wolfe (Alg. 3)	Seção 1.4.2
GP	Algoritmo Gradiente Projetado (Alg. 2)	Seção 1.4.1

Tabela 4.2 – Custo por Iteração (pior caso)

Sigla	**Custo por Iteração**
AG	$O(mn)$
AGA	$O(mn)$
AGG	$O(mn)$
LP1 (Dual-Simplex)	$O(m^2 + mn)$
LP2 (Dual-Simplex)	$O(n^2 + mn)$
FW	$O(2mn)$
GP	$O\left(2mn + \frac{(n+1)n}{2}\right)$

realizados dois produtos matriz-vetor. Já no AGG, tendo em vista a discussão da Seção 2.3, tais produtos são evitados interpretando Se_i como v_i e Sx_k como p_k.

Na Tabela 4.2, relembramos o custo por iteração (no pior caso) para cada um dos algoritmos considerados.

Os experimentos foram organizados em quatro cenários:

(a) $p \in \text{conv}(S)$ longe da fronteira;

(b) $p \notin \text{conv}(S)$ longe da fronteira;

(c) $p \in \text{conv}(S)$ perto da fronteira;

(d) $p \notin \text{conv}(S)$ perto da fronteira.

Com isso, esperamos avaliar a performance do Algoritmo Geométrico (e variantes) nos melhores e piores cenários. Para todos eles consideramos a dimensão $m = 100$.

Para cada cenário e para cada valor de $n > m$ foram geradas 10 instâncias. Assim, nas Figuras 4.2 até 4.12, os tempos reportados são obtidos pela média aritmética e as legendas seguem a Tabela 4.1.

Cenário (a): $p \in \text{conv}(S)$ suficientemente dentro do interior relativo

Neste cenário consideramos $p = 0 \in \mathbb{R}^m$, o centro da bola unitária em $\mathbb{R}^m$. Como os elementos v_j de S vem de uma distribuição uniforme na bola unitária, com uma grande quantidade n de pontos, é quase certo que $p \in \text{conv}(S)$.

Além disso, é provável que $B(p, \rho) \subset \text{conv}(S)$, para $\rho > 0$ afastado de zero.

Consideramos inicialmente a tolerância $\varepsilon = 10^{-2}$ nos algoritmos geométricos e variamos $n = 500, 1000, \ldots, 5000$. Optamos por escolher a precisão de $\varepsilon = 10^{-2}$ neste cenário, pois em testes preliminares, com precisões 10^{-3} e 10^{-4}, o Algoritmo Geométrico com pivô simples (AG) excedeu o número máximo de iterações. Este fenômeno já era esperado do Teorema 3.4 que estima um número máximo de $\lceil 48\varepsilon^{-2} \rceil$ iterações.

Por outro lado, visto que p está no interior relativo do $\text{conv}(S)$, pela análise de complexidade alternativa do Teorema 3.14, esperamos que o Algoritmo Geométrico Ganancioso apresente um bom desempenho neste cenário.

A Figura 4.2 mostra os resultados para este cenário e todos os algoritmos considerados. Vemos que as formulações lineares (resolvidas via `linprog`) apresentaram um maior tempo de resolução comparados aos demais. Na Figura 4.3 apresentamos os mesmos resultados sem as formulações lineares, para uma melhor comparação dos demais algoritmos.

Vemos também que, para $n \leq 5000$, os algoritmos FW, GP, AGA e AGG apresentam um desempenho superior ao AG. Sendo o Algoritmo Geométrico Ganancioso o mais rápido nesse caso.

Já para instâncias com um maior número de pontos: $n = 10000, \ldots, 100000$, é possível observar pela Figura 4.4 que as variantes do Algoritmo Geométrico apresentaram boa performance em relação ao Gradiente Projetado e Frank-Wolfe quanto ao tempo de execução, sendo que os algoritmos AGG e AGA apresentaram o melhor desempenho.

Em parte, o desempenho do AGG é explicado pelo Teorema 3.14, uma vez que tal algoritmo usará um pivô estrito forte sempre que possível, apesar de precisar percorrer todos os n pontos de S. Assim, considerando o custo por iteração da Tabela 4.2 e a complexidade do

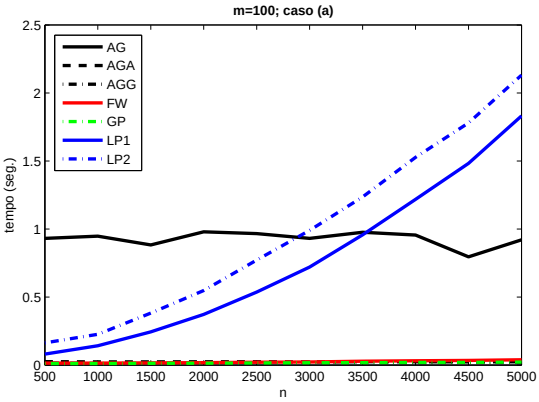

Figura 4.2 – Tempo médio em 10 instâncias com $m = 100$, $p = 0 \in \text{conv}(S)$, para $n = 500, 1000, \ldots, 5000$, $\varepsilon = 10^{-2}$.

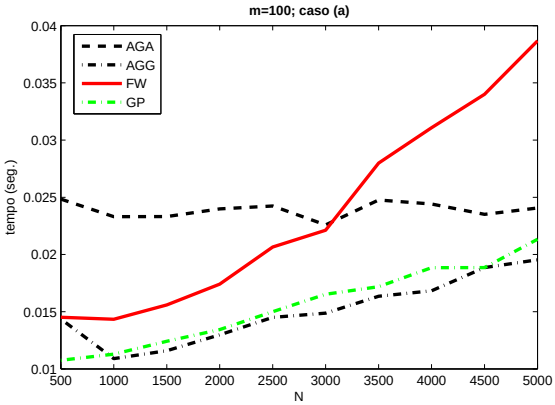

Figura 4.3 – Tempo médio em 10 instâncias com $m = 100$, $p = 0 \in \text{conv}(S)$, para $n = 500, 1000, \ldots, 5000$, $\varepsilon = 10^{-2}$.

Teorema 3.14, o custo total do AGG é

$$O\left(mn\frac{R^2}{\rho^2}\ln\frac{\delta_0}{\varepsilon R}\right).$$

Nos testes desse cenário o valor máximo de R foi 1 e $0.85 \leq \rho \leq \delta_0 \leq 0.92$ de modo que as iterações não passaram de 100.

Além disso, como m, ε estão fixos, esperamos que o tempos dos algoritmos geométricos cresçam linearmente com o aumento de n.

Sabemos que a diferença entre o Frank-Wolfe e o Algoritmo Geométrico Ganancioso é que o FW terá um custo adicional de dois produtos matriz-vetor por iteração. Isto explica seu tempo superior ao do AGG apesar do número de iterações de ambos se manter praticamente o mesmo e inferior a 40 iterações.

Notamos ainda que o GP teve um desempenho similar ao AGG, apesar de um custo maior por iteração, em decorrência do baixo número de iterações (no máximo 7).

Perceba também que nestes testes, o desempenho do AGA foi superior ao do AGG. Possivelmente, como a quantidade de pontos é muito grande, a cada iteração, o AGA conseguiu encontrar um pivô que satisfaz a condição do ângulo, sem precisar percorrer todos os pontos de S. Isto também explica o fato do AG manter sua média de tempo em torno de 1 segundo, mesmo para valores elevados de n.

Destacamos ainda que nos problemas deste cenário o fator de visibilidade observado se manteve afastado de zero.

Cenário (b): $p \notin \text{conv}(S)$ e afastado da fronteira

Neste cenário consideramos $p \notin \text{conv}(S)$, mais especificamente, consideramos um vetor p aleatório tal que $\|p\| = 2$. Como $S \subset B(0,1)$, temos certamente que $p \notin \text{conv}(S)$.

Para estes testes consideramos $\varepsilon = 10^{-4}$, como em [1]. No entanto, vimos no Teorema 3.4 que a complexidade de iteração não depende de ε e sim das constantes Δ e R. Como em problemas práticos não as conhecemos, optamos por manter $\varepsilon = 10^{-4}$. Lembramos que a tolerância ε influencia nos critérios de paradas do FW e GP e também na decisão de p pertencer ou não ao conv(S).

A Figura 4.5 mostra os resultados para este cenário, com $n = 500, \ldots, 5000$. Observamos que novamente os resultados das formulações lineares não foram satisfatórios comparados aos demais, devido ao fato de resolverem o problema de maneira exata, como observamos anteriormente.

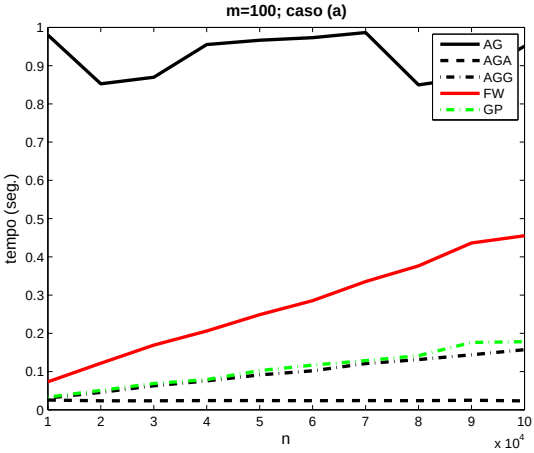

Figura 4.4 – Tempo médio em 10 instâncias com $m = 100$, $p = 0 \in \text{conv}(S)$, para $n = 10000, \ldots, 100000$, $\varepsilon = 10^{-2}$.

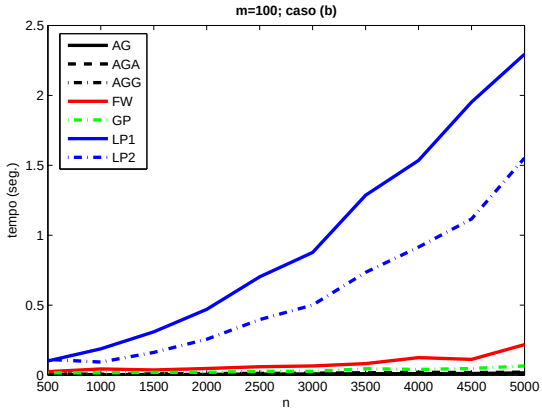

Figura 4.5 – Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 2$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$.

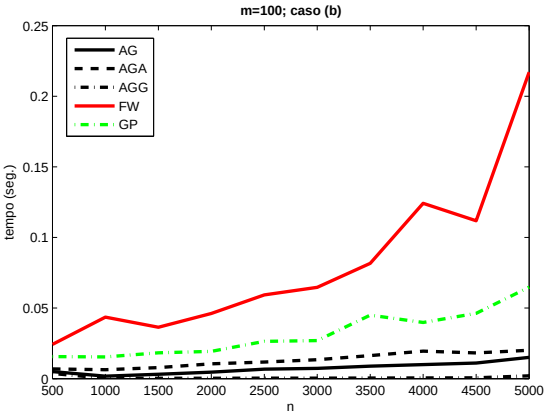

Figura 4.6 – Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 2$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$.

Novamente, para facilitar a comparação dos demais algoritmos, na Figura 4.7, omitimos LP1 e LP2, apresentando os experimentos para $10^4 \leq n \leq 10^5$.

Podemos observar que todos algoritmos tiveram um tempo médio inferior a 1 segundo.

Além disso, o Algoritmo Geométrico e suas variantes novamente apresentaram uma boa performance em relação ao GP e FW, que se acentua a medida que o número de pontos n aumenta (justamente pelo custo por iteração visto na Tabela 4.2).

Note que no caso em que $p \notin \text{conv}(S)$ o custo total do Algoritmo Geométrico e suas variantes é

$$O\left(mn \min\left\{\frac{R^2}{\Delta^2}, \frac{1}{c}\ln\frac{\delta_0}{\Delta}\right\}\right).$$

Para as instâncias geradas neste cenário temos que R não excede 3 e Δ, δ_0 são maiores ou iguais a 1. Como visto na Seção 3.4 o fator de visibilidade efetivo, c, depende diretamente do pivô escolhido. No AG precisamos de um pivô simples, enquanto no AGG buscamos um pivô, v, que minimiza $v^T(p' - p)$. Isto faz com que o fator do primeiro seja menor, explicando o fato do Algoritmo Geométrico ter uma performance inferior a do Algoritmo Geométrico Ganancioso, mesmo não precisando percorrer todos os n pontos de S a cada iteração.

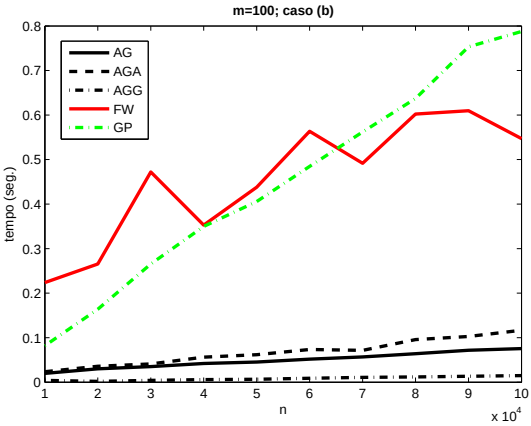

Figura 4.7 – Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 2$, para $n = 10000, \ldots, 100000$, $\varepsilon = 10^{-4}$.

Cenário (c): $p \in \text{conv}(S)$, mas próximo da fronteira

Nesse cenário buscamos gerar um ponto p dentro do $\text{conv}(S)$ mas próximo a fronteira para ilustrar o pior caso dos algoritmos geométricos.

Primeiro determinamos dois pontos de S, digamos v_ℓ e v_q, que possuem os maiores valores do funcional linear $\psi(v) = e^T v$. A seguir, fazemos

$$p = \frac{1}{2}v_\ell + \frac{1}{2}v_q.$$

Além disso, para evitar que v_ℓ e v_q sejam os pontos de S mais próximos de p, adicionamos a S um novo ponto v_s tal que $d(v_s, p) < d(v_\ell, v_q)/2$.

Como vimos no exemplo da Figura 3.6, neste caso, a constante de visibilidade ν pode ficar próxima de 1 (e a constante c próxima de zero). Assim, uma vez que $1/c$ pode ser relativamente grande, esperamos observar a complexidade do Teorema 3.4 ao invés da do Teorema 3.11 para o Algoritmo Geométrico e variantes (em contraste ao cenário (a)).

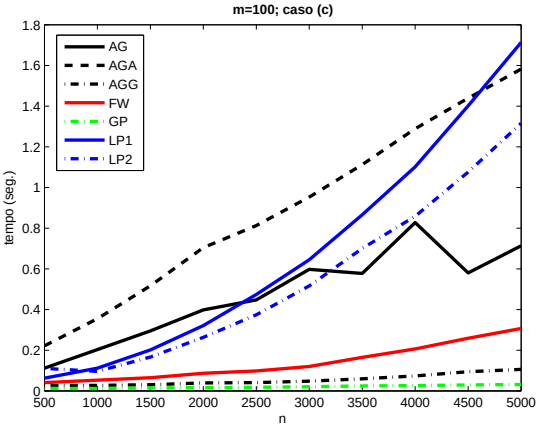

Figura 4.8 – Tempo médio em 10 instâncias com $m = 100$, $p = \frac{1}{2}v_\ell + \frac{1}{2}v_q$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-2}$.

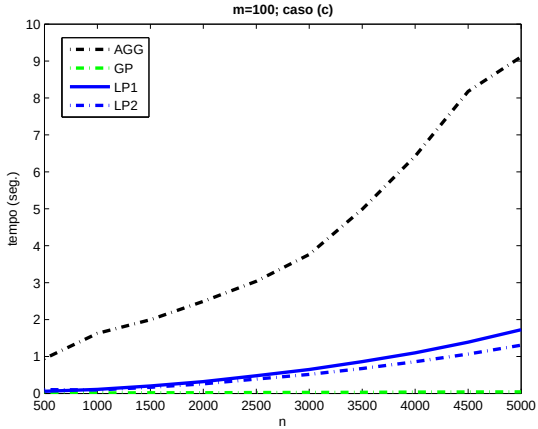

Figura 4.9 – Tempo médio em 10 instâncias com $m = 100$, $p = \frac{1}{2}v_\ell + \frac{1}{2}v_q$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-3}$.

Começamos com tolerância $\varepsilon = 10^{-2}$. Nas instâncias deste cenário, justamente por $\angle pp'v$ poder ficar arbitratiamente próximo de $\pi/2$, na Figura 4.8 podemos ver que o AGA (Algoritmo Geométrico com condição do ângulo), com $\kappa = 0.14$, apresentou um dos piores desempenhos,

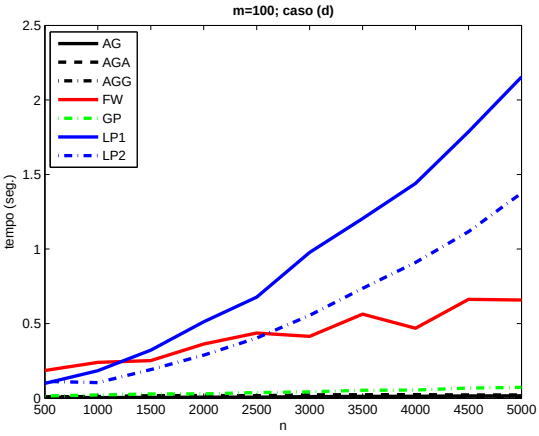

Figura 4.10 – Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 1.01$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$.

seguido pelas formulações lineares e pelo Algoritmo Geométrico original. Nestes testes, o Algoritmo Geométrico Ganancioso e o Gradiente Projetado apresentaram os melhores resultados.

A seguir, diminuímos a tolerância para $\varepsilon = 10^{-3}$. Neste caso, os algoritmos AG, AGA e FW não conseguiram resolver cada um dos problemas dentro do tempo limite de 20 segundos. Assim, na Figura 4.9 comparamos apenas as formulações lineares com o AGG e o Gradiente Projetado. Nestes problemas, em consequência de uma constante de visibilidade ruim, o Algoritmo Geométrico Ganancioso encontrou dificuldades para alcançar a precisão exigida (como já esperávamos do Teorema 3.4 e do Exemplo 3.13) e teve um desempenho inferior aos demais.

Além disso, a maior número de iterações do GP foi de aproximadamente 50 contra 58000 do AGG.

Cenário (d): fora, mas próximo da fronteira

Neste cenário consideramos p tal que $\|p\| = 1.01$. Assim como no cenário (b) temos que $p \notin \text{conv}(S)$ (pois $\text{conv}(S) \subset B(0,1)$), porém agora p está mais próximo da fronteira de $\text{conv}(S)$ de modo que a constante Δ será bem menor que no caso (b).

Apesar da complexidade dos algoritmos geométricos não depender

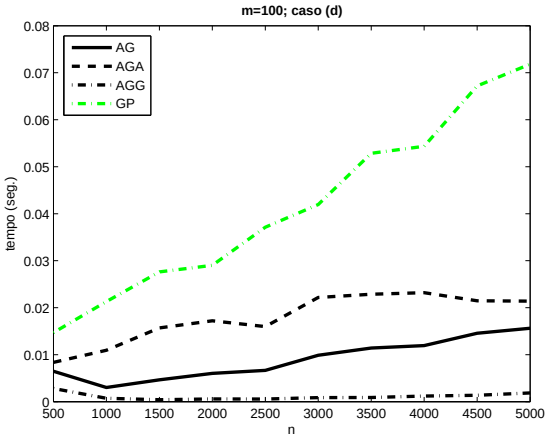

Figura 4.11 – Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 1.01$, para $n = 500, \ldots, 5000$, $\varepsilon = 10^{-4}$.

da tolerância ε neste caso, consideramos $\varepsilon = 10^{-4}$ pelos mesmos motivos do caso (b). Para $n \leq 5000$ obtemos os resultados da Figura 4.10 e da Figura 4.11.

O método de Frank-Wolfe apresentou um custo total mais elevado, pois foi necessário um maior número de iterações. Salientamos que no caso em que $p \notin \text{conv}(S)$, o Algoritmo Geométrico e suas variantes possuem um critério de parada diferenciado: tais algoritmos param assim que encontram uma testemunha.

Deste modo, enquanto os algoritmos geométricos não passaram de 20 iterações, FW teve uma média de 1500 iterações por problema. Já o GP teve uma média de 40 iterações, por esse motivo, mesmo que seu custo seja um pouco mais elevado que o do FW (por conta do cálculo da projeção), o GP obteve um desempenho melhor em função do tempo.

Assim, para o caso em que $n \geq 10000$, deixando fora LP1, LP2 e FW obtivemos o resultado da Figura 4.12. Podemos observar que nesse caso o Algoritmo Geométrico e suas formulações obtiveram resultados muito bons, mesmo com o Δ do Teorema 3.4 sendo pequeno em relação a R.

Analisando os resultados obtidos nos quatro cenários propostos, podemos notar que, em geral, o Algoritmo Geométrico e suas variantes obtiveram um desempenho melhor em relação às formulações lineares (resolvidas exatamente com o Dual-Simplex) e quadráticas (resolvidas

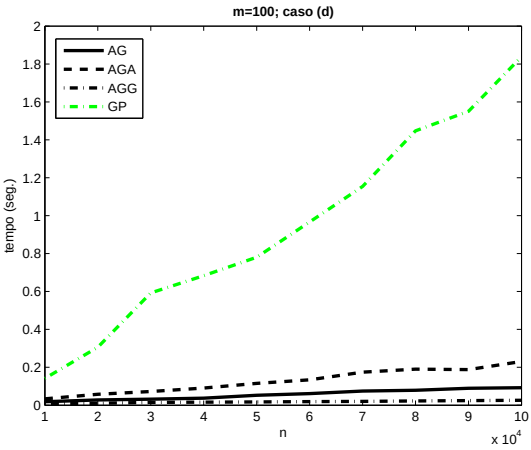

Figura 4.12 – Tempo médio em 10 instâncias com $m = 100$, $\|p\| = 1.01$, para $n = 10000, \ldots, 100000$.

com Frank-Wolfe e Gradiente Projetado), principalmente nos casos (b) e (d), quando $p \notin \text{conv}(S)$.

Pelo resultado de complexidade do Teorema 3.4 já esperávamos tal situação, pois nesse caso a complexidade depende exclusivamente da geometria do problema, isto é, da distribuição dos pontos de S e da posição relativa de p, o que determina as constantes Δ e R.

No cenário (a) vimos que o Algoritmo Geométrico Ganancioso se destacou perante aos demais, como previsto no Teorema 3.14, pelo fato de escolher um pivô estrito *forte* (sempre que possível).

No cenário (c), já esperávamos um desempenho não muito bom do Algoritmo Geométrico e suas variantes, justamente por conta do Teorema 3.11 e do Exemplo 3.13.

5 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho realizamos um estudo teórico e prático do Algoritmo Geométrico, proposto em [1], para o problema de inclusão no envoltório convexo (PIEC).

A corretude do algoritmo é baseada em um teorema de alternativas chamado Dualidade de Distâncias (Teorema 2.3) para o qual fornecemos uma nova demonstração inspirada no Teorema da Projeção e no Teorema do Hiperplano Separador.

Além disso, interpretando o PIEC como um problema de otimização, estabelecemos a relação do Algoritmo Geométrico com o clássico método de Frank-Wolfe, mostrando que o Algoritmo Geométrico pode ser visto como um método de Frank-Wolfe inexato.

Embora a análise de complexidade do Algoritmo Geométrico seja totalmente baseada em [1], fornecemos caracterizações alternativas para pivôs simples e estritos que nos permitiram analisar o algoritmo mais sob um ponto de vista de otimização. Essas caracterizações não só nos permitiram enxergar o Algoritmo Geométrico como um algoritmo de direções factíveis e de descida, como também motivaram a proposta de variantes do algoritmo original: o Algoritmo Geométrico com condição do ângulo e o Algoritmo Geométrico Ganancioso. Este último também foi motivado pela relação entre o algoritmo original e o método de Gradiente Condicional (Frank-Wolfe). Vimos que o Algoritmo Geométrico puro encontra $p' \in \text{conv}(S)$ tal que $d(p', p) < \varepsilon R$ em no máximo $O(\varepsilon^{-2})$ iterações. Outra característica interessante do algoritmo é que, quando $p \notin \text{conv}(S)$, o número de iterações para computar uma p-testemunha depende apenas da geometria do conjunto S e da posição relativa de p. Se utilizarmos pivôs estritos fortes ao invés de pivôs simples, no caso em que p pertence ao interior relativo de $\text{conv}(S)$, a complexidade de iteração passa de $O(\varepsilon^{-2})$ para $O(\ln \varepsilon^{-1})$.

Os resultados teóricos foram corroborados pelos experimentos computacionais realizados. Para ter uma ideia do desempenho prático do Algoritmo Geométrico, considerados formulações de programação linear e quadrática para o PIEC, e comparamos o AG com métodos clássicos para estas formulações. Os resultados obtidos indicam que o Algoritmo Geométrico e variantes apresentam um desempenho bom frente aos métodos clássicos, pelo menos nos problemas teste artificiais considerados neste trabalho.

Assim, concluímos que o Algoritmo Geométrico figura como um

alternativa promissora para o problema de inclusão no envoltório convexo, apresentando um bom desempenho prático, fundamentado por uma teoria clara e com resultados de complexidade.

Em trabalhos futuros pretendemos testar o Algoritmo Geométrico e variantes em problemas reais advindos da Geometria Computacional como, por exemplo, o problema da irredundância [5].

Outro tópico interessante a ser analisado, diz respeito a constante de visibilidade. Pela teoria, e por nossa experiência numérica, notamos que o pior caso para o Algoritmo Geométrico é quando o ponto $p \in \text{conv}(S)$ e está muito perto de (ou sobre) a fronteria. Neste caso, a constante de visibilidade ν pode estar muito próxima de 1 e com isso a redução no gap de otimalidade será muito lenta, ocorrendo o fenômeno de "zigzag". No futuro, pretendemos estudar como melhorar essa constante de visibilidade. Uma ideia, proposta em [1], é a de incluir *pivôs auxiliares* no conjunto S sempre que não for possível obter um pivô v simples/estrito tal que o ângulo $\angle pp'v$ seja pequeno o suficiente. Por exemplo, podemos incluir como pivôs auxiliares combinações convexas dos pivôs escolhidos com mais frequência ao longo das iterações.

Por fim, uma outra aplicação interessante do Algoritmo Geométrico está ligada ao problema de factibilidade em programação linear. Seja

$$\Omega = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\},$$

com $A = [a_1, a_2, \ldots, a_n]$, sendo $b, a_i \in \mathbb{R}^m$. É possível mostrar que, quando Ω é um conjunto limitado, verificar se $\Omega \neq \emptyset$ equivale a decidir se

$$0 \in \text{conv}\{a_1, a_2, \ldots, a_n, -b\}.$$

REFERÊNCIAS

[1] KALANTARI, B. A characterization theorem and an algorithm for a convex hull problem. *Annals of Operations Research*, Springer, v. 226, n. 1, p. 301–349, 2015.

[2] BAZARAA, M. S.; JARVIS, J. J. *Linear programming and network flows*. [S.l.]: Wiley New York, 1977.

[3] LUENBERGER, D. G.; YE, Y. *Linear and Nonlinear Programming*. [S.l.]: Springer, 2008.

[4] DANTZIG, G. B. *Linear Programming and Extensions*. [S.l.]: Princeton Uinversity Press, USA, 1963.

[5] GOODMAN, E. J.; O'ROURKE, J. *Handbook of discrete and computational geometry*. [S.l.]: CRC, 1997. ISBN 0084938524.

[6] CHEN, L.-L.; WOO, T. C. Computational geometry on the sphere with application to automated machining. *Journal of Mechanical Design*, American Society of Mechanical Engineers, v. 114, n. 2, p. 288–295, 1992.

[7] BERTSEKAS, D. P. *Nonlinear programming*. [S.l.]: Athena Scientific Belmont, 1999.

[8] ROCKAFELLAR, R. T. *Convex Analysis*. [S.l.]: Princeton University Press, 1970.

[9] MARTÍNEZ, J. M.; SANTOS, S. A. *Métodos Computacionais de Otimização*. [S.l.]: IMPA, 1995.

[10] FRANK, M.; WOLFE, P. An algorithm for quadratic programming. *Naval research logistics quarterly*, v. 3, n. 1-2, p. 95–110, 1956.

[11] BAILEY, T.; COWLES, J. A convex hull inclusion test. *IEEE transactions on pattern analysis and machine intelligence*, n. 2, p. 312–316, 1987.

[12] JAGGI, M. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In: *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*. [s.n.], 2013. v. 28, n. 1, p. 427–435. Disponível em: <http://jmlr.csail.mit.edu/proceedings/papers/v28/jaggi13.pdf>.

[13] LI, M.; KALANTARI, B. Experimental study of the convex hull decision problem via a new geometric algorithm. In: *23rd Annual Fall Workshop on Computational Geometry, City College of New York*. [S.l.: s.n.], 2013.

[14] MICHELOT, C. A finite algorithm for finding the projection of a point onto the canonical simplex of $\mathbb{R}^n$. *Journal of Optimization Theory and Applications*, Springer, v. 50, n. 1, p. 195–200, 1986.

[15] GONÇALVES, D. S.; GOMES-RUGGIERO, M. A.; LAVOR, C. A projected gradient method for optimization over density matrices. *Optimization Methods and Software*, Taylor & Francis, v. 31, n. 2, p. 328–341, 2016.

[16] DEUTSCH, F. R. Rate of Convergence of the Method of Alternating Projections. In: BROSOWSKI, B.; DEUTSCH, F. R. (Ed.). *Parametric Optimization and Approximation*. Basel: Birkhäuser, Basel, 1984. p. 96–107.

[17] HALPERIN, I. The product of projection operators. *Acta Sci. Math. (Szeged)*, v. 23, p. 96–99, 1962.

[18] HARMAN, R.; LACKO, V. On decompositional algorithms for uniform sampling from n-spheres and n-balls. *Journal of Multivariate Analysis*, v. 101, n. 10, p. 2297–2304, 2010.

[19] NOCEDAL, J.; WRIGHT, S. J. *Numerical Optimization*. 2. ed. [S.l.]: Springer-Verlag, 2006.

[20] LIMA, E. L. *Curso de análise*. 11^a. [S.l.]: IMPA, 2015. v. 2.

[21] CLARKSON, K. L. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Transactions on Algorithms (TALG)*, ACM, v. 6, n. 4, p. 63:1–30, 2010.

A ALGUMAS DEMONSTRAÇÕES DO CAPÍTULO 1

Neste apêndice serão apresentadas algumas demonstrações dos teoremas que foram mais utilizados ao decorrer da dissertação. Tendo em vista que as demonstrações aqui feitas podem ser encontradas em [7] e [3]

Demonstração: Teorema 1.17. (a) Fixe $x \in \mathbb{R}^n$ e seja w algum elemento de Ω. Minimizar $\|x - z\|$ para $z \in \Omega$ é equivalente a minimizar a mesma função sujeita a $z \in \Omega$ tal que $\|x - z\| \leq \|x - w\|$, que é um conjunto compacto. Mais ainda, a função g definida por $g(z) = (1/2)\|z - x\|^2$ é contínua, e pelo teorema de Weierstrass [20], existe $P_\Omega(x) \in \Omega$ minimizador global de $g(z)$ em Ω. Para provar a unicidade, note que g é estritamente convexa e pela Proposição 1.9(b) tal minimizador global é único.

(b) Seja $z = P_\Omega(x)$, isto é: $\|x - z\| \leq \|x - y\|, \quad \forall y \in \Omega$. Como g é estritamente convexa e continuamente diferenciável em Ω convexo, pelo Teorema 1.13, z é minimizador global de g em Ω se, e somente se, $\nabla g(z)^T(y - z) \geq 0$, para todo $y \in \Omega$. Mas $\nabla g(z)^T(y - z) = (z - x)^T(y - z) = -(x - z)^T(y - z)$, de onde segue que $(y - z)^T(x - z) \leq 0, \forall y \in \Omega$.

(c) Do item (b) temos que, dado $v \in \mathbb{R}^n$, $(w - P_\Omega(v))^T(v - P_\Omega(v)) \leq 0$ para todo $w \in \Omega$. Fazendo $w = P_\Omega(y)$ e $v = x$, e depois $w = P_\Omega(x)$ e $v = y$, obtemos
$(P_\Omega(y) - P_\Omega(x))^T(x - P_\Omega(x)) \leq 0$ e $(P_\Omega(x) - P_\Omega(y))^T(y - P_\Omega(y)) \leq 0$. Adicionando ambas:

$$(P_\Omega(y) - P_\Omega(x))^T(x - P_\Omega(x) - y + P_\Omega(y)) \leq 0.$$

Da desigualdade acima, e usando Cauchy-Schwarz, obtemos

$$\begin{aligned}\|P_\Omega(x) - P_\Omega(y)\|^2 &\leq (P_\Omega(x) - P_\Omega(y))^T(x - y) \\ &\leq \|P_\Omega(x) - P_\Omega(y)\| \, \|x - y\|.\end{aligned}$$

Portanto, assumindo $P_\Omega(x) \neq P_\Omega(y)$, chegamos a

$$\|P_\Omega(x) - P_\Omega(y)\| \leq \|y - x\|. \tag{A.1}$$

Quando $P_\Omega(x) = P_\Omega(y)$ a desigualdade acima é trivialmente satisfeita.

Como (A.1) é válida para $x, y \in \mathbb{R}^n$ quaisquer, temos que $f(x) = P_\Omega(x)$ é função Lipschitz e portanto contínua.

(d) Segue trivialmente do item (c) e do fato de que $P_\Omega(y) = y$ se $y \in \Omega$.

□

Demonstração: Teorema 1.18. Seja x_* um ponto estacionário para (1.2). Então, do Teorema 1.13, temos que

$$\nabla f(x_*)^T(x - x_*) \geq 0, \quad \forall x \in \Omega.$$

A desigualdade acima equivale a $(x_* - \nabla f(x_*) - x_*)^T(x - x_*) \leq 0, \forall x \in \Omega$, e pela caracterização da projeção (item (b) do Teorema 1.17), isto é válido se, e somente se,

$$x_* = P_\Omega(x_* - \nabla f(x_*)).$$

□

Demonstração: Teorema 1.21. Seja $\delta = \min_{y \in \Omega} |y - x| = |P_\Omega(x) - x| > 0$, pois $x \notin \Omega$. Vamos mostrar que definindo $a = P_\Omega(x) - x$, chegamos ao resultado desejado. Pelo item (b) do Teorema 1.17, temos que:

$$(P_\Omega(x) - x)^T(y - P_\Omega(x)) \geq 0, \quad \forall y \in \Omega. \tag{A.2}$$

De (A.2), temos que para todo $y \in \Omega$:

$$\begin{aligned}
(P_\Omega(x) - x)^T y &\geq (P_\Omega(x) - x)^T P_\Omega(x) \\
&= (P_\Omega(x) - x)^T x + (P_\Omega(x) - x)^T (P_\Omega(x) - x) \\
&\geq (P_\Omega(x) - x)^T x + \delta^2 \\
&> (P_\Omega(x) - x)^T x.
\end{aligned}$$

Assim, fazendo $a = (P_\Omega(x) - x)$ está provado o enunciado. □