Notas em Matemática Aplicada e-ISSN 2236-5915

Volume 49, 2012

Editado por

Célia A. Zorzo Barcelos
Universidade Federal de Uberlândia - UFU
Uberlândia, MG, Brasil

Eliana X.L. de Andrade
Universidade Estadual Paulista - UNESP
São José do Rio Preto, SP, Brasil

Maurílio Boaventura
Universidade Estadual Paulista - UNESP
São José do Rio Preto, SP, Brasil

Sociedade Brasileira de Matemática Aplicada e Computacional

2012

A Sociedade Brasileira de Matemática Aplicada e Computacional - SBMAC publica, desde as primeiras edições do evento, monografias dos cursos que são ministrados nos CNMAC.

Para a comemoração dos 25 anos da SBMAC, que ocorreu durante o XXVI CNMAC em 2003, foi criada a série **Notas em Matemática Aplicada** para publicar as monografias dos minicursos ministrados nos CNMAC, o que permaneceu até o XXXIII CNMAC em 2010.

A partir de 2011, a série passa a publicar, também, livros nas áreas de interesse da SBMAC. Os autores que submeterem textos à série Notas em Matemática Aplicada devem estar cientes de que poderão ser convidados a ministrarem minicursos nos eventos patrocinados pela SBMAC, em especial nos CNMAC, sobre assunto a que se refere o texto.

O livro deve ser preparado em **Latex (compatível com o Miktex versão 2.7), as figuras em eps** e deve ter entre **80 e 150 páginas**. O texto deve ser redigido de forma clara, acompanhado de uma excelente revisão bibliográfica e de **exercícios de verificação de aprendizagem** ao final de cada capítulo.

Veja todos os títulos publicados nesta série na página
http://www.sbmac.org.br/notas.php

SBMAC **Sociedade Brasileira de Matemática Aplicada e Computacional**

2012

MÉTODOS SEM DERIVADAS PARA MINIMIZAÇÃO IRRESTRITA

Maria Aparecida Diniz-Ehrhardt
cheti@ime.unicamp.br
Véra Lucia da Rocha Lopes
vlopes@ime.unicamp.br
Lucas Garcia Pedroso
lucasgp@ime.unicamp.br

Departamento de Matemática Aplicada
Instituto de Matemática, Estatística e Computação Científica
Universidade Estadual de Campinas

Sociedade Brasileira de Matemática Aplicada e Computacional

São Carlos - SP, Brasil
2012

Coordenação Editorial: Elbert Einstein Nehrer Macau

Coordenação Editorial da Série: Eliana Xavier Linhares de Andrade

Editora: SBMAC

Capa: Matheus Botossi Trindade

Patrocínio: SBMAC

Catalogação elaborada pela Biblioteca do IBILCE/UNESP
Bibliotecária: Maria Luiza Fernandes Jardim Froner

Diniz-Ehrhardt, Maria Aparecida.
Métodos sem Derivadas para Minimização Irrestrita.
- São Carlos, SP: SBMAC, 2012, 87 p., 20.5 cm
- (Notas em Matemática Aplicada; v. 49)

e-ISBN 978-85-8215-009-2

1. Minimização Irrestrita. 2. Métodos sem Derivadas.
3. Busca Direta. 4. Interpolação Polinomial.
I. Diniz-Ehrhardt, Maria Aparecida. II. Lopes, Véra Lucia da Rocha. III. Pedroso, Lucas Garcia. IV. Título. V. Série

CDD - 51

Esta é uma republicação em formato de e-book do livro original do mesmo título publicado em 2010 nesta mesma série pela SBMAC.

Agradecimentos

Ao Mario, por nos introduzir no mundo da otimização sem derivadas, e pela sua valiosíssima participação em alguns dos trabalhos citados neste texto.

Aos estudantes e professores do Grupo de Otimização do DMA, pelas sugestões recebidas sobre vários tópicos deste trabalho, em muitas oportunidades que tivemos de discutir este assunto especialmente nas tardes de segundas-feiras.

Ao Matheus, pela preciosa ajuda na construção de algumas das figuras aqui exibidas.

À Capes, Fapesp e Pronex-Otimização que nos deram apoio financeiro.

Conteúdo

Prefácio

Os métodos sem derivadas *(Derivative-Free Methods)* para problemas de otimização começaram a receber maior atenção da comunidade científica na década de 1960, e teem sido estudados com maior frequência e interesse nos últimos anos. Renomados pesquisadores, como John Dennis, Michael Powell, Virgínia Torczon e Philippe Toint, têm se dedicado muito a esta área de pesquisa. Sua aplicabilidade em problemas práticos tem aumentado cada vez mais. Citamos, entre eles, problemas de simulações e problemas que têm como função objetivo um arquivo executável cujo código não está acessível ao usuário, conhecidos como caixas-pretas.

O contato do Grupo de Otimização do Departamento de Matemática Aplicada, DMA, do IMECC-UNICAMP com esse assunto, quer via consultas bibliográficas, quer através de contatos pessoais com outros pesquisadores, despertou nosso interesse em trabalhar com métodos sem derivadas para problemas de otimização. Começamos com um trabalho de mestrado em 2003, mas o "vírus"derivative-free logo se espalhou entre nós. Hoje, o número de professores interessados no assunto aumentou, e o trabalho do grupo tem sido bastante intenso. Já formamos alguns mestres e doutores neste tema. Durante essas orientações, sentimos necessidade de oferecer cursos de tópicos para alunos de pós graduação, que tiveram como objeto de estudo bibliografia especializada no assunto. A audiência tem sido bastante grande, o que mostra a boa receptividade de nossa proposta.

Animados e motivados com essa nossa experiência no DMA, interessamonos em ampliar o âmbito dos nossos estudos, e consideramos que a melhor maneira de atingir esse objetivo é levando o assunto ao CNMAC. Assim, nos reunimos e fizemos uma proposta de minicurso em otimização irrestrita, que submetemos ao XXXIII CNMAC. Para nossa satisfação, a proposta foi aceita e, com enorme prazer, preparamos este texto.

Nossa ideia foi organizar um curso introdutório sobre métodos sem derivadas para minimização irrestrita, no qual possamos passar aos interessados o ferramental matemático indispensável ao assunto e discutir aspectos teóricos e computacionais dos algoritmos mais importantes.

Principalmente por ser a primeira versão do texto, certamente surgirão sugestões, dúvidas e correções a serem feitas. Seremos gratos aos leitores que nos contactarem a esse respeito.

Esperamos ter conseguido atingir nosso objetivo básico, e esperamos também ter a chance de manter um contato mais estreito com alunos e pesquisadores brasileiros que queiram, conosco, formar um grupo "Derivative-Free" mais amplo.

Campinas, abril de 2010.

Maria Aparecida Diniz-Ehrhardt
Véra Lucia da Rocha Lopes
Lucas Garcia Pedroso

Capítulo 1

Introdução

Métodos sem derivadas, como a própria expressão explica, são aqueles que não utilizam derivadas em nenhum de seus passos. Estes vêm recebendo crescente atenção por parte da comunidade científica. Muitos procedimentos clássicos dessa categoria foram propostos na década de 60. A resistência enfrentada pelos autores na época foi grande, principalmente pela ausência de bons resultados teóricos. No entanto, já há muitos anos as publicações direcionadas ao assunto trazem teoremas tão expressivos quanto os apresentados para métodos baseados em derivadas.

Ao abandonar o uso de derivadas das funções envolvidas nos problemas, seja na forma de seus gradientes, hessianas, derivadas direcionais ou outros, somos privados de aplicar muitas estratégias corriqueiras em otimização. Apesar de ainda assim bons resultados teóricos serem atingidos, é evidente que o não uso de derivadas acarreta um desempenho inferior em relação aos métodos clássicos da literatura, especialmente no que se refere ao tempo computacional. A proposta da otimização sem derivadas não é, pois, desenvolver algoritmos que superem os tradicionais, senão resolver os problemas onde as derivadas não estão disponíveis. Esse é o caso quando a função objetivo é uma caixa-preta, ou seja, um arquivo executável cujo código não está acessível ao usuário. Ou simplesmente quando não há expressão para a função objetivo, como na situação em que esta é uma medida de desempenho computacional de um algoritmo [4]. Mas talvez o contexto mais relevante de impossibilidade de cálculo de derivadas seja o de simulações. Quando a função objetivo é uma simulação, não faz sentido calcularmos suas derivadas. Isso é muito frequente em problemas práticos, notoriamente naqueles que envolvem *design* de veículos aéreos [3, 7]. Neste caso,

propostas de algoritmos sem derivadas eficazes se fazem necessárias.

Outra razão que justifica o estudo de métodos sem derivadas é a possibilidade de aplicá-los de modo quase imediato, seja pela estrutura via de regra simples dos algoritmos sem derivadas clássicos, que se traduz em uma programação descomplicada, ou pelo fato de não ser necessário implementar gradientes ou hessianas ou mesmo recorrer a pacotes de diferenciação automática. Não há dúvidas que o tempo despendido pelo usuário para transcrever o problema para código computacional se reduz drasticamente ao não ser mais necessário implementar as derivadas, ainda que o tempo de execução do programa seja potencialmente grande.

Há uma classe especial de algoritmos sem derivadas que merece ser destacada, que é a dos métodos de busca direta [19].

Definição 1.1. *Dizemos que um método é de busca direta se, além de não computar derivadas, ele não utilizar os valores de função em nenhum cálculo.*

Só é permitido a um algoritmo de busca direta verificar qual entre dois pontos tem o menor valor de função objetivo. Os valores de função nunca precisam ser explicitados de fato. Por exemplo, um algoritmo que interpola a função objetivo por quadráticas pode ser sem derivadas, porém não de busca direta. Uma consequência imediata dessa definição é que tais métodos não podem impor critérios de decréscimo suficiente da função objetivo.

A expressão busca direta surgiu em 1961, em um artigo de Hooke e Jeeves [16], onde a definem da seguinte forma:

"We use the phrase direct search to describe sequential examination of trial solutions involving comparison of each trial solution with the best obtained up to that time together with a strategy for determining (as a function of earlier results) what the next trial solution will be."

Aqui pretendemos tratar de alguns algoritmos para minimização irrestrita sem derivadas propostos na literatura, muitos deles classificados como de busca direta. Sem a pretensão de querer esgotar a extensa lista de algoritmos desse tipo, nosso intuito é apresentar os métodos que consideramos mais importantes nessa área, de autores que vêm trabalhando no assunto já a algum tempo. Destacamos as contribuições de Virgínia Torczon, Michael Powell [30, 28], Philippe Toint, Luis Vicente, entre outros.

Há diversos trabalhos dignos de atenção, seja pela importância histórica, pelo bom desempenho prático ou pela popularidade gozada pelos algoritmos. Citamos alguns deles abaixo, em ordem de publicação. O intuito é apresentar uma cronologia da minimização irrestrita sem derivadas, para que consigamos estabelecer um bom roteiro de estudo sobre o assunto. Apesar das muitas omissões, cremos que a lista abaixo dá uma boa ideia de como os métodos sem derivadas foram ganhando espaço nos diversos meios de divulgação científica.

- **1961:** Hooke e Jeeves propõem a expressão *busca direta,* juntamente com um algoritmo sem derivadas;
- **1962:** Spendley, Hext e Himsworth apresentam um algoritmo sem derivadas baseado no conceito de simplex no $\mathbb{R}^n$;
- **1965:** Nelder e Mead propõem uma versão modificada do Algoritmo de Spendley et al, acrescentando novas possibilidades de modificação dos simplex. Este algoritmo ainda é um dos algoritmos sem derivadas mais utilizados na prática;
- **1971:** Polak apresenta sua versão do Algoritmo de Busca Coordenada, sob o nome de Variações Locais. Essa é uma das várias formalizações da busca coordenada, que foi apresentado por diversos autores devido ao seu caráter intuitivo;
- **1997:** Torczon aborda os algoritmos de busca padrão, categoria que engloba, por exemplo, os algoritmos de Busca Coordenada e de Hooke e Jeeves. Uma teoria de convergência que vale para esses algoritmos é apresentada;
- **1998:** McKinnon demonstra que, para uma determindada classe de funções diferenciáveis e estritamente convexas em $\mathbb{R}^2$, o Algoritmo Nelder-Mead converge a um ponto não estacionário, acabando com as esperanças de encontrar uma teoria de convergência para o método;
- **2002:** Powell apresenta a primeira versão de seu algoritmo sem derivadas que utiliza interpolações quadráticas da função objetivo, que até hoje é um dos mais bem sucedidos dentre os que fazem uso dessa estratégia;
- **2006:** Audet e Dennis introduzem o algoritmo MADS para minimização com e sem restrições que, além de possuir outros aspectos interessantes, generaliza a busca padrão para o caso de funções não suaves;

- **2009:** Conn, Scheinberg e Vicente lançam um livro sobre métodos sem derivadas, resumindo boa parte do que foi publicado até o momento na área, possivelmente o primeiro da literatura.

Nosso texto está organizado da seguinte maneira. No Capítulo 2, apresentamos conceitos e resultados fundamentais em otimização, bem como descrevemos os métodos clássicos para o problema de minimização sem restrições. Embora o objetivo deste texto seja apresentar aspectos da otimização sem derivadas, consideramos que este Capítulo 2 se faz necessário para apontar as dificuldades que podemos ter quando temos que nos privar do uso das derivadas. Nos capítulos restantes, tratamos das classes mais comuns de algoritmos sem derivadas. No Capítulo 3, abordamos os métodos baseados em simplex, como o Algoritmo Nelder-Mead. No Capítulo 4, discutimos os métodos que utilizam direções de busca, como a Busca Coordenada e a Busca Padrão. Por fim, no Capítulo 5, tratamos dos algoritmos introduzidos por Powell, que utilizam aproximações quadráticas para a função objetivo.

Capítulo 2

Introdução à Otimização

Vamos considerar o problema

$$\min f(x), \quad \text{sujeita a} \quad x \in S, \tag{2.1}$$

onde $S \subseteq \mathbb{R}^n$ e $f : S \to \mathbb{R}$.

Definição 2.1. *Um ponto $x^* \in S$ é um* **minimizador local** *de f em S se e somente se existe $\varepsilon > 0$ tal que $f(x) \geq f(x^*)$, para todo $x \in S \cap B(x^*, \varepsilon)$. Se $f(x) > f(x^*)$, para todo $x \in S \cap B(x^*, \varepsilon)$ e $x \neq x^*$, então x^* é um* **minimizador local estrito** *de f.*

Definição 2.2. *Um ponto $x^* \in S$ é um* **minimizador global** *de f em S se e somente se $f(x) \geq f(x^*)$, para todo $x \in S$. Se $f(x) > f(x^*)$, para todo $x \in S$ e $x \neq x^*$, então x^* é um* **minimizador global estrito** *de f (se x^* é minimizador global, dizemos também que x^* é a solução ótima de (2.1)).*

Consideremos a partir de agora o caso $S \equiv \mathbb{R}^n$, ou seja, o problema

$$\min f(x), \tag{2.2}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$.

2.1 Condições de Otimalidade

Sabemos que, se $f : \mathbb{R} \to \mathbb{R}$ e:

- f é diferenciável;

- x^* é um minimizador local de f em $\mathbb{R}$;

então $f'(x^*) = 0$.

E se:

- f é duas vezes diferenciável;
- x^* é um minimizador local de f em $\mathbb{R}$;

então $f'(x^*) = 0$ e $f''(x^*) \geq 0$.

Por outro lado, se $f'(x^*) = 0$ e $f''(x^*) > 0$, então x^* é um minimizador local de f em $\mathbb{R}$.

Em $\mathbb{R}^n$:

Proposição 2.1. [Condições Necessárias de Primeira Ordem] *Seja $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^1$. Se x^* é um minimizador local de f em $\mathbb{R}^n$, então $\nabla f(x^*) = 0$ (um ponto que satisfaz condições necessárias de primeira ordem é chamado de estacionário).*

Demonstração: Seja $d \in \mathbb{R}^n$ arbitrário. Consideremos a função $\phi : \mathbb{R} \to \mathbb{R}$ definida por:

$$\phi(\lambda) = f(x^* + \lambda d).$$

Como x^* é minimizador local de f, então existe $\varepsilon > 0$ tal que $f(x^*) \leq f(x)$, para todo $x \in B(x^*, \varepsilon)$. Se tomarmos λ suficientemente pequeno, então $f(x^*) \leq f(x^* + \lambda d)$. Mas $\phi(0) = f(x^*)$, portanto $\phi(0) \leq \phi(\lambda)$, o que implica que $\lambda = 0$ é minimizador local de ϕ. Sabemos então que $\phi'(0) = 0$. Como $\phi'(\lambda) = \nabla^T f(x^* + \lambda d)d$, então $\phi'(0) = \nabla^T f(x^*)d = 0$. Logo $\nabla f(x^*)$ é ortogonal a d, um vetor arbitrário de $\mathbb{R}^n$, ou seja, $\nabla f(x^*)$ é ortogonal a qualquer vetor de $\mathbb{R}^n$. Portanto $\nabla f(x^*) = 0$. ■

Proposição 2.2. [Condições Necessárias de Segunda Ordem] *Seja $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^2$. Se x^* é um minimizador local de f em $\mathbb{R}^n$, então:*

1. $\nabla f(x^*) = 0$;
2. $\nabla^2 f(x^*)$ *é semidefinida positiva.*

Demonstração:

1. ver Proposição 2.1.

2. Consideremos $\phi(\lambda)$ como na Proposição 2.1. Vimos que $\lambda = 0$ é um minimizador local de ϕ. Então sabemos que $\phi''(0) \geq 0$. Como $\phi'(\lambda) = \nabla^T f(x^* + \lambda d)d$, temos que:

$$\phi''(\lambda) = d^T \nabla^2 f(x^* + \lambda d)d$$

$$\phi''(0) = d^T \nabla^2 f(x^*)d \geq 0.$$

Como isto ocorre para todo $d \in I\!R^n$, concluímos que $\nabla^2 f(x^*)$ é semi-definida positiva. ■

Proposição 2.3. [Condições Suficientes de Segunda Ordem] *Seja $f : I\!R^n \to I\!R$, $f \in C^2$. Se $x^* \in I\!R^n$ satisfaz:*

1. $\nabla f(x^*) = 0$;
2. $\nabla^2 f(x^*)$ *definida positiva,*

então x^* é um minimizador local estrito de f em $I\!R^n$.

Demonstração: Seja $B = \{h \in I\!R^n : \|h\| = 1\}$ e consideremos a função:

$$\Gamma : B \subset I\!R^n \to I\!R$$

$$\Gamma(h) = h^T \nabla^2 f(x^*)h.$$

Por hipótese, $\Gamma(h) > 0$ para todo $h \in B$. Além disso, vemos que $\Gamma(h)$ é uma função contínua e B um conjunto fechado e limitado. Usando o teorema de Weierstrass, temos que $\Gamma(h)$ atinge um valor mínimo e um valor máximo em B. Se a é seu valor mínimo, então

$$\Gamma(h) \geq a.$$

Supondo $a = \Gamma(h^*)$, $h^* \in B$, e desde que $\Gamma(h) > 0$, para todo $h \in B$, então $a > 0$. Consideremos agora $d \in I\!R^n$ arbitrário, $d \neq 0$. Temos que $\frac{d}{\|d\|} \in B$. Como $\Gamma(h) = h^T \nabla^2 f(x^*)h \geq a$ para todo $h \in B$, então

$$d^T \nabla^2 f(x^*)d \geq a\|d\|^2. \tag{2.1.1}$$

Usando a expansão de Taylor de $f(x^* + d)$ em torno de x^* temos:

$$f(x^* + d) - f(x^*) = \nabla^T f(x^*)d + \frac{1}{2}d^T \nabla^2 f(x^*)d + o(\|d\|^2).$$

Como $\nabla f(x^*) = 0$, por (2.1.1):

$$f(x^* + d) - f(x^*) \geq \frac{a}{2}\|d\|^2 + o(\|d\|^2).$$

Para d tal que $\|d\|$ é suficientemente pequena, o primeiro termo do lado direito da desigualdade define o sinal deste lado. Como $\frac{a}{2}\|d\|^2 > 0$, então $f(x^*+d) > f(x^*)$ para $\|d\|$ suficientemente pequena não nula ($0 < \|d\| < \varepsilon$), o que implica que $f(x) > f(x^*)$ para todo $x \in B(x^*, \varepsilon)$. Portanto, x^* é minimizador local estrito de $f(x)$. ■

2.2 Direções de Descida

Dado $x \in I\!R^n$, se $\nabla f(x) \neq 0$, sabemos que x não é minimizador local de $f(x)$ em $I\!R^n$. Portanto, em toda vizinhança de x existe z tal que $f(z) < f(x)$. Como nosso objetivo é minimizar $f(x)$, nos interessa caracterizar as direções a partir de x em que é possível achar $z \in I\!R^n$ tal que $f(z) < f(x)$. Antes, porém, o seguinte lema é importante.

Lema 2.1. *Seja $f : I\!R^n \to I\!R$ continuamente diferenciável em um conjunto aberto e convexo $D \subset I\!R^n$. Então para todo $x \in D$ e $d \in I\!R^n$, a derivada direcional de f em x na direção d, definida por:*

$$D_d f(x) = lim_{\lambda \to 0} \frac{f(x + \lambda d) - f(x)}{\lambda},$$

existe e é igual a $\nabla^T f(x)d$.

Demonstração: ver [12]

Definição 2.3. *Dados $x \in I\!R^n$, $d \in I\!R^n$ é uma direção de descida para f a partir de x se existe $\varepsilon > 0$ tal que para todo $\lambda \in (0, \varepsilon]$, $f(x + \lambda d) < f(x)$.*

Proposição 2.4. *Seja $f : I\!R^n \to I\!R$, $f \in C^1$, e $x \in I\!R^n$ tal que $\nabla f(x) \neq 0$. Seja $d \in I\!R^n$ e $\nabla^T f(x)d < 0$. Então existe $\bar{\lambda} > 0$ tal que $f(x + \lambda d) < f(x)$, para todo $\lambda \in (0, \bar{\lambda}]$. (Em outras palavras, se $D_d f(x) < 0$, d é direção de descida).*

Demonstração: ver [14]

Os métodos para resolução de problemas da forma de (2.2) são procedimentos iterativos. A partir de uma aproximação inicial $x^0 \in I\!R^n$ dada, a cada iteração k, obtida a aproximação x^k, calculamos uma direção de descida d^k sobre a qual deve estar a nova aproximação x^{k+1}. Algoritmos que encaixam–se neste esquema geral são conhecidos como **métodos de**

descida.

No entanto, uma direção de descida pode ser adequada somente localmente.

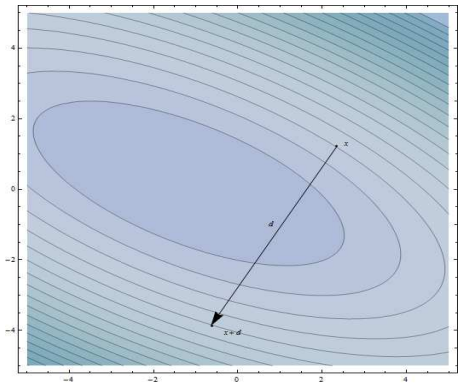

Figura 2.1: Longe da solução.

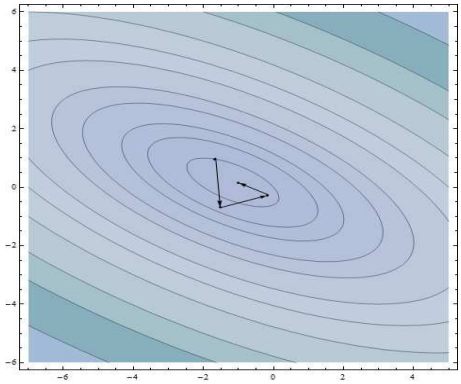

Figura 2.2: Próximo à solução.

Vemos, na Figura 2.1, que $f(x+d) > f(x)$ e isto pode levar à divergência do método. Este problema em geral é contornado quando ocorre perto da solução (ver Figura 2.2); ou seja, perto da solução, ainda que, em alguma iteração, $f(x+d) > f(x)$, o método, em geral, volta a uma região onde a função passa a decrescer, até convergir à solução.

Para contornar então as dificuldades que podem existir quando partimos de um x^0 arbitrário, possivelmente longe da solução, surgem estratégias que garantem o decréscimo da função f em toda iteração, e que podem ser incorporadas a um método de descida, garantindo-lhe resultados de convergência a pontos estacionários de f, qualquer que seja x^0. Dentre estas técnicas, destacamos o procedimento de **busca linear**, que abordaremos na seção seguinte.

2.3 Busca Linear

Vamos descrever, a seguir, um esquema geral para métodos de descida, que prevê o cálculo de um passo escalar, ao longo da direção de busca d^k, que faz com que o valor da função f, nesta direção, decresça de x^k para x^{k+1}.

Algoritmo 2.1.

Dado $x^0 \in I\!R^n$; faça $k = 0$;
Enquanto $\nabla f(x^k) \neq 0$:

Passo 1. *Direção de busca.*
Determine uma direção de descida d^k;

Passo 2. *Busca linear.*
Calcule $\lambda_k > 0$ tal que

$$f(x^k + \lambda_k d^k) < f(x^k); \tag{2.3.1}$$

Passo 3. *Nova aproximação.*
Faça $x^{k+1} = x^k + \lambda_k d^k$;

Passo 4. *Atualização da iteração.*
Faça $k = k + 1$.

Com a busca linear, desejamos encontrar o tamanho do passo λ_k que, tomado na direção d^k, forneça uma aproximação x^{k+1}, onde o valor da função objetivo f seja menor. Definimos novamente uma função $\phi : I\!R \to I\!R$ por:

$$\phi(\lambda) = f(x^k + \lambda d^k).$$

Uma primeira ideia seria buscarmos λ_k que simplesmente satisfizesse o critério (2.3.1) do Algoritmo 2.1, ou seja, um decréscimo simples de f.

No entanto, além desta pequena exigência poder tornar o procedimento geral muito lento, podemos exibir exemplos simples que mostram que esta condição não garante a convergência da sequência gerada pelo método a uma solução do problema (2.2).

Exemplo 2.1. [12] *Seja* $f(x) = x^2$, $x_0 = 2$ *e* $d_k = \{-1\}$, *para todo* k. *Vamos tomar* $\lambda_k = 2^{-k-1}$. *A sequência* $\{x_k\}$ *gerada será:*

$$\{2, \frac{3}{2}, \frac{5}{4}, \frac{9}{8}, \cdots\} = \{1 + 2^{-k}\}.$$

Verificamos que d_k *é sempre uma direção de descida para todo* k, $f(x_k)$ *decresce monotonicamente e* $lim_{k\to\infty} = 1 \neq x_*$, *o minimizador local, que é zero!*

Por outro lado, buscar λ_k que minimize $\phi(\lambda)$, ou seja,

$$\lambda_k = argmin_\lambda f(x^k + \lambda d^k),$$

pode se tornar um procedimento computacionalmente caro.

Surgem então alternativas que exigem um "decréscimo suficiente"da função f ao longo da direção de busca, garantindo ainda resultados de convergência global. Entre estas vamos destacar uma das mais usadas: o critério de Armijo [1]. Até o final deste capítulo, vamos supor que f é continuamente diferenciável.

A ideia do critério de Armijo é medir um decréscimo suficiente da função objetivo, verificando se a taxa média em que f decresce de x^k para $x^k + \lambda d^k$ é, pelo menos, uma fração da taxa inicial de decréscimo nesta direção. Ou seja, pedimos que, para $\alpha \in (0, 1)$:

$$\frac{f(x^k + \lambda d^k) - f(x^k)}{\lambda} \leq \alpha \nabla^T f(x^k) d^k,$$

ou:

$$f(x^k + \lambda d^k) \leq f(x^k) + \alpha \lambda \nabla^T f(x^k) d^k. \tag{2.3.2}$$

A desigualdade (2.3.2) é conhecida como o critério de Armijo, que está representado geometricamente na Figura 2.3 [14]. Incorporando este critério ao Algoritmo 2.1, o passo de busca linear fica modificado, gerando o algoritmo a seguir.

Algoritmo 2.2.

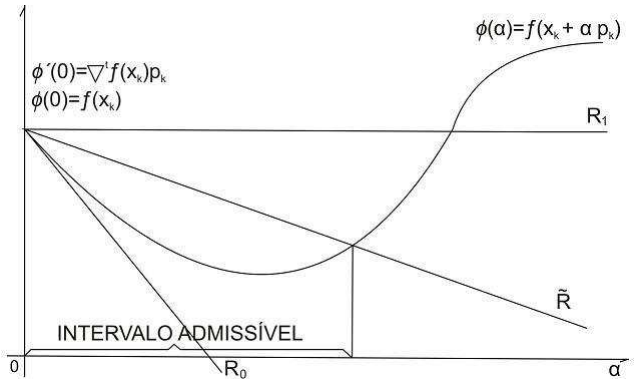

Figura 2.3: Critério de Armijo.

Dado $x^0 \in I\!R^n$; faça $k = 0$;
Os Passos 1, 3 e 4 do Algoritmo 2.2 são idênticos àqueles do Algoritmo 2.1.
O Passo 2 é substituído por:

Passo 2. *Busca linear.*
Faça $\lambda = 1$.
Enquanto

$$f(x^k + \lambda d^k) > f(x^k) + \alpha\lambda\nabla^T f(x^k)d^k,$$

reduza λ.
Faça $\lambda_k = \lambda$.

Além de algumas outras condições de salvaguarda que devem ser verificadas em relação à direção de busca, o Algoritmo 2.2 possui convergência global, no sentido de que, qualquer que seja $x^0 \in I\!R^n$, todo ponto limite da sequência $\{x^k\}$ gerada pelo algoritmo é um ponto estacionário de f. Para mais detalhes sobre este resultado, ver [12, 14].

2.4 Métodos Clássicos de Descida

Como ilustrado na Figura 2.2, se aplicarmos um método de descida ao problema (2.2) a partir de uma aproximação inicial x^0 próxima de um minimizador x^*, ainda que, em alguma iteração, o valor de f não decresça, o método pode convergir a uma solução do problema. Este tipo de convergência é conhecida como convergência local. Esta convergência pode se

dar sob várias taxas. Vamos definir a seguir as taxas ou ordens padrões de convergência.

Definição 2.4. *Seja* $\{x^k\} \subset \mathbb{R}^n$ *e* $x^* \in \mathbb{R}^n$*. Então:*

- $x^k \to x^*$ *quadraticamente se* $x^k \to x^*$ *e existe uma constante* $c > 0$ *tal que*
$$\|x^{k+1} - x^*\| \leq c\|x^k - x^*\|^2;$$
- $x^k \to x^*$ *superlinearmente se* $x^k \to x^*$ *e*
$$lim_{k\to\infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} = 0;$$
- $x^k \to x^*$ *linearmente se* $x^k \to x^*$ *e existe uma constante* $c \in (0, 1)$ *tal que*
$$\|x^{k+1} - x^*\| \leq c\|x^k - x^*\|,$$
para k *suficientemente grande.*

2.4.1 Método de Máxima Descida

O **Método de Máxima Descida (MMD)** é também conhecido como método do gradiente ou método de Cauchy. A direção de busca, neste algoritmo, é dada por:
$$d^k = -\nabla f(x^k). \tag{2.4.1}$$

O método tem este nome, "máxima descida", pois a direção $\frac{d}{\|d\|}$, com $d = -\nabla f(x)$ é, de fato, aquela em que a função f decresce mais rapidamente a partir de x, uma vez que, entre todas as direções com norma igual a 1, a direção oposta à do gradiente é a que fornece o menor valor para a derivada direcional $\nabla^T f(x)d$.

No entanto, o MMD apresenta algumas dificuldades. Ele exige, a cada iteração, um procedimento de busca linear exata ao longo da direção d^k, pois, caso contrário, mesmo em um exemplo simples como o descrito abaixo, a sequência pode divergir.

Exemplo 2.2. *Seja* $f(x) = x_1^2 + x_2^2$*. Suponha* $x^0 = (0\ \alpha)^T$*, para algum* $\alpha > 0$.
$\nabla f(x) = (2x_1\ 2x_2)^T$ *e* $\nabla f(x^0) = (0\ 2\alpha)^T$*. Assim:*
$$x^1 = x^0 - \nabla f(x^0) = (0\ -\alpha)^T$$

$$\nabla f(x^1) = (0 \ -2\alpha)^T$$
$$x^2 = x^1 - \nabla f(x^1) = (0 \ \alpha)^T = x^0.$$

Portanto a sequência diverge.

No caso quadrático, podemos adotar o procedimento de busca linear exata, pois o passo λ_k que minimiza a função f ao longo de d^k pode ser facilmente calculado (ver exercício 2). Já no caso geral isto pode se tornar um procedimento custoso computacionalmente. De qualquer forma, o MMD, com a direção de busca calculada por (2.4.1) e com busca linear exata, apresenta o seguinte resultado de convergência.

Teorema 2.1. *Seja $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^2$. Seja $x^* \in \mathbb{R}^n$ um minimizador local de f, tal que $\nabla^2 f(x^*)$ é definida positiva, sendo a e A, respectivamente, seu menor e maior autovalor. Se $\{x^k\}$ é uma sequência gerada pelo MMD que converge a x^* , então a sequência de valores $\{f(x^k)\}$ converge a $f(x^*)$ linearmente com taxa de convergência não maior que $(\frac{A-a}{A+a})^2$.*

Demonstração: ver [21]

Vemos então, pelo teorema acima, que quanto maior for a diferença entre a e A, a convergência do MMD se torna mais lenta. Veremos, a seguir, métodos que podem ter um desempenho melhor na prática.

2.4.2 Método de Newton

A motivação do **Método de Newton** está em, dado x^k, aproximar $f(x)$ por uma função quadrática $q_k(x)$ em torno de x^k, e obter x^{k+1} a partir da minimização de $q_k(x)$. Neste momento, vamos supor $f \in C^2$.

$$f(x) \approx q_k(x) = f(x^k) + \nabla^T f(x^k)(x - x^k) + \frac{1}{2}(x - x^k)^T \nabla^2 f(x^k)(x - x^k).$$

Queremos x^{k+1} tal que $\nabla q_k(x^{k+1}) = 0$.

$$\nabla q_k(x) = \nabla f(x^k) + \nabla^2 f(x^k)(x - x^k).$$

Portanto $\nabla q_k(x) = 0$ implica $\nabla^2 f(x^k)(x - x^k) = -\nabla f(x^k)$. Vamos supor que $\nabla^2 f(x^k)$ é definida positiva. Então x^{k+1} vai ser obtido por:

$$x^{k+1} = x^k + d^k, \tag{2.4.2}$$
$$d^k = -[\nabla^2 f(x^k)]^{-1} \nabla f(x^k). \tag{2.4.3}$$

Computacionalmente a direção d^k é obtida a partir da resolução do sistema linear:

$$\nabla^2 f(x^k)d = -\nabla f(x^k),$$

em que a matriz de coeficientes, por hipótese, é simétrica definida positiva. A fatoração de Cholesky [38] pode ser usada nesta resolução.

Observemos que, se $\nabla^2 f(x^k)$ é definida positiva, d^k é direção de descida (exercício 3).

Para estabelecermos resultados de convergência local para o método de Newton, a seguinte definição é necessária.

Definição 2.5. *Uma função $g : D \subset \mathbb{R}^n \to \Omega$ é Lipschitz contínua em D se*

$$\|g(x) - g(y)\| \leq \gamma \|x - y\| \tag{2.4.4}$$

para todo $x,\ y \in D$ e γ uma constante positiva.

Teorema 2.2. *Suponha que f é duas vezes continuamente diferenciável em um conjunto aberto e convexo $D \subset \mathbb{R}^n$ e $\nabla^2 f(x)$ é Lipschitz contínua em D. Suponha ainda que $x^* \in D$ é tal que $\nabla f(x^*) = 0$ e $\nabla^2 f(x^*)$ é definida positiva. Então existe $\varepsilon > 0$ tal que, se $x^0 \in B(x^*, \varepsilon)$, a sequência $\{x^k\}$ gerada por (2.4.2–2.4.3) verifica:*

- *$\nabla^2 f(x^k)$ é definida positiva, para todo $k = 0, 1, 2, \cdots$;*
- *$lim_{k\to\infty} x^k = x^*$;*
- *existe $c > 0$ tal que $\|x^{k+1} - x^*\| \leq c\|x^k - x^*\|^2$, para todo $k = 0, 1, 2, \cdots$ (taxa de convergência quadrática).*

Demonstração: ver [17].

Algumas observações sobre o método de Newton se fazem necessárias. Em primeiro lugar, a grande vantagem deste algoritmo é sua taxa quadrática de convergência. Por outro lado, o método apresenta desvantagens, que podem ser resumidas por:

- a exigência do cálculo das derivadas parciais de primeira e segunda ordens de f;
- a necessidade, a cada iteração, da resolução de um sistema linear;

- problemas se $\nabla^2 f(x^k)$ não é definida positiva.

Há métodos e estratégias alternativas que surgem para contornar estas dificuldades que o método de Newton pode apresentar. Na próxima subseção, vamos apresentar os métodos quase-Newton, que, por exemplo, não exigem o cálculo das derivadas de segunda ordem de f. No entanto, não conseguem manter a taxa de convergência quadrática.

2.4.3 Métodos Quase-Newton

Vimos que, o cálculo da hessiana, como também a necessidade da resolução de um sistema linear a cada iteração podem se constituir em grandes dificuldades para o método de Newton. A classe de **Métodos Quase-Newton** substitui a iteração de Newton por:

$$x^{k+1} = x^k + d^k, \tag{2.4.5}$$

$$d^k = -B_k^{-1} \nabla f(x^k). \tag{2.4.6}$$

A ideia é tentar que as matrizes B_k sejam aproximações razoáveis das hessianas. Se usarmos discretizações para as derivadas por diferenças finitas [12], por exemplo, estaríamos trabalhando com um método quase-Newton. Mas a classe dos **métodos secantes** é das mais usadas, pois estes conseguem, geralmente, aproximações satisfatórias exigindo que as matrizes B_k satisfaçam a **equação secante**. Para motivar esta ideia, vamos novamente trabalhar com a função quadrática. Suponha que:

$$f(x) = \frac{1}{2} x^T H x + b^T x + c,$$

tal que $\nabla f(x) = Hx + b$, com H simétrica definida positiva. Então:

$$\nabla f(x + d) = H(x + d) + b = Hx + b + Hd = \nabla f(x) + Hd.$$

Assim,

$$\nabla f(x + d) - \nabla f(x) = Hd,$$

Esta equação é conhecida como **equação secante** e é a base dos métodos secantes para aproximar B_k. Os métodos secantes, com aproximações simétricas definidas positivas para as hessianas, obedecem ao seguinte esquema geral:

- dados $x^0 \in \mathbb{R}^n$ e B_0 simétrica definida positiva;
- em cada iteração k, obter x^{k+1} por (2.4.5–2.4.6);

- determinar B_{k+1} simétrica definida positiva tal que

$$\begin{aligned} B_{k+1}s_k &= y_k, \\ s_k &= x^{k+1} - x^k, \\ y_k &= \nabla f(x^{k+1}) - \nabla f(x^k). \end{aligned}$$

Observamos, agora, que a equação secante foi generalizada e que não há uma única matriz que a satisfaça. As diferentes escolhas para a atualização B_{k+1} geram os diferentes métodos secantes. Vamos destacar aqui o **método BFGS**, que recebe este nome pois foi proposto por Broyden, Fletcher, Goldfarb e Shanno, em 1970.

Atualização BFGS para B_{k+1}:

$$B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^t B_k}{s_k^T B_k s_k}. \tag{2.4.7}$$

O método BFGS tem a vantagem, em relação ao método de Newton, de não incluir, em suas iterações, nenhuma avaliação de derivadas parciais de segunda ordem. No entanto, a taxa de convergência quadrática não se mantém nos métodos secantes. O teorema a seguir contempla os resultados de convergência local do algoritmo que usa atualizações BFGS.

Teorema 2.3. *Sob as mesmas hipóteses do Teorema 2.2, existem constantes positivas ε e δ tais que, se $x^0 \in B(x^*, \varepsilon)$ e $\|B_0 - \nabla^2 f(x^*)\| \leq \delta$, então o método BFGS (2.4.5–2.4.7) está bem definido e converge q–superlinearmente a x^*.*

Demonstração: ver [12]

Os Teoremas 2.2 e 2.3 estabelecem resultados de convergência local para os métodos de Newton e BFGS. Devemos salientar, no entanto, que se o procedimento de busca linear implementado no Passo 2 do Algoritmo 2.2 for acrescentado a estes métodos, eles apresentarão também convergência global.

Uma alternativa à estratégia de busca linear com decréscimo suficiente para obtenção de resultados de convergência global está na utilização de regiões de confiança, objeto da nossa próxima subseção.

2.5 Região de Confiança

Os métodos de **região de confiança** geram passos baseados na minimização de um modelo quadrático da função objetivo. Nesses métodos, definimos

uma região ao redor da atual aproximação x^k, na qual confiamos no modelo como uma representação adequada da função objetivo. Minimizando o modelo nesta região, calculamos um passo p^k, a partir de x^k. Se não houve um decréscimo suficiente de f ao longo de p^k, este passo não é aceito; então reduzimos o raio da região e achamos um novo minimizador. Se, por outro lado, a minimização do modelo gerou passos bons, predizendo o comportamento da função objetivo com precisão ao longo destes passos, o raio da região de confiança é uniformemente aumentado, permitindo novos passos longos e ambiciosos.

Para obter cada passo, buscaremos uma solução do subproblema

$$\begin{array}{l} \min\limits_{p\in\mathbb{R}^n} m_k(p) = f(x^k) + \nabla^T f(x^k)p + \dfrac{1}{2}p^T B_k p, \\ s.a \ \ \|p\| \leq \Delta_k, \end{array} \tag{2.5.1}$$

onde $m_k(p)$ é o modelo quadrático que aproxima f, B_k é uma aproximação para $\nabla^2 f(x^k)$ e $\Delta_k > 0$ é o raio da região de confiança.

Para atualização do raio da região de confiança Δ_k a cada iteração, consideramos a relação entre a função modelo m_k e a função objetivo f, determinando

$$\rho_k = \frac{f(x^k) - f(x^k + p^k)}{m_k(0) - m_k(p^k)}. \tag{2.5.2}$$

- Se ρ_k é negativo, o novo valor da função objetivo $f(x^k + p^k)$ é maior que o valor atual $f(x^k)$, e portanto o passo deve ser rejeitado.
- Se ρ_k está próximo de 1, há boa relação entre o modelo m_k e a função f, assim é seguro ampliar a região de confiança.
- Se ρ_k é positivo mas não próximo de 1, não alteraremos a região de confiança, mas se está perto de zero ou é negativo, encolheremos a região de confiança.

Algoritmo 2.3.

Dados $\bar{\Delta} > 0, \Delta_0 \in (0, \bar{\Delta})$, e $\eta \in [0, \frac{1}{4})$; para $k = 0, 1, 2, \cdots$, faça:

Passo 1: *Cálculo do passo p^k.*
Obtenha p^k (aproximadamente) resolvendo (2.5.1);

Passo 2: *Cálculo de ρ_k.*
Avalie ρ_k por (2.5.2);

Passo 3: *Atualização do raio da região de confiança.*
Se $\rho_k < \frac{1}{4}$
$\Delta_{k+1} = \frac{1}{4} \left\| p^k \right\|$
Senão
Se $\rho_k > \frac{3}{4}$ e $\left\| p^k \right\| = \Delta_k$
$\Delta_{k+1} = \min(2\Delta_k, \bar{\Delta})$
Senão
$\Delta_{k+1} = \Delta_k$;

Passo 4: *Atualização da iteração.*
Se $\rho_k > \eta$
$x^{k+1} = x^k + p^k$
Senão
$x^{k+1} = x^k$.

A análise de convergência global baseada em região de confiança depende do parâmetro η. Se $\eta = 0$, a sucessão $\{\nabla f(x^k)\}$ tem um limite no ponto zero. Para o teste de aceitação mais estrito, com $\eta > 0$, temos o resultado: $lim_{k\to\infty} \nabla f(x^k) = 0$.

2.6 Exercícios

1. Classifique os pontos estacionários de
$$f(x) = 2x_1^3 - 3x_1^2 - 6x_1x_2(x_1 - x_2 - 1) \ .$$
No caso de minimizadores ou maximizadores, indique se são locais ou globais.

2. Se o método de máxima descida com busca linear exata é aplicado ao problema (2.2), onde f é uma função quadrática com hessiana definida positiva, mostre que, em cada iteração k, o passo ótimo da busca linear é dado por
$$\lambda_k = \frac{\nabla f(x^k)^T \nabla f(x^k)}{\nabla f(x^k)^T \nabla^2 f(x) \nabla f(x^k)} \ .$$

3. Mostre que, no método de Newton, se $\nabla^2 f(x^k)$ é definida positiva, então d^k é direção de descida.

4. Seja $f(x) = {x_1}^4 + x_1x_2 + (1 + x_2)^2$ e tome $x^0 = (0\ 0)^T$.
(a) Por que o método de Newton para minimizar $f(x)$ não pode ser aplicado satisfatoriamente a partir de x^0 dado?
(b) Considere uma modificação ao método de Newton tal que d^k seja obtida por:
$$d^k = -(\nabla^2 f(x^k) + \mu_k I)^{-1} \nabla f(x^k)$$
para $\mu_k > 0$ dado. Se $\mu_0 = \frac{1}{2}$, calcule d^0 por este método. O resultado foi satisfatório? Por que? Analise esta modificação.

5. Mostre que as matrizes B_{k+1} obtidas pela atualização BFGS são simétricas se B_k é simétrica, e satisfazem a equação secante.

6. Considere
$$f(x) = x_1^2 + 2x_1x_2 + 2x_2^2 + x_2^4,$$
que tem o minimizador $x^* = (0\ 0)^T$ e $\nabla^2 f(x^*) = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}$.
Mostre que, se $x^0 = (1\ \ -1)^T$ e $B_0 = \nabla^2 f(x^0)$, então o método BFGS gera uma sequência de aproximações $\{B_k\}$ para as hessianas que obedece
$$lim_{k \to \infty} B_k = \nabla^2 f(x^*).$$
Discuta este resultado.

Capítulo 3

Métodos Baseados em Simplex

A primeira classe de métodos sem derivadas para minimização irrestrita que estudaremos é a dos métodos simplex. Um simplex é um conjunto de $n+1$ pontos em $\mathbb{R}^n$. Buscaremos sempre simplex não degenerados, definidos por pontos $x^1, x^2, \ldots, x^{n+1}$ (conhecidos como vértices do simplex) tais que o conjunto $\{x^1 - x^{n+1}, \ldots, x^n - x^{n+1}\}$ é linearmente independente. Ou seja, pontos que definem simplex degenerados são colineares no $\mathbb{R}^2$, coplanares no $\mathbb{R}^3$ e assim por diante. As figuras abaixo mostram exemplos de conjuntos $\{x^1, x^2, x^3\}$ em $\mathbb{R}^2$ que formam um simplex não degenerado e um degenerado.

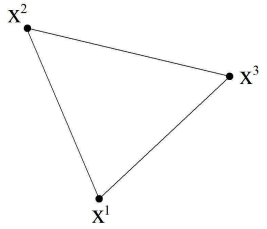

Figura 3.1: Simplex não degenerado em $\mathbb{R}^2$

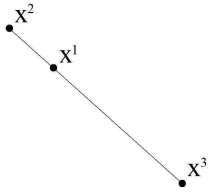

Figura 3.2: Simplex degenerado em $\mathbb{R}^2$.

A ideia de utilizar apenas $n+1$ pontos por iteração para definir um algoritmo de busca direta é razoável, visto que $n+1$ pontos seriam suficientes,

por exemplo, para aproximar o gradiente da função objetivo por diferenças finitas. Muitos métodos de busca direta calculam a função entre $2n$ e 2^n vezes por iteração, de onde já podemos deduzir uma vantagem dos métodos simplex: são econômicos, no que diz respeito a avaliações de função por iteração, quando comparados a outros métodos de busca direta.

Os algoritmos simplex formam um conjunto particular de métodos, com motivações geométricas e teorias de convergência próprias. Possuem também uma grande importância histórica. Por essa razão, julgamos importante dedicar um capítulo à classe. Trataremos aqui do Algoritmo Nelder-Mead, que é certamente um dos algoritmos sem derivadas mais famosos. Além dele, abordaremos rapidamente o Algoritmo de Busca Multidirecional, menos conhecido, mas com interessantes propriedades teóricas.

3.1 Algoritmo Nelder-Mead

O primeiro método simplex da literatura foi publicado em 1962 e é devido a Spendley, Hext e Himsworth [33]. O método procura substituir o pior vértice refletindo-o através do centroide da face oposta, como na Figura 3.3

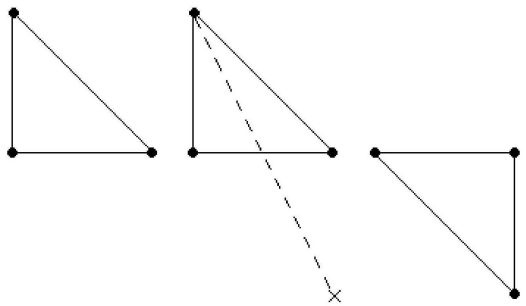

Figura 3.3: Reflexão de um vértice através do centroide da face oposta.

O método de Nelder-Mead [24], publicado em 1965, é provavelmente o mais utilizado método de busca direta. A contribuição do Algoritmo Nelder-Mead à classe dos métodos simplex reside na possibilidade de contração ou expansão do simplex. A Figura 3.4 mostra as possibilidades de substituição para o pior ponto, enquanto a Figura 3.5 mostra como é a redução.

São necessários 4 coeficientes escalares no algoritmo de Nelder-Mead. São eles:

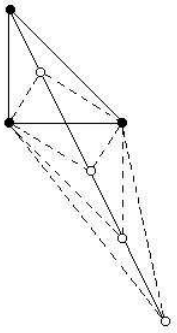

Figura 3.4: Substituição do pior vértice no Algoritmo Nelder-Mead.

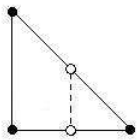

Figura 3.5: Redução do simplex no Algoritmo Nelder-Mead.

- coeficiente de reflexão: $\rho > 0$,
- coeficiente de expansão: $\chi > 1$ com $\chi > \rho$,
- coeficiente de contração: $0 < \gamma < 1$ e
- coeficiente de redução: $0 < \sigma < 1$.

Os valores mais utilizados na prática para os parâmetros são $\rho = 1$, que permite uma reflexão exata (que não expande nem contrai o vetor refletido), $\chi = 2$, de modo que a expansão dobre a distância entre o novo vértice e o centroide da face oposta, em comparação com a distância original entre o pior vértice e o centroide, $\gamma = 0.5$, de modo que as distâncias nas contrações sejam metade das originais e $\sigma = 0.5$, o que faz com que a distância entre o melhor vértice e os restantes seja reduzida pela metade em caso de passo de redução.

Para uma função $f : \mathbb{R}^n \to \mathbb{R}$ e um simplex no $\mathbb{R}^n$, a iteração de Nelder-Mead se dá conforme o algoritmo abaixo:

Algoritmo 3.1. *Algoritmo Nelder-Mead*

Passo 1: *(Ordenação)* Ordene os vértices do simplex de maneira que $f(x^1) \leq f(x^2) \leq \cdots \leq f(x^{n+1})$. Calcule o centroide dos n melhores pontos,

$\overline{x} = \sum_{i=1}^{n} x^i/n$.

Passo 2: *(Reflexão)* Calcule o ponto de reflexão $x^r = (1+\rho)\overline{x} - \rho x^{n+1}$. Se $f(x^1) \leq f(x^r) < f(x^n)$, aceite o ponto x^r e termine a iteração.

Passo 3: *(Expansão)* Se $f(x^r) < f(x^1)$, calcule o ponto de expansão $x^e = (1+\rho\chi)\overline{x} - \rho\chi x^{n+1}$. Se $f(x^e) < f(x^r)$, aceite x^e e termine a iteração. Caso contrário ($f(x^e) \geq f(x^r)$), aceite x^r e termine a iteração.

Passo 4: *(Contração)* Se $f(x^r) \geq f(x^n)$ faça uma contração:

- *(Contração Externa)* Se $f(x^n) \leq f(x^r) < f(x^{n+1})$, calcule $x^c = (1+\rho\gamma)\overline{x} - \rho\gamma x^{n+1}$. Se $f(x^c) \leq f(x^r)$ aceite x^c e termine a iteração. Caso contrário, vá para o Passo 5.
- *(Contração Interna)* Se $f(x^r) \geq f(x^{n+1})$, calcule $x^c = (1-\gamma)\overline{x} + \gamma x^{n+1}$. Se $f(x^c) < f(x^{n+1})$, aceite x^c e termine a iteração. Caso contrário, vá para o Passo 5.

Passo 5: *(Redução)* Calcule os pontos $v^i = x^1 + \sigma(x^i - x^1), i = 2, \ldots, n+1$. Os vértices (ainda fora de ordem) para a próxima iteração são x^1, v^2, ..., v^{n+1}.

Possíveis critérios de parada para o algoritmo envolvem a distância entre os vértices e a diferença entre os valores de função nos vértices, que devem ser menores do que tolerâncias escolhidas pelo usuário.

No processo de ordenação dos vértices do simplex no início da iteração podem ocorrer empates entre o valor da função de dois ou mais pontos. No artigo original de Nelder e Mead não é mencionado como proceder em tal situação. Em [18], é sugerida uma regra de desempate:

Iteração sem redução do simplex: Quando não há redução do simplex, apenas o vértice de índice $n+1$ é descartado e substituído por um ponto que chamaremos de v. O ponto v tomará a posição $j+1$ no simplex da iteração seguinte, onde $j = \max_{0 \leq m \leq n}\{m \mid f(v) < f(x_{m+1})\}$. Os outros vértices continuam na ordem que foi estabelecida antes de se iniciar a iteração.

Iteração com redução do simplex: Nesse caso, apenas o vértice x^1 será aproveitado na iteração seguinte. Apenas uma regra de desempate é necessária: caso o vértice x^1 empate com um ou mais vetores como sendo o ponto de melhor valor da função no novo simplex. Caso isso ocorra, faremos

$x^{1,k+1} = x^{1,k}$. Em qualquer outra situação, poderemos fazer uso de qualquer outra regra de desempate depois de uma redução.

Apesar de ser largamente empregado, o Método Nelder-Mead não tem garantias de convergência para dimensões superiores a 2. Em [18] os autores provam os seguintes resultados:

Teorema 3.1. *Seja $f : \mathbb{R} \to \mathbb{R}$ estritamente convexa com curvas de nível limitadas. Se o simplex inicial para o Algoritmo Nelder-Mead com $\rho\chi \geq 1$ é não degenerado, então os dois vértices convergem ao minimizador.*

Teorema 3.2. *Seja $f : \mathbb{R}^2 \to \mathbb{R}$ estritamente convexa com curvas de nível limitadas. Assuma que o Método Nelder-Mead está usando os parâmetros $\rho = 1$ e $\gamma = 0.5$. Se o simplex inicial é não degenerado, então os três vértices convergem a pontos com mesmo valor de função.*

Teorema 3.3. *Seja $f : \mathbb{R}^2 \to \mathbb{R}$ estritamente convexa com curvas de nível limitadas. Assuma que o Método Nelder-Mead está usando os parâmetros $\rho = 1$, $\chi = 2$ e $\gamma = 0.5$. Se o simplex inicial é não degenerado, então a sequência de diâmetros dos simplex tende a zero, ou seja,*

$$\max_{i \neq j} \|x^{ik} - x^{jk}\| \to 0.$$

É importante frisar que os Teoremas 3.2 e 3.3 não garantem a convergência para funções estritamente convexas em $\mathbb{R}^2$, pois o limite da sequência dos vértices pode ser um ponto não estacionário.

Não é raro encontrarmos na prática problemas para os quais o algoritmo converge a pontos não estacionários, fenômeno normalmente atribuído ao fato da direção de busca (ou seja, a direção definida pelo pior vértice e pelo centroide dos vértices restantes) ficar numericamente ortogonal ao gradiente. No entanto, em [22], o autor apresenta uma família de funções estritamente convexas e diferenciáveis onde o método falha ao convergir a um ponto não estacionário, deixando patente que não há como garantir a convergência para o método sob condições razoáveis. Por exemplo, para a função estritamente convexa e com derivadas contínuas

$$f(x,y) = \begin{cases} 360x^2 + y + y^2, & x \leq 0, \\ 6x^2 + y + y^2, & x > 0, \end{cases} \tag{3.1.1}$$

o método converge para a origem, que não é sequer ponto estacionário, uma vez que $\nabla f(x,y) = (0,1)^T$. Entretanto, o mau comportamento do método pode ser evitado impondo, por exemplo, decréscimo suficiente e restrições sobre os ângulos internos do simplex, como em [36], onde o autor propõe um Algoritmo Nelder-Mead modificado com garantias de convergência.

3.2 Busca Multidirecional

O Algoritmo de Busca Multidirecional foi introduzido por Dennis e Torczon em [35]. Inspirado nos trabalhos de Spendley, Hext e Himsworth, além de Nelder e Mead, o algoritmo traz o diferencial de possuir resultados de convergência (o algoritmo de Spendley, Hext e Himsworth possui teoria de convergência apenas se modificado em alguns detalhes, e somente para funções convexas).

Ao invés de se concentrar em modificar o pior vértice, como no contexto de Nelder-Mead, o algoritmo rotaciona todo o simplex, mantendo fixo apenas o melhor vértice. Se o melhor entre os vértices novos é melhor do que o melhor vértice antigo, então uma expansão é experimentada. Caso contrário, uma redução é feita. As Figuras 3.6 e 3.7 mostram os passos possíveis para o método.

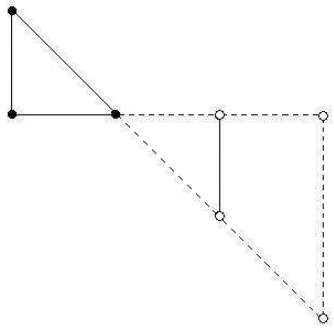

Figura 3.6: Rotação do simplex para a Busca Multidirecional.

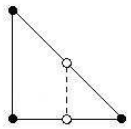

Figura 3.7: Redução do simplex para a Busca Multidirecional.

Uma iteração do algoritmo é mostrada abaixo. São necessários os parâmetros $0 < \sigma < 1 < \chi$, além do simplex $\{x^1, \ldots, x^{n+1}\}$.

Algoritmo 3.2. *Algoritmo de Busca Multidirecional*

Passo 1: *(Ordenação)* Ordene os vértices do simplex de maneira que $f(x^1) \leq f(x^2) \leq \cdots \leq f(x^{n+1})$.

Passo 2: *(Rotação)* Rotacione o simplex ao redor do melhor vértice,

fazendo

$$x_r^i = x^1 - (x^i - x^1), \;\; i = 2, \ldots, n+1.$$

Faça $f^r = \min_{i=2,\ldots,n+1}\{f(x_r^i)\}$. Se $f^r < f(x^1)$, vá ao Passo 3. Caso contrário, vá ao Passo 4.

Passo 3: *(Expansão)* Expanda o simplex rotacionado, fazendo

$$x_e^i = x^1 - \chi(x^i - x^1), \;\; i = 2, \ldots, n+1.$$

Faça $f^e = \min_{i=2,\ldots,n+1}\{f(x_e^i)\}$. Se $f^e < f^r$, aceite a expansão. Caso contrário, aceite a rotação. Termine a iteração.

Passo 4: *(Redução)* Calcule os pontos $v^i = x^1 + \sigma(x^i - x^1), i = 2, \ldots, n+1$. Os vértices (ainda fora de ordem) para a próxima iteração são x^1, v^2, $\ldots$, v^{n+1}.

A convergência para uma subsequência, conforme enunciado no Teorema 3.4, depende de suposições sobre os parâmetros σ e χ e o simplex inicial.

Teorema 3.4. *Suponha que σ, χ são racionais e seja o simplex inicial formado pelos vértices $x^i = Gz^i$, onde $G \in \mathbb{R}^{n\times n}$ é não singular e z^i é um vetor com componentes inteiras, $i = 1, \ldots, n+1$. Assuma que $L(x^0) = \{x \in \mathbb{R}^n$ tal que $f(x) \leq f(x^0)\}$ é compacto para x^0 sendo o melhor vértice do simplex inicial, e que f é continuamente diferenciável em $L(x^0)$. Então a sequência de pontos gerada pelo Algoritmo 3.2 possui um ponto limite estacionário.*

As hipóteses sobre os valores que devem ser racionais ou inteiros garante que os simplex vão permanecer sobre uma treliça racional, o que é fundamental na teoria de convergência. Em [34], a autora verifica que a busca multidirecional pode ser vista tanto como um algoritmo simplex como uma busca padrão, que será assunto no próximo capítulo.

3.3 Exercícios

1. Como são os simplex não degenerados em $\mathbb{R}^1$? E os degenerados? E em $\mathbb{R}^3$?

2. Suponha que um algoritmo simplex que reflete o pior vértice possui um vértice em um minimizador de uma função no $\mathbb{R}^2$. Suponha que o vértice refletido nunca é o pior vértice. Mostre com um desenho que é possível que sucessivas reflexões do pior vértice levem de volta ao simplex inicial.

3. Discuta como construir o simplex inicial no algoritmo de Nelder-Mead.

4. Prove que a função (3.1.1) tem derivada contínua em todo $\mathbb{R}^2$.

5. Aponte as principais diferenças entre os Algoritmos Nelder-Mead e de Busca Multidirecional, especialmente quanto ao número de avaliações de função por iteração e às modificações no formato do simplex.

Capítulo 4

Métodos de Busca Direta Direcionais

Seria extremamente injusto fazer um estudo sobre métodos sem derivadas e não mencionar os de busca direta direcionais. Tais métodos sintetizam muito da filosofia da otimização sem derivadas, por seus algoritmos simples, de forte apelo geométrico, com robusta teoria de convergência. No início deste capítulo, trataremos dos algoritmos de busca direta direcionais mais antigos, uma vez que sua análise é importante inclusive para a compreensão dos algoritmos sem derivadas como um todo. Na segunda parte, citaremos alguns trabalhos mais recentes, de promissora aplicabilidade e que abrem espaço para novas possibilidades de pesquisa.

Suporemos, em geral, que a função objetivo $f(x)$ é continuamente diferenciável. Somente na seção 1.4, a função precisa apenas ser Lipschitz, uma vez que a teoria será nessa ocasião estendida para funções não suaves.

4.1 Métodos de Busca Padrão

Um dos métodos de busca direta direcional mais simples encontrados na literatura é o Método de Busca Coordenada, também conhecido como Método das Variações Locais [27]. Dado o ponto e o passo correntes $x^k \in I\!R^n$ e $\alpha_k \in I\!R_+$, o algoritmo analisa os pontos da forma $x^k + \alpha_k d$, onde $d \in D$, sendo D o conjunto das direções canônicas positivas e negativas; em outras palavras, D é o conjunto formado pelas colunas das matrizes I e $-I$, onde I é a matriz identidade de ordem n. A intenção é encontrar um ponto melhor

do que o atual, ou seja, encontrar $x^{k+1} = x^k + \alpha_k d$ tal que $f(x^{k+1}) < f(x^k)$. Se isso não for possível, o que ocorre se todos os pontos da forma $x^k + \alpha_k d$, $d \in D$ têm valor de função maior que $f(x^k)$, então continuamos no mesmo ponto, fazendo $x^{k+1} = x^k$, mas reduzimos o tamanho do passo, $\alpha_{k+1} = \alpha_k/2$. Desse modo, o algoritmo toma a seguinte forma:

Algoritmo 4.1. *Algoritmo de Busca Coordenada*

dados x^0 e α_0, para $k = 0, 1, 2, \ldots$

Passo 1: Se $f(x^k + \alpha_k \overline{d}) < f(x^k)$ para algum $\overline{d} \in D$, faça $x^{k+1} = x^k + \alpha_k d$ de modo que $f(x^k + \alpha_k d) < f(x^k)$, $d \in D$ e declare a iteração bem sucedida. Caso contrário, declare fracasso e faça $x^{k+1} = x^k$.

Passo 2: Se a iteração foi bem sucedida, defina $\alpha_{k+1} = \alpha_k$. Caso contrário, $\alpha_{k+1} = \alpha_k/2$.

As Figuras 4.1 e 4.2 mostram alguns passos do algoritmo em $\mathbb{R}^2$.

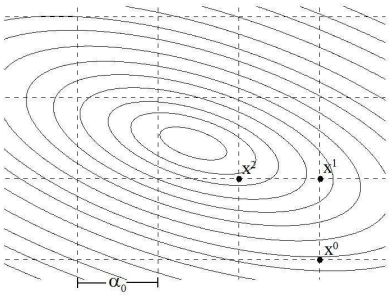

Figura 4.1: Busca Coordenada, $\alpha = \alpha_0$.

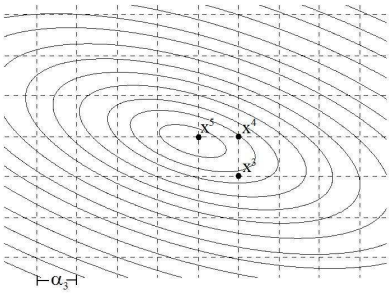

Figura 4.2: Busca Coordenada, $\alpha = \alpha_0/2$.

Iniciamos a busca no ponto x^0 com passo α_0, como mostrado na Figura 4.1. Os pontos possíveis de serem visitados estão acima, abaixo, à direita e à esquerda do ponto corrente, ou seja, nas direções coordenadas. De x^0 para x^1 há decréscimo da função; o mesmo ocorre de x^1 para x^2. Assim sendo, os passos são mantidos, $\alpha_2 = \alpha_1 = \alpha_0$. Já em x^2, não há decréscimo nas direções coordenadas para o passo α_2. Declaramos então fracasso na iteração, fazemos $x^3 = x^2$ e $\alpha_3 = \alpha_2/2$. As iterações seguintes estão representadas na Figura 4.2. É possível nos movermos de x^3 para x^4 e depois para x^5 com passos $\alpha_5 = \alpha_4 = \alpha_3$. Já em x^5, é necessário que declaremos

fracasso e reduzamos novamente o tamanho do passo, fazendo $\alpha_6 = \alpha_5/2$, $x^6 = x^5$ e assim sucessivamente.

Há diversos detalhes que podem originar diferentes métodos de Busca Coordenada. Poderíamos ter definido outro fator de redução do passo em caso de fracasso ao invés de $1/2$. Em geral, podemos escolhê-lo como sendo qualquer número racional no intervalo $(0, 1)$. Além disso, há a possibilidade de aumentarmos o tamanho do passo em caso de sucesso, o que pode ser útil se estivermos longe do minimizador. Por fim, a ordem em que as direções $d \in D$ são testadas é de livre escolha do usuário. Cabe também ao usuário decidir se a busca será completa, ou seja, se todas as direções serão testadas a cada iteração e aquela onde há maior decréscimo será tomada, ou se será oportunista, encerrando a iteração assim que encontra uma direção tal que $f(x^k + \alpha_k d) < f(x^k)$. Dada essa liberdade de escolhas, cada autor define de forma diferente os métodos de Busca Coordenada, deixando ou não livre a escolha dos parâmetros, da ordem das direções e do tipo de busca. Ver, por exemplo, os algoritmos em [9] e [34]. Em [10], os autores propõem uma estratégia de ordenação para as direções de busca da Busca Padrão de acordo com o ângulo formado entre estas e uma aproximação para o gradiente (chamada gradiente simplex). Em [11], é comentado que de fato essa estratégia pode ser usada em qualquer algoritmo que usa direções que geram positivamente o $\mathbb{R}^n$.

A maioria dos métodos baseados em direções utiliza busca monótona, ou seja, dado um ponto corrente x^k, o próximo iterando deve satisfazer $f(x^{k+1}) < f(x^k)$, ou possivelmente uma condição mais severa de decréscimo suficiente. Essa é uma das razões pelas quais o gradiente da função objetivo tem papel fundamental nos algoritmos direcionais baseados em derivadas, uma vez que, se o ponto corrente não é estacionário então podemos, utilizando o gradiente de f, verificar se uma determinada direção d^k é de descida, de modo que para α_k suficientemente pequeno tenhamos $f(x^k + \alpha_k d^k) < f(x^k)$. Isso sem considerar que $\nabla f(x^k)$ é um ótimo precursor de direções de descida, como a direção de máxima descida $-\nabla f(x^k)$ ou a direção de Newton $-\nabla^2 f(x^k)\nabla f(x^k)$. No contexto dos métodos sem derivadas, não podemos verificar se uma direção d é de descida a partir de x, pois não podemos computar $\nabla f(x)^T d$. O Algoritmo de Busca Coordenada contorna esse problema de modo muito semelhante ao feito por vários outros algoritmos de busca direta.

As direções $d \in D$ satisfazem uma propriedade conveniente, de gerar positivamente o $\mathbb{R}^n$. Em outras palavras, qualquer vetor $d \in \mathbb{R}^n$ pode ser

escrito como combinação linear dos vetores de D com coeficientes não negativos. É fácil observar isso, uma vez que $d = \sum_{i|d_i>0} d_i e^i + \sum_{i|d_i<0} |d_i|(-e^i)$. Se um conjunto $\overline{D}$ gera positivamente o $\mathbb{R}^n$, então para todo $v \in \mathbb{R}^n$, $v \neq 0$, existe ao menos uma direção $d \in \overline{D}$ tal que $v^T d < 0$ (analogamente, existe uma direção $\overline{d}$ tal que $v^T \overline{d} > 0$). Desse modo, concluímos que sempre há em $\overline{D}$ uma direção de descida a partir de qualquer ponto não estacionário. Além disso, analisando $f(x)$ através de direções que geram positivamente o $\mathbb{R}^n$ e usando apenas passos positivos, temos uma noção satisfatória de seu comportamento local, responsabilidade que nos métodos baseados em derivadas é em geral do gradiente.

O Algoritmo de Busca Coordenada pertence a uma classe mais ampla de métodos, os de Busca Padrão, ou Busca Padrão Generalizada (ou GPS, do inglês *Generalized Pattern Search*). Os Métodos de Busca Padrão também utilizam direções que geram positivamente o $\mathbb{R}^n$, e esse é um ingrediente fundamental tanto para a teoria de convergência como para um bom desempenho prático. A grosso modo, os diferentes métodos de Busca Padrão são definidos pelas diferentes direções de busca que utilizam. Em [34], a autora traça um modelo geral de algoritmo de Busca Padrão, onde se enquadram vários algoritmos preexistentes, entre eles o de Busca Coordenada. Uma teoria de convergência tão robusta quanto as teorias de métodos com derivadas é então apresentada.

Seja a *matriz base* $B \in \mathbb{R}^{n\times n}$ uma matriz não singular. Consideremos $M \subset \mathbb{Z}^{n\times n}$ um conjunto finito de matrizes não singulares e escolhamos $p \in \mathbb{N}$ tal que $p > 2n$. Na k-ésima iteração do algoritmo, teremos um padrão definido pela matriz

$$P^k = [BM^k \quad -BM^k \quad BL^k] = [B\Gamma^k \quad BL^k],$$

onde $M^k \in M$ e a matriz $L^k \in \mathbb{Z}^{n\times(p-2n)}$ contém ao menos uma coluna, que é a coluna formada por zeros. Os passos de tamanho α_k podem ser dados nas direções pertencentes a P^k. Notemos que o conjunto de direções dado pela matriz P^k possui um subconjunto que gera positivamente o $\mathbb{R}^n$, formado pela submatriz $B\Gamma^k$. A atualização do passo depende do status da iteração corrente: em caso de sucesso, o passo pode ser mantido ou aumentado. Já em caso de fracasso, ele necessariamente é reduzido. Em linhas gerais, o algoritmo toma a seguinte forma:

Algoritmo 4.2. *Algoritmo de Busca Padrão*

dados $x^0 \in \mathbb{R}^n$, $\alpha_0 \in \mathbb{R}$, $\omega_0, \omega_1, \ldots, \omega_L \in \mathbb{Z}$, $\omega_0 < 0$, $\omega_1, \ldots, \omega_L \geq 0$, τ racional, $\tau > 1$, para $k = 0, 1, \ldots$

Passo 1: Se $f(x^k + \alpha_k d^k) < f(x^k)$ para algum $d^k \in B\Gamma^k$, faça $x^{k+1} = x^k + \alpha_k p^k$ para algum $p^k \in P^k$ tal que $f(x^k + \alpha_k p^k) < f(x^k)$. Se for possível encontrar tal p^k, declare sucesso. Caso contrário, declare fracasso e faça $x^{k+1} = x^k$.

Passo 2: Se houve sucesso, faça $\alpha_{k+1} = \tau^{\omega_i}\alpha_k$, para algum $1 \leq i \leq L$. Caso contrário, faça $\alpha_{k+1} = \tau^{\omega_0}\alpha_k$.

O Passo 1 nos diz que, sempre que existe uma direção $d^k \in B\Gamma^k$ onde há descida com passo α_k, somos obrigados a modificar x^k. O novo iterando deve ser sempre melhor que o anterior, no sentido do decrésimo simples. No Passo 2 temos que, em caso de fracasso, o passo é reduzido por um fator τ^{ω_0}. Já em caso de sucesso, há a possibilidade de mantermos o valor do passo ou aumentá-lo, liberdade proporcionada pela escolha dos parâmetros algorítmicos $\omega_1, \ldots, \omega_L$.

A relevância teórica dos vetores representados nas matrizes BL^k é nula. Essa matriz tem apenas a função de deixar o algoritmo mais abrangente, de modo que vários outros algoritmos possam ser considerados casos particulares de Busca Padrão. De fato, tais direções não precisam ser testadas, ou podemos tomar sempre $L^k = 0$. Já as direções $B\Gamma^k$, que geram positivamente o $\mathbb{R}^n$, devem ser sempre testadas antes que o fracasso seja declarado, e são fundamentais nas demostrações de convergência.

A primeira propriedade teórica demonstrada em [34] é a de que toda sequência $\{x^k\}$ gerada pelo Algoritmo de Busca Padrão possui uma subsequência que converge a um ponto estacionário. Esse tipo de convergência, para subsequências, é muito utilizado em otimização. Dada a estrutura do algoritmo em questão, de busca monótona, com iterandos possivelmente cada vez mais próximos entre si (já que α_k potencialmente está indo para zero), é improvável que a sequência gerada fique "pulando"indefinidamente, possuindo diversos pontos de acumulação para suas diferentes subsequências. Mas o que é possível garantir do ponto de vista teórico é que, caso isso ocorra, ao menos um dos pontos de acumulação é estacionário. Uma hipótese que deve ser cumprida para que consigamos provar a convergência é a de que o conjunto de nível $L(x^0) = \{x \in \mathbb{R}^n \text{ tal que } f(x) \leq f(x^0)\}$ é compacto. Essa propriedade joga um papel importantíssimo, pois como a busca é monótona, $x^k \in L(x^0)$ para todo k e, como sabemos, toda sequência num compacto possui um ponto de acumulação. O resultado de convergência proposto é expresso no teorema abaixo.

Teorema 4.1. *Seja* $\{x^k\}$ *uma sequência gerada pelo Algoritmo de Busca Padrão. Suponha que* $L(x^0)$ *é compacto e* $f(x)$ *é continuamente diferenciável em uma vizinhança de* $L(x^0)$. *Então* $\liminf_{k\to\infty} \|\nabla f(x^k)\| = 0$.

A demonstração é baseada em duas propriedades do Algoritmo de Busca Padrão, propriedades essas que foram amplamente exploradas por publicações posteriores de diversos autores. A primeira é a de que os pontos visitados estão em treliças racionais, que podem ficar mais densas à medida que α_k diminui. Isso é usado em um lema auxiliar que estuda o comportamento de α_k, que deve possuir uma subsequência tendendo a zero, o que é provado utilizando o fato de que $L(x^0)$ é compacto, e sua intersecção com qualquer treliça racional possui finitos pontos. A outra propriedade é a de que as direções de busca geram positivamente o $\mathbb{R}^n$ pois contêm colunas das matrizes BM^k e $-BM^k$.

É possível também demonstrar a convergência num sentido mais forte, onde a sequência toda, e não apenas uma subsequência, converge a um ponto estacionário. Para tanto, algumas hipóteses extras são necessárias, além de alterações no algoritmo que o deixam mais exigente no momento de aceitar novos pontos. As modificações mais relevantes são as de que a sequência α_k de passos deve convergir a zero e a exigência de que o passo dado em uma iteração deve ser tal que $f(x^k + \alpha_k d^k) \leq \min\{f(x^k \pm \alpha_k y) \text{ com } y \in BM^k\}$, ou seja, uma direção no mínimo tão boa quanto a melhor entre as direções BM^k e $-BM^k$ deve sempre ser tomada. Esse requerimento é chamado *hipótese forte sobre os movimentos exploratórios.* Sob essas condições, o seguinte teorema é demonstrado em [34]:

Teorema 4.2. *Seja* $\{x^k\}$ *uma sequência gerada pelo Algoritmo de Busca Padrão. Suponha que* $L(x^0)$ *é compacto e* $f(x)$ *é continuamente diferenciável em uma vizinhança de* $L(x^0)$. *Além disso, suponha que* $\lim_{k\to\infty} \alpha_k = 0$ *e o algoritmo satisfaz a hipótese forte sobre os movimentos exploratórios. Então* $\lim_{k\to\infty} \|\nabla f(x^k)\| = 0$.

4.2 Algoritmo de Hooke e Jeeves

Além de introduzir a expressão busca direta, Hooke e Jeeves apresentaram em [16] um algoritmo que, posteriormente, foi reconhecido em [34] como de Busca Padrão. Julgamos justo fazer menção ao método por sua importância histórica, além de sua relevância prática, uma vez que é até hoje empregado na resolução de problemas provenientes de diversas áreas.

O algoritmo é baseado na Busca Coordenada, mas acrescenta uma estratégia que visa acelerar a convergência, aproveitando as informações obtidas nas iterações anteriores. Em caso de sucesso na última iteração, a itereção atual faz Busca Coordenada a partir do ponto $x^k + (x^k - x^{k-1})$. Essa mudança de ponto de referência para a Busca Coordenada é chamada *passo padrão*, e este tenta aproveitar o fato de que $x^k - x^{k-1}$ é uma direção promissora, já que potencialmente é uma direção de descida. A Figura 4.3 mostra a disposição desses pontos para melhor compreensão.

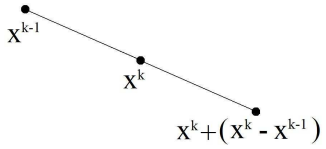

Figura 4.3: Exemplo de pontos utilizados no Algoritmo de Hooke e Jeeves.

Após uma iteração bem sucedida, se obtivermos sucesso na Busca Coordenada ao redor de $x^k + (x^k - x^{k-1})$, está proverá o novo iterando x^{k+1}. Por outro lado, se essa busca resultar em fracasso, então uma iteração de Busca Coordenada é realizada em torno de x^k. Sempre que uma iteração de Busca Coordenada ao redor de x^k falha, fazemos uma nova busca ao redor deste ponto, mas reduzindo o tamanho do passo.

Em [16], o algoritmo é introduzido de uma forma bastante descritiva e, no final do artigo, apresentado na forma de diagramas. Em [34], a autora demonstra que o algoritmo é de Busca Padrão, explicitando quais as matrizes B, M^k e L^k envolvidas. Dessa forma, todos os resultados de convergência da seção anterior valem para o algoritmo de Hooke e Jeeves.

4.3 Algoritmo de Lucidi e Sciandrone

Em 2002, Lucidi e Sciandrone propuseram um algoritmo de busca direta para minimização irrestrita [20] que possui várias semelhanças com o Algoritmo de Busca Padrão de Torczon. Nos ateremos então às diferenças entre os métodos.

No trabalho desenvolvido pelos autores, é possível usar um conjunto mais amplo de direções de busca em relação às da Busca Padrão, formado

por direções $\{p^{ik}\}$, $i = 1, \ldots, r$ que satisfazem a seguinte condição:

Condição 4.1. *As direções* $\{p^{ik}\}$, $i = 1, \ldots, r$ *são limitadas e tais que*

$$\lim_{k\to\infty} \|\nabla f(x^k)\| = 0 \Leftrightarrow \lim_{k\to\infty} \sum_{i=1}^{r} \min\{0, \nabla f(x^k)^T p^{ik}\} = 0.$$

A condição acima nos diz que as direções de busca $\{p^{ik}\}$ utilizadas são tais que se a derivada direcional em cada uma das direções está tendendo a valores não negativos (ou seja, se todas as direções estão tendendo a direções que não são de descida) isso deve implicar que $\{x^k\}$ necessariamente está tendendo a um ponto estacionário. Em outras palavras, o comportamento do gradiente de $f(x)$ nas direções p^{ik} deve nos dar uma boa noção de estacionariedade. Dois exemplos de sequências de direções que satisfazem essa condição são apresentados:

1. p^{ik} tais que as sequências $\{p^{ik}\}$, com $i = 1, \ldots, r$ são limitadas, e todo ponto limite $(\overline{p}^1, \ldots, \overline{p}^r)$ da sequência $\{(p^{ik}, \ldots, p^{rk})\}$ é tal que os vetores $\overline{p}^i$ geram positivamente o $\mathbb{R}^n$. Direções que satisfazem essas condições são fáceis de construir. Basta tomar, por exemplo,

   $$p^{ik} = e^i, \qquad i = 1, \ldots, n,$$

   e fazer

   $$p^{n+i,k} = -e^i, \qquad i = 1, \ldots, n,$$

   ou

   $$p^{n+1,k} = -\sum_{i=1}^{n} e^i.$$

2. p^{ik}, $i = 1, \ldots, n$ uniformemente linearmente independente e $p^{n+1,k}$ da forma

   $$p^{n+1,k} = \frac{x^k - x^k_{max}}{\xi_k},$$

   onde $x^k_{max} = \text{argmax}_{i=1,\ldots,n}\{f(x^k + \xi_k p^{ik})\}$ e $\xi_k \to 0$ quando $k \to \infty$.

Os autores propõem então dois algoritmos que são baseados em cada um dos exemplos de direções dados.

Algoritmo 4.3. *Algoritmo de Lucidi e Sciandrone*

Sejam $k = 0$, $x^0 \in \mathbb{R}^n$, $\widetilde{\alpha}^0_i > 0$, $i = 1, \ldots, r$, $\gamma > 0$, δ, $\theta \in (0, 1)$.

Passo 1: Faça $i = 1$ e $y^{1k} = x^k$.

Passo 2: Se $f(y^{ik} + \widetilde{\alpha}_i^k p^{ik}) \leq f(y^{ik}) - \gamma(\widetilde{\alpha}_i^k)^2$, então calcule α_i^k por Procedimento LS$(\widetilde{\alpha}_i^k, y^{ik}, p^{ik}, \gamma, \delta)$ e faça $\widetilde{\alpha}_i^{k+1} = \alpha_i^k$.
Caso contrário $\alpha_i^k = 0$ e $\widetilde{\alpha}_i^{k+1} = \theta\widetilde{\alpha}_i^k$.
Faça $y^{i+1,k} = y^{ik} + \alpha_i^k p^{ik}$.

Passo 3: Se $i < r$, faça $i = i + 1$ e volte para o Passo 2.

Passo 4: Encontre x^{k+1} tal que $f(x^{k+1}) \leq f(y^{r+1,k})$, faça $k = k + 1$ e volte para o Passo 1.

Procedimento LS$(\widetilde{\alpha}_i^k, y^{ik}, p^{ik}, \gamma, \delta)$: Encontre $\alpha_i^k = \min\{\delta^{-j}\widetilde{\alpha}_i^k : j = 0, 1, \ldots\}$ tal que

$$\begin{aligned} f(y^{ik} + \alpha_i^k p^{ik}) &\leq f(y^{ik}) - \gamma(\alpha_i^k)^2, \\ f\left(y^{ik} + \frac{\alpha_i^k}{\delta} p^{ik}\right) &\geq \max\left\{f(y^{ik} + \alpha_i^k p^{ik}), f(y^{ik}) - \gamma\left(\frac{\alpha_i^k}{\delta}\right)^2\right\}. \end{aligned}$$

Podemos utilizar, no Algoritmo 4.3, um conjunto de direções que gere positivamente o $\mathbb{R}^n$, da mesma forma que na Busca Padrão. As diferenças entre Busca Padrão e o algoritmo que estamos discutindo serão comentadas em breve. A cada iteração k, o comportamento da função é analisado em todas as direções de busca p^{ik}. Quando algum p^{ik} cumpre o critério de decréscimo suficiente, o procedimento de busca linear tenta dar um passo significativo nesta direção. A imposição de decréscimo suficiente permite uma liberdade maior na escolha das direções.

Para cada $i = 1, 2, \ldots, r$, foram adotadas sequências de passos $\{\alpha_i^k\}_{k=1}^{\infty}$ diferentes. Isso permite que o comportamento da função em cada um dos r conjuntos de direções $\{p^{ik}\}_{k=1}^{\infty}$ seja analisado de maneira independente, fato útil principalmente se as direções de busca são as mesmas a cada iteração $(p^{ik} = \overline{p}^i, i = 1, \ldots, r)$.

Por fim, observamos que no Passo 4 há a liberdade para o uso de qualquer esquema de extrapolação, que pode ser uma boa ideia para obtermos melhores resultados práticos. Os autores provam o seguinte resultado:

Teorema 4.3. *Seja $\{x^k\}$ uma sequência gerada pelo Algoritmo 4.3. Suponha que as sequências de direções $\{p^{1k}, p^{2k}, \ldots, p^{rk}\}$ satisfazem a Condição 4.1. Então o Algoritmo 4.3 está bem definido e*

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

O próximo algoritmo, inspirado no segundo exemplo de direções que satisfazem a Condição 4.1, tenta emular a direção de máxima descida a cada iteração através da expressão $p^{n+1,k} = (v^k_{min} - v^k_{max})/\xi_k$, onde v^k_{min} e v^k_{max} são os pontos, dentre os visitados desde o início da iteração, onde a função objetivo atinge seu menor e maior valor, respectivamente, e ξ_k é um passo convenientemente escolhido.

Algoritmo 4.4. *Algoritmo de Lucidi e Sciandrone com aproximação para a direção de máxima descida*

Sejam $k = 0$, $x^0 \in I\!R^n$, $c > 0$, $\widetilde{\alpha}^0_i > 0$, $i = 1, \ldots, n+1$, $\gamma > 0$, δ, $\theta \in (0,1)$.

Passo 1: Faça $i = 1$ e $y^{1k} = x^k$, $V^k = \{y^{1k}\}$, $S^k = \{\emptyset\}$.

Passo 2: Se $f(y^{ik} + \widetilde{\alpha}^k_i p^{ik}) \leq f(y^{ik}) - \gamma(\widetilde{\alpha}^k_i)^2$, então calcule α^k_i por Procedimento LS($\widetilde{\alpha}^k_i, y^{ik}, p^{ik}, \gamma, \delta$) e faça $\widetilde{\alpha}^{k+1}_i = \alpha^k_i$, $V^k = V^k \cup \{y^{ik} + \alpha^k_i p^{ik}\}$, $S^k = S^k \cup \{\alpha^k_i\}$.
Caso contrário $\alpha^k_i = 0$ e $\widetilde{\alpha}^{k+1}_i = \theta\widetilde{\alpha}^k_i$, $V^k = V^k \cup \{y^{ik} + \widetilde{\alpha}^k_i p^{ik}\}$, $S^k = S^k \cup \{\widetilde{\alpha}^k_i\}$.
Faça $y^{i+1,k} = y^{ik} + \alpha^k_i p^{ik}$.

Passo 3: Se $i < n$, faça $i = i + 1$ e volte para o Passo 2.

Passo 4: Calcule $\alpha^k_{min} = \min_{\alpha \in S^k}\{\alpha\}$ e $\alpha^k_{max} = \max_{\alpha \in S^k}\{\alpha\}$. Se $\dfrac{\alpha^k_{max}}{\alpha^k_{min}} \leq c$, então calcule $p^{n+1,k}$ tal que

$$p^{n+1,k} = \frac{v^k_{min} - v^k_{max}}{\xi_k},$$

onde $v^k_{max} = \arg\max_{v \in V^k}\{f(v)\}$, $v^k_{min} = \arg\min_{v \in V^k}\{f(v)\}$, e $\xi_k \in [\alpha^k_{min}, \alpha^k_{max}]$. Caso contrário, faça

$$p^{n+1,k} = -\sum_{i=1}^{n} p^{ik}.$$

Passo 5: Se $f(y^{nk} + \widetilde{\alpha}^k_{n+1} p^{n+1,k}) \leq f(y^{nk}) - \gamma(\widetilde{\alpha}^k_{n+1})^2$, então calcule α^k_{n+1} por Procedimento LS($\widetilde{\alpha}^k_{n+1}, y^{nk}, p^{n+1,k}, \gamma, \delta$) e faça $\widetilde{\alpha}^{k+1}_{n+1} = \alpha^k_{n+1}$.
Caso contrário $\alpha^k_{n+1} = 0$ e $\widetilde{\alpha}^{k+1}_{n+1} = \theta\widetilde{\alpha}^k_{n+1}$.
Faça $y^{n+1,k} = y^{nk} + \alpha^k_{n+1} p^{n+1,k}$.

Passo 6: Encontre x^{k+1} tal que $f(x^{k+1}) \leq f(y^k_{n+1})$, faça $k = k + 1$, e volte para o Passo 1.

Para o Algoritmo 4.4, os autores provam em [20] o seguinte resultado de convergência:

Teorema 4.4. *Seja $\{x^k\}$ uma sequência gerada pelo Algoritmo 4.4. Suponha que os vetores $\{p^{ik}\}$, com $i = 1, \ldots, n$, são limitados e uniformemente linearmente independentes. Então o Algoritmo 4.4 está bem definido e temos*

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

Devemos ressaltar duas diferenças entre o algoritmo de Lucidi e Sciandrone e a Busca Padrão de Torczon. A primeira é que direções promissoras são exploradas ao máximo, através do Procedimento LS, que realiza uma busca linear. Isso significa que, se há decréscimo em uma direção, o algoritmo tentará andar o máximo possível nesta. A segunda diferença, mais importante, é que decréscimo suficiente é exigido, em detrimento ao decréscimo simples da Busca Padrão. Um novo ponto só será aceito se satisfizer a relação $f(x^k + \alpha^k_i p^{ik}) \leq f(x^k) - \gamma(\alpha^k_i)^2$, para um $\gamma > 0$ escolhido no início do algoritmo. Os passos são expressos através do vetor α^k, uma vez que são permitidos passos diferentes para cada direção, sendo esses atualizados de maneira independente.

O fato de decréscimo suficiente ser utilizado muda a maneira como as demonstrações podem ser feitas. Não estamos mais presos à exigência de que os iterandos permaneçam sobre uma treliça racional. Em trabalhos posteriores, relacionados à minimização com restrições, os autores envolvidos com a Busca Padrão passaram a dar a opção de versões de seus algoritmos que usam decréscimo suficiente, o que aumenta a liberdade na escolha das direções de busca e parâmetros algorítmicos.

Os autores não apresentam experimentos numéricos em [20]. Em [25], são realizados experimentos numéricos que testam a eficiência do Algoritmo de Lucidi e Sciandrone e do Algoritmo de Nelder-Mead. A conclusão é que Nelder-Mead tem bom desempenho (superior ao Algoritmo de Lucidi e Sciandrone) apenas para problemas de dimensões pequenas, pois para n grande ele performa uma quantidade intolerável de avaliações de funções.

4.4 Algoritmo MADS

Os métodos que utilizam um conjunto finito de direções de busca jamais serão adequados a problemas não suaves. Se a derivada é descontínua em algum ponto, não há garantias de que existe ao menos uma direção de descida mesmo que o conjunto de direções de busca gerem positivamente o $\mathbb{R}^n$. Um conjunto pequeno de direções de busca pode trazer dificuldades mesmo em problemas suaves, por exemplo, através de uma lentidão intolerável na convergência. Para tentar resolver essas questões, Audet e Dennis desenvolveram um método semelhante conceitualmente à Busca Padrão, mas que utiliza um número infinito de direções de busca, chamado MADS (*Mesh Adaptive Direct Search*) [5]. O artigo aborda não apenas problemas irrestritos, mas qualquer tipo de restrições, impondo que $f(x) = \infty$ se x é infactível, o que é chamado de *barreira extrema.* De fato, para estendermos os resultados apresentados para o contexto de minimização irrestrita, basta fazermos $\Omega = \mathbb{R}^n$, onde Ω é o conjunto de pontos factíveis, como denotado no artigo. Toda a teoria de convergência é baseada no cálculo introduzido por Clarke para funções não suaves [6], de modo que o único requerimento teórico para as funções envolvidas é que sejam Lipschitz. O decréscimo exigido para um novo ponto é o decréscimo simples.

Há dois passos onde um ponto factível melhor é procurado. O primeiro é opcional, e é chamado *passo de busca* (*search step*). Ele consiste em uma busca sobre a malha M_k dada por

$$M_k = \bigcup_{x \in S_k} \{x + \Delta_k^m Dz \text{ tal que } z \in \mathbb{N}^{n_D}\},$$

onde S_k é o conjunto de todos os pontos onde a função objetivo foi computada desde o início da iteração k, e D, apesar de vir de uma definição mais complexa, pode ser visto apenas como uma matriz real $n \times n_D$.

O diferencial do algoritmo está no *passo de pesquisa* (*poll step*), onde são investigados pontos pertencentes à *estrutura* (*frame*) P_k definida por

$$P_k = \{x^k + \Delta_k^m d \text{ tal que } d \in D_k\} \subset M_k,$$

onde o conjunto D_k gera positivamente o $\mathbb{R}^n$, suas componentes são combinações inteiras não negativas das direções em D, a distância dos pontos investigados a x^k é limitada pelo passo da estrutura, Δ_k^p, e os limites das direções d normalizadas devem gerar positivamente o $\mathbb{R}^n$. Os passos devem satisfazer a propriedade $\Delta_k^m \leq \Delta_k^p$. A cada iteração, é performado um passo

de busca, seguido por um passo de pesquisa se o primeiro falhar em melhorar a função objetivo. Similarmente ao que ocorre em Busca Padrão, o passo da malha Δ_k^m pode aumentar se um ponto melhor for encontrado, e deve ser reduzido em caso contrário, sempre multiplicando-o por escalares pertencentes a um conjunto finito convenientemente escolhido. As Figuras 4.4 e 4.5 mostram exemplos de estruturas no plano. Os pontos p^1, p^2 e p^3 são escolhas possíveis para formarem a estrutura, através das direções $p^i - x^k$, $i = 1, 2, 3$. As linhas escuras representam a caixa onde os pontos da estrutura devem estar contidos, definida pelo valor Δ_k^p.

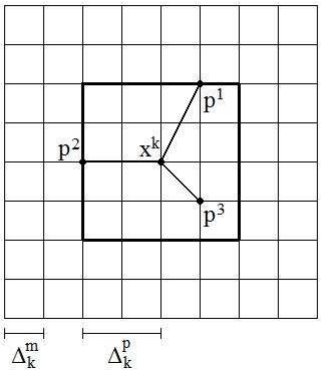

Figura 4.4: Exemplo de estrutura para $\Delta_k^m = \frac{1}{4}$ e $\Delta_k^p = \frac{1}{2}$.

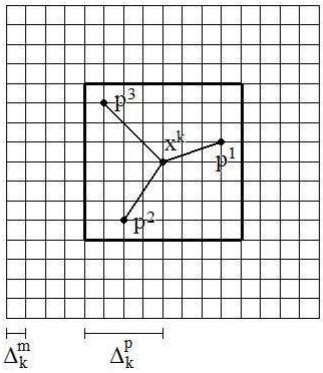

Figura 4.5: Exemplo de estrutura para $\Delta_k^m = \frac{1}{8}$ e $\Delta_k^p = \frac{1}{2}$.

Um ponto x^k é chamado *centro mínimo da estrutura* (*minimal frame center*) se $\Delta_{k+1}^m < \Delta_k^m$. Sendo K um subconjunto dos índices dos centros mínimos da estrutura, $\{x^k\}_{k\in K}$ é chamada *subsequência de refinamento* (*refining subsequence*) se $\{\Delta_k^m\}_{k\in K}$ converge a zero. Chamemos seu limite de $\widehat{x}$. Se d^k é uma subsequência de direções de pesquisa e se existe um subconjunto $L \subseteq K$ tal que o limite de $\{d^k/\|d^k\|\}_{k\in L}$ existe e $x^k + \Delta_k^m d^k$ é factível para infinitos $k \in L$, então esse limite é chamado *direção de refinamento para* $\widehat{x}$. Os autores já haviam demonstrado em artigo prévio que há ao menos uma subsequência de refinamento. Mas no trabalho que estamos agora discutindo, provam que se o conjunto de direções de refinamento para $\widehat{x}$ é denso no cone hipertangente, então $\widehat{x}$ é um ponto estacionário de Clarke para o problema.

A grosso modo, omitindo os detalhes, o algoritmo pode ser assim colocado:

Algoritmo 4.5. *Algoritmo MADS*

Seja $x^0 \in I\!R^n$, $\Delta_0^m \leq \Delta_0^p$ e parâmetros algorítmicos. Faça $k = 0$.

Passo 1: Performe o passo de busca e possivelmente o passo de pesquisa até que um novo ponto seja encontrado sobre a malha M_k.

- **Busca opcional:** calcule f em um subconjunto finito de pontos sobre a malha M_k.
- **Pesquisa local:** calcule f sobre a estrutura P_k.

Passo 2: Atualize convenientemente Δ_{k+1}^m e Δ_{k+1}^p, faça $k = k + 1$ e volte para o Passo 1.

Os teoremas envolvendo o algoritmo acima não serão citados, pois para entendê-los é necessário um estundo de cones tangentes, derivadas de Clarke, entre outros assuntos que fogem do escopo do texto. Para o leitor interessado, recomendamos a leitura do artigo [5], além das referências nele contidas, em particular as relacionadas ao Cálculo para funções não contínuas. No caso diferenciável, um corolário do teorema principal nos diz que o limite de uma sequência de refinamento é um ponto estacionário de Clarke.

A proposta dos autores para cumprir a exigência de que as direções sejam densas é baseada em direções aleatórias. O processo de geração dessas direções tem como ponto de partida a criação de uma matriz triangular inferior com componentes inteiras aleatórias, que depois de alguns cálculos trará as direções desejadas. O algoritmo MADS com a escolha de direções apresentada é chamado LTMADS. Como é baseado em componentes randômicas, o resultado teórico é que as direções de pesquisa são assintoticamente densas no cone hipertangente com probabilidade 1. Uma maneira determinística de se computar direções densas foi recentemente divulgada em [2], baseada em sequências quase randômicas. O algoritmo obtido com essa escolha de direções recebe o nome de ORTHOMADS. Os algoritmos LTMADS e ORTHOMADS fazem parte do software para otimização NOMAD (*Nonsmooth Optimization by Mesh Adaptive Direct Search*) [39].

4.5 Algoritmo de Direções Aleatórias

O Algoritmo de Direções Aleatórias foi proposto em 2008 em um trabalho de Diniz-Ehrhardt, Martínez e Raydan [13]. Os algoritmos apresentados

possuem algumas diferenças em relação aos das seções anteriores, no entanto decidimos mantê-lo neste capítulo pois o espírito de busca direcional permanece. Baseados em algumas técnicas *derivative-free* da literatura, os autores construíram um método globalmente convergente com estratégia de busca linear não monótona. Esse é o grande diferencial do trabalho, por permitir acréscimo na função objetivo, que eventualmente pode utilizar inclusive direções de subida.

Inicialmente, escolhemos o parâmetro $M \in \mathbb{N}$. Em uma iteração k, definimos

$$\overline{f}_k = \max\{f(x^k), \ldots, f(x^{\max\{k-M+1,0\}})\}.$$

Se a inequação

$$f(x^k + \alpha d) \leq \overline{f}^k + \eta_k - \alpha^2 \beta_k. \tag{4.5.1}$$

for satisfeita, tomamos $x^{k+1} = x^k + \alpha d$. O parâmetro η_k deve ser escolhido de maneira que

$$\eta_k > 0 \ \ \forall \ k \in \mathbb{N} \qquad \text{e} \qquad \sum_{k=0}^{\infty} \eta_k < \infty. \tag{4.5.2}$$

Já o parâmetro β_k deve ser tal que $\{\beta_k\}$ seja uma sequência limitada, ou seja, exista $C \in \mathbb{R}$ tal que

$$\beta_k < C \ \ \forall \ k \in \mathbb{N}.$$

Além disso é necessário que

$$\beta_k > 0 \ \ \forall \ k \in \mathbb{N}$$

e, para qualquer subconjunto infinito de índices $K \subset \mathbb{N}$,

$$\lim_{k \in K} \beta_k = 0 \Rightarrow \lim_{k \in K} \nabla f(x^k) = 0. \tag{4.5.3}$$

Observemos que a escolha $\beta_k \equiv 1$ é admissível.

Com o critério (4.5.1), eventualmente serão admitidos pontos que representem um acréscimo na função objetivo. Essa característica se deve ao parâmetro η_k e ao fato de utilizarmos $\overline{f}^k$ ao invés de $f(x^k)$. Já β_k tem um papel semelhante ao escalar γ do algoritmo de Lucidi e Sciandrone.

A inequação (4.5.1) é a base para um modelo de algoritmo globalmente convergente, e para mostrar a eficiência de se utilizar um critério não monótono, os autores propõem um algoritmo onde as direções de busca são geradas aleatoriamente.

Algoritmo 4.6. *Algoritmo de direções aleatórias*

Sejam $\{\beta_k\}$ e $\{\eta_k\}$ satisfazendo as relações (4.5.2) - (4.5.3). Consideremos também τ_{min} e τ_{max} tais que $0 < \tau_{min} < \tau_{max} < 1$, $M \in \mathbb{N}$, $0 < \Delta_{min} < \Delta_{max} < \infty$ e $c_{max} \in \mathbb{R}^+$. Os passos para determinar x^{k+1} a partir de x^k são os seguintes:

Passo 1: Calcule uma direção aleatória $d \in \mathbb{R}^n$ tal que $\Delta_{min} \leq \|d\| \leq \Delta_{max}$.

Passo 2: Se $f(x^k + d) \leq f(x^k) + \eta_k - \beta_k$, faça $\alpha_k = 1, d^k = d$ e vá para o Passo 5.
Se $f(x^k - d) \leq f(x^k) + \eta_k - \beta_k$, faça $\alpha_k = 1, d^k = -d$ e vá para o Passo 5.

Passo 3: *Interpolação Quadrática.* Calcule $\widetilde{\alpha}$, o minimizador da parábola que interpola os pontos $(-1, f(x^k + d))$, $(0, f(x^k))$ e $(1, f(x^k + d))$.

1. Se $\widetilde{\alpha}$ existe e pertence a $[\tau_{min}, \tau_{max}]$, faça $d^k = d$ e vá para o Passo 4.
2. Se $\widetilde{\alpha}$ existe e pertence a $[-\tau_{min}, -\tau_{max}]$, faça $d^k = -d$ e vá para o Passo 4.
3. Se $f(x^k + d) \leq f(x^k - d)$, faça $d^k = d$, $\widetilde{\alpha} = 1/2$ e vá para o Passo 4.
4. Se $f(x^k + d) > f(x^k - d)$, faça $d^k = -d$, $\widetilde{\alpha} = 1/2$ e vá para o Passo 4.

Passo 4: *Backtracking.* Faça $\alpha = \widetilde{\alpha}$. Se (4.5.1) for satisfeita, faça $\alpha_k = \alpha$, $x^{k+1} = x^k + \alpha_k d^k$ e termine a iteração. Caso contrário, calcule $\alpha_{new} \in [\tau_{min}\alpha, \tau_{max}\alpha]$ usando interpolação quadrática com salvaguarda, faça $\alpha = \alpha_{new}$ e repita o teste (4.5.1).

Passo 5: *Extrapolação.*

1. Faça $c = 1$.
2. Se $2c > c_{max}$, faça $x^{k+1} = x^k + c\alpha_k d^k$ e termine a iteração.
3. Se $f(x^k + 2c\alpha_k d^k) > f(x^k + c\alpha_k d^k)$, faça $x^{k+1} = x^k + c\alpha_k d^k$ e termine a iteração.
4. Faça $c = 2c$ e vá para o Passo 2 da extrapolação.

Mesmo utilizando direções aleatoriamente geradas, o desempenho do algoritmo é satisfatório, pois este apresenta bons resultados na grande maioria

dos testes realizados. Observamos que o critério (4.5.1) de decréscimo suficiente apresenta algumas semelhanças com o utilizado por Lucidi e Sciandrone [20].

Os autores provam os seguintes resultados de convergência.

Teorema 4.5. *Assuma que $f(x^k)$ possui derivadas parciais contínuas. Seja uma sequência $\{x^k\}$ gerada pelo Algoritmo 4.6 tal que $\{f(x^k)\}$ seja limitada inferiormente. Se (x^*, d) é ponto limite da sequência $\{(x^k, d^k)\}$, então $\nabla f(x^*)^T d \geq 0$.*

Teorema 4.6. *Assuma que $f(x^k)$ possui derivadas parciais contínuas e que o conjunto $\{x \in I\!R^n$ tal que $f(x) \leq f(x^0) + \sum_{k=0}^{\infty} \eta_k\}$ é limitado. Suponha que, a cada iteração, uma direção d^k é gerada aleatoriamente de forma que:*

- *$d^0, d^1, d^2, \ldots$ são variáveis aleatórias independentes n-dimensionais;*
- *existem $\theta, p \in (0,1)$, $0 < \Delta_{\min} < \Delta_{\max} < \infty$ tais que, para todo $k \in I\!N$, a probabilidade de que*

$$\|d^k\| \in [\Delta_{\min}, \Delta_{\max}] \;\; e \;\; \nabla f(x^*)^T d \leq -\theta \|\nabla f(x^k)\| \|d^k\|$$

  é maior do que p.

Então, dado $\varepsilon > 0$, com probabilidade 1 existe um $k \in I\!N$ tal que $\|\nabla f(x^k)\| \leq \varepsilon$.

4.6 Exercícios

1. Implemente o Algoritmo de Busca Coordenada.

2. Faça desenhos em $\mathbb{R}^2$ mostrando como ficaria a malha da Busca Padrão para escolhas das direções diferentes das direções coordenadas.

3. Cite um critério de parada adequado para o algoritmo de Busca Padrão.

4. Construa um exemplo de Busca Padrão onde as direções são fixas a cada iteração, mas a matriz L^k possui colunas não nulas. Qual a importância dessa matriz?

5. Considere o Passo 4 do Algoritmo 4.4. Prove que se as direções $p^{1k}, p^{2k}, \ldots, p^{nk}$ são linearmente independentes e $\alpha^k_{\max}/\alpha^k_{\min} > c$, então as direções da iteração corrente geram positivamente o $\mathbb{R}^n$.

6. No Algoritmo de Direções Aleatórias, mesmo sendo contra a definição de η_k, suponha que $\eta_k \equiv 0$. Mostre que ainda assim a condição 4.5.1 pode ser satisfeita para pontos com valor de f maiores do que o corrente.

7. Quais as principais diferenças entre o Algoritmo de Busca Padrão e o Algoritmo MADS?

Capítulo 5

Métodos que Usam Interpolação Polinomial

5.1 Introdução

Neste capítulo veremos métodos para construir modelos de aproximação de problemas não lineares de otimização, via interpolação polinomial de valores da função objetivo, f. Nestes métodos, o vetor de aproximação da solução é gerado em uma região de confiança $S \subset I\!R^n$. O objetivo básico é aproximar a função

$$f : I\!R^n \to I\!R$$

por uma função quadrática Q, que será minimizada a cada iteração.

Embora neste curso estejamos trabalhando apenas com métodos que não fazem uso de derivadas, a existência das mesmas tem grande importância teórica na escolha das aproximações a serem feitas. Em geral os teoremas de convergência têm como hipótese um número de derivadas contínuas da função objetivo f.

Os polinômios usados na aproximação de f serão polinômios quadráticos que interpolam f num conjunto de pontos $Y = \{y^0, y^1, \ldots, y^m\}$ em $I\!R^n$, ou seja $Q(y^j) = f(y^j)$, $j = 1, \ldots, m$. Tais polinômios fornecem uma maneira de estimar a primeira e a segunda derivadas da função objetivo. Justificamos essa escolha também pelo fato de que polinômios de grau 2 são simples de serem construídos, e geram métodos que possuem melhores taxas de convergência (local) que os polinômios lineares, que é a opção polinomial mais

simples de todas. Vale a pena observar, ainda, que os modelos lineares não aproveitam a curvatura de funções não lineares, e portanto não se prestam para aproximar derivadas segundas de f.

Obviamente os polinômios usados não são a única opção não linear para aproximar funções não lineares. Podemos citar como exemplo as funções radiais, que cumprem bem a tarefa de modelar a curvatura da função em questão (ver [9]).

Nosso objetivo básico é sugerir como aproximação da função objetivo modelos para os quais possamos estabelecer uma fundamentação teórica para algoritmos sem derivadas, sempre visando teoremas de convergência global para estes. Para tal, o modelo quadrático Q precisa ter algumas características importantes.

Se usássemos, por exemplo, o truncamento da série de Taylor para construir o polinômio interpolador - aqui o sentido de interpolação não é o mesmo daquele ao qual nos referimos - contaríamos com um limitante superior para o erro cometido, herdado naturalmente da própria série.

A construção dos polinômios interpoladores depende de vários parâmetros, e chamamos a atenção para a escolha dos pontos de interpolação, que possui importância crucial nesse contexto. Tais pontos devem manter a qualidade do modelo a cada iteração, o que é garantido pelas várias constantes das quais depende. Além de querermos que os valores de f em x^k, fornecidos pelo algoritmo que vamos utilizar, se aproximem cada vez mais da solução do problema de otimização, esperamos que esses valores forneçam também uma boa aproximação para a atualização do modelo quadrático Q. Devemos estudar cuidadosamente a relação entre as constantes envolvidas, pois elas desempenham um papel fundamental na medida do erro, no caso, na norma de Frobenius. Por uma *boa escolha* de pontos numa interpolação, podemos estar até querendo dizer a troca de um ponto que estava sendo usado por um outro que seja mais adequado, ou até mesmo a mudança do número de pontos de interpolação a serem utilizados.

Os métodos aos quais nos referiremos neste capítulo são de autoria de Powell, [9], [31], [29] [30], e são métodos para minimização irrestrita que não fazem uso de derivadas. No entanto, podemos citar vários outros autores que trabalham com o mesmo tipo de métodos, como (ver [9] e [8]). Na realidade, nos focaremos aqui em dois métodos de Powell: o método UOBYQA (ver [31]), introduzido em 2002, e posteriormente o método NEWUOA (ver [30]),

proposto em 2004, onde Powell desenvolveu um novo algoritmo, análogo ao primeiro, mas que necessita de menos esforço computacional.

Em ambos os casos, os algoritmos consistem, basicamente, em:

- construir uma aproximação quadrática Q $Q(x)$ para a função objetivo f, para gerar um novo vetor de aproximação da solução e,
- resolver o seguinte subproblema de região de confiança:

$$\text{minimizar } Q(x^k + s) \text{ sujeito a } s \in \Omega,$$

  onde k representa a iteração atual. O conjunto $\Omega = \{s \in I\!R^n : \|s\| \leq \Delta\}$ é a região de confiança de raio Δ na qual iremos trabalhar.
- Maximizar a $j-$ésima função de Lagrange dentro de uma região de confiança N.

A seguir descreveremos alguns conceitos fundamentais sobre polinômios de grau d em $I\!R^n$, tratando com mais detalhes as bases desse espaço, em particular os polinômios interpoladores de Lagrange. Também teceremos comentários sobre os métodos de Powell, apresentando, no final do capítulo, alguns resultados numéricos.

5.2 Alguns Conceitos Fundamentais e os Polinômios Interpoladores de Lagrange

5.2.1 Polinômios de Grau d em $I\!R^n$

Usaremos a mesma notação do livro de Conn, Scheinberg e Vicente [9], chamando de P_n^d o subespaço dos polinômios de grau $\leq d$ em $I\!R^n$ e de p_1, sua dimensão. Se iniciarmos a indicação dos vetores de uma base qualquer desses subespaços com o índice 0, é fácil ver que a dimensão do subespaço dos polinômios de grau $d \leq 1$ em $I\!R^n$ é $p_1 = n + 1$.

Consideremos, como exemplo, $n = 3$. Podemos escrever um polinômio qualquer de grau ≤ 1 em $I\!R^3$, $p(x)$, da seguinte forma:

$$p(x) = a_0 + a_1x_1 + a_2x_2 + a_3x_3,$$

ou seja, temos quatro polinômios linearmente independentes, dos quais $p(x)$ é combinação linear, o que mostra claramente que a dimensão de P_3^1 é $p_1 = 4 = n + 1$, donde

$$B = \{1,\, x_1,\, x_2,\, x_3\}$$

é uma base de P_3^1.

Ainda em $\mathbb{R}^3$, seja agora $d = 2$. Da mesma forma que anteriormente, podemos escrever qualquer polinômio de grau ≤ 2 em $\mathbb{R}^3$, da seguinte forma:

$$p(x) = a_0+a_1x_1+a_2x_2+a_3x_3+a_4x_1^2+a_5x_2^2+a_6x_3^2+a_7x_1x_2+a_8x_1x_3+a_9x_2x_3,$$

ou seja, temos 10 polinômios linearmente independentes dos quais uma combinação linear forma qualquer polinômio de grau 2 em $\mathbb{R}^3$, o que mostra que a dimensão do subespaços de polinômios de grau ≤ 2 em $\mathbb{R}^3$ é 10. Assim,

$$B_1 = \{1,\, x_1,\, x_2,\, x_3,\, x_1^2,\, x_2^2,\, x_3^2,\, x_1x_2,\, x_1x_3,\, x_2x_3\}$$

é uma base de P_3^2.

Uma outra base muito usada em P_3^2 provém do desenvolvimento de $f(x)$ pela fórmula de Taylor em torno de um ponto $x_0 \in \mathbb{R}^n$. Vejamos o caso de P_3^2: para qualquer x numa vizinhança de x_0,

$$f(x) = f(x_0) + \nabla f(x_0)^T(x - x_0) + \frac{1}{2}(x - x_0)^T H(x_0)(x - x_0), \qquad (5.2.1)$$

onde $H(x_0)$ é a matriz hessiana de f em x_0. Para qualquer x em $\mathbb{R}^3$,

$$\nabla f(x) = \left[\frac{\partial f}{\partial x_1}(x),\, \frac{\partial f}{\partial x_2}(x),\, \frac{\partial f}{\partial x_3}(x)\right]^T.$$

Usando o fato de que a matriz hessiana de $f(x)$ é simétrica,

$$H(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_1 x_2}(x) & \frac{\partial^2 f}{\partial x_1 x_3}(x) \\ \frac{\partial^2 f}{\partial x_1 x_2}(x) & \frac{\partial^2 f}{\partial x_2^2}(x) & \frac{\partial^2 f}{\partial x_2 x_3}(x) \\ \frac{\partial^2 f}{\partial x_1 x_3}(x) & \frac{\partial^2 f}{\partial x_2 x_3}(x) & \frac{\partial^2 f}{\partial x_3^2}(x) \end{bmatrix}.$$

Assim, chamando $z = (x - x_0)$, pela equação (5.2.1),

$$\begin{aligned} f(x) \;=\; & f(x_0) + \frac{\partial f}{\partial x_1}(x_0)z_1 + \frac{\partial f}{\partial x_2}(x_0)z_2 + \frac{\partial f}{\partial x_3}(x_0)z_3 \\ & + \frac{\partial^2 f}{\partial x_1^2}(x_0)\frac{z_1^2}{2} + \frac{\partial^2 f}{\partial x_1 x_2}(x_0)z_1z_2 + \frac{\partial^2 f}{\partial x_1 x_3}(x_0)z_1z_3 \\ & + \frac{\partial^2 f}{\partial x_2^2}(x_0)\frac{z_2^2}{2} + \frac{\partial^2 f}{\partial x_2 x_3}(x_0)z_2z_3 + \frac{\partial^2 f}{\partial x_3^2}(x_0)\frac{z_3^2}{2}, \end{aligned}$$

o que nos fornece a base

$$B_2 = \left\{1,\, z_1,\, z_2,\, z_3,\, \frac{z_1^2}{2},\, \frac{z_2^2}{2},\, \frac{z_3^2}{2},\, z_1z_2,\, z_1z_3,\, z_2z_3\right\}$$

para P_3^2.

Vimos até agora dois exemplos de base em P_3^2. No entanto, neste trabalho estamos interessados numa terceira base, chamada base de Lagrange, que será vista em breve.

5.2.2 Polinômios Interpoladores

Os polinômios que constituem a base de Lagrange têm uma característica especial, que é a de serem polinômios interpoladores da função f em alguns pontos.

Definição 5.1. *Dizemos que o polinômio $p(x)$ interpola f em $z \in \mathbb{R}^n$ se $p(z) = f(z)$.*

Consideremos o conjunto $Y = \{y^0,\, y^1,\, \ldots,\, y^p\}$ de $p+1$ pontos (vetores) em $\mathbb{R}^n$ e uma base

$$\phi = \{\phi^0(x),\, \phi^1(x),\, \ldots,\, \phi^p(x)\}$$

de P_n^d, subespaço dos polinômios de grau $\leq d$ em $\mathbb{R}^n$. Então, qualquer polinômio de grau d em $\mathbb{R}^n$ pode ser escrito como:

$$p(x) = \sum_{j=0}^{p} \alpha_j \phi^j(x).$$

Impondo as condições de interpolação em $y^0,\, y^1,\, \ldots,\, y^p$, ou seja, exigindo que

$$p(y^i) = \sum_{j=0}^{p} \alpha_j \phi^j(y^i) = f(y^i), \qquad i = 0, 1, \ldots, p, \tag{5.2.2}$$

obtemos um sistema de equações lineares, cuja solução, que sempre esperamos existir e ser única (o que nem sempre acontece), fornece os coeficientes de $p(x)$.

Abrindo as equações (5.2.2) em forma de sistema, percebemos que temos que resolver $M(\phi, Y)\,\alpha_\phi = f(Y)$, onde

$$M(\phi, Y) = \begin{pmatrix} \phi_0(y^0) & \phi_1(y^0) & \dots & \phi_p(y^0) \\ \phi_0(y^1) & \phi_1(y^1) & \dots & \phi_p(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_0(y^p) & \phi_1(y^p) & \dots & \phi_p(y^p) \end{pmatrix},$$

$$f(Y) = (f(y^0),\, f(y^1),\, \dots,\, f(y^p))^T \qquad \text{e} \qquad \alpha_\phi = (\alpha_0,\, \alpha_1,\, \dots,\, \alpha_p)^T.$$

Vamos exemplificar o que acabamos de dizer, considerando $n = d = 2$, ou seja, P_2^2. Qualquer polinômio $p(v) \in P_2^2$ pode ser escrito como combinação linear da base

$$B_1 = \{(1,\, v_1,\, v_2,\, v_1^2,\, v_2^2,\, v_1 v_2)\},$$

como vimos anteriormente. Como a dimensão de P_2^2 é 6, para construirmos um polinômio de grau $\leq$ 2 em $\mathbb{R}^2$ que interpole f nos pontos y^0, y^1, y^2, y^3, y^4 e y^5 trabalhando com a base B_1, temos que resolver o sistema linear $M\alpha = \overline{f}$, onde:

$$\alpha = (\alpha_0,\, \alpha_1,\, \alpha_2,\, \alpha_3,\, \alpha_4,\, \alpha_5,)^T \text{ e}$$
$$\overline{f} = (f(y^0),\, f(y^1),\, f(y^2),\, f(y^3),\, f(y^4),\, f(y^5))^T.$$

Notemos que $y^i = (y_1^i,\, y_2^i)^T,\ 0 \leq i \leq 5$, e que a matriz M do sistema é:

$$M(B_1, Y) = \begin{pmatrix} 1 & y_1^0 & y_2^0 & (y_1^0)^2 & (y_2^0)^2 & y_1^0 y_2^0 \\ 1 & y_1^1 & y_2^1 & (y_1^1)^2 & (y_2^1)^2 & y_1^1 y_2^1 \\ 1 & y_1^2 & y_2^2 & (y_1^2)^2 & (y_2^2)^2 & y_1^2 y_2^2 \\ 1 & y_1^3 & y_2^3 & (y_1^3)^2 & (y_2^3)^2 & y_1^3 y_2^3 \\ 1 & y_1^4 & y_2^4 & (y_1^4)^2 & (y_2^4)^2 & y_1^4 y_2^4 \\ 1 & y_1^5 & y_2^5 & (y_1^5)^2 & (y_2^5)^2 & y_1^5 y_2^5 \end{pmatrix}.$$

Teremos unicidade de solução do sistema $M\alpha = \overline{F}$ se, e somente se, a matriz $M(B_1, Y)$ for não singular.

Definição 5.2. *Dizemos que o conjunto $Y = \{y^0,\, y^1,\, \dots,\, y^p\}$ é posicionado para interpolação polinomial em $\mathbb{R}^n$ se a matriz $M(\phi, Y)$ for não singular para a base ϕ de P_n^d.*

Como todas as bases em espaços vetoriais de dimensão finita são equivalentes, a definição acima vale se $M(\phi, Y)$ for não singular, qualquer que seja ϕ base de P_n^d.

Em [9] há uma demonstração para o seguinte resultado:

Lema 5.1. *Dada uma função $f : \mathbb{R}^n \to R$ e um conjunto posicionado $Y \in \mathbb{R}^n$, o polinômio interpolador de $f(x)$ em Y, $p(x)$, existe e é único.*

5.2.3 Os Polinômios Interpoladores de Lagrange

Definição 5.3. *Dado um conjunto* $Y = \{y^0, y^1, \ldots y^p\}$ *de pontos de interpolação, uma base de* $p_1 = p + 1$ *polinômios em* P_n^d, $l_j(x)$, $j = 0, 2, \ldots, p$ *é chamada de* base de polinômios de Lagrange *se*

$$\ell_j(y^i) = \delta_{ij} = \begin{cases} 1 & se \quad i = j, \\ 0 & se \quad i \neq j. \end{cases}$$

Lema 5.2. *Se o conjunto* Y *é posicionado, então a base de polinômios de Lagrange existe e é única.*

Demonstração: Ver [9].

Por outro lado, se existirem $p+1$ polinômios de Lagrange e uma base Φ em P_n^d, seja A_Φ a matriz cujas colunas são os coeficientes de cada polinômio de Lagrange em relação a essa base. Assim, $M(\Phi, Y)A_\Phi = I$, o que implica na não singularidade da matriz $M(\Phi, Y)$, donde o conjunto Y é posicionado, por definição.

Exemplifiquemos o resultado acima, no caso de $n = 1$ e $d = 2$. De acordo com o que vimos anteriormente, a dimensão de P_1^2 é 3 e uma base Φ para P_1^2 pode ser $\Phi = \{1, x, x^2\}$. Consideremos $Y = \{x_0, x_1, x_2\}$. Os 3 polinômios de Lagrange em relação a Y, são:

$$\ell_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = \frac{x^2 - (x_1 + x_2)x + x_1x_2}{d_0},$$

onde $d_0 = (x_0 - x_1)(x_0 - x_2)$ e, a primeira coluna da matriz A_Φ é, então $\left(\frac{x_1x_2}{d_0}, \frac{(-x_1 + x_2}{d_0}, \frac{1}{d_0}\right)^T$, que são os coeficientes de $\ell_0(x)$ em relação à base Φ.

Analogamente,

$$\ell_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} = \frac{x^2 - (x_0 + x_2)x + x_0x_2}{d_1},$$

onde $d_1 = (x_1 - x_0)(x_1 - x_2)$ e, a segunda coluna da matriz A_Φ é $\left(\frac{x_0x_2}{d_1}, \frac{-(x_0 + x_2)}{d_1}, \frac{1}{d_1}\right)^T$, que são os coeficientes de $\ell_1(x)$ em relação à base Φ, e

$$\ell_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} = \frac{x^2 - (x_0 + x_1)x + x_0x_1}{d_2},$$

onde $d_2 = (x_2 - x_0)(x_2 - x_1)$ e a terceira coluna da matriz A_Φ é $\left(\frac{x_0 x_1}{d_2}, \frac{-(x_0 + x_1)}{d_2}, \frac{1}{d_2}\right)^T$, que são os coeficientes de $\ell_2(x)$ em relação à base Φ.

Assim, a matriz A_Φ é dada por:

$$A_\Phi = \begin{pmatrix} \frac{x_1 x_2}{d_0} & \frac{x_0 x_2}{d_1} & \frac{x_0 x_1}{d_2} \\ \frac{-(x_1+x_2)}{d_0} & \frac{-(x_0+x_2)}{d_1} & \frac{-(x_0+x_1)}{d_2} \\ \frac{1}{d_0} & \frac{1}{d_1} & \frac{1}{d_2} \end{pmatrix}$$

e

$$M(\Phi, Y) = \begin{pmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{pmatrix}.$$

Após efetuarmos os cálculos, temos que

$$M(\Phi, Y) A_\Phi = I.$$

Para demonstrar que o conjunto de polinômios de Lagrange definidos da maneira que fizemos até agora forma uma base de P_n^d, temos que mostrar que qualquer polinômio $p(x)$ em P_n^d é uma combinação linear desses polinômios. Como estamos trabalhando com Y posicionado, já vimos que qualquer polinômio em P_n^d é univocamente determinado por seus valores em Y. O Lema a seguir afirma que, na realidade, esses valores são os coeficientes da combinação linear de $p(x)$, em relação aos polinômios de Lagrange.

Lema 5.3. *Para qualquer função $f : \mathbb{R}^n \to \mathbb{R}$ e qualquer conjunto posicionado $Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n$, o único polinômio $p(x)$ que interpola f em Y pode ser escrito como*

$$p(x) = \sum_{i=0}^{p} f(y^i) \ell_i(x),$$

onde $\{\ell_i(x),\, i = 0, \ldots, p\}$ é a base de polinômios de Lagrange para Y.

Demonstração: Ver [9].

Ilustraremos este lema com um exemplo em P_3^2, ou seja, $n = 2$ e $d = 3$. Consideremos o conjunto de pontos de interpolação

$$Y = \{(0,0)\,,\, (1,0)\,,\, (0,1)\,,\, (2,0)\,,\, (1,1)\,,\, (0,2)\},$$

que é posicionado em $\mathbb{R}^2$. Seja $p(x)$ um polinônio de grau ≤ 2 em $\mathbb{R}^2$. Tomaremos tal polinômio como

$$p(x) = a_0 + a_1x_1 + a_2x_2 + a_3x_1^2 + a_4x_2^2 + a_5x_1x_2,$$

em relação à base $\Phi = B_1$. Temos, então,

$$\begin{aligned}
p(0,0) &= a_0, \\
p(1,0) &= a_0 + a_1 + a_3, \\
p(0,1) &= a_0 + a_2 + a_4, \\
p(2,0) &= a_0 + 2a_1 + 4a_3, \\
p(1,1) &= a_0 + a_1 + a_2 + a_3 + a_4 + a_5 \quad \text{e} \\
p(0,2) &= a_0 + 2a_2 + 4a_4.
\end{aligned}$$

Assim, obtemos a matriz

$$M(\Phi, Y) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 2 & 0 & 4 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 2 & 0 & 4 & 0 \end{pmatrix}.$$

O fato de $\det M(\Phi, Y) = -4$ ser diferente de zero prova que Y é posicionado em $\mathbb{R}^2$. Resolvendo agora os 6 sistemas lineares oriundos do fato de que $\ell_i(y^j) = \delta_{ij}$, para todos os polinômios interpoladores de Lagrange, ou seja, resolvendo os 6 sistemas lneares $M(\Phi, Y)\ell = b^j,\ j = 0, 1, 2, 3, 4, 5$, onde

$$\begin{aligned}
b^0 &= (1\ 0\ 0\ 0\ 0\ 0)^T \\
b^1 &= (0\ 1\ 0\ 0\ 0\ 0)^T \\
b^2 &= (0\ 0\ 1\ 0\ 0\ 0)^T \\
b^3 &= (0\ 0\ 0\ 1\ 0\ 0)^T \\
b^4 &= (0\ 0\ 0\ 0\ 1\ 0)^T \\
b^5 &= (0\ 0\ 0\ 0\ 0\ 1)^T,
\end{aligned}$$

obtemos os polinômios interpoladores de Lagrange, em Y :

$$\begin{aligned}
\ell_0(x_1, x_2) &= 1 - 3/2x_1 - 3/2x_2 + 1/2x_1^2 + 1/2x_2^2 + x_1x_2, \\
\ell_1(x_1, x_2) &= 2x_1 - x_1^2 - x_1x_2, \\
\ell_2(x_1, x_2) &= 2x_2 - x_2^2 - x_1x_2, \\
\ell_3(x_1, x_2) &= -1/2x_1 + 1/2x_1^2, \\
\ell_4(x_1, x_2) &= x_1x_2, \\
\ell_5(x_1, x_2) &= -1/2x_2 + 1/2x_2^2.
\end{aligned}$$

Consideremos, agora, a função $f(x_1, x_2) = x_1 + x_1^2 - x_2 + 4x_1^3 - 3x_2^3$. Temos que

$$\begin{array}{lll} f(y^0) = 0, & f(y^1) = 6, & f(y^2) = -4, \\ f(y^3) = 38, & f(y^4) = 2 \quad \text{e} & f(y^5) = -26. \end{array}$$

Verifica-se trivialmente que o polinômio que interpola f em Y é dado por

$$p(x) = 0\ell_0(x) + 6\ell_1(x) - 4\ell_2(x) + 38\ell_3(x) + 2\ell_4(x) - 26\ell_5(x).$$

Para encerrar esta seção faremos algumas observações importantes sobre os polinômios de Lagrange:

- Dado um conjunto Y de pontos interpoladores, o número de operações efetuadas para o cálculo de uma base de polinômios de Lagrange sobre Y é da ordem de p^3.
- A ordem do número de operações efetuadas para calcular o valor numérico de $p(x)$, o polinômio que interpola f em Y, é também p^3.
- Se quisermos acrescentar mais um ponto no conjunto Y, o faremos com ordem de p^2 operações. Não verificaremos essa afirmação neste texto. Sugerimos o livro [9] para maiores detalhes.

5.3 Considerações Sobre o Método de Powell

Uma vez feitos uma introdução, vários comentários e dados alguns exemplos que julgamos úteis à compreensão dos polinômios quadráticos interpoladores de Lagrange, passaremos a tecer nossos comentários sobre o(s) método(s) de Powell, [26], [31], [30], [37], [32].

O último algoritmo proposto dessa classe foi o NEWUOA, em 2004 [30], mas a ideia básica de todos eles é a minimização de um aproximação quadrática local Q para f numa região de confiança. A aproximação quadrática é feita via polinômios interpoladores de Lagrange.

Uma iteração típica dos métodos de Powell é caracterizada por:

1. Aproximar a função objetivo f por uma função quadrática Q que interpole f em alguns pontos escolhidos adequadamente;
2. Minimizar Q numa região se confiança limitada. Os valores de f nos pontos que minimizam Q vão fornecer uma boa informação para a

atualização do modelo quadrático. Cabe observar que às vezes são necessários alguns outros valores, também escolhidos adequadamente, para evitar a degeneração do modelo quadrático.

5.3.1 Sobre a Escolha de m

O controle do número m de pontos de interpolação e as componentes de cada ponto pode ser feito desprezando um dos pontos em uso, que dará lugar a um outro ponto mais adequado.

O número de pontos sugeridos por Powell é $m = 2n+1$ [30], o que torna o número de avaliações da função f da ordem de n, que é a dimensão do espaço onde estamos trabalhando. Este fato é de grande importância, pois nos permite trabalhar com valores grandes de n.

O sucesso dos métodos é devido á técnica usada para atualizar Q. Minimizar Q significa minimizar a norma de Frobenius da variação de $\nabla^2 Q(x)$, entre o valor atual e o candidato a próximo valor, sujeito às condições de interpolação que são impostas.

Conforme dissemos, m é o número de pontos que usaremos para interpolar $f(x)$. Essa quantidade, supondo que $G(x)$, a matriz hessiana de $Q(x)$, seja simétrica, é dada por

$$m = \frac{(n+1)(n+2)}{2},$$

ou seja $m = 1 + n + \dfrac{(n+1)n}{2}$, onde

- $1 \rightarrow$ o termo constante;
- $n \rightarrow$ as coordenadas do gradiente da quadrática;
- $\dfrac{(n+1)(n)}{2} \rightarrow$ os elementos da matriz hessiana - diagonal principal e as $(n-1)$ diagonais secundárias.

No entanto, outras condições são impostas, de forma a conseguirmos trabalhar com $m = 2n+1$, que é o valor sugerido por Powell. Esse valor nos permite manter pelo menos as informações da diagonal principal de $G(x)$, o que, na realidade é obrigatório, dada a escolha exigida para m. Estamos exigindo

- $1 \rightarrow$ o termo constante;

- $n \rightarrow$ as coordenadas do gradiente da quadrática;
- $n \rightarrow$ os elementos da diagonal principal da matriz hessiana.

Dentre todas as opções, o número m de pontos de interpolação a serem utilizados deve estar entre

$$n+2 \leq m \leq \frac{(n+1)(n+2)}{2}.$$

A unicidade de solução é garantida, ou através da resolução de um sistema linear com matriz não singular, ou pela minimização da diferença entre a norma de Frobenius das matrizes hessianas de duas quadráticas consecutivas.

Observamos que, de uma iteração para outra, apenas um ponto de interpolação pode ser modificado. No início de cada iteração, começamos com o vetor x^0 e geramos as posições dos demais pontos iniciais a partir de x^0.

Seguindo a sugestão de Powell, escolhemos $m = 2n+1$, como nos experimentos numéricos que ele apresenta em seus artigos, forçando assim $G(x)$ a ser matriz diagonal. Tais experimentos comprovaram a boa qualidade dessa escolha. No entanto, apresentaremos a seguir uma análise de todas as possíveis escolhas: $n+2 \leq m \leq 2n$ e $m > 2n+1$.

Na primeira iteração, escrevemos o modelo quadrático na forma

$$Q(x_0+d) = Q(x_0) + d^T \nabla Q(x_0) + \frac{1}{2} d^T \nabla^2 Q(x_0) d, \quad d \in I\!R^n.$$

- Se $m = \dfrac{(n+1)(n+2)}{2} = \widehat{m}$:

  Os pontos de interpolação x^i, $i = 1, 2, \cdots, \widehat{m}$ são escolhidos da seguinte forma: tomamos $x^1 = x^0$, o valor inicial; os índices pares $x^{2j} = x^0 + \rho^j e^j$, $j = 1, 2, \cdots, n$, com e^j o $j-$ésimo vetor coordenado em $I\!R^n$. ρ é um limitante inferior para o raio da região de confiança na qual iremos trabalhar. A escolha de x^{2j+1} depende dos valores de $f(x^{2j})$. Powell toma σ_j como -1 no caso em que $f(x^{2j}) \geq f(x^0)$ ou $\sigma_j = 1$ quando $f(x^{2j}) < f(x^0)$. Em UOBYQA é aplicada a seguinte fórmula: para $j = 1, 2, \cdots, n$,

$$x^{2j+1} = \begin{cases} x^0 - \rho_{beg} e^j, & \text{se} \quad \sigma_j = -1 \\ x^0 + 2\rho_{beg} e^j, & \text{se} \quad \sigma_j = +1 \end{cases}$$

Estamos chamando de ρ_{beg} o raio inicial da região de confiança. Além disso, temos os índices i da forma $i(p,q)$ assim definidos:

$$i(p,q) = 2n + 1 + p + \frac{1}{2}(q-1)(q-2), \quad 1 \leq p < q \leq n,$$

e os demais pontos de interpolação iniciais nas posições $i(p,q)$,

$$x^{i(p,q)} = x^0 + \rho_{beg}(\rho_p e^p + \rho_q e^q), \quad 1 \leq p < q \leq n.$$

Usando esta definição, garantimos que todos os índices dos vetores ficarão no intervalo $[2n+2, \widehat{m}]$ de números inteiros.

- Se $m \geq 2n+1$:

  Na igualdade, forçamos $\nabla^2 Q(x_0)$ a ser diagonal. Os primeiros $2n+1$ pontos iniciais são escolhidos da seguinte forma: $x^1 = x^0$ e para $i = 1, 2, \ldots, n$,

$$\left\{ \begin{array}{ccccc} x^{(i+1)} & = & x^0 & + & \rho_{beg} e^i \\ x^{(i+n+1)} & = & x^0 & - & \rho_{beg} e^i \end{array} \right.$$

  Temos então que $Q(x^0)$, $\nabla Q(x^0)$, e os elementos da diagonal $\nabla^2 Q_{ii}$ de $\nabla^2 Q$: para $1 \leq i \leq n$, são determinados univocamente pelas equações de interpolação

$$Q(x^i) = f(x^i),\ 1 \leq i \leq m.$$

- Se $n+2 \leq m \leq 2n$:
  Também neste caso os pontos iniciais de interpolação são os primeiros m vetores do item anterior. Dessa forma, $Q(x_0)$, as $(m-n-1)$ coordenadas de $\nabla Q(x_0)$ e os $(m-n-1)$ elementos da diagonal principal de $\nabla^2 Q(x_0)$ são definidos como anteriormente. Os demais elementos da diagonal de $\nabla^2 Q(x_0)$ são nulos e as demais coordenadas de $\nabla Q(x_0)$, ou seja, para $m-n \leq i \leq n$, as coordenadas de $\nabla Q(x_0)$, assumem os valores $\dfrac{f(x^0 + \rho_{beg} e^i) - f(x^0)}{\rho_{beg}}$.

5.3.2 Sobre Região de Confiança

Esse tema já foi abordado anteriormente na Seção 2.5. Gostaríamos apenas de fazer alguns comentários.

Dada a quadrática $Q^k(x)$ obtida em uma iteração, resolvemos o problema de região de confiança dado por:

$$\begin{aligned} &\text{minimizar } Q(x^{opt}+d) \\ &\text{sujeita a } ||s|| \leq \Delta, \end{aligned}$$

onde k representa a iteração atual e é um número entre 1 e $\overline{m}$, com $\overline{m}$ no intervalo $[n+2, m]$, $m = \dfrac{(n+1)(n+2)}{2}$ e Δ é o raio da região de confiança na qual iremos trabalhar e é limitado inferiormente por ρ. O ponto de interpolação com menor valor da função objetivo f na $k-$ésima iteração é x^{opt}. Esse problema de região de confiança é resolvido por uma versão do algoritmo de gradientes conjugados truncado [17].

A cada iteração obtemos uma direção d a partir de x^{opt} para o cálculo de um novo valor de x, x^{new}.

A razão

$$\frac{F(x^{opt}) - F(x^{opt}+d)}{Q(x^{opt}) - Q(x^{opt}+d)}$$

é então analisada para ver se $Q(x)$ está sendo uma boa representação local para $f(x)$. Essa análise pode levar à substituição de um dos pontos de interpolação. Um ponto de interpolação poderá ter que ser substituído também se a direção d devolvida pelo algoritmo de região de confiança tiver módulo muito pequeno. Assim, o novo ponto de interpolação poderá ser $(x^{opt}+d)$ ou um outro ponto que vise o bom condicionamento do sistema linear que define $Q(x)$.

Para as regras de atualização dos parâmetros Δ e ρ, além de várias outras questões técnicas, sugerimos a consulta do artigo [30], onde todos os detalhes são cuidadosamente explicados.

Uma observação que julgamos pertinente é que, mesmo Powell tendo se baseado em vários teoremas, ao que sabemos até o momento, não há demonstração de convergência dos seus algoritmos, o que não é surpreendente num algoritmo tão complexo como são os apresentados nos trabalhos que estamos estudando.

As tabelas que mostram resultados obtidos com esse algoritmo, apresentadas na próxima seção, comprovam o bom desempenho do mesmo. Observamos que o algoritmo NEWUOA é competitivo para valores grandes de n, uma vez que o esforço computacional por iteração é da ordem de $(m+n)^2$.

5.4 Função de Rosenbrock

Nesta seção e na próxima mostraremos os resultados numéricos obtidos com a implementação do método UOBYQA, com pequenas variações do mesmo, e com o método NEWUOA de Powell, implementados em [37]. Tanto o método UOBYQA quanto o NEWUOA, em cada iteraçao iteração, definem um conjunto de pontos de interpolação para construir o polinômio interpolante $Q(x)$ da função objetivo $f(x)$ e uma aproximação do gradiente e da matriz hessiana do modelo quadrático $Q(x)$. Ambos minimizam o subproblema no contexto de região de confiança, vistos na Seção 2.5. Os diferentes problemas irrestritos resolvidos em [37], aos quais nos referimos neste texto, foram tirados da coleção de Hock e Schittkowski [15].

Foram usadas, para os testes, funções objetivo onde verificamos o comportamento de um algoritmo que chamaremos UOBYQA$_{mod}$, que é o método UOBYQA com as seguintes modificações feitas em alguns dos parâmetros sugeridos em [31]: (i) o parâmetro inicial $\rho_{beg} = 0.2x_1^k$ é trocado por $\rho_{beg} = 0.2$ e (ii) $\frac{1}{6}M$ é trocado $0.5|f(x^k)|$, onde x^k é o novo vetor obtido na k-ésima iteração, o que verifica se o modelo quadrático Q é uma boa aproximação para a função objetivo f no ponto x^k. Os resultados obtidos pelo método UOBYQA$_{mod}$ serão comparados com os que foram obtidos pelo método NEWUOA.

Apresentaremos, a seguir, os resultados obtidos para uma função particular, que é a função de *Rosenbrock* [37] , com $n = 2$:

$$f(x) = 100(x_1 - x_2^2)^2 + (1 - x_1)^2. \tag{5.4.1}$$

Os algoritmos foram implementados em Fortran 77, com os seguintes valores para os parâmetros:

- $\rho_{beg} = 0.2$, raio inicial da região de confiança;
- $\rho_{end} = 10^{-9}$, menor valor aceito para o raio da região de confiança;
- $tol = 10^{-9}$, o algoritmo para quando o tamanho do passo é $\leq tol$;
- $x^0 = (-1.2, 1)$;
- $MAXFUN = 5000$ é o número máximo permitido de avaliações de função.

Função de *Rosenbrock* $n = 2$ e $m = 6$		
Avaliações de f	***O melhor valor de*** f	ρ_{end}
8	2.45994703	10^{-1}
19	1.36825020	10^{-2}
82	2.13697140 x 10^{-6}	10^{-3}
84	1.28499232 x 10^{-6}	10^{-4}
87	6.59415017 x 10^{-11}	10^{-5}
88	1.82715237 x 10^{-13}	10^{-6}
90	1.65634729 x 10^{-15}	10^{-7}
91	2.26337043 x 10^{-21}	10^{-8}
92	5.67320443 x 10^{-25}	10^{-9}

Tabela 5.1: Solução do problema (5.4.1) utilizando $UOBYQA_{mod}$.

A Tabela 5.1 mostra os resultados para $UOBYQA_{mod}$, para o qual $m = \frac{1}{2}(n+1)(n+2) = 6$.

Testando a função de *Rosenbrock* com os mesmos valores iniciais, mas utilizando agora o algoritmo NEWUOA, que usa $m = 2n+1 = 5$ pontos de interpolação, obtemos os resultados da Tabela 5.2.

Função de *Rosenbrock* $n = 2$ **e** $m = 5$		
Avaliações de f	***O melhor valor de*** f	ρ_{end}
10	8.831410	10^{-1}
15	8.831410	10^{-2}
20	6.112364	10^{-3}
25	6.099923	10^{-4}
250	5.647275 x 10^{-10}	10^{-5}
254	4.070714 x 10^{-10}	10^{-6}
266	2.499902 x 10^{-15}	10^{-7}
270	2.222798 x 10^{-17}	10^{-8}
278	7.093898 x 10^{-24}	10^{-9}

Tabela 5.2: Solução do problema (5.4.1) utilizando NEWUOA.

Usamos como medidas de comparação o número de avaliações de função e o valor ótimo de f para cada valor de ρ. Com estas medidas, concluímos que o método UOBYQA$_{mod}$ teve um melhor desempenho que NEWUOA. Vale observar que neste exemplo $n = 2$ é muito pequeno.

Uma outra observação pertinente, feita pelo próprio autor [30], é a limitação do número de pontos com que os métodos UOBYQA, ou mesmo UOBYQA$_{mod}$, podem trabalhar: $m_1 = \frac{1}{2}(n+1)(n+2)$. Comparando com NEWUOA, que usa $m_2 = 2n+1$ vetores, vemos que $m_1 > m_2$ para todo $n > 1$ e assim, para $n > 20$, já não é recomendado o uso de UOBYQA ou UOBYQA$_{mod}$. Por exemplo, se $n = 30$, $m_1 = 496$, ao passo que NEWUOA utiliza uma quantidade muito menor de pontos, $m_2 = 61$.

Quando $f(x^*) = 0$ consideramos que obtivemos sucesso se o valor da função objetivo no ponto final do algoritmo de otimização for menor que 10^{-9}.

5.5 Outros Experimentos Numéricos

Nesta seção mostraremos mais alguns resultados obtidos com *NEWUOA* e teceremos comentários sobre eles. Foram realizados testes numéricos em MatLab, usando problemas irrestritos da coleção proposta por Hock e Schittkowski [15] , [37], e em Fortran, usando problemas da coleção de problemas de Moré, Garbow e Hillstrom [23], [32], que são problemas clássicos de minimização irrestrita. Trataremos primeiramente dos problemas de Moré, Garbow e Hillstrom, dando alguns detalhes da implementação e apresentando alguns resultados obtidos em [32]. Nesses casos, as funções objetivo são soma de quadrados e diferenciáveis.

Mais alguns detalhes da implementação, segundo o software desenvolvido por Powell; o Algoritmo foi programado em Fortran 77, com os seguintes parâmetros:

- $\rho_{beg} = 0.2x_1^0$, raio inicial da região de confiança;
- $\rho_{end} = 10^{-6}$, menor valor aceito para o raio da região de confiança;
- $m = 2n+1$, número de pontos de interpolação ;
- o número máximo de avaliações da função não foi fixado;
- o valor usado para x^0 em cada problema, é o mesmo sugerido em [23].

5.5.1 Conjunto de testes

Em [32] foram realizados testes com os 35 problemas de Moré, Garbow e Hillstrom, cujas dimensões variam de $n = 2$ a $n = 100$. Os problemas são numerados de 1 a 35. Apresentamos aqui a maioria dos resultados e fazemos

uma pequena análise dos mesmos. Nos problemas de 1 a 20, foram mantidas as dimensões sugeridas. Acrescentamos a seguir, mais alguns comentários:

1. As dimensões dos problemas de 1 a 20 são:
   - problemas de 1 a 6, dimensão 2;
   - problemas de 7 a 12, dimensão 3;
   - problemas de 13 a 16, dimensão 4;
   - problema 17, dimensão 5;
   - problema 18, dimensão 6;
   - problema 19, dimensão 11;
   - problema 20, dimensão 12.
2. Para os problemas de 1 a 9 a convergência ocorreu de maneira extremamente rápida, quase instantânea.
3. Para os problemas de 10 a 19 o tempo gasto para a convergência foi menor do que 1 segundo.

Os problemas de 21 a 35 possuem dimensão variável, que foi fixada em $n = 100$ nos testes aqui apresentados, na Tabela 5.3.

Prob	***Avals. de f***	***Tempo(s)***	***Min f***
21	73030	1288.76	4.03244524365 E-08
22	146803	2394.68	1.36997550215 E-09
23	48424	888.79	9.02490976826 E-04
24	33937	470.79	9.70960839546 E+04
25	152525	2145.93	4.34587701001 E-09
26	11209	167.06	1.85702203535 E-07
27	342130	3800.06	3.11784326225 E-09
30	2661	35.14	6.62843877667 E-10
31	4915	65.37	2.91005647371 E-09
32	423	3.04	4.99999999999 E+01
33	1545	21.12	3.71262458471 E+01
34	1603	21.01	3.86262626262 E+01
35	117300	2014.98	8.21466042244 E-03

Tabela 5.3: Testes de Moré, Garbow e Hillstrom com $n = 100$.

As colunas da tabela significam:

- ***Prob*** é o número do problema, atribuído pelos autores em [23];
- ***Aval f*** é o número de avaliações de função efetuadas;
- ***Tempo(s)*** é o tempo gasto, em segundos, para encontrar o minimizador de f;
- ***Min f*** é o valor para o mínimo encontrado em [32].

Observamos que a convergência foi obtida com um número pequeno de avaliações de função e em pouco tempo computacional.

Além dos testes acima, realizados com $m = 2n + 1$, os problemas de 21 a 35 foram testados também com

$$m = \frac{(n+1)(n+2)}{2}.$$

O tempo computacional exigido foi inviável: mais de 5 horas. Esse fato confirma que $m = 2n + 1$ é realmente uma excelente escolha para valores "grandes"de n.

Concluindo este capítulo, gostaríamos de tecer alguns comentários sobre os métodos baseados em interpolação polinomial:

1. Os exemplos apresentados mostram que eles podem ser uma boa alternativa para problemas de otimização nos quais o uso de derivadas não é indicado, como por exemplo, os problemas onde o cálculo das derivadas é extremamente caro, e os problemas tipo caixa-preta, para os quais só se possuem dados, sem qualquer tipo de código.

2. Com NEWUOA vemos que, ainda que a dimensão n do problema seja grande, podemos trabalhar com um número m de pontos de interpolação bem menor do que o número originalmente necessário para a construção de uma quadrática que aproxime f em $\mathbb{R}^n$, que é $m = \frac{(n+1)(n+2)}{2}$.

3. A aproximação da função objetivo por uma função quadrática, usa mais informações da função objetivo do que outros tipos de métodos que trabalham apenas com valores da função.

4. Para os métodos baseados em interpolação polinomial, ainda não existem resultados de convergência.

Finalizamos nosso trabalho, ressaltando que o nosso objetivo neste texto foi apresentar ao leitor alguns métodos sem derivada para problemas de otimização sem restrição, e não comparar os tipos de método entre si, no sentido de mostrar que certo método é melhor ou pior que outro.

5.6 Exercícios

1. Prove que, se $M(\phi, Y)$ é não singular para alguma base ϕ de P_n^d, então ela é não singular para qualquer base de P_n^d.

2. Construa um conjunto de polinômios de Lagrange em $\mathbb{R}^2$ para o conjunto de pontos $Y = \{(0\,,\,0), (1\,,\,0), (0\,,\,1), (-1\,,\,0), (1\,,\,1), (0\,,\,-1)\}$.

3. Mostre que o conjunto $Y = \{(0\,,\,0), (1\,,\,0), (0\,,\,1), (2\,,\,0), (1\,,\,1), (0\,,\,2)\}$ é posicionado para interpolação polinomial em $\mathbb{R}^2$ relativamente a P_3^2 usando como base o conjunto B_2 dado no texto.

4. Idem ao exercício anterior, usando a base dos polinômios de Lagrange.

5. Verifique a equação $M(\phi, Y)A_\phi = I$.

6. Faça os cálculos para obter os 6 polinômios de Lagrange apresentados no final da Seção 1.2.3.

7. Mostre que seis pontos em um círculo em $\mathbb{R}^2$ não são posicionados para interpolação por polinômios quadráticos, mas são posicionados para interpolação num espaço de polinômios cúbicos que não possuem termos quadráticos nem termos constantes.

Bibliografia

[1] L. Armijo, Minimization of functions having Lipschitz-continuous first partial derivatives, *Pacific J. Math.* 17, pp. 1-3, 1966.

[2] M.A. Abramson, C. Audet, J.E. Dennis Jr. e S. Le Digabel, OrthoMads: a deterministic MADS instance with orthogonal directions, a aparecer no *SIAM Journal on Optimization.*

[3] C. Audet, J.E. Dennis Jr e S. Le Digabel, Globalization strategies for mesh adaptative direct search, Technical Report, GERAD G-2008-74, GERAD, 2008.

[4] C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, *SIAM Journal on Optimization* 17, pp. 642-664, 2006.

[5] C. Audet e J.E. Dennis Jr., Mesh adaptive direct search algorithms for constrained optimization, *SIAM Journal on Optimization* 17, pp. 188-217, 2006.

[6] F.H. Clarke, "Optimization and Nonsmooth Analysis", John Wiley & Sons, New York, 1983.

[7] A.R. Conn, K. Scheinberg e P.L. Toint, A derivative free optimization algorithm in practice, Proceedings of the American Institute of Aeronautics and Astronautics Conference, St Louis, 1998.

[8] A.R. Conn, K. Scheinberg e P.L. Toint, On the convergence of derivative-free methods for unconstrained optimization, *Approximation Theory and Optimization: Tributes to M.J.D. Powell,* Cambridge University Press, Cambridge, UK, pp. 83-108, 1997.

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

[10] A.L. Custódio e L.N. Vicente, Using sampling and simplex derivatives in pattern search methods, *SIAM Journal on Optimization* 18, pp. 537-555, 2007.

[11] A.L. Custódio, J.E. Dennis Jr. e L.N. Vicente, Using simplex gradients of nonsmooth functions in direct search methods, *IMA Journal of Numerical Analysis* 28, pp. 770-784, 2008.

[12] J.E. Dennis Jr e R.B. Schanbel, "Numerical Methods for Unconstrained Optimization and Nonlinear Equations", SIAM, Philadelphia, 1996.

[13] M.A. Diniz-Ehrhardt, J.M. Martínez e M. Raydan, A derivative-free nonmonotone line search technique for unconstrained optimization, *Journal of Computational and Applied Mathematics* 219, pp. 383-397, 2008.

[14] A. Friedlander, "Elementos de Programação Não Linear", Editora da Unicamp, Campinas, SP, 1994.

[15] W. Hock e K. Schittkowski, *Testing exemples for nonlinear programming codes,* Springer, Berlin, 1981.

[16] R. Hooke e T.A. Jeeves, Direct search solution of numerical and statistical problems, *Journal of the ACM* 8, pp. 221-229, 1961.

[17] C.T. Kelley, "Iterative Methods for Linear and Nonlinear Equations", Society for Industrial and Applied Mathematics - SIAM, 1999.

[18] J.C. Lagarias, J.A. Reeds, M.H. Wright e P.E. Wright, Convergence properties of the Nelder-Mead simplex algorithm in low dimensions, *SIAM Journal on Optimization* 9, pp. 112-147, 1998.

[19] R.M. Lewis, V. Torczon e M.W. Trosset, Direct search methods: then and now, ICASE Report 2000-26, ICASE, NASA Langley Research Center, Hampton, VA, 2000.

[20] S. Lucidi e M. Sciandrone, On the global convergence of derivative-free methods for unconstrained optimization, *SIAM Journal on Optimization* 13, pp. 97-116, 2002.

[21] D.G. Luenberger e Y. Ye, "Linear and Nonlinear Programming", 3ª edição, Springer, 2008.

[22] K.I.M. McKinnon, Convergence of the Nelder-Mead simplex method to a nonstationaty point, *SIAM Journal on Optimization* 9, pp. 148-158, 1998.

[23] J. Moré, B.S. Garbow K.E. Hillstrom, *Testing unconstrained optimization software*, ACM Transactions on Mathematical Software 7, pp 17-41, 1981.

[24] J.A. Nelder e R. Mead, A simplex method for function minimization, *The Computer Journal* 7, pp. 308-313, 1965.

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

[26] L.G. Pedroso, M.A. Diniz-Ehrhardt (coorientadora) e J.M. Martínez (orientador), "Programação Não Linear Sem Derivadas", tese de doutorado, IMECC, UNICAMP, 2009.

[27] E. Polak, Computational methods in optimization: a unified approach, *Academic Press,* New York, 1971.

[28] M.J.D. Powell, Direct search algorithms for optimization calculations, Acta Numererica 7, pp 287-336, 1998.

[29] M.J.D. Powell, On the use of quadratic models in unconstrained minimization without derivatives, *Optmin. Methods Software* 19, pp 399-411, 2004.

[30] M.J.D. Powell, The NEWUOA software for unconstrained minimization without derivatives, *Large-Scale Nonlinear Optimization,* pp. 255-297, 2006.

[31] M.J.D. Powell, UOBYQA: unconstrained optimization by quadratic approximation, *Mathematical Programming* 92, pp 555-582, 2002.

[32] T. Rincão e M.A. Diniz-Ehrhardt (orientadora), "Otimização Irrestrita sem Derivadas Baseada em Interpolação Polinomial", Dissertação de Mestrado, IMECC, UNICAMP, 2009.

[33] W. Spendley, G.R. Hext, e F.R. Himsworth, Sequential application of simplex designs in optimization and evolutionary operation, *Technometrics* 4, pp. 441-461, 1962.

[34] V. Torczon, On the convergence of pattern search algorithms, *SIAM Journal on Optimization* 7, pp. 1–25, 1997.

[35] V. Torczon e J.E. Dennis Jr (orientador), "Multi-directional Search: a Direct Search Algorithm for Parallel Machines", tese de doutorado, Department of Mathematical Sciences, Rice University, Houston, USA, 1989.

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

[37] L. Urrea-Jiménez, V.L.R. Lopes (orientadora), "Um Método de Região de Confiança para Minimização Irrestrita sem Derivadas", Dissertação de Mestrado, IMECC, UNICAMP, 2008.

[38] D.S.Watkins, "Fundamentals of Matrix Computations", John Wiley & Sons, 2ª edição, 2002.

[39] http://www.gerad.ca/nomad/, consultado em 26/04/2010.

Índice

Notas em Matemática Aplicada

Arquivos em pdf disponíveis em http://www.sbmac.org.br/notas.php

1. Restauração de Imagens com Aplicações em Biologia e Engenharia
   Geraldo Cidade, Antônio Silva Neto e Nilson Costa Roberty
2. Fundamentos, Potencialidades e Aplicações de Algoritmos Evolutivos
   Leandro dos Santos Coelho
3. Modelos Matemáticos e Métodos Numéricos em Águas Subterrâneas
   Edson Wendlander
4. Métodos Numéricos para Equações Diferenciais Parciais
   Maria Cristina de Castro Cunha e Maria Amélia Novais Schleicher
5. Modelagem em Biomatemática
   Joyce da Silva Bevilacqua, Marat Rafikov e Cláudia de Lello Courtouke Guedes
6. Métodos de Otimização Randômica: algoritmos genéticos e "simulated annealing"
   Sezimária F. Pereira Saramago
7. "Matemática Aplicada à Fisiologia e Epidemiologia"
   H.M. Yang, R. Sampaio e A. Sri Ranga
8. Uma Introdução à Computação Quântica
   Renato Portugal, Carlile Campos Lavor, Luiz Mariano Carvalho e Nelson Maculan
9. Aplicações de Análise Fatorial de Correspondências para Análise de Dados
   Homero Chaib Filho

10. Modelos Matemáticos baseados em autômatos celulares para Geoprocessamento

    Marilton Sanchotene de Aguiar, Fábia Amorim da Costa,

    Graçaliz Pereira Dimuro e Antônio Carlos da Rocha Costa

11. Computabilidade: os limites da Computação

    Regivan H. N. Santiago e Benjamín R. C. Bedregal

12. Modelagem Multiescala em Materiais e Estruturas

    Fernando Rochinha e Alexandre Madureira

13. Modelagem em Biomatemática (Coraci Malta ed.)

    1 - "Modelagem matemática do comportamento elétrico de neurônios e algumas aplicações"

        Reynaldo D. Pinto

    2 - "Redes complexas e aplicações nas Ciências"

        José Carlos M. Mombach

    3 - "Possíveis níveis de complexidade na modelagem de sistemas biológicos"

        Henrique L. Lenzi, Waldemiro de Souza Romanha e Marcelo Pelajo- Machado

14. A lógica na construção dos argumentos

    Angela Cruz e José Eduardo de Almeida Moura

15. Modelagem Matemática e Simulação Numérica em Dinâmica dos Fluidos

    Valdemir G. Ferreira, Hélio A. Navarro, Magda K. Kaibara

16. Introdução ao Tratamento da Informação nos Ensinos Fundamental e Médio

    Marcilia Andrade Campos, Paulo Figueiredo Lima

17. Teoria dos Conjuntos Fuzzy com Aplicações

    Rosana Sueli da Motta Jafelice, Laércio Carvalho de Barros,

    Rodney Carlos Bassanezi

18. Introdução à Construção de Modelos de Otimização Linear e Inteira

    Socorro Rangel

19. Observar e Pensar, antes de Modelar
    Flavio Shigeo Yamamoto, Sérgio Alves, Edson P. Marques Filho, Amauri P. de Oliveira
20. Frações Contínuas: Propriedades e Aplicações
    Eliana Xavier Linhares de Andrade, Cleonice Fátima Bracciali
21. Uma Introdução à Teoria de Códigos
    Carlile Campos Lavor, Marcelo Muniz Silva Alves, Rogério Monteiro de Siqueira, Sueli Irene Rodrigues Costa
22. Análise e Processamento de Sinais
    Rubens Sampaio, Edson Cataldo, Alexandre de Souza Brandão
23. Introdução aos Métodos Discretos de Análise Numérica de EDO e EDP
    David Soares Pinto Júnior
24. Representações Computacionais de Grafos
    Lílian Markenzon, Oswaldo Vernet
25. Ondas Oceânicas de Superfície
    Leandro Farina
26. Técnicas de Modelagem de Processos Epidêmicos e Evolucionários
    Domingos Alves, Henrique Fabrício Gagliardi
27. Introdução à teoria espectral de grafos com aplicações
    Nair Maria Maia de Abreu, Renata Raposo Del-Vecchio, Cybele Tavares Maia Vinagre e Dragan Stevanović
28. Modelagem e convexidade
    Eduardo Cursi e Rubens Sampaio
29. Modelagem matemática em finanças quantitativas em tempo discreto
    Max Oliveira de Souza e Jorge Zubelli
30. Programação não linear em dois níveis: aplicação em Engenharia Mecânica
    Ana Friedlander e Eduardo Fancello

31. Funções simétricas e aplicações em Combinatória
    José Plinio de Oliveira Santos e Robson da Silva
32. Semigrupos aplicados a sistemas dissipativos em EDP
    Carlos Raposo da Cunha
33. Introdução à Simulação Estocástica para Atuária e Finanças Usando R
    Hélio Côrtes Vieira, Alejandro C. Frery e Luciano Vereda
34. Modelos de Sustentabilidade nas Paisagens Amazônicas Alagáveis
    Maurício Vieira Kritz, Jaqueline Maria da Silva e Cláudia Mazza
35. Uma Introdução à Dinâmica Estocástica de Populações
    Leonardo Paulo Maia
36. Geometria de Algoritmos Numéricos
    Gregorio Malajovich
37. Equações Diferenciais, Teorema do Resíduo e as Transformadas Integrais
    Edmundo Capelas de Oliveira e Jayme Vaz Júnior
38. Métodos Matemáticos e Computacionais em Música
    Paulo Cezar Carvalho,Luiz Velho, Marcelo Cicconet e Sergio Krakowski
39. Métodos para Problemas Inversos de Grande Porte
    Fermín S. Viloche Bazán e Leonardo Silveira Borges
40. TerraME : Suporte a Modelagem Ambiental Multi-Escalas Integrada a Bancos de Dados Geográficos
    Tiago Garcia de Senna Carneiro e Gilberto Camara
41. Técnicas de Inteligência Computacional Inspiradas na Natureza - Aplicações em Problemas Inversos em Transferência Radiativa
    Antônio J. Silva Neto e José Carlos Becceneri
42. Avanços em Métodos de Krylov para Solução de Sistemas Lineares de Grande Porte
    Luiz Mariano Carvalho e Serge Gratton

43. Uma Abordagem para Modelagem de Dados com o Uso de Sistemas Neuro-Fuzzy: Aplicações Geoespaciais

    Luiz Carlos Benini e Messias Meneguette Jr

44. Construções Concretas e Geometria Dinâmica: Abordagens Interligadas para o Estudo de Cônicas

    Angela Rocha dos Santos