Deise Gonçalves Ferreira

Sobre Métodos de Busca Padrão para Minimização de Funções com Restrições Lineares

Campinas
2013

Universidade Estadual de Campinas

Instituto de Matemática, Estatística
e Computação Científica

Deise Gonçalves Ferreira

Sobre Métodos de Busca Padrão para Minimização de Funções com Restrições Lineares

Orientadora: Profa. Dra. Maria Aparecida Diniz Ehrhardt

Dissertação de mestrado apresentada ao Instituto de Matemática, Estatística e Computação Científica da Unicamp para obtenção do título de mestra em matemática aplicada.

Este exemplar corresponde à versão final da dissertação defendida pela aluna Deise Gonçalves Ferreira, e orientada pela Profa. Dra. Maria Aparecida Diniz Ehrhardt

Assinatura do Orientador

Campinas
2013

FICHA CATALOGRÁFICA ELABORADA POR
MARIA FABIANA BEZERRA MULLER - CRB8/6162
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E
COMPUTAÇÃO CIENTÍFICA - UNICAMP

Ferreira, Deise Gonçalves, 1988-
F413s Sobre métodos de busca padrão para minimização de funções com restrições lineares / Deise Gonçalves Ferreira. – Campinas, SP : [s.n.], 2013.

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

1. Otimização com restrições. 2. Otimização sem derivadas. 3. Métodos de busca padrão. I. Ehrhardt, Maria Aparecida Diniz,1956-. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Informações para Biblioteca Digital

Título em inglês: On pattern search methods for linearly constrained minimization
Palavras-chave em inglês:
Constrained optimization
Derivative-free Optmization
Pattern search methods
Área de concentração: Matemática Aplicada
Titulação: Mestra em Matemática Aplicada
Banca examinadora:
Maria Aparecida Diniz Ehrhardt [Orientador]
Márcia Aparecida Gomes Ruggiero
Elizabeth Wegner Karas
Data de defesa: 04-03-2013
Programa de Pós-Graduação: Matemática Aplicada

Dissertação de Mestrado defendida em 04 de março de 2013 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.

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

Prof.(a). Dr(a). MÁRCIA APARECIDA GOMES RUGGIERO

Prof.(a). Dr(a). ELIZABETH WEGNER KARAS

Ao meu pai (in memoriam)...

"Por vezes sentimos que aquilo que fazemos não é senão uma gota de água no mar. Mas o mar seria menor se lhe faltasse uma gota."
(Madre Teresa de Calcutá)

Agradecimentos

Agradeço acima de tudo a Deus por me dar a cada dia a saúde e a coragem para correr atrás de meus objetivos e por todas as oportunidades que surgiram durante essa longa caminhada de mais de sete anos desde que alimentei o sonho de estudar na Unicamp. A Jesus e a Nossa Senhora, intercessores infalíveis, aos quais recorro nos momentos de dificuldade.

À minha mãe, por ser sempre um exemplo de força e humildade, uma pessoa que apesar de todas as dificuldades que enfrentou na vida sempre esteve de pé com essa força que mantém nossa família unida, muito obrigada pelo apoio e acima de tudo por todo o amor que recebo de ti.

Aos meus irmãos, Lúcia, Nadir, Wanda, Margareth, Claudete e Sérgio, e aos meus sobrinhos, Jéssica, Karoline, Bruno, Hugo, Leonardo, Thainá, Bianca, Lavínia e Victória, por todo o apoio e companheirismo durante toda minha vida.

À Daiane, minha fiel companheira, juntas alimentamos esse sonho e juntas corremos atrás para realizá-lo, não consigo pensar nesta trajetória sem a sua presença, só você e Deus, sabem em detalhes a minha história, e você mais do que ninguém, sabe que essa conquista não é minha, é nossa.

À minha orientadora, professora Cheti que com muito carinho e dedicação me guiou durante a realização deste trabalho, seus ensinamentos foram e são fundamentais para minha formação, não somente do ponto de vista acadêmico, mas sobretudo como exemplo de comprometimento e amor pelo que faz, o que me impulsionou a dar o melhor de mim neste trabalho. Fico feliz e grata, em saber que poderei contar com sua orientação e amizade no Doutorado.

À professora Márcia Gomes Ruggiero, com quem aprendi muito, obrigada pelas preciosas sugestões e leitura cuidadosa da versão preliminar deste texto, que juntamente com as sugestões do professor José Mario Martínez ajudaram a enriquecer esse trabalho.

Aos meus professores na educação básica, por despertarem em mim a vontade de seguir em frente, em especial gostaria de citar as professoras, Marilene, Adriana, Juraci, Ilza, Sueli, Maria Antonia, Alessandra, o professor Zabeu, e tantos outros que apesar de todas as dificuldades de uma escola pública mostraram que é possível fazer a diferença. Gostaria de agradecer de modo muito especial ao diretor da escola Augusta, professor Joaquim, que não mediu esforços para ajudar, eu e minha irmã, na luta por esse sonho de estudar na Unicamp, obrigada por acreditar que seríamos capazes.

Aos meus professores do IMECC, por terem me proporcionado uma formação matemática sólida tornando possível essa empreitada, em especial aos professores: Gilli, cujos ensinamentos vão além do Cálculo, um grande amigo por quem tenho profundo carinho, respeito e admiração. Ao Luis Felipe, pelas diversas conversas que tivemos que me ajudaram a escolher a área de Otimização e principalmente na escolha correta da minha orientadora. Sem mais delongas gostaria de citar também os professores, Petronio Pulino, Benjamin Bordin, Sandra Santos, Margarida Mello, A.C. Patrocínio e Aurélio Ribeiro, pessoas especiais que fizeram a diferença na minha formação.

Aos meus amigos, que me proporcionaram ao longo destes anos muitos momentos de alegria e descontração, que me permitiam voltar aos estudos com as energias renovadas.

Aos funcionários do IMECC, em especial aos funcionários da Secretária de graduação e pós-graduação sempre tão solícitos e prestativos, todas as vezes que precisei.

Finalizando, agradeço à Capes pelo apoio financeiro que possibilitou o desenvolvimento deste trabalho.

Resumo

Neste trabalho voltamos nossa atenção para métodos de otimização que não fazem uso de derivadas. Dentre esses, estamos interessadas em um método de busca padrão para minimização de funções com restrições lineares. Abordamos um algoritmo proposto por Lewis e Torczon, cuja ideia geral é que o padrão deve conter direções de busca ao longo das quais iterações factíveis sejam determinadas. O algoritmo possui resultados de convergência global. Realizamos sua implementação computacional, e propomos novas estratégias de busca e atualização do tamanho do passo, além de um novo padrão de direções de busca. Realizamos testes numéricos, de modo a analisar o desempenho das estratégias propostas e comparar o desempenho do padrão de direções que introduzimos com o proposto por Lewis e Torczon.

Palavras-chave: Minimização com restrições lineares, Otimização sem derivadas, Métodos de Busca Padrão.

Abstract

In this work, our interest lies on derivative-free optimization methods. Among these, our aim is to study a pattern search method for linearly constrained minimization. We studied an algorithm proposed by Lewis and Torczon, whose general idea is that the pattern must contain search directions in which feasible iterations must be computed. The algorithm has global convergence results. We accomplished its computational implementation and we propose new strategies of search and updating rule for the step-length control parameter. We also propose a new pattern of search directions. We accomplished numerical experiments in order to analize the performace of our proposals and also to compare the performance of our pattern with the one proposed by Lewis and Torczon.

Keywords: Linearly constrained minimization, Derivative-free Optimization, Pattern Search Methods.

Notação Utilizada

$\mathbb{Z}, \mathbb{Q}$ e $\mathbb{R}$ representam os conjuntos dos números inteiros, racionais e reais, respectivamente;

$\mathbb{B}(x, \epsilon)$ representa a bola de raio ϵ centrada no ponto x;

Ω representa o conjunto viável;

$x^\star$ representa a solução ótima conhecida;

∇f representa o vetor gradiente da função f;

$\nabla^2 f$ representa a matriz hessiana de f;

$L(x^0)$ representa o conjunto de nível da função f;

$L_\Omega(x^0)$ representa o conjunto de nível de f em Ω;

$\emptyset$ representa o conjunto vazio;

$\|\bullet\|$ representa a norma-2 vetorial;

P_Ω representa a projeção no espaço viável Ω;

I representa a matriz identidade;

K e K° representam cones, onde K° é o polar do cone K;

$D(\bar{x})$ e $T(\bar{x})$ representam o cone viável linearizado e o cone de direções tangentes, respectivamente;

$\mathbb{I}, \mathbb{D}, \mathbb{D}_l, \mathbb{D}_u, \mathbb{D}_e$ representam conjuntos de índices;

a_i representa a linha i da matriz A

α^k representa o tamanho do passo na iteração k;

P^k respresenta o padrão de direções na iteração k;

$c^k \in P^k$ representa uma direção (coluna) do padrão P^k;

$c_i^k \in P^k$ representa a coluna i do padrão na iteração k;

$s^k \in \alpha^k P^k$ representa uma direção do padrão P^k escalada pelo tamanho do passo α^k;

$s_i^k \in \alpha^k P^k$ representa a direção i do padrão P^k escalada pelo tamanho do passo α^k;

$\mathcal{N}(A)$ representa o núcleo da matriz A;

$\mathcal{R}(A)$ representa a imagem da matriz A;

Conteúdo

Introdução **1**

1 Conceitos Fundamentais de Otimização **3**
1.1 O Problema Geral de Otimização 3
1.2 Condições de Otimalidade 4
1.2.1 Minimização Irrestrita 4
1.2.2 Minimização com Restrições 5

2 Métodos de Busca Direta **9**
2.1 Métodos de Busca Direta Direcionais 10
2.1.1 Método de Busca Coordenada 10
2.1.2 Método de Hooke e Jeeves 11
2.2 Métodos Simplex 11

3 Métodos de Busca Padrão **13**
3.1 Direções de Busca 13
3.2 Movimentos Exploratórios 15
3.3 Algoritmo Geral 16
3.3.1 Atualizações 17
3.3.2 Critério de Parada 18
3.4 Análise de Convergência do Método 18

4 Método de Busca Padrão para Minimização com Restrições Lineares **21**
4.1 O Padrão de Direções 22
4.1.1 Restrições Geométricas no Padrão 23
4.2 Movimentos Exploratórios e Atualizações 25
4.3 Algoritmo Geral 25
4.3.1 Critério de Parada 26
4.4 Análise de Convergência 26

5 Gerando o Padrão de Direções de Busca **29**
5.1 Minimização em Caixas 30
5.2 Restrições Lineares de Igualdade 30
5.3 Conjunto de Trabalho Vazio 31

5.4 Problemas com Restrições Lineares Gerais 32
5.4.1 Conjunto de Trabalho Não-degenerado 32
5.4.2 Conjunto de Trabalho Degenerado 38

6 Implementação Computacional do Método **41**
6.1 Ponto Inicial 41
6.2 Estratégias de Busca 41
6.2.1 Busca Simples 42
6.2.2 Busca Completa 43
6.2.3 Busca Simples Ordenando o Padrão 43
6.3 Atualização do Tamanho do Passo 45
6.4 Informações Complementares 45

7 Testes Numéricos **47**
7.1 Problemas Suaves 48
7.1.1 Busca Simples 48
7.1.2 Busca Completa 51
7.1.3 Busca Simples Ordenando o Padrão 53
7.1.4 Comparação entre as Estratégias de Busca 54
7.2 Problemas Não Suaves 56
7.2.1 Busca Simples 57
7.2.2 Busca Completa 59
7.2.3 Busca Simples Ordenando o Padrão 61
7.2.4 Comparação entre as Estratégias de Busca 63
7.3 Modelo Suave para Problemas Minimax 63
7.4 Desempenho Geral do Método 66
7.4.1 Busca Simples 67
7.4.2 Busca Completa 69
7.4.3 Busca Simples Ordenando o Padrão 71
7.4.4 Comparação entre as estratégias de busca 73
7.5 Problemas Degenerados 74

8 Considerações Finais **79**

A Tabelas com os Resultados dos Testes **81**
A.1 Problemas Suaves 81
A.2 Problemas Não Suaves 102
A.3 Problemas Adicionais 111

Bibliografia **117**

Introdução

Neste trabalho trataremos de problemas de Otimização da forma:

$$(PRL)\begin{cases} min & f(x) \\ sujeita\ a & l \leq Ax \leq u, \end{cases}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$, $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $l, u \in \mathbb{R}^m$. Podemos ter $l_i = -\infty$ ou $u_i = \infty$, $i \in \{1, ..., m\}$. Restrições de igualdade podem ser consideradas permitindo $l_i = u_i$. Em um primeiro momento, não vamos fazer nenhuma hipótese sobre a diferenciabilidade da função f. Mas, de qualquer modo, ainda que f seja continuamente diferenciável em $\mathbb{R}^n$, não esperamos que suas derivadas possam ser calculadas ou aproximadas explicitamente.

Problemas de otimização em que as derivadas não podem ser calculadas têm aparecido de modo crescente em aplicações científicas e industriais, como por exemplo, em problemas de simulações; em problemas que têm como função objetivo um arquivo executável cujo código não está disponível ao usuário, conhecidos como caixas-pretas; em problemas em que a função objetivo é a medida de desempenho computacional de um algoritmo, sendo assim não temos uma expressão para a função; ou mesmo em situações em que o custo computacional de calcular as derivadas da função é muito alto, tornando-se inviável utilizar métodos que fazem uso de derivadas.

Para resolver esses problemas, faz-se necessário o desenvolvimento de algoritmos eficientes que não necessitam das derivadas da função. Voltamos nossa atenção para a resolução do problema de minimização com restrições lineares, uma vez que há inúmeras aplicações que nos remetem à resolução de problemas com restrições apenas lineares. Além disso, esse tipo de problema aparece muitas vezes como um subproblema em métodos aplicados a problemas gerais [3].

Estudamos uma classe muito especial de métodos sem derivadas, os de busca direta [10], que além de não computar as derivadas, não utilizam os valores explicitos da função em qualquer tipo de cálculo. Só é permitido a tais métodos questionar qual entre dois pontos tem o menor valor de função objetivo. São de fácil implementação e baixo custo da iteração, e aparecem como uma alternativa em ocasiões onde métodos mais elaborados falham.

Os principais tipos de métodos de busca direta são os métodos simplex e os de busca padrão. Considerando a minimização irrestrita, temos, no primeiro caso, o clássico método de Nelder-Mead [18]. Sua popularidade está nos bons resultados práticos que, em geral, apresenta, apesar de não possuir garantias teóricas de convergência.

Já os métodos de busca padrão, que abordamos com detalhes neste trabalho, sintetizam o espírito dos bons métodos de busca direta, de algoritmos com forte apelo geométrico e garantias teóricas de convergência. Em [24], Virginia Torczon define o modelo geral de algoritmo de busca padrão e apresenta uma robusta teoria de convergência. Assim, muitos dos métodos de busca direta existentes na época foram classificados como de busca padrão e passaram a ter teoria de convergência.

Realizamos um estudo detalhado da abordagem de Lewis e Torczon [11]: o algoritmo proposto pelos autores se caracteriza como um método de busca padrão estendido para minimização com restrições lineares, o qual possui resultados de convergência global. A ideia geral é que o padrão deve conter direções de busca que consistam em um conjunto de geradores para o cone de direções factíveis.

Além do estudo teórico, abordamos o algoritmo sob o ponto de vista computacional, seguindo parte do que está discutido no trabalho de R. M. Lewis, A. Shepherd e V. Torczon [12]. O desempenho dos métodos de busca padrão está diretamente ligado às estratégias de busca, simples ou completa, e atualização do tamanho do passo utilizados, mas principalmente ao padrão de direções considerado. Sendo assim, propomos novas estratégias de busca e atualização do tamanho do passo, além de um novo padrão de direções, com o objetivo de melhorar o desempenho do método estudado.

Realizamos experimentos numéricos, utilizando problemas testes suaves e não suaves, e comparamos o desempenho do método utilizando as estratégias e o padrão que propomos com o proposto pelos autores em [12].

Capítulo 1

Conceitos Fundamentais de Otimização

Neste capítulo organizamos alguns conceitos e resultados importantes no estudo de Otimização, e que são a base para o desenvolvimento de qualquer método de Otimização seja ele com ou sem derivadas. Algumas referências sobre esse assunto são [1, 14, 19].

1.1 O Problema Geral de Otimização

O objetivo geral ao resolver um problema de Otimização é encontrar um ponto que minimize (ou maximize) o valor de uma função, entre os pontos que pertencem a uma determinada região do espaço, a de soluções viáveis. Então o problema geral de Otimização é dado por:

$$(P) \quad \begin{cases} \min & f(x) \\ s.a & g_i(x) \leq 0 \quad para\ i = 1, \ldots, m, \\ & h_i(x) = 0 \quad para\ i = 1, \ldots, l, \end{cases}$$

onde $f, g_1, \ldots, g_m, h_1, \ldots, h_l$ são funções reais definidas em $\mathbb{R}^n$. Chamaremos de Ω a região viável, ou seja, o conjunto dos pontos que satisfazem todas as restrições do problema. Todo problema de Otimização pode ser escrito desta forma.

Definição 1.1 ***(Minimizador)*** *Considere uma função $f : \mathbb{R}^n \to \mathbb{R}$ e $x^\star \in \Omega \subset \mathbb{R}^n$. Dizemos que $x^\star$ é um minimizador local de f em Ω se existe $\delta > 0$ tal que $f(x^\star) \leq f(x)$, para todo $x \in \mathbb{B}(x^\star, \delta) \cap \Omega$. Caso $f(x^\star) \leq f(x)$, para todo $x \in \Omega$, $x^\star$ é chamado de minimizador global de f em Ω.*

Na Definição 1.1, quando as desigualdades forem estritas para $x \neq x^\star$, dizemos que $x^\star$ é um minimizador local (global) estrito de f em Ω. Nosso objetivo final é encontrar os minimizadores, porém precisamos conhecer algumas condições sobre o problema para garantir que eles de fato existem. Os resultados abaixo nos fornecem tais condições.

Teorema 1.1 ***(Weierstrass)*** *Sejam $f : \mathbb{R}^n \to \mathbb{R}$ contínua e $\Omega \subset \mathbb{R}$ compacto e não vazio. Então existe minimizador global de f em Ω.*
Demonstração: *Ver [22].*

Corolário 1.1 *Seja $f : \mathbb{R}^n \to \mathbb{R}$ contínua e suponha que existe $c \in \mathbb{R}$ tal que o conjunto de nível $L_c = \{x \in \mathbb{R}^n \ : \ f(x) \leq c\}$ é compacto e não vazio. Então f tem um minimizador global.*
Demonstração: *Ver [22].*

Os resultados acima em geral são a base para o estudo de convergência de algoritmos, uma vez que temos a garantia teórica de existência da solução. Neste trabalho, para o estudo de convergência do método, são exigidas sobre o problema as condições presentes no Corolário 1.1.

1.2 Condições de Otimalidade

Embora as Condições de Otimalidade definidas aqui utilizem informações do gradiente da função objetivo e das restrições do problema, todas as demonstrações de convergência para os métodos sem derivadas são baseadas nestas condições, ainda que sejam desenvolvidos outros critérios para análise de otimalidade.

Definição 1.2 ***(Direção viável)*** *Seja $\Omega \subset \mathbb{R}^n$ o conjunto viável. Uma direção $d \in \mathbb{R}^n$ é dita viável, ou factível, a partir de $x \in \Omega$ se existe $\bar{\alpha} > 0$ tal que $(x + \alpha d) \in \Omega, \forall\ \alpha \in [0, \bar{\alpha}]$.*

Teorema 1.2 (Direção de descida) *Seja $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável em $\bar{x}$. Se existe um vetor d tal que $\nabla f(\bar{x})^T d < 0$, então existe $\delta > 0$ tal que $f(\bar{x} + \lambda d) < f(\bar{x})$ para cada $\lambda \in (0, \delta)$, e d é chamada de* direção de descida *para f a partir de $\bar{x}$.*
Demonstração: *Ver [1].*

A otimalidade de um ponto caracteriza-se pela inexistência de uma direção de descida factível a partir deste ponto. Porém na prática não é possível verificar todas as direções para garantir que não existe tal direção. Portanto é necessário estabelecer outras condições que possam ser verificadas para podermos determinar a otimalidade de uma solução. Na próxima seção estabelecemos algumas condições, para o caso irrestrito.

1.2.1 Minimização Irrestrita

Teorema 1.3 ***(Condições Necessárias de 1^a ordem)*** *Seja $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável em $x^\star \in \mathbb{R}$. Se $x^\star$ é minimizador local de f, então $x^\star$ é estacionário, ou seja:*

$$\nabla f(x^\star) = 0.$$

Demontração: *Ver [22].*

Teorema 1.4 ***(Condições Necessárias de*** 2^a ***ordem)*** *Seja* $f : \mathbb{R}^n \to \mathbb{R}$ *duas vezes diferenciável em* $x^\star \in \mathbb{R}$. *Se* $x^\star$ *é minimizador local de* f, *então* $\nabla f(x^\star) = 0$ *e a matriz Hessiana de* f *é semidefinida positiva, ou seja:*

$$d^t \nabla^2 f(x^\star) d \geq 0$$

para todo $d \in \mathbb{R}^n$.
Demonstração: *Ver [1]*

Teorema 1.5 ***(Condições suficientes de*** 2^a ***ordem)*** *Seja* $f : \mathbb{R}^n \to \mathbb{R}$ *duas vezes diferenciável em* $x^\star \in \mathbb{R}$. *Se* $\nabla f(x^\star) = 0$ *e* $\nabla^2 f(x^\star)$ *é definida positiva, ou seja:*

$$d^t \nabla^2 f(x^\star) d > 0$$

para todo $d \in \mathbb{R}^n$. *Então* $x^\star$ *é minimizador local estrito de f.*
Demonstração: *Ver [1].*

As condições de otimalidade também são importantes por fornecerem um critério de parada para os algoritmos que fazem uso de derivadas. Na prática é utilizada como critério de parada somente a condição necessária de 1^a ordem, com isso temos a garantia de convergência para um ponto estacionário, mas não para um minimizador do problema.

Para problemas restritos é necessário que o minimizador satisfaça todas as restrições do problema. Como pode ocorrer de os pontos estacionários estarem fora da região viável, o minimizador para o problema restrito em geral não satisfaz as condições de otimalidade 1.3, 1.4 e 1.5. Na próxima seção apresentamos as condições de otimalidade do problema restrito.

1.2.2 Minimização com Restrições

Para estabelecer as condições de otimalidade para o problema de otimização com restrições, precisamos antes definir alguns conceitos fundamentais da teoria de cones, os quais também serão úteis para a construção de direções de busca no método que estudamos.

Definição 1.3 ***(Cone)*** *Um conjunto não vazio* $K \in \mathbb{R}^n$ *é chamado de cone com vértice na origem se para todo* $x \in K$ *temos que* $\alpha x \in K$, *para todo* $\alpha \geq 0$. *Se além disso o conjunto* K *é convexo,* K *é chamado de cone convexo.*

Definição 1.4 ***(Cone Polar)*** *Seja* K *um conjunto não vazio em* $\mathbb{R}^n$. *Então o cone polar de* K, *denotado por* K°, *é dado por:* $K^\circ = \{p : p^t x \leq 0,$ *para todo* $x \in K\}$. *Se* $K = \emptyset$ *então consideramos* $K^\circ = \mathbb{R}^n$.

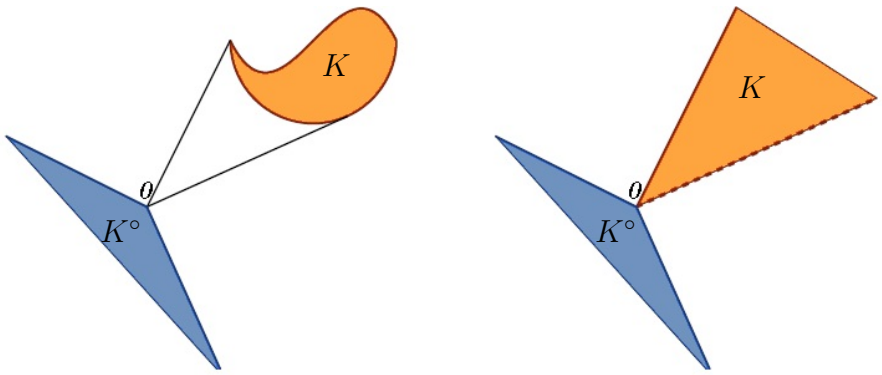


Figura 1.1: Ilustração de cone Polar

Definição 1.5 ***(Restrição Ativa)*** *Uma restrição de desigualdade g_i é dita ativa em $\bar{x}$ se $g_i(\bar{x}) = 0$. Caso $g_i(\bar{x}) < 0$, dizemos que g_i é inativa em $\bar{x}$.*

Denotaremos por $\mathcal{I}(\bar{x})$ o conjunto de índices das restrições de desigualdades ativas em um ponto viável $\bar{x}$, ou seja:

$$\mathcal{I}(\bar{x}) = \{i : g_i(\bar{x}) = 0\}.$$

Definição 1.6 ***(Cone viável linearizado)*** *Dado $\bar{x} \in \Omega$, definimos o cone viável linearizado de Ω em torno de $\bar{x}$ por:*

$$D(\bar{x}) = \left\{d \in \mathbb{R}^n : \nabla g_i(\bar{x})^t d \leq 0, \forall i \in \mathcal{I}(\bar{x}) \text{ e } \nabla h_j(\bar{x})^t d = 0, \forall j = 1, \ldots, m\right\}.$$

Definição 1.7 ***(Direções Tangentes)*** *Uma direção $d \in \mathbb{R}^n$ é dita tangente a $\Omega \subset \mathbb{R}^n$ a partir de $\bar{x} \in \Omega$ quando é nula ou existe uma sequência de pontos viáveis $(x^k) \subset \Omega$ tal que $x^k \to \bar{x}$ e*

$$\frac{x^k - \bar{x}}{\|x^k - \bar{x}\|} \longrightarrow \frac{d}{\|d\|}.$$

Perceba que, como a direção nula é uma direção tangente e dado $\alpha > 0$, se d é uma direção tangente, αd também é, então o conjunto das direções tangentes num determinado ponto é um cone, o qual chamaremos de cone tangente. Denotaremos por $T(\bar{x})$ o cone de direções tangentes a Ω a partir de $\bar{x} \in \Omega$.

O cone viável linearizado e o cone tangente são úteis por nos fornecer aproximações para o cone de direções viáveis no ponto $\bar{x}$.

Definição 1.8 ***(Condições de Karush-Kuhn-Tucker)*** *Um ponto $x^\star$ factível para o problema (P) é chamado de ponto KKT se existem $\mu^\star \in \mathbb{R}^p$ e $\lambda^\star \in \mathbb{R}^m$ tais que:*

$$\begin{cases} -\nabla f(x^\star) = \sum_{i=1}^{p} \mu_i^\star \nabla g_i(x^\star) + \sum_{j=1}^{m} \lambda_i^\star \nabla h_i(x^\star) \\ \mu_i^\star \geq 0, \quad i = 1, \ldots, p \\ \mu_i^\star \nabla g_i(x^\star) = 0, \quad i = 1, \ldots, p \end{cases} \tag{1.1}$$

As condições (1.1) são chamadas de Condições de Karush-Kuhn-Tucker (KKT).

Teorema 1.6 ***(Condições de Otimalidade - KKT)*** *Seja $x^\star \in \Omega$ um minimizador local do problema (P) e suponha que $(T(x^\star))^\circ = (D(x^\star))^\circ$, então $x^\star$ é um ponto KKT.*
Demonstração: *ver [1, 22].*

A hipótese sobre os cones $T(x^\star)$ e $D(x^\star)$ feita no Teorema 1.6, é chamada de Condição de Qualificação. Esta é a condição mais fraca possível para se provar as condições de KKT em um minimizador. Entretanto, pode ser difícil obter os cones $T(x^\star)$ e $D(x^\star)$, e verificar se a condição $(T(x^\star))^\circ = (D(x^\star))^\circ$ é satisfeita. Devido essa dificuldade, existem outras Condições de Qualificação mais fáceis de serem verificadas na prática, e que implicam nesta condição.

O resultado abaixo é importante, pois garante que o problema com restrições lineares, o qual focamos neste trabalho, é qualificado e portanto, as condições de KKT se verificam em um minimizador.

Considere o problema:

$$(PRL) \begin{cases} min & f(x) \\ s.a. & l \leq Ax \leq u, \end{cases}$$

onde $f : \mathbb{R}^n \to \mathbb{R}, A \in \mathbb{R}^{m \times n}, l, u \in \mathbb{R}^m$.

Teorema 1.7 *Se $x^\star$ é um minimizador local do problema (PRL), então $x^\star$ satifaz as condições de KKT.*
Demonstração: *Ver [22].*

A Proposição 1.1 nos oferece outro critério de otimalidade, o qual será utilizado na demonstração de convergência para o Método de Busca Padrão para Minimização de funções com Restrições Lineares, que abordaremos no Capítulo 4.

Definição 1.9 ***(Gradiente Projetado)*** *Seja P_Ω a projeção no espaço viável Ω. Então para um ponto viável $x \in \Omega$ definimos:*

$$q(x) = P_\Omega(x - \nabla f(x)) - x$$

Proposição 1.1 *Seja $x \in \Omega$. Então*

$$\|q(x)\| \leq \|\nabla f(x)\|$$

Além disso temos que x é um ponto KKT para o problema (PRL) (quando as restrições do problema são lineares), se e somente se, $q(x) = 0$.

Este resultado é importante e pode ser encontrado em [1]. Ele nos fornece um critério para análise de otimalidade de um ponto viável. Utilizaremos esse resultado no desenvolvimento do método de busca padrão aplicado ao problema com restrições lineares, no Capítulo 4.

Capítulo 2

Métodos de Busca Direta

Dentre os métodos "derivative - free", como são conhecidos os métodos que não fazem uso de derivadas, os Métodos de Busca Direta são sem dúvida os mais simples e com aplicação quase que imediata. O termo foi introduzido em 1961 por R. Hooke e T.A. Jeeves [8]. Essa classe de métodos foi bastante estudada na década de 60, e amplamente utilizada. Com algoritmos simples, de baixo custo computacional, os métodos eram adequados para trabalhar com as limitações dos computadores na época.

Com o avanço tecnológico muitos métodos de busca direta cederam lugar a técnicas mais sofisticadas, e muitos usuários passaram a trabalhar com alguma variante dos métodos quase-Newton [1, 14], que passaram a funcionar muito bem com as novas ferramentas disponíveis. Porém ainda hoje, com todas as ferramentas computacionais que possuimos, os Métodos de Busca Direta continuam populares, pois são simples e em geral funcionam bem na prática, podendo ser utilizados para resolver quase todo tipo de problema de otimização não linear.

Em geral, os Métodos de Busca Direta obtêm sucesso quando rotinas mais elaboradas falham, uma vez que estas não são aplicáveis a todos os problemas de otimização não linear. Estes métodos evitam o uso de certas estratégias que podem falhar em métodos mais sofisticados. Além disso o uso de Métodos de Busca Direta exige muito pouco do usuário, pois são extremamente simples de serem implementados, uma vez que poucos parâmetros precisam ser definidos, não há a necessidade do cálculo das derivadas, e podem ser aplicados quase que imediatamente. Veja abaixo a caracterização dessa classe de métodos.

Definição 2.1 (Método de Busca Direta) *Dizemos que um método é de busca direta se, além de não calcular as derivadas, não utilizar os valores de função em nenhum cálculo.*

Como consequência desta definição, não é possível exigir condições de decréscimo suficiente para tais métodos, sendo permitido apenas utilizar o valor da função para fazer comparações e decidir qual entre dois pontos tem o menor valor de função objetivo. Como o valor da função não é utilizado em nenhum passo do algoritmo, não é necessário explicitá-lo de fato.

Existem muitos métodos que podem ser classificados como sendo Métodos de Busca Direta. Há vários trabalhos [4, 5, 10, 21] onde é possível encontrar a descrição de vários desses métodos. Aqui comentaremos rapidamente dois tipos, os Métodos Direcionais e os Métodos Simplex.

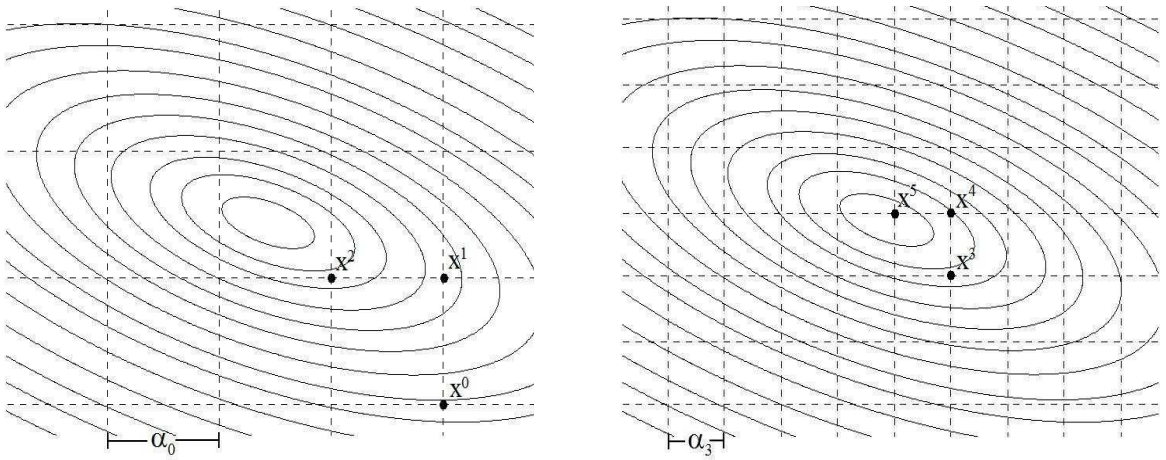
2.1 Métodos de Busca Direta Direcionais

Os Métodos de Busca Direta Direcionais sintetizam muitos dos conceitos da Otimização sem derivadas, pois possuem algoritmos simples com forte apelo geométrico, além de possuir uma robusta teoria de convergêcia.

Sem sombra de dúvidas os principais Métodos de Busca Direta Direcionais são os Métodos de Busca Padrão, classe em que se encaixa o método que estudamos. Sendo assim, apresentamos uma discussão detalhada sobre esses métodos no próximo capítulo.

Dentre os Métodos de Busca Direta Direcionais vale destacar o Método de Busca Coordenada e o método introduzido por Hooke e Jeeves em 1961, os quais também são métodos de Busca Padrão.

2.1.1 Método de Busca Coordenada



(a) $\alpha = \alpha_0$ (b) $\alpha = \frac{\alpha_0}{2}$

Figura 2.1: Método de Busca Coordenanda

Um dos métodos mais simples encontrados na literatura é o Método de Busca Coordenada para minimização irrestrita, que recebe esse nome pois a busca por um iterando que forneça um valor de função objetivo melhor é feita nas direções coordenadas a partir do ponto corrente. Ou seja, dado $\alpha_k \in \mathbb{R},\ \alpha_k > 0$, a partir do ponto corrente x^k buscamos um novo ponto da forma $x^{k+1} = x^k + \alpha_k(\pm e_i)$ tal que $f(x^{k+1}) < f(x^k)$, onde $(\pm e_i) \in [I - I]$, sendo I a matriz identidade de ordem n. Dessa forma buscamos nas direções canônicas positivas e negativas um ponto que possua um menor valor de função objetivo que o ponto atual.

Caso não encontremos um ponto melhor, então diminuímos o tamanho do passo α_k, em geral fazemos $\alpha_{k+1} = \frac{\alpha_k}{2}$, e a busca é refeita a partir do mesmo ponto. Veja a Figura 2.1.

2.1.2 Método de Hooke e Jeeves

Além de introduzir o termo busca direta em [8], os autores apresentaram um algoritmo desta classe. Achamos importante apresentar as principais características do método, o qual não possui apenas importância histórica, sendo também bastante relevante na prática e muito utilizado até hoje.

O método é bastante semelhante à busca coordenada, porém utiliza uma estratégia com o intuito de acelerar a convergência. Em caso de sucesso na iteração k, na próxima iteração o algoritmo faz a busca coordenada ao redor do ponto $x^{k+1} + (x^{k+1} - x^k)$, ao invés de realizar a busca ao redor do ponto x^{k+1}. Esta estratégia é baseada no fato de que a direção $(x^{k+1} - x^k)$ é potencialmente uma direção de descida, uma vez que foi a direção que ofereceu decréscimo na iteração anterior. Portanto, se obtivermos sucesso, ao realizar a busca ao redor deste ponto, "economizamos uma iteração"; caso essa busca fracasse, refazemos a busca coordenada ao redor do ponto x^{k+1}. Caso ainda assim o sucesso não seja obtido, diminuímos o tamanho do passo e refazemos a busca ao redor do ponto x^{k+1}.

2.2 Métodos Simplex

Os métodos de busca baseados em simplex são caracterizados por um conjunto de pontos utilizados para guiar a busca. Primeiramente veja abaixo a definição de um simplex, o qual não pode ser confundido com o método Simplex de programação linear.

Definição 2.2 (Simplex) *Um simplex é um conjunto de $n+1$ pontos em $\mathbb{R}^n$. Um simplex definido pelos pontos $x^1, x^2, \ldots, x^n, x^{n+1}$ (chamados de vértices do simplex) é dito não degenerado quando o conjunto $\{x^1 - x^{n+1}, \ldots, x^n - x^{n+1}\}$ é linearmente independente.*

O primeiro Método Simplex foi proposto por Spendley, Hext e Himsworth [23] em 1962, o qual consiste em ordenar os vértices do simplex (não degenerado) considerando os valores da função objetivo nos pontos. Então substituímos o pior vértice, rotacionando os vértices do simplex ou, como é mais comum, o pior vértice é refletido através do centróide da face oposta, como mostra a Figura 2.2.

O Método de Spendley et al tem importância histórica, mas o Método Simplex mais conhecido e muito utilizado até hoje é o Método de Nelder e Mead [18]. Este método, embora não possua garantias teóricas de convergência, funciona muito bem na prática e, por esse motivo, é largamente empregado.

O método proposto por Nelder e Mead visa otimizar a busca simplex adicionando movimentos para acelerar a busca. A alternativa para o movimento de reflexão do método de Spendley et al é permitir a expansão e contração do simplex, que assim pode ser deformado para melhor se adaptar às características da função objetivo.

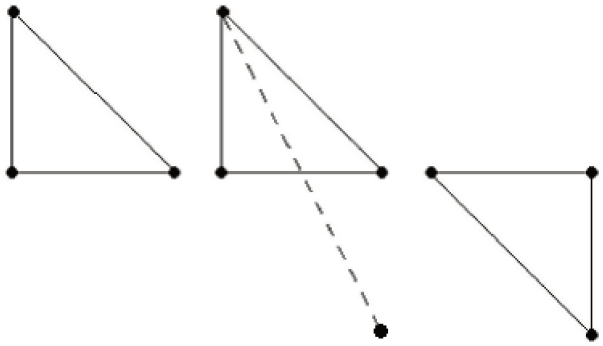
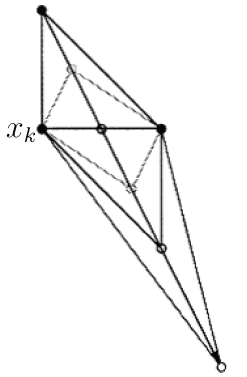
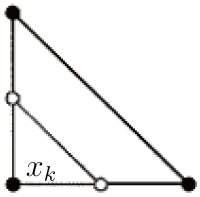


Figura 2.2: Reflexão de um ponto pelo centróide da face oposta



(a) Reflexão com expansão ou contração do Simplex



(b) Redução Simplex

Figura 2.3: Movimentos do Método de Nelder-Mead

Assim, se após o movimento de reflexão, o novo vértice do simplex possuir valor de função objetivo menor que o melhor vértice da iteração anterior, então experimentamos a expansão do simplex, com o intuito de melhorar ainda mais. Já se o novo valor obtido for menor que o do pior vértice, mas não superar o melhor deles, tentamos a contração do simplex buscando encontrar um ponto melhor. Porém se o novo vértice possuir valor da função objetivo maior que o pior vértice da iteração anterior, então o simplex é reduzido, mantendo apenas o melhor vértice. Veja a figura 2.3 que ilustra os movimentos de expansão, contração e redução do simplex.

O algoritmo e uma descrição mais detalhada do Método de Nelder-Mead podem ser encontrados em [4, 5, 18, 21]. Vale destacar que, o Método de Nelder-Mead, embora muito popular e amplamente empregado em problemas práticos, não possui garantias de convergência para dimensões superiores a 2. Existem exemplos simples em que o método converge a pontos não estacionários. Sendo assim não é possível desenvolver uma teoria de convergência para este método. Porém o mau comportamento do método pode ser evitado impondo algumas condições sobre o simplex, como em [25]. No entanto, ainda que o método falhe em muitos casos, quando converge, em geral funciona muito bem, e este é o motivo para ser tão popular.

Capítulo 3

Métodos de Busca Padrão

Caracterizados como sendo métodos de busca direta direcional, os métodos de busca padrão sintetizam muitos dos conceitos de otimização sem derivadas, possuem algoritmos simples, de forte apelo geométrico, e com aplicação quase que imediata, além de possuir uma robusta teoria de convergência.

Em [24], Virginia Torczon generaliza o algoritmo dessa classe de métodos, aplicado ao problema de minimização irrestrito, englobando muitos dos métodos existentes, como o conhecido Método de Busca Coordenada, um dos mais simples da literatura. Neste trabalho a autora também apresenta uma robusta teoria de convergência, sendo o primeiro resultado geral de convergência para essa classe de métodos. Dessa forma todos os algoritmos classificados como de busca Padrão passaram a ter garantias teóricas de convergência.

A maioria dos Métodos de Busca Padrão utiliza busca monótona, ou seja, a partir de um ponto corrente x^k, buscamos em um conjunto determinado de direções P^k, chamado de padrão, com tamanho de passo fixo, um novo iterando x^{k+1} que satisfaça $f(x^{k+1}) < f(x^k)$. É exigido apenas decréscimo simples no valor da função objetivo, não impomos nenhuma condição de decréscimo suficiente.

Perceba também que além de não computar as derivadas da função objetivo, o valor da função só é utilizado para fazer comparações, não sendo utilizado em nenhum cálculo, o que caracteriza este tipo de método como sendo método de busca direta.

3.1 Direções de Busca

Nesta seção trataremos dos aspectos teóricos envolvidos na construção do padrão de direções P^k para o caso irrestrito.

Na maioria dos métodos que utilizam busca monótona, a escolha das direções de busca é determinante para obter um bom desempenho para o método. Neste contexto o uso do gradiente da função objetivo tem papel importante, pois permite verificar se uma determinada direção é ou não de descida, além do fato de a própria direção de $-\nabla f(x)$ ser uma direção de descida a ser tomada.

Logo, ao abandonar o uso de derivadas, perdemos essa ferramenta importante e não podemos mais saber se uma determinada direção é ou não de descida. Para contornar esse problema, o Método de Busca Padrão considera, a cada iteração $k = 1, 2, \ldots$, um conjunto de direções de busca P^k que gere positivamente o $\mathbb{R}^n$, ou seja, qualquer elemento $x \in \mathbb{R}^n$ pode ser escrito como combinação linear das direções em P^k apenas com coeficientes positivos. Assim é possível garantir que, se o ponto corrente não é estacionário, ou seja, $\nabla f(x^k) \neq 0$, então ao menos uma das direções do conjunto de busca é de descida. Esta é a chave para o resultado de convergência do método.

Para definir o padrão de direções que gere positivamente o $\mathbb{R}^n$, precisamos de duas componentes, a matriz base, que denotaremos por B, e a matriz geradora, que denotaremos por C_k.

A matriz base pode ser qualquer matriz não singular $B \in \mathbb{R}^{n \times n}$, ou seja, uma base qualquer para $\mathbb{R}^n$.

A matriz geradora $C_k \in \mathbb{Z}^{n \times p}$, onde $p > 2n$, é particionada da seguinte forma:

$$C_k = [M_k \ \ -M_k \ \ L_k] = [\Gamma_k \ \ L_k]\,,$$

onde $M_k \in \mathbf{M} \subset \mathbb{Z}^{n \times n}$, $\mathbf{M}$ é um conjunto finito de matrizes não singulares e a matriz $L_k \in \mathbb{Z}^{n \times (p-2n)}$ tem a finalidade de deixar o algoritmo mais geral e abrangente, e contém ao menos uma coluna, a coluna de zeros.

Então o padrão de direções P^k é definido como:

$$P^k = BC_k = [BM_k \ \ -BM_k \ BL_k] = [B\Gamma_k \ BL_k]\,.$$

Os passos de busca do algoritmo são dados nas direções de P^k, com tamanho de passo $\alpha_k > 0$; o tamanho do passo é atualizado de acordo com o status da iteração.

As formas de atualização do padrão e do tamanho do passo variam de um método para outro, desde que seguindo certos critérios que garantem a convergência, os quais comentaremos mais adiante. E é justamente nas diferentes estratégias de atualização e no padrão de busca utilizados que se originam os diferentes Métodos de Busca Padrão.

Exemplo 3.1 *Vejamos que o Método de Busca Coordenada é um Método de Busca Padrão:*

- *as direções de busca do método são as direções coordenadas. Para $n = 2$ as direções de busca seriam:*

$$P = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix};$$

- *neste método as direções de busca não mudam de uma iteração para outra, ou seja, $P^k = P$, $\forall k$. Assim, temos que as matrizes que geram o padrão são dadas por:*

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \ \ M_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix};$$

- *perceba que para o Método de Busca Coordenada, como o padrão é fixo, a matriz M_k também não muda de uma iteração para outra;*

- *neste exemplo, o padrão não tem direções adicionais. Sendo assim, temos que a matriz L_k contém apenas a coluna de zeros. Do ponto de vista teórico, a coluna de zeros representa os passos em que houve fracasso. Quando a busca fracassa, permanecemos no mesmo ponto, o que é equivalente a dar o passo na direção nula.*

3.2 Movimentos Exploratórios

Sabemos que o método utiliza busca monótona a partir do ponto corrente, procurando um novo ponto que ofereça decréscimo no valor da função objetivo. Porém, para que os resultados de convergência sejam garantidos, existem certos critérios que devem ser satisfeitos ao realizarmos a busca, o que chamamos de hipóteses sobre os movimentos exploratórios. Os resultados de convergência serão tratados com detalhes na seção 3.4.

A iteração será declarada bem sucedida se encontrarmos um novo ponto da forma $x^{k+1} = x^k + \alpha_k c_i^k$, no qual o valor da função objetivo é menor que em x^k. Neste caso o tamanho do passo pode ser mantido ou mesmo aumentar. Já o fracasso é declarado se todas as direções $c_i^k \in P^k$ forem testadas e não encontrarmos um novo ponto que ofereça decréscimo no valor da função objetivo; neste caso o tamanho do passo α_k deve necessariamente diminuir e a busca é refeita com o novo tamanho de passo.

Nenhuma condição de decréscimo suficiente é exigida, é necessário apenas que o passo aceito forneça um decréscimo simples no valor da função objetivo e seja da forma $s_i^k = \alpha_k c_i^k$, ou seja, o tamanho do passo é fixo na iteração corrente e as direções devem pertencer ao padrão P^k, conforme hipótese 1 abaixo. Além disso, antes que o fracasso seja declarado, todas as direções do padrão devem ser testadas e somente se nenhuma delas oferecer decréscimo à função objetivo o fracasso é declarado, conforme a hipótese 2.

Hipóteses fracas sobre os movimentos exploratórios

1. $s^k \in \alpha_k P^k$

2. Se $min\left\{f(x^k + y) : \ \ y \in \alpha_k P^k\right\} < f(x^k)$, então:

$$f(x^k + s^k) < f(x^k).$$

As hipóteses acima, juntamente com a hipótese de que o conjunto de nível de f seja compacto, garantem que a sequência gerada pelo algoritmo possui uma subsequência que converge para um ponto estacionário do problema, ou seja:

$$\liminf_{k\to\infty} \left\|\nabla f(x^k)\right\| = 0.$$

Alterando a segunda hipótese por uma mais restritiva podemos obter um resultado de convergência mais forte, garantindo que toda subsequência convergente gerada pelo algoritmo converge para um ponto estacionário, ou seja:

$$\lim_{k\to\infty} \left\|\nabla f(x^k)\right\| = 0.$$

Para obter esse resultado, ao invés de tomarmos qualquer uma das direções que ofereça decréscimo para o valor da função objetivo, exigimos que a direção tomada ofereça o maior decréscimo possível entre as direções contidas no padrão. A principal mudança no ponto de vista prático é que, em toda iteração, devemos necessariamente testar todas as direções do padrão para procurar aquela que oferece o maior decréscimo, enquanto que pela hipótese fraca, assim que obtemos um ponto melhor, podemos encerrar a iteração e aceitar o passo. Então a versão mais forte das hipóteses sobre os movimentos exploratórios é dada por:

Hipóteses fortes sobre os movimentos exploratórios

1. $s^k \in \alpha_k P^k$
2. Se $min\left\{f(x^k + y) : \ \ y \in \alpha_k P^k\right\} < f(x^k)$, então:

$$f(x^k + s^k) \leq min\left\{f(x^k + y) : \ \ y \in \alpha_k P^k\right\}.$$

3.3 Algoritmo Geral

Segue abaixo o algoritmo geral para os Métodos de Busca Padrão para minimização irrestrita. Os algoritmos particulares são obtidos ao especificarmos as matrizes que geram o padrão, ou seja, a base B e as matrizes geradoras C_k, além das estratégias para realizar a busca e realizar as atualizações, desde que mantendo as características gerais que garantem a convergência.

Algoritmo 3.1 - *Busca Padrão irrestrito*

1. *Dados* $x^0 \in \mathbb{R}^n,\ \alpha_0 \in \mathbb{R},\ \alpha_0 > 0$*;*

   Para $k = 0, 1, ...$

2. *Compute* $f(x^k)$.
3. *Se* $f(x^k + \alpha_k c^k) < f(x^k)$ *para algum* $c^k \in P^k$*, conforme as hipóteses sobre os movimentos exploratórios, então declare sucesso e faça* $x^{k+1} = x^k + \alpha_k c^k$*;*
   caso contrário declare fracasso e faça $x^{k+1} = x^k$.
4. *Atualize* α_k *e* P^k.

Perceba que a cada iteração do algoritmo 3.1 é muito simples e barata, sendo o passo mais caro o de realizar a busca, pois envolve a avaliação do valor da função em muitos pontos. Além disso, em problemas em que as derivadas não estão disponíveis, o custo de avaliar a função costuma ser alto. Por esses motivos, ao compararmos o desempenho de diferentes Métodos de Busca Padrão, focamos nossa atenção no número de avaliações de função realizadas pelo método para obter a convergência.

Neste sentido, ao pedir que sejam satisfeitas as hipóteses fortes sobre os movimentos exploratórios, o custo da iteração aumenta significativamente, uma vez que testamos todas as direções do padrão antes de finalizar a busca. Dessa forma o número de avaliações de função por iteração pode ser muito grande, dependendo das dimensões do problema. Isso inviabiliza o uso do método para problemas maiores, uma vez que realizamos no mínimo $2n$ avaliações de função por iteração, considerando que não estamos necessariamente testando as direções determinadas pela matriz L_k.

Deste modo, embora ofereça um resultado de convergência mais forte, realizar busca completa nem sempre oferece o melhor desempenho para o método. Em geral, com busca completa realizamos um número menor de iterações para obter convergência, no entanto o custo da iteração é bem maior. Na maioria das vezes é possível observar a convergência do método, mesmo com busca simples, fazendo com que ele funcione melhor em muitos casos.

3.3.1 Atualizações

As atualizações do padrão e do tamanho do passo variam de método para método, porém essas atualizações devem satisfazer alguns critérios, para um método ser classificado como busca padrão.

No caso do padrão de direções P^k é necessário que este seja sempre da forma $P^k = BC_k$, conforme descrito na primeira seção deste capítulo. Portanto, desde que todos os padrões utilizados possam ser escritos dessa forma, e contenham uma base que gere positivamente o $\mathbb{R}^n$, os critérios de atualização são livres. Inclusive podem ser mantidos fixos em todas as iterações, como acontece com o método de busca coordenada, por exemplo.

Já o tamanho do passo é atualizado de acordo com o status da iteração, podendo aumentar ou ser mantido em caso de sucesso e devendo necessariamente diminuir em caso de fracasso. Os critérios de atualização no entanto não são livres e devem satisfazer as condições descritas abaixo:

Atualização do tamanho do passo
Dado $\alpha_k > 0$, o tamanho passo α_{k+1} na iteração seguinte será:

- Em caso de sucesso fazemos $\alpha_{k+1} = \lambda_k \alpha_k$, onde $\lambda_k \geq 1$ e $|\lambda_k| < +\infty$.
- Em caso de fracasso fazemos $\alpha_{k+1} = \theta_k \alpha_k$, onde $\theta_k \in (0, 1)$.

Além disso os parâmetros λ_k e θ_k devem ser da forma:

- Sejam $\tau \in \mathbb{Q},\ \tau > 1$, e $\{w_0, ..., w_L\} \subset \mathbb{Z}$, ordenados com $w_0 < 0,\ w_L \geq 0$. Então:
- $\theta_k = \tau^{w_i}$ para algum $w_i \in \{w_0, ..., w_L\}$ tal que $w_i < 0$;
- $\lambda_k = \tau^{w_j}$ para algum $w_j \in \{w_0, ..., w_L\}$ tal que $w_j \geq 0$.;

3.3.2 Critério de Parada

Pela estrutura do algoritmo é possível perceber que, embora estejamos interessados em iterações bem sucedidas, à medida que as iterações se aproximam do minimizador, o tamanho do passo deve ficar cada vez menor. Isso acontece porque ao atingir o minimizador não será possível encontrar um ponto que melhore o valor da função objetivo, sendo assim o tamanho do passo deve tender a zero.

Pelas considerações anteriores, é intuitivo tomar como critério de parada para o algoritmo uma tolerância para o tamanho do passo, uma vez que o tamanho do passo muito pequeno pode indicar que estamos próximos à solução.

Em [24], a autora não fala a respeito de critérios de parada para o método. Entretanto na prática usamos esta estratégia para finalizar o algoritmo, além de limitar o número máximo de iterações ou avaliações da função.

3.4 Análise de Convergência do Método

Tendo definido as principais características dos Métodos de Busca Padrão, vamos exibir alguns resultados e comentários a respeito da convergência do método. Todos os resultados e suas demonstrações estão detalhados em [24].

São apresentados dois resultados principais de convergência, como comentamos anteriormente. A chave para a demonstração destes resultados está no fato de que os pontos visitados pelo algoritmo de Busca Padrão estão em malhas racionais, conforme detalhado em [24]. Além disso é exigido que o conjunto de nível $L(x^0) = \{x \in \mathbb{R}^n$ tal que $f(x) \leq f(x^0)\}$ seja compacto. Esta hipótese é importante pois, como todos os pontos visitados pelo algoritmo pertencem a $L(x^0)$, e pelo fato da sequência ser monótona decrescente, sabemos que a sequência gerada pelo algoritmo possui ao menos um ponto de acumulação.

Outro ponto importante a ser destacado no estudo de convergência é que, apesar de se tratar de um método sem derivadas, é necessário que a função objetivo seja continuamente diferenciável em uma vizinhança de $L(x^0)$ para obtermos o resultado de convergência global. Sendo assim o método não possui garantias de convergência para funções não diferenciáveis. Entretanto, isso não impede seu uso para resolver esse tipo de problema, uma vez que o algoritmo em si não faz uso das derivadas da função objetivo.

Seguem abaixo dois resultados necessários para a demonstração do teorema, que garante que a sequência gerada pelo algoritmo possui ao menos uma subsequência convergente.

Teorema 3.1 *Assuma que $L(x^0)$ é compacto. Então:*

$$\liminf_{k\to\infty} \alpha_k = 0.$$

Demonstração: *Ver [24, Teorema 3.3].*

Proposição 3.1 *Assuma que $L(x^0)$ é compacto, f é continuamente diferenciável em uma vizinhança de $L(x^0)$, e $\liminf_{k\to\infty} \left\|\nabla f(x^k)\right\| \neq 0$. Então existe uma constante α_{LI} tal que, para todo k temos que $\alpha_k \geq \alpha_{LI}$.*
Demonstração: *Ver [24, Proposição 3.4].*

Teorema 3.2 *Seja $\left\{x^k\right\}$ uma sequência gerada pelo algoritmo de Busca Padrão. Suponha que $L(x^0)$ é compacto e f é continuamente diferenciável em uma vizinhança de $L(x^0)$. Então:*

$$\liminf_{k\to\infty} \left\|\nabla f(x^k)\right\| = 0.$$

Demonstração: *Ver [24, Teorema 3.5].*

Esse resultado nos diz que toda sequência $\left\{x^k\right\}$ gerada pelo algoritmo de Busca Padrão possui ao menos uma subsequência que converge a um ponto estacionário. Considerando a estrutura do algoritmo, com iterandos cada vez mais próximos entre si, uma vez que $\alpha^k \to 0$, é improvável que a sequência fique alternando entre diversos pontos, possuindo diferentes pontos de acumulação para suas diversas subsequências. Porém esse resultado nos garante que, caso isso ocorra, ao menos um dos pontos de acumulação é um ponto estacionário, e a existência desse ponto de acumulação é garantida pela hipótese de que $L(x^0)$ é compacto.

O resultado acima é obtido se forem satisfeitas apenas as hipóteses fracas sobre os movimentos exploratórios. Acrescentando algumas hipóteses extras e pedindo que o algoritmo satisfaça as hipóteses fortes sobre os movimentos exploratórios, é possível demonstrar um resultado de convergência mais forte, que garante que toda subsequência convergente gerada pelo algoritmo converge para um ponto estacionário. O resultado é o seguinte:

Teorema 3.3 *Seja $\left\{x^k\right\}$ uma sequência gerada pelo algoritmo de Busca Padrão. Suponha que $L(x^0)$ é compacto e f é diferenciável numa vizinhança de $L(x^0)$. Além disso, suponha que $\lim_{k\to\infty} \alpha^k = 0$, que o algoritmo satisfaz as hipóteses fortes sobre os movimentos exploratórios, e as colunas das matrizes geradoras C_k são limitadas em norma. Então:*

$$\lim_{k\to\infty} \left\|\nabla f(x^k)\right\| = 0.$$

Demonstração: *Ver [24, Teorema 3.7].*

Capítulo 4

Método de Busca Padrão para Minimização com Restrições Lineares

Neste capítulo, seguindo a abordagem proposta por Virginia Torczon e Robert Michael Lewis em [11], estendemos o Método de Busca Padrão aplicado ao problema de minimização com restrições lineares definido por:

$$(PRL)\begin{cases} min & f(x) \\ s.a & l \leq Ax \leq u, \end{cases}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$, $x \in \mathbb{R}^n$, $A \in \mathbb{Q}^{m\times n}$, $l, u \in \mathbb{R}^m$. A exigência de que as entradas da matriz de restrições sejam racionais é necessária para a demonstração de convergência do método. Na prática, testes computacionais mostraram que essa hipótese não precisa ser satisfeita para o método convergir.

Assim como no Método de Busca Padrão para minimização irrestrita, ao estender o método para o problema com restrições lineares, sob algumas hipóteses, é possível demonstrar dois resultados de convergência. Um que garante a existência de uma subsequência gerada pelo algoritmo que converge para um ponto KKT do problema e outro, mais forte, no qual toda subsequência convergente gerada pelo algoritmo converge a um ponto KKT do problema. Vale destacar que, sob a hipótese de que o conjunto de nível de f é compacto, o método gera ao menos uma subsequência que converge para o minimizador restrito do problema, mas não realizamos nenhuma estimativa para os multiplicadores de Lagrange associados à solução.

A ideia geral do método segue os padrões que definimos no caso irrestrito; a grande diferença é que o ponto inicial deve ser factível, e a cada iteração a viabilidade não pode ser perdida. Assim, ao aceitar um novo iterando, além de pedir que o novo ponto possua o valor da função objetivo menor do que no iterando anterior, o ponto deve ser viável. Dessa forma, ao construir o padrão de direções de busca devemos levar em consideração a geometria da fronteira, de forma a ter a possibilidade de dar passos suficientemente longos.

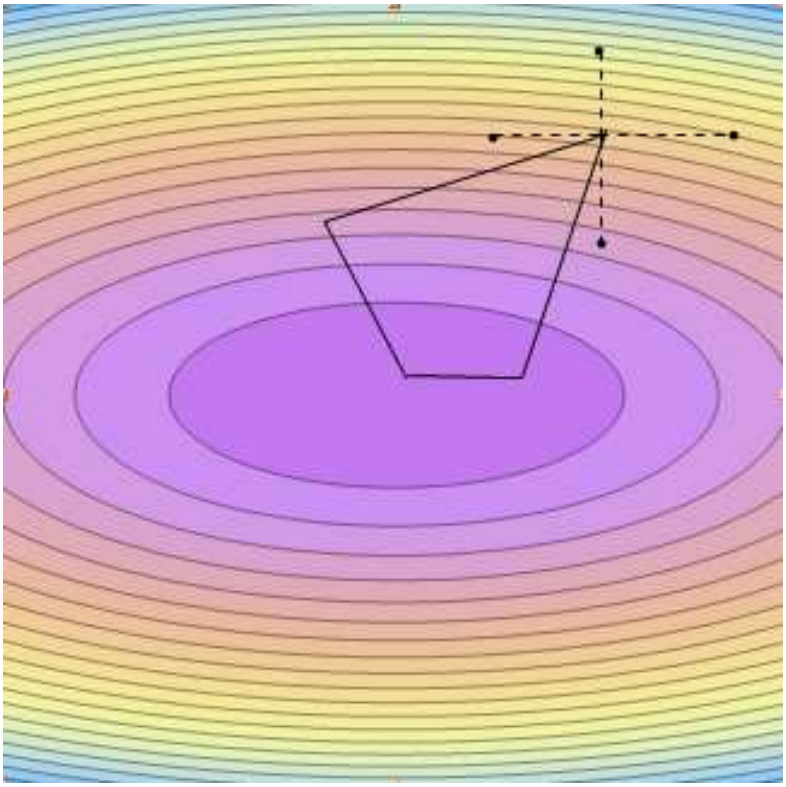


Figura 4.1: Direções fora do conjunto viável

4.1 O Padrão de Direções

Para que o método funcione bem, a escolha do padrão de direções adequadas é essencial. Quando aplicamos o método ao problema irrestrito, a principal característica do padrão é que as direções de busca gerem positivamente o $\mathbb{R}^n$. Já para o caso restrito, a escolha do padrão é menos flexível, como podemos observar pela Figura 4.1.

Para o problema com restrições lineares é preciso considerar informações específicas do problema, uma vez que, quando estamos próximos da fronteira da região viável, o padrão de pontos deve seguir sua geometria. A ideia geral neste caso é que o padrão de direções de busca deverá conter um conjunto de geradores positivos para o cone de direções viáveis. Como o padrão deve seguir a fronteira da região viável, é natural que modifiquemos as direções de busca a cada iteração, o que é feito de acordo com as restrições ϵ-ativas no ponto corrente.

Vimos que, para o caso irrestrito, o padrão de direções é da forma $P^k = BC_k$, onde a matriz B é uma base para $\mathbb{R}^n$, e a matriz $C_k \in \mathbb{Z}^{n\times p}$ é da forma $C_k = [\Gamma_k \ \ L_k]$. Para o problema com restrições lineares, como devemos incluir informações das restrições no padrão, por simplicidade tomaremos $B = I$. Dessa forma trabalharemos com as restrições diretamente no padrão P^k, mais especificamente as informações das restrições estarão na matriz Γ_k, e a matriz L_k continua com o papel de deixar o algoritmo mais geral e deve conter ao menos uma coluna.

Outra característica teórica importante de se observar é que pedimos que as componentes das direções no padrão sejam inteiras, o que é necessário para se obter o resultado de convergência para o método. Porém, do ponto de vista prático, como veremos mais adiante, conseguimos observar bons resultados de convergência utilizando direções de busca não inteiras.

4.1.1 Restrições Geométricas no Padrão

No problema de minimização com Restrições Lineares o núcleo do padrão Γ_k deve refletir a geometria da região viável quando o iterando encontra-se próximo à fronteira da região. Como já foi dito, os Métodos de Busca Padrão não realizam nenhuma aproximação para o gradiente da função objetivo. Por outro lado, é preciso garantir que, se não estamos em um ponto estacionário, as direções do padrão nos permitam sair do ponto corrente, diminuindo o valor da função objetivo. Além disso é necessário garantir que seja possível dar passos suficientemente longos sem deixar a região factível. Vamos discutir agora as condições geométricas no padrão que tornam isso possível na presença de restrições lineares.

Seja a_i^t a i-ésima linha da matriz A, que define as restrições do problema, então definimos:

$$Al_i = \left\{ x \,\middle|\, a_i^t x = l_i \right\}$$

$$Au_i = \left\{ x \,\middle|\, a_i^t x = u_i \right\}$$

Esses conjuntos definem as fronteiras dos semi-espaços cuja intersecção define o espaço viável $\Omega = \{x \in \mathbb{R}^n \mid l \leq Ax \leq u\}$. Para incluir no padrão precisamos conhecer também a região próximo à fronteira, logo definimos os seguintes conjuntos:

$$\partial\Omega_{l_i}(\epsilon) = \{x \in \Omega \,|dist(x, Al_i) \leq \epsilon\}$$

$$\partial\Omega_{u_i}(\epsilon) = \{x \in \Omega \,|dist(x, Au_i) \leq \epsilon\}$$

e

$$\partial\Omega(\epsilon) = \bigcup_{i=1}^{m} \partial\Omega_{l_i}(\epsilon) \cup \partial\Omega_{u_i}(\epsilon)$$

Perceba que o conjunto $\partial\Omega(\epsilon)$ contém todos os pontos factíveis que estão à uma distância menor do que ϵ da fronteira da região. Assim, dado um iterando x_k, é interessante saber a localização do ponto em relação à fronteira da região de busca. Ao definir esse conjunto, ϵ representa a menor distância que desejamos andar em uma determinada iteração, e portanto devemos considerar essa informação ao gerar o padrão de direções.

Considere os conjuntos abaixo, chamados de conjuntos de índices.

Dado $x \in \Omega$ e $\epsilon > 0$, considere

$$\mathbb{D}_l(x, \epsilon) = \{ i \,|x \in \partial\Omega_{l_i}(\epsilon) \} ,$$

e

$$\mathbb{D}_u(x, \epsilon) = \{ i \,|x \in \partial\Omega_{u_i}(\epsilon) \} .$$

Se $i \in \mathbb{D}_l(x, \epsilon) \cup \mathbb{D}_u(x, \epsilon)$, dizemos que a restrição i é ϵ-ativa no ponto x. Definidas as restrições ϵ-ativas no ponto corrente podemos construir o cone de direções factíveis a partir deste ponto.

Para $i \in \mathbb{D}_l(x, \epsilon)$ defina:

$$v_{l_i}(x, \epsilon) = -a_i.$$

Para $i \in \mathbb{D}_u(x, \epsilon)$ defina:

$$v_{u_i}(x, \epsilon) = a_i.$$

Perceba que esses vetores assim definidos apontam sempre para fora da região viável, e são ortogonais às faces. Então, dado $x \in \Omega$, definimos o cone $K(x, \epsilon)$ como o cone gerado pelos vetores $v_{l_i}(x, \epsilon)$ e $v_{u_i}(x, \epsilon)$. Como o cone $K(x, \epsilon)$ aponta para fora da região viável tudo indica que se andarmos por direções no polar do cone $K(x, \epsilon)$ permaneceremos dentro do conjunto viável, desde que o tamanho do passo não ultrapasse o valor de ϵ. Isso é mostrado na Figura 4.2, onde denotamos por $K^o(x, \epsilon)$ o polar do cone $K(x, \epsilon)$.

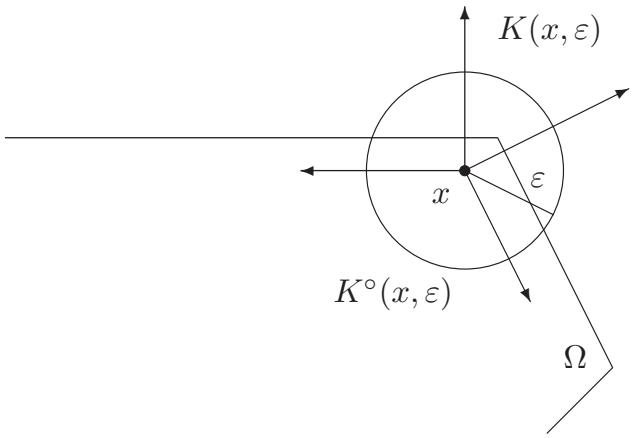


Figura 4.2: Geometria próximo à fronteira

Portanto, o padrão de direções deverá conter um conjunto de direções que contenha um gerador positivo para o cone $K^o(x, \epsilon)$. Faz parte do nosso trabalho estudar diferentes formas de construir esse conjunto gerador e analisar qual estratégia funciona melhor na prática. Em geral, para construir o padrão, precisamos considerar o tipo de problema com o qual estamos trabalhando e criar estratégias específicas para cada problema. No próximo capítulo faremos isso com detalhes.

4.2 Movimentos Exploratórios e Atualizações

Os movimentos exploratórios para o Método aplicado ao problema com restrições lineares é bastante semelhante à estratégia adotada para o caso irrestrito, poucas alterações são necessárias. O parâmetro α_k que determina o quanto iremos andar nas direções do padrão deve satisfazer os mesmos critérios já discutidos no capítulo anterior e, assim como para o caso irrestrito, nos fornecerá o critério de parada para o algoritmo.

No passo de busca podemos adotar as mesmas hipóteses que já discutimos para o caso irrestrito, ou seja, o algoritmo pode satisfazer as hipóteses fracas ou fortes sob os movimentos exploratórios. O que muda é que, na presença de restrições, temos que nos preocupar em manter a viabilidade do problema. Para aplicar o método temos que fornecer um ponto inicial viável, e a cada iteração a viabilidade deve ser mantida. Assim, ao determinar um ponto a ser testado, antes de calcular o valor da função neste ponto, testamos se a viabilidade não será perdida. Dessa forma, só calculamos o valor da função se o ponto puder ser aproveitado.

4.3 Algoritmo Geral

O algoritmo 4.1 abaixo contém os principais passos do Método de Busca Padrão aplicado ao problema de minimização com restrições lineares (PRL). Os passos de busca e atualização do tamanho do passo serão tratados com detalhes no capítulo 6 e a escolha do padrão de direções será abordada no capítulo 5.

Algoritmo 4.1 ***Busca Padrão - Restrições Lineares***

Dados $x^0 \in \Omega$, $\alpha_0 \in \mathbb{R}$, $\alpha_0 > 0$.
Inicialização: construa o padrão $P^0 \in \mathbb{R}^{n \times p}$.
Para $k = 0, 1, ...,$

1. *Calcule* $f(x^k)$.

2. *Performe uma busca, de acordo com o Algoritmo 6.1 (seção 6.2, capítulo 6), para obter* x^{k+1} *tal que:*

   (i) $x^{k+1} \in \Omega$, *e*
   (ii) $f(x^{k+1}) < f(x^k)$.

   Se (i) ou (ii) não ocorrer, faça $x^{k+1} = x^k$ *e declare fracasso.*

3. *Atualize* α_k *de acordo com o algoritmo 6.2 (seção 6.3, capítulo 6).*

4. *Determine as restrições* ϵ*-ativas e atualize o conjunto de trabalho, como discutido no capítulo 5.*

5. *Atualize* P^k *de acordo com o algoritmo 5.1 (seção 5.4.1, capítulo 5), ou de acordo com o algoritmo 5.2 (seção 5.4.2, capítulo 5).*

4.3.1 Critério de Parada

Como já comentamos, adotaremos como critério de parada para o algoritmo uma tolerância para o tamanho do parâmetro α_k, que determina o tamanho do passo dado pelo algoritmo. É bastante intuitivo que seja adotado esse critério para parar o algoritmo, uma vez que, ao nos aproximarmos da solução, os passos dados pelo algortimo tendem a ficar cada vez menores.

Mais do que simplesmente intuitivo, existem resultados teóricos que garantem que este é um bom critério de parada e que de fato estamos próximos à solução do problema na presença de restrições lineares.

Proposição 4.1 *Suponha que ∇f é Lipschitz contínuo em $L_\Omega(x^0)$. Se x^k é o ponto obtido numa iteração do algoritmo 4.1, onde houve fracasso, então existe uma constante $\hat{c} > 0$ que satisfaz:*

$$\left\|q(x^k)\right\|^2 \leq \hat{c}\alpha_k,$$

onde $q(x^k)$ é o gradiente projetado dado pela definição 1.9.
Demonstração: *Ver [11, Proposição 7.1]*

A Proposição 4.1 nos indica que quando α_k tende a zero, a norma do gradiente projetado também tende a zero. Então, pela Proposição 1.1, teremos que x^k é um ponto KKT para o problema (PRL).

4.4 Análise de Convergência

Antes de apresentar os resultados de convergência do método estudado, vamos organizar uma série de hipóteses, já discutidas, que são necessárias para análise de convergência.

1. Características gerais do algoritmo e do padrão de direções:

   (a) o padrão $P^k = [\Gamma_k \;\; L_k] \in \mathbb{Z}^{n\times p_k}$, $p_k \geq n+1$;
   Logo todas as direções de busca devem ser vetores com entradas inteiras, escalados por $\alpha_k \in \mathbb{R}$, ou seja, todos os passos dados pelo algoritmo devem ser da forma $s_i^k = \alpha_k c_i^k$, onde c_i^k denota a coluna i do padrão P^k;

   (b) o núcleo do padrão, a matriz $\Gamma^k \in \mathbb{Z}^{n\times r_k}$, $r_k \geq n+1$, pertence à Γ, que é um conjunto finito de matrizes inteiras cujas colunas incluem geradores para todos os cones $K^o(x^k, \epsilon), 0 \leq \epsilon \leq \epsilon^\star$, para algum $\epsilon^\star$, que não depende de k;

   (c) a matriz $L^k \in \mathbb{Z}^{n\times p_k - r_k}$ contém pelo menos a coluna de zeros;

   (d) as atualizações do tamanho do passo α_k seguem as regras especificadas na seção 3.3.1;

(e) os movimentos exploratórios realizados pelo algoritmo retornam passos que satisfazem as condições abaixo:

i. $s^k \in \alpha_k P^k$;

ii. $(x^k + s^k) \in \Omega$;

iii. se $min\left\{ f(x^k + y) \,\middle|\, y \in \alpha_k P^k \; e \; (x^k + y) \in \Omega \right\} < f(x^k)$, então:
$$f(x^k + s^k) < f(x^k).$$

2. A matriz de restrições A é racional.

3. O conjunto de nível $L_\Omega(x^0) = \{x \in \Omega \,|\, f(x) \leq f(x^0)\}$ é compacto.

4. A função objetivo f é continuamente diferenciável em uma vizinhança aberta D de $L_\Omega(x^0)$.

Teorema 4.1 *Assuma que as hipóteses 1-4 são satisfeitas. Seja* $\left\{x^k\right\}$ *uma sequência gerada pelo algoritmo 4.1, então:*

$$\liminf_{k\to\infty} \left\| q(x^k) \right\| = 0.$$

Demonstração: *Ver [11, Teorema 4.2 (seção 6)].*

Como consequência imediata deste teorema segue o seguinte resultado:

Corolário 4.1 *Suponha que as hipóteses 1-4 são satisfeitas, então existe um ponto limite de* $\left\{x^k\right\}$ *que é um ponto KKT para o problema (PRL).*

Note que a Hipótese 3 nos garante a existência de tal ponto limite.

O Corolário 4.1 nos garante que existe uma subsequência da sequência gerada pelo método que converge a um ponto estacionário para o problema com restrições lineares. Porém, assim como fizemos no caso irrestrito, alterando algumas das hipóteses e adicionando outras, é possível obter um resultado mais forte, que garante que toda subsequência convergente gerada pelo algoritmo 4.1 converge a um ponto KKT para o problema. Logo, considere também as seguintes hipóteses:

5. As colunas da matriz P^k permanecem limitadas em norma, ou seja, existe uma constante $c > 0$ tal que para todo k, $\left\| c_i^k \right\| < c$, para todo $i = 1, \ldots, p_k$.

6. As condições sobre os movimentos exploratórios apresentados no item (e) devem ser substituídas por condições mais fortes dadas por:

(a) $s^k \in \alpha_k P^k$

(b) $(x^k + s^k) \in \Omega$.

(c) Se $min\left\{ f(x^k + y) \,\middle|\, y \in \alpha_k P^k \; e \; (x^k + y) \in \Omega \right\} < f(x^k)$, então:

$$f(x^k + s^k) \leq min\left\{ f(x^k + y) \,\middle|\, y \in \alpha_k P^k \; e \; (x^k + y) \in \Omega \right\}$$

7. Temos que: $\lim_{k\to\infty} \alpha_k = 0$

Perceba que a hipótese 6 requer a busca completa no padrão de direções antes de finalizar a iteração, o que é bem mais caro computacionalmente, tendo em vista que a hipótese exigida anteriormente permite finalizar a busca ao encontrarmos um passo que ofereça decréscimo no valor da função objetivo. Vale destacar, que na Hipótese 7 não exigimos que a sequência $\{\alpha_k\}$ seja monótona decrescente, exigimos apenas que no limite a sequência vá para zero, o que é natural do método, como vemos no teorema 4.2.

Teorema 4.2 *Suponha que sejam satisfeitas as hipóteses 1-4. Então temos que:*

$$\liminf_{k\to\infty} \alpha_k = 0.$$

Demonstração: *Ver [11, Teorema 6.5].*

Teorema 4.3 *Assuma que as hipóteses fortes 1-7 são satisfeitas. Seja* $\left\{x^k\right\}$ *uma sequência gerada pelo algoritmo 4.1, então:*

$$\lim_{k\to\infty} \left\| q(x^k) \right\| = 0.$$

Demonstração: *Ver [11, Teorema 4.4 (seção 6)].*

Então segue imediatamente o seguinte corolário:

Corolário 4.2 *Assuma que as hipóteses fortes 1-7 são satisfeitas. Então, todo ponto limite de* $\left\{x^k\right\}$ *é um ponto KKT para o problema (PRL).*

Novamente temos que a Hipótese 3 garante a existência de pelo menos um ponto limite para a sequência.

É importante ressaltar que, todas as hipóteses que assumimos aqui são necessárias para demonstrar os resultados de convergência apresentados. Logo, são importantes do ponto de vista teórico. Na prática, testes computacionais mostraram que mesmo que algumas hipóteses não sejam satisfeitas é possível observar um bom comportamento do método.

Capítulo 5

Gerando o Padrão de Direções de Busca

A escolha do conjunto de direções de busca adequadas para o tipo de problema com o qual estamos trabalhando é a chave para o bom desempenho do método. Neste capítulo, discutimos diferentes formas de gerar o padrão de direções, considerando as características específicas das restriçoẽs que constituem o problema, ou mais especificamente o conjunto de trabalho na iteração corrente.

Na Figura 5.1, organizamos , de acordo com [12, Fig 5.3], os diferentes casos com os quais podemos nos deparar quando trabalhamos com problemas de otimização com restrições lineares. Em seguida, discutimos cada caso separadamente.

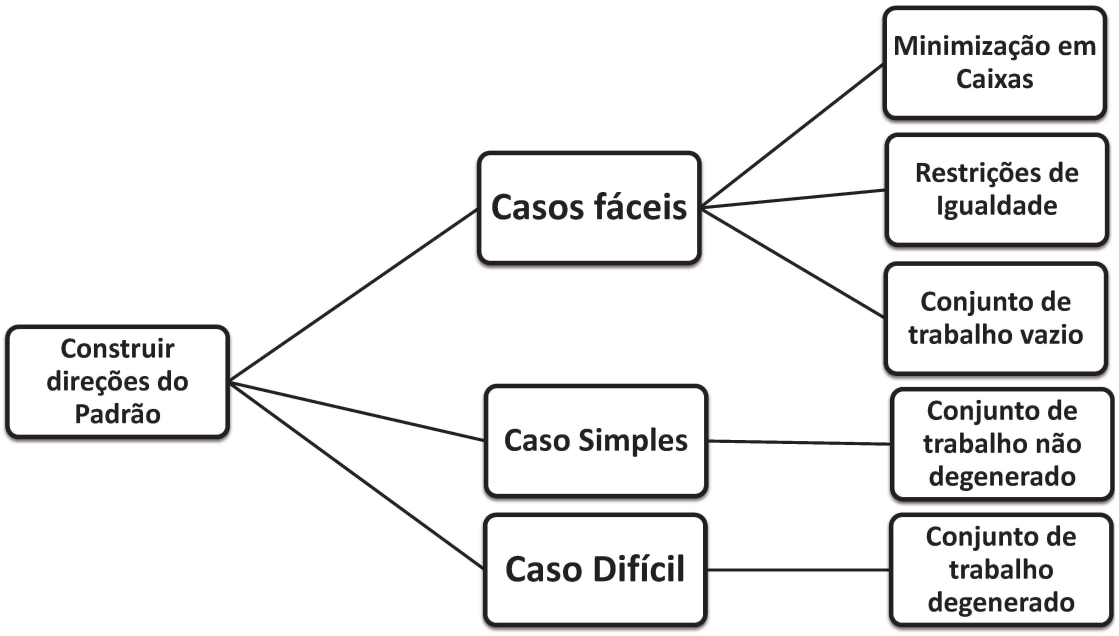


Figura 5.1: Considerações ao gerar o padrão de direções de busca

5.1 Minimização em Caixas

Em problemas de otimização, é comum nos depararmos com a necessidade de resolver um problema onde as restrições se resumem a limitantes para o valor das variáveis, ou seja, um problema da seguinte forma:

$$(1) \quad \begin{cases} min & f(x) \\ s.a & l \leq x \leq u, \end{cases}$$

onde $f : \mathbb{R}^n \to \mathbb{R}, \quad x, l, u \in \mathbb{R}^n$. Quando o problema apresentar esse tipo de restrição, diremos que estamos minimizando em uma caixa.

O problema de minimizar em uma caixa é um caso particular do problema de minimização com restrições lineares, onde a matriz de restrições A é igual a matriz identidade. Sendo assim, podemos aplicar o método descrito no capítulo anterior para resolver esse tipo de problema.

Em [9], Lewis e Torczon fazem um estudo detalhado do Método de Busca Padrão aplicado ao problema de minimizar em caixas. Neste trabalho, que é anterior ao estudo do método aplicado ao problema com restrições lineares [11], os autores definem o padrão P^k como descrevemos na seção 3.1, porém concluem que a matriz BM_k deve ser diagonal. Deste modo, as direções de busca serão múltiplos escalares das direções coordenadas.

Pensando no problema de minimizar em uma caixa como um caso particular do problema de restrições lineares, com $A = I$, percebemos que o cone $K(x, \epsilon)$, definido no capítulo 4, será gerado sempre por um subconjunto dos vetores coordenados $\pm e_i$. Logo, pela definição de cone polar, é fácil perceber que o cone $K^\circ(x, \epsilon)$ será gerado pelas demais direções coordenadas, ou seja, as direções que não estão no conjunto gerador do cone $K(x, \epsilon)$.

Portanto, é possível conhecer de antemão todos os possíveis geradores para o cone $K^\circ(x, \epsilon)$, independentemente do ponto $x \in \Omega$ e do valor de ϵ. Logo, como sabemos que o padrão P^k deve conter um conjunto de geradores positivos para o cone $K^\circ(x, \epsilon)$, podemos tomar $P^k = [I \quad -I]$ e utilizar este padrão de direções em todas as iterações do método.

Assim, ao utilizar este padrão, realizamos no pior caso $2n$ avaliações de função por iteração, o que em geral não ocorre. Além disso, ao manter o mesmo padrão em todas as iterações, não é necessário atualizá-lo, diminuindo assim o custo da iteração, o que pode compensar possíveis cálculos extras do valor da função objetivo.

5.2 Restrições Lineares de Igualdade

O problema de minimização com restrições lineares de igualdade pode ser escrito da seguinte forma:

$$(2) \quad \begin{cases} min & f(x) \\ s.a & Ax = b \end{cases}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$, $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$ e $b \in \mathbb{R}^m$.

Perceba que, ao aplicar o Método de Busca Padrão a este problema, não temos nenhuma flexibilidade ao gerar o padrão, uma vez que pequenos passos podem fazer com que a viabilidade seja perdida.

Tendo em vista que a principal característica do método é manter a viabilidade em todas as iterações, a cada iteração devemos ter:

$$A(x^k + s^k) \;=\; b.$$

Mas como temos que o ponto x^k é viável, ou seja, $Ax^k = b$, obtemos que as direções de busca devem satisfazer:

$$As^k \;=\; 0.$$

Logo, as direções de busca devem necessariamente estar no núcleo da matriz de restrições A, $\mathcal{N}(A)$. Assim, temos que o padrão P^k deve conter um conjunto gerador positivo para o núcleo da matriz A.

Portanto, se todas as restrições do problema são de igualdade, construímos um padrão contendo uma base positiva para $\mathcal{N}(A)$ antes de iniciar o passo de busca, e o mesmo padrão é utilizado em todas as iterações, uma vez que nenhuma outra direção de busca é possível.

Vale a pena destacar que, ainda que tenhamos somente uma restrição de igualdade, se desejamos preservar a viabilidade, é necessário caminhar por direções no núcleo desta restrição. Porém, na presença de restrições de desigualdade, estas também devem influenciar na escolha do padrão. Neste caso, construímos o padrão considerando apenas as restrições de desigualdade, depois projetamos no núcleo das restrições de igualdade, conforme veremos mais adiante.

5.3 Conjunto de Trabalho Vazio

Estamos trabalhando com problemas restritos, porém, caso estejamos no interior do conjunto viável, ou seja, nenhuma das restrições está a uma distância menor do que ϵ do ponto corrente, então, pelo menos por uma certa distância, podemos andar livremente em qualquer direção do $\mathbb{R}^n$. Deste modo momentaneamente recaímos no caso de minimizar sem restrições.

Sendo assim, o padrão, neste caso, deve seguir as condições descritas no capítulo 3. No entanto, em geral as restrições influenciam na solução do problema, e portanto permanecemos no interior do conjunto viável apenas durante algumas iterações. Por simplicidade, neste caso, realizamos uma busca coordenada. Isso quer dizer que utilizamos a matriz $P^k = [I \;\; -I]$ como padrão, sempre que nenhuma restrição for ϵ-ativa no ponto corrente.

5.4 Problemas com Restrições Lineares Gerais

Como vimos, em alguns casos o padrão de direções P^k pode ser obtido facilmente. No entanto, em geral nem sempre esta tarefa é trivial, e muitas características do problema podem dificultar no momento de escolher o padrão mais adequado.

Para começar vamos definir o que entendemos por conjunto de trabalho degenerado e não-degenerado.

Definição 5.1 *Seja V a matriz cujas colunas geram o cone $K(x^k, \epsilon)$. Se as colunas de V formam um conjunto linearmente independente, diremos que o conjunto de trabalho em x^k, ou seja, na iteração corrente, é não-degenerado. Caso contrário, diremos que o conjunto de trabalho na iteração corrente é degenerado.*

Quando o conjunto de trabalho é não-degenerado podemos construir o padrão de um modo relativamente simples, uma vez que, conseguimos provar analiticamente a existência de uma base racional para o cone $K^\circ(x^k, \epsilon)$, satisfazendo as condições que garantem a convergência do método. Já para o caso degenerado, embora seja possível demonstrar a existência, determinar essa base nem sempre é uma tarefa fácil. O resultado abaixo, o qual é discutido com detalhes em [11], garante que é possível construir a base para o cone $K^\circ(x^k, \epsilon)$, tanto para problemas não-degenerados como para os degenerados.

Proposição 5.1 *Suponha que K é um cone com gerador racional V. Então existe um conjunto gerador racional para o cone K°.*
Demonstração: *Ver [11, Proposição 8.1].*

Observe que, com a hipótese de que a matriz A é racional, a matriz V também será raciona. Dessa forma é possível obter o padrão satisfazendo as hipóteses de convergência, uma vez que podemos obter um padrão com direções inteiras a partir das direções racionais. Para isso basta "eliminarmos" os denominadores.

5.4.1 Conjunto de Trabalho Não-degenerado

Sabemos que sempre é possível determinar um padrão de direções. No entanto, ao demonstrar a Proposição 5.1, os autores concluem que determinar o padrão é um caso particular de determinar os pontos extremos de um poliedro, o que em geral não é fácil. Porém, quando a matriz V tem posto completo, o resultado abaixo, cuja demonstração pode ser encontrada em [11], nos fornece uma forma simples de obter o conjunto gerador para o cone $K^\circ(x, \epsilon)$. Por este motivo julgamos importante discutir sua demonstração aqui.

Proposição 5.2 *Suponha que para algum δ, $K(x, \delta)$ tem um conjunto linearmente independente de geradores racionais V. Seja N uma base racional positiva para o espaço nulo de V^T. Então para qualquer ϵ, $0 \leq \epsilon \leq \delta$, um conjunto racional de geradores para $K^\circ(x, \epsilon)$ pode ser obtido entre as colunas de $N, V(V^TV)^{-1}$, e $-V(V^TV)^{-1}$.*

Demonstração: Dados $x \in \Omega$ e $\delta > 0$, seja $K = K(x, \delta)$. Suponha $w \in K^\circ$, então $\forall\, v \in K$ temos pela definição de cone polar que:

$$\langle w, v \rangle \leq 0.$$

Como V é um gerador do cone K temos que qualquer $v \in K$ pode ser escrito da forma $v = V\lambda,\ \lambda \geq 0$. Assim, temos:

$$\langle w, V\lambda \rangle \leq 0,$$

Mas $w = Proj_{\mathcal{N}(V^T)}w + Proj_{\mathcal{R}(V)}w$, logo

$$\left\langle Proj_{\mathcal{N}(V^T)}w + Proj_{\mathcal{R}(V)}w, V\lambda \right\rangle \leq 0.$$

Como, por hipótese a matriz V tem posto completo, temos que as projeções no núcleo de V^T e na imagem de V são dadas por, $I - V(V^TV)^{-1}V^T)w$ e $(V(V^TV)^{-1}V^T)w$, respectivamente. Então aplicando na desigualdade acima temos:

$$\left\langle (I - V(V^TV)^{-1}V^T)w + (V(V^TV)^{-1}V^T)w, V\lambda \right\rangle \leq 0,$$

$$\underbrace{\left\langle (I - V(V^TV)^{-1}V^T)w, V\lambda \right\rangle}_{0} + \left\langle (V(V^TV)^{-1}V^T)w, V\lambda \right\rangle \leq 0.$$

Como $\mathcal{N}(V^T)$ e $\mathcal{R}(V)$ são complementos ortogonais no $\mathbb{R}^n$, temos que o primeiro produto interno se anula, e utilizando a definição do produto interno usual no segundo termo obtemos:

$$\lambda^T \underbrace{V^TV(V^TV)^{-1}}_{I} V^Tw \leq 0,$$

$$\lambda^T \underbrace{V^Tw}_{\tau} \leq 0 \Rightarrow \langle \tau, \lambda \rangle \leq 0.$$

Logo, como a desigualdade é válida para todo $\lambda \geq 0$, temos que $\tau \leq 0$.

Utilizando a hipótese que N é uma base positiva para $\mathcal{N}(V^T)$, temos que $w \in K^\circ$ pode ser escrito da forma:

$$w = N\varsigma - V(V^TV)^{-1}\bar{\tau},$$

onde $\varsigma \geq 0$ e $\bar{\tau} \geq 0$.

Portanto, as colunas de N e $-V(V^TV)^{-1}$ são um conjunto gerador positivo para K°. Entretanto, sem perda de generalidade, suponha que a matriz V tem p colunas e que $\lambda_1, \ldots, \lambda_r = 0,\ r < p$, então teremos que $\tau_{r+1}, \ldots, \tau_p \leq 0$. Entretanto $\tau_1 \ldots \tau_r$ são livres de sinal. Sendo assim, de modo a abranger todos os casos, temos que, se tomarmos as colunas de $V(V^TV)^{-1},\ -V(V^TV)^{-1}$ e N, sempre conterão um conjunto gerador positivo para $K^\circ(x, \delta)$, conforme queríamos demonstrar.

Esse resultado é importante e nos fornece um modo simples de construir o padrão de direções, que deve conter um gerador positivo para o cone K°. Abaixo descrevemos dois padrões diferentes, baseados neste resultado, o primeiro deles é o padrão proposto em [12] e o segundo propomos neste trabalho.

Antes de descrever cada um dos padrões em detalhes, vamos discutir alguns pontos comuns, referentes às alterações necessárias no conjunto de índice, de modo a identificar as restrições de igualdade. Vale destacar que, quando aplicamos o método com restrições gerais, ou mesmo contendo somente restrições de igualdade, não incluímos na matriz A a restrições de caixa que o problema possa ter. Ao invés disse, analisamos se essas restrições estão sendo violadas no momento de verificar a viabilidade do passo, conforme discutiremos na seção 6.2.

Como discutimos na seção 5.2, quando o problema possui restrições de igualdade, só podemos nos mover por direções que estejam no núcleo de tais restrições. Deste modo, é fundamental para o bom funcionamento do método que essas restrições sejam corretamente consideradas no momento de gerar o padrão, de forma a evitar que todas as direções construídas sejam inviáveis para o problema.

O problema que estamos resolvendo é definido por:

$$(PRL)\begin{cases} min & f(x) \\ s.a & l \leq Ax \leq u, \end{cases}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$, $x \in \mathbb{R}^n$, $A \in \mathbb{Q}^{m\times n}$, $l, u \in \mathbb{R}^m$. Teremos uma restrição de igualdade quando $l_i = u_i$. Consideremos os seguintes conjuntos de índices, de modo a identificar as restrições de igualdade e desigualdade:

$$\mathbb{I} = \{i : \ l_i = u_i\} \ \ e \ \ \mathbb{D} = \{i : \ l_i < u_i\}.$$

Deste modo as restrições de desigualdade ativas serão identificadas pelos seguintes conjuntos de índices:

$$\mathbb{D}_l(x, \epsilon) = \{i : x \in \partial\Omega_{l_i}(\epsilon)\},$$

$$\mathbb{D}_u(x, \epsilon) = \{i : x \in \partial\Omega_{u_i}(\epsilon)\},$$

onde $\partial\Omega_{l_i}(\epsilon)$ e $\Omega_{u_i}(\epsilon)$, foram definidos na seção 4.1.1. Pode acontecer também de uma determinada restrição estar no conjunto de restrições ϵ-ativas, tanto pelo seu limitante inferior quanto pelo limitante superior, como mostra a figura 5.2. Neste caso:

$$\mathbb{D}_e(x, \epsilon) = \{i : i \in \mathbb{D}_l(x, \epsilon) \cap \mathbb{D}_u(x, \epsilon)\}.$$

Embora as restrições i tais que $i \in \mathbb{D}_e$ sejam de desigualdade, elas aparecem ativas tanto pelo seu limitante inferior, quanto pelo limitante superior. Então, ao serem consideradas como restrições de desigualdade, nesta iteração, a restrição aparecerá, com o sinal trocado, duas vezes no conjunto gerador do cone $K(x, \epsilon)$, originando um cojunto de trabalho degenerado.

De modo a evitar trabalhar com conjuntos degenerados, quando esse fato ocorrer diremos que as restrições envolvidas são momentaneamente de igualdade, e trabalharemos com elas juntamente com as restrições de igualdade do problema.

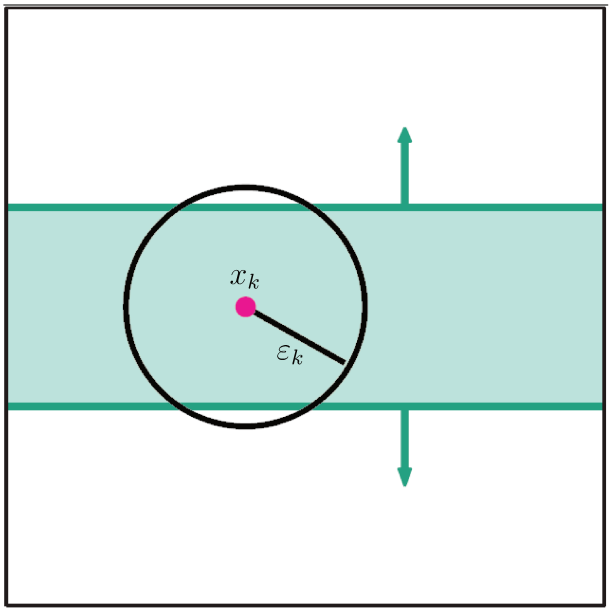


Figura 5.2: Restrições momentaneamente de igualdade

Padrão 1:

O padrão proposto por Lewis, Shepherd e Torczon em [12] assume que conhecemos de antemão que o conjunto de trabalho é não-degenerado. Sendo assim é necessário testar o posto da matriz antes de construir o padrão, embora as direções possam ser obtidas mesmo na presença de degenerescência.

Vamos separar o conjunto de trabalho em duas matrizes da seguinte forma:

$$V_e = \{a_i : \ i \in \mathbb{I}\} \bigcup \{a_i : \ i \in \mathbb{D}_e\},$$

$$V_d = \left\{a_i^T : \ i \in \mathbb{D}_l \backslash \mathbb{D}_e\right\} \bigcup \left\{-a_i^T : \ i \in \mathbb{D}_u \backslash \mathbb{D}_e\right\},$$

onde a_i indica a i-ésima linha da matriz de restrições A. Então, para construir o padrão P^k devemos seguir os seguintes passos:

- encontre uma base ortonormal Z para o espaço nulo de V_e, lembrando que as linhas desta matriz contém as linhas da matriz A correspondentes as restrições de igualdade da iteração corrente;
- calcule a matriz $B = Z^T V_d$, lembrando que as colunas de V_d contém as linhas de A correspondentes às restrições de desigualdade ativas no ponto corrente, e verifique se as colunas de B formam um conjunto linearmente independente de vetores;

- caso B não possua posto completo, o conjunto de trabalho é degenerado e outra estratégia deve ser utilizada, o que comentaremos mais adiante;
- se B tem posto completo então calcule a pseudoinversa R da matriz B^T e uma base ortonormal positiva N para o núcleo de B^T, ou seja, calculamos uma base ortononormal Q para o núcleo da matriz B^T e fazemos $N = [Q \ -Q]$;
- então o padrão P^k é dado por:

$$P^k = [-ZR \ \ ZN].$$

Embora os autores não comentem, tudo leva a crer que é realizada ao menos uma decomposição em valor singular (SVD) por iteração na construção do padrão, uma vez que, além de testar o posto da matriz B, é calculada a pseudoinversa e uma base ortogonal para o núcleo da matriz B^T e V_e.

No entanto, o custo de se realizar uma decomposição em valor singular por iteração é muito alto, o que torna o método inviável para problemas maiores. Além disso, no caso de restrições degeneradas, embora o padrão possa ser calculado, testes computacionais mostraram que esta não é uma boa escolha, uma vez que muitas vezes o método converge para pontos que não são estacionários. Segue abaixo o algoritmo que utilizamos na construção do Padrão 1.

Algoritmo 5.1 Construção do Padrão 1

Dados V_e e V_d.

1. *Se $V_e = \emptyset$ e $V_d = \emptyset$, faça: $P^k = [I \ \ -I]$, e finalize.*
2. *Obtenha a decomposição em valores singulares da matriz V_e.*
3. *Determine uma base ortogonal Z para o núcleo da matriz V_e utilizando a decomposição obtida no passo 2.*
4. *Faça: $B = Z^T V_d$.*
5. *Obtenha a decomposição em valores singulares da matriz B^T.*
6. *Determine a pseudoinversa R da matriz B^T, utilizando a decomposição obtida no passo 5.*
7. *Determine uma base ortogonal Q, para o núcleo da matriz B^T, utilizando a decomposição obtida no passo 5.*
8. *Faça: $N_1 = ZR$.*
9. *Faça: $N_2 = ZQ$.*
10. *Faça: $P^K = [-N_1 \ \ N_2 \ \ -N2]$.*

Padrão 2:

O padrão que propomos neste trabalho é baseado no resultado obtido na proposição 5.2. Baseado em projeções e utilizando direções e bases racionais, o custo de se calcular este padrão é bem inferior ao anterior. Além disso, os testes computacionais apresentaram bons resultados, obtendo uma boa precisão em praticamente todos os problemas testados, inclusive alguns com base degenerada, para os quais o Padrão 1 falhou.

Em alguns testes percebemos que ao alcançar a face ótima, o método acaba ficando "preso"nesta face. Isso acontece pois as direções do padrão não permitiam sair do ponto corrente por pontos viáveis diminuindo o valor da função objetivo, embora o ponto corrente não fosse estacionário.

Por esse motivo, quando estamos em uma das faces, ou seja, uma ou mais restrições de desigualdade atingiram um de seus limitantes, adicionamos ao padrão direções que estejam no núcleo destas restrições. Dessa forma permitimos que o método possa se mover na face. É importante ressaltar que essas direções são consideradas separadamente, uma vez que, se não estamos em uma face ótima, deve ser possível sair desta face. Sendo assim considere V_e e V_d, como no Padrão 1 e também:

$$\mathbb{I}_2(x^k, \epsilon_2) = \left\{ i : \ |a_i^T x^k - l_i| \leq \epsilon_2 \ ou \ |a_i^T x^k - u_i| \leq \epsilon_2 \right\},$$

$$V_{e2} = \left\{ a_i : \ i \in \mathbb{I}_2(x^k, \epsilon_2) \right\}.$$

Dessa forma, dadas as matrizes V_e, V_d e V_{e2}, o padrão é obtido da seguinte forma:

- determine uma base racional T para o núcleo da matriz V_e;
- determine uma base racional N para o núcleo da matriz V_d;
- determine uma base racional T_2 para o núcleo da matriz V_{e2};
- determine a matriz $F = V_d(V_d^T V_d)^{-1}$, a qual é obtida pela resolução de vários sistemas lineares utilizando a mesma matriz de coeficientes, sendo assim, utilizamos a fatoração de Cholesky da matriz $(V_d^T V_d)$;
- o padrão P^k fica da seguinte forma:

$$[P^k] = [-T \ \ T \ \ -N \ \ N \ \ F \ \ -F \ \ T_2 \ \ -T_2].$$

Observe que o Padrão 2 possui um número maior de direções de busca, mas cumpre a hipótese de convergência de conter um conjunto gerador positivo para o cone $K^\circ(x^k, \epsilon)$.

O Padrão 2 é construído calculando apenas uma fatoração de Cholesky. Com isso, o custo de obter essas direções é bem inferior, comparado ao custo de se calcular a SVD, como é feito no caso do Padrão 1.

Pensando no problema com restrições degeneradas, é possivel perceber que, no Padrão 2, não será possível determinar a matriz F, uma vez que a matriz $(V_d^T V_d)$ não

possui inversa para o caso degenerado. Perceba, no entanto, que não testamos o posto de V_d, pois para o caso degenerado propomos uma alternativa para obter a matriz F. Na próxima seção discutimos as alternativas de cada um dos padrões apresentados, para o caso degenerado.

5.4.2 Conjunto de Trabalho Degenerado

Quando o conjunto de trabalho é degenerado, a construção do padrão de direções é mais complicada, e nem sempre as direções obtidas funcionam bem para o problema. Além disso, frequentemente a convergência é lenta, e a precisão da solução obtida tende a ser menor. Abaixo, descrevemos como cada um dos padrões propostos trabalha com esse tipo de problema.

Padrão 1:

Os autores comentam em [12] que determinar o padrão de direções, na presença de restrições degeneradas, é um caso particular de se determinar os vértices e raios extremos de um poliedro, uma vez que $K(x, \epsilon)$ é um cone com vértice degenerado na origem.

Esse problema foi bastante explorado na literatura e muitos algoritmos práticos foram desenvolvidos para resolver esse tipo de problema. Para resolver o problema os autores utilizam o pacote *cddlib* denvolvido e disponibilizado por Komei Fukuda [6].

Sendo assim, toda vez que é detectado um conjunto de trabalho degenerado, é chamada uma rotina externa para determinar as direções de busca. Segundo os autores, esta estratégia funcionou muito bem na maioria dos problemas. Além disso, o tempo de execução da rotina *cddlib* é perfeitamente aceitável, permitindo a resolução de grande parte dos problemas testados em um tempo razoável.

Padrão 2:

Diferentemente do que acontece com o Padrão 1, em que a construção é possível mesmo com restrições degeneradas, o modo como o Padrão 2 é construído impede o cálculo da componente F do padrão. Isso acontece porque se o conjunto de trabalho é degenerado, a matriz $(V_d^T V_d)$ será singular, não sendo possível obter a solução do sistema linear, o qual determina o termo principal do padrão.

Porém, ainda que o sistema não tenha solução exata, a solução de Quadrados Mínimos sempre pode ser obtida. Logo, o que propomos é utilizar a solução de Quadrados Mínimos, obtida através da fatoração QR, no lugar da solução exata do sistema linear.

Utilizando esta estratégia, o padrão para o caso degenerado pode ser obtido praticamente com o mesmo custo do caso não degenerado, exceto pelo custo de calcular a fatoração QR, que é mais cara que a fatoração de Cholesky. No entanto, se essas direções forem satisfatórias, ainda que a convergência seja mais lenta, por se tratar de um conjunto degenerado, o problema pode ser resolvido sem que haja a necessidade de utilizarmos uma rotina externa para construir o conjunto de busca.

Nos próximos capítulos, faremos testes computacionais com o intuito de analisar se esta estratégia é viável, ou seja, fornece uma solução aceitável em um tempo de execução razoável.

Segue abaixo o algoritmo utlizado na construção do Padrão 2, já englobando as alternativas propostas para o caso degenerado:

Algoritmo 5.2 Construção do Padrão 2

Dados V_e, V_d e V_{e2}.

1. *Se $V_e = \emptyset$ e $V_d = \emptyset$, faça: $P^k = [I \quad -I]$, e finalize.*
2. *Obtenha a forma escalonada reduzida das matrizes V_e, V_d e V_{e2}.*
3. *A partir das matrizes obtidas no passo 2, determine:*
   - *i. uma base racional T para o núcleo da matriz V_e;*
   - *ii. uma base racional T_2 para o núcleo da matriz V_{e2};*
   - *iii. uma base racional N para o núcleo da matriz V_d.*
4. *Obtenha a Fatoração de Cholesky da matriz $V_d^T V_d$. Se a fatoração tiver sucesso, vá para o passo 7.*
5. *Obtenha a fatoração QR da matriz $V_d^T V_d$.*
6. *Obtenha a matriz F, através da resolução de p problemas de Quadrados Mínimos, utilizando a Fatoração QR obtida no passo 5, de modo que:*

$$(V_d^T V_d) F^T = V_d^T.$$

   Vá para o passo 8.

7. *Obtenha a matriz F através da resolução de p sistemas lineares, usando o fator de Cholesky obtido no passo 4, de modo que:*

$$F^T = (V_d^T V_d)^{-1} V_d^T.$$

8. *Faça:*

$$P^k = [T \ -T \ \ N \ -N \ \ F \ -F \ \ T_2 \ -T_2].$$

Capítulo 6

Implementação Computacional do Método

Como vimos nos capítulos 3 e 4, para que um determinado algoritmo seja classificado como um Método de Busca Padrão, algumas características que garantem a convergência do método devem ser preservadas. Porém, além do padrão de direções utilizado, diferentes estratégias de busca e formas de atualizar o tamanho do passo definem novos métodos, também classificados como sendo de Busca Padrão.

Neste capítulo, discutiremos os detalhes práticos, necessários para aplicar o método, tais como, ponto inicial, estratégias de busca e atualização do padrão, e os critérios de parada e tolerância utilizados.

6.1 Ponto Inicial

Para aplicar o método descrito no capítulo 4, é preciso fornecer, como dado de entrada para o algoritmo, um ponto inicial viável. Em geral, obter pontos viáveis nem sempre é uma tarefa fácil na prática. Entretanto, estamos trabalhando apenas com restrições lineares ou de caixa, e tendo em vista que, a viabilidade que procuramos não depende da função objetivo, que é não linear, podemos aplicar técnicas de programação linear para obter um ponto viável.

Programamos o método Preditor-Corretor [16, 17], o qual utilizamos com um vetor de custos qualquer, este rapidamente fornece um ponto viável. A fase I do Método Simplex de Programação Linear [2] também pode ser utilizada, no entanto o método de pontos interiores se mostrou mais eficiente.

6.2 Estratégias de Busca

Sabemos que, para preservar as características de convergência do método, a busca deve satisfazer uma das hipóteses sobre os movimentos exploratórios discutidas na seção 3.2. Outro detalhe importante é a viablidade, que deve ser preservada em todas as iterações.

Em [12], os autores utilizam direções de busca com norma 1, e as restrições ϵ-ativas são determinadas levando em consideração o tamanho do passo da iteração corrente. Desta forma, teoricamente a viabilidade seria preservada. Porém, na prática, isso não foi observado. A maioria das soluções que obtemos, utilizando esta estratégia, não eram viáveis, dentro das tolerâncias que consideramos aceitáveis.

Por esse motivo, adotamos outra estratégia, a qual se mostrou melhor, tanto no sentido de manter a viabilidade quanto em eficiência. O que fazemos é testar a viabilidade do ponto antes de calcular o valor da função objetivo no mesmo. Sendo assim, somente passos viáveis são aceitos. Deste modo a viabilidade sempre é preservada, além de evitar avaliações de função desnecessárias.

No caso de eficiência observamos que, muitas vezes, devido às características do problema, as direções de busca geradas a partir das restrições ϵ-ativas não funcionam bem. Porém, adicionando algumas direções ao padrão, o método volta a funcionar bem. Portanto, como estamos testando a viabilidade do passo, podemos adicionar essas direções sem correr o risco de dar passos infactíveis.

Vale destacar também que, sempre que o problema possui restrições de caixa, estas são consideradas neste passo do algoritmo, onde fiscalizamos para que sejam sempre satisfeitas, ou seja, trabalhamos com o problema definido da seguinte forma:

$$\begin{cases} min & f(x) \\ s.a & l \leq Ax \leq u \\ & bl \leq x \leq bu. \end{cases}$$

Abordando o problema desta forma, em um primeiro momento não nos preocupamos com as restrições de caixa. Sendo assim, ao gerar o padrão de direções, trabalhamos com um conjunto reduzido de restrições e somente ao testar a viabilidade do passo fiscalizamos se todas as restrições estão sendo satisfeitas, inclusive as de caixa. Caso optemos pela estratégia utilizada em [12], todas as restrições devem ser incluídas na matriz A, para que a viabilidade seja garantida.

Logo, em todas as estratégias de busca que descrevemos abaixo, tomamos o cuidado de satisfazer uma das hipóteses sobre os movimentos exploratórios, e utilizamos a estratégia descrita acima para preservar a viabilidade.

6.2.1 Busca Simples

A busca simples é a mais comum e provavelmente a mais utilizada em Métodos de Busca Direta em geral.

Esta estratégia consiste em buscar, entre as direções do padrão, um ponto que possua um valor de função objetivo menor que o do ponto atual, exigindo apenas decréscimo simples, sem impor nenhuma condição de decréscimo suficiente. A principal característica desta estratégia é finalizar a busca e atualizar o tamanho do passo no momento que encontra um ponto com valor de função objetivo melhor. Para garantir que as hipóteses

fracas sobre os movimentos exploratórios sejam satisfeitas, buscamos em todas as direções do padrão antes de declarar o fracasso da iteração.

Note que, esta estratégia de busca garante que existe um ponto limite da sequência gerada pelo método que é um ponto KKT do problema. Logo, a busca simples oferece a vantagem de não avaliar a função em todas as direções do padrão, em todas as iterações. No entanto, pode ocorrer do método não convergir para a solução, uma vez que temos a garantia apenas da existência de um ponto limite que converge para a solução.

Portanto, ao utilizar esta estratégia de busca, esperamos que o método seja eficiente, mas não tão robusto. Analisaremos o desempenho da estratégia no próximo capítulo.

6.2.2 Busca Completa

A busca completa consiste em procurar, entre as direções do padrão, aquela que oferece o maior decréscimo no valor da função objetivo. Isso quer dizer que, ao contrário da busca simples, ao encontrar uma direção que oferece decréscimo no valor da função objetivo, não finalizamos a busca. Para garantir que estamos andando na direção que oferece o maior decréscimo, todas as direções do padrão devem ser testadas.

Observe que esta estratégia de busca satisfaz as hipóteses fortes sobre os movimentos exploratórios, e portanto a teoria de convergência garante que todo ponto limite da sequência gerada pelo método converge é um ponto KKT do problema.

É fácil perceber que, ao utilizar a busca completa, o custo da iteração aumenta significativamente. Isso se deve ao fato que em métodos sem derivadas o custo tem como base o número de avaliações da função objetivo, que em geral representa o aspecto mais caro do algoritmo.

No entanto, apesar do aumento no número de avaliações da função objetivo, por iteração, em geral as iterações necessárias para convergência diminuem. Isso pode compensar o custo maior da iteração, associado à vantagem de satisfazer as condições que fornecem um resultado de convergência mais forte. Logo, esperamos que com esta estratégia o método seja mais robusto, ainda que possivelmente seja menos eficiente.

6.2.3 Busca Simples Ordenando o Padrão

Pensando no fato que, ao realizar a busca simples, a iteração é encerrada ao encontrar uma direção de descida factível, é possível perceber que a ordem em que testamos as direções pode alterar o número de avaliações da função objetivo. Então, podemos pensar em ordenar as direções do padrão, levando em conta os resultados de iterações bem sucedidas. Esta estratégia pode melhorar o desempenho do método na iteração seguinte.

Porém, para aplicar esta estratégia, alguns aspectos do problema devem ser considerados. Perceba que não faz sentido pensar em ordenar as direções considerando as direções de busca em uma iteração, se na iteração seguinte o padrão mudar, como acontece quando estamos trabalhando com problemas que possuem restrições de desigualdade.

Portanto, só faz sentido pensar em ordenar o padrão se este for mantido fixo em todas as iterações, ou seja, em problemas que não possuam restrições de desigualdade. Sendo assim, a estratégia de ordenação somente será utilizada em problemas com restrições lineares de igualdade e restrições de caixas, conforme discutimos no capítulo anterior.

A estratégia de ordenação que utilizamos é bastante simples, uma vez que não é vantajoso usar estratégias elaboradas para economizar algumas avaliações da função. O que fazemos é simplesmente testar a direção que ofereceu decréscimo na última iteração antes de testar as demais direções do padrão.

O algoritmo abaixo organiza as estratégias de busca que acabamos de discutir.

Algoritmo 6.1 Estratégias de busca

Dados $x^k \in \Omega$, $val = f(x^k)$, $\alpha_k > 0$ *e* $P^k \in \mathbb{R}^{n\times p}$.
Para $i = 1, \ldots, p$,

1. *Calcule* $x = x^k + \alpha_k P^k(:, i)$.
2. *Verifique se* $x \in \Omega$. *Se* $x \notin \Omega$, *vá para o passo 5.*
3. *Calcule* $v = f(x)$.
4. *Se* $v < val$ *selecione uma das estratégias de busca abaixo, caso contrário vá para o passo 5;*
   (a) *Busca Simples*
      i. *Faça* $x^{k+1} = x$;
      ii. *Faça* $val = v$;
      iii. *Faça* $i = p$;
      iv. *Declare sucesso.*
   (b) *Busca Simples Ordenando o Padrão*
      i. *Faça* $x^{k+1} = x$;
      ii. *Faça* $val = v$;
      iii. *Faça* $P^{k+1}(:, 1) = P^k(:, i)$ *e* $P^{k+1}(:, i) = P^k(:, 1)$;
      iv. *Faça* $i = p$;
      v. *Declare sucesso.*
   (c) *Busca Completa*
      i. *Faça* $x^{k+1} = x$;
      ii. *Faça* $val = v$;
      iii. *Declare sucesso.*
5. *Faça* $i = i + 1$. *Se* $i \leq p$ *volte ao passo 1, caso contrário finalize a busca.*

6.3 Atualização do Tamanho do Passo

Como descrevemos anteriormente, a atualização do tamanho do passo deve satisfazer alguns critérios para termos garantias de convergência. Obedecendo a esses critérios obrigatórios, propomos uma estratégia para atualizar o tamanho do passo que leva em consideração as iterações bem-sucedidas consecutivas.

Criamos a variável *suc* que é incrementada sempre que a iteração é bem-sucedida, e quando a iteração resulta em fracasso atribuímos zero a esta variável, que volta a ser incrementada somente quando o método obtiver sucesso novamente.

Então a atualização do tamanho do passo é segue o algoritmo abaixo:

Algoritmo 6.2 Atualização do tamanho do Passo

Dados $\alpha_k > 0,\ \tau > 1,\ suc \geq 0$.

1. *Se* $suc > 0$ *(sucesso) faça:* $\alpha_{k+1} = min\{1000, \tau^{suc}\alpha_k\}$.

2. *Se* $suc = 0$ *(fracasso) faça:* $\alpha_{k+1} = 0.5\alpha_k$.

Observe que essa estratégia de atualização segue todas as condições descritas na seção 3.3.1. Adotamos essa estratégia pensando na possibilidade de dar passos mais ambiciosos quando o método está indo bem. É importante ressaltar também que estabelecemos um limite para o tamanho do passo, que não pode crescer indefinidamente, uma vez que a sequência $\{\alpha_k\}$ deve ir para zero.

A estratégia utilizada em caso de fracasso é a usual, ou seja, apenas dividimos o tamanho do passo por 2, porém nada impede de que adotemos uma estratégia que leve em consideração os fracassos consecutivos.

É importante destacar que, quando a iteração fracassa, após uma sequência de sucessos, não seguimos o algoritmo 6.2, mas voltamos ao último passo que obteve sucesso, o qual foi armazenado e não precisa ser calculado. Fazemos isso porque após uma sequência de sucessos, dependendo do valor utilizado para τ, o tamanho do passo cresce muito rápido e talvez, voltando ao último passo que ofereceu sucesso, possamos voltar a aumentar o tamanho do passo mais lentamente.

6.4 Informações Complementares

Programamos o método em Matlab versão 7.10.0. Utilizamos como critério de parada uma tolerância de 10^{-8} para o tamanho do passo α_{k+1}, e limitamos o número máximo de avaliações da função, cujo valor variamos com o intuito de analisar o desempenho do método.

Para determinar as restrições ϵ-ativas da iteração corrente fazemos $\epsilon = min\{\alpha_k, 1\}$, de modo a minimizar as chances de trabalharmos com conjuntos degenerados. Como testamos a viabilidade, não precisamos nos preocupar em gerar apenas direções que sejam viáveis para o tamanho de passo da iteração corrente.

Capítulo 7

Testes Numéricos

Analisamos o desempenho do método utilizando os dois diferentes padrões de direções, discutidos nos capítulo 5, e cada uma das estratégias, de busca e atualização do tamanho do passo, descritas no capítulo 6. O conjunto de testes é composto por 62 problemas extraídos de [7, 15, 20] e programados em Matlab.

Sabemos que o método estudado possui teoria de convergência global para problemas suaves. Porém, a motivação ao se estudar um método sem derivadas é aplicá-lo a problemas onde as derivadas não estão disponíveis ou nos quais o custo de calcular as derivadas é muito alto. Sendo assim, dividimos o conjunto de testes em 2 subconjuntos, um contendo os problemas suaves extraídos de [7, 20] e o outro composto por problemas não suaves, do tipo minimax, extraídos de [15].

Para cada um dos conjuntos de testes, além de comparar o desempenho do método variando o padrão de direções, queremos observar também como este se comporta para cada uma das estratégias, de busca e atualização do tamanho do passo, que propomos no capítulo 6.

Em todos os testes analisamos a eficiência e a robustez do método, variando de 10 a 10^5 o número máximo de avaliações de função permitido. Consideramos que o problema foi resolvido se o melhor valor da função objetivo que obtemos for menor ou estiver a uma distância menor que 10^{-7} do valor ótimo reportado nas referências. Quando a bibliografia não disponibiliza este resultado, usamos, para comparação o valor obtido pela rotina $fmincon$ do Matlab.

Os resultados estão apresentados em gráficos, onde o eixo das abscissas representa o número máximo de avaliações da função objetivo, e o eixo das ordenadas, em uma escala de 0 a 1, representa a proporção de problemas resolvidos. Deste modo, 0 indica que nenhum dos problemas foi resolvido e 1 que todos os problemas foram resolvidos. Maiores detalhes podem ser obtidos no Apêndice A.

Neste capítulo nos restringimos a apresentar os resultados e fazer uma análise comparativa dos mesmos. Nas considerações finais, apresentamos uma análise mais crítica, justificando o desempenho do método com as alterações propostas.

7.1 Problemas Suaves

O conjunto de problemas suaves é composto por 42 problemas: todos os 23 problemas com restrições lineares ou de caixa de [7] e problemas do mesmo tipo com até 50 variáveis, selecionados de [20]. Os problemas estão detalhados na tabela 7.1.

Para cada um dos padrões testamos as três estratégias de busca e utilizamos $\tau = 1$, $\tau = 1.5$ e $\tau = 2$, de modo a comparar como cada uma das estratégias se comporta ao variarmos o parâmetro de atualização do tamanho do passo.

Os resultados serão organizados da seguinte forma:

- Busca Simples
  - Padrão 1: variando o parâmetro de atualização do tamanho do passo.
  - Padrão 2: variando o parâmetro de atualização do tamanho do passo.
  - Padrão 1 $\times$ Padrão 2 : Busca simples (melhor τ).
- Busca Completa
  - Padrão 1: variando o parâmetro de atualização do tamanho do passo.
  - Padrão 2: variando o parâmetro de atualização do tamanho do passo.
  - Padrão 1 $\times$ Padrão 2 : Busca completa (melhor τ).
- Busca Simples Ordenando o Padrão
  - Padrão 1: variando o parâmetro de atualização do tamanho do passo.
  - Padrão 2: variando o parâmetro de atualização do tamanho do passo.
  - Padrão 1 $\times$ Padrão 2 : Busca Simples Ordenando o Padrão (melhor τ).

Após verificar o melhor valor para o parâmetro τ, comparamos o desempenho das estratégias de busca da seguinte forma:

- Padrão 1: comparação entre as estratégias de busca.
- Padrão 2: comparação entre as estratégias de busca.

7.1.1 Busca Simples

A busca simples com $\tau = 1$ seria a estratégia usual em Métodos de Busca Padrão, onde o tamanho do passo é mantido fixo em caso de sucesso. Como podemos observar pela Figura 7.1, para o Padrão 1, proposto em [12], o desempenho do método para $\tau = 1$ e $\tau = 2$ foi equivalente, ambos resolveram aproximadamente 14% dos problemas com apenas 10 avaliações de função, com até 10^5 avaliações de função foram resolvidos aproximadamente 88% dos problemas.

Problemas Suaves					
Prob.	n	# Desigualdades	# Igualdades	Lim. Inf.	Lim. Sup.
1	3	2	0	3	3
2	2	0	0	1	0
3	2	1	0	2	2
4	2	3	0	2	0
5	4	0	1	4	4
6	3	0	1	0	0
7	3	1	0	3	0
8	3	1	0	3	3
9	3	2	0	3	3
10	4	0	0	4	4
11	4	0	1	4	4
12	4	6	0	4	0
13	5	0	0	5	5
14	5	0	2	0	0
15	5	0	2	0	0
16	5	0	3	0	0
17	5	0	3	0	0
18	3	0	1	3	3
19	4	3	0	4	0
20	10	0	0	10	10
21	5	10	0	5	0
22	8	1	0	8	8
23	15	29	0	15	15
24	2	0	0	1	0
25	2	0	0	1	0
26	2	0	0	1	0
27	2	0	0	2	0
28	2	0	1	0	0
29	2	1	0	2	2
30	3	0	1	0	0
31	3	1	0	0	0
32	3	1	0	3	0
33	3	2	0	0	0
34	4	0	1	4	4
35	4	6	0	4	0
36	5	0	2	0	0
37	5	0	3	0	0
38	5	0	3	0	0
39	5	0	3	0	0
40	3	0	1	3	3
41	4	3	0	4	0
42	4	15	0	4	4

Tabela 7.1: Problemas Suaves

Vale destacar que, com apenas 10 avaliações da função, em vários casos consideramos que o problema foi resolvido, mas isso não indica que o tamanho do passo já tenha atingido a tolerância de 10^{-8} estipulada. Porém já é possivel observar a convergência para o ponto ótimo, já conhecido, de acordo com os critérios estabelecidos.

Ainda pela Figura 7.1, é possível observar que, para $\tau = 1.5$, o desempenho foi bem inferior, menos eficiente: com 10 avaliações da função o algoritmo resolve apenas 4% dos problemas, 10% a menos que para os outros dois parâmetros. Quanto à robustez, a diferença foi pequena, com 10^5 avaliações da função foram resolvidos 85% dos problemas.

Propomos o valor de $\tau = 1.5$, pensando em um crescimento moderado no tamanho do passo, uma vez que para $\tau = 2$ o tamanho do passo cresce rapidamente. No entanto, como podemos observar, o desempenho apresentado foi inferior ao que esperávamos

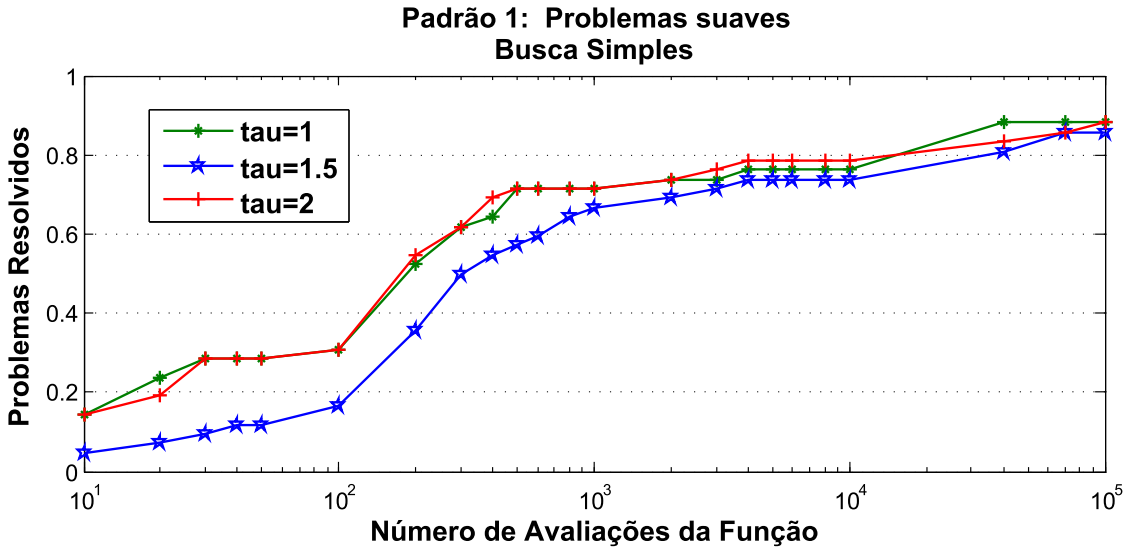


Figura 7.1: Problemas suaves: Padrão 1 - Busca Simples

Já para o Padrão 2, que propomos neste trabalho, de acordo com a figura 7.2, podemos observar um comportamento semelhante ao que observamos para o Padrão 1, cujo desempenho foi aproximadamente o mesmo para $\tau = 1$ e $\tau = 2$ e desempenho inferior para $\tau = 1.5$.

Porém, pela figura 7.3, fica evidente que o Padrão 2 teve um desempenho superior ao do Padrão 1, o que foi observado para todos os valores de τ. O algoritmo, utilizando o Padrão 2, resolve 19% e 14% dos problemas para $\tau = 1$ e $\tau = 2$ respectivamente, com apenas 10 avaliações da função. Além disso o método mostrou-se bastante robusto, utilizando o Padrão 2, resolvendo 95% dos problemas com até 10^5 avalaliações da função, contra 88% utilizando o Padrão 1 . O bom desempenho do Padrão 2 também pode ser observado pelo fato de que, com apenas 1000 avaliações de função, resolvemos 80% dos problemas.

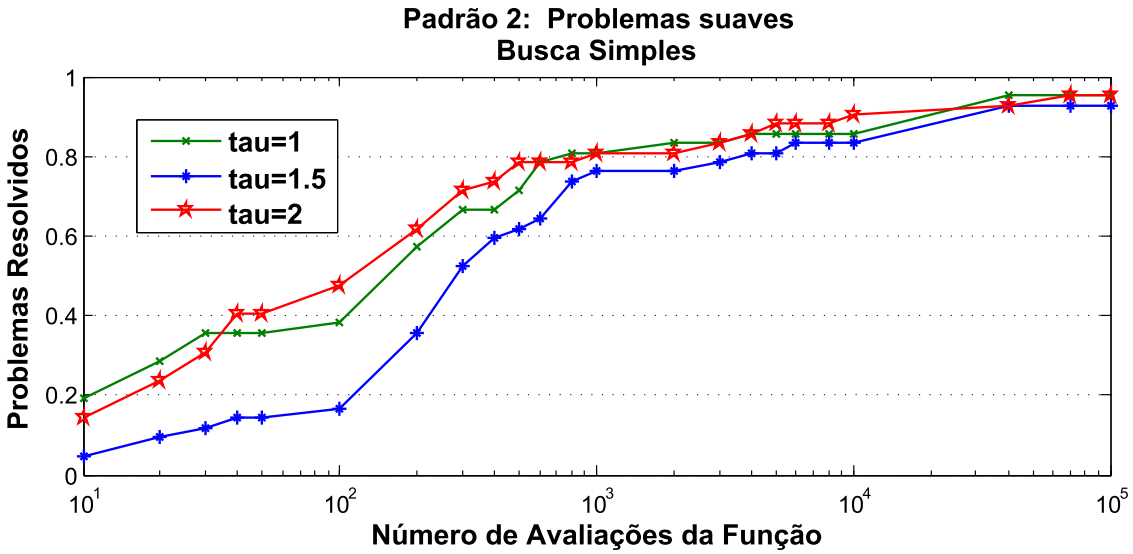


Figura 7.2: Atualização do tamanho do passo: Padrão 2 - Busca Simples

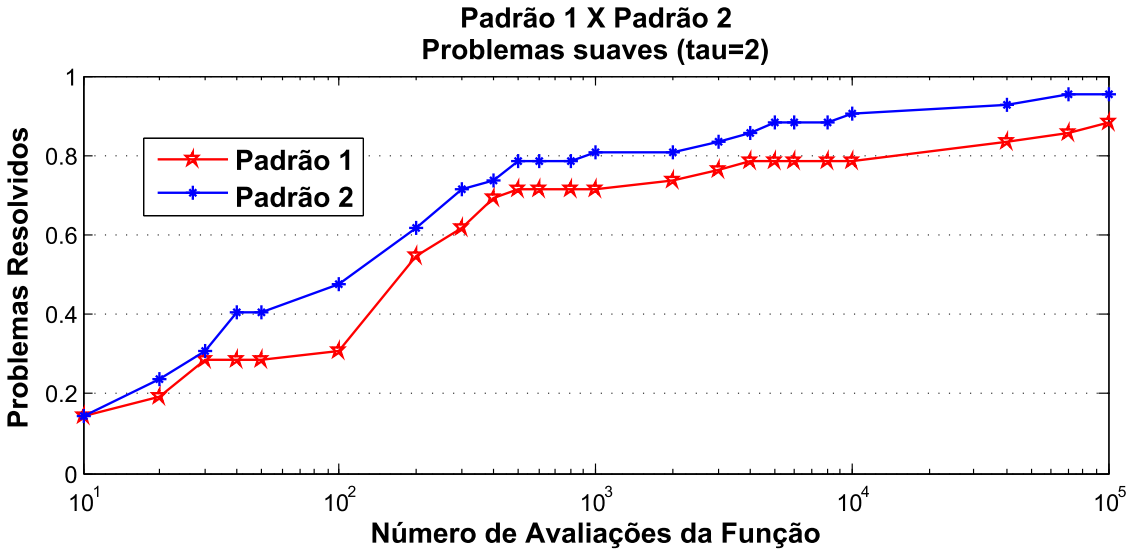


Figura 7.3: Padrão 1 × Padrão 2 - Busca Simples ($\tau = 2$)

7.1.2 Busca Completa

Observe, pela Figura 7.4, que utilizando o Padrão 1, ao contrário do que esperávamos, o método não ficou mais robusto do que utilizando busca simples, uma vez que, com 10^5 avaliações da função, o algoritmo resolve 83%, 86% e 88%, para $\tau = 1$, $\tau = 1.5$ *e* $\tau = 2$ respectivamente.

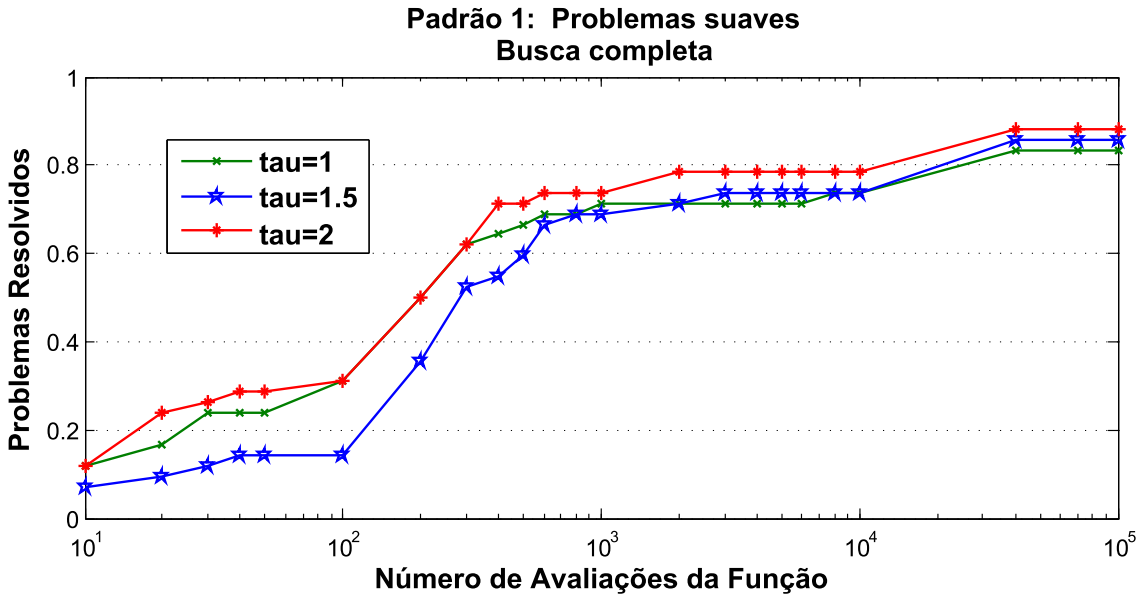


Figura 7.4: Atualização do tamanho do passo: Padrão 1 - Busca Completa

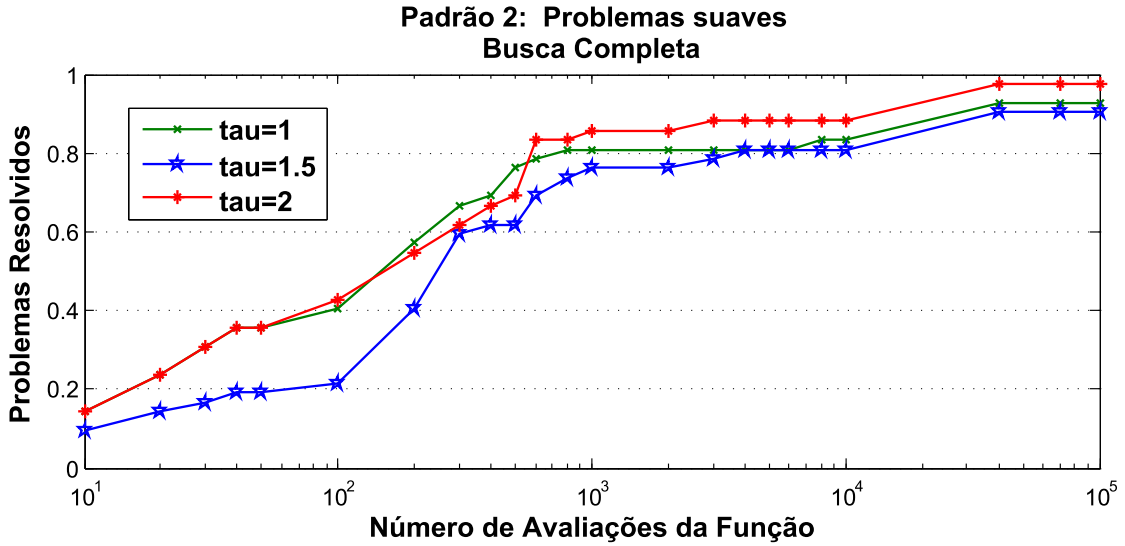


Figura 7.5: Atualização do tamanho do passo: Padrão 2 - Busca Completa

Já o Padrão 2 novamente apresentou um excelente resultado, sendo, como esperado, mais robusto: são resolvidos 97.6% dos problemas com $\tau = 2$, como pode ser observado na Figura 7.5.

Vale destacar ainda que, dos 42 problemas suaves, 41 foram considerados resolvidos segundo os critérios que indicamos. O único problema que foi considerado como não resolvido estava a uma distância de 4.812774×10^{-7} do ponto ótimo reportado, e ao atingir o número máximo de avaliações da função permitido tínhamos $\alpha_k = 1.525879 \times 10^{-5}$. Isso indica que, se permitirmos um número maior de avaliações de função, este

problema provavelmente também atingiria a precisão que estipulamos, levando a 100% dos problemas suaves resolvidos, com uma precisão que pode ser considerada alta.

Este resultado indica que o padrão que propomos está bem definido, sendo possível resolver 100% dos problemas suaves testados.

Comparando o desempenho do Padrão 1 com o Padrão 2, utilizando busca completa, pela Figura 7.6, novamente é possível ver claramente o desempenho superior do Padrão 2, resolvendo 88% dos problemas, com 3000 avaliações de função, a mesma porcentagem que o Padrão 1 atinge com 10^5 avaliações de função.

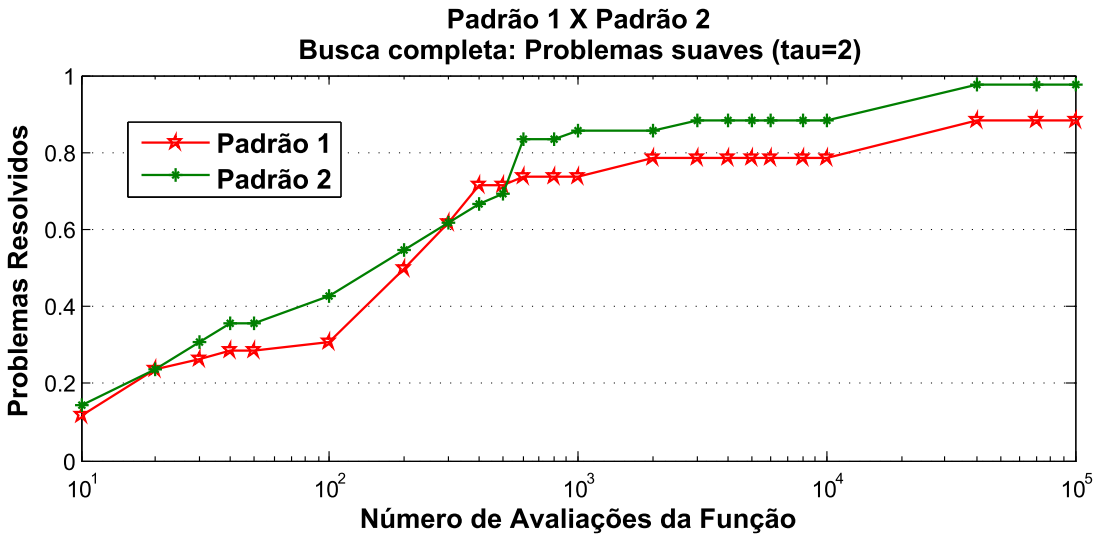


Figura 7.6: Padrão 1 × Padrão 2 - Busca Completa ($\tau = 2$)

7.1.3 Busca Simples Ordenando o Padrão

Ao utilizar busca simples ordenando o Padrão esperávamos que o número de avaliações da função fosse diminuir, mas observando as Figuras 7.7 e 7.8, percebemos que, para ambos os padrões, o desempenho foi equivalente ao observado na busca simples. Esse resultado indica que, para problemas suaves, a estratégia de ordenar o Padrão que utilizamos não interferiu no desempenho do método, o qual apresentou praticamente o mesmo desempenho que observamos utilizando apenas busca simples.

Pela Figura 7.9, fica fácil perceber que de fato o desempenho de ambos os padrões foi o mesmo que observamos na Figura 7.3, quando utilizamos apenas busca simples sem a estratégia de ordenar o padrão.

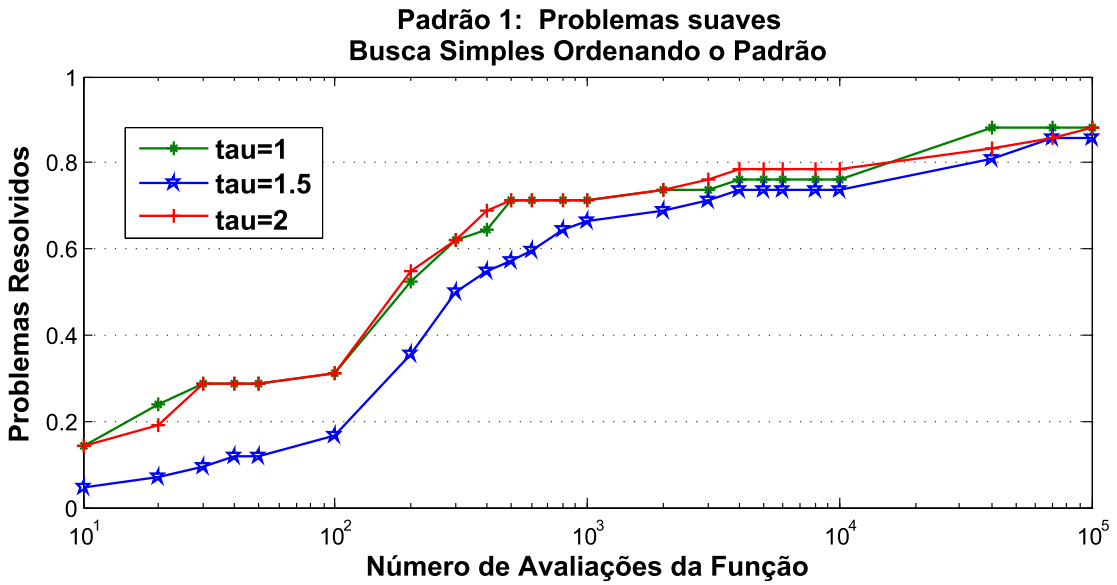


Figura 7.7: Atualização do tamanho do passo: Padrão 1 - Busca Simples Ordenando o Padrão

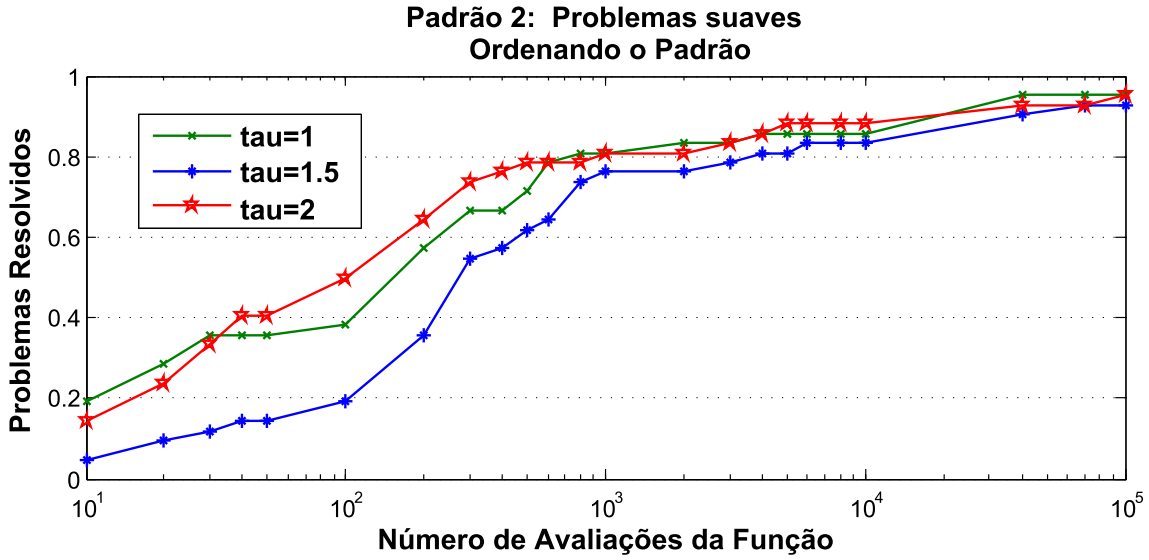


Figura 7.8: Atualização do tamanho do passo: Padrão 2 - Busca Simples Ordenando o Padrão

7.1.4 Comparação entre as Estratégias de Busca

Tendo analisado o desempenho dos Padrões 1 e 2 para todas as estratégias de busca e atualização do tamanho do passo, vejamos agora qual a melhor estratégia para problemas suaves.

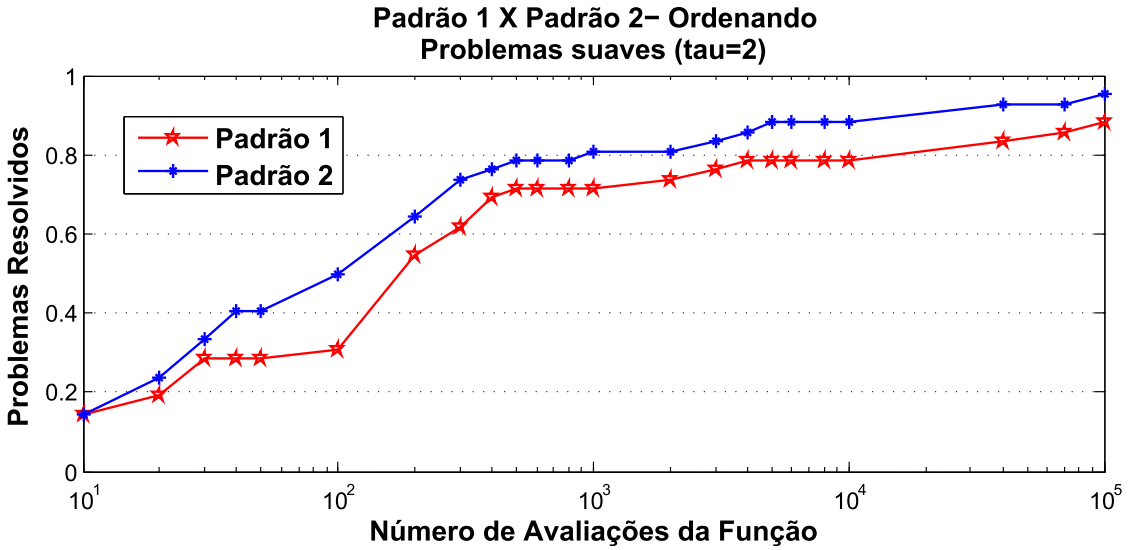


Figura 7.9: Padrão 1 × Padrão 2 - Busca Simples Ordenando o Padrão ($\tau = 2$)

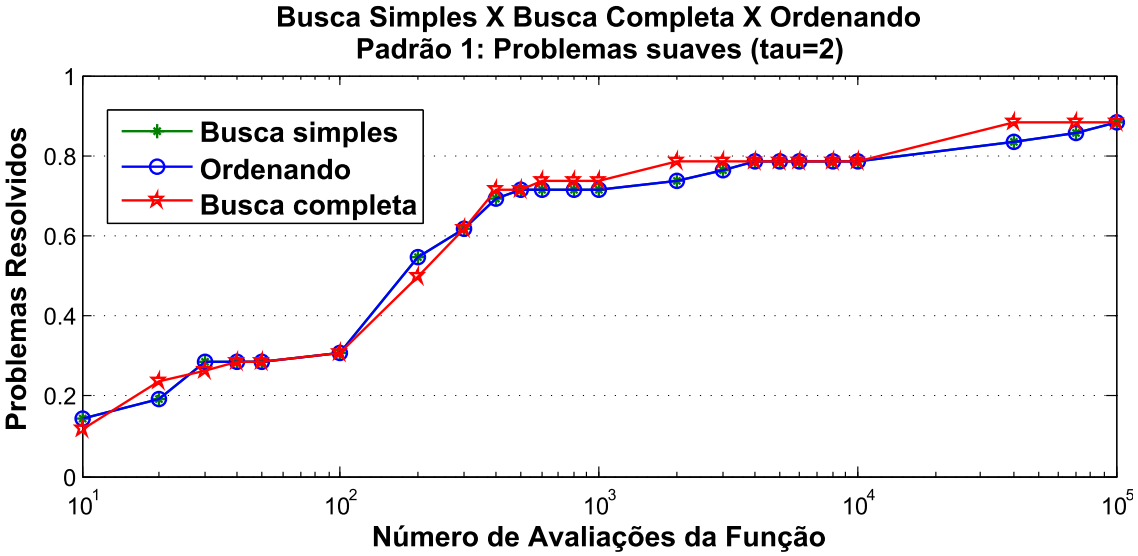


Figura 7.10: Padrão 1 - Estratégias de Busca ($\tau = 2$)

Em praticamente todos os testes, observamos que, usando os parâmetros $\tau = 1$ e $\tau = 2$, o método apresentou um desempenho equivalente, exceto para busca completa que, com $\tau = 2$, mostrou mais robustez. Sendo assim, nas Figuras 7.10 e 7.11, fixamos $\tau = 2$ e plotamos o desempenho do método variando as estratégias de busca.

Observe, pela Figura 7.10, que o Padrão 1 teve praticamente o mesmo desempenho para todas as estratégias de busca. Já com o Padrão 2, vemos pela Figura 7.11 que a busca simples foi mais eficiente, resolvendo mais problemas até aproximadamente 800 avaliações da função. A partir daí o desempenho do método com busca completa fica melhor do que

com busca simples e se mostra mais robusto.

Deste modo, a melhor estratégia para problemas suaves é utilizar o Padrão 2 com busca completa e $\tau = 2$, se dispomos de mais de 800 avaliações da função. Porém, se queremos eficiência, resolvendo os problemas em menos de 800 avaliações da função, utilizamos o Padrão 2 com busca simples e $\tau = 2$.Vale ressaltar mais uma vez que o padrão que propomos foi o melhor em todas as estratégias de busca.

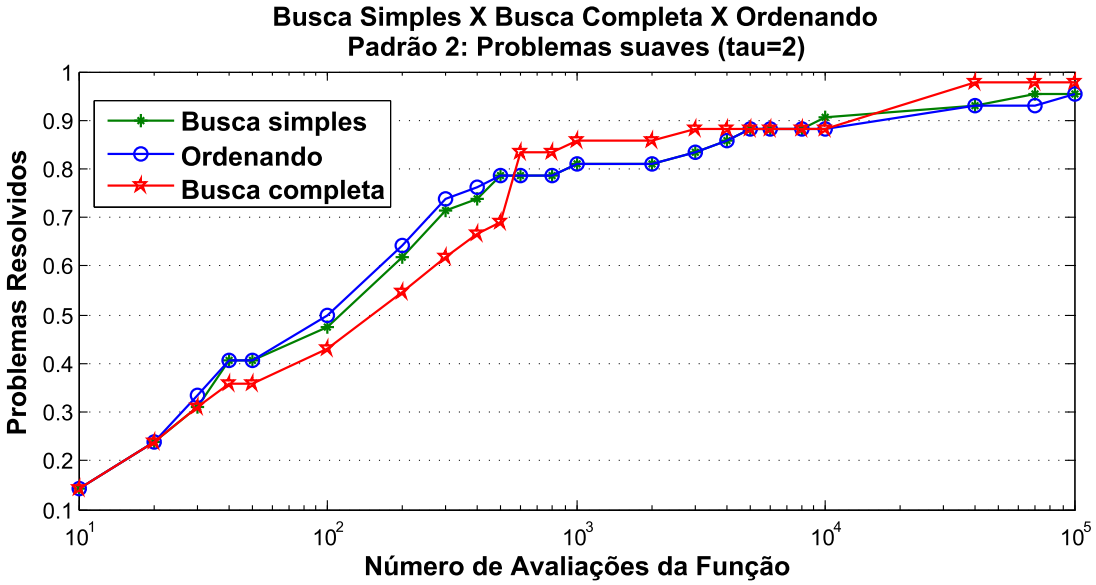


Figura 7.11: Padrão 2 - Estratégias de Busca ($\tau = 2$)

7.2 Problemas Não Suaves

O conjunto de problemas não suaves que testamos é composto por todos os 15 problemas do tipo minimax com restrições lineares de [15], ou seja, problemas da forma:

$$(P)\begin{cases} min & f(x) \\ s.a & l \leq Ax \leq u, \end{cases}$$

onde $f(x) = max\{f_1(x), f_2(x), \ldots, f_p(x)\}$, $f_i : \mathbb{R}^n \to \mathbb{R}$, $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $l, u \in \mathbb{R}^m$.

Perceba que, além da função objetivo ser não suave, seu custo de avaliação é equivalente ao custo de avaliar p funções, sendo, portanto uma função de alto custo dependendo da magnitude de p.

Trabalhamos com problemas minimax inspirados no trabalho de G. Liuzzi, S. Lucide e M. Sciandrone em [13], que desenvolveram um método sem derivadas aplicado a problemas minimax irrestritos ou com restrições lineares.

No Método de Busca Padrão, o problema pode ser abordado de duas formas. A primeira, utilizando a função objetivo da forma que ela está definida, ou seja, diretamente,

Problemas Minimax						
Prob.	n	p	# Desigualdades	# Igualdades	Lim. Inf.	Lim. Sup.
1	2	3	1	0	0	0
2	2	3	1	0	0	0
3	2	3	1	0	1	0
4	2	3	1	0	1	0
5	7	163	8	0	2	1
6	6	3	15	0	0	0
7	8	3	0	1	8	0
8	10	6	3	0	0	0
9	20	14	4	0	0	0
10	9	124	4	0	0	0
11	20	38	0	0	10	0
12	10	9	4	1	10	10
13	7	13	2	0	7	7
14	8	4	3	0	8	8
15	16	19	1	0	16	16

Tabela 7.2: Problemas Minimax

sem nenhum modelo, pois apesar da teoria de convergência do método ser para problemas suaves, é permitido o uso de função não suave, desde que possamos avaliá-las nos pontos visitados pelo algoritmo. A outra forma é utilizando um modelo suave para a função objetivo, como é feito pelos autores em [13], onde eles provam a convergência do método para problemas suaves e utilizam um modelo exponencial para a função objetivo. Assim conforme diminuímos o fator de ajuste do modelo, este fornece uma aproximação cada vez melhor para a função objetivo original do problema.

Apresentamos inicialmente os resultados obtidos utilizando diretamente a função objetivo, e comentaremos mais adiante a respeito do uso de modelos suaves para os problemas minimax. Os resultados abaixo foram organizados da mesma forma que fizemos para os problemas suaves.

7.2.1 Busca Simples

Obviamente, o desempenho do método para problemas não suaves é bem inferior ao observado com os problemas suaves. Mas levando em consideração que esse tipo de problema não pode ser abordado diretamente pelos métodos tradicionais, uma vez que as derivadas não estão disponíveis, conseguimos alguns resultados satisfatórios, dentro das limitações impostas pelo problema.

Pelas Figuras 7.12 e 7.13, vemos que, ao contrário do que aconteceu com os problemas suaves, o desempenho do método foi superior para $\tau = 1.5$, atingindo 53.3% dos

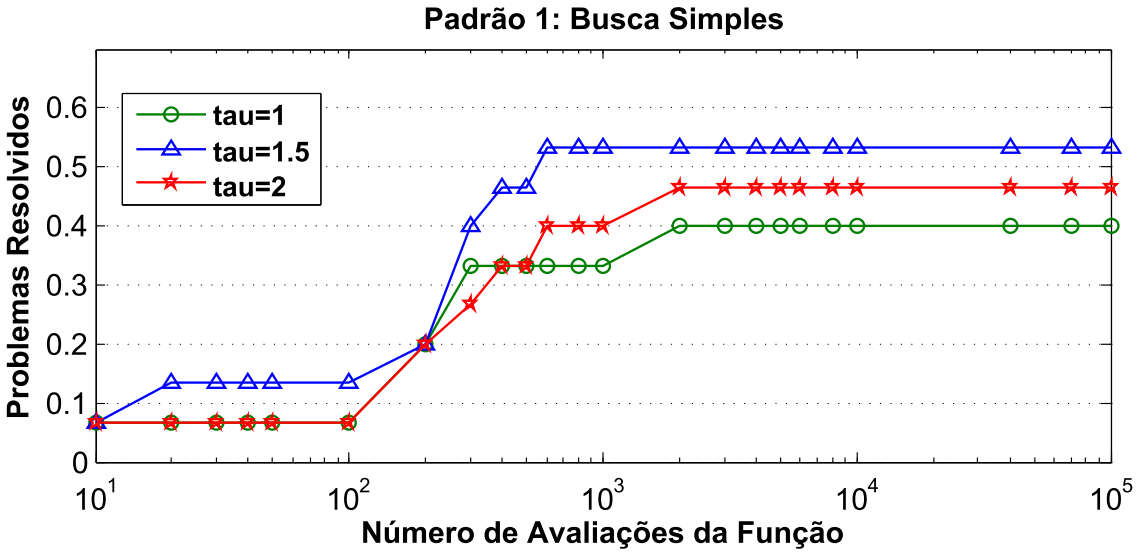


Figura 7.12: Problemas não suaves: Padrão 1 - Busca Simples

problemas resolvidos, sendo que, para os demais parâmetros, o algoritmo resolveu apenas 40% e 46.75% para $\tau = 1$ e $\tau = 2$ respectivamente, o que mostra que o comportamento do método muda drasticamente quando estamos trabalhando com problemas não suaves.

Vale destacar ainda que, para $\tau = 1.5$, com apenas 300 avaliações da função resolvemos 40% dos problemas, com 400 avaliações da função resolvemos 46.7% dos problemas e com 600 avaliações da função resolvemos 53.3% dos problemas propostos.

É importante deixar claro também que este resultado foi obtido considerando a precisão de 10^{-7} que estipulamos. Se considerarmos a precisão de 10^{-4}, atingimos 60% dos problemas resolvidos, como podemos verificar nas Tabelas A.7 e A.8. Os outros 40% dos problemas ficaram longe da solução reportada na referência.

Para o Padrão 2, pela figura 7.13, também observamos um bom comportamento para o parâmetro $\tau = 1.5$, aproximadamente o mesmo comportamento do primeiro padrão. Para $\tau = 2$, como podemos observar melhor pela figura 7.14, o desempenho do Padrão 2 foi superior ao do Padrão 1, e com $\tau = 1$ o Padrão 2 também foi mais eficiente, embora tenha atingido a mesma robustez, de apenas 40% dos problemas resolvidos.

Observe que a eficiência obtida pelo Padrão 2 foi muito boa. Neste caso, o método resolveu 13.3% com 10 avaliações da função, apenas cerca de 5% menor do que a observada para os problemas suaves. Isso mostra que, apesar de não ser robusto para problemas não suaves, o método é eficiente resolvendo os problemas com poucas avaliações da função, a grande maioria com menos de 1000 avaliações da função. Isto é uma vantagem, levando em consideração que o custo de avaliar a função destes problemas é alto.

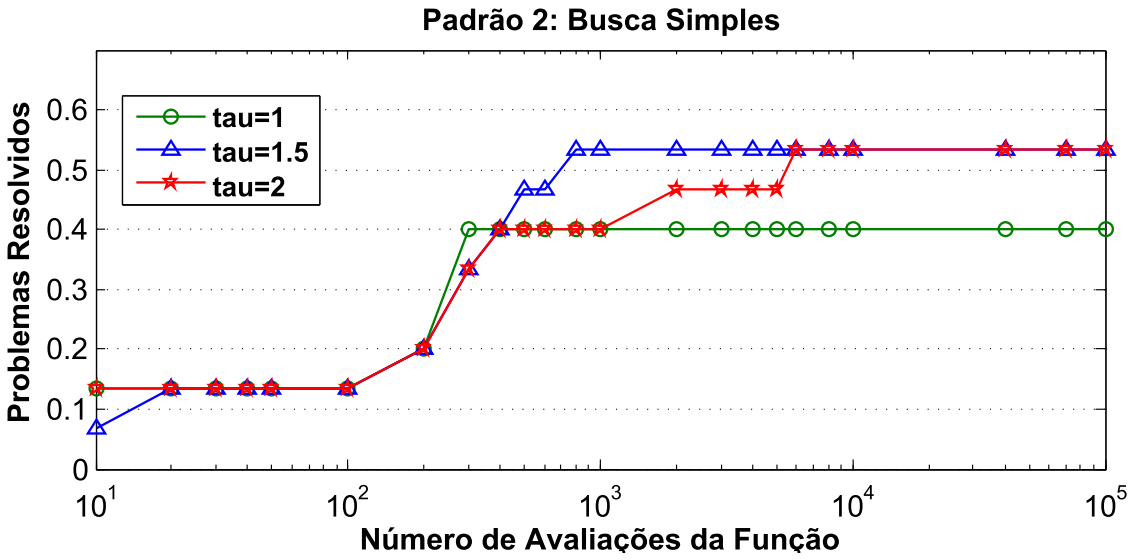


Figura 7.13: Problemas não suaves: Padrão 2 - Busca Simples

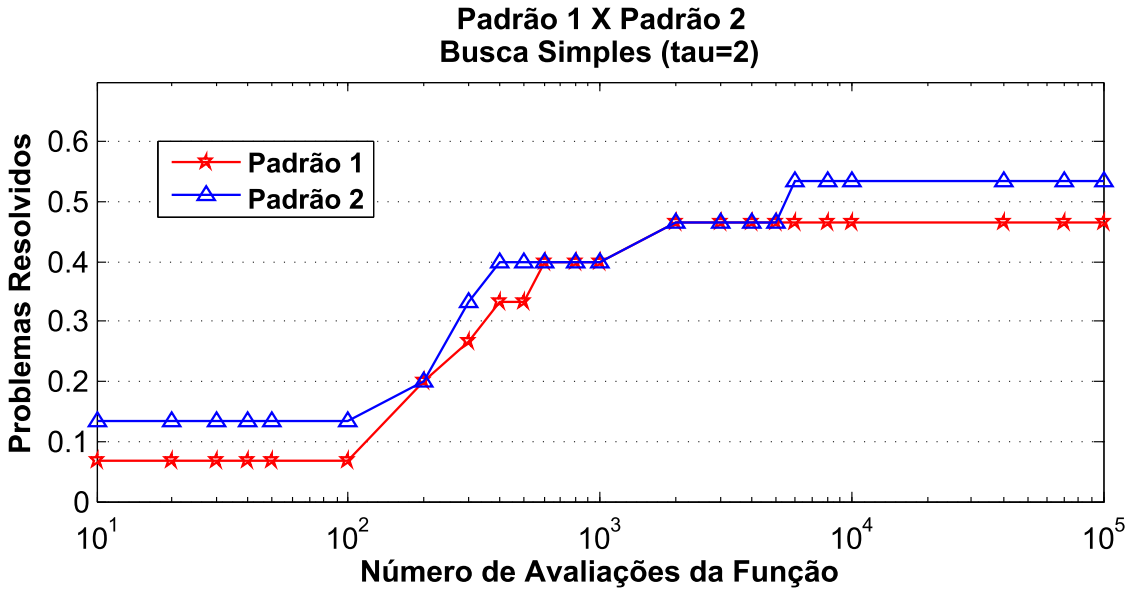


Figura 7.14: Padrão 1 $\times$ Padrão 2: Problemas não suaves - Busca Simples ($\tau = 2$)

7.2.2 Busca Completa

Ao contrário do que observamos no primeiro conjunto de testes, contendo apenas problemas suaves, para esse segundo conjunto de testes vemos que o desempenho do Padrão 2 foi o mesmo do observado com busca simples. Já utilizando o Padrão 1 melhoramos a robustez do método atingindo 60% dos problemas resolvidos com $\tau = 1.5$. Para $\tau = 1$, a eficiência e a robustez foi melhor e, para $\tau = 2$, o método ganhou em eficiência.

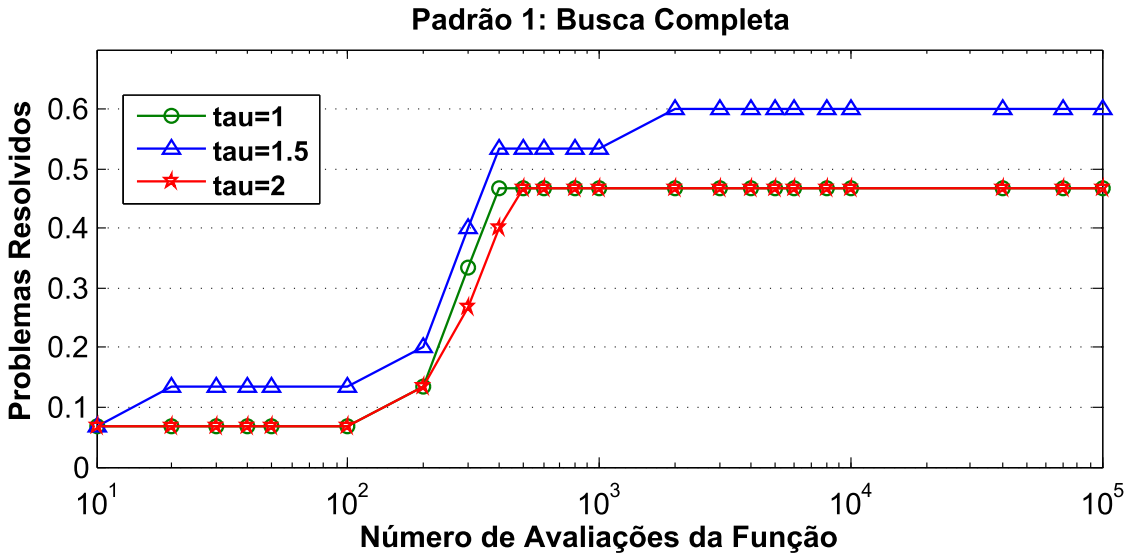


Figura 7.15: Problemas não suaves: Padrão 1 - Busca Completa

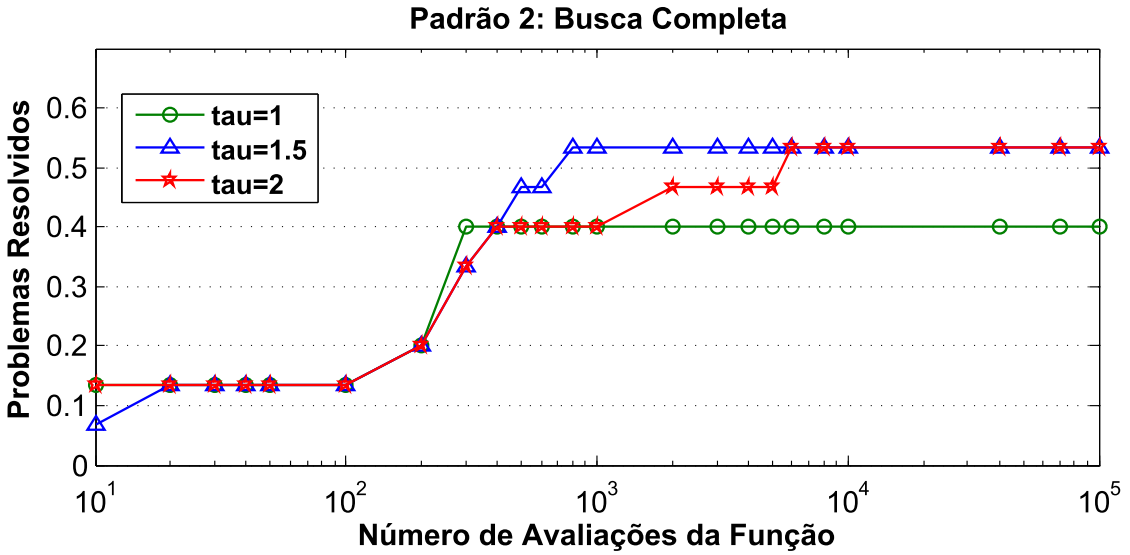


Figura 7.16: Problemas não suaves: Padrão 2 - Busca Completa

Veja, pela Figura 7.17, que de fato o Padrão 1 se mostrou mais robusto que o Padrão 2, com busca completa e $\tau = 1.5$, atingindo uma robustez razoável, levando em consideração as dificuldades dos problemas que estamos resolvendo. Considerando uma precisão da ordem de 10^{-2}, o Padrão 2 também atinge a robustez de 60% dos problemas resolvidos .

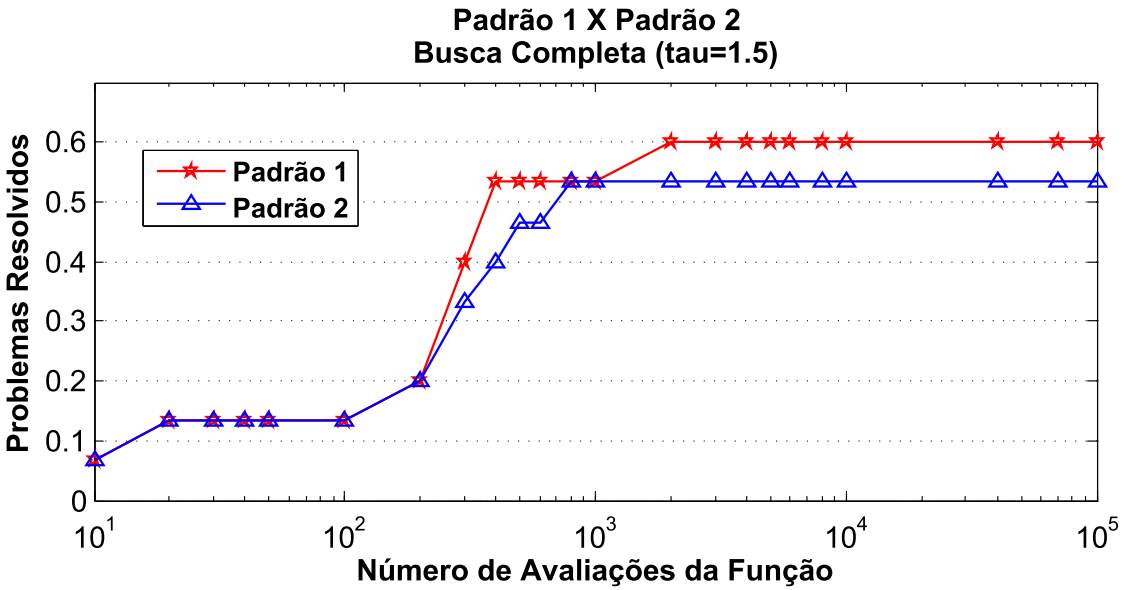


Figura 7.17: Padrão 1 × Padrão 2 - Busca Completa ($\tau = 1.5$)

7.2.3 Busca Simples Ordenando o Padrão

Utilizando busca simples com a estratégia de ordenar o padrão, observamos um comportamento do método semelhante ao observado com os problemas suaves, ou seja, a estratégia de ordenar o Padrão não alterou praticamente em nada o desempenho do método, comparado à busca simples sem ordenar o padrão.

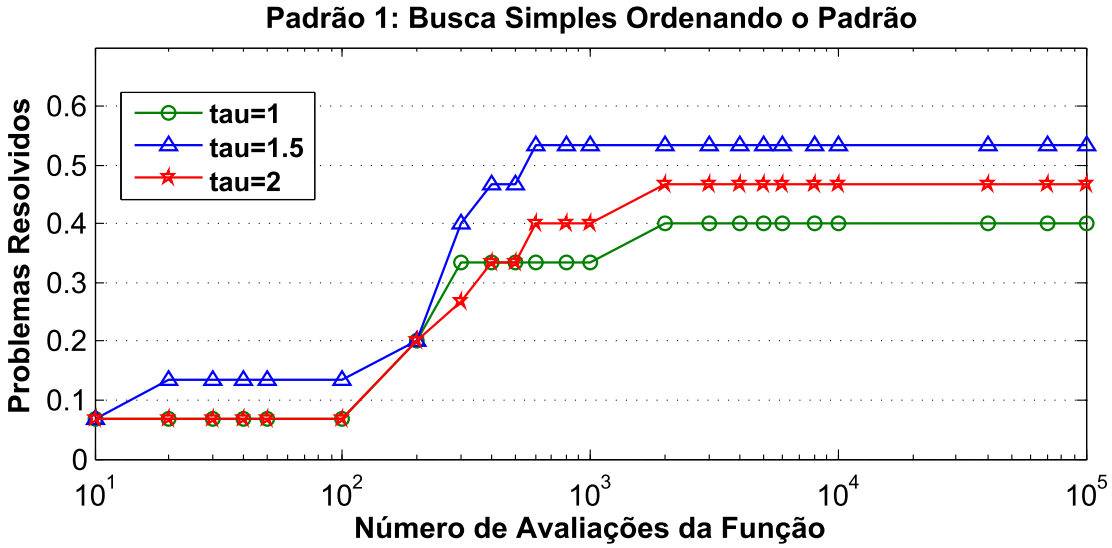


Figura 7.18: Problemas não suaves: Padrão 1 - Busca Simples Ordenando o Padrão

Observando as Figuras 7.18 e 7.19, vemos que a curva dos gráficos é praticamente a mesma observada nas Figuras 7.12 e 7.13, a qual utiliza apenas busca simples.

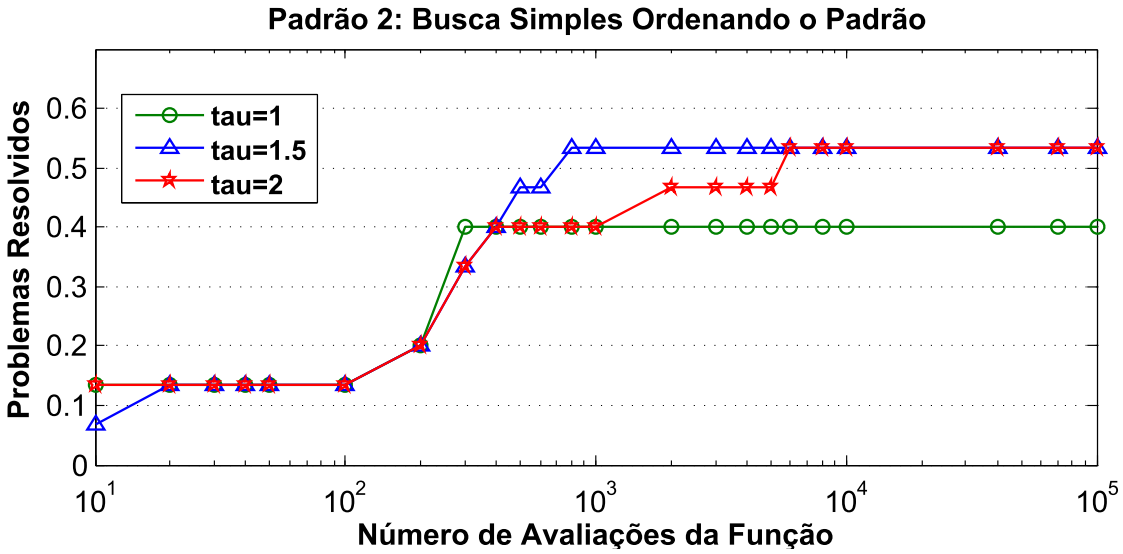


Figura 7.19: Problemas não suaves: Padrão 2 - Busca Simples Ordenando o Padrão

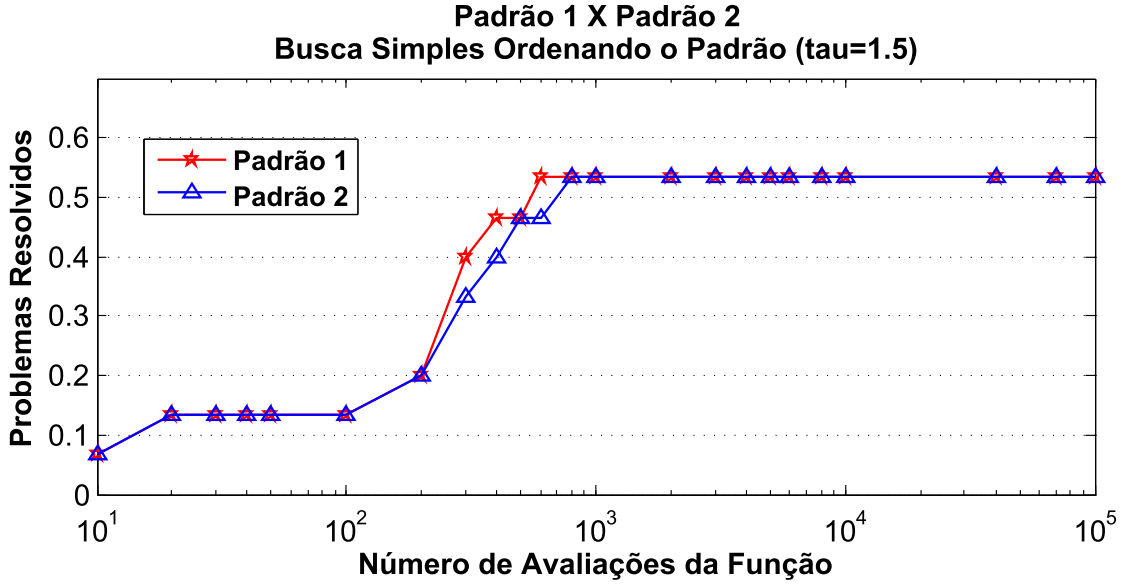


Figura 7.20: Padrão 1 $\times$ Padrão 2: Problemas não suaves - Busca Simples Ordenando o Padrão ($\tau = 1.5$)

Comparando o desempenho dos dois padrões, para esta estratégia de busca, vemos pela Figura 7.20 que, assim como utilizando busca simples sem ordenar o padrão, os dois padrões obtiveram praticamente o mesmo desempenho para $\tau = 1.5$, que foi o melhor valor para o parâmetro, tendo o Padrão 1 superado o Padrão 2 apenas em alguns casos. Porém para os outros dois valores de τ é possível observar um desempenho superior do Padrão 2, ganhando em eficiência e robustez para $\tau = 2$, e em eficiência para $\tau = 1$.

7.2.4 Comparação entre as Estratégias de Busca

Tendo analisado todas as estratégias de busca variando o parâmetro de atualização do tamanho do passo, vemos que o método apresentou os melhores resultados para $\tau = 1.5$, para ambos os padrões.

Para o Padrão 1, observando a Figura 7.21, é evidente que a busca completa apresentou o melhor resultado, sendo a mais eficiente e também a mais robusta. É fácil ver também que a busca simples e a busca simples ordenando o padrão, são equivalentes. Já para o Padrão 2, segundo a Figura 7.22, fica claro que o desempenho do método foi equivalente para todas as estratégias de busca.

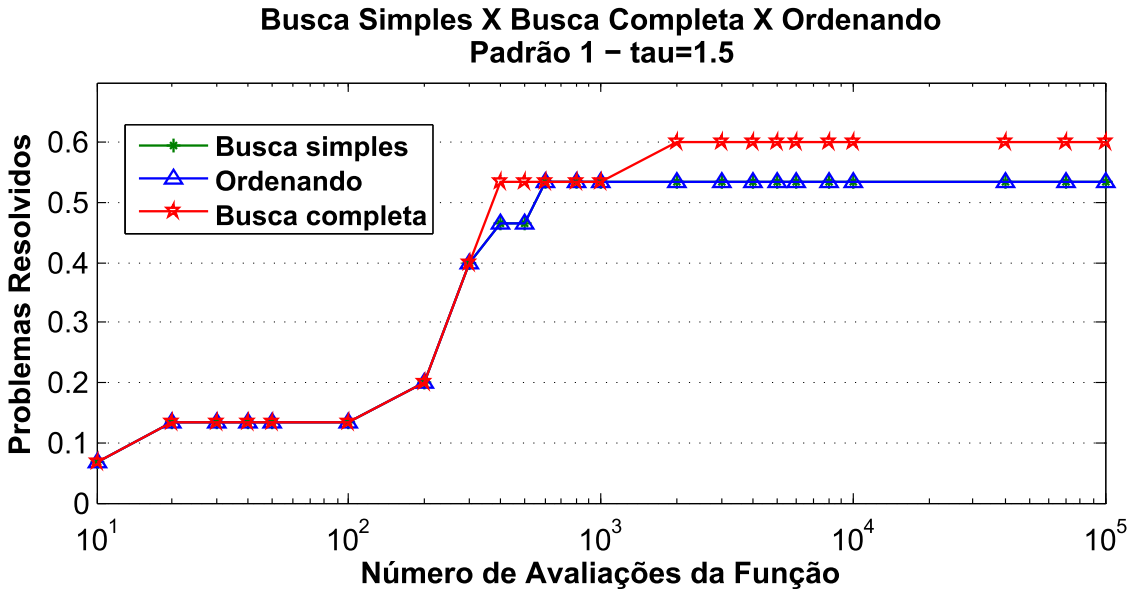


Figura 7.21: Problemas não suaves: Padrão 1 - Estratégias de Busca ($\tau = 1.5$)

É interessante destacar que a estratégia de busca usual do Método de Busca Padrão, ou seja, busca simples com $\tau = 1$, não apresentou um bom desempenho para os problemas não suaves, mostrou-se pouco robusto em ambos os padrões, resolvendo apenas 40% dos problemas, e para o Padrão 1 também é pouco eficiente. Logo, as estratégias que propomos foram importantes para melhorar o desempenho do método para esse tipo de problema.

7.3 Modelo Suave para Problemas Minimax

Pelos testes que acabamos de apresentar, vemos que, embora não possamos resolver os problemas minimax com a mesma eficiência e robustez que conseguimos para problemas suaves, para os quais temos teoria de convergência, obtemos resultados razoáveis. Podemos dizer que este resultado é satisfatório, principalmente levando em consideração que esse tipo de problema não pode ser abordado diretamente por nenhum dos métodos tradicionais, que fazem uso de derivadas.

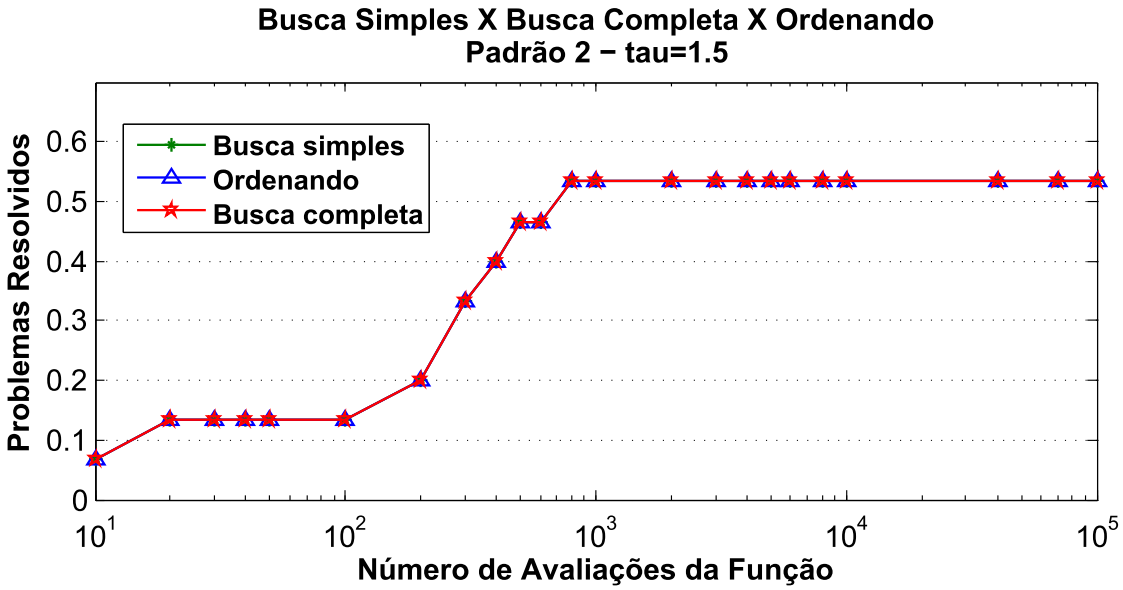


Figura 7.22: Problemas não suaves: Padrão 2 - Estratégias de Busca ($\tau = 1.5$)

No entanto, existem muitos métodos voltados para a resolução deste tipo de problema. Todos eles utilizam um modelo suave para a função objetivo. Sendo assim, considerando que o Método de Busca Padrão possui teoria de convergência para problemas suaves, o uso de um modelo suave para a função objetivo poderia melhorar o desempenho do método, principalmente na robustez. Não esperávamos melhora na eficiência devido à necessidade de atualização do parâmetro de ajuste do modelo, o que evidentemente conduz a necessidade de avaliar a função objetivo mais vezes.

Utilizamos o mesmo modelo proposto pelos autores em [13], um modelo exponencial, o qual é utilizado na grande maioria dos trabalhos estudados que apresentam métodos para problemas minimax, sejam eles com ou sem derivadas. O modelo utilizado é dado por:

$$f(x,\mu) = f(x) + \mu ln \sum_{i=1}^{p} exp\left(\frac{f_i(x) - f(x)}{\mu}\right),$$

onde o parâmetro de ajuste do modelo μ deve tender a zero, e portanto deve ir diminuindo durante o processo iterativo.

Ao contrário do que esperávamos, não conseguimos bons resultados utilizando esse modelo, perdemos não somente em eficiência como também em robustez. Os resultados que apresentamos, onde não utilizamos um modelo para a função, foram muito melhores, uma vez que, com o modelo mantendo-se a precisão que estipulamos da ordem de 10^{-7}, pouquissímos problemas são resolvidos, como podemos observar pela figura 7.23.

No entanto, apesar de acreditar que poderíamos resolver um número maior de problemas, também era esperado que o resultado fosse menos preciso, uma vez que não estamos utilizando a função original, a própria representação da função através de um

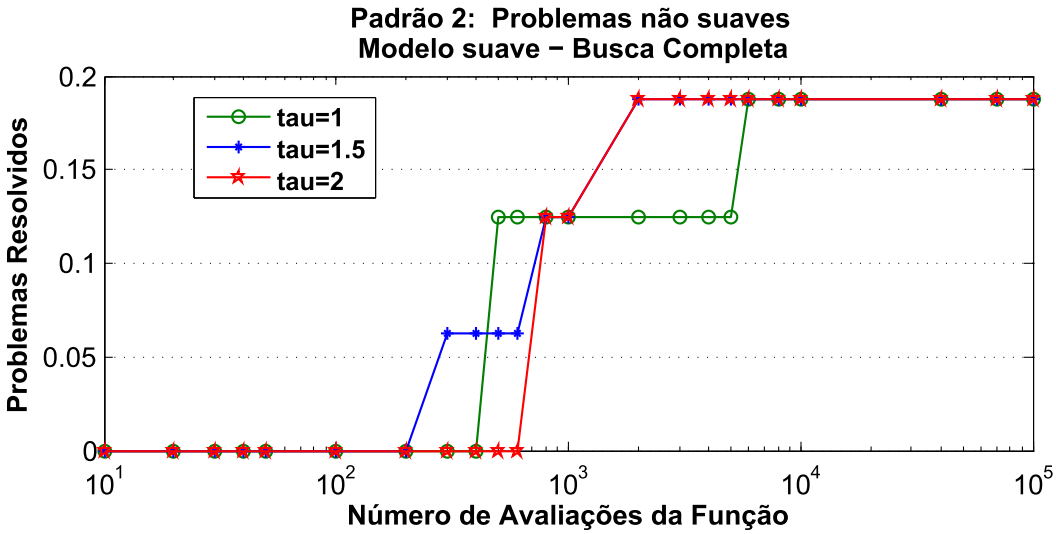


Figura 7.23: Busca completa utilizando o modelo exponencial - precisão 10^{-7}

modelo carrega um erro, e o erro da representação obviamente interfere na qualidade do resultado final obtido.

Sendo assim, relaxamos as exigências que estipulamos para considerar que o problema foi resolvido, ou seja, em todos os testes até aqui o problema era considerado resolvido se:

$$f(\bar{x}) - f(x^\star) \leq 10^{-7} \;\; ou \;\; \|\bar{x} - x^\star\| \leq 10^{-7},$$

onde $\bar{x}$ é o ponto liberado como solução pelo nosso algoritmo e $x^\star$ é o ponto ótimo conhecido.

Relaxando essas exigências, de modo a analisar se o uso do modelo, embora não forneça soluções precisas, fornece boas aproximações para a solução, consideramos que o problema foi resolvido se:

$$f(\bar{x}) - f(x^\star) \leq 10^{-2} \;\; ou \;\; \|\bar{x} - x^\star\| \leq 10^{-2}.$$

Observe, pela Figura 7.24, que mesmo relaxando a precisão exigida, não conseguimos bons resultados: não resolvemos mais do que 37% dos problemas propostos. Levando em consideração que utilizando a mesma estratégia, sem o modelo, resolvemos 53.3% dos problemas com a precisão de 10^{-7} na solução, e que com a mesma precisão resolvemos menos de 20% dos problemas utilizando o modelo, como pode ser verificado na Figura 7.23, podemos afirmar que o uso do modelo não é uma boa estratégia, uma vez que obtemos resultados muito melhores sem o modelo.

É importante destacar que em [13], o algoritmo proposto pelos autores é aplicado a problemas irrestritos e com restrições lineares. No entanto, entre os 24 problemas apresentados, apenas 5 deles possuem restrições lineares, e para apenas 2 desses problemas o resultado obtido estaria dentro da precisão de 10^{-7} com a qual trabalhamos. Considerando

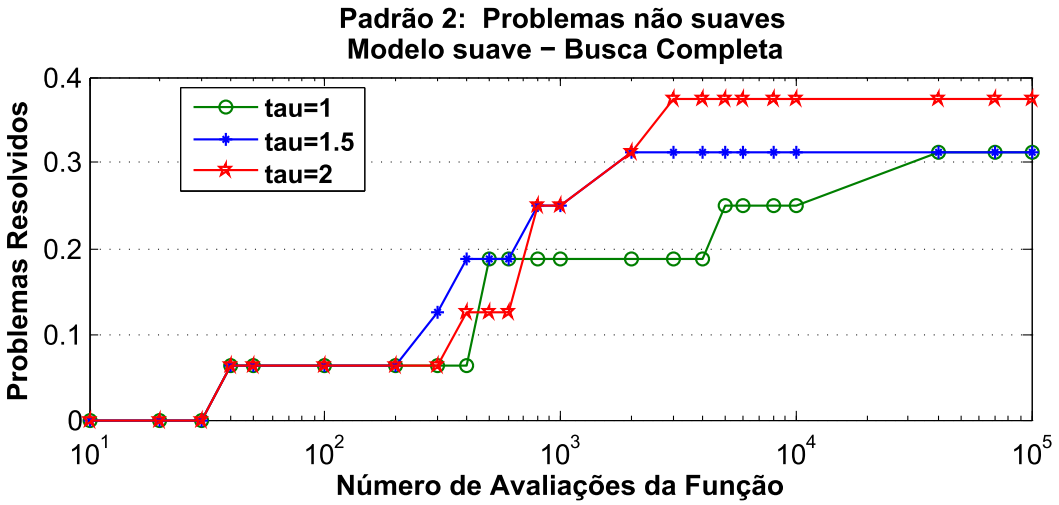


Figura 7.24: Busca completa utilizando o modelo exponencial - precisão 10^{-2}

a precisão de 10^{-2}, teríamos 4 problemas, o que mostra que a precisão obtida pelos autores também não foi boa. Além disso, vale frisar que para esses 5 problemas, atingimos a precisão de 10^{-2} para todos eles utilizando o modelo, e usando diretamente a função objetivo atingimos a precisão de 10^{-7}. Isto deixa claro que, apesar de não termos obtidos excelentes resultados, estes foram melhores do que o obtido em muitos trabalhos.

7.4 Desempenho Geral do Método

Na prática, os métodos sem derivadas são aplicados a problemas onde o custo de avaliar a função objetivo é alto ou, mais precisamente, o custo de obter as derivadas, ainda que elas estejam disponíveis, torna inviável o uso dos métodos tradicionais. É comum não possuirmos uma expressão explícita para a função objetivo do problema, sendo assim, muitas vezes não temos informações suficientes a respeito do problema para classificá-lo como suave ou não suave.

Vimos, pelos testes apresentados até agora, que o desempenho do método muda bastante, dependendo da natureza do problema que estamos resolvendo. Ficou claro que a melhor estratégia para problemas suaves nem sempre é uma boa alternativa para problemas não suaves, e vice e versa. Logo, como na prática nem sempre possuímos informações suficientes a respeito do problema para poder escolher a melhor estratégia para resolvê-lo, é importante saber, de modo geral, como o método se comporta variando as estratégias propostas.

Nos resultados apresentados abaixo, o conjunto de testes é composto por todos os problemas que trabalhamos, 62 problemas no total, ou seja, os problemas suaves detalhados na Tabela 7.1, os problemas minimax detalhados na Tabela 7.2, e os problemas adicionais extraídos de [7, 20], detalhados na tabela 7.3, entre os quais temos problemas

não suaves e problemas cuja função objetivo é cara ou não podemos classificá-la como suave ou não suave. Faremos, para esse conjunto mais geral, a mesma análise que fizemos para os outros dois conjuntos de testes, de modo a ter um indicativo sobre qual a melhor estratégia a ser utilizada quando não sabemos o tipo de problema com o qual estamos trabalhando.

Problemas Adicionais					
Prob.	n	# Desigualdades	# Igualdades	Lim. Inf.	Lim. Sup.
1	3	0	0	3	3
2	49	0	15	49	49
3	31	20	11	31	11
4	46	0	31	46	46
5	20	0	15	20	20

Tabela 7.3: Problemas testes adicionais

7.4.1 Busca Simples

Pela Figura 7.25, vemos que utilizando o Padrão 1 e busca simples, atingimos aproximadamente a mesma robustez para todos os valores de τ. Conseguimos resolver cerca de 70% dos problemas com até 10^5 avaliações da função, e utilizando até 10 avaliações da função resolvemos 11% dos problemas, para $\tau = 1$ e $\tau = 2$, e 5% dos problemas para $\tau = 1.5$.

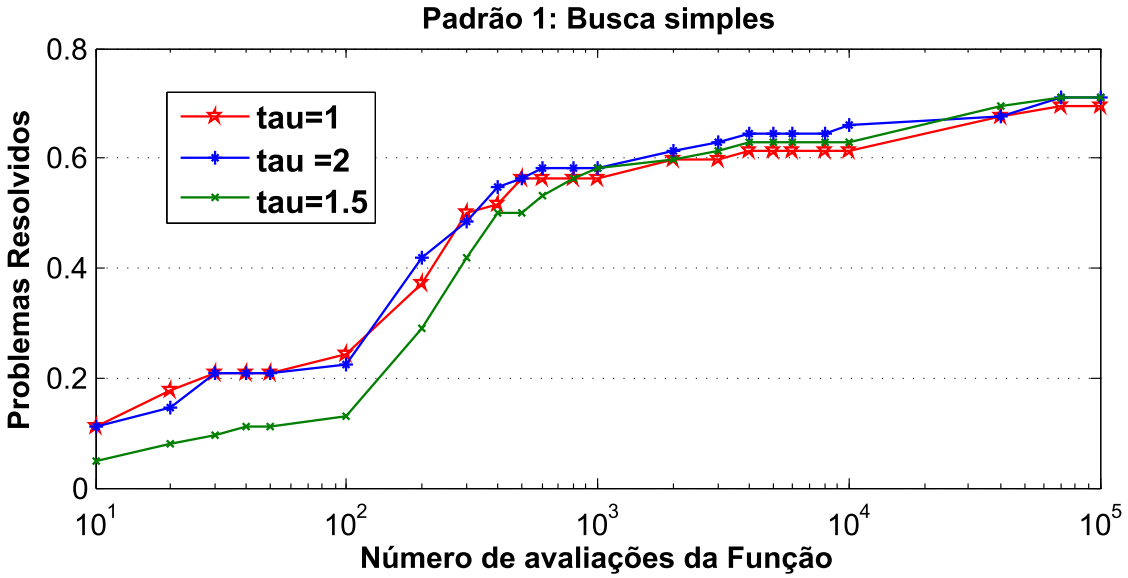


Figura 7.25: Atualização do tamanho do passo: Padrão 1 - Busca Simples

Perceba que, para $\tau = 1.5$, o método é menos eficiente, no entanto atinge a mesma

robustez obtida pelos outros dois parâmetros, o que não aconteceu quando o conjunto de testes tinha apenas problemas suaves. Esse resultado vem do fato que, para problemas não suaves, o desempenho do método para $\tau = 1.5$ foi superior ao observado para os outros dois parâmetros. O mesmo foi verificado para $\tau = 1$, com o qual o método não obteve um bom resultado para problemas não suaves, porém funcionando bem para problemas suaves.

Para o Padrão 2, vemos na Figura 7.26 que o método fica mais robusto, resolvendo 79% dos problemas para $\tau = 2$, 9% a mais que a robustez atingida pelo Padrão 1. Vemos que a eficiência também é maior, resolvendo cerca de 16% dos problemas para $\tau = 1$.

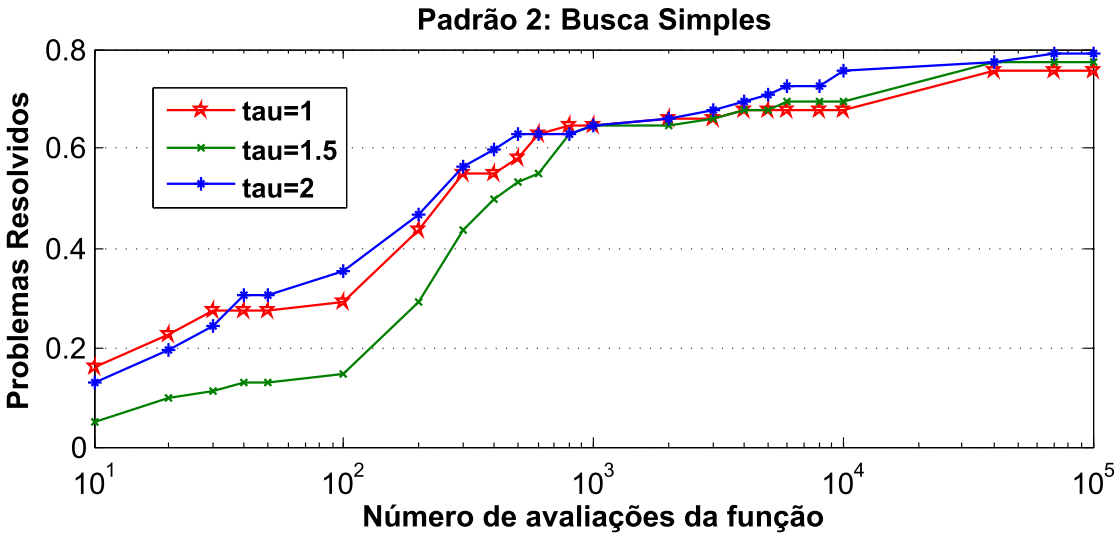


Figura 7.26: Atualização do tamanho do passo: Padrão 2 - Busca Simples

Levando em consideração que o Padrão 2 foi melhor que o Padrão 1 em todas as estratégias, tanto para problemas suaves quanto para os não suaves, era esperado um desempenho superior deste padrão de um modo geral, como pode ser verificado pela figura 7.27. É importante destacar que o desempenho alcançado pelo Padrão 1, considerando até 10^5 avaliações de função, foi atingido pelo Padrão 2 com 5000 avaliações da função. Consideramos este resultado muito bom, tendo em vista que a mesma proporção de problemas resolvidos é atingida pelo Padrão 1 apenas com 70000 avaliações da função, uma diferença muito grande.

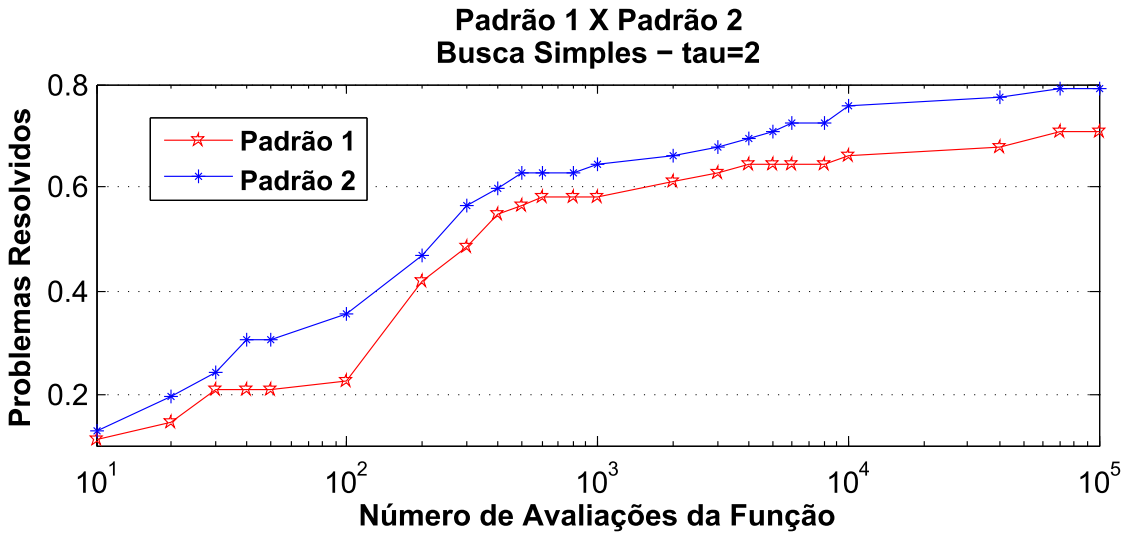


Figura 7.27: Padrão 1 × Padrão 2 - Busca Simples ($\tau = 2$)

7.4.2 Busca Completa

Vimos que, para os problemas suaves com o Padrão 1, a busca completa não deixou o método mais robusto. Isto é o contrário do que era esperado devido ao resultado de convergência mais forte. Como podemos observar pela Figura 7.28, de modo geral a robustez e a eficiência obtidas pelo Padrão 1, com busca completa, é aproximadamente a mesma verificada para busca simples, embora para os problemas não suaves tenhamos observado uma melhora na robustez obtida para $\tau = 1.5$, o que reflete numa melhora pequena para esse parâmetro no caso geral.

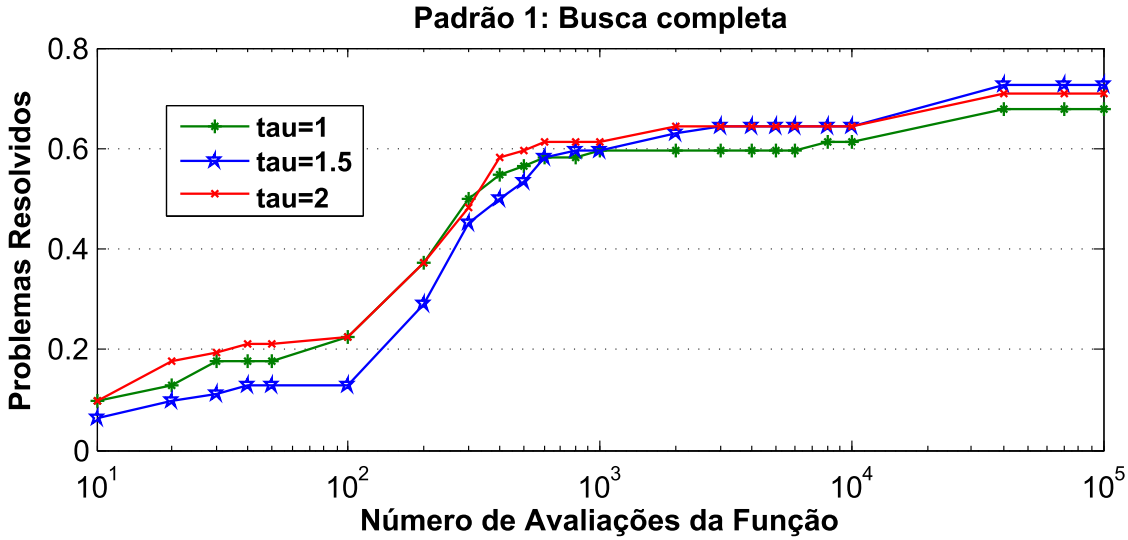


Figura 7.28: Atualização do tamanho do passo: Padrão 1 - Busca Completa

Já para o Padrão 2, assim como observamos para os problemas suaves, a robustez aumenta, cerca de 2% no caso geral, quando utilizamos busca completa, porém a eficiência diminui. Como podemos ver na Figura 7.29, obtemos um melhor desempenho para $\tau = 2$, resolvendo 80.6% dos problemas com 40000 avaliações da função. Para os demais parâmetros resolvemos 77.4% e 75.8% dos problemas para $\tau = 1$ e $\tau = 1.5$ respectivamente. Deste modo, a robustez maior foi atingida somente para $\tau = 2$. Para os demais parâmetros os resultados obtidos foram aproximadamente os mesmos que verificamos para a busca simples, o que indica que o uso da busca completa só é vantajoso se utilizarmos $\tau = 2$. Nos demais casos, a busca simples é melhor, pois é mais eficiente.

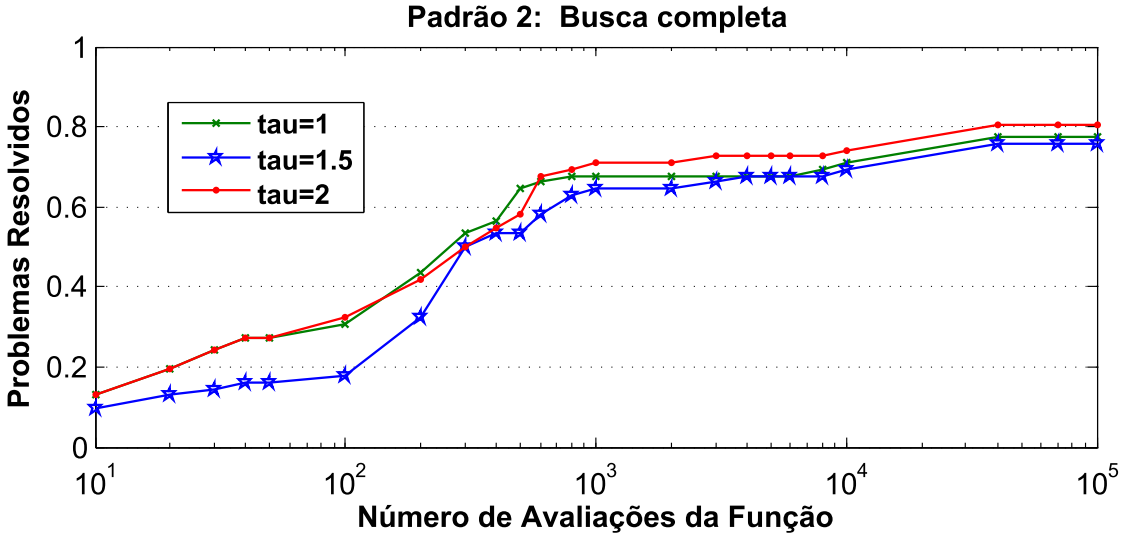


Figura 7.29: Atualização do tamanho do passo: Padrão 2 - Busca Completa

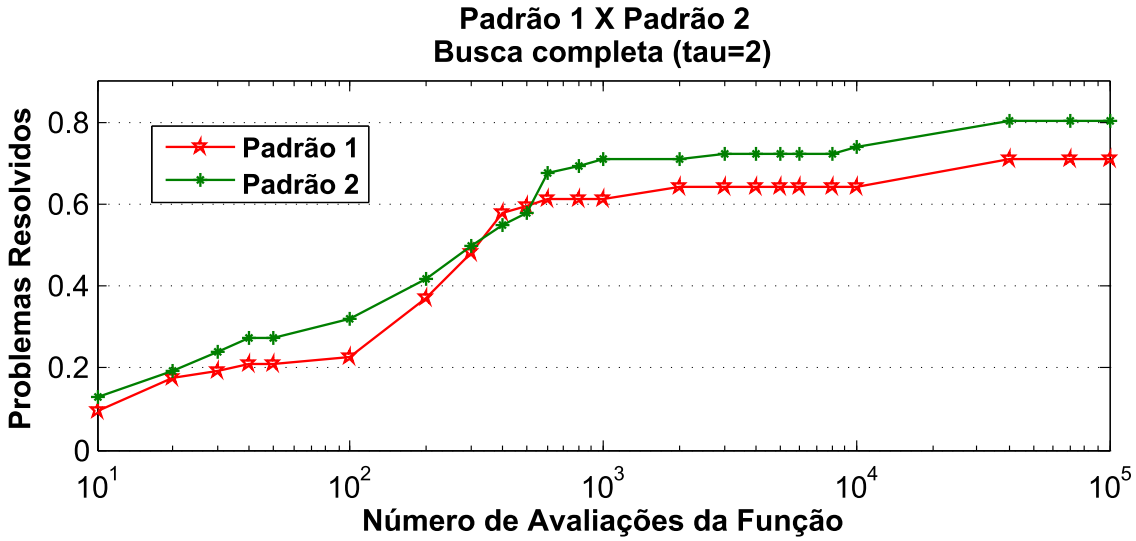


Figura 7.30: Padrão 1 $\times$ Padrão 2 - Busca Completa ($\tau = 2$)

Comparando os desempenhos dos dois padrões vemos, novamente, a superioridade do Padrão 2, que atinge a proporção máxima atingida pelo Padrão 1 com apenas 1000 avaliações da função, como podemos observar na Figura 7.30. Isto indica que a busca completa utilizando o Padrão 2 é mais eficiente que as demais estratégias a partir de 1000 avaliações da função.

7.4.3 Busca Simples Ordenando o Padrão

Nos testes anteriores vimos que a estratégia de ordenar o padrão que propomos é indiferente quando estamos trabalhando com problemas suaves, mas pode ser uma boa estratégia para determinados tipos de problemas. Este é o caso do problema adicional 1, um problema que embora pequeno é de dificil solução, como podemos observar pelas tabelas no apêndice A.

Para este problema em questão, utilizando busca simples e busca completa não atingimos a precisão estipulada para nenhum dos testes, considerando até 10^5 avaliações da função. No entanto, utilizando a estratégia de ordenar o Padrão que propomos obtemos uma precisão da ordem de 10^{-17} realizando menos de 3000 avalições da função.

Esperamos assim que no caso geral o uso dessa estratégia seja válido pois, além de não adicionar nenhum custo à iteração, não influenciou negativamente no desempenho do método, em nenhum dos testes apresentados, e pode melhorar os resultados obtidos para determinados tipos de problema.

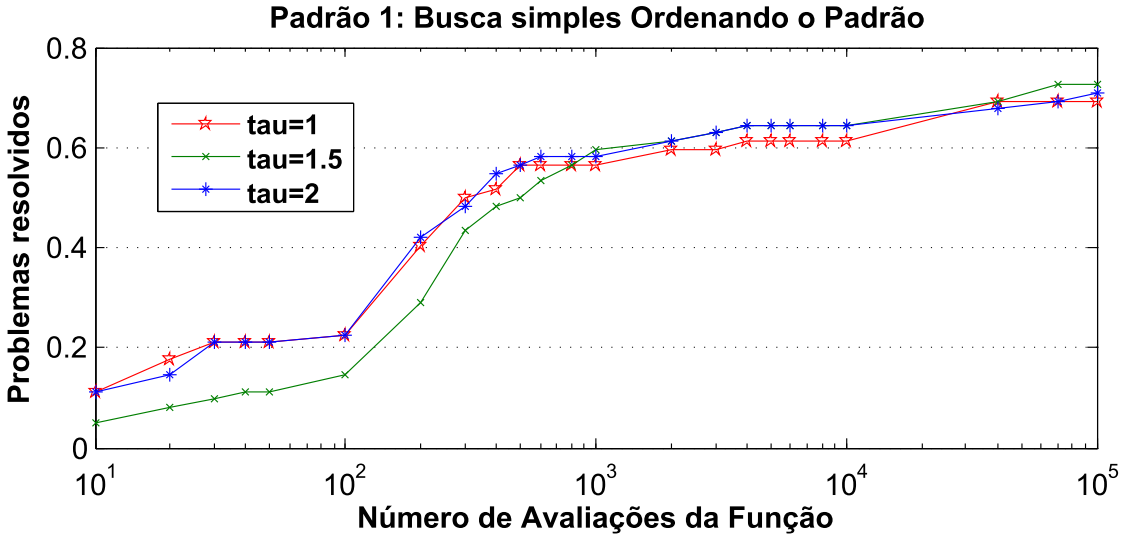


Figura 7.31: Atualização do tamanho do passo: Padrão 1 - Busca Simples Ordenando o Padrão

Como podemos observar na Figura 7.31, utilizando o Padrão 1, o uso da estratégia, combinado com o parâmetro $\tau = 1.5$, melhorou em aproximadamente 2% a robustez atingida pelo método, mantendo a mesma eficiência, se considerarmos os resultados obtidos pelo mesmo parâmetro com busca simples. Para os outros valores de τ, o uso da

estratégia não alterou o desempenho do método em relação à busca simples, apresentando a mesma robustez e eficiência.

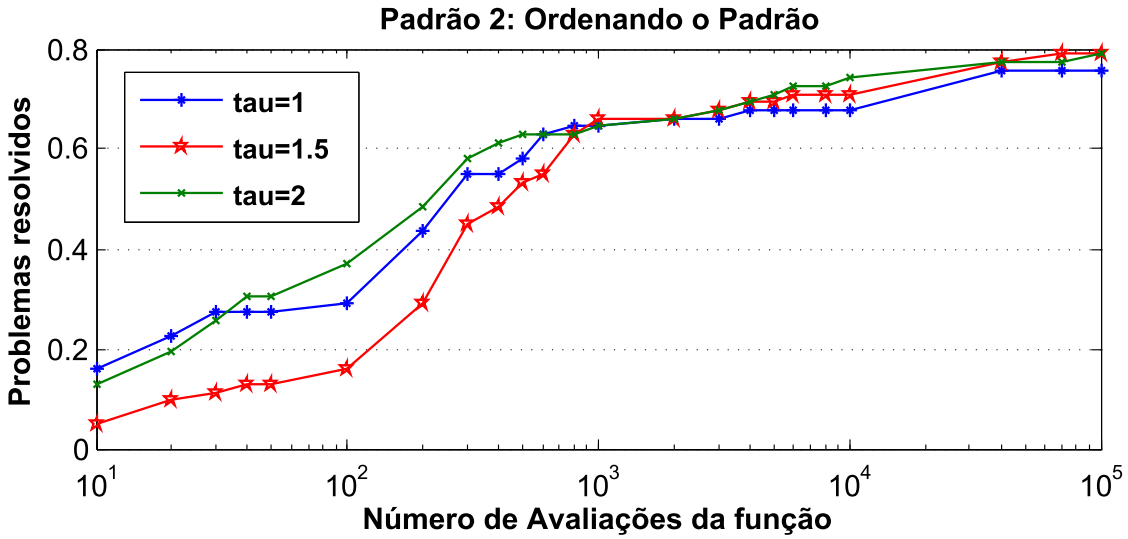


Figura 7.32: Atualização do tamanho do passo: Padrão 2 - Busca Simples Ordenando o Padrão

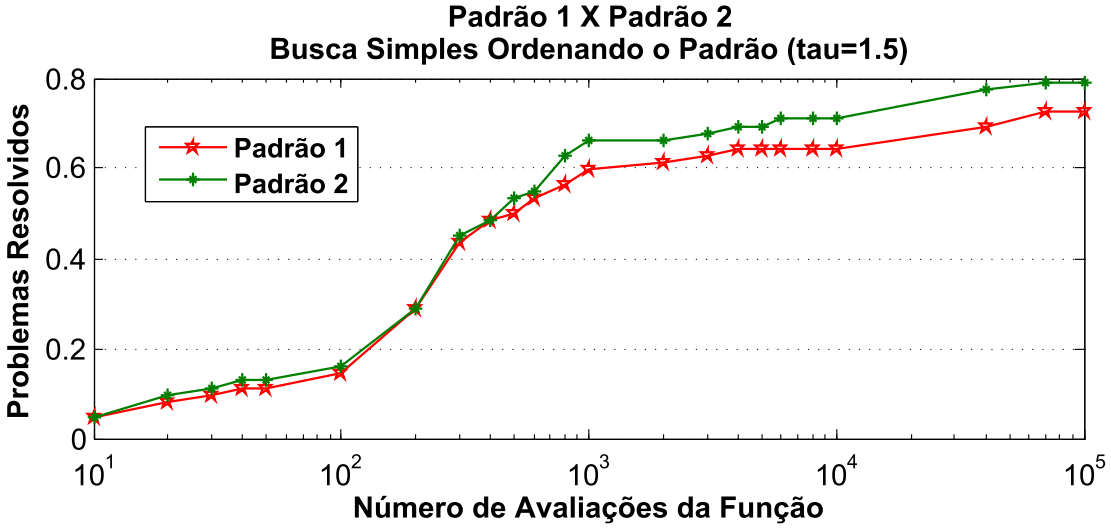


Figura 7.33: Padrão 1 $\times$ Padrão 2 - Busca Simples Ordenando o Padrão ($\tau = 1.5$)

Para o Padrão 2, vemos na Figura 7.32 que o uso da estratégia de ordenar o padrão que utilizamos não interferiu no resultado obtido pelo método com busca simples sem esta estratégia, para $\tau = 1$ e $\tau = 2$. No entanto, assim como para o Padrão 1, aumentamos a robustez e mantemos a eficiência para $\tau = 1.5$. Vale ainda destacar que, com esta estratégia, ao contrário do que verificamos para o Padrão 1, a robustez obtida pelo Padrão 2 utilizando $\tau = 1.5$ foi a mesma obtida para $\tau = 2$. De toda forma continua

claro pela Figura 7.32 que o uso de $\tau = 2$ continua a melhor opção, uma vez que se mostra o mais eficiente a partir de 40 avaliações da função.

Comparando o desempenho do Padrão 1 com o Padrão 2, na Figura 7.33, verificamos que para $\tau = 1.5$ o desempenho de ambos os padrões é aproximadamente o mesmo até 400 avaliações da função. A partir daí vemos que o Padrão 2 fica superior, sendo bem mais robusto. Porém, se considerarmos os demais valores de τ, o gráfico de desempenho é equivalente ao observado na Figura 7.27, onde utilizamos apenas busca simples, e portanto o Padrão 2 é superior ao Padrão 1 para todo número máximo de avaliações da função considerados.

7.4.4 Comparação entre as estratégias de busca

No geral é possível ver, pela figura 7.34, que o Padrão 1 teve um desempenho muito semelhante para todas as estratégias de busca. Logo, se optarmos por utilizar o Padrão 1, qualquer das estratégias de busca é válida.

Já com respeito ao parâmetro de atualização do tamanho do passo, no conjunto de todos os testes, o método se mostrou menos robusto para $\tau = 1$. Isso provavelmente ocorre devido ao desempenho ruim do método, com $\tau = 1$, para problemas não suaves. Para $\tau = 1.5$, o método se mostrou mais robusto, resolvendo 72.6% dos problemas contra 71% se utilizarmos $\tau = 2$, exceto com busca simples sem ordenar o padrão, com o que resolvemos 71% dos problemas para ambos os valores de τ. Porém, com $\tau = 1.5$, o método é menos eficiente resolvendo aproximadamente 5% do problemas contra 11% utilizando $\tau = 2$.

Para o Padrão 2, segundo a Figura 7.35, percebemos que o método fica mais robusto se utilizarmos busca completa, porém é mais eficiente com busca simples. Logo, se queremos resolver o maior número de problemas considerando até 800 avaliações da função, devemos utilizar busca simples, mas se estamos interessados na robustez, e dispomos de mais de 800 avaliações da função, devemos utilizar busca completa. Em todos os testes o Padrão 2 obteve melhores resultados para $\tau = 2$, tanto em robustez quanto em eficiência, se considerarmos os resultados acima de 40 avaliações da função.

Sendo assim, levando em consideração o desempenho superior do Padrão 2 em todos os testes, podemos afirmar que a melhor estratégia de busca é utilizar o padrão 2 com o parâmetro de atualização do passo $\tau = 2$, com busca simples ordenando o Padrão ou busca completa, dependendo se priorizamos a eficiência ou a robustez, conforme comentamos acima. Vale ressaltar que, embora o desempenho do método no geral seja equivalente ao realizar a busca simples, com ou sem a estratégia de ordenar o padrão, o uso dessa estratégia pode melhorar o desempenho em caso de problemas não suaves, e tendo em vista que não adicionamos nenhum custo, em termos de operações, ao ordenar o padrão, esta estratégia vale a pena quando realizamos busca simples.

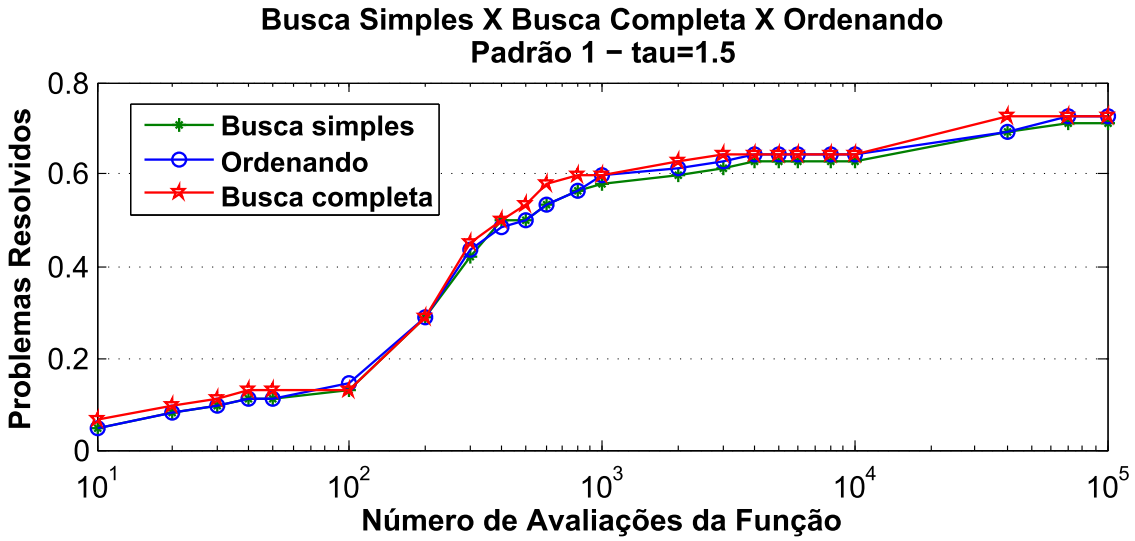


Figura 7.34: Padrão 1 - Estratégias de Busca ($\tau = 1.5$)

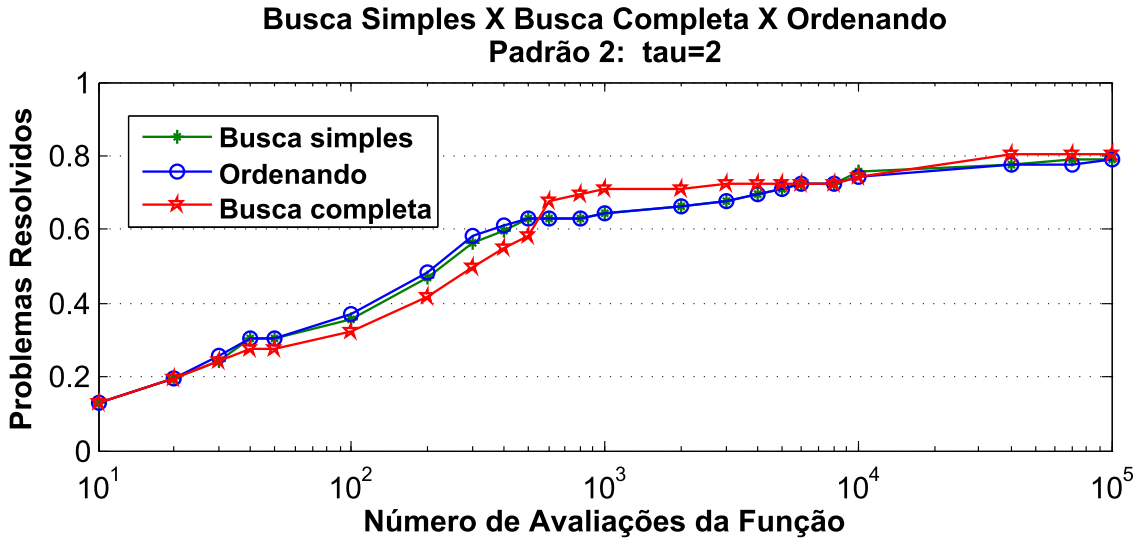


Figura 7.35: Padrão 2 - Estratégias de Busca ($\tau = 2$)

7.5 Problemas Degenerados

Conforme discutimos na seção 5.4.2, na presença de restrições degeneradas, ambos os padrões propostos precisam de certas modificações para que o método possa ser aplicado ao problema.

Usando o Padrão 1, proposto por Lewis, Shepherd e Torczon em [12], embora as direções possam ser calculadas, o método não apresenta bons resultados. Os autores propõem o uso do pacote cddlib, de Fukuda [6], que é uma rotina externa acionada pelo

método para calcular as direções de busca, na presença de restrições degeneradas.

Já o Padrão 2, devido á necessidade da resolução de um sistema linear para calcular as direções de busca, não pode ser calculado na presença de restrições degeneradas. Sendo assim, propomos o uso da solução de Quadrados Mínimos, como alternativa a solução exata do sistema linear.

Apenas três dos problemas do conjunto de testes apresentaram restrições degeneradas. Porém, no conjunto de testes apresentado em [12], a maioria dos problemas considerados pelos autores como sendo degenerados, não apresentam degenerescência das restrições seguindo a abordagem que fizemos neste trabalho.

Em nossa abordagem, descrita na seção 6.2, apenas problemas com restrições de desigualdade poderiam apresentar conjuntos de trabalho degenerados durante a execução do algoritmo. No entanto, em [12], dos 5 problemas apresentados como sendo fortemente degenerados, apenas 1 deles possui restrições de desigualdade, os demais possuem apenas restrições de igualdade e restrições de caixas.

Uma vez que os autores não comentam como as restrições de caixa são abordadas quando aparecem juntamente com outras restrições, sejam elas de igualdade ou desigualdade, acreditamos que eles adicionam essas restrições à matriz de restrições do problema, como desigualdades, e por esse motivo é necessário trabalhar com conjuntos degenerados durante as iterações.

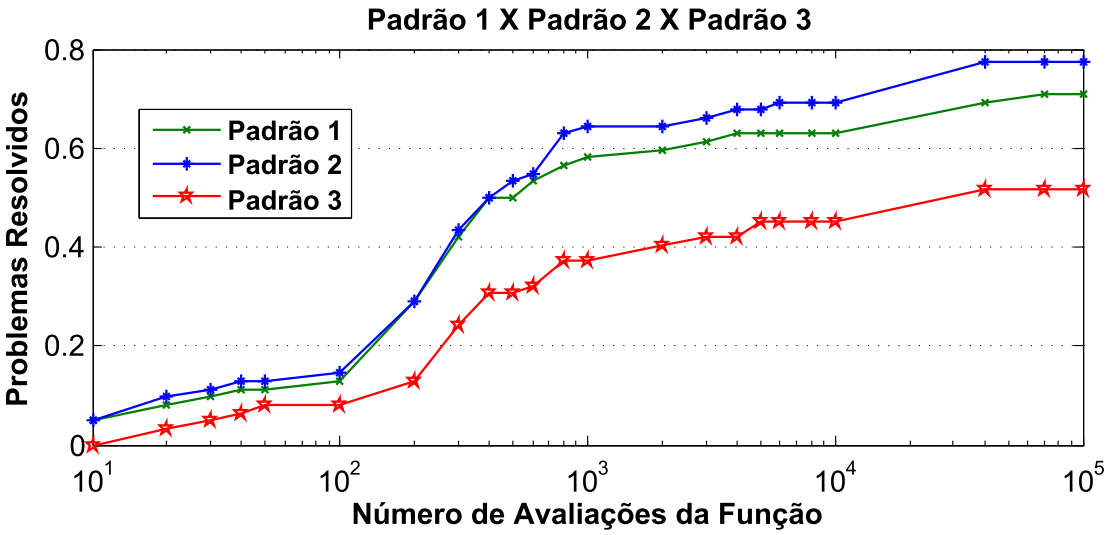


Figura 7.36: Adicionando as restrições de caixas à matriz A

De fato, quando adicionamos as restrições de caixa às restrições do problema, muitos dos problemas testes que trabalhamos apresentaram degenerescência das restrições. Além disso, seguindo essa abordagem, o desempenho do método foi muito inferior, como pode ser observado na Figura 7.36, onde chamamos de Padrão 3, a versão do algoritmo com essa abordagem e utilizando o Padrão 1, o qual supomos ser o algoritmo utilizado pelos autores em [12].

Observe que ao adicionarmos as restrições de caixa às restrições do problema o desempenho do método cai drasticamente. Por esse motivo julgamos que a estratégia descrita na seção 6.2, a qual utilizamos em todos os testes apresentados, é sem sombra de dúvidas a melhor opção, uma vez, que além de realmente preservar a viabilidade dos pontos visitados pelo algoritmo, é mais eficiente e evita degenerescência.

Como temos poucos problemas degenerados no conjunto de testes, não podemos afirmar se a alternativa que propomos para construir o padrão no caso degenerado é ou não adequada. Porém, apesar de não podermos tirar uma conclusão, podemos analisar o comportamento dos dois padrões para esses 3 problemas.

Problemas Degenerados				
Problema	# variáveis	# Restrições de desigualdade	Lim. Inf.	Lim. Sup.
#118 [7]	15	29	15	15
Pentagon [15]	6	15	0	0
Hatfldh [20]	4	7	4	4

Tabela 7.4: Problemas Degenerados

Vemos, pela Tabela 7.4, que os problemas têm a característica comum de possuir mais restrições do que variáveis, que é realmente uma característica de conjuntos de trabalho degenerados.

Pelos resultados apresentados na Tabela 7.5, vemos que o Padrão 2 resolve todos os problemas propostos com a precisão que estipulamos, utilizando busca simples. Já o Padrão 1 atinge essa precisão apenas para um dos problemas. Além disso vemos que, para o primeiro problema com até 10^5 avaliações da função, o resultado obtido pelo Padrão 1 está distante da solução. Já com o Padrão 2, com 4596 avaliações de função, obtemos um valor de função objetivo melhor do que o ótimo reportado na referência.

Utilizando busca completa, pela Tabela 7.6, observamos que não atingimos a precisão de 10^{-7} para o problema Pentagon, para nenhum dos padrões. Porém, vale destacar que este é um problema não suave e o fato de não obtermos uma boa precisão não indica que o padrão que propomos, para problemas degenerados, não é adequado, uma vez que o problema não possui garantias de convergência independentemente de ser degenerado ou não.

Busca Simples - $\tau = 2$				
Problema	$f(\bar{x}) - f(x^\star)$		Número de avaliações da Função	
	Padrão 1	Padrão 2	Padrão 1	Padrão 2
#118 [7]	2.122493e+000	-2.054082e-008	100003	4596
Pentagon [15]	4.068901e-004	2.022496e-009	2996	2096
Hatfldh [20]	0.000000e+000	0.000000e+000	128	131

Tabela 7.5: Problemas Degenerados - Busca Simples

Busca Completa - $\tau = 2$				
Problema	$f(\bar{x}) - f(x^\star)$		Número de avaliações da Função	
	Padrão 1	Padrão 2	Padrão 1	Padrão 2
#118 [7]	1.447373e+000	-4.827814e-008	42155	15149
Pentagon [15]	3.643071e-003	3.456558e-002	1539	1717
Hatfldh [20]	0.000000e+000	0.000000e+000	132	137

Tabela 7.6: Problemas Degenerados - Busca Completa

Pelos resultados apresentados fica claro que de fato o Padrão 1, da forma como é definido, não é adequado para se trabalhar com problemas degenerados, uma vez que de 3 problemas resolve apenas 1. Ainda por cima este é um problema pequeno, que embora tenha restrições degeneradas, tem função objetivo suave e bem comportada. Provavelmente devido ao comportamento do padrão para problemas degenerados, os autores em [12] utilizam o pacote cddlib, o qual não utilizamos neste trabalho.

Já o desempenho do Padrão 2, com a alternativa que propomos, foi satisfatório. Isso indica que o uso dessa estratégia pode ser viável na presença de restrições degeneradas. Porém, vale ressaltar mais uma vez, que o conjunto de testes é muito reduzido, o que impossibilita afirmar que de fato a estratégia funciona na prática.

Capítulo 8

Considerações Finais

O Método de Busca Padrão possui um algoritmo bem simples e com grande apelo geométrico, podendo ser empregado quase que imediatamente em grande parte das aplicações, pois exige muito pouco do problema em si, basta que sejamos capazes de comparar o valor da função objetivo nos pontos visitados pelo algoritmo.

A grande vantagem do método é que, mesmo sem exigir informações do gradiente da função objetivo, o algoritmo possui teoria de convergência global, para problemas suaves.

Apesar de não possuir garantias teóricas de convergência para problemas não suaves, ficou claro, pelos experimentos numéricos apresentados, que embora não possamos esperar que todos os problemas não suaves sejam resolvidos, o método de busca padrão pode ser aplicado diretamente, sem o uso de qualquer modelo, obtendo bons resultados.

Com base em todos os testes apresentados, podemos afirmar que o Método de Busca Padrão, de um modo geral, mostrou-se bastante robusto, uma vez que conseguimos resolver grande parte dos problemas testes propostos com uma precisão considerada alta, inclusive para os tradicionais métodos que fazem uso de derivadas.

Quanto às propostas de melhorias que apresentamos neste trabalho, ficou claro que o resultado obtido foi satisfatório, melhorando o desempenho do algoritmo em todos os conjuntos de testes. O Padrão 2 que propomos foi muito superior ao Padrão 1 proposto em [12] em praticamente todas as abordagens:

- Para os problemas suaves, utilizando o Padrão 2, juntamente com a busca completa e a estratégia de atualização do tamanho do passo que propomos com $\tau = 2$, resolvemos 100% dos problemas propostos, com uma excelente precisão nos resultados obtidos.

- Considerando o mesmo conjunto de testes, apenas com problemas suaves, com o Padrão 1, ainda que utilizando as estratégias de busca que propomos, resolvemos no máximo 88% dos problemas.

- Com a mesma precisão na solução obtida, resolvemos 60% dos problemas minimax propostos. Ou seja, ainda que sem teoria de convergência, boa parte dos proble-

mas não suaves podem ser resolvidos utilizando o Método de Busca Padrão, sem a necessidade do uso de qualquer modelo para a função objetivo.

- O uso de modelos, para os problemas minimax, não apresentou bons resultados na prática.

- As estratégias que utilizamos diminui as chances de trabalharmos com conjuntos de trabalho degenerados e, quando obtemos restrições degeneradas, o padrão que propomos com busca simples resolveu todos os problemas. Entretanto, devido ao número reduzido de problemas, não podemos afirmar que de fato o padrão de direções proposto funciona na presença de restrições degeneradas.

O bom desempenho do Padrão 2 em relação ao Padrão 1 se deve principalmente ao fato de que o conjunto de direções de busca tomadas pelo Padrão 2 é mais amplo. O grande diferencial é a busca na face, que proporcionou uma grande melhora nos resultados obtidos, aumentando significativamente a robustez do método, além de fornecer resultados mais precisos.

Vale destacar também que a busca na face só é possível associada à estratégia proposta de testar a viabiliddade antes de calcular o valor da função objetivo. Isto porque, com essas direções adicionais no padrão, não temos garantias de permanecer dentro do conjunto viável. Mais do que isto, esta estratégia proporcionou diversas melhorias no método, pois nos permitiu, além de testar novas direções, dar passos mais ambiciosos, quando possível.

Outro aspecto positivo observado no uso desta estratégia é o fato de podermos trabalhar com as restrições de caixa separadamente, o que diminuiu drasticamente a presença de restrições degeneradas no conjunto de trabalho. Isto também contribuiu para o bom desempenho do método com as alterações que propomos.

Além disso, é importante frisar que melhoramos o desempenho do método mantendo todas as características que garantem sua convergência. O Padrão 2 também tem a vantagem de ser mais barato computacionalmente, uma vez que substituímos a Decomposição em Valores Singulares pela Decomposição de Cholesky ou pela Fatoração QR, no caso degenerado.

Para finalizar, vale acrescentar que a escolha dos parâmetros e estratégias de busca apresentados, e mesmo do padrão de direções, depende muito do problema que estamos trabalhando. Ficou claro que uma configuração que funciona muito bem para um tipo de problema pode ser ruim para outro. Isso acontece principalmente quando estamos trabalhando com problemas não suaves, quando não podemos prever o comportamento da função nos pontos visitados pelo algoritmo. Sendo assim, é sempre importante conhecer o problema que estamos resolvendo e, na dúvida, mais de uma estratégia deve ser testada, de modo a escolher a que apresentar os melhores resultados.

Apêndice A

Tabelas com os Resultados dos Testes

Nas tabelas a seguir organizamos todos os resultados obtidos. As tabelas, contendo a precisão obtida, o número de iterações e avaliações da função, foram agrupadas da mesma forma que apresentamos nos gráficos do capítulo 7.

A.1 Problemas Suaves

Padrão 1 : Problemas Suaves - Busca simples					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	0.000000e+000	151	7.450581e-009	737
2	2	8.881784e-012	11826	7.450581e-009	20034
3	2	0.000000e+000	30	7.450581e-009	91
4	2	2.225558e-009	63	7.450581e-009	220
5	4	1.827902e+000	44	7.450581e-009	117
6	3	5.114904e-017	82	7.450581e-009	274
7	3	-6.106227e-016	70	7.450581e-009	485
8	3	0.000000e+000	44	7.450581e-009	157
9	3	4.547474e-013	124	7.450581e-009	762
10	4	1.249001e-011	10239	7.450581e-009	47739
11	4	7.939364e-004	51	7.450581e-009	124
12	4	0.000000e+000	35	7.450581e-009	101
13	5	0.000000e+000	33	7.450581e-009	146
14	5	1.377533e-016	103	7.450581e-009	501
15	5	2.752816e-008	18831	1.907349e-006	100003
16	5	5.662451e-017	125	7.450581e-009	441
17	5	8.694490e-017	73	7.450581e-009	243

18	3	-7.318173e-006	199	7.450581e-009	438
19	4	2.500592e-002	225	7.450581e-009	1113
20	10	2.553719e-009	194	7.450581e-009	2323
21	5	2.942972e-005	144	7.450581e-009	762
22	8	-2.074192e+000	9611	7.450581e-009	78353
23	15	1.299771e+000	1112	7.450581e-009	14806
24	2	-7.836311e-008	11241	7.450581e-009	18764
25	2	-8.331882e-011	5812	7.450581e-009	21816
26	2	0.000000e+000	30	7.450581e-009	92
27	2	0.000000e+000	31	7.450581e-009	66
28	2	0.000000e+000	32	7.450581e-009	61
29	2	0.000000e+000	36	7.450581e-009	109
30	3	5.114904e-017	82	7.450581e-009	274
31	3	-7.668809e-009	85	7.450581e-009	595
32	3	-2.056488e-011	54	7.450581e-009	300
33	3	-1.310582e-008	124	7.450581e-009	763
34	4	1.917796e-003	62	7.450581e-009	152
35	4	0.000000e+000	35	7.450581e-009	102
36	5	9.841593e-017	97	7.450581e-009	480
37	5	1.695866e-016	70	7.450581e-009	247
38	5	1.142965e-016	87	7.450581e-009	292
39	5	-8.881784e-016	66	7.450581e-009	224
40	3	-2.302841e-009	199	7.450581e-009	438
41	4	-7.519374e-008	40	7.450581e-009	320
42	4	0.000000e+000	31	7.450581e-009	132
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	-1.054691e-006	169	7.759711e-009	1029
2	2	2.450366e-011	10426	9.640566e-009	28045
3	2	0.000000e+000	77	5.587935e-009	220
4	2	6.860515e-010	92	8.381903e-009	276
5	4	1.827902e+000	49	7.450581e-009	130
6	3	5.294179e-017	92	6.713094e-009	324
7	3	-3.885781e-016	78	6.286427e-009	544
8	3	3.220004e+001	84	7.072231e-009	375
9	3	-5.788329e-007	121	6.372195e-009	737
10	4	1.972445e-011	9621	8.878279e-009	58431
11	4	6.521606e-003	75	9.429641e-009	169
12	4	-1.949957e-009	78	5.304173e-009	239
13	5	0.000000e+000	37	7.450581e-009	160

14	5	2.606570e-016	151	7.168719e-009	746
15	5	2.222768e-008	18636	5.205807e-006	100000
16	5	1.631564e-016	107	5.664173e-009	374
17	5	3.121161e-016	97	7.072231e-009	326
18	3	-7.318195e-006	190	8.729675e-009	541
19	4	2.500591e-002	412	9.708951e-009	2222
20	10	2.553733e-009	473	6.214778e-009	7217
21	5	-8.368180e-008	269	8.514020e-009	1405
22	8	-2.074192e+000	1553	5.986045e-009	18048
23	15	1.754469e+000	232	7.759711e-009	3575
24	2	-7.835155e-008	10350	8.804816e-009	27766
25	2	-6.683790e-010	6254	5.653098e-009	23136
26	2	1.387779e-022	85	5.587935e-009	229
27	2	-3.725290e-009	59	5.587935e-009	135
28	2	5.551115e-017	37	5.587935e-009	71
29	2	-1.303846e-010	59	6.286427e-009	178
30	3	5.294179e-017	92	6.713094e-009	324
31	3	-7.668809e-009	94	6.286427e-009	665
32	3	-2.056844e-011	151	7.168719e-009	747
33	3	7.557778e-007	136	8.064809e-009	846
34	4	1.376448e-003	67	6.286427e-009	155
35	4	-1.061708e-007	162	8.381903e-009	602
36	5	1.333488e-016	128	5.967195e-009	628
37	5	1.616090e-016	80	6.286427e-009	281
38	5	2.833311e-017	92	6.286427e-009	323
39	5	0.000000e+000	104	7.956260e-009	355
40	3	-2.324668e-009	190	8.729675e-009	541
41	4	-8.444912e-008	161	7.072231e-009	988
42	4	0.000000e+000	31	7.450581e-009	131
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	4.547474e-013	99	7.450581e-009	637
2	2	0.000000e+000	31	7.450581e-009	118
3	2	0.000000e+000	31	7.450581e-009	94
4	2	2.225558e-009	65	7.450581e-009	204
5	4	1.827902e+000	49	7.450581e-009	130
6	3	5.114904e-017	93	7.450581e-009	324
7	3	2.775558e-016	75	7.450581e-009	519
8	3	1.320000e+002	42	7.450581e-009	205
9	3	0.000000e+000	52	7.450581e-009	328

10	4	3.233094e-011	16392	1.192093e-007	100000
11	4	7.939364e-004	53	7.450581e-009	123
12	4	0.000000e+000	35	7.450581e-009	96
13	5	0.000000e+000	37	7.450581e-009	160
14	5	1.377533e-016	145	7.450581e-009	725
15	5	3.307578e-008	18729	3.814697e-006	100003
16	5	5.662479e-017	110	7.450581e-009	390
17	5	8.694489e-017	93	7.450581e-009	310
18	3	-7.318220e-006	189	7.450581e-009	546
19	4	2.500591e-002	357	7.450581e-009	1891
20	10	2.553719e-009	329	7.450581e-009	4888
21	5	1.343535e-007	354	7.450581e-009	1857
22	8	-2.074192e+000	2117	7.450581e-009	25510
23	15	2.122493e+000	4977	1.907349e-006	100003
24	2	-7.837199e-008	3323	7.450581e-009	9174
25	2	-8.331882e-011	5639	7.450581e-009	21172
26	2	0.000000e+000	31	7.450581e-009	95
27	2	0.000000e+000	31	7.450581e-009	65
28	2	1.110223e-016	37	7.450581e-009	72
29	2	0.000000e+000	37	7.450581e-009	112
30	3	5.114904e-017	93	7.450581e-009	324
31	3	-7.668809e-009	93	7.450581e-009	659
32	3	-2.056488e-011	44	7.450581e-009	242
33	3	-1.310627e-008	52	7.450581e-009	330
34	4	1.917796e-003	67	7.450581e-009	153
35	4	0.000000e+000	39	7.450581e-009	114
36	5	9.841594e-017	121	7.450581e-009	609
37	5	1.695866e-016	75	7.450581e-009	267
38	5	1.142965e-016	103	7.450581e-009	356
39	5	-8.881784e-016	77	7.450581e-009	265
40	3	-2.350134e-009	189	7.450581e-009	546
41	4	-7.416073e-008	45	7.450581e-009	312
42	4	0.000000e+000	31	7.450581e-009	128

Tabela A.1: Padrão 1 - Busca Simples

Padrão 2 : Problemas Suaves - Busca simples					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	0.000000e+000	145	7.450581e-009	815
2	2	8.881784e-012	11826	7.450581e-009	20034
3	2	0.000000e+000	30	7.450581e-009	91
4	2	-1.028654e-008	56	7.450581e-009	177
5	4	-8.881784e-016	33	7.450581e-009	36
6	3	2.097088e-016	186	7.450581e-009	512
7	3	-3.885781e-016	63	7.450581e-009	618
8	3	0.000000e+000	44	7.450581e-009	185
9	3	0.000000e+000	124	7.450581e-009	911
10	4	1.249001e-011	10239	7.450581e-009	47739
11	4	2.220446e-016	75	7.450581e-009	313
12	4	0.000000e+000	35	7.450581e-009	195
13	5	0.000000e+000	33	7.450581e-009	146
14	5	8.881784e-016	228	7.450581e-009	1011
15	5	1.189016e-007	37409	1.907349e-006	100002
16	5	1.109336e-031	90	7.450581e-009	253
17	5	2.465190e-032	34	7.450581e-009	127
18	3	-7.318245e-006	76	7.450581e-009	239
19	4	-8.885240e-008	133	7.450581e-009	1021
20	10	2.553719e-009	194	7.450581e-009	2323
21	5	-3.995657e-008	592	7.450581e-009	2117
22	8	-1.999408e+000	11768	9.765625e-004	100008
23	15	3.217862e+000	249	7.450581e-009	2315
24	2	-7.836311e-008	11241	7.450581e-009	18764
25	2	-8.331882e-011	5812	7.450581e-009	21816
26	2	0.000000e+000	30	7.450581e-009	92
27	2	0.000000e+000	31	7.450581e-009	66
28	2	0.000000e+000	32	7.450581e-009	65
29	2	0.000000e+000	36	7.450581e-009	109
30	3	2.097088e-016	186	7.450581e-009	512
31	3	-7.668808e-009	69	7.450581e-009	641
32	3	-2.056488e-011	54	7.450581e-009	300
33	3	-1.310627e-008	124	7.450581e-009	910
34	4	0.000000e+000	59	7.450581e-009	227
35	4	0.000000e+000	35	7.450581e-009	190
36	5	8.881784e-016	213	7.450581e-009	960

37	5	1.430611e-016	108	7.450581e-009	350
38	5	1.774937e-030	32	7.450581e-009	122
39	5	0.000000e+000	68	7.450581e-009	243
40	3	-2.375600e-009	76	7.450581e-009	239
41	4	-5.324604e-008	36	7.450581e-009	454
42	4	0.000000e+000	31	7.450581e-009	133
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.235658e-007	126	6.372195e-009	806
2	2	2.450366e-011	10426	9.640566e-009	28045
3	2	0.000000e+000	77	5.587935e-009	220
4	2	-5.294385e-009	89	9.429641e-009	277
5	4	-8.881784e-016	37	7.450581e-009	40
6	3	3.797850e-017	197	6.547256e-009	642
7	3	-3.885781e-016	63	7.450581e-009	618
8	3	9.999882e+001	108	8.950792e-009	578
9	3	-1.768158e-007	129	7.168719e-009	823
10	4	1.972445e-011	9621	8.878279e-009	58431
11	4	2.220446e-016	75	7.450581e-009	306
12	4	-5.508009e-008	153	5.967195e-009	702
13	5	0.000000e+000	37	7.450581e-009	160
14	5	2.616418e-016	161	9.558292e-009	781
15	5	1.756698e-007	23289	4.881204e-006	100004
16	5	2.194360e-016	109	7.552231e-009	369
17	5	2.775558e-017	87	5.587935e-009	288
18	3	-7.318245e-006	86	5.967195e-009	270
19	4	-8.713650e-008	161	9.688698e-009	1113
20	10	2.553733e-009	473	6.214778e-009	7217
21	5	-7.160949e-009	146	7.956260e-009	793
22	8	-2.074192e+000	2003	7.319985e-009	23511
23	15	-5.737843e-008	386	9.341653e-009	5335
24	2	-7.835155e-008	10350	8.804816e-009	27766
25	2	-6.683790e-010	6254	5.653098e-009	23136
26	2	1.387779e-022	85	5.587935e-009	229
27	2	-3.725290e-009	59	5.587935e-009	135
28	2	0.000000e+000	33	5.587935e-009	67
29	2	-1.303846e-010	59	6.286427e-009	178
30	3	3.797850e-017	197	6.547256e-009	642
31	3	-7.668807e-009	98	5.967195e-009	924
32	3	-2.056844e-011	151	7.168719e-009	747

33	3	-1.899220e-007	131	7.168719e-009	845
34	4	-4.963820e-010	104	7.956260e-009	392
35	4	-1.294954e-007	177	5.034821e-009	817
36	5	4.048078e-018	135	5.034821e-009	689
37	5	1.430611e-016	135	7.450581e-009	458
38	5	4.009139e-017	85	5.587935e-009	310
39	5	0.000000e+000	61	5.587935e-009	221
40	3	-2.375600e-009	86	5.967195e-009	270
41	4	-7.378992e-008	162	6.286427e-009	1176
42	4	0.000000e+000	31	7.450581e-009	131
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	0.000000e+000	103	7.450581e-009	839
2	2	0.000000e+000	31	7.450581e-009	118
3	2	0.000000e+000	31	7.450581e-009	94
4	2	-1.028654e-008	61	7.450581e-009	192
5	4	-8.881784e-016	37	7.450581e-009	40
6	3	2.097088e-016	195	7.450581e-009	620
7	3	-3.885781e-016	63	7.450581e-009	614
8	3	1.000000e+002	40	7.450581e-009	252
9	3	0.000000e+000	44	7.450581e-009	385
10	4	3.233094e-011	16392	1.192093e-007	100000
11	4	2.220446e-016	75	7.450581e-009	306
12	4	0.000000e+000	35	7.450581e-009	184
13	5	0.000000e+000	37	7.450581e-009	160
14	5	7.149052e-031	61	7.450581e-009	326
15	5	2.789620e-007	22201	3.814697e-006	100001
16	5	6.039716e-031	52	7.450581e-009	187
17	5	7.395571e-032	39	7.450581e-009	145
18	3	-7.318245e-006	85	7.450581e-009	269
19	4	-8.843087e-008	159	7.450581e-009	1397
20	10	2.553719e-009	329	7.450581e-009	4888
21	5	-7.304921e-008	191	7.450581e-009	997
22	8	-2.074192e+000	2147	7.450581e-009	26110
23	15	-2.054082e-008	348	7.450581e-009	4596
24	2	-7.837199e-008	3323	7.450581e-009	9174
25	2	-8.331882e-011	5639	7.450581e-009	21172
26	2	0.000000e+000	31	7.450581e-009	95
27	2	0.000000e+000	31	7.450581e-009	65
28	2	0.000000e+000	35	7.450581e-009	70

29	2	0.000000e+000	37	7.450581e-009	112
30	3	2.097088e-016	195	7.450581e-009	620
31	3	-7.668808e-009	83	7.450581e-009	809
32	3	-2.056488e-011	44	7.450581e-009	242
33	3	-1.310627e-008	44	7.450581e-009	391
34	4	0.000000e+000	67	7.450581e-009	248
35	4	0.000000e+000	39	7.450581e-009	205
36	5	3.947460e-016	165	7.450581e-009	825
37	5	1.430611e-016	135	7.450581e-009	458
38	5	1.774937e-030	33	7.450581e-009	128
39	5	0.000000e+000	71	7.450581e-009	255
40	3	-2.375600e-009	85	7.450581e-009	269
41	4	-6.465160e-008	43	7.450581e-009	478
42	4	0.000000e+000	31	7.450581e-009	131

Tabela A.2: Padrão 2 - Busca Simples

Padrão 1 : Problemas Suaves - Busca Completa					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	0.000000e+000	50	7.450581e-009	364
2	2	8.881784e-012	7815	7.450581e-009	31261
3	2	0.000000e+000	30	7.450581e-009	91
4	2	1.008689e-008	57	7.450581e-009	253
5	4	1.827902e+000	44	7.450581e-009	129
6	3	5.114904e-017	75	7.450581e-009	301
7	3	-6.106227e-016	64	7.450581e-009	495
8	3	1.320000e+002	38	7.450581e-009	221
9	3	0.000000e+000	38	7.450581e-009	289
10	4	7.876855e+000	12500	1.907349e-006	100001
11	4	7.939364e-004	49	7.450581e-009	150
12	4	0.000000e+000	32	7.450581e-009	111
13	5	0.000000e+000	29	7.450581e-009	151
14	5	1.377533e-016	81	7.450581e-009	487
15	5	9.760870e-009	16667	4.768372e-007	100003
16	5	5.662453e-017	109	7.450581e-009	437
17	5	8.694490e-017	65	7.450581e-009	261
18	3	-7.318173e-006	156	7.450581e-009	609
19	4	2.104872e-002	88	7.450581e-009	588
20	10	2.553719e-009	59	7.450581e-009	1151

21	5	9.393858e-009	158	7.450581e-009	1170
22	8	-2.074192e+000	2948	7.450581e-009	43776
23	15	1.409637e+000	2787	7.450581e-009	79700
24	2	-7.836311e-008	7423	7.450581e-009	29693
25	2	-8.331882e-011	4420	7.450581e-009	17650
26	2	0.000000e+000	30	7.450581e-009	92
27	2	0.000000e+000	30	7.450581e-009	64
28	2	0.000000e+000	32	7.450581e-009	65
29	2	0.000000e+000	36	7.450581e-009	117
30	3	5.114904e-017	75	7.450581e-009	301
31	3	-7.668809e-009	78	7.450581e-009	621
32	3	-2.056488e-011	38	7.450581e-009	229
33	3	-1.310627e-008	38	7.450581e-009	289
34	4	1.917796e-003	57	7.450581e-009	193
35	4	0.000000e+000	31	7.450581e-009	107
36	5	9.841593e-017	80	7.450581e-009	481
37	5	1.695866e-016	69	7.450581e-009	277
38	5	1.142965e-016	75	7.450581e-009	301
39	5	-8.881784e-016	63	7.450581e-009	253
40	3	-2.302841e-009	156	7.450581e-009	609
41	4	-7.519374e-008	38	7.450581e-009	339
42	4	0.000000e+000	31	7.450581e-009	136
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	-1.172026e-006	90	8.950792e-009	688
2	2	1.347010e-011	7254	8.738984e-009	29017
3	2	0.000000e+000	77	5.587935e-009	232
4	2	1.008689e-008	57	7.450581e-009	238
5	4	1.827902e+000	49	7.450581e-009	142
6	3	1.929837e-017	80	6.286427e-009	321
7	3	-6.106227e-016	65	7.450581e-009	500
8	3	1.130937e+002	61	6.286427e-009	369
9	3	-1.139939e-006	77	6.286427e-009	584
10	4	2.951148e-012	4118	7.890426e-009	32945
11	4	7.939364e-004	49	7.450581e-009	148
12	4	3.725290e-008	59	5.587935e-009	213
13	5	0.000000e+000	29	7.450581e-009	150
14	5	1.112260e-016	101	7.072231e-009	607
15	5	9.258332e-009	16667	1.527869e-006	100003
16	5	4.037882e-018	89	7.956260e-009	357

17	5	5.490179e-017	75	5.587935e-009	301
18	3	-7.318253e-006	149	9.688698e-009	581
19	4	2.500315e-002	91	5.587935e-009	658
20	10	2.553719e-009	59	7.450581e-009	1151
21	5	3.882275e-008	191	8.286371e-009	1296
22	8	-2.074192e+000	926	7.259100e-009	15499
23	15	2.522457e+000	125	7.552231e-009	3033
24	2	-7.836547e-008	7192	6.293132e-009	28769
25	2	-1.730067e-009	4516	5.189709e-009	18013
26	2	1.387779e-022	85	5.587935e-009	257
27	2	0.000000e+000	31	7.450581e-009	66
28	2	5.551115e-017	37	5.587935e-009	75
29	2	-1.303846e-010	59	6.286427e-009	194
30	3	1.929837e-017	80	6.286427e-009	321
31	3	-7.668809e-009	79	7.450581e-009	629
32	3	-2.056488e-011	71	6.286427e-009	426
33	3	-1.153045e-006	77	6.286427e-009	584
34	4	3.567322e-004	64	8.381903e-009	214
35	4	7.450581e-009	85	5.587935e-009	375
36	5	5.489751e-017	86	6.286427e-009	517
37	5	1.695866e-016	73	7.450581e-009	293
38	5	1.142965e-016	77	7.450581e-009	309
39	5	0.000000e+000	74	8.381903e-009	297
40	3	-2.382876e-009	149	9.688698e-009	581
41	4	-6.823141e-008	40	8.381903e-009	323
42	4	0.000000e+000	31	7.450581e-009	135
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	4.547474e-013	99	7.450581e-009	637
2	2	0.000000e+000	31	7.450581e-009	118
3	2	0.000000e+000	31	7.450581e-009	94
4	2	2.225558e-009	65	7.450581e-009	204
5	4	1.827902e+000	49	7.450581e-009	130
6	3	5.114904e-017	93	7.450581e-009	324
7	3	2.775558e-016	75	7.450581e-009	519
8	3	1.320000e+002	42	7.450581e-009	205
9	3	0.000000e+000	52	7.450581e-009	328
10	4	3.233094e-011	16392	1.192093e-007	100000
11	4	7.939364e-004	53	7.450581e-009	123
12	4	0.000000e+000	35	7.450581e-009	96

13	5	0.000000e+000	37	7.450581e-009	160
14	5	1.377533e-016	145	7.450581e-009	725
15	5	3.307578e-008	18729	3.814697e-006	100003
16	5	5.662479e-017	110	7.450581e-009	390
17	5	8.694489e-017	93	7.450581e-009	310
18	3	-7.318220e-006	189	7.450581e-009	546
19	4	2.500591e-002	357	7.450581e-009	1891
20	10	2.553719e-009	329	7.450581e-009	4888
21	5	1.343535e-007	354	7.450581e-009	1857
22	8	-2.074192e+000	2117	7.450581e-009	25510
23	15	2.122493e+000	4977	1.907349e-006	100003
24	2	-7.837199e-008	3323	7.450581e-009	9174
25	2	-8.331882e-011	5639	7.450581e-009	21172
26	2	0.000000e+000	31	7.450581e-009	95
27	2	0.000000e+000	31	7.450581e-009	65
28	2	1.110223e-016	37	7.450581e-009	72
29	2	0.000000e+000	37	7.450581e-009	112
30	3	5.114904e-017	93	7.450581e-009	324
31	3	-7.668809e-009	93	7.450581e-009	659
32	3	-2.056488e-011	44	7.450581e-009	242
33	3	-1.310627e-008	52	7.450581e-009	330
34	4	1.917796e-003	67	7.450581e-009	153
35	4	0.000000e+000	39	7.450581e-009	114
36	5	9.841594e-017	121	7.450581e-009	609
37	5	1.695866e-016	75	7.450581e-009	267
38	5	1.142965e-016	103	7.450581e-009	356
39	5	-8.881784e-016	77	7.450581e-009	265
40	3	-2.350134e-009	189	7.450581e-009	546
41	4	-7.416073e-008	45	7.450581e-009	312
42	4	0.000000e+000	31	7.450581e-009	128

Tabela A.3: Padrão 1 - Busca Completa

Padrão 2 : Problemas Suaves - Busca Completa					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	9.094947e-013	75	7.450581e-009	801
2	2	8.881784e-012	7815	7.450581e-009	31261
3	2	0.000000e+000	30	7.450581e-009	91
4	2	-1.028654e-008	52	7.450581e-009	201
5	4	-8.881784e-016	31	7.450581e-009	39
6	3	2.097088e-016	130	7.450581e-009	521
7	3	-3.885781e-016	63	7.450581e-009	731
8	3	1.000000e+002	37	7.450581e-009	272
9	3	0.000000e+000	35	7.450581e-009	381
10	4	7.876855e+000	12500	1.907349e-006	100001
11	4	2.220446e-016	75	7.450581e-009	371
12	4	0.000000e+000	32	7.450581e-009	200
13	5	0.000000e+000	29	7.450581e-009	151
14	5	8.881784e-016	138	7.450581e-009	829
15	5	2.572554e-007	16667	1.907349e-006	100003
16	5	1.109336e-031	64	7.450581e-009	257
17	5	7.395571e-032	32	7.450581e-009	129
18	3	-7.318245e-006	74	7.450581e-009	276
19	4	-8.953954e-008	69	7.450581e-009	908
20	10	2.553719e-009	59	7.450581e-009	1151
21	5	2.017347e-008	3873	7.450581e-009	27271
22	8	-2.074192e+000	2949	7.450581e-009	43791
23	15	-1.270598e-008	42	7.450581e-009	923
24	2	-7.836311e-008	7423	7.450581e-009	29693
25	2	-8.331882e-011	4420	7.450581e-009	17650
26	2	0.000000e+000	30	7.450581e-009	92
27	2	0.000000e+000	30	7.450581e-009	64
28	2	0.000000e+000	32	7.450581e-009	65
29	2	0.000000e+000	36	7.450581e-009	117
30	3	2.097088e-016	130	7.450581e-009	521
31	3	-7.668808e-009	55	7.450581e-009	653
32	3	-2.056488e-011	38	7.450581e-009	229
33	3	-1.310627e-008	35	7.450581e-009	381
34	4	0.000000e+000	52	7.450581e-009	251
35	4	0.000000e+000	31	7.450581e-009	196
36	5	8.881784e-016	134	7.450581e-009	805

37	5	1.430611e-016	81	7.450581e-009	325
38	5	1.774937e-030	29	7.450581e-009	117
39	5	0.000000e+000	65	7.450581e-009	261
40	3	-2.375600e-009	74	7.450581e-009	276
41	4	-5.324604e-008	34	7.450581e-009	467
42	4	0.000000e+000	31	7.450581e-009	139
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	-1.153198e-006	97	6.713094e-009	753
2	2	1.347010e-011	7254	8.738984e-009	29017
3	2	0.000000e+000	77	5.587935e-009	232
4	2	-4.083544e-009	75	5.587935e-009	299
5	4	-8.881784e-016	33	7.450581e-009	41
6	3	5.011423e-017	139	5.587935e-009	557
7	3	-3.885781e-016	63	7.450581e-009	731
8	3	8.531898e+001	71	6.286427e-009	456
9	3	4.442409e-007	94	7.072231e-009	730
10	4	2.951148e-012	4118	7.890426e-009	32945
11	4	2.220446e-016	75	7.450581e-009	364
12	4	-7.897616e-008	81	5.587935e-009	592
13	5	0.000000e+000	29	7.450581e-009	150
14	5	8.018277e-017	100	8.381903e-009	601
15	5	1.894916e-007	16667	6.538767e-006	100003
16	5	2.257283e-016	79	7.956260e-009	317
17	5	4.625929e-017	85	5.587935e-009	341
18	3	-7.318245e-006	80	5.967195e-009	297
19	4	-9.526582e-008	111	7.072231e-009	1027
20	10	2.553719e-009	59	7.450581e-009	1151
21	5	-1.322996e-007	98	9.429641e-009	785
22	8	-2.074192e+000	716	7.931551e-009	12300
23	15	-1.025065e-007	159	5.449893e-009	3964
24	2	-7.836547e-008	7192	6.293132e-009	28769
25	2	-1.730067e-009	4516	5.189709e-009	18013
26	2	1.387779e-022	85	5.587935e-009	257
27	2	0.000000e+000	31	7.450581e-009	66
28	2	0.000000e+000	33	5.587935e-009	67
29	2	-1.303846e-010	59	6.286427e-009	194
30	3	5.011423e-017	139	5.587935e-009	557
31	3	-7.668808e-009	55	7.450581e-009	653
32	3	-2.056488e-011	71	6.286427e-009	426

33	3	4.311346e-007	94	7.072231e-009	730
34	4	0.000000e+000	53	7.450581e-009	249
35	4	-1.221895e-007	95	5.587935e-009	623
36	5	9.560253e-017	113	5.587935e-009	679
37	5	1.430611e-016	81	7.450581e-009	325
38	5	1.774937e-030	29	7.450581e-009	117
39	5	0.000000e+000	65	7.450581e-009	261
40	3	-2.375600e-009	80	5.967195e-009	297
41	4	-7.582747e-008	39	5.587935e-009	508
42	4	0.000000e+000	31	7.450581e-009	137
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	9.094947e-013	74	7.450581e-009	767
2	2	0.000000e+000	31	7.450581e-009	124
3	2	0.000000e+000	31	7.450581e-009	94
4	2	-1.028654e-008	53	7.450581e-009	197
5	4	-8.881784e-016	33	7.450581e-009	41
6	3	2.097088e-016	229	7.450581e-009	917
7	3	-3.885781e-016	63	7.450581e-009	727
8	3	0.000000e+000	37	7.450581e-009	203
9	3	0.000000e+000	83	7.450581e-009	943
10	4	1.249001e-011	5561	7.450581e-009	44489
11	4	2.220446e-016	75	7.450581e-009	364
12	4	0.000000e+000	33	7.450581e-009	201
13	5	0.000000e+000	29	7.450581e-009	150
14	5	8.881784e-016	191	7.450581e-009	1147
15	5	4.812774e-007	16667	1.525879e-005	100003
16	5	3.081488e-031	46	7.450581e-009	185
17	5	7.395571e-032	37	7.450581e-009	149
18	3	-7.318245e-006	81	7.450581e-009	295
19	4	-8.816525e-008	81	7.450581e-009	1052
20	10	2.553719e-009	59	7.450581e-009	1151
21	5	-4.447571e-009	81	7.450581e-009	627
22	8	-2.074192e+000	957	7.450581e-009	16250
23	15	-4.827814e-008	680	7.450581e-009	15149
24	2	-7.836311e-008	8275	7.450581e-009	33101
25	2	-8.331882e-011	4485	7.450581e-009	17903
26	2	0.000000e+000	31	7.450581e-009	95
27	2	0.000000e+000	31	7.450581e-009	66
28	2	0.000000e+000	35	7.450581e-009	71

29	2	0.000000e+000	37	7.450581e-009	117
30	3	2.097088e-016	229	7.450581e-009	917
31	3	-7.668808e-009	55	7.450581e-009	653
32	3	-2.056488e-011	42	7.450581e-009	249
33	3	-1.310627e-008	83	7.450581e-009	943
34	4	0.000000e+000	53	7.450581e-009	249
35	4	0.000000e+000	33	7.450581e-009	196
36	5	8.881784e-016	181	7.450581e-009	1087
37	5	1.430611e-016	81	7.450581e-009	325
38	5	1.774937e-030	29	7.450581e-009	117
39	5	0.000000e+000	65	7.450581e-009	261
40	3	-2.375600e-009	81	7.450581e-009	295
41	4	-6.465160e-008	39	7.450581e-009	503
42	4	0.000000e+000	31	7.450581e-009	137

Tabela A.4: Padrão 2 - Busca Completa

Padrão 1 : Problemas Suaves - Busca Simples Ordenando o Padrão					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	0.000000e+000	151	7.450581e-009	737
2	2	8.881784e-012	11856	7.450581e-009	24684
3	2	0.000000e+000	30	7.450581e-009	91
4	2	2.225558e-009	63	7.450581e-009	220
5	4	1.827902e+000	44	7.450581e-009	113
6	3	5.114904e-017	88	7.450581e-009	291
7	3	-6.106227e-016	70	7.450581e-009	485
8	3	0.000000e+000	44	7.450581e-009	157
9	3	4.547474e-013	124	7.450581e-009	762
10	4	1.249001e-011	15249	7.450581e-009	44577
11	4	7.009928e-004	51	7.450581e-009	129
12	4	0.000000e+000	35	7.450581e-009	101
13	5	0.000000e+000	33	7.450581e-009	148
14	5	1.377533e-016	102	7.450581e-009	477
15	5	1.210640e-008	29095	9.536743e-007	100000
16	5	5.662453e-017	120	7.450581e-009	346
17	5	8.694490e-017	73	7.450581e-009	244
18	3	-7.318173e-006	199	7.450581e-009	489
19	4	2.500592e-002	225	7.450581e-009	1113
20	10	2.553719e-009	194	7.450581e-009	2195

21	5	2.942972e-005	144	7.450581e-009	762
22	8	-2.074192e+000	9611	7.450581e-009	78353
23	15	1.299771e+000	1112	7.450581e-009	14806
24	2	-7.836311e-008	11241	7.450581e-009	23732
25	2	-8.331882e-011	5812	7.450581e-009	14029
26	2	0.000000e+000	30	7.450581e-009	91
27	2	0.000000e+000	31	7.450581e-009	62
28	2	0.000000e+000	32	7.450581e-009	63
29	2	0.000000e+000	36	7.450581e-009	109
30	3	5.114904e-017	88	7.450581e-009	291
31	3	-7.668809e-009	85	7.450581e-009	595
32	3	-2.056488e-011	54	7.450581e-009	300
33	3	-1.310582e-008	124	7.450581e-009	763
34	4	1.584968e-003	61	7.450581e-009	154
35	4	0.000000e+000	35	7.450581e-009	102
36	5	9.841594e-017	95	7.450581e-009	437
37	5	1.695866e-016	70	7.450581e-009	239
38	5	1.142965e-016	85	7.450581e-009	276
39	5	-8.881784e-016	64	7.450581e-009	220
40	3	-2.302841e-009	199	7.450581e-009	489
41	4	-7.519374e-008	40	7.450581e-009	320
42	4	0.000000e+000	31	7.450581e-009	132
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	-1.054691e-006	169	7.759711e-009	1029
2	2	2.546553e-011	10996	9.610627e-009	32900
3	2	0.000000e+000	77	5.587935e-009	220
4	2	6.860515e-010	92	8.381903e-009	276
5	4	1.827902e+000	49	7.450581e-009	126
6	3	5.079537e-017	97	5.034821e-009	324
7	3	-3.885781e-016	78	6.286427e-009	544
8	3	3.220004e+001	84	7.072231e-009	375
9	3	-5.788329e-007	121	6.372195e-009	737
10	4	1.362202e-011	14587	8.570141e-009	83557
11	4	6.521574e-003	75	9.429641e-009	180
12	4	-1.949957e-009	78	5.304173e-009	239
13	5	0.000000e+000	37	7.450581e-009	160
14	5	1.721424e-016	133	5.664173e-009	623
15	5	1.853816e-008	20921	1.400069e-006	100000
16	5	6.979409e-017	101	7.552231e-009	343

17	5	8.794392e-017	97	5.304173e-009	335
18	3	-7.318202e-006	206	7.365663e-009	612
19	4	2.500591e-002	412	9.708951e-009	2222
20	10	2.553712e-009	485	6.547256e-009	7225
21	5	-8.368180e-008	269	8.514020e-009	1405
22	8	-2.074192e+000	1553	5.986045e-009	18048
23	15	1.754469e+000	232	7.759711e-009	3575
24	2	-7.834877e-008	10855	9.481272e-009	32535
25	2	-3.528897e-010	6280	9.055170e-009	20450
26	2	1.387779e-022	85	5.587935e-009	236
27	2	-3.725290e-009	59	5.587935e-009	107
28	2	5.551115e-017	37	5.587935e-009	73
29	2	-1.303846e-010	59	6.286427e-009	178
30	3	5.079537e-017	97	5.034821e-009	324
31	3	-7.668809e-009	94	6.286427e-009	665
32	3	-2.056844e-011	151	7.168719e-009	747
33	3	7.557778e-007	136	8.064809e-009	846
34	4	8.044339e-003	81	5.967195e-009	204
35	4	-1.061708e-007	162	8.381903e-009	602
36	5	1.746613e-016	133	6.713094e-009	640
37	5	2.008161e-016	76	8.381903e-009	262
38	5	2.833311e-017	92	6.286427e-009	314
39	5	8.881784e-016	92	5.967195e-009	316
40	3	-2.331944e-009	206	7.365663e-009	612
41	4	-8.444912e-008	161	7.072231e-009	988
42	4	0.000000e+000	31	7.450581e-009	131
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	4.547474e-013	99	7.450581e-009	637
2	2	0.000000e+000	31	7.450581e-009	121
3	2	0.000000e+000	31	7.450581e-009	94
4	2	2.225558e-009	65	7.450581e-009	204
5	4	1.827902e+000	49	7.450581e-009	126
6	3	5.114904e-017	99	7.450581e-009	329
7	3	2.775558e-016	75	7.450581e-009	519
8	3	1.320000e+002	42	7.450581e-009	205
9	3	0.000000e+000	52	7.450581e-009	328
10	4	9.255293e-010	16736	1.192093e-007	100004
11	4	7.009928e-004	53	7.450581e-009	130
12	4	0.000000e+000	35	7.450581e-009	96

13	5	0.000000e+000	37	7.450581e-009	160
14	5	1.377533e-016	133	7.450581e-009	639
15	5	2.222890e-008	20521	3.814697e-006	100004
16	5	5.662477e-017	114	7.450581e-009	380
17	5	8.694489e-017	89	7.450581e-009	304
18	3	-7.318216e-006	217	7.450581e-009	677
19	4	2.500591e-002	357	7.450581e-009	1891
20	10	2.553719e-009	329	7.450581e-009	4888
21	5	1.343535e-007	354	7.450581e-009	1857
22	8	-2.074192e+000	2117	7.450581e-009	25510
23	15	2.122493e+000	4977	1.907349e-006	100003
24	2	-7.836311e-008	12071	7.450581e-009	37454
25	2	-8.331882e-011	5775	7.450581e-009	18606
26	2	0.000000e+000	31	7.450581e-009	95
27	2	0.000000e+000	31	7.450581e-009	63
28	2	1.110223e-016	37	7.450581e-009	74
29	2	0.000000e+000	37	7.450581e-009	112
30	3	5.114904e-017	99	7.450581e-009	329
31	3	-7.668809e-009	93	7.450581e-009	659
32	3	-2.056488e-011	44	7.450581e-009	242
33	3	-1.310627e-008	52	7.450581e-009	330
34	4	1.584968e-003	63	7.450581e-009	154
35	4	0.000000e+000	39	7.450581e-009	114
36	5	9.841594e-017	121	7.450581e-009	580
37	5	1.695866e-016	75	7.450581e-009	255
38	5	1.142965e-016	101	7.450581e-009	357
39	5	-8.881784e-016	77	7.450581e-009	280
40	3	-2.346496e-009	217	7.450581e-009	677
41	4	-7.416073e-008	45	7.450581e-009	312
42	4	0.000000e+000	31	7.450581e-009	128

Tabela A.5: Padrão 1 - Busca Simples Ordenando o Padrão

Padrão 2 : Problemas Suaves - Busca Simples Ordenando o Padrão					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	0.000000e+000	145	7.450581e-009	815
2	2	8.881784e-012	11856	7.450581e-009	24684
3	2	0.000000e+000	30	7.450581e-009	91
4	2	-1.028654e-008	56	7.450581e-009	177
5	4	-8.881784e-016	33	7.450581e-009	34
6	3	2.097088e-016	177	7.450581e-009	533
7	3	-3.885781e-016	63	7.450581e-009	618
8	3	0.000000e+000	44	7.450581e-009	185
9	3	0.000000e+000	124	7.450581e-009	911
10	4	1.249001e-011	15249	7.450581e-009	44577
11	4	2.220446e-016	75	7.450581e-009	337
12	4	0.000000e+000	35	7.450581e-009	195
13	5	0.000000e+000	33	7.450581e-009	148
14	5	8.881784e-016	228	7.450581e-009	1011
15	5	1.192259e-007	37474	1.907349e-006	100001
16	5	1.109336e-031	86	7.450581e-009	245
17	5	7.395571e-032	34	7.450581e-009	127
18	3	-7.318245e-006	76	7.450581e-009	239
19	4	-8.885240e-008	133	7.450581e-009	1021
20	10	2.553719e-009	194	7.450581e-009	2195
21	5	-3.995657e-008	592	7.450581e-009	2117
22	8	-1.999408e+000	11768	9.765625e-004	100008
23	15	3.217862e+000	249	7.450581e-009	2315
24	2	-7.836311e-008	11241	7.450581e-009	23732
25	2	-8.331882e-011	5812	7.450581e-009	14029
26	2	0.000000e+000	30	7.450581e-009	91
27	2	0.000000e+000	31	7.450581e-009	62
28	2	0.000000e+000	32	7.450581e-009	63
29	2	0.000000e+000	36	7.450581e-009	109
30	3	2.097088e-016	177	7.450581e-009	533
31	3	-7.668808e-009	69	7.450581e-009	641
32	3	-2.056488e-011	54	7.450581e-009	300
33	3	-1.310627e-008	124	7.450581e-009	910
34	4	0.000000e+000	59	7.450581e-009	238
35	4	0.000000e+000	35	7.450581e-009	190
36	5	8.881784e-016	215	7.450581e-009	964

37	5	1.430611e-016	108	7.450581e-009	311
38	5	1.774937e-030	30	7.450581e-009	115
39	5	0.000000e+000	67	7.450581e-009	233
40	3	-2.375600e-009	76	7.450581e-009	239
41	4	-5.324604e-008	36	7.450581e-009	454
42	4	0.000000e+000	31	7.450581e-009	133

$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.235658e-007	126	6.372195e-009	806
2	2	2.546553e-011	10996	9.610627e-009	32900
3	2	0.000000e+000	77	5.587935e-009	220
4	2	-5.294385e-009	89	9.429641e-009	277
5	4	-8.881784e-016	37	7.450581e-009	40
6	3	2.202552e-016	198	7.759711e-009	649
7	3	-3.885781e-016	63	7.450581e-009	618
8	3	9.999882e+001	108	8.950792e-009	578
9	3	-1.768158e-007	129	7.168719e-009	823
10	4	1.362202e-011	14587	8.570141e-009	83557
11	4	2.220446e-016	75	7.450581e-009	330
12	4	-5.508009e-008	153	5.967195e-009	702
13	5	0.000000e+000	37	7.450581e-009	160
14	5	2.095413e-016	171	9.558292e-009	817
15	5	2.863911e-007	23800	6.261785e-006	100005
16	5	1.040848e-016	106	5.034821e-009	360
17	5	2.775558e-017	87	5.587935e-009	309
18	3	-7.318245e-006	86	5.967195e-009	275
19	4	-8.713650e-008	161	9.688698e-009	1113
20	10	2.553712e-009	485	6.547256e-009	7225
21	5	-7.160949e-009	146	7.956260e-009	793
22	8	-2.074192e+000	2003	7.319985e-009	23511
23	15	-5.737843e-008	386	9.341653e-009	5335
24	2	-7.834877e-008	10855	9.481272e-009	32535
25	2	-3.528897e-010	6280	9.055170e-009	20450
26	2	1.387779e-022	85	5.587935e-009	236
27	2	-3.725290e-009	59	5.587935e-009	107
28	2	0.000000e+000	33	5.587935e-009	65
29	2	-1.303846e-010	59	6.286427e-009	178
30	3	2.202552e-016	198	7.759711e-009	649
31	3	-7.668807e-009	98	5.967195e-009	924
32	3	-2.056844e-011	151	7.168719e-009	747

33	3	-1.899220e-007	131	7.168719e-009	845
34	4	-7.780223e-010	99	8.950792e-009	387
35	4	-1.294954e-007	177	5.034821e-009	817
36	5	2.497315e-016	107	5.304173e-009	533
37	5	1.430611e-016	135	7.450581e-009	458
38	5	4.009139e-017	85	5.587935e-009	289
39	5	0.000000e+000	70	6.286427e-009	239
40	3	-2.375600e-009	86	5.967195e-009	275
41	4	-7.378992e-008	162	6.286427e-009	1176
42	4	0.000000e+000	31	7.450581e-009	131

$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	0.000000e+000	103	7.450581e-009	839
2	2	0.000000e+000	31	7.450581e-009	121
3	2	0.000000e+000	31	7.450581e-009	94
4	2	-1.028654e-008	61	7.450581e-009	192
5	4	-8.881784e-016	37	7.450581e-009	40
6	3	2.097088e-016	189	7.450581e-009	640
7	3	-3.885781e-016	63	7.450581e-009	614
8	3	1.000000e+002	40	7.450581e-009	252
9	3	0.000000e+000	44	7.450581e-009	385
10	4	9.255293e-010	16736	1.192093e-007	100004
11	4	2.220446e-016	75	7.450581e-009	330
12	4	0.000000e+000	35	7.450581e-009	184
13	5	0.000000e+000	37	7.450581e-009	160
14	5	5.546678e-031	51	7.450581e-009	274
15	5	2.785442e-007	22224	7.629395e-006	100001
16	5	1.232595e-030	48	7.450581e-009	177
17	5	7.395571e-032	35	7.450581e-009	136
18	3	-7.318245e-006	81	7.450581e-009	256
19	4	-8.843087e-008	159	7.450581e-009	1397
20	10	2.553719e-009	329	7.450581e-009	4888
21	5	-7.304921e-008	191	7.450581e-009	997
22	8	-2.074192e+000	2147	7.450581e-009	26110
23	15	-2.054082e-008	348	7.450581e-009	4596
24	2	-7.836311e-008	12071	7.450581e-009	37454
25	2	-8.331882e-011	5775	7.450581e-009	18606
26	2	0.000000e+000	31	7.450581e-009	95
27	2	0.000000e+000	31	7.450581e-009	63
28	2	0.000000e+000	35	7.450581e-009	68

29	2	0.000000e+000	37	7.450581e-009	112
30	3	2.097088e-016	189	7.450581e-009	640
31	3	-7.668808e-009	83	7.450581e-009	809
32	3	-2.056488e-011	44	7.450581e-009	242
33	3	-1.310627e-008	44	7.450581e-009	391
34	4	0.000000e+000	67	7.450581e-009	249
35	4	0.000000e+000	39	7.450581e-009	205
36	5	2.465190e-031	45	7.450581e-009	245
37	5	1.430611e-016	135	7.450581e-009	458
38	5	1.774937e-030	33	7.450581e-009	125
39	5	0.000000e+000	71	7.450581e-009	248
40	3	-2.375600e-009	81	7.450581e-009	256
41	4	-6.465160e-008	43	7.450581e-009	478
42	4	0.000000e+000	31	7.450581e-009	131

Tabela A.6: Padrão 2 - Busca Simples Ordenando o Padrão

A.2 Problemas Não Suaves

Padrão 1 : Problemas Suaves - Busca Simples					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	55	7.450581e-009	252
2	2	-2.857143e-009	67	7.450581e-009	297
3	2	-1.138375e-008	76	7.450581e-009	330
4	2	-4.594747e-009	66	7.450581e-009	285
5	7	-6.135003e-002	97	7.450581e-009	771
6	6	3.887032e-002	124	7.450581e-009	1220
7	8	8.059047e-001	142	7.450581e-009	1243
8	10	1.344999e+002	6315	3.051758e-005	100010
9	20	-3.594332e+001	226	7.450581e-009	7682
10	9	1.793377e+002	99999	1.000000e+000	100000
11	20	1.867363e+004	3086	7.450581e-009	74626
12	10	7.881355e+002	11896	1.220703e-004	100000
13	7	4.561450e+002	5691	1.953125e-003	100012
14	8	6.225775e+003	12309	1.250000e-001	100003
15	16	7.494061e+001	2308	7.450581e-009	47482

$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	8.830733e-009	67	6.286427e-009	303
2	2	-2.857143e-009	65	5.587935e-009	292
3	2	1.769100e-009	88	6.713094e-009	398
4	2	-5.724178e-009	69	9.429641e-009	306
5	7	-5.494116e-002	244	5.045345e-009	2710
6	6	1.237618e-005	174	7.266524e-009	1986
7	8	-1.229395e+003	35	5.587935e-009	219
8	10	-4.981793e+000	212	5.449893e-009	4115
9	20	-3.745081e+001	304	8.286371e-009	12664
10	9	1.724300e+002	99999	1.000000e+003	100000
11	20	1.701857e+004	279	8.612176e-009	7345
12	10	3.037882e+002	4375	7.850199e-009	83713
13	7	1.907176e+002	4614	1.329849e-003	100016
14	8	5.808596e+002	1450	9.701700e-009	27675
15	16	8.040924e+001	946	8.166487e-009	29747
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	59	7.450581e-009	267
2	2	-2.857143e-009	69	7.450581e-009	304
3	2	-1.138375e-008	99	7.450581e-009	443
4	2	-4.594746e-009	57	7.450581e-009	266
5	7	-5.986835e-002	163	7.450581e-009	1696
6	6	4.068839e-004	272	7.450581e-009	2996
7	8	6.503330e-002	147	7.450581e-009	1488
8	10	-5.863814e+000	168	7.450581e-009	3395
9	20	-3.515186e+001	406	7.450581e-009	18971
10	9	1.724300e+002	99999	1.000000e+003	100000
11	20	1.308040e+003	190	7.450581e-009	5178
12	10	5.816131e+002	4941	5.000000e-001	100004
13	7	4.049297e+002	4688	1.953125e-003	100024
14	8	1.890786e+003	550	7.275958e-009	9735
15	16	7.862894e+001	1843	7.450581e-009	60657

Tabela A.7: Padrão 1 - Busca Simples

Padrão 2 : Problemas não Suaves - Busca Simples					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	55	7.450581e-009	326
2	2	-2.857143e-009	67	7.450581e-009	387
3	2	4.129627e-009	65	7.450581e-009	339
4	2	-8.077437e-009	76	7.450581e-009	318
5	7	-5.948101e-002	101	7.450581e-009	647
6	6	1.918060e-003	147	7.450581e-009	1525
7	8	-2.666667e+000	33	7.450581e-009	139
8	10	1.338757e+002	106	7.450581e-009	2614
9	20	5.106458e+000	219	7.450581e-009	9035
10	9	1.812836e+002	14286	1.000000e+000	100002
11	20	1.867363e+004	3086	7.450581e-009	74626
12	10	7.786919e+002	13161	1.220703e-004	100020
13	7	3.804374e+002	263	7.450581e-009	1503
14	8	4.923683e+003	13136	7.450581e-009	26067
15	16	8.715440e+001	3751	7.450581e-009	43001
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	8.830733e-009	67	6.286427e-009	397
2	2	-2.857143e-009	65	5.587935e-009	385
3	2	4.129627e-009	65	5.587935e-009	345
4	2	-8.216257e-009	75	7.072231e-009	321
5	7	-6.016716e-002	157	7.655268e-009	1268
6	6	4.029404e-002	181	5.449893e-009	2909
7	8	-2.666667e+000	37	7.450581e-009	139
8	10	-5.381789e+000	192	6.804683e-009	4310
9	20	-4.108810e+001	323	7.865579e-009	15335
10	9	1.743761e+002	14288	1.000000e+003	100005
11	20	1.701857e+004	279	8.612176e-009	7345
12	10	6.853214e+002	1722	5.195017e-009	26248
13	7	3.170448e+002	329	7.481761e-009	4508
14	8	1.077687e+003	1216	5.217966e-009	18141
15	16	7.754168e+001	911	7.259100e-009	21092

$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	59	7.450581e-009	341
2	2	-2.857143e-009	69	7.450581e-009	396
3	2	4.129627e-009	71	7.450581e-009	375
4	2	-4.523671e-009	55	7.450581e-009	325
5	7	-6.295865e-002	154	7.450581e-009	1450
6	6	-4.163308e-009	191	7.450581e-009	2096
7	8	-2.666667e+000	37	7.450581e-009	139
8	10	2.954733e+000	322	7.450581e-009	9186
9	20	-3.298396e+001	448	7.450581e-009	27226
10	9	1.743516e+002	14287	1.000000e+003	100001
11	20	1.308040e+003	190	7.450581e-009	5178
12	10	6.991448e+002	777	7.450581e-009	12180
13	7	3.972076e+002	384	7.450581e-009	4811
14	8	7.117415e+003	79	7.275958e-009	1096
15	16	7.754167e+001	685	7.450581e-009	23422

Tabela A.8: Padrão 2 - Busca Simples

Padrão 1 : Problemas não Suaves - Busca Completa					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	53	7.450581e-009	266
2	2	-2.857143e-009	65	7.450581e-009	326
3	2	2.836844e-009	68	7.450581e-009	405
4	2	-4.594747e-009	62	7.450581e-009	315
5	7	-6.065803e-002	88	7.450581e-009	1097
6	6	2.831459e-005	103	7.450581e-009	1666
7	8	4.425752e-001	76	7.450581e-009	1020
8	10	-4.521307e+000	94	7.450581e-009	2605
9	20	-4.097531e+001	104	7.450581e-009	6701
10	9	1.518470e+000	62	7.450581e-009	1129
11	20	3.493052e+000	130	7.450581e-009	4901
12	10	7.745321e+002	3724	1.220703e-004	100026
13	7	4.667147e+002	99	7.450581e-009	1990
14	8	4.215340e+003	3125	1.000000e+000	100001
15	16	8.568294e+001	834	7.450581e-009	39769

$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	53	7.450581e-009	266
2	2	-2.857143e-009	65	7.450581e-009	326
3	2	-7.323610e-009	77	9.429641e-009	448
4	2	-4.594747e-009	63	7.450581e-009	317
5	7	-6.122029e-002	83	7.072231e-009	1105
6	6	-4.172506e-009	149	6.804683e-009	2150
7	8	-1.229395e+003	35	5.587935e-009	234
8	10	-4.537050e+000	153	5.449893e-009	3986
9	20	-4.214772e+001	159	5.103512e-009	10361
10	9	1.529645e+000	82	7.072231e-009	1470
11	20	4.934763e+002	93	5.034821e-009	2991
12	10	6.648654e+002	480	5.421635e-009	11114
13	7	4.708460e+002	121	6.048607e-009	2375
14	8	1.195265e+003	426	5.142250e-009	11092
15	16	7.707217e+001	233	6.911939e-009	10203
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	53	7.450581e-009	266
2	2	-2.857143e-009	65	7.450581e-009	326
3	2	8.007970e-009	69	7.450581e-009	404
4	2	-4.594747e-009	63	7.450581e-009	317
5	7	-5.878375e-002	77	7.450581e-009	1087
6	6	3.643065e-003	95	7.450581e-009	1539
7	8	4.520293e-001	89	7.450581e-009	1174
8	10	-5.033677e+000	185	7.450581e-009	5021
9	20	-3.759183e+001	310	7.450581e-009	19821
10	9	1.509158e+000	61	7.450581e-009	1104
11	20	5.199305e+001	50	7.450581e-009	1832
12	10	6.896666e+002	388	7.450581e-009	9217
13	7	4.666314e+002	4002	2.441406e-004	100000
14	8	8.786882e+002	1180	7.275958e-009	33460
15	16	8.568294e+001	234	7.450581e-009	10533

Tabela A.9: Padrão 1 - Busca Completa

Padrão 2 : Problemas não Suaves - Busca Completa					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	55	7.450581e-009	326
2	2	-2.857143e-009	67	7.450581e-009	387
3	2	4.129627e-009	65	7.450581e-009	339
4	2	-8.077437e-009	76	7.450581e-009	318
5	7	-5.948101e-002	101	7.450581e-009	647
6	6	1.918060e-003	147	7.450581e-009	1525
7	8	-2.666667e+000	33	7.450581e-009	139
8	10	1.338757e+002	106	7.450581e-009	2614
9	20	5.106458e+000	219	7.450581e-009	9035
10	9	1.812836e+002	14286	1.000000e+000	100002
11	20	1.867363e+004	3086	7.450581e-009	74626
12	10	7.786919e+002	13161	1.220703e-004	100020
13	7	3.804374e+002	263	7.450581e-009	1503
14	8	4.923683e+003	13136	7.450581e-009	26067
15	16	8.715440e+001	3751	7.450581e-009	43001
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	8.830733e-009	67	6.286427e-009	397
2	2	-2.857143e-009	65	5.587935e-009	385
3	2	4.129627e-009	65	5.587935e-009	345
4	2	-8.216257e-009	75	7.072231e-009	321
5	7	-6.016716e-002	157	7.655268e-009	1268
6	6	4.029404e-002	181	5.449893e-009	2909
7	8	-2.666667e+000	37	7.450581e-009	139
8	10	-5.381789e+000	192	6.804683e-009	4310
9	20	-4.108810e+001	323	7.865579e-009	15335
10	9	1.743761e+002	14288	1.000000e+003	100005
11	20	1.701857e+004	279	8.612176e-009	7345
12	10	6.853214e+002	1722	5.195017e-009	26248
13	7	3.170448e+002	329	7.481761e-009	4508
14	8	1.077687e+003	1216	5.217966e-009	18141
15	16	7.754168e+001	911	7.259100e-009	21092

$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	59	7.450581e-009	341
2	2	-2.857143e-009	69	7.450581e-009	396
3	2	4.129627e-009	71	7.450581e-009	375
4	2	-4.523671e-009	55	7.450581e-009	325
5	7	-6.295865e-002	154	7.450581e-009	1450
6	6	-4.163308e-009	191	7.450581e-009	2096
7	8	-2.666667e+000	37	7.450581e-009	139
8	10	2.954733e+000	322	7.450581e-009	9186
9	20	-3.298396e+001	448	7.450581e-009	27226
10	9	1.743516e+002	14287	1.000000e+003	100001
11	20	1.308040e+003	190	7.450581e-009	5178
12	10	6.991448e+002	777	7.450581e-009	12180
13	7	3.972076e+002	384	7.450581e-009	4811
14	8	7.117415e+003	79	7.275958e-009	1096
15	16	7.754167e+001	685	7.450581e-009	23422

Tabela A.10: Padrão 2 - Busca Completa

Padrão 1 : Problemas não Suaves - Busca Simples Ordenando o Padrão					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	55	7.450581e-009	252
2	2	-2.857143e-009	67	7.450581e-009	297
3	2	-1.138375e-008	76	7.450581e-009	330
4	2	-4.594747e-009	66	7.450581e-009	285
5	7	-6.135003e-002	97	7.450581e-009	771
6	6	3.887032e-002	124	7.450581e-009	1220
7	8	1.089194e+000	127	7.450581e-009	1109
8	10	1.344999e+002	6315	3.051758e-005	100010
9	20	-3.594332e+001	226	7.450581e-009	7682
10	9	1.793377e+002	99999	1.000000e+000	100000
11	20	1.867363e+004	3088	7.450581e-009	67513
12	10	7.881355e+002	11896	1.220703e-004	100000
13	7	4.561450e+002	5691	1.953125e-003	100012
14	8	6.225775e+003	12309	1.250000e-001	100003
15	16	7.494061e+001	2308	7.450581e-009	47482

$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	8.830733e-009	67	6.286427e-009	303
2	2	-2.857143e-009	65	5.587935e-009	292
3	2	1.769100e-009	88	6.713094e-009	398
4	2	-5.724178e-009	69	9.429641e-009	306
5	7	-5.494116e-002	244	5.045345e-009	2710
6	6	1.237618e-005	174	7.266524e-009	1986
7	8	-1.229395e+003	35	5.587935e-009	220
8	10	-4.981793e+000	212	5.449893e-009	4115
9	20	-3.745081e+001	304	8.286371e-009	12664
10	9	1.724300e+002	99999	1.000000e+003	100000
11	20	1.701857e+004	320	9.196694e-009	8071
12	10	3.037882e+002	4375	7.850199e-009	83713
13	7	1.907176e+002	4614	1.329849e-003	100016
14	8	5.808596e+002	1450	9.701700e-009	27675
15	16	8.040924e+001	946	8.166487e-009	29747
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	59	7.450581e-009	267
2	2	-2.857143e-009	69	7.450581e-009	304
3	2	-1.138375e-008	99	7.450581e-009	443
4	2	-4.594746e-009	57	7.450581e-009	266
5	7	-5.986835e-002	163	7.450581e-009	1696
6	6	4.068839e-004	272	7.450581e-009	2996
7	8	8.072815e-001	123	7.450581e-009	1227
8	10	-5.863814e+000	168	7.450581e-009	3395
9	20	-3.515186e+001	406	7.450581e-009	18971
10	9	1.724300e+002	99999	1.000000e+003	100000
11	20	1.308040e+003	186	7.450581e-009	4815
12	10	5.816131e+002	4941	5.000000e-001	100004
13	7	4.049297e+002	4688	1.953125e-003	100024
14	8	1.890786e+003	550	7.275958e-009	9735
15	16	7.862894e+001	1843	7.450581e-009	60657

Tabela A.11: Padrão 1 - Busca Simples Ordenando o Padrão

Padrão 2 : Problemas não Suaves - Busca Simples Ordenando o Padrão					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	55	7.450581e-009	326
2	2	-2.857143e-009	67	7.450581e-009	387
3	2	4.129627e-009	65	7.450581e-009	339
4	2	-8.077437e-009	76	7.450581e-009	318
5	7	-5.948101e-002	101	7.450581e-009	647
6	6	1.918066e-003	147	7.450581e-009	1525
7	8	-2.830548e+000	33	7.450581e-009	134
8	10	1.338757e+002	106	7.450581e-009	2614
9	20	5.106458e+000	219	7.450581e-009	9035
10	9	1.812836e+002	14286	1.000000e+000	100002
11	20	1.867363e+004	3088	7.450581e-009	67513
12	10	7.786919e+002	13161	1.220703e-004	100020
13	7	3.804374e+002	263	7.450581e-009	1503
14	8	4.923683e+003	13136	7.450581e-009	26067
15	16	8.715440e+001	3751	7.450581e-009	43001
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	8.830733e-009	67	6.286427e-009	397
2	2	-2.857143e-009	65	5.587935e-009	385
3	2	4.129627e-009	65	5.587935e-009	345
4	2	-8.216257e-009	75	7.072231e-009	321
5	7	-6.016716e-002	157	7.655268e-009	1268
6	6	4.029405e-002	181	5.449893e-009	2909
7	8	-2.666667e+000	37	7.450581e-009	139
8	10	-5.381789e+000	192	6.804683e-009	4310
9	20	-4.108810e+001	323	7.865579e-009	15335
10	9	1.743761e+002	14288	1.000000e+003	100005
11	20	1.701857e+004	320	9.196694e-009	8071
12	10	6.853214e+002	1722	5.195017e-009	26248
13	7	3.170448e+002	329	7.481761e-009	4508
14	8	1.077687e+003	1216	5.217966e-009	18141
15	16	7.754168e+001	911	7.259100e-009	21092

$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	2	1.185875e-008	59	7.450581e-009	341
2	2	-2.857143e-009	69	7.450581e-009	396
3	2	4.129627e-009	71	7.450581e-009	375
4	2	-4.523671e-009	55	7.450581e-009	325
5	7	-6.295865e-002	154	7.450581e-009	1450
6	6	2.022496e-009	191	7.450581e-009	2096
7	8	-2.666667e+000	37	7.450581e-009	139
8	10	2.954733e+000	322	7.450581e-009	9186
9	20	-3.298396e+001	448	7.450581e-009	27226
10	9	1.743516e+002	14287	1.000000e+003	100001
11	20	1.308040e+003	186	7.450581e-009	4815
12	10	6.991448e+002	777	7.450581e-009	12180
13	7	3.972076e+002	384	7.450581e-009	4811
14	8	7.117415e+003	79	7.275958e-009	1096
15	16	7.754167e+001	685	7.450581e-009	23422

Tabela A.12: Padrão 2 - Busca Simples Ordenando o Padrão

A.3 Problemas Adicionais

Padrão 1 : Problemas adicionais - Busca Simples					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	4.144378e-003	25770	4.882813e-004	100001
2	49	4.444400e+009	7507	6.250000e-002	100009
3	31	7.761963e+003	30	7.450581e-009	4
4	46	5.493932e+006	4035	7.450581e-009	8085
5	20	3.645119e-002	33	7.450581e-009	60
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.075431e-006	24578	4.571479e-005	100001
2	49	4.444377e+009	5771	1.822072e-002	100009
3	31	7.755535e+003	31	5.587935e-009	3
4	46	8.318078e+002	8137	4.786320e-001	100015
5	20	3.645119e-002	33	7.450581e-009	58

$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	4.876376e-004	20910	9.765625e-004	100004
2	49	4.444377e+009	5749	1.525879e-002	100009
3	31	7.749131e+003	31	7.450581e-009	3
4	46	2.053838e+003	8958	3.125000e-002	100008
5	20	3.645119e-002	33	7.450581e-009	57

Tabela A.13: Problemas adicionais: Padrão 1 - Busca Simples

Padrão 2 : Problemas adicionais - Busca Simples					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	4.144378e-003	25770	4.882813e-004	100001
2	49	1.627320e+004	5583	2.500000e-001	100012
3	31	3.859376e-001	128	7.450581e-009	101
4	46	-8.746210e-006	3266	7.450581e-009	34306
5	20	-6.620537e-004	27	7.450581e-009	112
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.075431e-006	24578	4.571479e-005	100001
2	49	2.637211e+002	3475	4.377338e-002	100010
3	31	4.881611e-003	41	6.713094e-009	7
4	46	-8.744461e-006	1148	7.719477e-009	23507
5	20	-6.620537e-004	27	7.450581e-009	112
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	4.876376e-004	20910	9.765625e-004	100004
2	49	7.685774e+002	3584	1.220703e-001	100000
3	31	4.208903e-003	40	7.450581e-009	6
4	46	-8.746225e-006	722	7.450581e-009	14375
5	20	-6.620537e-004	27	7.450581e-009	112

Tabela A.14: Problemas adicionais: Padrão 2 - Busca Simples

Padrão 1 : Problemas adicionais - Busca Completa					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	6.174287e-004	17474	2.441406e-004	100000
2	49	4.444407e+009	5041	1.250000e-001	100012
3	31	7.748146e+003	29	7.450581e-009	16
4	46	6.401400e+001	4020	3.906250e-003	100023
5	20	3.645119e-002	33	7.450581e-009	68
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.119631e-003	16691	3.875619e-003	100001
2	49	4.444377e+009	5199	9.911147e-001	100014
3	31	7.748146e+003	29	7.450581e-009	16
4	46	2.698073e+002	4850	1.174427e-002	100006
5	20	3.645119e-002	33	7.450581e-009	66
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	9.382633e-004	16693	7.812500e-003	100003
2	49	4.444377e+009	5284	7.629395e-003	100007
3	31	7.748146e+003	29	7.450581e-009	16
4	46	6.525207e+002	6023	3.906250e-003	100011
5	20	3.645119e-002	33	7.450581e-009	65

Tabela A.15: Problemas adicionais: Padrão 1 - Busca Completa

Padrão 2 : Problemas adicionais - Busca Completa					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	6.174287e-004	17474	2.441406e-004	100000
2	49	8.527086e+003	2990	2.500000e-001	100022
3	31	4.847701e-001	29	7.450581e-009	19
4	46	-8.746003e-006	392	7.450581e-009	11896
5	20	-6.620537e-004	27	7.450581e-009	112

$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.119631e-003	16691	3.875619e-003	100001
2	49	5.843252e+003	2917	2.526775e+000	100000
3	31	4.847701e-001	29	7.450581e-009	19
4	46	-8.740499e-006	407	9.975682e-009	12119
5	20	-6.620537e-004	27	7.450581e-009	112
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	9.382633e-004	16693	7.812500e-003	100003
2	49	3.683860e+003	2870	2.441406e-001	100001
3	31	4.847701e-001	29	7.450581e-009	19
4	46	-8.746181e-006	434	7.450581e-009	12852
5	20	-6.620537e-004	27	7.450581e-009	112

Tabela A.16: Problemas adicionais: Padrão 2 - Busca Completa

Padrão 1 : Problemas adicionais - Busca Simples Odenando o Padrão					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	2.819739e-004	32407	2.441406e-004	100000
2	49	4.444399e+009	7765	6.250000e-002	100006
3	31	7.761963e+003	30	7.450581e-009	4
4	46	5.078133e+006	7523	7.450581e-009	19264
5	20	3.645119e-002	33	7.450581e-009	60
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.832675e-017	502	8.885808e-009	2315
2	49	4.444377e+009	5837	1.155440e-002	100013
3	31	7.755535e+003	31	5.587935e-009	3
4	46	1.694159e+002	8323	3.214557e-002	100016
5	20	3.645119e-002	33	7.450581e-009	58

$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.900971e-005	20574	9.765625e-004	100000
2	49	4.444377e+009	5799	3.814697e-003	100018
3	31	7.749131e+003	31	7.450581e-009	3
4	46	2.627977e+002	8469	1.250000e-001	100005
5	20	3.645119e-002	33	7.450581e-009	57

Tabela A.17: Problemas adicionais: Padrão 1 - Busca Simples ordenando o Padrão

Padrão 2 : Problemas adicionais - Busca Simples Ordenando o Padrão					
$\tau = 1$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	2.819739e-004	32407	2.441406e-004	100000
2	49	1.627320e+004	5583	2.500000e-001	100012
3	31	3.859376e-001	128	7.450581e-009	101
4	46	-8.746185e-006	3601	7.450581e-009	39432
5	20	-6.620537e-004	27	7.450581e-009	112
$\tau = 1.5$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.832675e-017	502	8.885808e-009	2315
2	49	4.164705e+001	2986	3.576211e-003	100038
3	31	4.881611e-003	41	6.713094e-009	7
4	46	-8.745632e-006	1153	9.149009e-009	23766
5	20	-6.620537e-004	27	7.450581e-009	112
$\tau = 2$					
Teste	n. var	$f(\bar{x}) - f(x^\star)$	n. iterações	α^k	n. Avaliações da função
1	3	1.900971e-005	20574	9.765625e-004	100000
2	49	7.649484e+002	3588	4.882813e-001	100024
3	31	4.208903e-003	40	7.450581e-009	6
4	46	-8.746225e-006	714	7.450581e-009	14209
5	20	-6.620537e-004	27	7.450581e-009	112

Tabela A.18: Problemas adicionais: Padrão 2 - Busca Simples Ordenando o Padrão

Bibliografia

[1] M.S. Bazaraa, H.D. Sherali, C.M. Shetty. Nonlinear Programming: Theory and Algorithms, 3 ed., New York: John Wiley, (2006).

[2] M.S. Bazaraa, J.J. Jarvis, H.D. Sherali. Linear Programming and Network Flows, 4 ed. John Wiley & Sons (2010).

[3] L.F.Bueno, A. Friedlander, J.M. Martínez, F.N.C. Sobral. Inexact Restoration Method for derivative-free optimization with smooth constraints. Aceito para publicação no SIAM Journal on Optimization.

[4] A.R. Conn, K. Scheinberg, L.N. Vicente. Introduction to Derivative- free Optimization. MPS-SIAM Series on Optimization, SIAM, Philadelphia, (2009).

[5] M. A.Diniz-Ehrhardt, V.L.R. Lopes, L.G. Pedroso. Métodos sem derivadas para minimização irrestrita. 01. ed. São Carlos, SP: Sociedade Brasileira de Matemática Aplicada e Computacional (SBMAC) v.01. 87 p, (2010).

[6] K. Fukuda. cddlib, *http://www.ifor.math.ethz.ch/˜fukuda/cdd home*,(2005).

[7] W. Hock, K. Schittkowski. "Test examples for nonlinear programming code", Springer, Berlin, (1981).

[8] R. Hooke, T.A. Jeeves. Direct search solution of numerical and statistical problems. J. Assoc. Comput. Mach. 8 pp. 212-229, (1961).

[9] R. M. Lewis, V. Torczon. Pattern search algorithms for bound constrained minimization. SIAM Journal on Optimization 9, pp. 1082-1099, (1999).

[10] R.M. Lewis, V. Torczon, M.W. Trosset. Direct search methods: then and now. Journal of Computational and Applied Mathematics 124, 1-2, pp. 191- 207 (2000).

[11] R. M. Lewis, V. Torczon. Pattern search algorithms for linearly constrained minimization. SIAM Journal on Optimization 10, pp. 917-941, (2000).

[12] R. M. Lewis, A. Shepherd, V. Torczon. Implementing generating set search methods for linearly constrained minimization. SIAM Journal on Scientific Computing 29, 6, pp. 2507-2530, (2007).

[13] G. Liuzzi, S. Lucidi, M. Sciandrone. A derivative-free algorithm for linearly constrained finite minimax problems. SIAM Journal on Optimization 16, pp. 1054-1075, (2006).

[14] D.G. Luenberger, Y. Ye. Linear and Nonlinear Programming, 3 ed., Springer, (2008).

[15] L. Luksan, J. Vlcek. Test Problems for Nonsmooth Unconstrained and Linearly Constrained Optimization. Technical report No. 798, (2000).

[16] I.J. Lustig, R.E. Marsten, D.F. Shanno. On implementing Mehrotra´s predictor-corrector interior point method for linear programming, SIAM Journal on Optimization 2 pp. 435-449, (1992).

[17] S. Mehrotra. On the implementation of a primal-dual interior point method. SIAM Journal on Optimization 2, pp.575-601, (1992).

[18] J.A. Nelder, R. Mead. A simplex method for function minimization. The Computer Journal 7, pp. 308-313, (1965).

[19] J. Nocedal, S.J. Wright. Numerical Optimization. Springer, (1999).

[20] D. Orban, N.I.M. Gould, P.L. Toint. CUTEr: Constrained and Unconstrained Testing Environment, revisited,*http://www.cuter.rl.ac.uk* (2011).

[21] L.G. Pedroso, M.A. Diniz-Ehrhardt (orientadora). Sobre o Desempenho de Métodos de Busca Direta para Minimização Irrestrita. Dissertação de mestrado, IMECC, UNICAMP, (2005).

[22] A.A.Ribeiro, E. W. Karas.Um curso de Otimização, manuscrito, Curitiba, (2011).

[23] W. Spendley, G.R. Hext, F.R. Himsworth. Sequential application of simplex designs in optimization and evolutionary operation. Techno- metrics 4, pp. 441-461, (1962).

[24] V. Torczon. On the convergence of pattern search algorithms. SIAM Journal on Optimization 7, pp. 1-25 (1997).

[25] P. Tseng. Fortified-descent simplicial search method: A general approach. SIAM Journal on Optimization 10, pp. 269-288, (1999).