
Bruno Henrique Cervelin

# Sobre um Método de Minimização Irrestrita Baseado em Derivadas Simplex

**Campinas**
**2013**


Universidade Estadual de Campinas

Instituto de Matemática, Estatística
e Computação Científica

Bruno Henrique Cervelin

# Sobre um Método de Minimização Irrestrita Baseado em Derivadas Simplex

Orientadora: Profa. Dra. Maria Aparecida Diniz Ehrhardt.

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

Este exemplar corresponde à versão final da dissertação defendida pelo aluno Bruno Henrique Cervelin, e orientada pela Profa. Dra. Maria Aparecida Diniz Ehrhardt.

Assinatura da Orientadora

Campinas
2013

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

Cervelin, Bruno Henrique, 1988-
C337s Sobre um método de minimização irrestrita baseado em derivadas simplex / Bruno Henrique Cervelin. – Campinas, SP : [s.n.], 2013.

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

1. Otimização sem derivadas. 2. Programação não-linear. 3. Métodos de busca padrão. 4. Otimização matemática. 5. Algoritmos. I. Ehrhardt, Maria Aparecida Diniz,1956-. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Informações para Biblioteca Digital

**Título em inglês:** About an unconstrained minimization method based on simplex derivatives
**Palavras-chave em inglês:**
Derivative-free optimization
Nonlinear programming
Pattern search methods
Mathematical optimization
Algorithms
**Área de concentração:** Matemática Aplicada
**Titulação:** Mestre em Matemática Aplicada
**Banca examinadora:**
Maria Aparecida Diniz Ehrhardt [Orientador]
José Mario Martínez Perez
Ernesto Julián Goldberg Birgin
**Data de defesa:** 08-04-2013
**Programa de Pós-Graduação:** Matemática Aplicada

**Dissertação de Mestrado defendida em 08 de abril de 2013 e aprovada**

**Pela Banca Examinadora composta pelos Profs. Drs.**

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

**Prof.(a). Dr(a). JOSÉ MARIO MARTÍNEZ PÉREZ**

**Prof.(a). Dr(a). ERNESTO JULIÁN GOLDBERG BIRGIN**

# Agradecimentos

Agradeço à Capes pela ajuda financeira.

Aos meus amigos que tanto me ajudaram e tanto me atrapalharam a concluir este projeto.

À Sandra por me fazer sorrir nas horas em que tudo parecia perdido.

À Cheti, a melhor orientadora que qualquer aluno poderia esperar.

Mas, principalmente, agradeço aos meus pais, sem o seu suporte jamais teria concluído mais esta etapa.

# Resumo

O objetivo deste trabalho é apresentar alguns métodos de minimização irrestrita sem derivadas, tais como, Nelder-Mead, busca padrão e SID-PSM, assim como compará-los. Ainda pretendemos apresentar o problema de otimização de parâmetros de algoritmos, e aplicar o método SID-PSM de modo a encontrar parâmetros ótimos para o próprio método SID-PSM em relação ao número de avaliações de função que o método realiza. Os experimentos numéricos realizados mostram que o SID-PSM é mais robusto e mais eficiente que os métodos clássicos sem derivadas (busca padrão e Nelder-Mead). Outros experimentos nos mostram o potencial do problema de otimização de parâmetros de algoritmos em melhorar tanto a eficiência quanto a robustez dos métodos.

**Palavras chave:** Otimização sem derivadas; Otimização de parâmetros de algoritmos; SID-PSM; Otimização irrestrita; Derivadas simplex.

# Abstract

The aim of this paper is to present some derivative-free methods for unconstrained minimization problems, such as Nelder-Mead, pattern search and SID-PSM, and compare them. We also intend to present the problem of optimal algorithmic parameters, and apply the method SID-PSM in order to find optimal parameters for the method SID-PSM itself in relation to the number of function evaluations performed by the method. The numerical experiments performed show that the SID-PSM is more robust and more efficient than the classical derivative-free methods (pattern search and Nelder-Mead). Other experiments show us the potential of the problem of optimal algorithmic parameters to improve both the efficiency and the robustness of the methods.

**Keywords:** Derivative-free optimization; Optimal algorithmic parameters; SID-PSM; Unconstrained minimization; Simplex derivative.

# Lista de Notação

Aqui incluímos uma lista de notações usadas ao longo deste projeto:

- $\|x\| \longrightarrow$ norma-2 de $x \in I\!R^n$,

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2};$$

- $\|x\|_\infty \longrightarrow$ norma-infinito de $x \in I\!R^n$,

$$\|x\|_\infty = \max_{i=1,\ldots,n} \{|x_i|\};$$

- $\|H\|_F \longrightarrow$ norma de Frobenius de $H \in I\!R^{m\times n}$,

$$\|H\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} h_{ij}^2};$$

- $|S| \longrightarrow$ cardinalidade do conjunto $S$;

- $\mathcal{L}(x_0) \longrightarrow$ conjunto de nível de uma função $f$, associado a $f(x_0)$, com $x_0 \in I\!R^n$,

$$\mathcal{L}(x_0) = \{x \in I\!R^n \,:\, f(x) \leq f(x_0)\};$$

- $L(S) \longrightarrow$ matriz de diferença entre pontos do conjunto $S = \{y^0, y^1, \ldots, y^q\} \subset I\!R^n$,

$$L(S) = \left[y^1 - y^0, \ldots, y^q - y^0\right].$$

  Podemos também omitir qual o conjunto no qual estamos avaliando a função quando não atrapalhar no entendimento do texto, ou seja, podemos representar $L(S) = L$;

- $\delta f(S) \longrightarrow$ vetor de diferença de valor da função $f$ nos pontos do conjunto $S = \{y^0, y^1, \ldots, y^q\} \subset I\!R^n$,

$$\delta f(S) = \begin{bmatrix} f(y^1) - f(y^0) \\ f(y^2) - f(y^0) \\ \vdots \\ f(y^q) - f(y^0) \end{bmatrix};$$

- $N(A) \longrightarrow$ núcleo da matriz $A \in \mathbb{R}^{m\times n}$,

$$N(A) = \{x \in \mathbb{R}^n : Ax = 0\};$$

- $sign(c) \longrightarrow$ sinal do número $c \in \mathbb{R}$,

$$sign(c) = \begin{cases} +1 & \text{se } c \geq 0 \\ -1 & \text{se } c < 0 \end{cases};$$

- $B(y, \Delta) \longrightarrow$ bola centrada em $y \in \mathbb{R}^n$ de raio $\Delta > 0$,

$$B(y, \Delta) = \{x \in \mathbb{R}^n \, : \, \|x - y\| \leq \Delta\}.$$

# Conteúdo

# Introdução

Neste trabalho estamos interessados em resolver problemas do tipo

$$\min_{x \in \mathbb{R}^n} f(x), \tag{1}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$ é uma função continuamente diferenciável. Em geral, para resolver tais problemas utilizamos métodos que se baseiam em alguma derivada de $f$. Exemplos são o método de Newton e os métodos Quase-Newton [3, 14]. Apesar da já conhecida eficiência desses métodos e de toda sua base teórica, vários pesquisadores trabalham com métodos que não utilizam derivadas nem aproximações locais para elas. Esses métodos constituem a classe de *derivative-free methods*.

Desenvolvidos inicialmente como heurísticas, hoje esses métodos ganharam uma ampla teoria de convergência, em alguns casos, tão forte quanto a dos métodos que se baseiam em derivadas [13]. Exemplos de subclasses são os métodos de busca direta, dentre os quais podemos citar os métodos Nelder-Mead [16] e busca coordenada [6], e também os métodos de busca linear, aos quais pertence o método do filtro implícito [11].

A ideia principal dos métodos que não utilizam derivadas não é competir com os métodos clássicos de otimização, mas sim utilizá-los quando estes falham, como por exemplo nos casos em que, apesar de sabermos teoricamente que as derivadas existem, não podemos calculá-las analiticamente. Outra vantagem desses métodos é a sua simpliciade de implementação, além de bons resultados numéricos na prática.

Um exemplo de problema onde, apesar de sabermos da existência teórica das derivadas, não temos acesso a elas, pode ser encontrado em [17]. O autor resolve o problema de minimizar a área de figuras, em que a função área é diferenciável. Porém pode ser complicado obtê-la e uma alternativa é resolver o problema utilizando simulação. Neste caso as funções apresentam "ruídos", tornando inviável a diferenciação numérica. Logo a melhor alternativa é utilizar métodos sem derivadas.

O método Nelder-Mead está entre os métodos sem derivadas mais utilizados. Apesar de não possuir convergência teórica garantida e resultados insatisfatórios em alguns problemas [20], no geral, ele funciona bem. Esse método consiste em, a partir de um simplex, substituir o pior ponto (com maior valor de função objetivo) por outro melhor, fazendo reflexão, expansão ou contração através da face oposta ao vértice. Ou ainda, quando todas as outras formas falham, redução do tamanho do simplex. Esse método está na classe dos métodos de busca simplex.

Outra classe que se destaca é a de busca padrão. Partindo de uma aproximação para o minimizador da função, estes métodos procuram uma aproximação melhor, andando por direções de um conjunto ordenado por um padrão pré-definido. Dentro dessa classe está o método de busca coordenada que, apesar de sua simplicidade, possui ampla teoria de convergência.

Alguns métodos da classe de busca padrão podem fazer parte da classe de busca direta, na qual só se faz uso do valor da função de forma comparativa. Nesse caso não precisamos avaliar numericamente o valor da função, precisamos apenas avaliar se o ponto é melhor ou pior que o atual. Entretanto não podemos utilizar estratégias mais elaboradas para tentar acelerar a convergência, como por exemplo, a condição de decréscimo suficiente de Armijo-Goldstein-Wolfe [9] ou busca por direções dadas por derivadas simplex [11].

Um método de busca padrão que se destaca é o método SID-PSM (SImplex Derivatives in Pattern Search Method - derivadas simplex no método de busca padrão), desenvolvido por Custódio e Vicente [7]. Esse método não é de busca direta, pois nele utilizamos os valores da função objetivo para calcular derivadas simplex. O diferencial desse método é a possibilidade de utilizar estas derivadas para procurar uma direção de descida em potencial, calcular o tamanho do passo na próxima iteração, ordenar e/ou podar o conjunto de direções de pesquisa, ou até mesmo como critério de parada.

Neste trabalho discutiremos, no Capítulo 1, algumas características dos métodos de busca direcional e algumas de suas subclasses. No Capítulo 2, mostraremos algumas propriedades interessantes sobre derivadas simplex, para então, no Capítulo 3, trabalharmos com métodos de busca linear sem derivadas. No Capítulo 4, faremos um estudo sobre o método SID-PSM, mostrando quais as opções e possibilidades de uso. No Capítulo 5, seguiremos com alguns experimentos numéricos comparando o método SID-PSM com outros métodos estudados. E, por fim, no Capítulo 6 apresentaremos o problema de otimização de parâmetros de algoritmos e aplicaremos o método SID-PSM para otimizar a ele mesmo (além de apresentarmos algumas comparações).

# Capítulo 1

# Métodos de Busca Direta

Os métodos de busca direta são métodos iterativos que fazem parte da classe *derivative-free*, cuja principal característica é não calcular diretamente as derivadas das funções envolvidas no problema. Nos métodos de busca direta não calculamos nem aproximamos as derivadas [13], e ainda utilizamos os valores da função objetivo apenas de forma comparativa, ou seja, em uma iteração $k$, ordenamos $p$ pontos $x_k^0, x_k^1, \cdots, x_k^p$ de modo que

$$f(x_k^0) \leq f(x_k^1) \leq \ldots \leq f(x_k^p). \tag{1.1}$$

Assim não precisamos avaliar numericamente a função objetivo; porém não podemos utilizar estratégias de decréscimo suficiente, como por exemplo a condição de Armijo-Goldstein-Wolfe [9].

Esta classe de métodos pode ser divida em duas subclasses: métodos de busca simplex e métodos de busca padrão.

## 1.1 Métodos de Busca Simplex

Métodos de busca simplex têm uma grande importância em aplicações, visto que o método de Nelder-Mead [16], que faz parte desta classe, mesmo com sua falta de resultados teóricos em relação à convergência, é um dos métodos de busca direta mais utilizados, isto devido à facilidade de implementação e aos bons resultados obtidos na prática.

Um simplex é um conjunto de $n+1$ pontos no $\mathbb{R}^n$. Chamamos os pontos $x_0, x_1, \ldots, x_n$, que definem o simplex, de vértices, e trabalhamos sempre com simplex não degenerados, ou seja, o conjunto $\{x_1 - x_0, x_2 - x_0, \ldots, x_n - x_0\}$ é linearmente independente.

Métodos simplex consistem em, a cada iteração $k$, ordernar $n+1$ pontos $x_k^0, x_k^1, \ldots, x_k^n$ como em (1.1) e substituir o ponto $x_k^n$ por um de menor valor de função objetivo. A maneira mais usual é fazer este novo ponto como a reflexão de $x_k^n$ através do centróide da face oposta do simplex, como mostrado na Figura 1.1.

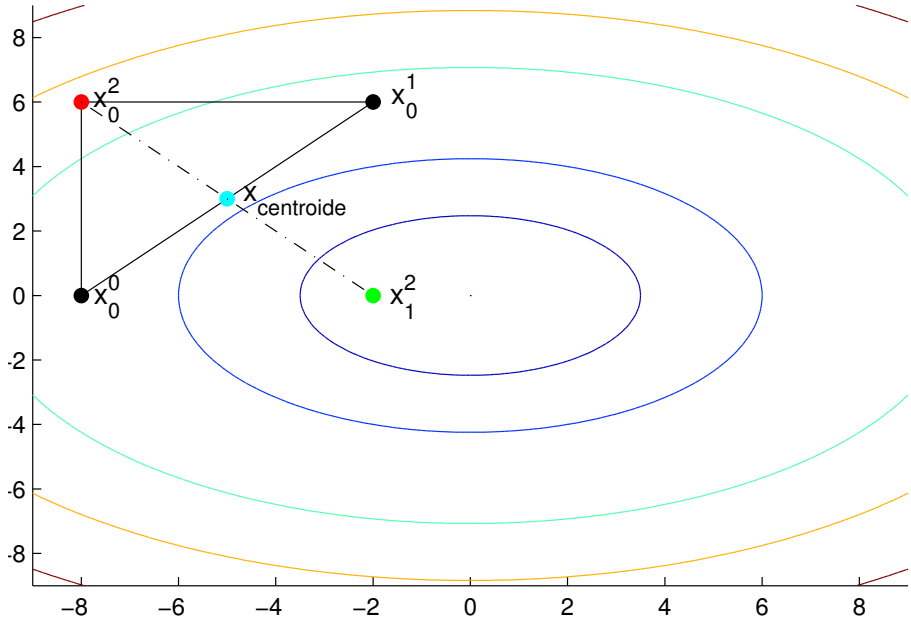



Figura 1.1: Simplex refletindo vértice $x_0^2$ pelo centróide da face oposta

### 1.1.1 Método de Nelder-Mead

O método de Nelder-Mead [16] é um método de busca simplex; a cada iteração $k$ ordenamos os $n+1$ pontos e tentamos substituir o pior deles, $x_k^n$, por um ponto de menor valor de função objetivo. O diferencial deste método é a possibilidade de contração ou expansão do simplex. Assim, a cada iteração, após ordenar os pontos, tentamos:

1. refletir $x_k^n$ pelo centróide da face oposta calculando $x_k^r = (1+\rho)\bar{x} - \rho x_k^n$, onde $\rho > 0$ é o coeficiente de reflexão e $\bar{x} = \sum_{i=0}^{n-1} x_k^i/n$ é o centróide da face oposta a $x_k^n$;

2. se $f(x_k^r) < f(x_k^0)$, tentamos expandir o simplex calculando $x_k^e = (1+\rho\chi)\bar{x} - \rho\chi x_k^n$, onde $\chi > 1$ é o coeficiente de expansão; se $f(x_k^e) < f(x_k^r)$ aceitamos $x_k^e$ e terminamos a iteração. Caso contrário aceitamos $x_k^r$ e terminamos a iteração.

3. Se $f(x^r) \leq f(x^{n-1})$ aceitamos $x_k^r$ e terminamos a iteração.

4. Se $f(x_k^r) > f(x_k^{n-1})$, fazemos uma contração. Dois tipos de contrações são permitidos:

   - Contração externa: se $f(x_k^{n-1}) < f(x_k^r) < f(x_k^n)$ calculamos $x_k^c = (1+\rho\gamma)\bar{x} - \rho\gamma x^n$, onde $0 < \gamma < 1$. Se $f(x_k^c) < f(x_k^r)$ aceitamos $x_k^c$ e terminamos a iteração, caso contrário, vá para o passo 5.
   - Contração interna: Se $f(x_k^r) \geq f(x_k^n)$, calculamos $x_k^c = (1-\gamma)\bar{x} + \gamma x^n$. Se $f(x_k^c) < f(x_k^n)$ aceitamos $x_k^c$ e terminamos a iteração, caso contrário, vá para o passo 5.

5. Fazemos a redução do simplex, calculando $v_k^i = x_k^0 + \sigma(x_k^i - x_k^1)$, para todo $i = 1, \ldots, n$. Os vértices da próxima iteração serão $x_k^0, v_k^1, v_k^2 \ldots v_k^n$.

Como dito anteriormente, o método de Nelder-Mead não possui garantia de convergência para problemas $n$-dimensionais, porém para problemas com até 2 variáveis temos os seguintes teoremas demonstrados em [12].

**Teorema 1** *Seja $f : \mathbb{R} \to \mathbb{R}$ estritamente convexa, limitada inferiormente. Se aplicamos o Algoritmo de Nelder-Mead, com $\rho\chi \geq 1$, partindo de um simplex inicial não degenerado, então a sequência de pontos gerados pelo método converge ao minimizador.*

**Teorema 2** *Seja $f : \mathbb{R}^2 \to \mathbb{R}$ estritamente convexa com curvas de nível limitadas. Se o simplex inicial é não degenerado e o Algoritmo de Nelder-Mead usa $\rho = 1$ e $\gamma = 0,5$, então os três vértices limites do simplex possuem o mesmo valor de função.*

**Teorema 3** *Seja $f : \mathbb{R}^2 \to \mathbb{R}$ estritamente convexa com curvas de nível limitadas. Se o simplex inicial é não degenerado e o Algoritmo de Nelder-Mead usa $\rho = 1$, $\chi = 2$ e $\gamma = 0,5$, então a sequência de diâmetros dos simplex tende a zero, ou seja,*

$$\max_{i \neq j} \|x_k^i - x_k^j\| \to 0.$$

Devemos enfatizar que mesmo para funções estritamente convexas no $\mathbb{R}^2$, não temos garantia que o método converge para um ponto estacionário.

## 1.2 Métodos de Busca Padrão

Os métodos desta classe avaliam a função objetivo em um padrão de pontos para então decidir qual o próximo iterando (aproximação para o minimizador). O padrão mais intuitivo é definido pelas coordenadas cartesianas, positivas e negativas, pois desta forma temos um conjunto gerador positivo para o $\mathbb{R}^n$.

**Definição 1** *Um conjunto gerador positivo no $\mathbb{R}^n$ é um conjunto de vetores cuja combinação linear positiva gera o $\mathbb{R}^n$.*

*O conjunto $\{v_1, \ldots, v_k\}$ é dito positivamente dependente se um de seus vetores está no cone convexo positivo gerado pelos vetores restantes, ou seja, um de seus vetores é uma combinação linear positiva dos outros; caso contrário, o conjunto é positivamente independente.*

*Uma base positiva no $\mathbb{R}^n$ é um conjunto positivamente independente que gera postivamente o $\mathbb{R}^n$.*

No método de busca coordenada [8], a partir de um ponto $x_k$, procuramos um ponto $x_{k+1}$ que diminua o valor da função objetivo andando uma distância $\alpha_k > 0$ nas direções cartesianas (positivas e negativas), ou seja, procuramos $x_{k+1}$ no conjunto $\{x_k + \alpha_k d : d \in \{e_1, e_2, \ldots, e_n, -e_1, \ldots, -e_n\}\}$. Este processo está representado na Figura 1.2.

Se não for encontrado nenhum ponto que diminua o valor da função, declaramos fracasso da iteração, reduzimos o tamanho do passo $\alpha_k$ e repetimos o processo anterior. Caso contrário, declaramos sucesso, aceitamos $x_{k+1}$ e repetimos o processo anterior. Um possível critério de parada é o tamanho de $\alpha_k$.

Por simplicidade de notação trataremos as matrizes com que trabalhamos como conjuntos; por exemplo, se tivermos uma matriz $A \in \mathbb{R}^{n\times m}$ e $a^i \in \mathbb{R}^n$ for uma coluna de $A$, diremos que $a^i \in A$.

Podemos descrever os métodos de busca padrão de uma forma mais geral, assim como feito em [6], em que os autores dividem este tipo de método em três passos.

No primeiro passo, o de busca, avaliamos finitos pontos em uma malha

$$M_k = \{x_k + \alpha_k D z_k \,:\, z_k \in \mathbb{Z}_+^{|D|}\}, \tag{1.2}$$

onde $D$ é um conjunto gerador positivo do $\mathbb{R}^n$; se encontrarmos $y \in M_k$ tal que $f(y) < f(x_k)$ declaramos sucesso da iteração, fazemos $x_{k+1} = y$ e pulamos o segundo passo.

No segundo passo, o de pesquisa, avaliamos os pontos do conjunto

$$P_k = \{x_k + \alpha_k d \,:\, d \in D_k\},$$

onde $D_k \subset D$ é uma base geradora positiva do $\mathbb{R}^n$, seguindo um padrão pré-definido, até encontrarmos $x \in P_k$ tal que $f(x) < f(x_k)$. Neste caso, declaramos sucesso e definimos $x_{k+1} = x$. Senão, tendo avaliado todos os pontos de $P_k$ sem obter decréscimo de $f$, declaramos fracasso. Notemos que $P_k \subset M_k$.

O terceiro passo é a atualização de parâmetros. Se declaramos sucesso, podemos manter ou aumentar o tamanho do passo, fazendo $\alpha_{k+1} \in [\alpha_k\,,\, \gamma\alpha_k]$, com $\gamma \geq 1$; caso contrário, fazemos $x_{k+1} = x_k$ e diminuímos o tamanho do passo, tomando $\alpha_{k+1} \in [\beta_1\alpha_k\,,\, \beta_2\alpha_k]$, com $0 < \beta_1 \leq \beta_2 < 1$.

Existem muitos trabalhos que exploram a questão da convergência para os diferentes tipos de métodos de busca padrão, porém Virginia Torczon, em [18], unificou estes métodos sob a mesma teoria de convergência. Apresentaremos, a seguir, alguns resultados deste artigo.

Na iteração $k$, as direções nas quais procuramos o candidato a $x_{k+1}$ devem estar em um padrão definido por uma base $B \in \mathbb{R}^{n\times n}$ e uma matriz geradora $C_k \in \mathbb{Z}^{n\times p}$, onde $p > 2n$, que pode ser particionada em três componentes:

$$C_k = [A_k \qquad -A_k \qquad L_k] = [\Gamma_k \qquad L_k].$$

Devemos impor que $A_k$ esteja em um subconjunto finito $A \subset \mathbb{Z}^{n\times n}$ de matrizes não singulares, e $L_k \in \mathbb{Z}^{n\times(p-2n)}$ contenha pelo menos uma coluna, a coluna de zeros. O padrão de pontos é definido pelas colunas da matriz $\bar{P}_k = BC_k$. Assim, dado o tamanho do passo $\alpha_k$, os canditados a $x_{k+1}$ têm a forma $\bar{x}^i_{k+1} = x_k + \alpha_k B c^i_k$, onde $c^i_k \in C_k$, para todo $i = 1, 2, \ldots, p$.

Vemos que esta definição engloba a forma como o método foi descrito anteriormente.

A autora impõe algumas hipóteses para demonstrar a convergência. Os passos de busca e pesquisa devem satisfazer:

- todo canditado a novo iterando na iteração $k$ é da forma

$$\bar{x}_{k+1} = x_k + \alpha_k d_k,$$

  com $d_k \in \bar{P}_k$;

- se $\min\{f(x_k + \alpha_k d_k) : d_k \in \bar{P}_k\} < f(x_k)$, então $f(x_{k+1}) < f(x_k)$.

Definimos $\theta = \tau^{\omega_0}$, com $1 < \tau \in \mathbb{Q}$ e $0 > \omega_0 \in \mathbb{Z}$, e $\lambda_k \in \{\tau^{\omega_1}, \tau^{\omega_2}, \ldots, \tau^{\omega_L}\}$, com $L < \infty$ e $\omega_j \in \mathbb{Z}_+ \forall j = 1, \ldots, L$.

- Se foi declarado sucesso, $\alpha_{k+1} = \lambda_k \alpha_k$.
- Caso contrário, $\alpha_{k+1} = \theta \alpha_k$.

Abaixo enunciaremos alguns resultados interessantes que estão demonstrados em [18].

**Lema 1** *Existe uma constante $\zeta_* > 0$, independente de $k$, tal que para qualquer ponto $\bar{x}_{k+1}$ testado pelo método temos que*

$$\|x_k - \bar{x}_{k+1}\| \geq \zeta_* \alpha_k.$$

**Teorema 4** *Qualquer iterando $x_k$ produzido por um método de busca padrão pode ser escrito como*

$$x_k = x_0 + (\beta^{r_{li}} \gamma^{-r_{ls}}) \alpha_0 B \sum_{j=0}^{k-1} z_j,$$

*onde:*

- *$x_0$ é o ponto inicial;*
- *$\beta/\gamma \equiv \tau$, com $\beta$, $\gamma \in \mathbb{N}$ e primos entre si, e $\tau$ é definido como na atualização de $\alpha_k$;*
- *$r_{ls}$ e $r_{li}$ dependem de $k$;*
- *$\alpha_0$ é o tamanho do passo inicial;*
- *$B$ é a matriz de base, e*
- *$z_k \in \mathbb{Z}^n$, $k = 0, \ldots, k-1$.*

Este teorema nos mostra que todo ponto analisado pelo método de busca padrão fica em uma malha inteira gerada por $x_0$ e as colunas de $\beta^{r_{li}} \gamma^{r_{ls}} \alpha_0 B$.

Com o próximo resultado a autora demonstra que, usando decréscimo simples, o método gera uma sequência de tamanhos de passo limitada inferiormente por 0.

**Teorema 5** *Se o conjunto de nível $\mathcal{L}(x_0)$ é compacto, então*

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

Em [19] a autora demonstra a seguinte proposição.

**Proposição 1** *Se o conjunto de nível $\mathcal{L}(x_0)$ é compacto, $f$ continuamente diferenciável e*

$$\liminf_{k\to+\infty} \|\nabla f(x_k)\| \neq 0,$$

*então existe uma constante $\alpha_{li} > 0$ tal que para todo $k$, $\alpha_k > \alpha_{li}$.*

Usando esta proposição, em [18] é demonstrada a convergência fraca do método.

**Teorema 6** *Suponha que $\mathcal{L}(x_0)$ é compacto e $f$ é continuamente diferenciável em uma vizinhança de $\mathcal{L}(x_0)$. Então a sequência dos iterandos $\{x_k\}$ gerada pelo método de busca padrão satisfaz*

$$\liminf_{k\to+\infty} \|\nabla f(x_k)\| = 0.$$

Podemos modificar o método para que ele tenha uma convergência forte, ou seja, para que satisfaça $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$. Para isso devemos impor novas condições.

A autora ainda demonstra a existência de um limitante superior, em termos de $\alpha_k$, para a norma das direções a serem avaliadas.

**Lema 2** *Dada a constante $\mathfrak{C} > 0$ tal que, para todo $k$, $\|c_k^j\| < \mathfrak{C}$ para $j = 1, \ldots, p$, temos que existe uma constante $\psi_* > 0$, independente de $k$, de modo que qualquer ponto $\bar{x}_{k+1}$ avaliado pelo método de busca padrão satisfaz:*

$$\alpha_k \geq \psi_* \|x_k - \bar{x}_{k+1}\|.$$

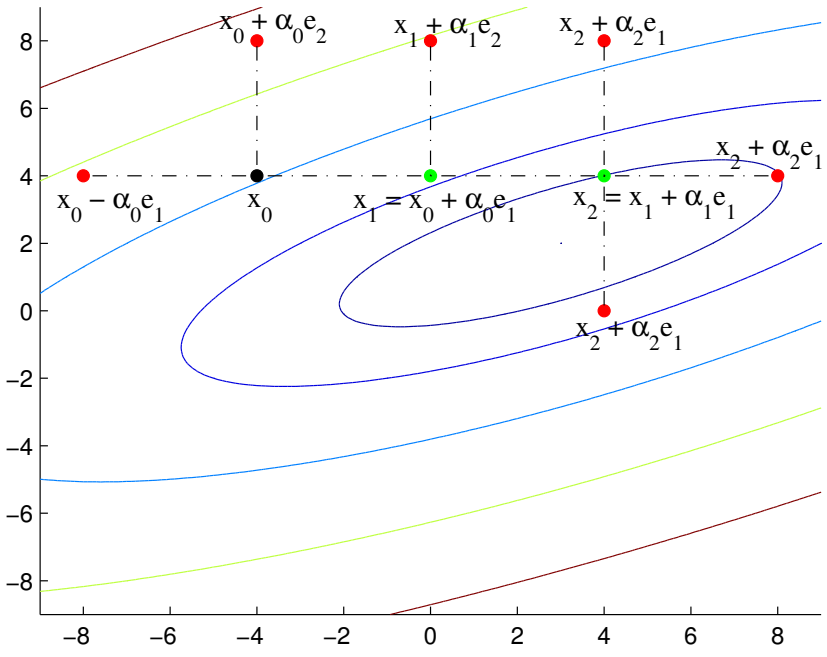
Para garantir que $\lim_{k\to+\infty} \alpha_k = 0$ podemos, por exemplo, impedir que o tamanho do passo cresça, tomando $\omega_i = 0$, $i = 1, \ldots, L$.

Assim é possível demonstrar a convergência forte:

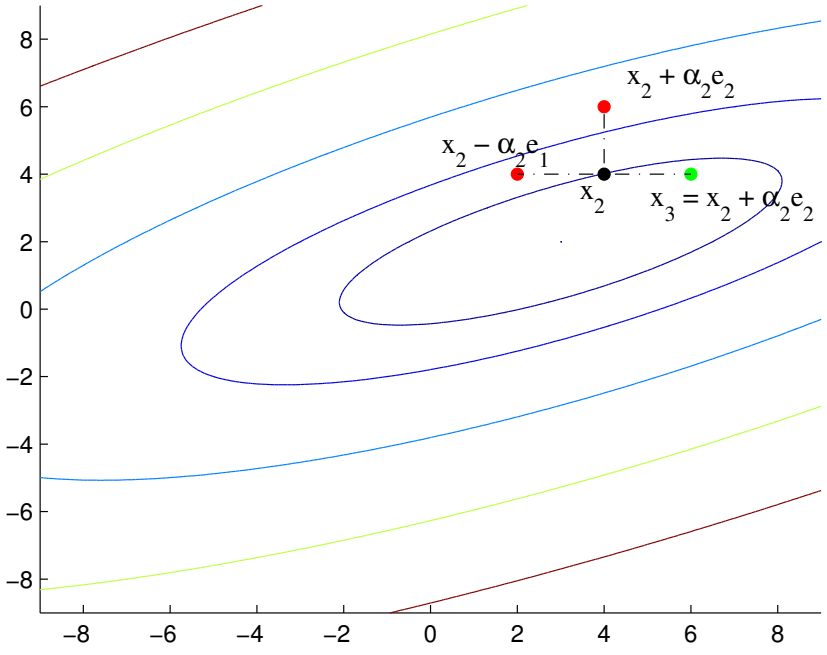
**Teorema 7** *Suponha $\mathcal{L}(x_0)$ compacto e $f$ continuamente diferenciável em uma vizinhança de $\mathcal{L}(x_0)$. Suponha ainda que as colunas das matrizes geradoras têm normas limitadas, que $\lim_{k\to+\infty} \alpha_k = 0$, e que, se $\min\{f(x_k + \alpha_k d) : d \in \bar{P}_k\} < f(x_k)$, então $f(x_{k+1}) \leq \min\{f(x_k + \alpha_k d) : d \in \bar{P}_k\}$. Daí a sequência dos iterandos gerada pelo método de busca padrão satisfaz:*

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

Existem muitos métodos de busca padrão que não se encaixam na definição de busca direta, pois utilizam o valor numérico da função objetivo. Um exemplo é o método SID-PSM [7], que utiliza o valor da função objetivo para calcular derivadas simplex e modelos para a função. Porém a teoria de convergência para estes métodos é a mesma que para os métodos de busca direta (desde que satisfaça as condições impostas pela autora).




(a) Busca padrão com sucesso até encontrar o ponto $x_2$; neste ponto a busca falha.




(b) Reduzindo o tamanho do passo pela metade a busca coordenada continua e encontra $x_3$.

Figura 1.2: Busca coordenada com ponto inicial $x_0 = (-4, 4)^\top$, $\alpha_0 = 4$ e padrão de pesquisa dado por $\{-e_1, e_2, e_1, -e_2\}$

# Capítulo 2

# Derivadas Simplex

Alguns métodos da classe *derivative-free* baseiam-se em modelos de interpolação para a função objetivo. Muitos tipos de modelo podem ser utilizados, mas, em geral, usam-se modelos de interpolação polinomial. Sob algumas hipóteses os coeficientes desses polinômios podem ser vistos como aproximações para as derivadas da função original.

## 2.1 Gradiente Simplex

Sejam $S = \{y^0, y^1, \ldots, y^n\}$ um simplex não degenerado e $f : I\!R^n \to I\!R$.

O gradiente simplex de $f$ baseado em $y^0$, calculado por $S$, é definido como a solução do sistema

$$L(S)^\top \nabla_S f(y^0) = \delta f(S).$$

Caso $|S| > n+1$, o gradiente simplex é definido como a solução do problema

$$\min_{x \in I\!R^n} \|L(S)^\top x - \delta f(S)\|, \tag{2.1}$$

e caso $|S| < n+1$, o gradiente simplex é definido como a solução de norma mínima do problema (2.1).

No caso onde $|S| = n+1$, temos

$$(y^j - y^0)^\top \nabla_S f(y^0) = f(y^j) - f(y^0),$$

para todo $j = 0, 1, \ldots, n$. Se definirmos a função linear

$$m(x) = f(y^0) + \nabla_S^\top f(y^0)(x - y^0), \tag{2.2}$$

vemos que $m(y^j) = f(y^j)$, para todo $j = 0, 1, \ldots, n$, ou seja, $m$ é o polinômio de grau 1 que a interpolação linear da função $f$ em torno de $y^0$ (avaliada) sobre os pontos de $S$ nos dá. Vemos assim que os elementos do vetor $\nabla_S f(y^0)$ são os coeficientes da interpolação linear de $f$ em torno de $y^0$.

Uma amostra $S$ com $q$ elementos é dita posicionada (para o cálculo do gradiente simplex) quando a matriz

$$M = \begin{bmatrix} 1 & y_1^0 & \dots & y_n^0 \\ 1 & y_1^1 & \dots & y_n^1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & y_1^q & \dots & y_n^q \end{bmatrix},$$

que define o sistema linear da interpolação, tem posto coluna completo. O fato da amostra ser posicionada não é suficiente para nossas aplicações, precisamos de uma medida de bom-posicionamento. Para isso usaremos a ideia de $\Lambda$-posicionamento, introduzida por Conn, Scheinberg e Vicente em [4]. Nesta definição usa-se a decomposição em valor singular da matriz $\frac{L^\top}{\Delta}$.

**Definição 2** *Seja $A \in I\!R^{m\times n}$. A decomposição em valor singular de $A$ (SVD, do inglês Singular Value Decomposition) é a fatoração*

$$A = UDV^\top,$$

*onde $U \in I\!R^{m\times m}$ e $V \in I\!R^{n\times n}$ são matrizes ortogonais, e $D \in I\!R^{m\times n}$ é uma matriz diagonal com $d_{11} \geq d_{22} \geq \ldots \geq d_{pp} \geq 0$, onde $p = \min\{m, n\}$.*

**Definição 3** *Se $A = UDV^\top \in I\!R^{m\times n}$ é a decomposição SVD de $A$, com posto$(A) = p$, então a decomposição SVD reduzida de $A$ é a fatoração*

$$A = \hat{U}\hat{D}\hat{V}^\top,$$

*onde $\hat{U} = [u^1, u^2, \ldots, u^p]$, $\hat{V} = [v^1, v^2, \ldots, v^p]$ e $\hat{D} \in I\!R^{p\times p}$ é um matriz diagonal, com $\hat{d}_{11} = d_{11}$, $\hat{d}_{22} = d_{22}$, $\ldots$, $\hat{d}_{pp} = d_{pp}$.*

Pela ortogonalidade de $U$ e $V$ pode-se demonstrar que $\hat{U}^\top\hat{U} = I$ e $\hat{V}^\top\hat{V} = I$ na decomposição SVD reduzida.

**Definição 4** *Seja $\Lambda > 0$ dado. Seja $\phi = \{\phi_0(x)\,, \phi_1(x)\,, \ldots\,, \phi_p(x)\}$ uma base para $\mathcal{P}_n^d$ (polinômios do $I\!R^n$ com grau menor ou igual a d).*

*Um conjunto $Y = \{y^0\,, y^1\,, \ldots\,, y^p\}$ é dito $\Lambda$-posicionado, para interpolação linear, na bola unitária centrada em 0 se, e somente se, para qualquer $x \in B(1, 0)$ existe $\lambda(x) \in I\!R^p$ tal que*

$$\sum_{i=0}^{p} \lambda_i(x)\phi(y^i) = \phi(x) \qquad \text{com} \qquad \|\lambda(x)\| \leq \Lambda.$$

Ainda em [4] os autores demonstram o seguinte teorema:

**Teorema 8** *Seja uma amostra $S \subset I\!R^n$ de modo que $|S| = p$, e tomemos $D$ matriz diagonal proveniente da decomposição SVD reduzida da matriz $\frac{L^\top}{\Delta}$.*

*Se* $\|D^{-1}\| \leq \Lambda$, *então o conjunto* $S$ *é* $\left(\sqrt{p}\Lambda\right)$*-posicionado na bola unitária centrada em* $0$. *Reciprocramente, se o conjunto* $S$ *é* $\Lambda$*-posicionado na bola unitária centrada em* $0$, *então* $D$ *é não singular e*

$$\|D^{-1}\| \leq \theta\Lambda,$$

*onde* $\theta > 0$ *depende de* $n$, *mas não depende de* $S$ *e* $\Lambda$.

Em particular, a segunda parte do teorema é válida com $\theta = 1$. Assim consideraremos, neste trabalho, que uma amostra é $\Lambda$-posicionada se $\|D^{-1}\| \leq \Lambda$, onde $D$ é matriz diagonal proveniente da decomposição SVD reduzida da matriz $\frac{L^\top}{\Delta}$.

Nas funções com gradiente Lipschitz contínuo, o gradiente simplex pode ser visto como uma aproximação para o vetor gradiente, contudo essa aproximação não é vista como uma aproximação local, dado que sua construção se é feita a partir de pontos de interpolação. A seguir apresentamos uma demonstração detalhada para esta propriedade, a demonstração foi baseada na apresentada em [6] porém apresentamos alguns detalhes omitidos na original.

**Teorema 9** *Seja* $S = \{y^0\,, y^1\,, \ldots\, y^q\}$ *uma amostra* $\Lambda$*-posicionada para o cálculo de um gradiente simplex em* $\mathbb{R}^n$. *Considere a bola fechada* $B(y^0, \Delta)$, *onde* $\Delta = \max_{j=1,\ldots,q} \|y^j - y^0\|$. *Seja* $L = [y^1 - y^0\,, \ldots\,, y^q - y^0]$ *e* $UDV^\top$ *a decomposição SVD reduzida de* $\frac{L^\top}{\Delta}$.

*Se* $\nabla f$ *for Lipschitz contínuo em um conjunto aberto* $\Omega \supset B(y^0, \Delta)$, *com constante* $\gamma > 0$, *então o erro do gradiente simplex em* $y^0$, *como uma aproximação para* $\nabla f(y^0)$, *satisfaz*

$$\|\hat{V}^\top[\nabla f(y^0) - \nabla_S f(y^0)]\| \leq \left(\sqrt{q}\frac{\gamma}{2}\Lambda\right)\Delta,$$

*onde* $\hat{V} = I$ *se* $q \geq n$ *e* $\hat{V} = V$ *se* $q < n$.

Demonstração: Pela definição do gradiente simplex temos

$$(y^i - y^0)^\top \nabla_S f(y^0) = f(y^i) - f(y^0). \tag{2.3}$$

O teorema do valor médio na forma integral nos diz que

$$f(y^i) - f(y^0) = \int_0^1 (y^i - y^0)^\top \nabla f(y^0 + t(y^i - y^0))dt. \tag{2.4}$$

Sabemos que

$$(y^i - y^0)^\top(\nabla f(y^0) - \nabla_S f(y^0)) = (y^i - y^0)^\top \nabla f(y^0) - (y^i - y^0)^\top \nabla_S f(y^0);$$

usando (2.3) obtemos

$$(y^i - y^0)^\top(\nabla f(y^0) - \nabla_S f(y^0)) = (y^i - y^0)^\top \nabla f(y^0) - f(y^i) + f(y^0).$$

Usando (2.4) temos

$$\begin{gathered}(y^i-y^0)^\top\nabla f(y^0)-f(y^i)+f(y^0)=\\ =(y^i-y^0)^\top\nabla f(y^0)-\int_0^1(y^i-y^0)^\top\nabla f(y^0+t(y^i-y^0))dt\leq\\ \leq|\int_0^1(y^i-y^0)^\top\nabla f(y^0+t(y^i-y^0))dt-(y^i-y^0)^\top\nabla f(y^0)|\leq\\ \leq\int_0^1|(y^i-y^0)^\top\nabla f(y^0+t(y^i-y^0))-(y^i-y^0)^\top\nabla f(y^0)|dt\leq\\ \leq\int_0^1\|(y^i-y^0)\|\,\|\nabla f(y^0+t(y^i-y^0))-\nabla f(y^0)\|dt.\end{gathered}$$

Do fato que $\nabla f$ é Lipschitz contínuo

$$\begin{gathered}\int_0^1\|(y^i-y^0)\|\,\|\nabla f(y^0+t(y^i-y^0))-\nabla f(y^0)\|dt\leq\\ \leq\|(y^i-y^0)\|\int_0^1\gamma\|y^0+t(y^i-y^0)-y^0\|dt=\\ =\|(y^i-y^0)\|\int_0^1\gamma t\|(y^i-y^0)\|dt=\|(y^i-y^0)\|^2\,\frac{\gamma}{2}.\end{gathered}$$

Assim mostramos que

$$(y^i-y^0)^\top(\nabla f(y^0)-\nabla_S f(y^0))\leq\frac{\gamma}{2}\|y^i-y^0\|^2,\quad\forall\, i=1,\ldots,q.$$

Como

$$\begin{gathered}\|L^\top(\nabla f(y^0)-\nabla_S f(y^0))\|\leq\sqrt{q}\|L^\top(\nabla f(y^0)-\nabla_S f(y^0))\|_\infty=\\ =\sqrt{q}\max_{i=1,\ldots,q}(y^i-y^0)^\top(\nabla f(y^0)-\nabla_S f(y^0))\leq\\ \leq\sqrt{q}\max_{i=1,\ldots,q}\|y^i-y^0\|^2\frac{\gamma}{2},\end{gathered}$$

então

$$\|L^\top(\nabla f(y^0)-\nabla_S f(y^0))\|\leq\frac{\gamma}{2}\sqrt{q}\Delta^2.$$

Multiplicando os dois lados da desigualdade por $\frac{1}{\Delta}$ e substituindo $\frac{L^\top}{\Delta}$ por sua decomposição SVD reduzida temos

$$\|UDV^\top(\nabla f(y^0)-\nabla_S f(y^0))\|\leq\frac{\gamma}{2}\sqrt{q}\Delta.\tag{2.5}$$

Estamos trabalhando sempre com amostras posicionadas, então $L$ tem sempre posto completo. Consideraremos dois casos:

1. $q\geq n$: neste caso, $U\in I\!R^{q\times n}$, $D\in I\!R^{n\times n}$, $V\in I\!R^{n\times n}$, $V$ é ortogonal e $U^\top U=I$;

$$\begin{gathered}\|U^\top UDV^\top(\nabla f(y^0)-\nabla_S f(y^0))\|\leq\frac{\gamma}{2}\sqrt{q}\Delta\Rightarrow\\ \Rightarrow\|DV^\top(\nabla f(y^0)-\nabla_S f(y^0))\|\leq\frac{\gamma}{2}\sqrt{q}\Delta,\end{gathered}$$

mas

$$\|D^{-1}DV^\top(\nabla f(y^0)-\nabla_S f(y^0))\| \leq \|D^{-1}\|\,\|DV^\top(\nabla f(y^0)-\nabla_S f(y^0))\| \leq \frac{\gamma}{2}\sqrt{q}\|D^{-1}\|\Delta,$$

daí,

$$\|V^\top(\nabla f(y^0) - \nabla_S f(y^0))\| \leq \frac{\gamma}{2}\sqrt{q}\|D^{-1}\|\Delta,$$

e como $V$ é ortogonal temos

$$\|\nabla f(y^0) - \nabla_S f(y^0\| \leq \frac{\gamma}{2}\sqrt{q}\|D^{-1}\|\Delta \leq \left(\sqrt{q}\frac{\gamma}{2}\Lambda\right)\Delta.$$

2. $q < n$: neste caso $U \in I\!R^{q\times q}$, $D \in I\!R^{q\times q}$, $V \in I\!R^{q\times n}$ e $U$ é ortogonal, assim a desigualdade (2.5) pode ser reescrita como

$$\|DV^\top(\nabla f(y^0) - \nabla_S f(y^0))\| \leq \frac{\gamma}{2}\sqrt{q}\Delta.$$

Como

$$\|D^{-1}DV^\top(\nabla f(y^0)-\nabla_S f(y^0))\| \leq \|D^{-1}\|\,\|DV^\top(\nabla f(y^0)-\nabla_S f(y^0)\| \leq \frac{\gamma}{2}\sqrt{q}\|D^{-1}\|\Delta,$$

concluímos que

$$\|V^\top(\nabla f(y^0) - \nabla_S f(y^0))\| \leq \frac{\gamma}{2}\sqrt{q}\|D^{-1}\|\Delta \leq \left(\sqrt{q}\frac{\gamma}{2}\Lambda\right)\Delta,$$

completando a demonstração. ■

Neste teorema vemos que o erro é projetado sobre $N\left(\frac{L^\top}{\Delta}\right)$; se $q < n$, então $N\left(\frac{L^\top}{\Delta}\right) \neq \emptyset$ e, portanto, não temos garantia sobre a precisão do gradiente simplex. Porém, mesmo no caso indeterminado, quando $q \approx n$, o gradiente simplex ainda pode nos fornecer informações relevantes sobre o gradiente.

Outra característica do gradiente simplex que usaremos é que, sob certas hipóteses, este é uma $\epsilon$-aproximação para as componentes grandes do gradiente, apresentaremos aqui a definição de $\epsilon$-aproximação seguida da demonstração da propriedade anterior como foi apresentada em [1].

**Definição 5** *Seja $g \in I\!R^n$ não nulo e $\epsilon \geq 0$. Defina $J^\epsilon(g) = \{i \in \{1,\ldots,n\} | |g_i| + \epsilon \geq \|g\|_\infty\}$, e para todo $i \in \{1,\ldots,n\}$ defina*

$$d_i^\epsilon(g) = \begin{cases} sign(g_i) & se \quad i \in J^\epsilon(g) \\ 0 & c.c. \end{cases}$$

*O vetor $g$ é uma $\epsilon$-aproximação para as componentes grandes de um vetor não nulo $v \in I\!R^n$ se, e somente se, $i \in J^\epsilon(g)$ sempre que $|v_i| = \|v\|_\infty$ e se $sign(g_i) = sign(v_i)$ para todo $i \in J^\epsilon(g)$.*

**Teorema 10** *Seja $B(x,\Delta)$ a bola centrada em $x$ com raio*

$$\Delta = \sigma\alpha \max_{b\in\bar{B}} \|b\|,$$

*onde $\sigma > 0$ e $\bar{B}$ é uma base positiva para o $\mathbb{R}^n$. Tomemos $S \subset B(x,\Delta)$ uma amostra $\Lambda$-posicionada. Assuma que $f$ é continuamente diferenciável em um conjunto aberto $\Omega$ contendo $B(x,\Delta)$, e que $\nabla f$ seja Lipschitz contínuo em $\Omega$ com constante $\gamma > 0$.*

*Então, se*

$$\alpha \leq \frac{\|\nabla f(x)\|_\infty}{\sqrt{q}\gamma\Lambda\sigma \max_{b\in\bar{B}} \|b\|}, \tag{2.6}$$

*o gradiente simplex negativo $-\nabla_S f(x)$ é uma $\epsilon$-aproximação para as componentes grandes de $-\nabla f(x)$, onde*

$$\epsilon = \left(\sqrt{q}\gamma\Lambda\sigma \max_{b\in\bar{B}} \|b\|\right)\alpha.$$

Demonstração: Para $i$ no conjunto

$$I = \{i \in \{1,\ldots,n\} | |\nabla f(x)_i| = \|\nabla f(x)\|_\infty\},$$

do Teorema 9 temos

$$\begin{aligned}
\|\nabla_S f(x)\|_\infty &= \|\nabla f(x) - \nabla f(x) + \nabla_S f(x)\|_\infty \\
&\leq \|\nabla f(x)\|_\infty + \|\nabla f(x) - \nabla_S f(x)\|_\infty \\
&= \|\nabla f(x) - \nabla_S f(x)\|_\infty + |\nabla f(x)_i| \\
&\leq 2\|\nabla f(x) - \nabla_S f(x)\| + |\nabla f(x)_i| \\
&\leq \sqrt{q}\gamma\Lambda\Delta + |\nabla f(x)_i| \\
&= \epsilon + |\nabla f(x)_i|.
\end{aligned}$$

Do Teorema 9 também obtemos que

$$-\nabla_S f(x)_i = -\nabla f(x)_i + \xi_i, \text{ onde } |\xi_i| \leq \sqrt{q}\frac{\gamma}{2}\Lambda\Delta.$$

Se $-\nabla f(x)_i$ e $\xi_i$ têm o mesmo sinal, então $\nabla f(x)_i$ e $\nabla_S f(x)_i$ também têm. Caso contrário, eles têm o mesmo sinal se

$$|\xi_i| \leq \frac{1}{2}\|\nabla f(x)\|_\infty = \frac{1}{2}|\nabla_S f(x)_i|.$$

Mas $\Delta = \sigma\alpha \max_{b\in\bar{B}} \|b\|$, e $\alpha \leq \frac{\|\nabla f(x)\|_\infty}{\sqrt{q}\gamma\Lambda\sigma \max_{b\in\bar{B}} \|b\|}$, assim obtemos

$$\begin{aligned}
|\xi_i| &\leq \tfrac{1}{2}\left(\sqrt{q}\gamma\Lambda\sigma \max_{b\in\bar{B}} \|b\|\right)\alpha \\
&\leq \tfrac{1}{2}\left(\sqrt{q}\gamma\Lambda\sigma \max_{b\in\bar{B}} \|b\|\right)\frac{\|\nabla f(x)\|_\infty}{\sqrt{q}\gamma\Lambda\sigma \max_{b\in\bar{B}} \|b\|} \\
&= \tfrac{1}{2}\|\nabla f(x)\|_\infty = \tfrac{1}{2}|\nabla_S f(x)_i|,
\end{aligned}$$

logo $\nabla f(x)_i$ e $\nabla_S f(x)_i$ têm o mesmo sinal, concluindo a demonstração. ■

## 2.2 Hessiana Simplex

Podemos tomar outra classe de modelos para $f$, como por exemplo modelos de interpolação quadrática, ou seja, criaremos uma função

$$m(x) = f(y^0) + (x - y^0)^\top \nabla_S f(y^0) + \frac{1}{2}(x - y^0)^\top \nabla_S^2 f(y^0)(x - y^0), \tag{2.7}$$

tal que para todo $j = 0, 1, \ldots, q$, $m(y^j) = f(y^j)$.

A matriz $\nabla_S^2 f(y^0)$ é chamada Hessiana simplex [6]. Supondo que $f$ é duas vezes diferenciável e $\nabla^2 f$ é Lipschitz contínua, então esta é uma matriz simétrica, logo, podemos impor que $\nabla_S^2 f(y^0)$ também seja, daí a Hessiana simplex está unicamente determinada quando nossa amostra $S$ possui $(n+1)(n+2)/2$ pontos.

**Definição 6** *Seja $Y = \{y^0, \ldots, y^q\}$ e $\Phi = \{\Phi_0, \ldots, \Phi_p\}$ uma base para $\mathcal{P}_n^2$. Se a matriz*

$$M(Y, \Phi) = \begin{bmatrix} \Phi_0(y^0) & \Phi_1(y^0) & \ldots & \Phi_p(y^0) \\ \Phi_0(y^1) & \Phi_1(y^1) & \ldots & \Phi_p(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \Phi_0(y^q) & \Phi_1(y^q) & \ldots & \Phi_p(y^q) \end{bmatrix},$$

*que define o sistema linear da interpolação quadrática, for não singular, então o conjunto $Y$ é dito posicionado (no sentido de interpolação quadrática).*

Em [6] os autores demonstram o seguinte teorema:

**Teorema 11** *Seja $Y = \{y^0, \ldots, y^q\}$, com $q + 1 = (n+1)(n+2)/2$, seja um conjunto posicionado (no sentido de interpolação quadrática) contido na bola $B(y^0, \Delta)$, onde $\Delta = \max_{i=1\ldots q} \|y^0 - y^i\|$.*

*Suponhamos que $f$ seja duas vezes diferenciável em um conjunto aberto $\Omega \supset B(y^0, \Delta)$ e $\nabla^2 f$ seja Lipschitz contínua em $\Omega$ com constante $\gamma > 0$.*

*Então:*

- *o erro entre a Hessiana simplex e a Hessiana da função satisfaz*

$$\|\nabla^2 f(y^0) - \nabla_S^2 f(y^0)\|_F \leq \kappa_{eh} \Delta;$$

- *o erro entre o gradiente simplex e o gradiente da função satisfaz*

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq \kappa_{eg} \Delta^2,$$

*onde $\kappa_{eh}$ e $\kappa_{eh}$ dependem de $\gamma$ e da geometria do conjunto $Y$.*

Quando temos menos de $(n+1)(n+2)/2$ pontos, nosso modelo fica indeterminado. Para este caso, em [6], é demonstrado o seguinte teorema:

**Teorema 12** *Sejam $Y = \{y^0, \ldots, y^q\}$, com $q+1 < (n+1)(n+2)/2$ e $f$ uma função continuamente diferenciável em um conjunto aberto na bola $B(y^0, \Delta)$, onde $\Delta = \max_{i=1\ldots q} \|y^0 - y^i\|$, com o gradiente Lipschitz contínuo na bola $B(y^0, \Delta)$. Se $Y$ for $\Lambda$-posicionado para interpolação linear então*

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq C_q \Lambda [\gamma + \|\nabla_S^2 f(y^0)\|_F] \Delta,$$

*onde $C_q > 0$ depende de $q$ e $\gamma$ é a constante de Lipschitz do gradiente da função.*

Uma escolha para $\nabla_S^2 f(y^0)$, sugerida em [5], é considerar a indeterminação apenas na matriz $\nabla_S^2 f(x_k)$, e escolhê-la de forma a minimizar a sua norma de Frobenius, ou seja, para encontrar o modelo resolvemos o problema:

$$\begin{array}{ll} \min\limits_{H \in \mathbb{R}^{n\times n}} & \frac{1}{4}\|H\|_F \\ \text{s.a :} & H = H^\top \\ & m_H(y) = f(y) \quad \forall \quad y \in Y, \end{array} \tag{2.8}$$

onde $m_H$ é a mesma função $m$ da equação (2.7) com $\nabla_S^2 f(y^0) = H$.

Em [6], os autores utilizam o conceito de $\Lambda$-posicionamento no sentido da norma mínima de Frobenius e demonstram que o uso desta estratégia limita o tamanho da Hessiana simplex.

**Definição 7** *Sejam $\Lambda > 0$ e um conjunto $B \in \mathbb{R}^n$ dados. Seja $\Phi = \{\phi_0(x)\,,\, \phi_1(x)\,,\, \ldots\,,\, \phi_q(x)\}$ uma base para $\mathcal{P}_n^2$ (polinômios do $\mathbb{R}^n$ com grau menor ou igual a 2).*

*Tomemos o problema*

$$\begin{array}{ll} \min & \|M(\Phi_Q, Y)^\top \lambda(x) - \Phi_Q(x)\|^2 \\ \text{s.a} & M(\Phi_L, Y)^\top \lambda(x) = \Phi_L(x), \end{array} \tag{2.9}$$

*onde $\Phi_L$ e $\Phi_Q$ são, respectivamente, as partes linear e quadrática de $\Phi$ e $M(\Phi, Y)$ é a matriz que define o sistema linear da interpolação nos pontos em $Y$.*

*Um conjunto $Y = \{y^0\,,\, y^1\,,\, \ldots\,,\, y^p\}$ é dito $\Lambda$-posicionado em $B$, no sentido de norma mínima de Frobeinus, se, e somente se, para qualquer $x \in B$, a solução $\lambda(x) \in \mathbb{R}^p$ do problema (2.9) é tal que*

$$\|\lambda(x)\|_\infty \leq \Lambda.$$

Resolver o problema (2.9) implica em resolver a parte linear da interpolação satisfazendo a igualdade e passar a inderterminação do problema para a parte quadrática da interpolação.

**Teorema 13** *Sejam $Y = \{y^0, \ldots, y^q\}$, com $q+1 < (n+1)(n+2)/2$ e $f$ uma função continuamente diferenciável em um conjunto aberto na bola $B(y^0, \Delta)$, onde $\Delta = \max_{i=1\ldots q} \|y^0 - y^i\|$, com o gradiente Lipschitz contínuo na bola $B(y^0, \Delta)$. Se $Y$ for $\Lambda_F$-posicionado no sentido da norma mínima de Frobenius então*

$$\|\nabla_S^2 f(y^0)\|_F \leq C_{p,q} \gamma \Lambda_F.$$

*onde $C_{p,q}$ é uma constante que depende de $p$ e $q$, e $\gamma$ é a constante de Lipschitz do gradiente da função.*

Portanto o limitante do erro no Teorema 12 passa a ser dado por

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq C_q \Lambda \gamma [1 + C_{p,q} \Lambda_F] \Delta.$$

Assim concluímos que os modelos construídos pela norma mínima de Frobenius representam bem a aproximação de Taylor de primeira ordem de $f$.

# Capítulo 3

# Métodos de Busca Linear

Métodos de busca linear sem derivadas seguem a mesma linha dos métodos com derivadas. Dada uma aproximação $x_k$ para o minimizador, quando temos as derivadas escolhemos uma direção de descida $d_k$ e definimos uma função $\phi : \mathbb{R}_+ \to \mathbb{R}$ tal que $\phi(\alpha) = f(x_k + \alpha d_k)$. A idéia é encontrar $\alpha_k$ tal que

$$f(x_k + \alpha_k d_k) < f(x_k), \tag{3.1}$$

se procuramos decréscimo simples, ou

$$f(x_k + \alpha_k d_k) < f(x_k) + \alpha_k \epsilon d_k^\top \nabla f(x_k), \tag{3.2}$$

com $\epsilon \in (0, 1)$, se exigirmos o decréscimo suficiente de Armijo [3].

Uma forma de encontrar $\alpha_k$ que satisfaça uma das duas condições é o *backtracking*, onde avaliamos a função $\phi(\alpha)$ construída a partir de uma direção de descida, começando com um $\alpha_{inicial}$ "grande" e vamos reduzindo-o até que a condição de decréscimo exigida seja satisfeita.

Quando não dispomos das derivadas não podemos testar se a condição (3.2) é satisfeita, dado que esta utiliza o gradiente da função. Porém se substituirmos o gradiente da função pelo gradiente simplex da função, não estaremos fazendo uso de derivadas e ainda exigiremos decréscimo suficiente [6].

Outra dificuldade encontrada quando não utilizamos derivadas é que não podemos testar se a direção $d_k$ é de descida, logo devemos ter alguma base teórica que nos dê uma direção de descida em potencial.

Neste caso definimos $\alpha_{final} < \alpha_{inicial}$, e realizamos o *backtracking*. Se atingirmos $\alpha \leq \alpha_{final}$ sem obter o decréscimo desejado, declaramos fracasso.

Uma direção de descida em potencial é dada por $-\nabla_S f(x_k)$. Exemplos de método que utilizam derivadas simplex como direção de busca são apresentados em [6] e [11].

## 3.1 Métodos de Busca Linear Baseados em Gradientes Simplex

Neste método consideraremos que a cada iteração temos um simplex $S^k = \{y_k^0, y_k^1, \ldots, y_k^n\}$. Vamos assumir que $S_k$ é posicionado no sentido de interpolação linear, ou seja, $L(S^k) = [y_k^1 - y_k^0, \ldots, y_k^n - y_k^0]$ é não singular.

O método apresentado em [6] é um método de busca linear, e consiste em, a cada iteração, dada uma aproximação $x_k$ para o minimizador, um inteiro $j_{max} > 0$ e $\mu \in (0, 1]$, atualizar a amostra até que possamos calcular o gradiente simplex desta satisfazendo $y_k^0 = x_k$ e

$$\Delta_k \leq \|\nabla_S f(x_k)\|.$$

Definindo $d_k = -\nabla_S f(x_k)$, fazemos uma busca linear utilizando processo de *backtracking*, avaliamos no máximo $j_{max}$ vezes $f(x_k + \alpha_k^j d_k)$, de modo que $\alpha_k^{j+1} < \alpha_k^j$ para todo $j = 1, \ldots, j_{max}$.

Se o processo de busca linear falhar devemos fazer $\mu = \frac{1}{2}\mu$, $j_{max} = j_{max} + 1$, calcular o gradiente simplex de uma amostra na qual $y_k^0 = x_k$ que satisfaça

$$\Delta_k \leq \mu \|\nabla_S f(x_k)\|,$$

definir $d_k = -\nabla_S f(x_k)$ e repetir a busca linear.

Neste método, durante o cálculo do gradiente simplex, podemos avaliar a função objetivo em pontos que nos dão algum decréscimo. Assim, a aproximação para o minimizador na iteração $k + 1$ será o ponto que oferece menor valor de função objetivo avaliado na iteração $k$. Com a condição de que $f$ seja limitada inferiormente no conjunto de nível $\mathcal{L}(x_0)$, seja continuamente diferenciável, e $\nabla f$ seja Lipschitz contínuo em um conjunto aberto contendo $\mathcal{L}(x_0)$ e os pontos avaliados pelo método, em [6] é demonstrado que este converge a pontos estacionários, ou seja,

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

## 3.2 Método do Filtro Implícito

O método do filtro implícito [11] também é um método de busca linear guiado por derivadas simplex. Este método utiliza estratégias quase-Newton para encontrar aproximações da Hessiana no ponto atual [3]. Da mesma forma que o método anterior, neste temos que calcular um gradiente simplex que satisfaça

$$\Delta_k \leq \|\nabla_S f(x_k)\|,$$

porém, ao invés de manter uma amostra e atualizá-la a cada iteração, este método cria uma nova a cada iteração.

Não é imposta nenhuma condição sobre a geometria do simplex, assim não é possível garantir o sucesso da busca linear, se esta falhar reinicia-se a aproximação para a Hessiana.

Este método vem sendo aplicado a funções com ruído, para as quais vem mostrando grande robustez numérica [6].

# Capítulo 4

# Método de busca padrão usando derivadas simplex

O método *SID-PSM* desenvolvido por Custódio e Vicente [7] tem como base os métodos de busca padrão apresentados no capítulo anterior. Porém não é um método exatamente de busca direta, uma vez que usamos o valor da função objetivo para resolver um sistema linear (ou um problema de quadrados mínimos). A resolução deste sistema é feita para encontrarmos alguma derivada simplex, que apesar de ser uma forma para aproximarmos a derivada da função, não é uma aproximação local, assim o método continua na classe *derivative-free*.

A cada iteração guardamos a informação dos pontos avaliados no final de uma lista $\mathcal{L}$, limitando o tamanho desta em no máximo $l_{max}$ pontos. Duas opções são oferecidas para incluir elementos na lista:

- armazenar somente informação dos pontos que fornecem decréscimo na função objetivo. Dessa forma, se $\mathcal{L}$ for implementada como uma fila, seus pontos estarão sempre ordenados pelo valor da função objetivo;

- armazenar todos os pontos onde a função objetivo foi avaliada, independentemente de haver ou não decréscimo. Nesse caso temos que garantir que o iterando atual não seja removido da lista quando a atualizamos.

Antes do passo de busca, se $\mathcal{L}$ possuir pelo menos $n_{min}$ pontos, procuramos o maior subconjunto de $\mathcal{L} \cap B(x_k, \Delta_k)$ (com no máximo $n_{max}$ pontos), que seja $\Lambda$-posicionado e contenha o ponto $x_k$, onde $x_k$ é o iterando atual,

$$\Delta_k = \sigma \alpha_k \max_{d \in D_k} \|d\|,$$

$n_{min} \leq n_{max}$ são inteiros positivos dados e $\sigma > 0$, ou seja, procuramos em $\mathcal{L}$ por elementos que distem no máximo $\Delta_k$ de $x_k$ e formem um conjunto $\Lambda$-posicionado. Se for possível encontrar esse conjunto, calculamos alguma derivada simplex em $x_k$ e definimos alguma direção de descida em potencial $d_p$. Um exemplo de direção de descida em potencial é $d_p = -\nabla_S f(x_k)$.

Usando as informações dos pontos armazenados na lista podemos construir um modelo de interpolação para a função $f$. Uma ideia bastante intuitiva para um ponto a ser testado no passo de busca é o minimizador desse modelo na bola $B(x_k, \Delta_k)$. Se o modelo construído for linear, seu minimizador na bola é dado por

$$x_k + \frac{\Delta_k}{\|d_p\|} d_p.$$

Para satisfazer as hipóteses de convergência precisamos garantir que todos os pontos avaliados na iteração $k$ estejam em uma malha inteira do tipo $M_k = \{x_k + \alpha_k D z : z \in \mathbb{Z}^n_+\}$, onde $D$ é um conjunto finito que contém as bases geradoras positivas utilizadas pelo método. Assim, no passo de busca, devemos projetar o minimizador do modelo na bola, no conjunto $M_k$.

Se o passo de busca fracassar, devemos ordenar os vetores do conjunto $D_k$, depois prosseguimos com o passo de pesquisa. Os autores fazem algumas sugestões de formas de ordenação:

- manter sempre a ordem inicial;
- ordenar de forma aleatória;
- se na iteração $k-1$ for declarado sucesso usando a direção $d^j_{k-1}$, definir

  $$D_k = \{d^j_{k-1}\,,\, d^{j+1}_{k-1}\,,\, \ldots\,,\, d^{|D|}_{k-1}\,,\, d^1_{k-1}\,,\, d^2_{k-1}\,,\, \ldots\,,\, d^{j-1}_{k-1}\}.$$

  Caso contrário, manter a ordenação anterior;
- caso exista alguma direção indicadora de descida, ordenar $D_k$ de modo que

  $$\cos(d^1_k, d_p) \;\geq\; \cos(d^2_k, d_p) \;\geq\; \ldots \;\geq\; \cos(d^{|D|}_k, d_p);$$

- alguma combinação dos itens anteriores.

O procedimento para escolher o passo de pesquisa e sua atualização é o mesmo de busca padrão. Se a iteração foi um fracasso, devemos diminuí-lo. Caso a iteração tenha sido um sucesso, devemos mantê-lo ou aumentá-lo. Algumas opções para a atualização são:

- aumentar o passo a todo sucesso;
- aumentar somente quando a mesma direção oferecer decréscimo em duas iteração sucessivas;
- criar um modelo $m_k$ utilizando as derivadas simplex e utilizar a ideia de decréscimo esperado, ou seja, definir

  $$\rho_k = \frac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})},$$

  e

– se $\rho_k > \gamma_2$, aumentar o passo;

– se $\gamma_1 < \rho_k \leq \gamma_2$, mantê-lo;

– se $\rho_k \leq \gamma_1$, diminuí-lo;

onde $\gamma_2 > \gamma_1 \geq 0$.

Para tentar diminuir o número de avaliações de função, Custódio e Vicente utilizam ideias apresentadas em [1]. Nesse artigo, os autores sugerem aplicar a ideia de poda no conjunto de direções de pesquisa para avaliarmos apenas direções de descida. Chamaremos de conjunto podado o conjunto resultante após a remoção das direções de subida.

**Definição 8** *Dado $D_k$ um conjunto gerador positivo para o $\mathbb{R}^n$, o conjunto de direções podado, denotado $D_k^p$, é dado por*

$$D_k^p = \{d \in D_k \,:\, d^\top \nabla f(x_k) \leq 0\}.$$

Pela definição, vemos que o tamanho do passo $\alpha_k$ não interfere no conjunto podado. Assim o conjunto de pesquisa podado é definido como

$$P_k^p = \{x_k + \alpha_k d \,:\, d \in D_k^p\}.$$

Apesar de não possuirmos o vetor $\nabla f$, podemos ter uma $\epsilon$-aproximação para este, por exemplo, o gradiente simplex. Ainda no artigo [1] por Abramson, Audet e Dennis é demonstrado que uma $\epsilon$-aproximação do gradiente poda o conjunto de direções de pesquisa para um conjunto unitário quando consideramos

$$D = \{-1, 0, 1\}^n,$$

e o conjunto gerador positivo

$$D_k = \{d^\epsilon(g_k)\} \cup \mathbb{A}(-\nabla f(x_k)),$$

onde $g_k$ é uma $\epsilon$-aproximação para $-\nabla f(x_k)$ e

$$\mathbb{A}(-\nabla f(x_k)) = \{d \in D : -\nabla f(x_k)^t d < 0\}$$

representa o conjunto de direções de subida em $D$. O único vetor de descida em $D_k$ é $d^\epsilon(g_k)$, assim o conjunto $D_k$ é podado para $\{d^\epsilon(g_k)\}$.

Sob as hipóteses do Teorema 10 do Capítulo 2, $-\nabla_S f(x_k)$ poda o conjunto gerador positivo do $\mathbb{R}^n$

$$D_k = \{d^{\epsilon_k}(-\nabla_S f(x_k))\} \cup \mathbb{A}(-\nabla f(x_k)),$$

para o conjunto unitário $\{d^{\epsilon_k}(-\nabla_S f(x_k))\}$, onde $\epsilon_k$ é dado pelo Teorema 10.

Não temos garantia de que a condição (2.6) seja satisfeita assintoticamente. Esta condição apenas nos indica o efeito da poda usando $-\nabla_S f$, e é mais provável que seja satisfeita em pontos onde o gradiente é relativamente grande.

Assim podemos resumir o método SID-PSM:

Algoritmo **1** *Método SID-PSM*

- *Inicialização:*

  *Dados $x_0 \in \mathbb{R}^n$, $\alpha_0 \in \mathbb{R}$ positivo e todas as constantes necessárias para a pesquisa, ordenação e atualização do passo, defina $k = 0$ e $\mathcal{L} = \{x_0\}$, a lista com os pontos já avaliados. Escolha o máximo de pontos a serem armazenados $l_{max}$, o máximo $n_{max}$ e o mínimo $n_{min}$ de pontos que serão utilizados nos cálculos das derivadas simplex. Escolha $\Lambda > 0$ e $\sigma > 0$.*

- *Identificação de conjunto $\Lambda$-posicionado e calculo de derivada simplex:*

  *Se $|\mathcal{L}| > n_{min}$, defina $\Delta_k = \sigma\alpha_k \max_{d \in D_k} \|d\|$. Procure um subconjunto $S_k$ de $\mathcal{L} \cap B(x_k; \Delta_k)$, com o maior número de pontos possível (até $n_{max}$), tal que $x_k \in S_k$ e $S_k$ seja $\Lambda$-posicionado. Se $|S_k| \geq n_{min}$ calcule uma derivada simplex baseado em $S_k$.*

- *Passo de busca:*

  *Esse passo é opcional; se alguma derivada simplex está disponível, calcule a direção $d_p$ que minimiza um modelo criado com esta derivada na bola $B(x_k; \Delta_k)$, e projete-a na malha $M_k$. Se esse ponto oferecer decréscimo na função objetivo, ele será o ponto $x_{k+1}$; declare sucesso e pule o passo de pesquisa (o ponto deve ou não ser armazenado na lista $\mathcal{L}$ dependendo da estratégia escolhida).*

- *Passo de pesquisa:*

  *Se a poda estiver sendo utilizada, encontre $D_k^p$; caso contrário, ordene os pontos utilizando a estratégia escolhida. Avalie os pontos no conjunto de pesquisa, armazenando na lista $\mathcal{L}$ conforme a estratégia adotada. Se encontrar decréscimo, esse será o ponto $x_{k+1}$; declare sucesso e termine a pesquisa. Caso contrário declare fracasso.*

- *Atualização do passo:*

  *Calcule o tamanho do passo $\alpha_{k+1}$ conforme a estratégia adotada, incremente $k$ em 1 e volte para o passo de cálculo das derivadas simplex.*

No próximo capítulo apresentaremos experimentos numéricos comparando a robustez e a eficiência (em termos do número de avaliações de função) do método SID-PSM com os métodos de busca padrão e Nelder-Mead.

# Capítulo 5

# Experimentos numéricos preliminares

Neste capítulo são apresentados resultados obtidos a partir de um conjunto de testes numéricos, extraído de [15], e algumas análises destes resultados. Este conjunto contém 35 problemas-teste de minimização irrestrita, onde a função objetivo é uma soma de quadrados. Os 20 primeiros têm dimensões entre 2 e 12. Os 15 últimos têm dimensão livre e foram resolvidos com dimensão 12 e 20. Implementamos o método de busca padrão no *software* `MatLab`. O comparamos com a função `fminsearch` do `MatLab`, que implementa o método de Nelder-Mead, e com o método SID-PSM, também implementado em `MatLab` disponível em [21].

Para fazer as comparações e analisar qual o melhor método, utilizamos a estratégia de perfil de desempenho apresentada em [10]. Esta estratégia compara $n_s$ métodos, de um conjunto $S$, para a resolução de $n_p$ problemas, de um conjunto $P$, usando medidas como número de iterações, número de avaliação de função ou tempo computacional. Definimos $m_{s,p}$ como o total da medida necessária para resolver o problema $p$ pelo método $s$. Para cada problema $p$ e método $s$, a taxa de desempenho $r_{s,p}$ é calculada:

$$r_{s,p} = \frac{m_{s,p}}{\min\{m_{s,p} \,:\, \forall s \in S\}}$$

se o problema $p$ é resolvido pelo método $s$; caso contrário

$$m_{s,p} = r_M,$$

onde $r_M$ é um parâmetro suficientemente grande.

Então, para cada $s \in S$, a função distribuição $\rho_s : I\!R \to [0,1]$ para a taxa de desempenho é construída:

$$\rho_s(t) = \frac{1}{n_p}|\{p \in P \,:\, r_{s,p} \leq t\}|.$$

Essa função representa o desempenho do método $s$, é não-decrescente e constante por partes. Na análise do método $s$, dois pontos nos dão informações muito importantes,

estes são: $\rho_s(1)$ e $\bar{t}$, tal que, $\rho_s(\bar{t}) = 1$. O valor de $\rho_s(1)$ representa a probabilidade do método $s$ ser o mais eficiente, em termos do conjunto $S$, usando a medida $m_{s,t}$. A robustez do método $s$ em termos do número de problemas que podem ser resolvidos é avaliada pelo menor valor de $t$, denotado $\bar{t}_s$, no qual $\rho_s(t) = 1$, se existe tal valor para $t < r_M$. Então, o melhor em termos de robustez será o método $\hat{s}$ para o qual $\bar{t}_{\hat{s}} = \min\{\bar{t}_s, \forall s \in S\}$.

No nosso caso usaremos como medida o número de avaliações de função realizadas por cada um dos métodos para resolver problemas apresentados em [15].

Antes de apresentar os resultados obtidos precisamos ainda definir o que é resolver os problemas.

**Definição 9** *Sejam $\theta_p(s)$ o menor valor de função obtido pelo método $s$, em um conjunto de métodos $S$, ao resolver o problema $p$ e $\hat{\theta}_p = \min_{\hat{s}\in S} \theta_p(\hat{s})$ . Então se o método atinge um passo com tamanho menor que uma tolerância $\epsilon_\alpha$ em menos de $10^6$ avaliações de função, e*

$$\frac{\theta_p(s) - \hat{\theta}_p}{\hat{\theta}_p} \leq 0,25 \quad \text{se } \hat{\theta}_p \neq 0,$$
$$\theta_p(s) \leq \texttt{eps}^2 \quad \text{se } \hat{\theta}_p = 0,$$

*onde `eps` é o épsilon da máquina, diz-se que o método $s$ resolve o problema $p$, em relação ao conjunto $S$.*

*Em outras palavras, dizemos que o método $s$ resolve o problema $p$ se a solução obtida por este é no máximo 25% pior que a melhor solução obtida.*

Fizemos alguns testes com o método SID-PSM para verificar quais parâmetros nos forneceriam um melhor desempenho, e depois comparamos o desempenho do método, usando esses parâmetros, com os métodos de busca padrão e Nelder-Mead. Em todos os testes os seguintes parâmetros foram mantidos fixos no SID-PSM:

- utilizamos o gradiente simplex definido, ou seja, procuramos um conjunto $\Lambda$-posicionado com $n+1$ elementos, desta forma $n_{min} = n_{max} = n + 1$;
- $\Lambda = 100$ e $\sigma = 2$;
- fazemos $\alpha_k = 2\alpha_{k-1}$ ao aumentar o tamanho do passo e $\alpha_k = \frac{1}{2}\alpha_{k-1}$ ao diminuir;
- realizamos o passo de busca construindo um modelo quadrático para a função objetivo com os pontos armazenados na lista. Quando temos menos de $(n+1)(n+2)/2$ pontos, resolvemos o problema (2.8) para encontrar o modelo. Quando temos entre $(n+1)(n+2)/2$ e $(n+1)(n+2)$ pontos, resolvemos o problema de regressão

  $$\min \frac{1}{2}\|M(\Phi, Y)\alpha - f(Y)\|^2,$$

  onde $\alpha$ nos dá os coeficientes do modelo quadrático criado. O ponto testado no passo de busca é calculado encontrando o minimizador do modelo na bola $B(x_k, \Delta_k)$;

- a direção de descida em potencial é dada por $d_p = -\nabla_S f(x_k)$;
- se temos a derivada simplex, ordenamos o conjunto de pesquisa pelo ângulo de seus vetores com a direção de descida em potencial, caso contrário, continuamos a pesquisa a partir do último vetor que ofereceu decréscimo;
- o método para se avalia mais de um milhão de vezes a função objetivo, ou quando o tamanho do passo é menor que $10^{-6}$.

## 5.1 Ajuste de parâmetros

Primeiramente comparamos as formas de armazenar os pontos na lista $L$: armazenar todos os pontos avaliados (SID-ALL), ou apenas os pontos que oferecem decréscimo da função objetivo (SID-SUC).

Utilizamos como conjunto de direções de pesquisa permutações do conjunto $D = \{-e, e_1, \ldots, e_n\}$ onde $e$ é o vetor unitário, e aumentamos o tamanho do passo em todo sucesso.

Em ambos os casos limitamos o tamanho da lista em $(n+1)(n+2)$ pontos.

Avaliamos os 20 primeiros problemas apresentados em [15]. Os resultados obtidos estão apresentados na tabela 1 no Anexo A. Com estes resultados fizemos o gráfico da Figura 5.1.

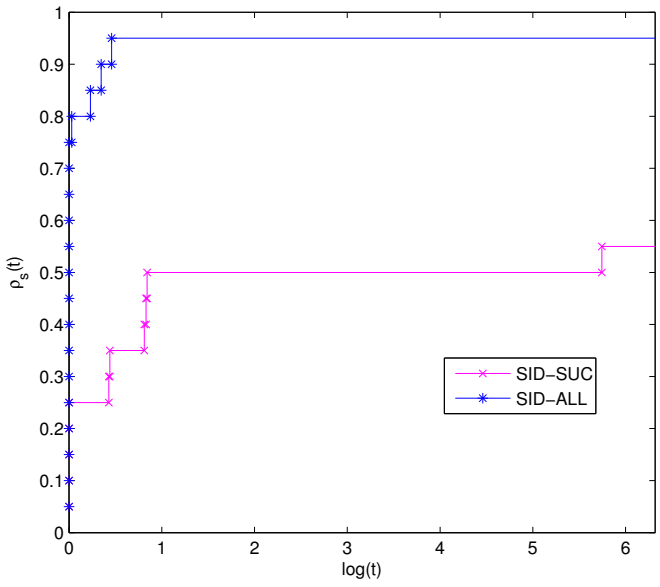



Figura 5.1: Perfil de desempenho dos métodos SID-SUC e SID-ALL.

Deste gráfico vemos claramente que o método SID-ALL é superior ao SID-SUC, tanto na eficiência quanto na robustez, resolvendo cerca de 75% dos problemas testados com o número mínimo de avaliações de função e 95% no total. Já o método SID-SUC

resolve apenas 25% dos problemas com o número mínimo de avaliações de função e no total apenas 55%. Podemos destacar que, para atingir esta robustez, o método SID-SUC realiza em torno de $2^5$ vezes o número de avaliações de função que o SID-ALL necessita para atingir a sua robustez máxima (o eixo $x$ está em escala logarítmica).

A partir disso decidimos utilizar em todas as versões do SID-PSM o armazenamento de todos os pontos.

Comparamos as quatro diferentes formas de definir os conjuntos de pesquisa sugeridas pelos autores do método. Estas são:

D0: permutações do conjunto $D = \{-e, e_1, \ldots, e_n\}$;

D1: permutações do conjunto $D = \{e_1, \ldots, e_n, -e_1, \ldots, -e_n\}$;

D2: permutações do conjunto $D = \{e, -e, e_1, \ldots, e_n, -e_1, \ldots, -e_n\}$;

D3: $D$ é uma base geradora positiva com ângulos de tamanho uniforme entre seus vetores[1], os conjuntos $D_k$'s são permutações de $D$.

Mantivemos a mesma forma de atualização do passo.

Os dados obtidos desses testes estão disponíveis nas tabelas 2 e 3 no Anexo A. Com estes dados fizemos o gráfico da Figura 5.2 que contem o perfil de desempenho para estes métodos.

Para facilitar a análise da eficiência, plotamos novamente estes dados na Figura 5.3, porém mudamos o limite do eixo $x$ para podermos verificar melhor como é a variação da eficiência dos métodos com poucas avaliações de função.

Por estes gráficos vemos que o método mais robusto é o D2, chegando a resolver 70% dos problemas testados, porém este é pouco eficiente, ficando empatado o D1 neste quesito, resolvendo apenas 25% dos problemas com o número mínimo de avaliações de função. O método D0 é o menos eficiente e também o menos robusto, já o D3 é o mais eficiente e, apesar de não ser muito robusto (resolvendo apenas 50% dos problemas testados), ele resolve 45% dos problemas com o menor número de avaliações de função objetivo.

Usando o conjunto de pesquisa dado por $D3$, analisamos o desempenho de 4 tipos de atualização do tamanho do passo, estes são:

P0: aumentar o tamanho do passo em todo sucesso;

P1: utilizar a ideia de decréscimo esperado permitindo que $\alpha_k$ diminua, com $\gamma_1 = \frac{1}{4}$ e $\gamma_2 = \frac{3}{4}$;

P2: utilizar a ideia de decréscimo esperado sem permitir que $\alpha_k$ diminua, neste caso $\gamma_1 = 0$ e $\gamma_2 = \frac{3}{4}$;

[1] ver Anexo B

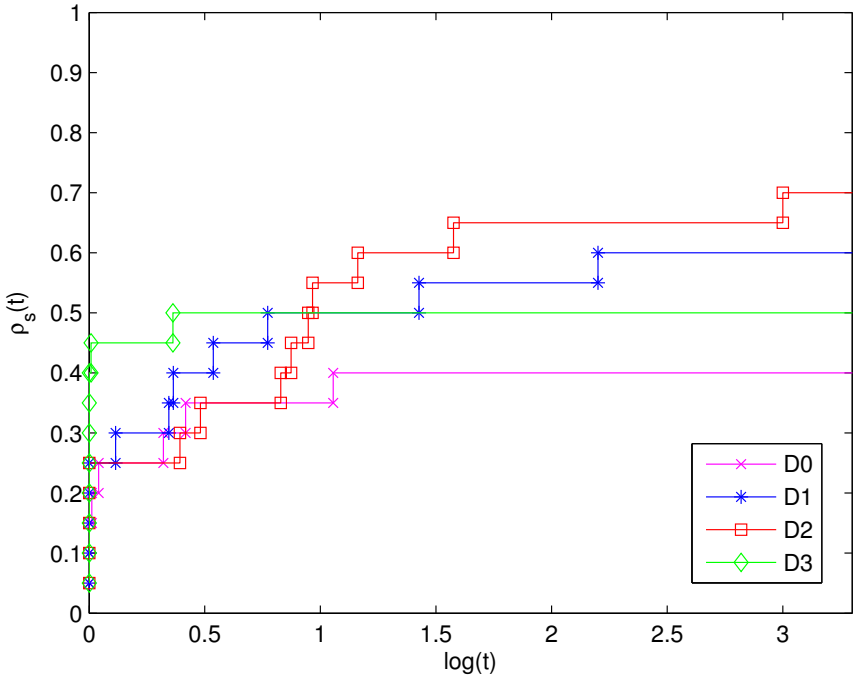



Figura 5.2: Perfil de desempenho do método SID-PSM utilizando os 4 tipos de bases disponíveis na implementação.

P3: aumentar o tamanho do passo somente quando uma mesma direção oferece decréscimo por duas iterações seguidas.

Os dados obtidos destes testes estão aprensentados nas tabelas 2 e 3 no Anexo A. Com estes dados fizemos o gráfico apresentado na Figura 5.4, com o perfil de desempenho para os métodos citados.

Novamente, para uma melhor análise da eficiência, plotamos o gráfico com um *zoom* no eixo $x$. Este novo gráfico está apresentado na Figura 5.5.

Destes gráficos vemos que a maior eficiência é obtida com P0, seguida por P1. A estratégia P2 foi a menos eficiente e menos robusta, sendo a maior robustez apresentada com P0, seguida por P1.

Comparamos, por fim, o uso ou não da poda. Testamos três possibilidades:

Poda 0: não utiliza a poda;

Poda 1: poda o conjunto $D_k$ e testa apenas a direção mais promissora;

Poda 2: poda o conjunto $D_k$ e testa todas as direções no conjunto podado.

Testamos duas combinações, a primeira utiliza D2 e P1 e a segunda utiliza D3 e P3. Os dados desses testes estão disponíveis nas tabelas 6, 7, 8 e 9 no Anexo A, com estes

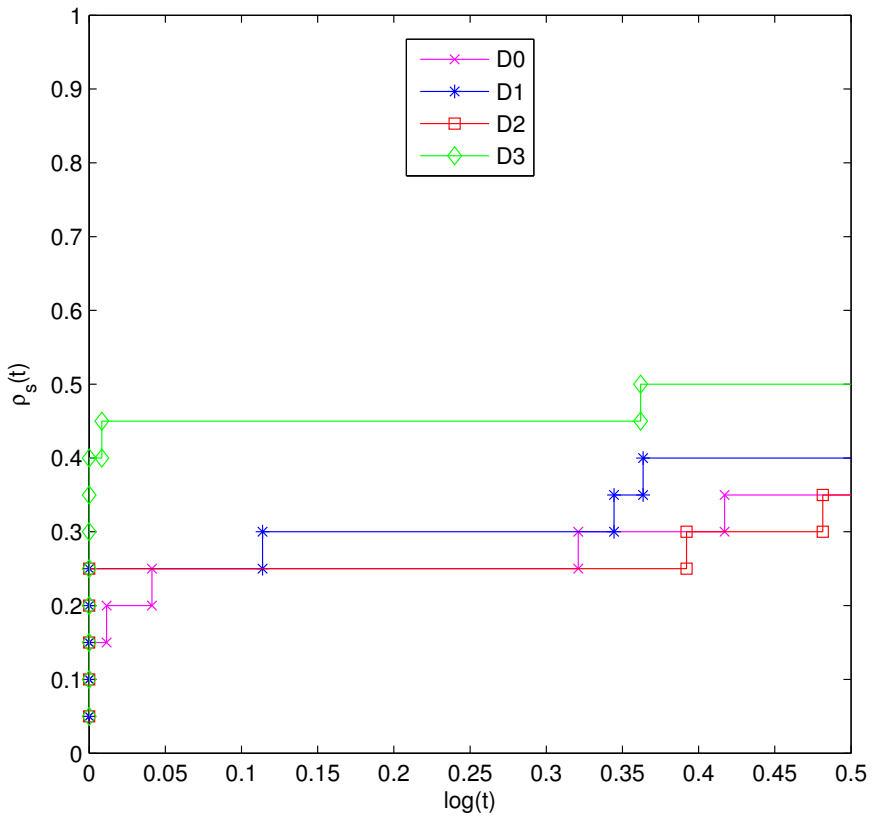



Figura 5.3: Perfil de desempenho do método SID-PSM utilizando os 4 tipos de bases disponíveis na implementação, com ênfase na eficiência dos métodos.

dados fizemos o gráfico apresentado na Figura 5.6 com o perfil de desempenho para este conjunto de métodos.

Da mesma forma que fizemos para os gráficos anteriores, construímos o gráfico na Figura 5.7 dando um *zoom* no eixo $x$ do gráfico da Figura 5.6.

Primeiramente vemos que, no geral, o uso de P1 D2 nos dá um método mais robusto que o uso de P3 D3. Já quando comparamos a eficiência dos métodos, aparentemente o uso de P3 D3 é melhor (apesar do método P1 D2 poda 1 ser o segundo mais eficiente). Quanto ao uso da poda, nada podemos concluir, dado que cada uma das versões apresentou resultados diferentes: o método P1 D2 poda 0 é o segundo em robustez (empatado com o P3 D3 poda 0) e o segundo menos eficiente, empatado com o método P3 D2 poda 1. Já o método P1 D2 poda 1 é o mais robusto e segundo em termos de eficiência e, ainda, o P1 D2 poda 2 fica em terceiro em robustez e é o segundo menos eficiente. O método P3 D3 poda 0 é o mais eficiente e está em segundo em robustez, já o P3 D3 poda 1 é o menos robusto e o terceiro mais eficiente, enquanto o P3 D3 poda 2 é o menos eficiente e o segundo pior em termos de robustez.

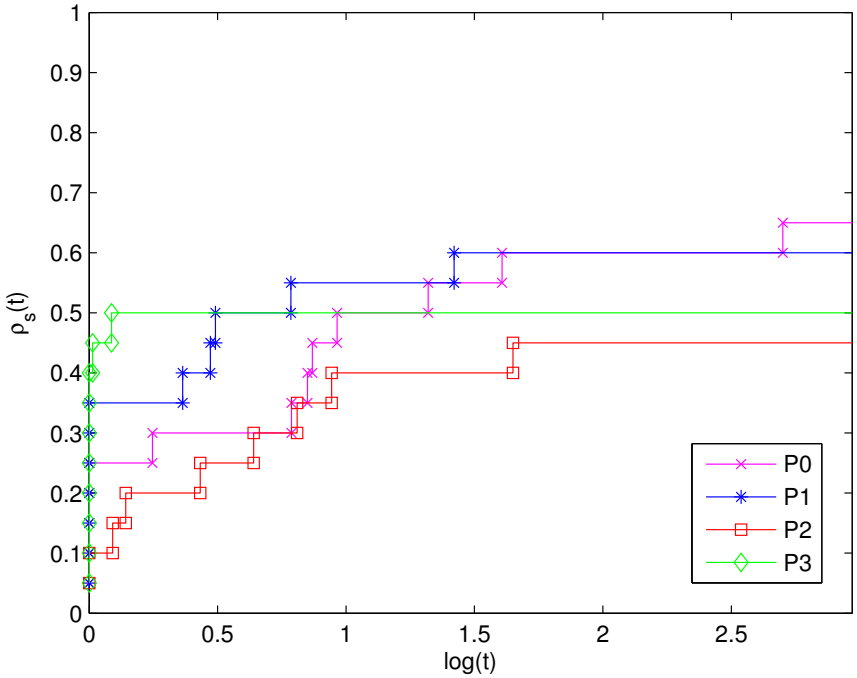



Figura 5.4: Perfil de desempenho do método SID-PSM utilizando os 4 tipos de atualização de passo disponíveis na implementação.

## 5.2 Comparação entre métodos

Com as análises feitas na seção anterior decidimos utilizar o método SID-PSM com os parâmetros P1 D2 poda 1, o qual chamamos de SID-PSM1, e P3 D3 poda 0, que chamamos de SID-PSM2.

Comparamos SID-PSM com o método Nelder-Mead e com o método de busca padrão sem o uso de derivadas simplex implementado de duas formas:

BP1: Os conjuntos de pesquisa são permutações do conjunto $D$, base positiva minimal com ângulos de tamanho uniforme entre seus vetores;

BP2: Os conjuntos de pesquisa são permutações do conjunto $D = \{e, -e, e_1, \ldots, e_n, -e_1, \ldots, -e_n\}$.

Para ambos os casos fizemos o passo aumentar sempre que uma mesma direção oferecesse decréscimo em duas iterações seguidas.

Utilizamos todos os testes apresentados em [15].

Os dados obtidos por esses métodos estão disponíveis nas tabelas 10, 11, 12 e 13 no Anexo A, com estes dados construímos o gráfico apresentado na Figura 5.8 com o perfil de desempenho dos métodos comparados.

Tanto o SID-PSM1 quanto o SID-PSM2 se destacam do restante dos métodos tanto na robustez quanto na eficiência, sendo que ambos possuem aproximadamente a mesma

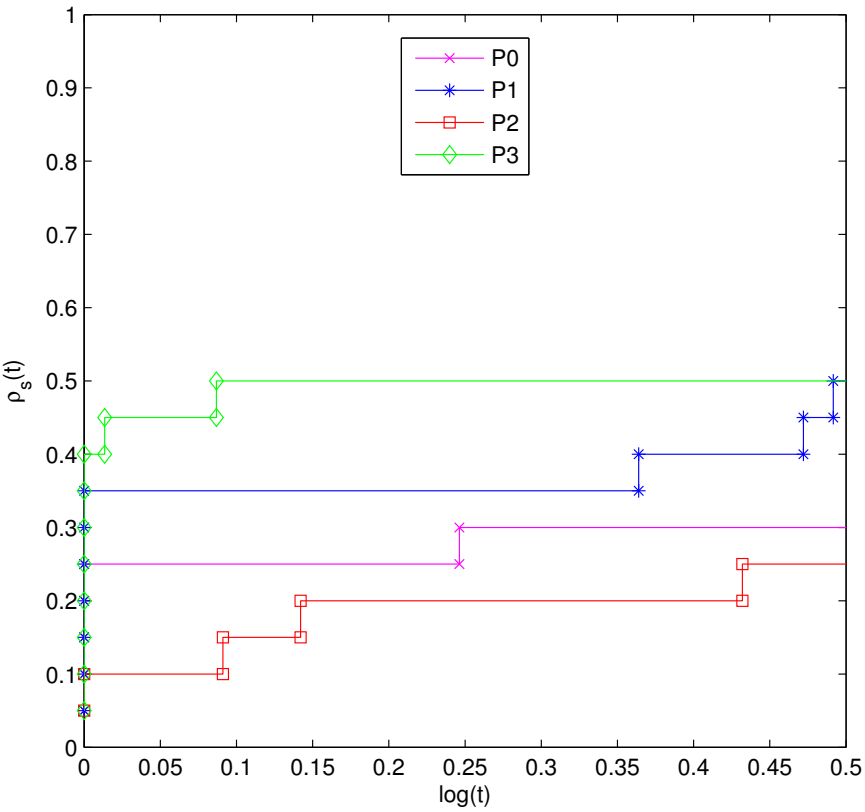



Figura 5.5: Perfil de desempenho do método SID-PSM utilizando os 4 tipos de atualização de passo disponíveis na implementação, com ênfase na eficiência dos métodos.

robustez (resolveram em torno de 60% dos problemas), e também a mesma eficiência, resolvendo cerca de 40% dos problemas com a menor quantidade de avaliações de função. Para efeito de comparação, o melhor entre os outros três métodos testados foi o Nelder-Mead, que resolveu apenas 15% dos problemas com o menor número de avalições de função e, no total, resolveu cerca de 50% dos problemas.

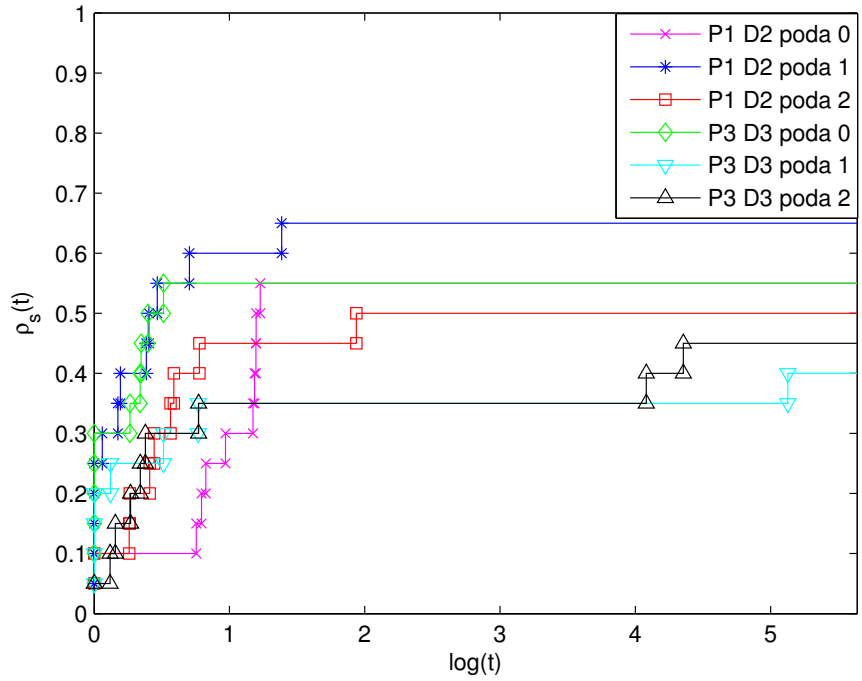



Figura 5.6: Perfil de desempenho do método SID-PSM sem poda, com poda e apenas direção mais promissora.

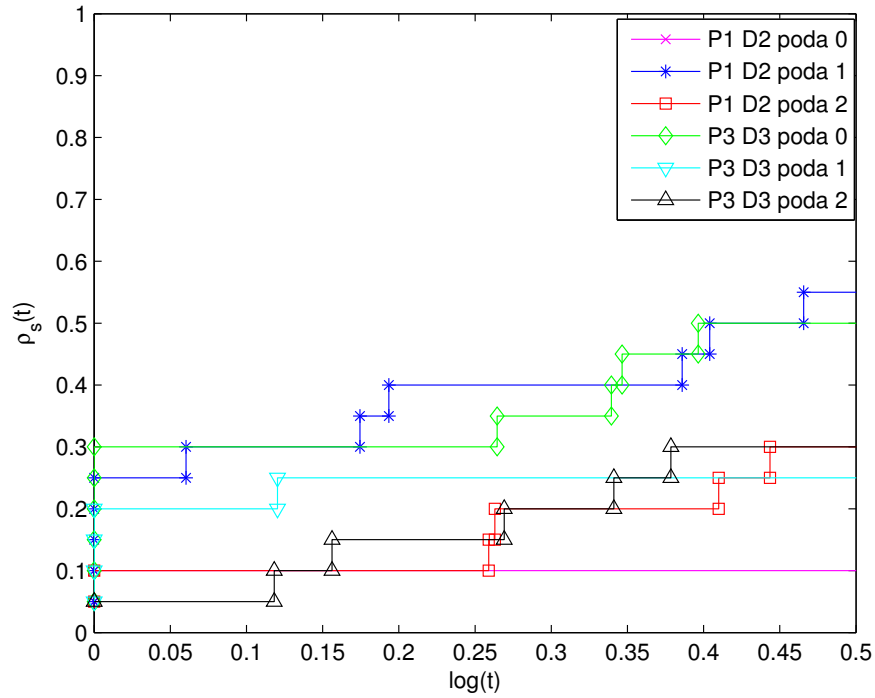



Figura 5.7: Perfil de desempenho do método SID-PSM sem poda, com poda e apenas direção mais promissora, com ênfase na efeiciência.

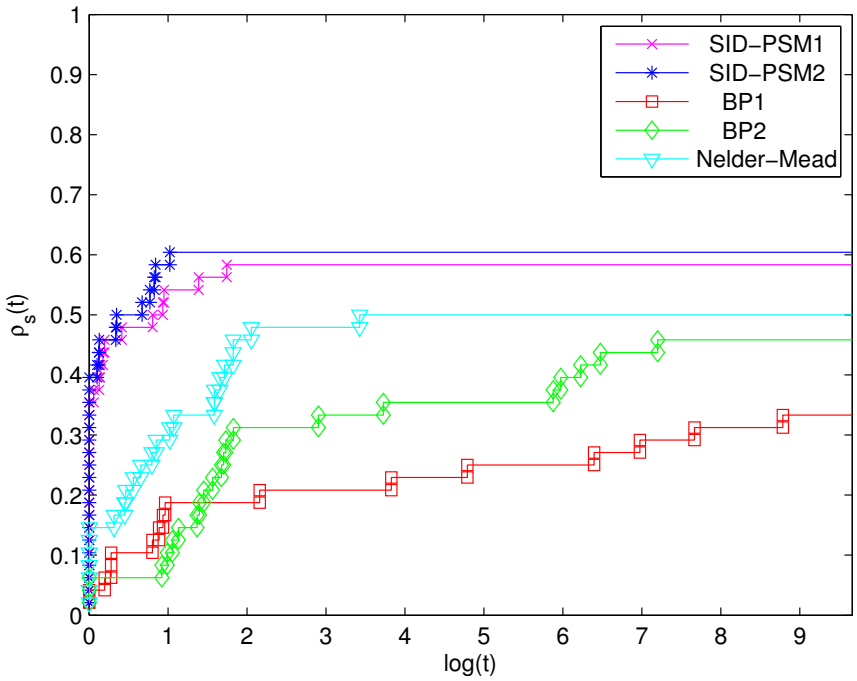



Figura 5.8: Perfil de desempenho dos método SID-PSM1,SID-PSM2, Nelder-Mead, BP1 e BP2.

# Capítulo 6

# Otimização de parâmetros de algoritmos

Em geral, métodos matemáticos dependem de alguns parâmetros que podem influenciar (muito) na sua eficiência. Os autores dos métodos costumam apresentar sugestões para esses parâmetros, algumas vezes totalmente arbitrários. Neste capítulo, seguindo as ideias apresentadas em [2], encontraremos os parâmetros ótimos, que maximizam a eficiência, do método SID-PSM. Este problema não é diferenciável, logo a teoria de convergência apresentada anteriormente não pode ser aplicada. Ainda assim podemos aplicar os métodos e esperar bons resultados.

Nosso problema é: dado um método $s$, que depende dos parâmetros $x_i$, $i = 1, \ldots, n$, encontrar parâmetros ótimos $x_i^*$, $i = 1, \ldots, n$, que minimizam alguma medida de desempenho do método.

Visto que no capítulo anterior tanto o SID-PSM1, quanto o SID-PSM2 obtiveram o desempenho muito semelhante, e o SID-PSM1 utiliza mais as derivadas simplex, o que pode siginificar um maior potencial a ser explorado, decidimos modificar alguns de seus parâmetros para minimizar o número de avaliações de função realizadas. Utilizamos um subconjunto $P$ de problemas do pacote CUTEr, disponível em [22]. Este subconjunto é uma seleção de 22 problemas-teste para minimização irrestrita cuja função objetivo é a soma de quadrados. Os autores deste subconjunto, Moré e Wild, selecionaram os problemas deste pacote especificamente para teste de *softwares derivative-free*. São sugeridas várias dimensões para os problemas, utilizamos sempre a menor, resolvendo apenas uma vez cada um dos problemas. Assim os problemas resolvidos têm dimensão entre 2 e 11. Como estamos utilizando o número de avaliações de função como medida de desempenho, definimos nossa função objetivo como a soma do número de avaliações de função realizadas pelo método SID-PSM1 para resolver cada um dos problemas em $P$, como sugerida em [2].

Os parâmetros que deixamos como variáveis do nosso problema foram: $\gamma_1$ e $\gamma_2$ (medidas de qualidade do modelo), $\phi$ (parâmetro de crescimento do passo), $\theta$ (parâmetro de decrescimento do passo), $\bar{\Lambda}$ e $\sigma$, onde $\bar{\Lambda} = \frac{\Lambda}{100}$ é um reescalamento feito para evitar erros numéricos. Essas variáveis não podem assumir quaisquer valores, então, teoricamente

estamos trabalhando com o problema restrito:

$$\begin{aligned} \min \quad & f(x) = \sum_{i \in P} f_i(x) \\ \text{s.a} \quad & x \in \Omega \subset I\!R^6, \end{aligned} \tag{6.1}$$

onde $f_i : \Omega \to I\!R$ é o número de avaliações de função que o método SID-PSM1 gasta para resolver o problema $i \in P$, utilizando os parâmetros dados por $x$, e

$$\Omega = \left\{ (\gamma_1, \gamma_2, \phi, \theta, \bar{\Lambda}, \sigma) : \begin{array}{l} \gamma_1 < \gamma_2 \\ 0 \leq \theta < 1, \\ \phi \geq 1, \\ \gamma_1,\ \gamma_2,\ \sigma,\ \bar{\Lambda} \geq 0 \end{array} \right\}.$$

Se definirmos $\theta = 0$, já no primeiro fracasso teremos o critério de parada satisfeito, porém, é provável que ainda estejamos muito longe do ótimo. Queremos evitar isso durante a otimização do método. Uma alternativa é penalizar a função $f$ quando a execução é interrompida longe do menor valor de função obtido pelo SID-PSM1, utilizando os parâmetros iniciais.

Vamos definir $g^0 \in I\!R^{|P|}$, o vetor com os menores valores de função obtidos ao aplicar o SID-PSM1 aos problemas em $P$, utilizando os parâmetros iniciais. Seja $g : \Omega \to I\!R^p$ tal que $g_i(x)$ é o menor valor de função obtido pelo método SID-PSM1 aplicado ao problema $i \in P$, utilizando os parâmetros dados por $x$.

Queremos que o método com os parâmetros otimizados resolva os problemas tão bem quanto o método com os parâmetros originais. Esperamos que o método otimizado satisfaça:

$$g(x^*) \leq 1.25 g^0, \tag{6.2}$$

onde $x^* \in \Omega$ é o vetor com os parâmetros ótimos. Não queremos restringir o problema apenas aos pontos que satisfazem essa condição, mas queremos evitar pontos que não a satisfazem. Para isso criamos o vetor $g^+(x) \in I\!R^{|P|}$ tal que

$$g_i^+(x) = \max\left\{ g_i(x) - 1.25 g_i^0 \,,\, 0 \right\},$$

para todo $i$. Dessa forma obtemos uma função de penalização $p : \Omega \to I\!R$, tal que

$$p(x) = \frac{\|g^+(x)\|}{\|g_0\|} M,$$

onde $M > 0$ é um valor grande o suficiente.

Utilizando esta função de penalização, definiremos $f^p : \Omega \to I\!R$, tal que

$$f^p(x) = f(x) + p(x).$$

O método SID-PSM, da forma como foi estudado nesse trabalho, é um método de minimização irrestrita, logo não podemos aplicá-lo para resolver (6.1). Vamos criar

uma função $\bar{f} : \mathbb{R}^6 \to \mathbb{R}$ tal que, se $x \in \Omega$, $\bar{f}(x) = f^p(x)$, caso contrário, $\bar{f}(x) = M \geq \sup_{y \in \Omega} f(y)$.

Para todo $i \in P$, $f_i$ é limitada superiormente pelo máximo de avaliações de função que o SID-PSM1 pode realizar, no nosso caso $10^6$. Como $f$ é a soma de todos os $f_i$'s, então $f$ é limitada superiormente por $|P|10^6$. Assim um possível valor para $M$ é $(|P|+1)10^6$.

Desta forma podemos definir $\bar{f}$ como

$$\bar{f}(x) = \begin{cases} f^p(x) & \text{se } x \in \Omega \\ (|P|+1)10^6 & \text{c. c.} \end{cases} . \tag{6.3}$$

Aplicamos os métodos SID-PSM1 e SID-PSM2 à função (6.3), a fim de obter os parâmetros ótimos do método SID-PSM1. O ponto inicial foi

$$x^0 = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \phi \\ \theta \\ \bar{\Lambda} \\ \sigma \end{bmatrix} = \begin{bmatrix} 0,25 \\ 0,75 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

onde $f^p(x^0) = 213551$, e obtivemos

$$x_1^* = \begin{bmatrix} 1,25047302246094 \\ 4,00047302246094 \\ 4,00047302246094 \\ 0,500473022460938 \\ 1,00041198730469 \\ 6,00047302246094 \end{bmatrix} \quad \text{e} \quad x_2^* = \begin{bmatrix} 0,356119791666667 \\ 0,727660636235692 \\ 2,24152294576982 \\ 0,227170815547733 \\ 0,448801810194877 \\ 2 \end{bmatrix},$$

em 175 e 199 avaliações de função, parâmetros ótimos usando SID-PSM1 e SID-PSM2, respectivamente. Os valores da função $f^p$ para esses dois pontos são

$$f^p(x_1^*) = 20782,8391252206 \qquad \text{e} \qquad f^p(x_2^*) = 24324,3851498036.$$

Comparamos o desempenho do SID-PSM1, original e otimizado (com as duas soluções encontradas), para resolver os problemas em $P$, os dados obtidos estão disponíveis na tabela 18 no Anexo A. A Figura 6.1 nos dá o perfil de desempenho destes métodos.

Resolvemos novamente todos os problemas apresentados em [15] com o método SID-PSM1, desta vez utilizando os parâmetros otimizados, os dados são apresentados nas tabelas 14 e 15 no Anexo A. O gráfico apresentado na Figura 6.2 mostra o perfil de desempenho comparando o SID-PSM1 com uso dos parâmetros originais e otimizados.

Em ambos os gráficos podemos ver que as duas soluções não foram satisfatórias, pois, pioram tanto a robustez quanto a eficiência do método. Pela parte não inteira de $f^p(x_1^*)$ e $f^p(x_2^*)$ sabemos que ambos os pontos não satisfazem a condição (6.2), assim

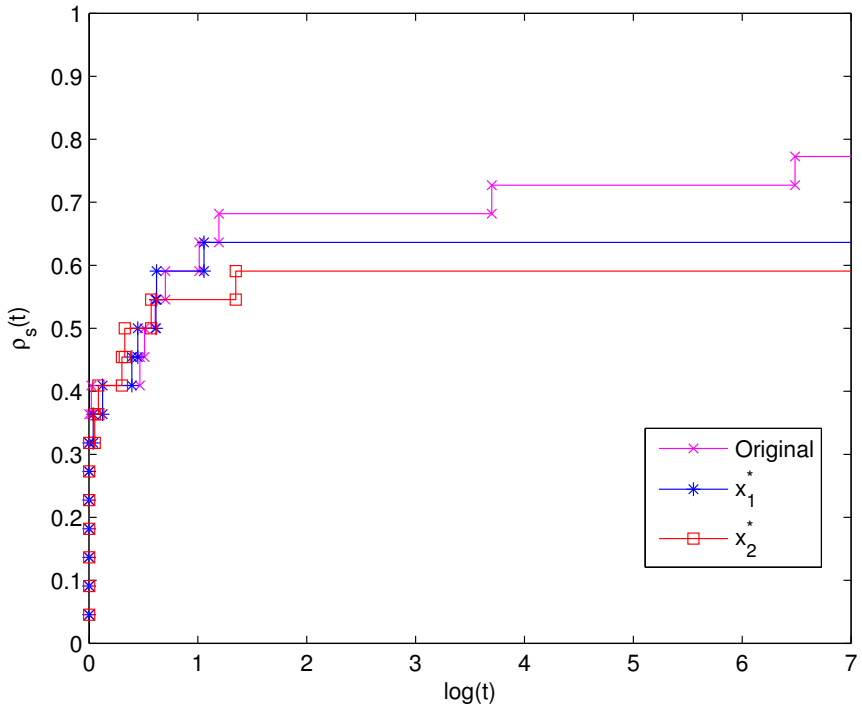



Figura 6.1: Perfil de desempenho do método SID-PSM1 com parâmetros originais, dados por $x_1^*$ e por $x_2^*$ resolvendo problemas $P$.

devemos aplicar uma mudança na forma de penalização para que a solução seja mais robusta. Em relação à eficiência, apesar de termos diminuído o valor de $f^p$, a eficiência, mesmo nos problemas do pacote CUTEr, não melhorou. Isso, provavelmente, porque as comparações são feitas problema a problema, enquanto a otimização é feita em conjunto, logo se um problema gasta $10^5$ avaliações de função e outros três gastam $10^2$, os três últimos podem aumentar o número de avaliações que realizam que não irão interferir tanto no valor final da função objetivo. Uma possível solução para isso é aplicar pesos para cada um dos problemas.

Estas duas estratégias serão aplicadas a seguir. Vamos definir $f_i^0$ como o número de avaliações de função necessárias para o método SID-PSM1, utilizando os parâmetros originais, resolver o problema $i$. Seja $\hat{f} : \mathbb{R}^6 \to \mathbb{R}$, tal que

$$\hat{f}(x) = \sum_{i \in P} \frac{f_i(x)}{f_i^0}.$$

Desta forma estamos dando pesos inversamente proporcionais ao número de avaliações de função gasto originalmente. Para tentarmos garantir a robustez do método com a solução encontrada, iremos aplicar uma penalização, de modo que a função a ser otimizada é

$$\hat{f}_{\bar{\Omega}}(x) = \begin{cases} \hat{f}(x) & \text{se } x \in \bar{\Omega} \\ (|P| + 1) & \text{c. c.} \end{cases}, \tag{6.4}$$

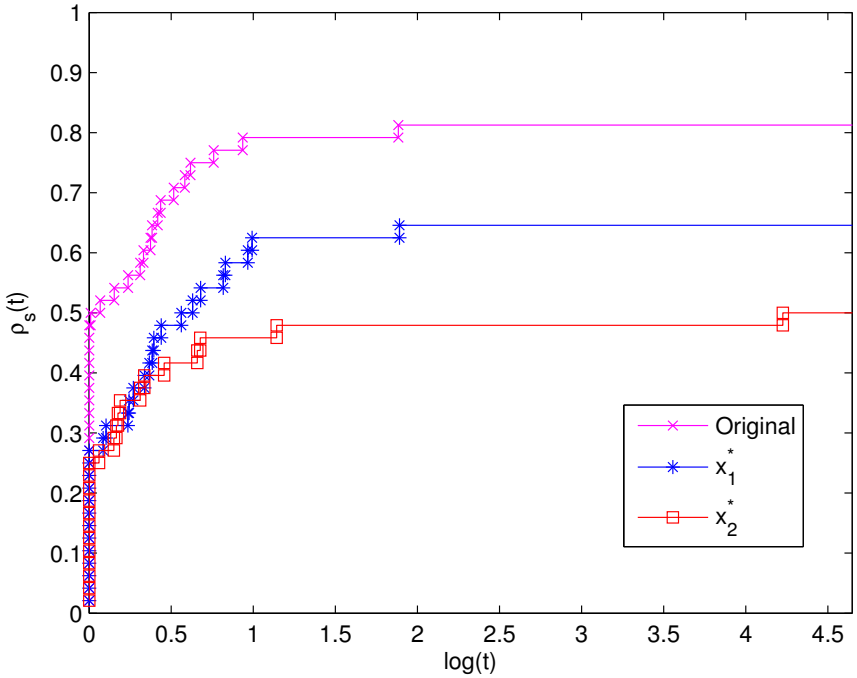



Figura 6.2: Perfil de desempenho do método SID-PSM1 com parâmetros originais e otimizados, resolvendo problemas apresentados em [15].

onde $\bar{\Omega}$ é o conjunto $\Omega$ original intersecção com os parâmetros que fazem com que os menores valores de função encontrados pelo método SID-PSM1, ao resolver os problemas em $P$, sejam 25% maiores que as soluções encontradas utilizando os parâmetros originais em no máximo um problema.

Aplicamos os métodos SID-PSM1 e SID-PSM2 à função (6.4), O ponto inicial foi

$$x^0 = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \phi \\ \theta \\ \bar{\Lambda} \\ \sigma \end{bmatrix} = \begin{bmatrix} 0,25 \\ 0,75 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

onde $\hat{f}_{\bar{\Omega}}(x^0) = 22$, e obtivemos

$$x_3^* = \begin{bmatrix} 0,101562500000000 \\ 0,750976562500000 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix} \quad \text{e} \quad x_4^* = \begin{bmatrix} 0,00520833333333333 \\ 0,719187084463023 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

em 165 e 172 avaliações de função, parâmetros ótimos usando SID-PSM1 e SID-PSM2, respectivamente. Os valores da função $\hat{f}_{\bar{\Omega}}(x)$ para esses dois pontos são

$$\hat{f}_{\bar{\Omega}}(x_3^*) = 17,0288769517664 \qquad \text{e} \qquad \hat{f}_{\bar{\Omega}}(x_4^*) = 18,6011524091986.$$

Comparamos o desempenho do SID-PSM1, original e otimizado (com as duas soluções encontradas), para resolver os problemas em $P$, os dados obtidos estão na tabela 19 no Anexo A. A Figura 6.3 nos dá o perfil de desempenho destes métodos.

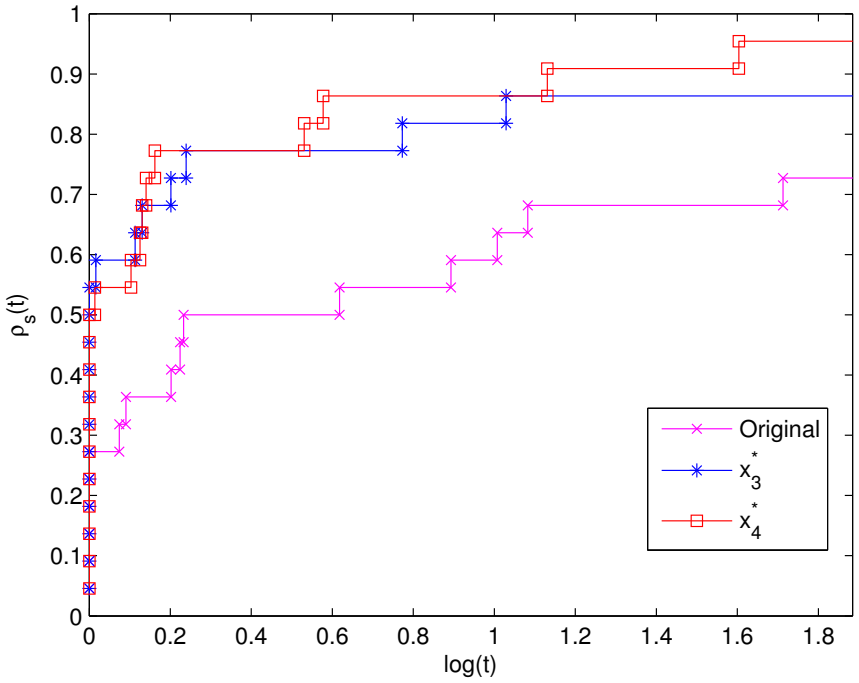



Figura 6.3: Perfil de desempenho do método SID-PSM1 com parâmetros originais, dados por $x_3^*$ e por $x_4^*$ resolvendo problemas $P$.

Resolvemos novamente todos os problemas apresentados em [15] com o método SID-PSM1, desta vez utilizando os parâmetros otimizados, disponibilizamos os resultados obtidos nas tabelas 16 e 17 no Anexo A. O gráfico apresentado na Figura 6.4 mostra o perfil de desempenho comparando o SID-PSM1 com uso dos parâmetros originais e otimizados.

Para facilitar a análise plotamos os dados do gráfico 6.4 mudando a escala do eixo $x$, para podermos analisar como se comporta o método com as diferentes soluções utilizando poucas avaliações de função, este novo gráfico está apresentado na figura 6.5.

Podemos ver que as soluções foram bastante satisfatórias, gerando parâmetros mais eficientes e mais robustos que os originais. Por exemplo quando passamos a utilizar os parâmetros dados por $x_4^*$, aumentamos a eficiência em cerca de 15% e a robustez em aproximadamente 10%. Isso mostra o grande potencial desta técnica para obter resultados melhores e mais rápidos.

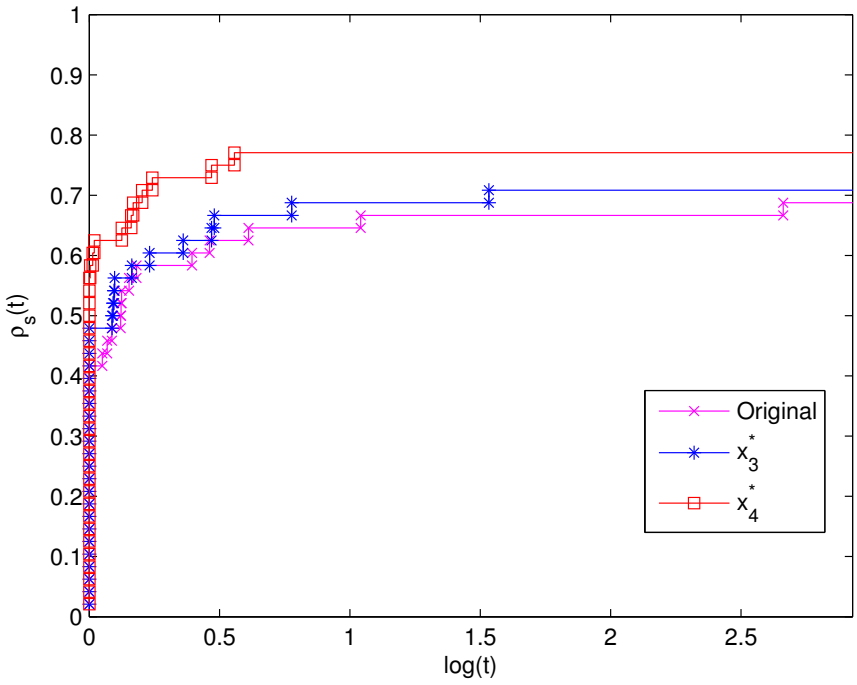



Figura 6.4: Perfil de desempenho do método SID-PSM1 com parâmetros originais e otimizados, resolvendo problemas apresentados em [15].

É importante ressaltar a escolha do valor $1,25$ nas duas formas de penalização. Esta escolha não está necessariamente ligada à forma como definimos "resolver o problema". Por exemplo, na figura 6.6 mudamos o limitante do erro na definição 9 do Capítulo 5 para

$$\frac{\theta_p(s) - \hat{\theta}_p}{\hat{\theta}_p} \leq 10^{-5} \qquad \text{se}\, \hat{\theta}_p \neq 0,$$

e resolvemos os problemas de [15] com os parâmetros originais, $x_3^*$ e $x_4^*$.

Neste caso vemos uma pequena perda de robustez quando utilizamos os parâmetros dados por $x_3^*$, porém quando utilizamos os parâmetros dados por $x_4^*$ continuamos com a melhora tanto na robustez quanto na eficiência do método.

Tentamos resolver o problema exigindo $g_i(x) \leq g_i^0$ (utilizando a definição original para "resolver os problemas"), porém continuamos sem resultados satisfatórios para a função $f^p$ e, ao otimizar a função $\hat{f}_{\bar{\Omega}}$, o ponto ótimo foi o ponto inicial.

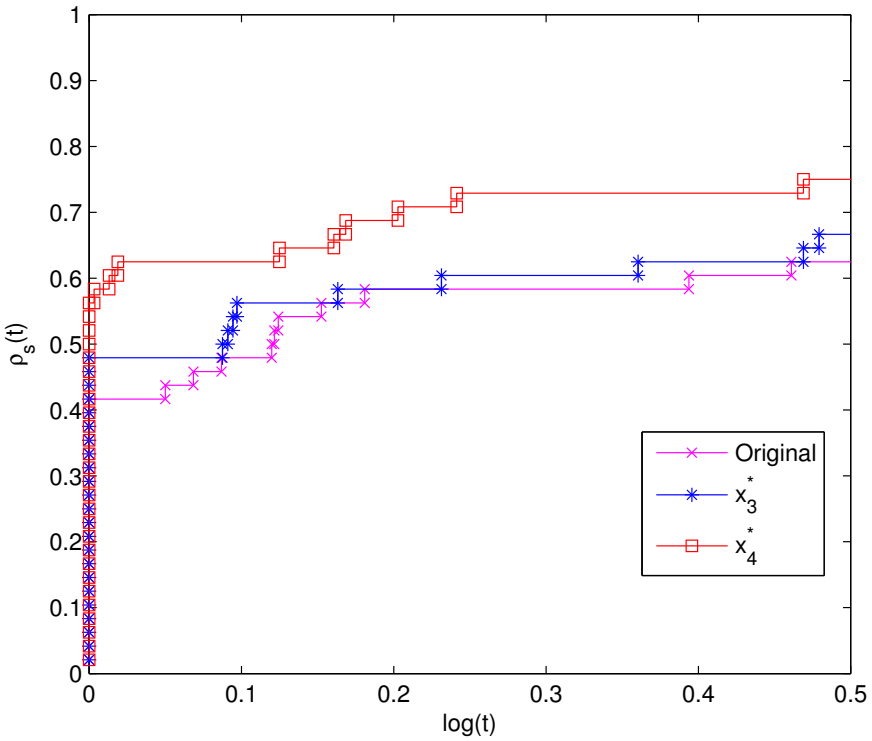



Figura 6.5: Perfil de desempenho do método SID-PSM1 com parâmetros originais e otimizados, resolvendo problemas apresentados em [15], com *zoom* no eixo $x$.

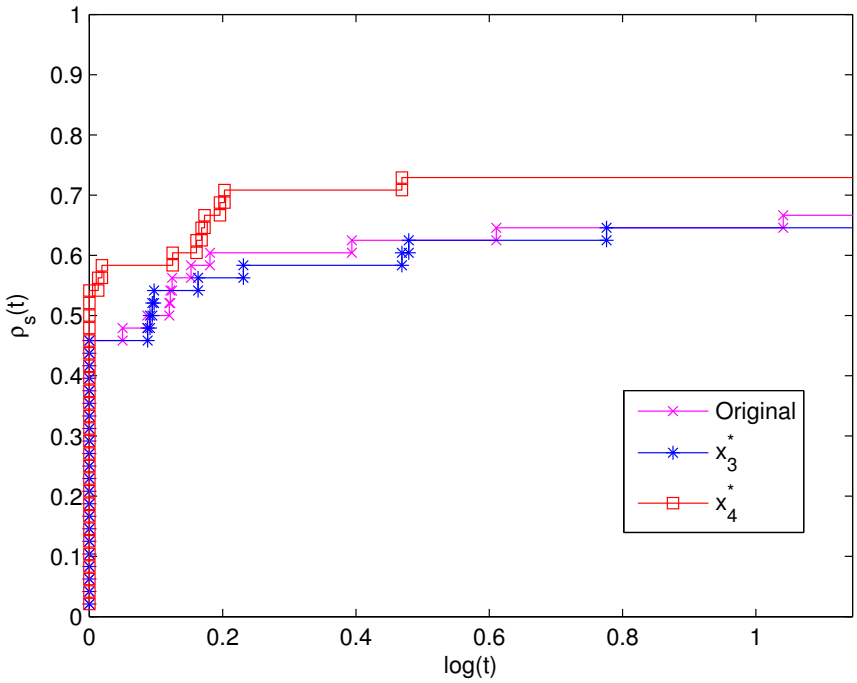



Figura 6.6: Perfil de desempenho do método SID-PSM1 com parâmetros originais e otimizados, resolvendo problemas apresentados em [15], com mudança na definição de resolver o problema.

# Capítulo 7

# Conclusão

O método SID-PSM se mostrou eficiente na resolução dos problemas sugeridos em [15]. Tanto em relação ao número de avaliações de função quanto em relação à quantidade de problemas resolvidos, esse método se mostrou superior aos métodos Nelder-Mead e de busca padrão.

No artigo [15], os autores sugerem uma lista de problemas de minimização irrestrita onde a função objetivo é uma soma de quadrados. Os problemas de 1 a 20 têm dimensão fixa. O restante tem dimensão livre. Nos problemas de dimensão livre usamos no máximo 20 variáveis, isso porque o SID-PSM possui iterações muito caras: a lista $\mathcal{L}$ cresce quadraticamente com o tamanho do problema. Ignorando a distância dos pontos da lista até o iterando, vemos que o total de verificações por $\Lambda$-posicionamente que o método realiza por iteração é da ordem de $n^2$.

Por esse motivo, o método SID-PSM pode não ser uma boa alternativa quando se trabalha com problemas grandes, cujo cálculo da função objetivo é rápido. Porém se o custo de avaliar a função for muito caro (como no caso do problema apresentado no Capítulo 6), pode ser uma boa alternativa aos métodos clássicos sem derivadas.

O problema de otimização de parâmetros, apresentado no Capítulo 6, foi resolvido utilizando o *software* SID-PSM. Escolhemos um método *derivative-free* pois não temos acesso nem mesmo à função, ainda menos a suas derivadas. Esse fato inviabiliza o uso de qualquer método que faz uso de derivadas. A escolha do SID-PSM dentre outros métodos sem derivadas se deu pelo fato que nosso problema é pequeno (6 variáveis) e o custo de avaliar a função objetivo é muito grande, que como visto anteriormente é o tipo de problema em que o SID-PSM se destaca.

Ainda sobre o problema de otimização de parâmetros de algoritmos, os resultados da otimização da primeira função sugerida não foram satisfatórios, algumas justificativas para este fato estão na simplicidade da função objetivo.

Já a segunda função, mais elaborada, com peso para cada um dos termos da sua soma, nos ofereceu resultados satisfatórios, melhorando tanto a robustez quanto a eficiência do método SID-PSM1. O que demonstra a possibilidade de utilizar o método SID-PSM, e outros métodos de otimização sem derivada, para melhorar métodos já conhecidos, fazendo com que estes atinjam o máximo de seu potencial.

# Anexo A - Resultados numéricos

Apresentamos aqui as tabelas com os resultados numéricos obtidos ao aplicar os métodos nos problemas:

| SID-PSM | Armazena todos | | Armazena sucesso | |
|---|---|---|---|---|
| | # de Avaliações | Menor valor | # de Avaliações | Menor valor |
| Rosen | 334 | 1,1938e-17 | 539 | 2,3669e-11 |
| Froth | 89 | 0 | 70 | 0 |
| BadscaP | 183 | 1,1256e-08 | 156 | 1,1285e-08 |
| BadscaB | 487 | 2,1261e-13 | 442 | 5,7579e-07 |
| Beale | 235 | 6,4168e-18 | 237 | 6,8072e-14 |
| JenSam | 251 | 1,2436e+02 | 246 | 1,2436e+02 |
| Helix | 440 | 7,1504e-17 | 658 | 7,8210e-10 |
| Bard | 401 | 8,2149e-03 | 540 | 8,2148e-03 |
| Gauss | 213 | 1,1279e-08 | 155 | 1,1281e-08 |
| Meyer | 254630 | 3,1796e+08 | 210111 | 1,6779e+04 |
| Gulf | 2649 | 1,9180e-17 | 1236 | 2,8976e-13 |
| Box | 420 | 5,9298e-02 | 570 | 5,9298e-02 |
| Sing | 803 | 1,1930e-12 | 2577 | 8,2100e-08 |
| Wood | 613 | 1,6825e-16 | 1590 | 3,8592e-09 |
| KowOsb | 873 | 3,0750e-04 | 1564 | 3,0750e-04 |
| BD | 474 | 8,5822e+04 | 843 | 8,5822e+04 |
| Osb1 | 171 | 1,1529 | 300 | 1,1529 |
| Biggs | 12854 | 5,7542e-11 | 44551 | 2,3753e-07 |
| Osb2 | 16971 | 3,4543e-01 | 909164 | 3,4543e-01 |
| Watson | 5208 | 2,8951e-05 | 773540 | 7,6699e-05 |

Tabela 1: Reultados numéricos da comparação do SID-PSM usando o armazenamento de todos os pontos ou apenas dos pontos que oferecem decréscimo

| # de Avaliações | D0 | D1 | D2 | D3 |
|---|---|---|---|---|
| Rosen | 334 | 617 | 579 | 323 |
| Froth | 89 | 113 | 158 | 89 |
| BadscaP | 183 | 1302 | 452 | 1000001 |
| BadscaB | 487 | 588 | 1205 | 103304 |
| Beale | 235 | 442 | 180 | 347 |
| JenSam | 251 | 249 | 480 | 320 |
| Helix | 440 | 690 | 591 | 447 |
| Bard | 401 | 413 | 588 | 321 |
| Gauss | 213 | 224 | 289 | 207 |
| Meyer | 254630 | 30007 | 1000003 | 92150 |
| Gulf | 2649 | 20428 | 19586 | 1871 |
| Box | 420 | 333 | 437 | 436 |
| Sing | 803 | 804 | 948 | 633 |
| Wood | 613 | 1091 | 923 | 562 |
| KowOsb | 873 | 1129 | 1252 | 420 |
| BD | 474 | 515 | 794 | 355 |
| Osb1 | 171 | 292 | 334 | 172 |
| Biggs | 12854 | 12052 | 18052 | 12500 |
| Osb2 | 16971 | 43681 | 75993 | 9503 |
| Watson | 5208 | 543221 | 1000004 | 7472 |

Tabela 2: Número de avaliações de função realizadas pelo método SID-PSM para resolver cada um dos problemas utilizandos os diferentes tipos de conjunto de pesquisa

| Menor valor | D0 | D1 | D2 | D3 |
|---|---|---|---|---|
| Rosen | 1,1938e-17 | 1,1052e-17 | 3,1543e-18 | 4,9021e-19 |
| Froth | 0 | 0 | 0 | 0 |
| BadscaP | 1,1256e-08 | 9,1785e-08 | 4,3248e-08 | 4,5941e-07 |
| BadscaB | 2,1261e-13 | 3,6647e-17 | 2,1793e-13 | 2,4288e-12 |
| Beale | 6,4168e-18 | 8,7687e-18 | 0 | 1,1874e-18 |
| JenSam | 1,2436e+02 | 1,2436e+02 | 1,2436e+02 | 1,2436e+02 |
| Helix | 7,1504e-17 | 8,6822e-18 | 5,3659e-19 | 1,2601e-16 |
| Bard | 8,2149e-03 | 8,2149e-03 | 8,2149e-03 | 8,2149e-03 |
| Gauss | 1,1279e-08 | 1,1279e-08 | 1,1279e-08 | 1,1279e-08 |
| Meyer | 3,1795e+08 | 8,8043e+01 | 4,2491e+04 | 7,7035e+03 |
| Gulf | 1,9180e-17 | 1,3798e-14 | 8,4610e-20 | 5,7945e-19 |
| Box | 5,9298e-02 | 0 | 0 | 3,0624e-23 |
| Sing | 1,1930e-12 | 2,1927e-13 | 1,0389e-10 | 1,1424e-12 |
| Wood | 1,6825e-16 | 2,1908e-15 | 1,2039e-16 | 1,0909e-12 |
| KowOsb | 3,0751e-04 | 3,0751e-04 | 3,0751e-04 | 3,0751e-04 |
| BD | 8,5822e+04 | 8,5822e+04 | 8,5822e+04 | 8,5822e+04 |
| Osb1 | 1,1529 | 1,1529 | 1,1529 | 1,1529 |
| Biggs | 5,7542e-11 | 1,5567e-08 | 6,0344e-13 | 2,4278e-01 |
| Osb2 | 3,4543e-01 | 4,0137e-02 | 4,0137e-02 | 4,0137e-02 |
| Watson | 2,8952e-05 | 2,3887e-06 | 4,8397e-05 | 5,5071e-08 |

Tabela 3: Menor valor de função encontrado pelo método SID-PSM ao resolver cada um dos problemas utilizandos os diferentes tipos de conjunto de pesquisa

| # de Avaliações | P0 | P1 | P2 | P3 |
|---|---|---|---|---|
| Rosen | 323 | 193 | 190 | 196 |
| Froth | 89 | 89 | 89 | 89 |
| BadscaP | 1000001 | 13490 | 13149 | 168 |
| BadscaB | 103304 | 414 | 411 | 1000000 |
| Beale | 347 | 107 | 166 | 139 |
| JenSam | 320 | 181 | 202 | 105 |
| Helix | 447 | 298 | 355 | 222 |
| Bard | 321 | 258 | 326 | 186 |
| Gauss | 207 | 106 | 143 | 107 |
| Meyer | 92150 | 29430 | 37422 | 1000000 |
| Gulf | 1871 | 1325 | 1406 | 220645 |
| Box | 436 | 227 | 254 | 189 |
| Sing | 633 | 659 | 501 | 581 |
| Wood | 562 | 664 | 452 | 292 |
| KowOsb | 420 | 296 | 245 | 230 |
| BD | 355 | 277 | 307 | 197 |
| Osb1 | 172 | 145 | 160 | 154 |
| Biggs | 12500 | 2203 | 16239 | 289893 |
| Osb2 | 9503 | 3914 | 4589 | 1462 |
| Watson | 7472 | 4385 | 5987 | 2188 |

Tabela 4: Número de avaliações de função realizadas pelo método SID-PSM para resolver cada um dos problemas utilizandos os diferentes tipos de atualização de passo

| Menor valor | D0 | D1 | D2 | D3 |
|---|---|---|---|---|
| Rosen | 4,9021e-19 | 1,0332e-17 | 8,1150e-18 | 1,09815e-17 |
| Froth | 0 | 0 | 0 | 0 |
| BadscaP | 4,5941e-07 | 2,6815e-04 | 5,7999e-04 | 1,3452e-01 |
| BadscaB | 2,4288e-12 | 1,0361e-13 | 6,0393e-14 | 9,6507e+11 |
| Beale | 1,1874e-18 | 2,3238e-16 | 1,4426e-17 | 1,2570e-18 |
| JenSam | 1,2436e+02 | 1,2436e+02 | 1,2436e+02 | 1,2436e+02 |
| Helix | 1,2601e-16 | 4,8701e-17 | 2,5418e-16 | 1,7358e-16 |
| Bard | 8,2149e-03 | 8,2149e-03 | 8,2149e-03 | 8,2149e-03 |
| Gauss | 1,1279e-08 | 1,1279e-08 | 1,1279e-08 | 1,1279e-08 |
| Meyer | 7,7035e+03 | 1,9638e+04 | 1,2553e+04 | 5,5192e+07 |
| Gulf | 5,7945e-19 | 1,6192e-17 | 1,2078e-18 | 2,6717e-07 |
| Box | 3,0625e-23 | 1,1656e-29 | 8,2608e-28 | 1,3259e-20 |
| Sing | 1,1424e-12 | 1,8215e-18 | 7,5256e-12 | 1,8877e-09 |
| Wood | 1,0909e-12 | 1,8726e-15 | 1,1499e-09 | 4,5421e-16 |
| KowOsb | 3,0750e-04 | 3,0750e-04 | 3,0750e-04 | 3,0750e-04 |
| BD | 8,5822e+04 | 8,5822e+04 | 8,5822e+04 | 8,5822e+04 |
| Osb1 | 1,1529 | 1,1529 | 1,1529 | 1,1529 |
| Biggs | 2,4279e-01 | 1,9731e-16 | 2,4274e-01 | 2,4275e-01 |
| Osb2 | 4,0137e-02 | 4,0137e-02 | 4,0137e-02 | 4,0137e-02 |
| Watson | 5,5071e-08 | 5,4737e-07 | 2,6957e-06 | 1,6966e-06 |

Tabela 5: Menor valor de função encontrado pelo método SID-PSM ao resolver cada um dos problemas de MGH utilizandos os diferentes tipos de atualizações de passo

| # de Avaliações | sem poda | mais promissora | poda |
|---|---|---|---|
| Rosen | 322 | 147 | 394 |
| Froth | 158 | 79 | 93 |
| BadscaP | 120 | 1121 | 176798 |
| BadscaB | 681 | 462 | 683 |
| Beale | 257 | 97 | 184 |
| JenSam | 186 | 171 | 126 |
| Helix | 450 | 290 | 113 |
| Bard | 288 | 203 | 147 |
| Gauss | 130 | 98 | 102 |
| Meyer | 49455 | 94025 | 16355 |
| Gulf | 1020 | 1275 | 1422 |
| Box | 314 | 178 | 199 |
| Sing | 547 | 483 | 507 |
| Wood | 528 | 389 | 593 |
| KowOsb | 523 | 263 | 346 |
| BD | 376 | 217 | 281 |
| Osb1 | 274 | 122 | 140 |
| Biggs | 1605 | 2682 | 1197 |
| Osb2 | 3348 | 3822 | 5600 |
| Watson | 12385 | 4498 | 3632 |

Tabela 6: Número de avaliações de função realizadas pelo método SID-PSM para resolver cada um dos problemas de MGH utilizandos D2, P1 e os diferentes tipos de poda

| Menor valor | sem poda | mais promissora | poda |
|---|---|---|---|
| Rosen | 1,5998e-017 | 5,7629e-017 | 1,3769e-017 |
| Froth | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 |
| BadscaP | 5,6801e-006 | 3,3184e-004 | 2,2243e-004 |
| BadscaB | 3,0924e-012 | 8,4295e-013 | 1,0178e-012 |
| Beale | 7,7912e-018 | 1,5973e-018 | 4,7222e-018 |
| JenSam | 1,2436e+002 | 1,2436e+002 | 1,2436e+002 |
| Helix | 1,6569e-017 | 3,9727e-018 | 0,0000e+000 |
| Bard | 8,2149e-003 | 8,2149e-003 | 8,2149e-003 |
| Gauss | 1,1279e-008 | 1,1279e-008 | 1,1279e-008 |
| Meyer | 4,1691e+003 | 4,6695e+002 | 1,9673e+004 |
| Gulf | 1,2157e-016 | 1,0486e-017 | 1,3257e-015 |
| Box | 1,9010e-027 | 6,4782e-028 | 1,6785e-026 |
| Sing | 6,2380e-011 | 6,1195e-015 | 3,4656e-011 |
| Wood | 3,4136e-017 | 5,0894e-016 | 3,0236e-015 |
| KowOsb | 3,0751e-004 | 3,0751e-004 | 3,0751e-004 |
| BD | 8,5822e+004 | 8,5822e+004 | 8,5822e+004 |
| Osb1 | 1,1529e+000 | 1,1529e+000 | 1,1529e+000 |
| Biggs | 2,0288e-016 | 1,9088e-014 | 1,8069e-013 |
| Osb2 | 4,0138e-002 | 4,0138e-002 | 4,0138e-002 |
| Watson | 2,0141e-004 | 3,7917e-005 | 3,0685e-005 |

Tabela 7: Menor valor de função encontrado pelo método SID-PSM ao resolver cada um dos problemas de MGH utilizandos D2, P1 e os diferentes tipos de poda

| # de Avaliações | sem poda | mais promissora | poda |
|---|---|---|---|
| Rosen | 196 | 172 | 185 |
| Froth | 89 | 70 | 78 |
| BadscaP | 168 | 241 | 164 |
| BadscaB | 1000000 | 1000000 | 1000000 |
| Beale | 139 | 98 | 107 |
| JenSam | 105 | 179 | 1777 |
| Helix | 222 | 190 | 311 |
| Bard | 186 | 5137 | 3009 |
| Gauss | 107 | 75 | 95 |
| Meyer | 1000000 | 1000000 | 16184 |
| Gulf | 220645 | 90211 | 75555 |
| Box | 189 | 12936 | 192 |
| Sing | 581 | 567 | 798 |
| Wood | 292 | 274 | 313 |
| KowOsb | 230 | 250 | 299 |
| BD | 197 | 164 | 178 |
| Osb1 | 154 | 117 | 141 |
| Biggs | 289893 | 302414 | 288325 |
| Osb2 | 1462 | 2085 | 2497 |
| Watson | 2188 | 11619 | 40741 |

Tabela 8: Número de avaliações de função realizadas pelo método SID-PSM para resolver cada um dos problemas de MGH utilizandos D3, P3 e os diferentes tipos de poda

| Menor valor | sem poda | mais promissora | poda |
|---|---|---|---|
| Rosen | 1,0981e-17 | 3,8352e-15 | 3,02e-17 |
| Froth | 0 | 0 | 0 |
| BadscaP | 1,3452e-01 | 1,2736e-01 | 1,3274e-01 |
| BadscaB | 9,6507e+11 | 9,6513e+11 | 9,6513e+11 |
| Beale | 1,2570e-18 | 2,1263e-16 | 1,5511e-17 |
| JenSam | 1,2436e+02 | 1,2436e+02 | 1,2436e+02 |
| Helix | 1,7358e-16 | 4,5445e-16 | 4,1853e-16 |
| Bard | 8,2149e-03 | 8,2149e-03 | 8,2149e-03 |
| Gauss | 1,1279e-08 | 1,1279e-08 | 1,1279e-08 |
| Meyer | 5,5192e+07 | 3,8658e+07 | 6,6232e+06 |
| Gulf | 2,6717e-07 | 6,5014e-05 | 1,4565e-04 |
| Box | 1,3259e-20 | 1,1605e-08 | 1,543e-20 |
| Sing | 1,8877e-09 | 2,8514e-10 | 1,8272e-09 |
| Wood | 4,5421e-16 | 5,5723e-17 | 4,1877e-17 |
| KowOsb | 3,0751e-04 | 3,0751e-04 | 3,0751e-04 |
| BD | 8,5822e+04 | 8,5822e+04 | 8,5822e+04 |
| Osb1 | 1,1529 | 1,1529 | 1,1529 |
| Biggs | 2,4276e-01 | 2,4275e-01 | 2,4275e-01 |
| Osb2 | 4,0138e-02 | 4,0138e-02 | 4,0138e-02 |
| Watson | 1.6967e-06 | 4.6896e-06 | 3.0766e-06 |

Tabela 9: Menor valor de função encontrado pelo método SID-PSM ao resolver cada um dos problemas de MGH utilizandos D3, P1 e os diferentes tipos de poda

| # de Avaliações | SID-PSM1 | SID-PSM2 | BP1 | BP2 | Nelder-Mead |
|---|---|---|---|---|---|
| Rosen | 287 | 196 | 11026 | 3532 | 189 |
| Froth | 158 | 89 | 70 | 139 | 87 |
| BadscaP | 236 | 168 | 67 | 356 | 727 |
| BadscaB | 8379 | 1000000 | 1000000 | 3197 | 307 |
| Beale | 216 | 139 | 2202 | 128 | 132 |
| JenSam | 223 | 105 | 2708 | 325 | 98 |
| Helix | 270 | 222 | 14351 | 3145 | 295 |
| Bard | 263 | 186 | 81994 | 11674 | 274 |
| Gauss | 182 | 107 | 438 | 258 | 170 |
| Meyer | 1000004 | 1000000 | 1000000 | 497419 | 1819 |
| Gulf | 419308 | 220645 | 1000000 | 374205 | 536 |
| Box | 7488 | 189 | 16500 | 692 | 477 |
| Sing | 477 | 581 | 56456 | 53317 | 761 |
| Wood | 433 | 292 | 31846 | 455 | 601 |
| KowOsb | 522 | 230 | 46734 | 17178 | 317 |
| BD | 500 | 197 | 2792 | 1473 | 400 |
| Osb1 | 287 | 154 | 148 | 267 | 192 |
| Biggs | 6059 | 289893 | 1000002 | 1895 | 1015 |
| Osb2 | 3534 | 1462 | 123039 | 19322 | 4609 |
| Watson | 16775 | 2188 | 1000004 | 459659 | 5344 |

Tabela 10: Número de avaliações de função realizadas pelos métodos SID-PSM1, SID-PSM2, BP1, BP2 e Nelder-Mead para resolver problemas de dimensão fixa de MGH

| # de Avaliações | SID-PSM1 | SID-PSM2 | BP1 | BP2 | Nelder-Mead |
|---|---|---|---|---|---|
| Rosex m = n = 12 | 8749 | 1895 | 254154 | 24144 | 18401 |
| Rosex m = n = 20 | 33510 | 4614 | 517808 | 39808 | 29007 |
| Singx m = n = 12 | 4864 | 2197 | 279141 | 173310 | 3284 |
| Singx m = n = 20 | 24897 | 4357 | 542012 | 283806 | 13606 |
| Pen I m = 13, n = 12 | 14895 | 3272 | 411284 | 480784 | 10743 |
| Pen I m = 21, n = 20 | 46687 | 5738 | 1000002 | 509604 | 61642 |
| Pen II m = 24, n = 12 | 96879 | 20937 | 1000001 | 1000001 | 14092 |
| Pen II m = 40, n = 20 | 184782 | 12438 | 1000011 | 730558 | 37370 |
| Vardim m = 14, n = 12 | 8159 | 793 | 19336 | 29918 | 8662 |
| Vardim m = 22, n = 20 | 63555 | 2452 | 192037 | 206980 | 15580 |
| Trig m = n = 12 | 1132 | 777 | 5541 | 1279 | 3511 |
| Trig m = n = 20 | 2500 | 2278 | 17281 | 1967 | 36771 |
| BV m = n = 12 | 1002 | 922 | 61607 | 15629 | 1974 |
| BV m = n = 20 | 2078 | 9529 | 431406 | 125031 | 15295 |
| IE m = n = 12 | 859 | 389 | 836 | 938 | 1940 |
| IE m = n = 20 | 1688 | 744 | 1920 | 1651 | 22820 |
| Trid m = n = 12 | 2564 | 526 | 1716 | 17046 | 1840 |
| Trid m = n = 20 | 13306 | 1118 | 40575 | 88863 | 5330 |
| Band m = n = 12 | 1625 | 586 | 850 | 18987 | 2291 |
| Band m = n = 20 | 67971 | 1746 | 2631 | 1000011 | 12338 |
| Lin m = 24, n = 12 | 581 | 332 | 305 | 534 | 552 |
| Lin m = 40, n = 20 | 925 | 459 | 451 | 862 | 937 |
| Lin1 m = 24, n = 12 | 667 | 317 | 353 | 604 | 611 |
| Lin1 m = 40, n = 20 | 1046 | 506 | 587 | 1035 | 1126 |
| Lin1 m = 200, n = 20 | 1045 | 503 | 547 | 997 | 1167 |
| Lin0 m = 24, n = 12 | 751 | 321 | 368 | 607 | 578 |
| Lin0 m = 40, n = 20 | 1070 | 622 | 636 | 997 | 1094 |
| Lin0 m = 200, n = 20 | 1290 | 565 | 610 | 941 | 1130 |

Tabela 11: Número de avaliações de função realizadas pelos métodos SID-PSM1, SID-PSM2, BP1, BP2 e Nelder-Mead para resolver problemas de dimensão livre de MGH

| Menor valor | SID-PSM1 | SID-PSM2 | BP1 | BP2 | Nelder-Mead |
|---|---|---|---|---|---|
| Rosen | 5,7629e-017 | 5,7629e-017 | 9,1263e-007 | 1,4699e-009 | 4,1940e-014 |
| Froth | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 |
| BadscaP | 3,3184e-004 | 3,3184e-004 | 1,3514e-001 | 1,7152e-006 | 1,2228e-021 |
| BadscaB | 8,4295e-013 | 8,4295e-013 | 9,5888e+011 | 8,5843e-003 | 4,2266e-014 |
| Beale | 1,5973e-018 | 1,5973e-018 | 5,7101e-009 | 0,0000e+000 | 1,7714e-014 |
| JenSam | 1,2436e+002 | 1,2436e+002 | 1,2436e+002 | 1,2436e+002 | 1,2436e+002 |
| Helix | 3,9727e-018 | 3,9727e-018 | 3,0241e-007 | 1,0613e-008 | 3,1602e-013 |
| Bard | 8,2149e-003 | 8,2149e-003 | 8,2150e-003 | 8,2149e-003 | 8,2149e-003 |
| Gauss | 1,1279e-008 | 1,1279e-008 | 1,1337e-008 | 1,1280e-008 | 1,1279e-008 |
| Meyer | 4,6695e+002 | 4,6695e+002 | 1,8481e+008 | 1,6780e+004 | 8,7946e+001 |
| Gulf | 1,0486e-017 | 1,0486e-017 | 2,4524e-003 | 3,6305e-009 | 2,0393e-017 |
| Box | 6,4782e-028 | 6,4782e-028 | 4,6997e-011 | 0,0000e+000 | 9,8127e-017 |
| Sing | 6,1195e-015 | 6,1195e-015 | 4,8412e-007 | 5,2989e-008 | 3,7045e-021 |
| Wood | 5,0894e-016 | 5,0894e-016 | 4,0008e-007 | 3,6835e-011 | 2,7519e-013 |
| KowOsb | 3,0751e-004 | 3,0751e-004 | 3,0751e-004 | 3,0751e-004 | 3,0751e-004 |
| BD | 8,5822e+004 | 8,5822e+004 | 8,5822e+004 | 8,5822e+004 | 8,5822e+004 |
| Osb1 | 1,1529e+000 | 1,1529e+000 | 1,1529e+000 | 1,1529e+000 | 1,1529e+000 |
| Biggs | 1,9088e-014 | 1,9088e-014 | 2,4306e-001 | 3,0637e-001 | 5,6556e-003 |
| Osb2 | 4,0138e-002 | 4,0138e-002 | 4,0138e-002 | 4,0138e-002 | 4,0138e-002 |
| Watson | 3,7917e-005 | 3,7917e-005 | 2,3050e-004 | 3,9131e-006 | 1,7000e-004 |

Tabela 12: Menor valor de função encontrado realizadas pelos métodos SID-PSM1, SID-PSM2, BP1, BP2 e Nelder-Mead para resolver problemas de dimensão fixa de MGH

| Menor valor | SID-PSM1 | SID-PSM2 | BP1 | BP2 | Nelder-Mead |
|---|---|---|---|---|---|
| Rosex m = n = 12 | 1,3370e-010 | 1,3370e-010 | 1,3933e-006 | 8,8191e-009 | 3,7562e+000 |
| Rosex m = n = 20 | 3,0081e-008 | 3,0081e-008 | 2,2349e-006 | 1,4699e-008 | 2,4634e+001 |
| Singx m = n = 12 | 8,5956e-010 | 8,5956e-010 | 1,5355e-006 | 1,5897e-007 | 9,5225e-006 |
| Singx m = n = 20 | 2,0885e-009 | 2,0885e-009 | 3,7795e-006 | 2,6495e-007 | 7,3816e-005 |
| Pen I m = 13, n = 12 | 8,7863e-005 | 8,7863e-005 | 8,7859e-005 | 8,7858e-005 | 9,7424e-005 |
| Pen I m = 21, n = 20 | 1,5778e-004 | 1,5778e-004 | 1,5778e-004 | 1,5778e-004 | 1,7276e-004 |
| Pen II m = 24, n = 12 | 6,2193e-004 | 6,2193e-004 | 6,1752e-004 | 6,1785e-004 | 6,2378e-004 |
| Pen II m = 40, n = 20 | 6,3937e-003 | 6,3937e-003 | 6,3903e-003 | 6,3897e-003 | 6,4302e-003 |
| Vardim m = 14, n = 12 | 3,1135e-010 | 3,1135e-010 | 3,7345e-008 | 2,1459e-008 | 4,5329e+000 |
| Vardim m = 22, n = 20 | 1,6500e-008 | 1,6500e-008 | 6,1844e-007 | 2,6920e-007 | 1,4634e+001 |
| Trig m = n = 12 | 3,8281e-004 | 3,8281e-004 | 3,0271e-005 | 3,8281e-004 | 6,2723e-007 |
| Trig m = n = 20 | 1,0851e-003 | 1,0851e-003 | 6,8622e-006 | 2,4771e-003 | 1,3492e-006 |
| BV m = n = 12 | 4,4791e-018 | 4,4791e-018 | 1,6459e-008 | 2,8913e-009 | 2,4817e-013 |
| BV m = n = 20 | 2,0992e-010 | 2,0992e-010 | 2,2072e-007 | 3,3530e-008 | 9,9779e-013 |
| IE m = n = 12 | 9,1433e-015 | 9,1433e-015 | 3,5390e-012 | 2,8998e-012 | 1,1014e-011 |
| IE m = n = 20 | 2,9859e-011 | 2,9859e-011 | 9,5698e-012 | 7,5604e-012 | 1,3676e-011 |
| Trid m = n = 12 | 1,0769e+000 | 1,0769e+000 | 1,7794e-009 | 1,0769e+000 | 1,2260e-010 |
| Trid m = n = 20 | 1,1142e+000 | 1,1142e+000 | 7,1253e-001 | 1,1142e+000 | 8,2382e-010 |
| Band m = n = 12 | 3,0546e+000 | 3,0546e+000 | 2,1916e-010 | 3,0546e+000 | 3,1833e-010 |
| Band m = n = 20 | 3,0762e+000 | 3,0762e+000 | 9,0233e-010 | 3,0762e+000 | 3,2082e-010 |
| Lin m = 24, n = 12 | 0,0000e+000 | 0,0000e+000 | 1,1846e-015 | 0,0000e+000 | 1,3182e-015 |
| Lin m = 40, n = 20 | 0,0000e+000 | 0,0000e+000 | 3,7032e-014 | 0,0000e+000 | 3,3102e-015 |
| Lin1 m = 24, n = 12 | 5,6327e+000 | 5,6327e+000 | 5,6327e+000 | 5,6327e+000 | 5,6327e+000 |
| Lin1 m = 40, n = 20 | 9,6296e+000 | 9,6296e+000 | 9,6296e+000 | 9,6296e+000 | 9,6296e+000 |
| Lin1 m = 200, n = 20 | 4,9627e+001 | 4,9627e+001 | 4,9626e+001 | 4,9626e+001 | 4,9626e+001 |
| Lin0 m = 24, n = 12 | 7,1333e+000 | 7,1333e+000 | 7,1333e+000 | 7,1333e+000 | 7,1333e+000 |
| Lin0 m = 40, n = 20 | 1,1130e+001 | 1,1130e+001 | 1,1130e+001 | 1,1130e+001 | 1,1130e+001 |
| Lin0 m = 200, n = 20 | 5,1126e+001 | 5,1126e+001 | 5,1126e+001 | 5,1126e+001 | 5,1126e+001 |

Tabela 13: Menor valor de função encontrado realizadas pelos métodos SID-PSM1, SID-PSM2, BP1, BP2 e Nelder-Mead para resolver problemas de dimensão livre de MGH

| # de Avaliações | $x_1^*$ | $x_2^*$ | $x_3^*$ | $x_4^*$ |
|---|---|---|---|---|
| Rosen | 204 | 237 | 232 | 178 |
| Froth | 61 | 77 | 79 | 79 |
| BadscaP | 578 | 450 | 1123 | 4915 |
| BadscaB | 509 | 421 | 584 | 515 |
| Beale | 205 | 112 | 97 | 97 |
| JenSam | 128 | 202 | 112 | 112 |
| Helix | 223 | 203 | 316 | 256 |
| Bard | 177 | 120 | 281 | 281 |
| Gauss | 79 | 90 | 98 | 98 |
| Meyer | 48840 | 22847 | 92225 | 100001 |
| Gulf | 1144 | 1063 | 1104 | 1354 |
| Box | 449 | 227 | 145 | 150 |
| Sing | 537 | 394 | 282 | 570 |
| Wood | 317 | 606 | 441 | 368 |
| KowOsb | 317 | 295 | 232 | 253 |
| BD | 233 | 245 | 243 | 219 |
| Osb1 | 125 | 97 | 122 | 122 |
| Biggs | 1703 | 1058 | 2490 | 1347 |
| Osb2 | 3867 | 3647 | 4981 | 2909 |
| Watson | 10181 | 3536 | 5955 | 4621 |

Tabela 14: Número de avaliações de função realizadas pelo método SID-PSM1 utilizando os parâmetros dados por $x_1^*$, $x_2^*$, $x_3^*$ e $x_4^*$ para resolver problemas de dimensão fixa de MGH

| # de Avaliações | $x_1^*$ | $x_2^*$ | $x_3^*$ | $x_4^*$ |
|---|---|---|---|---|
| Rosex m = n = 12 | 6713 | 4695 | 3419 | 3193 |
| Rosex m = n = 20 | 41270 | 19264 | 12438 | 10618 |
| Singx m = n = 12 | 5258 | 3408 | 2734 | 2151 |
| Singx m = n = 20 | 24082 | 8178 | 7518 | 6791 |
| Pen I m = 13, n = 12 | 10994 | 13774 | 8014 | 9180 |
| Pen I m = 21, n = 20 | 38209 | 30692 | 18320 | 21657 |
| Pen II m = 24, n = 12 | 2796 | 52291 | 4719 | 1631 |
| Pen II m = 40, n = 20 | 79998 | 21583 | 17433 | 17473 |
| Vardim m = 14, n = 12 | 8436 | 4912 | 6086 | 5903 |
| Vardim m = 22, n = 20 | 82246 | 45020 | 33865 | 35177 |
| Trig m = n = 12 | 715 | 807 | 834 | 845 |
| Trig m = n = 20 | 2864 | 289 | 1705 | 1594 |
| BV m = n = 12 | 525 | 531 | 602 | 576 |
| BV m = n = 20 | 1894 | 922 | 1299 | 1305 |
| IE m = n = 12 | 617 | 462 | 530 | 530 |
| IE m = n = 20 | 2036 | 1357 | 1001 | 1405 |
| Trid m = n = 12 | 768 | 1146 | 1080 | 1034 |
| Trid m = n = 20 | 2419 | 3853 | 6175 | 5811 |
| Band m = n = 12 | 2657 | 2356 | 1687 | 1580 |
| Band m = n = 20 | 2304 | 2857 | 2986 | 2142 |
| Lin m = 24, n = 12 | 159 | 135 | 207 | 207 |
| Lin m = 40, n = 20 | 272 | 207 | 270 | 270 |
| Lin1 m = 24, n = 12 | 312 | 247 | 267 | 267 |
| Lin1 m = 40, n = 20 | 431 | 353 | 318 | 366 |
| Lin1 m = 200, n = 20 | 450 | 293 | 281 | 281 |
| Lin0 m = 24, n = 12 | 291 | 223 | 327 | 307 |
| Lin0 m = 40, n = 20 | 480 | 270 | 365 | 408 |
| Lin0 m = 200, n = 20 | 375 | 317 | 319 | 319 |

Tabela 15: Número de avaliações de função realizadas pelo método SID-PSM1 utilizando os parâmetros dados por $x_1^*$, $x_2^*$, $x_3^*$ e $x_4^*$ para resolver problemas de dimensão livre de MGH

| Menor valor | $x_1^*$ | $x_2^*$ | $x_3^*$ | $x_4^*$ |
|---|---|---|---|---|
| Rosen | 2,1349e-016 | 2,1349e-016 | 4,7609e-017 | 9,2872e-018 |
| Froth | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 |
| BadscaP | 2,2765e-004 | 2,2765e-004 | 3,3184e-004 | 8,7713e-005 |
| BadscaB | 5,1046e-016 | 5,1046e-016 | 2,1891e-013 | 5,2554e-017 |
| Beale | 1,9765e-015 | 1,9765e-015 | 1,5973e-018 | 1,5973e-018 |
| JenSam | 1,2436e+002 | 1,2436e+002 | 1,2436e+002 | 1,2436e+002 |
| Helix | 2,9016e-017 | 2,9016e-017 | 3,7845e-017 | 9,5431e-017 |
| Bard | 8,2149e-003 | 8,2149e-003 | 8,2149e-003 | 8,2149e-003 |
| Gauss | 1,1279e-008 | 1,1279e-008 | 1,1279e-008 | 1,1279e-008 |
| Meyer | 1,7105e+003 | 1,7105e+003 | 6,4792e+002 | 2,0849e+003 |
| Gulf | 5,1944e-017 | 5,1944e-017 | 3,3097e-016 | 1,9783e-016 |
| Box | 6,6040e-022 | 6,6040e-022 | 4,8294e-031 | 2,6641e-029 |
| Sing | 4,4262e-017 | 4,4262e-017 | 4,1023e-016 | 8,1825e-022 |
| Wood | 1,1280e-013 | 1,1280e-013 | 2,0932e-017 | 1,2565e-014 |
| KowOsb | 3,0751e-004 | 3,0751e-004 | 3,0751e-004 | 3,0751e-004 |
| BD | 8,5822e+004 | 8,5822e+004 | 8,5822e+004 | 8,5822e+004 |
| Osb1 | 1,1529e+000 | 1,1529e+000 | 1,1529e+000 | 1,1529e+000 |
| Biggs | 7,8742e-016 | 7,8742e-016 | 2,6439e-013 | 3,2346e-012 |
| Osb2 | 4,0138e-002 | 4,0138e-002 | 4,0138e-002 | 4,0138e-002 |
| Watson | 1,3684e-006 | 1,3684e-006 | 1,8406e-005 | 1,3967e-005 |

Tabela 16: Menor valor de função encontrado realizadas pelo método SID-PSM1 utilizando os parâmetros dados por $x_1^*$, $x_2^*$, $x_3^*$ e $x_4^*$ para resolver os problemas de tamanho fixo de MGH

| Menor valor | $x_1^*$ | $x_2^*$ | $x_3^*$ | $x_4^*$ |
|---|---|---|---|---|
| Rosex m = n = 12 | 1,6211e-008 | 1,6211e-008 | 9,7624e-012 | 1,2817e-010 |
| Rosex m = n = 20 | 3,2858e-007 | 3,2858e-007 | 2,2831e-010 | 7,1620e-009 |
| Singx m = n = 12 | 2,2495e-006 | 2,2495e-006 | 2,2948e-011 | 2,2443e-008 |
| Singx m = n = 20 | 1,5672e-007 | 1,5672e-007 | 3,3618e-009 | 1,8259e-008 |
| Pen I m = 13, n = 12 | 8,7859e-005 | 8,7859e-005 | 8,7858e-005 | 8,7858e-005 |
| Pen I m = 21, n = 20 | 1,5794e-004 | 1,5794e-004 | 1,5778e-004 | 1,5778e-004 |
| Pen II m = 24, n = 12 | 6,3114e-004 | 6,3114e-004 | 6,2474e-004 | 6,2380e-004 |
| Pen II m = 40, n = 20 | 6,3949e-003 | 6,3949e-003 | 6,3955e-003 | 6,3963e-003 |
| Vardim m = 14, n = 12 | 1,1983e-009 | 1,1983e-009 | 1,0403e-011 | 4,4049e-012 |
| Vardim m = 22, n = 20 | 4,2687e-006 | 4,2687e-006 | 1,7284e-005 | 1,9693e-010 |
| Trig m = n = 12 | 1,0362e-012 | 1,0362e-012 | 3,8281e-004 | 3,8281e-004 |
| Trig m = n = 20 | 4,4712e-011 | 4,4712e-011 | 1,0851e-003 | 1,0851e-003 |
| BV m = n = 12 | 2,3958e-014 | 2,3958e-014 | 4,4791e-018 | 5,7830e-018 |
| BV m = n = 20 | 2,2322e-009 | 2,2322e-009 | 2,0992e-010 | 1,9748e-013 |
| IE m = n = 12 | 1,6531e-013 | 1,6531e-013 | 6,3419e-019 | 6,3419e-019 |
| IE m = n = 20 | 3,1073e-011 | 3,1073e-011 | 2,1020e-011 | 1,5413e-013 |
| Trid m = n = 12 | 5,9358e-010 | 5,9358e-010 | 1,0769e+000 | 1,0769e+000 |
| Trid m = n = 20 | 1,9509e-012 | 1,9509e-012 | 1,1142e+000 | 1,1142e+000 |
| Band m = n = 12 | 3,0546e+000 | 3,0546e+000 | 3,0546e+000 | 3,0546e+000 |
| Band m = n = 20 | 3,0762e+000 | 3,0762e+000 | 3,0762e+000 | 3,0762e+000 |
| Lin m = 24, n = 12 | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 |
| Lin m = 40, n = 20 | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 | 0,0000e+000 |
| Lin1 m = 24, n = 12 | 5,6327e+000 | 5,6327e+000 | 5,6327e+000 | 5,6327e+000 |
| Lin1 m = 40, n = 20 | 9,6296e+000 | 9,6296e+000 | 9,6296e+000 | 9,6296e+000 |
| Lin1 m = 200, n = 20 | 4,9626e+001 | 4,9626e+001 | 4,9627e+001 | 4,9627e+001 |
| Lin0 m = 24, n = 12 | 7,1333e+000 | 7,1333e+000 | 7,1333e+000 | 7,1333e+000 |
| Lin0 m = 40, n = 20 | 1,1130e+001 | 1,1130e+001 | 1,1130e+001 | 1,1130e+001 |
| Lin0 m = 200, n = 20 | 5,1126e+001 | 5,1126e+001 | 5,1126e+001 | 5,1126e+001 |

Tabela 17: Menor valor de função encontrado realizadas pelo método SID-PSM1 utilizando os parâmetros dados por $x_1^*$, $x_2^*$, $x_3^*$ e $x_4^*$ para resolver os problemas de tamanho livre de MGH

| # de Avaliações | $x_0$ | $x_1^*$ | $x_2^*$ | $x_3^*$ | $x_4^*$ |
|---|---|---|---|---|---|
| Lin m = 45, n = 9 | 217 | 329 | 214 | 217 | 217 |
| Lin1 m = 35, n = 7 | 213 | 172 | 131 | 213 | 213 |
| Lin0 m = 35, n = 7 | 201 | 202 | 132 | 220 | 220 |
| Rosen m = n = 2 | 2620 | 206 | 634 | 404 | 363 |
| Helix m = n = 2 | 464 | 356 | 768 | 317 | 473 |
| Sing m = n = 4 | 5872 | 460 | 1599 | 5548 | 766 |
| Froth m = n = 2 | 141 | 102 | 128 | 138 | 120 |
| Bard m = 15, n = 3 | 377 | 165 | 420 | 180 | 178 |
| KowOsb m = 11, n = 4 | 1010 | 1042 | 501 | 544 | 1653 |
| Meyer m = 16, n = 3 | 7442 | 182 | 167 | 7520 | 6372 |
| Watson m = 31, n = 6 | 77450 | 1473 | 864 | 65821 | 38527 |
| Box m = 10, n = 3 | 17384 | 562 | 258 | 3548 | 1739 |
| JenSam m = 10, n = 2 | 115 | 118 | 122 | 108 | 116 |
| BD m = 20, n = 4 | 202 | 142 | 175 | 202 | 226 |
| Cheby m = n = 6 | 607 | 438 | 512 | 448 | 689 |
| Brown m = n = 10 | 10801 | 2648 | 3831 | 11682 | 11779 |
| Osb1 m = 33, n = 5 | 47810 | 774 | 169 | 14584 | 31937 |
| Osb2 m = 65, n = 11 | 76798 | 5914 | 8779 | 50050 | 72303 |
| Bdqrtic m = n = 8 | 360 | 491 | 373 | 313 | 316 |
| Cube m = n = 5 | 117 | 123 | 171 | 117 | 117 |
| Mancino m = n = 5 | 320 | 345 | 265 | 304 | 335 |
| Heart8ls m = n = 8 | 13429 | 4448 | 4007 | 7493 | 12320 |

Tabela 18: Número de avaliações de função realizadas pelo método SID-PSM1 utilizando os parâmetros dados por $x_0$, $x_1^*$, $x_2^*$, $x_3^*$ e $x_4^*$ para resolver os problemas em [22]

| # de Avaliações | $x_0$ | $x_1^*$ | $x_2^*$ | $x_3^*$ | $x_4^*$ |
|---|---|---|---|---|---|
| Lin m = 45, n = 9 | 6 | 6 | 6 | 6 | 6 |
| Lin1 m = 35, n = 7 | 2,8949 | 2,8949 | 2,8949 | 2,8949 | 2,8949 |
| Lin0 m = 35, n = 7 | 3,1433 | 3,1433 | 3,1433 | 3,1433 | 3,1433 |
| Rosen m = n = 2 | 1,4980e-03 | 9,0807e-04 | 1,2509e-02 | 1,6624e-03 | 1,1383e-03 |
| Helix m = n = 2 | 1,5224e-03 | 4,2732e-03 | 2,0226e-03 | 1,2284e-03 | 1,0183e-03 |
| Sing m = n = 4 | 3,2039e-03 | 6,4713e-03 | 5,6258e-03 | 2,8553e-03 | 1,0325e-03 |
| Froth m = n = 2 | 6,9989 | 6,9989 | 6,9989 | 6,9989 | 6,9989 |
| Bard m = 15, n = 3 | 9,0636e-02 | 9,0636e-02 | 9,0729e-02 | 9,0639e-02 | 9,0636e-02 |
| KowOsb m = 11, n = 4 | 1,7545e-02 | 1,7539e-02 | 1,7537e-02 | 1,7536e-02 | 1,7540e-02 |
| Meyer m = 16, n = 3 | 7,4812e+03 | 5,1040e+03 | 4,9880e+03 | 7,6773e+03 | 6,4439e+03 |
| Watson m = 31, n = 6 | 6,3479e-02 | 7,4270e-02 | 5,3154e-02 | 6,7054e-03 | 6,8497e-02 |
| Box m = 10, n = 3 | 1,1662e-03 | 1,2536e-02 | 2,4179e-04 | 9,2911e-04 | 9,1736e-04 |
| JenSam m = 10, n = 2 | 1,1151e+01 | 1,1151e+01 | 1,1151e+01 | 1,1151e+01 | 1,1151e+01 |
| BD m = 20, n = 4 | 2,9295e+02 | 2,9295e+02 | 2,9295e+02 | 2,9295e+02 | 2,9295e+02 |
| Cheby m = n = 6 | 1,6536e-04 | 2,5070e-04 | 2,5660e-04 | 1,6676e-04 | 1,1754e-04 |
| Brown m = n = 10 | 4,4218e-03 | 2,1231e-03 | 8,0507e-03 | 4,4680e-03 | 3,5859e-03 |
| Osb1 m = 33, n = 5 | 9,3617e-01 | 4,6473e-01 | 1,0517 | 1,1231 | 9,3896e-01 |
| Osb2 m = 65, n = 11 | 2,0039e-01 | 2,0034e-01 | 2,0035e-01 | 2,0044e-01 | 20039e-01 |
| Bdqrtic m = n = 8 | 3,1998 | 3,1998 | 3,1998 | 3,1998 | 3,1998 |
| Cube m = n = 5 | 0 | 2,6862e-04 | 3,6902e-03 | 0 | 0 |
| Mancino m = n = 5 | 4,2324e-03 | 2,1923e-02 | 2,8594e-02 | 3,8492e-03 | 4,1139e-03 |
| Heart8ls m = n = 8 | 1,0983e-02 | 3,5251e-02 | 3,5542e-02 | 8,4200e-03 | 1,2180e-02 |

Tabela 19: Menor valor de função encontrado realizadas pelo método SID-PSM1 utilizando os parâmetros dados por $x_0$, $x_1^*$, $x_2^*$, $x_3^*$ e $x_4^*$ para resolver os problemas em [22]

# Anexo B - Base positiva com ângulos uniformes

Inicialmente neste Anexo demonstraremos alguns resultados que serão necessários para a construção da base positiva com ângulos de amplitude uniforme.

**Proposição 2** *Seja* $V = \{v_1, \ldots, v_n\} \subset \mathbb{R}^n$ *de modo que,* $\|v_j\| = 1 \ \forall j = 1, \ldots, n$ *e* $v_j^\top v_i = a \neq \pm 1$, $\forall i \neq j$, *então* $V$ *é uma base para o* $\mathbb{R}^n$.

Demonstração: Como $a \neq \pm 1$, claramente os vetores são linearmente independentes, logo, como temos um conjunto com $n$ vetores linearmente independentes no $\mathbb{R}^n$, este conjunto é uma base para o $\mathbb{R}^n$. ■

**Proposição 3** *Seja* $V = \{v_1, \ldots, v_n\} \subset \mathbb{R}^n$ *uma base para o* $\mathbb{R}^n$, *então* $\bar{V} = \{v_1, \ldots, v_n, -\sum_{i=1}^{n} v_i\}$ *é uma base positiva para o* $\mathbb{R}^n$

Demonstração: Seja $x \in \mathbb{R}^n$, como $V$ é base, isso implica que existem $\alpha_i$, $i = 1, \ldots, n$, tais que

$$x = \sum_{i=1}^{n} \alpha_i v_i.$$

Se $\alpha_i \geq 0 \ \forall i$, $V$ satisfaz a condição de ser base positiva, logo $\bar{V}$ também satisfaz. Caso contrário, seja $j = arg \min_{i=1,\ldots,n} \alpha_i$ e $\beta = -\alpha_j$, assim

$$\bar{\alpha}_i = \alpha_i + \beta \geq 0 \qquad \forall i = 1, \ldots, n,$$

daí temos

$$x = \sum_{i=1}^{n} \bar{\alpha}_i v_i + \beta(-\sum_{i=1}^{n} v_i\}),$$

ou seja, conseguimos construir $x$ como uma combinação linear positiva dos elementos de $\bar{V}$, logo $\bar{V}$ é base positiva, e ainda, como tem $n+1$ elementos é minimal. ■

Com esses resultados em mãos podemos começar a construir nossa base positiva com ângulos uniformes. Tomemos $n+1$ vetores $v_1, \ldots, v_{n+1}$ no $\mathbb{R}^n$ de modo que o ângulo entre os pares $v_i$, $v_j$ $(i \neq j)$ tenham a mesma amplitude $\alpha$. Assumindo que os $n+1$ vetores estão normalizados, essa imposição é expressa como

$$a = \cos(\alpha) = v_i^\top v_j, \qquad i, j \in \{1, \ldots, n+1\}, i \neq j, \tag{1}$$

onde $a \neq 1$.

**Proposição 4** *O valor de a na equação (1) é* $-1/n$.

Demonstração: Fixemos o índice $j$ e seja $\mathcal{I}_j = \{1, \ldots, n+1\} \setminus \{j\}$. Sabemos, pela Proposição 2, que o conjunto $\{v_i : i \in \mathcal{I}_j\}$ forma uma base para o $\mathbb{R}^n$. Logo existem constantes $\gamma_i$, $i \in \mathcal{I}_j$ tais que

$$v_j = \sum_{i \in \mathcal{I}_j} \gamma_i v_i. \tag{2}$$

Temos que $1 = \|v_j\|^2 = v_j^\top v_j$, assim substituindo um dos $v_j$ por (2) obtemos

$$\|v_j\|^2 = v_j^\top (\sum_{k \in \mathcal{I}_j} \gamma_k v_k) = \sum_{k \in \mathcal{I}_j} \gamma_k v_j^\top v_k = \sum_{k \in \mathcal{I}_j} \gamma_k a.$$

Assim

$$\sum_{k \in \mathcal{I}_j} \gamma_k = \frac{1}{a} \qquad \forall j = 1, \ldots, n+1.$$

Claramente esse sistema é determinado e por inspeção facilmente vemos que sua solução é

$$\gamma_i = \gamma_i = \frac{1}{an} \qquad \forall i, j = 1, \ldots, n+1. \tag{3}$$

Por outro lado sabemos que, para todo $k \neq j$, $v_k^\top v_j = a$, novamente utilizando (2), obtemos

$$v_k^\top v_j = v_k^\top (\sum_{i \in \mathcal{I}_j} \gamma_i v_i) = \sum_{i \in \mathcal{I}_j} \gamma_i v_k^\top v_i = \sum_{i \in \mathcal{I}_j \setminus \{k\}} \gamma_i a + \gamma_k.$$

Por (3) temos então

$$a = v_k^\top v_j = (n-1)\frac{a}{an} + \frac{1}{an}.$$

Multiplicando a igualdade por $an$ teremos

$$a^2 n - (n-1)a - 1 = 0.$$

Esta equação possui raiz em $a = 1$ e $a = -1/n$. Como, por hipótese, $a \neq 1$, concluímos a demonstração. ■

Agora que sabemos a forma como devem se comportar os vetores $v_i$'s, procuramos uma forma de encontrá-los. Primeiramente vamos encontrar a matriz $V = [v_1 \ldots v_n]$, $n \times n$, tal que

$$V^\top V = A$$

onde $A$ é a matriz dada por

$$A = \begin{bmatrix} 1 & -1/n & \ldots & -1/n \\ -1/n & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ -1/n & -1/n & \ldots & 1 \end{bmatrix}.$$

$A$ é simétrica e definida positiva. Logo, podemos utilizar a decomposição de Cholesky

$$A = GG^\top,$$

onde $G \in I\!R^{n\times n}$ é uma maitrz triangular inferior com diagonal positiva. Dada a decomposição, vemos claramente que uma escolha para $V$ é

$$V = [v_1 \dots v_n] = G^\top.$$

O vetor $v_{n+1}$ é calculado por

$$v_{n+1} = -\sum_{i=1}^{n} v_i. \tag{4}$$

Nos resta agora mostrar que $v_{n+1}^\top v_{n+1} = 1$ e $v_{n+1}^\top v_j = -1/n$ para todo $j = 1, \dots, n$.

Da própria equação (4) temos

$$v_{n+1}^\top v_{n+1} = (\sum_{i=1}^{n} v_i)^\top (\sum_{j=1}^{n} v_j) = \sum_{i=1}^{n}\sum_{j=1}^{n} v_i^\top v_j =$$

$$= \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} v_i^\top v_j + \sum_{i=1}^{n} v_i^\top v_i = (1-n) + n,$$

daí

$$v_{n+1}^\top v_{n+1} = 1.$$

Novamente utilizando a (4) temos, fixando $j$ arbitrário,

$$v_{n+1}^\top v_j = (-\sum_{i=1}^{n} v_i)^\top v_j = -\sum_{i=1}^{n} v_i^\top v_j = \frac{n-1}{n} - 1 = -\frac{1}{n}.$$

Assim, encontramos um conjunto $\{v_1, \dots, v_{n+1}\}$ no $I\!R^n$ com ângulos de tamanho uniforme entre si. E pela Proposição 3 este conjunto é uma base minimal para o $I\!R^n$.

# Bibliografia


[1] M.A. Abramson, C. Audet e J.E. Dennis, Generalized pattern searches with derivative information, *Math. Program.*,100, pp. 3-25, 2004.

[2] C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, *SIAM Journal on Optimization*, 3, pp. 642-664, 2006.

[3] M.S. Bazaraa, H.D. Sherali e C.M. Shetty, Nonlinear Programming: Theory and Algorithms, *John Wiley & Sons, Inc.*, New York, 1979.

[4] A.R. Conn, K. Scheinberg e L.N. Vicente, Geometry of sample sets in derivative free optimization: Polynomial regression and underdetermined interpolation, *IMA J. Numer. Anal.*, 28, pp. 721-748, 2008.

[5] A.L. Custódio, H. Rocha e L.N. Vicente, Incorporating minimum Frobenius norm models in direct search, *Computational Optimization and Applications*, pp. 265-278, 2010.

[6] A.L. Custódio, K. Scheinberg e L.N. Vicente, Introduction to Derivative-Free Optimization, *MPS-SIAM Series on Optimization* 8, 2009.

[7] 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.

[8] W.C. Davidson, Variable metric method or minimization, *SIAM Journal Optimization*, 1, pp. 1-17, 1991.

[9] J.E. Dennis e R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, *SIAM Classics in Applied Mathematics* 16, 1987.

[10] E.D. Dolan e J.J. Moré, Benchmarking optimization software with performance profiles, *Springer-Verlag*, 2001.

[11] C.T. Kelley, Iterative Methods for Optimization, *SIAM* 7, 1999.

[12] 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.

[13] R.M. Lewis, L.V. Torczon e M.W. Trosset, Direct search methods: then and now, *Journal of Computational and Applied Mathematics*, 124, Issues 1-2, pp. 191-207, 2000.

[14] D.G. Luenberger, Linear and Nonlinear Programming, 2ª edição, New York, *Addison - Wesley Publishing Company*, 1986.

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

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

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

[18] V. Torczon, On the convergence of pattern search algorithm, *SIAM Journal on Optimization*, 7, pp. 1-25, 1997.

[19] V. Torczon, On the convergence of the multidirectional search algorithm, *SIAM Journal on Optimization*, 1, pp. 123-145, 1991.

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

[21] http://ferrari.dmat.fct.unl.pt/personal/alcustodio/sid_psm_1.2.tar.gz acessado em 04/07/2012.

[22] http://www.mcs.anl.gov/∼more/dfo/ acessado em 03/10/2012.