Universidade Federal do Espírito Santo

Centro Universitário Norte do Espírito Santo

Curso de Bacharelado em Matemática Industrial

Elivandro Oliveira Grippa

MÉTODOS COMPUTACIONAIS PARA OTIMIZAÇÃO IRRESTRITA E COM RESTRIÇÕES SIMPLES ADEQUADOS A PROBLEMAS DE LARGA ESCALA

São Mateus

2022

Elivandro Oliveira Grippa

MÉTODOS COMPUTACIONAIS PARA OTIMIZAÇÃO IRRESTRITA E COM RESTRIÇÕES SIMPLES ADEQUADOS A PROBLEMAS DE LARGA ESCALA

Trabalho submetido ao Colegiado do Curso de Bacharelado em Matemática Industrial da UFES (Campus São Mateus), como requisito parcial para a obtenção do grau de Bacharel em Matemática.

São Mateus

2022

Elivandro Oliveira Grippa

MÉTODOS COMPUTACIONAIS PARA OTIMIZAÇÃO IRRESTRITA E COM RESTRIÇÕES SIMPLES ADEQUADOS A PROBLEMAS DE LARGA ESCALA

Trabalho submetido ao Colegiado do Curso de Bacharelado em Matemática Industrial da UFES (Campus São Mateus), como requisito parcial para a obtenção do grau de Bacharel em Matemática.

Aprovada em 24 de Março de 2022.

Comissão Examinadora

Prof. Dr. Leonardo Delarmelina Secchin
Universidade Federal do Espírito Santo
Orientador

Prof. Dr. Rui Marques Carvalho
Instituto Federal do Piauí

Prof. Dr. Isaac Pinheiro dos Santos
Universidade Federal do Espírito Santo

Agradecimentos

Antes de tudo agradeço a Deus, por me dar muita força e coragem para trilhar esse caminho ao longo dos anos, estando presente em momentos difíceis.

Ao meu orientador, prof. Dr. Leonardo Delarmelina Secchin, devido à paciência, compreensão e por me orientar da melhor maneira possível. Sem sua ajuda persistente, o objetivo deste trabalho não teria sido alcançado.

Aos professores da Universidade Federal do Espírito Santo, pelas aulas muito bem ministradas, discussões esclarecedoras e por incentivar o interesse pela área da matemática.

Agradeço aos meus colegas, David, Matheus, Lucas, Wisley e Uarlley pelo apoio e discussões proveitosas ao longo da graduação.

Aos meus pais e irmãos pelo apoio e incentivo nos momentos de dificuldade.

Agradeço a Universidade Federal do Espírito Santo, UFES, e o apoio financeiro da FAPES, processo 116/2019.

"Uma experiência nunca é um fracasso, pois sempre demonstra algo."

Thomas Edison

Resumo

Neste trabalho apresentamos uma descrição teórica e uma análise de desempenho de métodos computacionais conhecidos na literatura para problemas de otimização com um grande número de variáveis sem restrições ou com restrições simples tipo caixa. O estudo de método voltados para essa classe tem aumentado ao longo dos anos devido à alta na quantidade de dados e complexidade dos modelos matemáticos. Foram considerados os métodos de descida pelo gradiente projetado e pelo gradiente espectral projetado. Além desses, discutimos a generalização do método de direções conjugadas para minimização de funções não quadráticas proposto por Hager e Zhang, e o método quase-Newton BFGS com memória limitada. Para comparar o desempenho dos métodos foram realizados diversos testes numéricos, aplicados a problemas da coletânea CUTEst e outros, e a construção e análise de perfis de desempenho com relação ao tempo de CPU, avaliações de gradiente e número de iterações. Diante dos testes, observou-se para quais categorias de problemas cada método é adequado, assim como o desempenho relativo entre eles.

Palavras-chave: Métodos computacionais, Otimização irrestrita, Larga escala, restrições de caixa.

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

A	matriz
b	vetor coluna
A^t, b^t	transposta
$\nabla f(x)$	vetor gradiente
$\nabla^2 f(x)$	matriz hessiana
$\|\cdot\|$	norma euclidiana
$\|\cdot\|_\infty$	norma do máximo
$\subset_\infty$	subconjunto infinito

Sumário

1 Introdução

Com a crescente disponibilidade de dados, é de interesse a resolução de modelos de otimização cada vez mais complexos. É o caso, por exemplo, dos modelos de otimização irrestrita, oriundos da chamada inteligência artificial. Neste contexto, o emprego de métodos que utilizam derivadas de segunda ordem ou que usam algum processo computacionalmente custoso é inviável. Portanto, o estudo de métodos voltados para essa classe de problemas, a que chamamos larga escala, é fundamental.

Normalmente, os problemas de otimização podem ser divididos em categorias que costumam considerar as características da função objetivo e das restrições. Quando o modelo fornecer uma função qualquer sem restrições, chamamos *problema de otimização irrestrita*; caso as restrições possuam uma estrutura "fácil" de lidar, por exemplo, restrições convexas, intitularemos de *problema de otimização com restrições simples*. Este trabalho aborda métodos adequados à resolução de grandes problemas de otimização irrestrita ou com restrições convexas simples, majoritariamente de *caixa*, isto é, onde as únicas restrições são limitantes nas variáveis.

Vários métodos foram propostos na literatura para solucionar problemas de otimização irrestrita ou com restrições simples, e com um grande número de variáveis. Dentre eles, um dos mais conhecidos é o método de descida do gradiente proposto em 1847 por Cauchy [1]. Este método é muito simples e tem baixo custo computacional por iteração. Porém, conforme discutido por [2], apresenta uma convergência lenta próximo à solução. Mesmo assim, ele serve como base para métodos mais elaborados e eficazes, e por isso será tratado neste trabalho. Ainda, é fácil adaptá-lo à problemas com restrições convexas, gerando o método do gradiente projetado que também será abordado.

Dentre os métodos que se baseiam na ideia de descida pelo gradiente (ou gradiente projetado) melhoria do algoritmo básico do gradiente, citamos o método do Gradiente Espectral Projetado [3], que inclui uma estratégia quase-newtoniana para atualização do passo, e o método Gradiente Estocástico [4] (não abordado neste trabalho), utilizado com bastante frequência no aprendizado de máquina devido ao seu baixíssimo custo

computacional. Além dos métodos anteriores, outros foram desenvolvidos. Esse é o caso do método CG-descent proposto por Hager e Zhang [5, 6], uma generalização do método de direções conjugadas para funções não quadráticas, e o método L-BFGS [7], sendo uma adaptação do método quase-Newton BFGS (Broyden–Fletcher–Goldfarb-Shanno) com baixo uso de memória. Esses dois últimos métodos serão abordados.

Em resumo, o objetivo deste trabalho é apresentar a teoria e investigar o desempenho numérico dos seguintes métodos computacionais para a resolução de problemas de otimização irrestrita ou com restrições simples: Gradiente, Gradiente Projetado, Gradiente Espectral Projetado, CG-descent e L-BFGS.

Para verificar o desempenho dos métodos, compararemos o tempo de resolução, número de iterações e avaliações de gradiente para os problemas selecionados através de tabelas e *perfis de desempenho* propostos por Dolan e Moré [8]. Visamos assim determinar quais métodos são mais eficientes e eficazes para cada categoria de problema. Cabe ressaltar que os métodos que serão abordados nesse trabalho serão aplicados, em sua maioria, a problemas não lineares. Para problemas lineares ou quadráticos irrestritos, existem métodos eficientes especializados, como o pontos interiores e programação quadrática [7].

Realizamos diversos testes computacionais com bancos de problemas conhecidos, como a CUTEst [9] (do inglês *Constrained and Unconstrained Testing Environment with safe threads*) para problemas irrestritos ou com restrições simples, e problemas construídos a partir da coletânea de matrizes esparsas da Universidade da Flórida, conhecida como *Suite Sparse Matrix Collection* [10]. Consideramos ainda um modelo proveniente de uma aplicação na biologia celular.

Este trabalho está organizado da seguinte forma. O Capítulo 2 descreve os principais conceitos da otimização contínua utilizados tais como as condições de otimalidade e convexidade. No Capítulo 3 abordamos os clássicos métodos de descida pelo gradiente, como o gradiente projetado e o gradiente espectral projetado, falamos sobre estratégias de busca linear inexata, e por último apresentamos a convergência global desses métodos. Em seguida, no Capítulo 4, introduzimos o método de direções conjugadas para quadráticas, e logo após sua generalização CG-descent. Ademais, realizamos os testes computacionais com problemas da CUTEst e outros. No Capítulo 5, apresentamos o método L-BFGS e realizamos testes numéricos comparando o novo método com os demais. No Capítulo 6 apresentamos as conclusões encontradas a partir do estudo teórico, testes computacionais e análises de desempenho realizadas ao longo desse trabalho. Por fim, o Apêndice A traz

as tabelas completas dos testes realizados.

2 Elementos de otimização

Nesse capítulo, abordaremos alguns conceitos básicos em otimização contínua. Primeiramente, apresentamos o conceito de minimizadores e condições para sua existência; em seguida discutimos as condições de otimalidade para problemas de otimização irrestrita e com restrições convexas. Por último, abordamos um caso específico dos problemas com restrições convexas, o caso cujas restrições são de caixa. Algumas referências para esse capítulo são [1, 11–13].

2.1 O problema de otimização

Será considerado o seguinte problema de otimização:

$$\begin{aligned} &\text{Minimizar} && f(x) \\ &\text{Sujeito a} && x \in \Omega, \end{aligned} \tag{2.1}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$ continuamente diferenciável e $\Omega \subset \mathbb{R}^n$ qualquer. O conjunto Ω é chamado *conjunto factível* ou *viável*. Chamaremos gradiente de f, ou $\nabla f(x)$, o vetor coluna cujas n linhas são as derivadas parciais de f. Ou seja,

$$\nabla f(x) = \begin{bmatrix} \dfrac{\partial f}{\partial x_1} & \dfrac{\partial f}{\partial x_2} & \cdots & \dfrac{\partial f}{\partial x_n} \end{bmatrix}^t.$$

Definição 2.1. *Considere $f : \mathbb{R}^n \to \mathbb{R}$ e $x^* \in \Omega \subset \mathbb{R}^n$. Dizemos que x^* é um* minimizador local *de f em Ω quando existe $\delta > 0$ tal que $f(x^*) \leq f(x)$ para todo $x \in \Omega$ com $\|x - x^*\| < \delta$. Se $f(x^*) < f(x)$ para todo $x \in \Omega$ tal que $x \neq x^*$ e $\|x - x^*\| < \delta$, diremos que se trata de um* minimizador local estrito *em Ω.*

Definição 2.2. *Um ponto $x^* \in \Omega$ é um* minimizador global *de f em Ω se, e somente se, $f(x^*) \leq f(x)$ para todo $x \in \Omega$. Se $f(x^*) < f(x)$ para todo $x \in \Omega$ tal que $x^* \neq x$, diremos que se trata de um* minimizador global estrito *em Ω.*

Se omitirmos o conjunto Ω, significa que $\Omega = \mathbb{R}^n$ e teremos um problema de otimização

irrestrita. Veremos agora condições que garantem a existência de minimizadores. A primeira é a compacidade de Ω.

Teorema 2.1 (Teorema de Weierstrass)**.** *Sejam $f : \mathbb{R}^n \to \mathbb{R}$ contínua e $\Omega \subset \mathbb{R}^n$ compacto e não vazio. Então existe minimizador global de f em Ω.*

O Teorema de Weierstrass garante que se impormos ao problema restrições artificiais de modo a tornar Ω compacto, teremos a garantia da existência de minimizador. Isso é feito, por exemplo, impondo limitantes inferior e superior em cada variável (restrições de caixa). Essas restrições serão tratadas na Seção 2.3.1.

Outra forma comum em otimização de garantir existência de minimizadores é a seguinte [11]:

Teorema 2.2. *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função contínua. Suponha que f é* coerciva*, isto é, que $\lim_{\|x\|\to\infty} f(x) = \infty$. Então f admite um minimizador global.*

2.2 Condição de otimalidade de primeira ordem para problemas irrestritos

Abordaremos uma condição necessária que caracteriza um minimizador em problemas irrestritos, ou seja, problemas da forma (2.1) em que $\Omega = \mathbb{R}^n$. Nesse trabalho, concentraremos apenas nas condições de primeira ordem, pois os problemas objetivo são de larga escala, e até mesmo avaliação de gradiente é caro computacionalmente. Condições suficientes necessitam de considerações de segunda ordem como o cálculo da hessiana. Para mais informações sobre todo o contexto de condições de otimalidade necessárias e suficientes, consulte as referências do início do capítulo.

Teorema 2.3 (Condição necessária de primeira ordem)**.** *Seja $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável no ponto $x^* \in \mathbb{R}^n$. Se x^* é um minimizador local da função f, então*

$$\nabla f(x) = 0. \tag{2.2}$$

Demonstração. Suponha por contradição que $\nabla f(x^*)$ fosse não nulo. Como x^* é um minimizador local de f, existe $\delta > 0$ tal que

$$f(x^*) \leq f(x^* - t\nabla f(x^*)), \tag{2.3}$$

para todo $t \in (0, \delta)$. Pela expansão de Taylor de primeira ordem, temos

$$f(x^* - t\nabla f(x^*)) = f(x^*) - t\nabla f(x^*)^t \nabla f(x^*) + r(t), \tag{2.4}$$

com $\lim_{t\to 0} r(t)/t = 0$. Daí, usando (2.3) e dividindo ambos os lados de (2.4) por t, obtemos

$$-\|\nabla f(x^*)\|^2 + \frac{r(t)}{t} \geq 0,$$

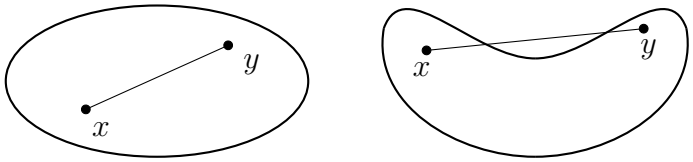
e tomando o limite quando $t \to 0^+$, obtemos $-\|\nabla f(x^*)\| > 0$, o que é uma contradição. Portanto, $\nabla f(x^*) = 0$. □

Definição 2.3. *Um ponto $x^* \in \mathbb{R}^n$ que cumpre a condição necessária de primeira ordem (2.2) é chamado* ponto crítico *ou* estacionário *da função f.*

2.3 Conjuntos convexos e propriedades

Um conjunto convexo é caracterizado por conter todos os segmentos de reta cujos extremos são pontos do conjunto. Mais especificadamente, temos a seguinte definição (observe ainda a Figura 1):

Definição 2.4. *Um conjunto $X \subset \mathbb{R}^n$ é convexo se, e somente se, para todo $x, y \in X$, $\alpha \in [0,1]$ se verifica $\alpha x + (1-\alpha)y \in X$.*



Convexo Não convexo

Figura 1: Conjuntos convexo e não convexo.

Nessa seção, realizaremos um estudo das condições de otimalidade para problemas da forma (2.1), tomando $\Omega = X$ convexo, fechado e não vazio. Inicialmente, constataremos a teoria por trás das projeções, sendo uma estratégia comum para problemas cujas restrições são convexas.

Definição 2.5. *Seja $X \subset \mathbb{R}^n$ e $v \in \mathbb{R}^n$. Se $p \in X$ é tal que*

$$\|p - v\| = \inf\{\|z - v\|; z \in X\} = d(v, X),$$

então p é uma projeção de v em X. Em particular, se X é fechado então $p \in X$.

Teorema 2.4. *Para qualquer conjunto fechado X e $v \in \mathbb{R}^n$ arbitrário, existe (pelo menos um) $p \in X$ o qual é a projeção de v em X. Além disso, se X é convexo então p é único.*

Demonstração. Pela definição de ínfimo, para cada $k \in \mathbb{N}$ existe um ponto $z_k \in X$ tal que $d(v, X) \leq \|z_k - v\| < d(v, X) + 1/k$, logo $\lim_{k\to\infty} \|z_k - v\| = d(v, X)$. Note que, a sequência $\{z_k\}$ é limitada, logo existe uma subsequência convergente $\{z_k\}_{k\in N_1}$, em que $N_1 \underset{\infty}{\subset} \mathbb{N}$, com $\lim_{k\in N_1} z_k = p$. Além disso, se X for fechado temos $p \in X$.

Visto que, X é fechado, $p \in X$ e $\|p-v\| = d(v, X)$. Assuma que X é convexo e suponha que exista duas projeções diferentes p e p' de tal modo que $\|p - v\| = \|p' - v\| = d(v, X)$. Então, pela convexidade de X, $z = (p + p')/2 \in X$. A identidade

$$\|x + y\|^2 + \|x - y\|^2 = 2\|x\|^2 + 2\|y\|^2$$

implica que

$$\begin{aligned}
\|z - v\|^2 &= \left\|\frac{p - v}{2} + \frac{p' - v}{2}\right\|^2 \\
&= 2\left\|\frac{p - v}{2}\right\|^2 + 2\left\|\frac{p' - v}{2}\right\|^2 - \left\|\frac{p - v}{2} - \left(\frac{p' - v}{2}\right)\right\|^2 \\
&= \frac{1}{2}\|p - v\|^2 + \frac{1}{2}\|p' - v\|^2 - \frac{1}{4}\|p - p'\|^2 \\
&= d^2(v, X) - \frac{1}{4}\|p - p'\|^2 < d^2(v, X),
\end{aligned}$$

contradizendo a definição de $d(v, X)$. Portanto, $p = p'$ e a projeção é única. □

A unicidade da projeção sobre conjuntos convexos nos permite definir a notação a seguir, usual em otimização.

Definição 2.6. *Seja $X \subset \mathbb{R}^n$ um conjunto convexo fechado. Para cada $v \in \mathbb{R}^n$ definimos $P_X(v)$ como a projeção (única) de v em X.*

Os dois resultados a seguir serão úteis ao longo do trabalho.

Teorema 2.5. *Seja $X \subset \mathbb{R}^n$ um conjunto convexo fechado. Então $p = P_X(v)$ se, e somente se,*

$$(x - p)^t(v - p) \leq 0$$

para todo $x \in X$.

Demonstração. Seja p a projeção de v em X, $x \in X$ arbitrário e assuma $z = \alpha x + (1-\alpha)y$.

Pela convexidade de X temos que $z \in X$ para qualquer $\alpha \in [0,1]$. A partir de

$$\begin{aligned}\|z-v\|^2 &= \|\alpha x + (1-\alpha)y\|^2 \\ &= \|\alpha(x-p) + (p-v)\|^2 \\ &= [\alpha(x-p) + (p-v)]^t[\alpha(x-p) + (p-v)] \\ &= \alpha^2\|x-p\|^2 + 2\alpha(x-p)^t(p-v) + \|p-v\|^2 \geq \|p-v\|^2.\end{aligned}$$

Temos que

$$\phi(\alpha) = \alpha^2\|x-p\|^2 + 2\alpha(x-p)^t(p-v) \geq 0,$$

para $\alpha \in [0,1]$. Portanto,

$$\phi'(0) = 2(x-p)^t(p-v) \geq 0 \Rightarrow (x-p)^t(v-p) \leq 0 \tag{2.5}$$

Agora assuma que valha (2.5), então, para qualquer $x \in X$ temos

$$\begin{aligned}\|x-v\|^2 &= \|(x-p) + (p-v)\|^2 \\ &= \|x-p\|^2 + 2(x-p)^t(p-v) + \|p-v\|^2 \geq \|p-v\|^2.\end{aligned}$$

Daí, p é a projeção de v em X. □

Teorema 2.6 (Não expansividade)**.** *Se $X \subset \mathbb{R}^n$ é um conjunto convexo fechado, então*

$$\|P_X(u) - P_X(v)\| \leq \|u-v\|,$$

para todo $u, v \in \mathbb{R}^n$.

Demonstração. Se $P_X(u) = P_X(v)$ o resultado é evidente. Assim, assumiremos $P_X(u) \neq P_X(v)$. Substituindo x por $P_X(v)$ no Teorema 2.5 obtemos

$$(P_X(v) - P_X(u))^t(u - P_X(u)) \leq 0,$$

e trocando u e v temos

$$(P_X(u) - P_X(v))^t(v - P_X(v)) \leq 0,$$

somando ambas as inequações

$$(P_X(u) - P_X(v))^t[P_X(u) - P_X(v) - (u-v)] \leq 0.$$

Efetuando um rearranjo simples e utilizando a desigualdade de Cauchy-Schwarz, temos

que

$$\begin{aligned} \|P_X(u) - P_X(v)\|^2 &\leq (P_X(u) - P_X(v))^t(u - v) \\ &\leq \|P_X(u) - P_X(v)\|\|u - v\|. \end{aligned}$$

Daí,

$$\|P_X(u) - P_X(v)\| \leq \|u - v\|.$$

□

A seguir, enunciamos a condição necessária de otimalidade para problemas da forma (2.1), onde o conjunto viável Ω é convexo e fechado. Essa a condição é utilizada para caracterizar minimizadores de (2.1), e logo serve como critério de parada para algoritmos, por exemplo, o método de gradiente projetado que veremos mais a frente.

Teorema 2.7 (Condição necessária de primeira ordem)**.** *Sejam $f : \mathbb{R}^n \to \mathbb{R}$ uma função difirenciável e $\Omega \subset \mathbb{R}^n$ convexo e fechado. Se $x^* \in \Omega$ é minimizador local de f em Ω, então*

$$P_\Omega(x^* - \alpha\nabla f(x^*)) - x^* = 0$$

para todo $\alpha > 0$.

Demonstração. Fixado $x \in \Omega$ e devido a x^* ser minimizador local, temos que $f(x^*) \leq f((1 - t)x^* + tx)$ para todo $t \geq 0$ suficientemente pequeno. Portanto, utilizando a aproximação de Taylor de primeira ordem, temos que

$$0 \leq f(x^* + t(x - x^*)) - f(x^*) = t\nabla f(x^*)^t(x - x^*) + r(t),$$

onde $\lim_{t\to 0} r(t)/t = 0$. Dividindo por t e passando o limite, obtemos $\nabla f(x^*)^t(x-x^*) \geq 0$. Assim, dado $\alpha > 0$, temos

$$(x^* - \alpha\nabla f(x^*) - x^*)^t(x - x^*) \leq 0.$$

Finalmente, pelo Teorema 2.5, temos que

$$P_\Omega(x^* - \alpha\nabla f(x^*)) = x^* \Leftrightarrow P_\Omega(x^* - \alpha\nabla f(x^*)) - \bar{x} = 0.$$

□

A estratégia de projeções mesmo sendo muito utilizada, pode se tornar extremamente custosa, visto que as restrições, que geram a região viável, podem ser não lineares. Ou

seja, mesmo com todas as propriedades de convexidade o cálculo da projeção pode ser tão custosa quanto a resolução do problema original.

2.3.1 Restrições de caixa

Um dos conjuntos viáveis convexos mais comuns são aqueles definidos por restrições de caixa, ou seja, aquelas que limitam apenas as variáveis. Nesse caso, temos o seguinte conjunto factível:

$$\Omega = \{x \in \mathbb{R}^n |\ l \leq x \leq u\}, \tag{2.6}$$

em que $l, u \in \mathbb{R}^n$ com $l < u$. Tais restrições costumam aparecer com bastante frequência em modelos físicos, cujos dados só fazem sentido em uma dada região, como problema de torção elástico-plástico [14] e problema do obstáculo [15].

Uma das vantagens em trabalhar com restrições de caixa é devido à facilidade com que a projeção é computada. De fato, tomando p_i como a i-ésima componente do vetor $p \in \mathbb{R}^n$, temos que a projeção de $v \in \mathbb{R}^n$ sobre a caixa é dada por $p = P_{[l,u]}(v)$, onde

$$p_i = \begin{cases} l_i, & \text{se } v_i \leq l_i \\ v_i, & \text{se } l_i \leq v_i \leq u_i \\ u_i, & \text{se } v_i \geq u_i \end{cases} = \min\{u_i, \max\{l_i, v_i\}\}, \tag{2.7}$$

para $i = 1, \ldots, n$. O comportamento da projeção é ilustrado na Figura 2.

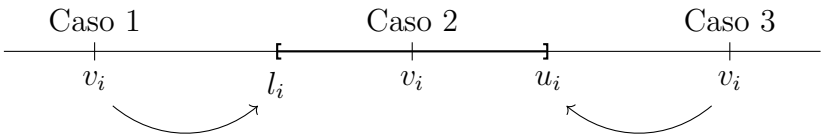


Figura 2: Visualização da projeção sobre a caixa por coordenada.

A estratégia da Equação (2.7) é empregada em diversos pacotes computacionais como ALGENCAN [16] e SPG [17].

2.4 Direções de descida

Um dos esquemas algorítmicos mais clássicos para minimização de funções são os esquemas de descida. Tais algoritmos consistem em caminhar em direções nas quais a função decresça localmente. A seguir, apresentamos os principais conceitos de direções de descida.

Definição 2.7. *Considere uma função $f : \mathbb{R}^n \to \mathbb{R}$, um ponto $\bar{x} \in \mathbb{R}^n$ e uma direção $d \in \mathbb{R}^n/\{0\}$. Dizemos que d é uma* direção de descida para f, a partir de $\bar{x}$, *quando existe $\delta > 0$ tal que $f(\bar{x} + td) < f(\bar{x})$, para todo $t \in (0, \delta)$.*

O teorema a seguir fornece uma condição suficiente, algébrica e computável, para que d seja direção de descida.

Teorema 2.8. *Se $\nabla f(\bar{x})^t d < 0$, então d é uma direção de descida para f a partir de $\bar{x}$.*

Demonstração. Sabemos que

$$\nabla f(\bar{x})^t d = \frac{\partial f}{\partial d}(\bar{x}) = \lim_{t \to 0} \frac{f(\bar{x} + td) - f(\bar{x})}{t}.$$

Pela hipótese, existe $\delta > 0$ tal que

$$\frac{f(\bar{x} + td) - f(\bar{x})}{t} < 0,$$

para todo $t \in (-\delta, \delta), t \neq 0$. Portanto, $f(\bar{x}+td) < f(\bar{x})$, para todo $t \in (0, \delta)$, completando a demonstração. □

O Teorema 2.8 pode ser interpretado geometricamente: as direções de descida são aquelas que formam um ângulo obtuso com $\nabla f(\bar{x})$. Observe a Figura 3.

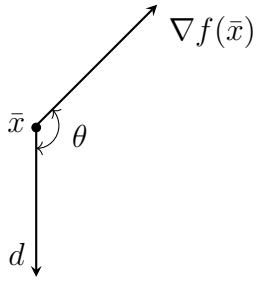


Figura 3: Interpretação geométrica do Teorema 2.8.

3 Métodos de descida pelo gradiente

Um dos métodos mais antigos e utilizados para se minimizar uma função de várias variáveis é o método de descida pelo gradiente, ou resumidamente, *método do gradiente*. Este método encontra-se na classe dos métodos de descida, onde tomamos $d = -\nabla f(x)$ na Definição 2.7 (esta direção é descida pelo Teorema 2.8 sempre que $\nabla f(x) \neq 0$). O método do gradiente consiste em dar um passo na direção $-\nabla f(x^k)$ a partir de um iterando x^k, gerando um novo iterando.

O método do gradiente possui grande importância do ponto de vista teórico, pois traz uma ideia fundamental e muito simples para minimização de funções. Métodos mais eficientes são frequentemente motivados por tentativas de modificar o algoritmo básico do gradiente, de modo que o novo método possua uma convergência/comportamento superior. O método do gradiente, portanto, não é apenas a "primeira tentativa" ao tentar resolver um novo problema, mas também o padrão de referência contra o qual outras técnicas são medidas. Algumas referências básicas para esse capítulo são [1, 2, 7, 11].

3.1 Método do gradiente para minimização irrestrita

3.1.1 Busca inexata: condição de Armijo

Em muito dos casos, determinar o tamanho do passo (*busca linear*) pode ser complicado, dado que para encontrar um passo ótimo necessitaria de muitas avaliações de função e, consequentemente, um maior tempo computacional. Com isso, o estudo de métodos voltados, especificamente, para o cálculo do tamanho do passo, tem grande importância na área de otimização visto que a eficiência dos métodos dependam do quão bom é a busca linear.

Uma escolha "ideal" para o tamanho do passo seria o minimizador da função $\phi(\cdot)$

definida por

$$\phi(\alpha) = f(x_k + \alpha d_k),\ \alpha > 0, \tag{3.1}$$

porém, para determiná-lo, é necessário minimizar exatamente a função ϕ (busca linear *exata*), que, dependendo de f, pode ser custoso especialmente se f for não convexa. Sendo assim, nessa seção estamos interessados em determinar um tamanho de passo *aproximado*, eficientemente, de modo que, ao dar o passo, haja uma redução suficiente de f (busca linear *inexata*).

Uma condição simples que pode ser imposta para determinar α_k, é exigir que ao dar o passo se tenha um *decréscimo simples* da função, ou seja, obtemos o tamanho do passo de modo que $f(x_k + \alpha_k d_k) < f(x_k)$. Tal tática possui problemas, visto que existe a possibilidade da obtenção de passos muito pequenos, tornando a convergência do método lenta, ou ainda pior, nem convergir para um minimizador. De fato, como ilustrado na Figura 4, a sequência gerada pelo método poderia se acumular em um ponto não ótimo. No exemplo da figura, consideramos $f(x) = x^2$ e tomamos uma sequência $x_k = 1 + \dfrac{1}{k+1}$. Repare que $x_k \to 1$, que não é estacionário. Para contornar esse problema, exigimos sobre α_k a condição de Armijo, que implica em um decréscimo suficiente em f.

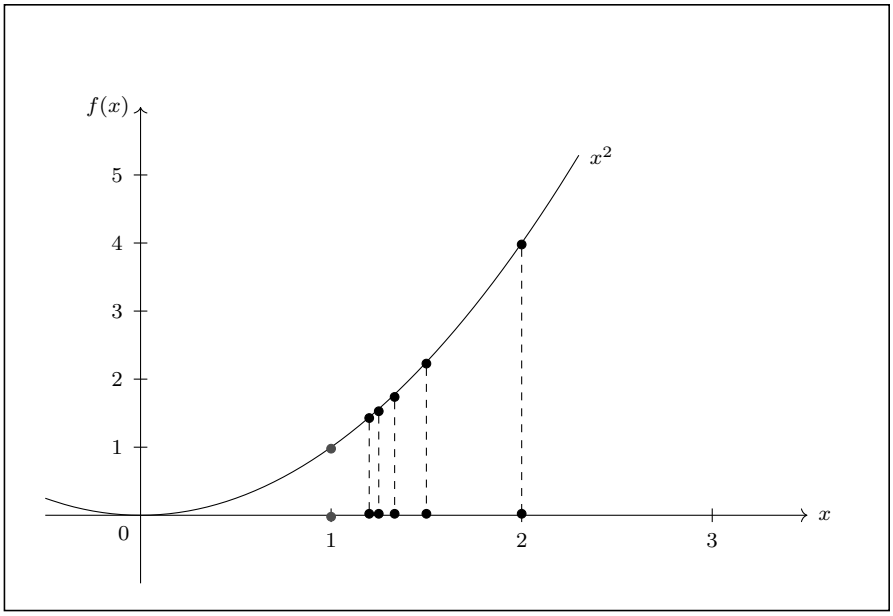


Figura 4: Decréscimo simples insuficiente para convergência.

A condição de Armijo consiste em determinar o tamanho de passo de modo que haja um *decréscimo suficiente* da função, mais especificamente exigimos que o decréscimo ao longo da direção de busca seja proporcional ao tamanho do passo e a derivada direcional $\nabla f(x_k)^t d_k$. Ou seja, temos como propósito determinar $\bar{\alpha}$, tal que

$$f(x_k + \bar{\alpha} d_k) \leq f(x_k) + \eta\, \bar{\alpha} \nabla f(x_k)^t d_k, \tag{3.2}$$

sendo $x_k \in \mathbb{R}^n$, $\eta \in (0,1)$ e d_k uma direção de descida para a função f. A Figura 5 ilustra a condição acima. A seguir, apresentamos o teorema, obtido de [11], que garante sempre a existência de $\bar{\alpha}$.

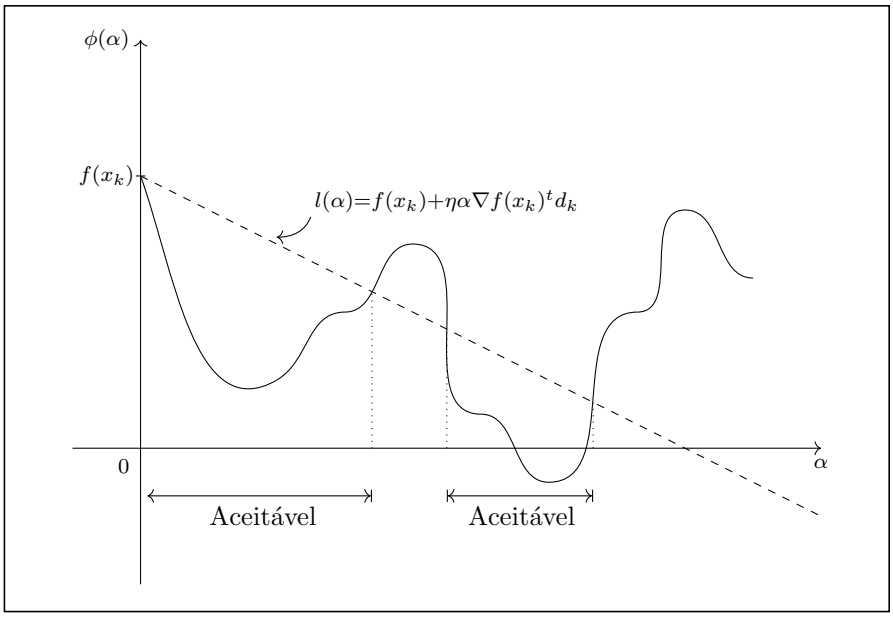


Figura 5: Interpretação da condição de Armijo.

Teorema 3.1. *Considere uma função diferenciável $f : \mathbb{R}^n \to \mathbb{R}$, um ponto $\bar{x} \in \mathbb{R}^n$, uma direção de descida $d \in \mathbb{R}^n$ e $\eta \in (0,1)$. Então existe $\delta > 0$ tal que*

$$f(\bar{x} + \alpha d) \leq f(\bar{x}) + \eta\alpha\nabla f(\bar{x})^t d,$$

para todo $\alpha \in [0, \delta]$.

Demonstração. Caso $\nabla f(\bar{x})^t d = 0$, o resultado segue da definição de direção de descida, Teorema 2.8. Suponha então que $\nabla f(\bar{x})^t d < 0$. Sendo assim, devido a $\eta < 1$, temos que

$$\lim_{\alpha \to 0} \frac{f(\bar{x} + \alpha d) - f(\bar{x})}{\alpha} = \nabla f(\bar{x})^t d < \eta \nabla f(\bar{x})^t d.$$

Portanto, existe $\delta > 0$ tal que

$$\frac{f(\bar{x} + \alpha d) - f(\bar{x})}{\alpha} < \eta \nabla f(\bar{x})^t d,$$

para todo $\alpha \in (0, \delta]$. Isto implica

$$f(\bar{x} + \alpha d) \leq f(\bar{x}) + \eta\alpha\nabla f(\bar{x})^t d,$$

o que conclui a demonstração. □

Uma estratégia geralmente utilizada para determinar um tamanho de passo que satisfaça a condição de Armijo é conhecida como *backtracking*, que consiste em tomar um tamanho de passo, inicialmente, fixo $\bar{\alpha} > 0$ e o reduzir a uma certa taxa até que satisfaça a desigualdade proposta, cuja existência é garantida pelo Teorema 3.1.

Outra estratégia utilizada em conjunto com o *backtracking* para acelerar a busca é a interpolação quadrática, que consiste em minimizar a função quadrática $p(\alpha)$ que interpola os pontos $(0, \phi(0))$ e $(\bar{\alpha}, \phi(\bar{\alpha}))$ e cuja derivada $p'(0) = \phi'(0)$. Nesse caso, devemos verificar se o minimizador de $p(\alpha)$ satisfaz a condição de Armijo. Repare que determinar tal função é simples. De fato, tomando $p(\alpha) = a\alpha^2 + b\alpha + c$, obtemos:

$$\begin{aligned} p(0) &= \phi(0) \Rightarrow c = \phi(0) = f(x_k) \\ p'(0) &= \phi'(0) \Rightarrow b = \phi'(0) = \nabla f(x_k)^t d_k. \end{aligned}$$

Daí, sabendo que $p(\bar{\alpha}) = \phi(\bar{\alpha})$, obtemos:

$$a = \frac{f(x_k + \bar{\alpha}d_k) - b\bar{\alpha} - c}{\bar{\alpha}^2} = \frac{f(x_k + \bar{\alpha}d_k) - \nabla f(x_k)^t d_k \bar{\alpha} - f(x_k)}{\bar{\alpha}^2}. \tag{3.3}$$

Note que, como d_k é uma direção de descida e $\bar{\alpha} > 0$, temos $a > 0$. Ou seja, a função p é uma quadrática com concavidade voltada para cima e seu gradiente nulo é suficiente para o cálculo do minimizador global. Portanto, $\alpha^* = \arg\min\{p(\alpha); \alpha > 0\}$ é dado por

$$\alpha^* = -\frac{b}{2a} = -\frac{\bar{\alpha}^2 \nabla f(x_k)^t d_k}{2[f(x_k + \bar{\alpha}d_k) - f(x_k) - \nabla f(x_k)^t d_k]}, \tag{3.4}$$

e verifica-se se α^* satisfaz a condição de Armijo. Caso esse tamanho de passo seja recusado, voltamos a aplicar *backtracking* normalmente.

De certo modo, realizar uma busca inexata com a estratégia mista de *backtracking* e interpolação quadrática é uma boa opção. Para melhorar essa tática propomos o uso de salvaguardas para a interpolação quadrática. Sendo assim, caso o tamanho de passo inicial α ou o passo α^* da interpolação seja relativamente inferior a um σ_1 pequeno, aplicamos apenas *backtracking*. Outra salvaguarda, é exigir que α^* seja ao menos menor do que ou igual a uma fração σ_2 do passo α recusado pela condição de Armijo. Novamente, caso α^* não cumpra este limite, aplicamos apenas *backtracking*.

A seguir, apresentamos um simples algoritmo para determinar o tamanho de passo que satisfaça Armijo. Ele traz uma estratégia mista com o uso do *backtracking* e interpolação quadrática com salvaguardas, conforme [3].

Algoritmo 1 Busca linear com *backtracking* e interpolação quadrática

Entrada: $\bar{\alpha} > 0$, $0 < \sigma_1 < \sigma_2 < 1$, $\rho \in (0,1)$, $\eta \in (0,1)$, x_k e $d_k \in \mathbb{R}^n$;
1: Tome $i = 0$ e $\alpha_i = \bar{\alpha}$;
2: **enquanto** $f(x_k + \alpha_i d_k) > f(x_k) + \eta\, \alpha_i \nabla f(x_k)^t d_k$ **faça**
3: **se** $\alpha_i < \sigma_1$ **então**
4: $\alpha_{i+1} = \rho\alpha_i$;
5: **senão**
6: Calcule α^* como proposto em (3.4) com $\bar{\alpha} = \alpha_i$.
7: **se** $\alpha^* \in [\sigma_1, \alpha_i\sigma_2]$ **então**
8: $\alpha_{i+1} = \alpha^*$;
9: **senão**
10: $\alpha_{i+1} = \rho\alpha_i$;
11: **fim se**
12: **fim se**
13: $i = i + 1$;
14: **fim enquanto**
Saída: $\alpha_k = \alpha_i$

3.1.2 O método

O método de descida pelo gradiente com busca linear exata consiste no processo iterativo definido da seguinte maneira:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \tag{3.5}$$

onde o tamanho do passo α_k é um número real positivo que minimiza $f(x_k - \alpha\nabla f(x_k))$. Ou seja, o tamanho de passo é determinado de modo a minimizar a função ao longo da direção de busca $-\nabla f(x_k)$. Com isso, o algoritmo proposto gera uma sequência $\{x_k\}$ que decresce a função. De forma análoga, podemos computar o tamanho do passo de maneira inexata utilizando a condição de Armijo apresentada na seção anterior. O método do gradiente que enunciaremos nessa seção segue esta estratégia. Nele, também há decréscimo da função f a cada passo, propriedade garantida pela condição de Armijo.

A escolha da direção $-\nabla f(x_k)$ deve-se ao fato de que, dentre todas as possíveis direções onde a função decresce localmente, a direção oposta ao do gradiente é a de maior decrescimento. De fato, se tomarmos $d = -\nabla f(x)$ e $u \in \mathbb{R}^n$ tal que $\|u\| = \|d\|$, então

$$\frac{\partial f}{\partial d}(x) = \nabla f(x)^t d = -\|\nabla f(x)\|^2 = -\|\nabla f(x)\|\|u\| \leq \nabla f(x)^t u = \frac{\partial f}{\partial u}(x),$$

ou seja, a direção d nos fornece a menor derivada direcional no ponto x, implicando em um maior decréscimo da função objetivo localmente.

A seguir, apresentamos um exemplo em que abordamos algumas propriedades do método do gradiente em um problema quadrático. O exemplo foi obtido de [1].

Exemplo 3.1. Considere o método do gradiente aplicado ao seguinte problema quadrático:

$$\text{Minimizar } \; q(x) = \frac{1}{2}x^t Gx - b^t x, \tag{3.6}$$

com G definida positiva (isto é, $x^t Gx > 0, \; \forall x \in \mathbb{R}^n \backslash \{0\}$). Seja $\tilde{x}$ a solução e suponha que x_0 possa ser escrito como $x_0 = \tilde{x} + \mu v$, onde v é um autovetor de G associado ao autovalor λ e μ é um número real. Primeiro, mostraremos que $\nabla q(x_0) = \mu\lambda v$. Em segundo lugar, mostraremos que se for realizada uma busca linear exata a partir de x_0, há convergência em uma única iteração. Mais ainda, o método do gradiente converge em uma iteração para qualquer x_0 sempre que G for da forma σI com $\sigma \in \mathbb{R}$.

De fato, como $\nabla q(x) = Gx - b$ e v é um autovetor de G associado ao autovalor λ, ou seja, $Gv = \lambda v$, então

$$\begin{aligned} \nabla q(x_0) &= Gx_0 - b \\ &= G(\tilde{x} + \mu v) - b \\ &= G\tilde{x} + G\mu v - b. \end{aligned}$$

Sendo assim, como $\tilde{x}$ é solução e v é um autovetor associado a matriz G, temos que

$$\nabla q(x_0) = \mu\lambda v. \tag{3.7}$$

Considere agora a primeira iteração do método do gradiente

$$x_1 = x_0 - \alpha_0 \nabla q(x_0),$$

em que α_0 é determinado de maneira exata. Derivando a função $\phi(\alpha) = q(x_0 - \alpha\nabla q(x_0))$ e igualando a zero, encontramos seu minimizador

$$\alpha_0 \;\; = \;\; \frac{\nabla q(x_0)^t \nabla q(x_0)}{\nabla q(x_0)^t G \nabla q(x_0)} \tag{3.8}$$

Daí, com o uso de (3.7) e sabendo que v é um autovetor associado ao autovalor λ da matriz G, temos que

$$\alpha_0 = \frac{\|\mu\lambda v\|^2}{(\mu\lambda v)^t G(\mu\lambda v)} = \frac{\|\mu\lambda v\|^2}{\mu^2\lambda^2 v^t G v},$$

e logo

$$\alpha_0 = \frac{1}{\lambda}. \tag{3.9}$$

Daí, usando (3.7) e sabendo que $\mu v = x_0 - \tilde{x}$ temos que

$$\begin{aligned} x_1 &= x_0 - \alpha_0 \nabla q(x_0) \\ &= x_0 - \frac{1}{\lambda}\mu\lambda v \\ &= x_0 - \mu v \\ &= x_0 - (x_0 - \tilde{x}). \end{aligned}$$

Portanto, $x_1 = \tilde{x}$.

Consideremos agora o caso onde a hessiana de $q(x)$ seja um múltiplo da identidade, ou seja, $G = \sigma I$. Lembrando que $\nabla q(x) = Gx - b = \sigma x - b$, de (3.8) temos que

$$\alpha_0 = \frac{(\sigma x_0 - b)^t(\sigma x_0 - b)}{(\sigma x_0 - b)^t \sigma I (\sigma x_0 - b)} = \frac{1}{\sigma}\frac{\|\sigma x_0 - b\|^2}{\|\sigma x_0 - b\|^2},$$

e logo

$$\alpha_0 = \frac{1}{\sigma}. \tag{3.10}$$

Sendo assim, como o uso de (3.10) o próximo iterando é da forma

$$x_1 = x_0 - \alpha_0 \nabla q(x_0) = x_0 - \frac{\sigma x_0 - b}{\sigma}.$$

Portanto, $x_1 = b/\sigma$ e como

$$\begin{aligned} \nabla q(x_1) &= Gx_1 - b \\ &= G\frac{b}{\sigma} - b \\ &= \sigma I \frac{b}{\sigma} - b \\ &= 0, \end{aligned}$$

temos que x_1 é minimizador. Em outras palavras, o método converge em uma iteração. □

Mesmo que na busca linear do Algoritmo 1 possamos tomar, inicialmente, $\bar{\alpha}$ como qualquer valor real positivo, na prática, a escolha usual $\bar{\alpha} = 1$ funciona bem. Existem estratégias que tentam aumentar o passo mediante algum critério de qualidade do decréscimo de f, como a estratégia de extrapolação usada por Birgin e Martínez [17] para problemas com restrições de caixa. Mas isso está fora do escopo desse trabalho.

A seguir, apresentamos o método do gradiente com a busca linear inexata proposta na seção anterior.

Algoritmo 2 Método do gradiente

Entrada: $x_0 \in \mathbb{R}^n$, $\rho \in (0,1), \eta \in (0,1)$ e uma tolerância $\epsilon > 0$;

1: Tome $k \leftarrow 0$;
2: **enquanto** $\|\nabla f(x_k)\| > \epsilon$ **faça**
3: $d_k = -\nabla f(x_k)$;
4: Tamanho do passo por Armijo: $\alpha_k = 1$;
5: Reduza α_k até que $f(x_k+\alpha_k d_k) \leq f(x_k)+\eta\alpha_k \nabla f(x_k)^T d_k$ utilizando *backtracking* e interpolação quadrática com salvaguardas, como proposto no Algoritmo 1. Assumindo que $\bar{\alpha} = \alpha_k$;
6: $x_{k+1} = x_k + \alpha_k d_k$;
7: $k \leftarrow k + 1$;
8: **fim enquanto**

Saída: x_k.

3.2 Minimização sobre convexos – Método do gradiente projetado

O método do gradiente é uma estratégia comum para resolução de problemas de otimização sem restrições, mas pode ser aplicada com sucesso em problemas cujas restrições são convexas. Nessa seção, falaremos de uma adaptação do método do gradiente para problemas da forma

$$\min f(x) \text{ s.a } x \in \Omega, \tag{3.11}$$

onde Ω é um conjunto convexo fechado e não vazio, chamado método do Gradiente Projetado (GP).

Um dos principais problemas enfrentados ao se acrescentar restrições convexas é manter a factibilidade dos pontos gerados, ou seja, garantir que x_{k+1} seja viável. Uma tática utilizada é tomar a direção de busca como a direção projetada. Isto é

$$d_k = P_\Omega(x_k - \nabla f(x_k)) - x_k. \tag{3.12}$$

Com o uso de (3.12), conseguimos uma garantia de que ao dar o passo $\alpha_k \in (0,1]$ o ponto obtido seja viável, ou seja, pertença ao conjunto factível. Veja a Figura 6. Outra estratégia é dar o passo e depois projetar o ponto gerado, mas experimentos não indicam que seja melhor. Portanto, adotamos a primeira por ser padrão em implementações como em [3].

Como nesse trabalho resolveremos computacionalmente apenas problemas cujas restrições são em formato de caixa, ou seja, $\Omega = \{x \in \mathbb{R}^n | \ l \leq x \leq u\}$, escolher a direção projetada não tem custo relevante, devido à projeção sobre a caixa ser muito simples de

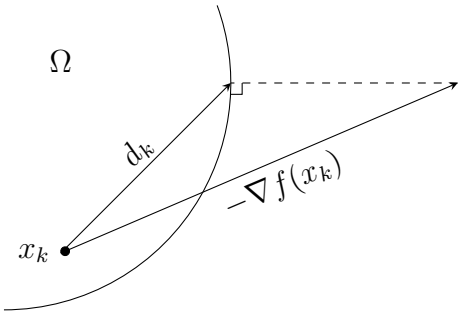


Figura 6: Interpretação geométrica da direção projetada

computar. De fato, é fácil mostrar que

$$[P_{[l,u]}(x - \nabla f(x)) - x]_i = \min\{u_i, \max\{l_i, x_i - \nabla f_i(x)\}\} - x_i, \tag{3.13}$$

para $i = 1, \ldots, n$. Em geral, é desejável que no gradiente projetado, a projeção seja simples de calcular. O caso de restrições de caixa é muito frequente em problemas da literatura.

Na inicialização do método devemos exigir que o ponto inicial x_0 pertença a Ω. Para isso, basta projetarmos o ponto x_0 no conjunto, ou seja, atualizamos $x_0 \leftarrow P_\Omega(x_0)$. O método do gradiente projetado é apresentado no Algoritmo 3 .

Algoritmo 3 Método do gradiente projetado

Entrada: $x_0 \in \mathbb{R}^n$, $\rho \in (0,1), \eta \in (0,1)$ e uma tolerância $\epsilon > 0$.

1: Tome $k \leftarrow 0$;
2: Se $x_0 \notin \Omega$ tome $x_0 \leftarrow P_\Omega(x_0)$;
3: **enquanto** $\|d_k\| > \epsilon$ **faça**
4: $\quad d_k = P_\Omega(x_k - \nabla f(x_k)) - x_k$,
5: $\quad$ Tome $\alpha_k = 1$;
6: $\quad$ Reduza α_k até que $f(x_k + \alpha_k d_k) \leq f(x_k) + \eta \alpha_k \nabla f(x_k)^T d_k$ utilizando *backtracking* e interpolação quadrática, como proposto no Algoritmo 1. Assumindo que $\bar{\alpha} = \alpha_k$;
7: $\quad x_{k+1} = x_k + \alpha_k d_k$;
8: $\quad k = k + 1$;
9: **fim enquanto**

Saída: x_k

3.3 Convergência

Nessa seção será discutida a convergência global dos algoritmos de descida tipo gradiente. Faremos a demonstração para o método do gradiente projetado visto que, pelo fato de $P_{\mathbb{R}^n}(x) = x$, o caso irrestrito é contemplado.

Os dois resultados a seguir serão úteis ao longo das demonstrações de convergência

dos métodos propostos.

Teorema 3.2. *Considere $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável, $x_k \in \mathbb{R}^n$ e $\Omega \subset \mathbb{R}^n$. Se $d_k = P_\Omega(x_k - \lambda\nabla f(x_k)) - x_k$ para algum $\lambda \in (0, \infty)$ fixo, então*

$$\nabla f(x_k)^t d_k \leq -\frac{1}{\lambda}\|d_k\|^2.$$

Em particular, se θ é o menor ângulo entre $\nabla f(x_k)$ e d_k, temos que

$$\cos(\theta) \leq -\frac{\|d_k\|}{\lambda\|\nabla f(x_k)\|}.$$

Demonstração. Seja $v_k = x_k - \lambda\nabla f(x_k)$ e $p_k = P_\Omega(v_k) = d_k + x_k$. Aplicando o Teorema 2.5, temos que

$$\begin{aligned}(x_k - p_k)^t(v_k - p_k) &= -d_k^t[x_k - \lambda\nabla f(x_k) - (d_k + x_k)] \\ &= -d_k^t(-\nabla f(x_k) - d_k) \\ &= \lambda\nabla f(x_k)^t d_k + \|d_k\|^2 \leq 0\end{aligned}$$

Sendo assim,

$$\nabla f(x_k)^t d_k \leq -\frac{1}{\lambda}\|d_k\|^2,$$

e ainda,

$$\cos(\theta) = \frac{\nabla f(x_k)^t d_k}{\|\nabla f(x_k)\|\|d_k\|} \leq -\frac{\|d_k\|}{\lambda\|\nabla f(x_k)\|},$$

o que conclui a demonstração (veja ilustração na Figura 7). □

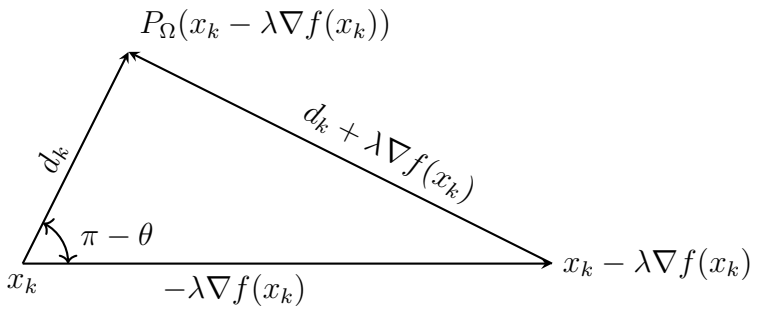


Figura 7: Interpretação geométrica Teorema 3.2.

Lema 3.1. *Se $d_k = P_\Omega(x_k - \nabla f(x_k)) - x_k$ então $\|d_k\| \leq \|\nabla f(x_k)\|$.*

Demonstração. Com o uso do Teorema 2.6 e sabendo que $x_k = P_\Omega(x_k)$ pois $x_k \in \Omega$, temos

que

$$\begin{aligned} \|d_k\| &= \|P_\Omega(x_k - \nabla f(x_k)) - x_k\|, \\ &= \|P_\Omega(x_k - \nabla f(x_k)) - P_\Omega(x_k)\|, \\ &\leq \|x_k - \nabla f(x_k) - x_k\| \\ &= \|\nabla f(x_k)\|. \end{aligned}$$

□

Segue-se a convergência do método do gradiente projetado.

Teorema 3.3. *Se x^* um ponto de acumulação da sequência $\{x_k\}$ gerada pelo Algoritmo 3 então $d^* = P_\Omega(x^* - \nabla f(x^*)) - x^* = 0$.*

Demonstração. Seja $\{x_k\}_{k\in N}$ subsequência de $\{x_k\}$ com limite x^*. Suponha por contradição que $\|d_k\| \geq \epsilon$, $\forall k \in N$ e um certo $\epsilon > 0$ fixo. Do Lema (3.1) segue que

$$\frac{\|d_k\|}{\|\nabla f(x_k)\|} \leq 1.$$

Como $\nabla f(x)$ é contínuo e $x_k \underset{k\in\mathbb{N}}{\to} x^*$, então $\|\nabla f(x_k)\| \leq u = \max\{1, 2\|\nabla f(x^*)\|\}$ para todo $k \in N$ suficientemente grande. Daí,

$$\frac{\epsilon}{u} \leq \frac{\|d_k\|}{\|\nabla f(x_k)\|} \leq 1$$

para todo $N \ni k \gg 1$. Tomando $\delta = \min\{\epsilon/u, 1\} \in (0, 1]$ e usando o Teorema 3.2 com $\lambda = 1$, obtemos $\cos(\theta) \leq -\delta$. Sendo assim,

$$\nabla f(x_k)d_k = \|\nabla f(x_k)\|\|d_k\|\cos(\theta) \leq -\delta\|\nabla f(x_k)\|\|d_k\|. \tag{3.14}$$

Caso 1. $\|\alpha_k d_k\| \geq \epsilon_1$ para algum $\epsilon_1 > 0$, e para infinitos $k \in N$. Da condição de Armijo e (3.14), temos

$$\begin{aligned} f(x_{k+1}) &= f(x_k + \alpha_k d_k), \\ &\leq f(x_k) + \eta\alpha_k \nabla f(x_k)^t d_k, \\ &\leq f(x_k) - \eta\alpha_k\delta\|\nabla f(x_k)\|\|d_k\|, \\ &\leq f(x_k) - \eta\epsilon_1\delta\|\nabla f(x_k)\|, \end{aligned}$$

para todos $k \subset N_1 \subset N$, para certo $N_1 \underset{\infty}{\subseteq} N$, onde $\underset{\infty}{\subseteq}$ denota subconjunto infinito. Daí,

$$\|\nabla f(x_k)\| \leq \frac{f(x_k) - f(x_{k+1})}{\eta\epsilon_1\delta}. \tag{3.15}$$

Como a sequência $\{f(x_k)\}$ é decrescente e $\{x_k\}$ converge sobre N_1, quando tomamos o limite na inequação (3.15) o lado direito é nulo. Assim,

$$\|\nabla f(x^*)\| = \lim_{k \in N_1} \|\nabla f(x_k)\| = 0.$$

Portanto, com o uso do Lema 3.1,

$$\|d_k\| \leq \|\nabla f(x_k)\| \Rightarrow \|d^*\| \leq \|\nabla f(x^*)\| = 0.$$

Chegamos a uma contradição. Sendo assim,

$$d^* = P_\Omega(x^* - \nabla f(x^*)) - x^* = 0.$$

Caso 2. Se $\|\alpha_k d_k\| \to 0$. Se forem feitas infinitas escolhas $\alpha_k = 1$ no algoritmo, digamos $\alpha_k = 1$ para todo $k \in N_2 \underset{\infty}{\subset} N$, então

$$\lim_{k \in N_2} \|d_k\| = \|d^*\| = 0.$$

Sendo assim,

$$d^* = P_\Omega(x^* - \nabla f(x^*)) - x^* = 0.$$

Consideramos agora o caso em que $\alpha = 1$ falha sempre, a partir de certa iteração do método. Ou seja, existe $k_1 \in N$ tal que $\alpha_k < 1$ para todo $k \geq k_1$. Seja $N_3 = \{k \in N; k \geq k_1\} \underset{\infty}{\subset} N$. Seja $\bar{\alpha}_k$ o passo rejeitado imediatamente antes do passo aceito α_k. Considere $\sigma_3 = \sigma_1/\bar{\alpha}_k$, temos que

$$\alpha_k \in [\bar{\alpha}_k \min\{\sigma_3, \rho\},\ \bar{\alpha}_k \max\{\sigma_2, \rho\}]$$

Daí,

$$\frac{\alpha_k}{\min\{\sigma_3, \rho\}} \geq \bar{\alpha}_k.$$

Sendo assim,

$$\|\bar{\alpha}_k d_k\| \leq \frac{1}{\min\{\sigma_3, \rho\}} \|\alpha_k d_k\|. \tag{3.16}$$

Note que, na equação (3.16), o lado esquerdo tende a zero, visto que $\|\alpha_k d_k\| \to 0$.

No método, a condição de Armijo falhou para $\bar{\alpha}_k$, ou seja,

$$f(x_k + \bar{\alpha}_k d_k) > f(x_k) + \eta \bar{\alpha}_k \nabla f(x_k)^t d_k. \tag{3.17}$$

A sequência $\{\bar{\alpha}_k d_k / \|\bar{\alpha}_k d_k\|\}$ é limitada e logo podemos tomar um ponto de acumulação,

digamos

$$v^* = \lim_{k \in N_4} \frac{\bar{\alpha}_k d_k}{\|\bar{\alpha}_k d_k\|}, \quad N_4 \underset{\infty}{\subset} N.$$

De (3.14) temos

$$\nabla f(x_k)^t d_k \leq -\delta \|\nabla f(x_k)\| \|d_k\|.$$

Daí,

$$\nabla f(x_k)^t \frac{\bar{\alpha}_k d_k}{\|\bar{\alpha}_k d_k\|} \leq -\delta \|\nabla f(x_k)\|. \tag{3.18}$$

Tomando o limite de (3.18) sobre N_4, obtemos

$$\nabla f(x^*) v^* \leq -\delta \lim_{k \in N_4} \|\nabla f(x_k)\| = -\delta \|\nabla f(x^*)\|. \tag{3.19}$$

Agora, se olharmos para a função

$$\psi(z) = f(x_k + z\bar{\alpha}_k d_k)$$

e aplicarmos o teorema do valor médio relativo ao intervalo $[0, 1]$, teríamos que existe $z_k \in (0, 1)$ tal que

$$\psi'(z_k) = \frac{\psi(1) - \psi(0)}{1 - 0} = \psi(1) - \psi(0).$$

Ou seja,

$$\bar{\alpha}_k \nabla f(x_k + z_k \bar{\alpha}_k d_k)^t d_k = f(x_k + \bar{\alpha}_k d_k) - f(x_k). \tag{3.20}$$

Portanto, pelo fracasso da condição de Armijo em (3.17), temos

$$f(x_k + \bar{\alpha}_k d_k) - f(x_k) > \eta \bar{\alpha}_k \nabla f(x_k)^t d_k,$$

que com o uso de (3.20), pode ser escrito como

$$\bar{\alpha}_k \nabla f(x_k + z_k \bar{\alpha}_k d_k)^t d_k > \eta \bar{\alpha}_k \nabla f(x_k)^t d_k.$$

Ou ainda, para todo $k \in N_3$, temos

$$\nabla f(x_k + z_k \bar{\alpha}_k d_k)^t \frac{\bar{\alpha}_k d_k}{\|\bar{\alpha}_k d_k\|} > \eta \nabla f(x_k)^t \frac{\bar{\alpha}_k d_k}{\|\bar{\alpha}_k d_k\|}. \tag{3.21}$$

Passando o limite para $k \in N_4$, temos que

$$\nabla f(x^*)^t v^* \geq \eta \nabla f(x^*)^t v^* \quad \Leftrightarrow \quad (1 - \eta) \nabla f(x^*)^t v^* \geq 0. \tag{3.22}$$

sendo assim, como $\eta \in (0, 1)$ e de (3.19) temos que $\nabla f(x^*)^t v^* \leq 0$. Se $\nabla f(x^*) \neq 0$ a expressão (3.22) seria contraditória. Logo, teríamos $\nabla f(x^*) = 0$. O que nos leva a outra

contradição, pois $\epsilon \leq \|d^*\| \leq \|\nabla f(x^*)\|$. Logo,

$$d^* = P_\Omega(x^* - \nabla f(x^*)) - x^* = 0.$$

□

3.4 Método do gradiente espectral projetado

Nessa seção apresentaremos o método do gradiente espectral projetado ou, mais resumidamente, método SPG (do inglês *Spectral Projected Gradient*), descrito por Birgin e Martínez [17], para problemas de otimização da forma (2.1) em que Ω é um conjunto convexo, fechado e não vazio.

3.4.1 Elementos de estratégias quase-newtonianas: a equação secante

Considere $f : \mathbb{R}^n \to \mathbb{R}$ duas vezes diferenciável. Nosso objetivo é encontrar o minimizador da função f ou um candidato a solução, ou seja, devemos resolver

$$\nabla f(x) = 0. \tag{3.23}$$

A resolução direta de (3.23) geralmente é impraticável, pois esse sistema pode ser não linear. Portanto, é comum aplicarmos um processo iterativo em que, dado uma estimativa da solução, obtêm-se novas estimativas que sob certas hipóteses converge ao ótimo.

A ideia do método de Newton é aproximar a função $\nabla f(x)$ em uma vizinhança pelo polinômio de Taylor de primeira ordem. Ou seja, dado uma estimativa x_k, consideramos a seguinte aproximação:

$$\nabla f(x) \approx L_k(x) = \nabla f(x_k) + \nabla^2 f(x_k)(x - x_k).$$

Como o objetivo é resolver (3.23), tomemos x_{k+1} como a solução do sistema linear

$$L_k(x) = 0.$$

Daí, cada iteração do método de Newton consiste em resolver o seguinte sistema linear:

$$\begin{aligned} \nabla^2 f(x_k)s_k &= -\nabla f(x_k), \\ x_{k+1} &= x_k + s_k. \end{aligned} \tag{3.24}$$

Ou ainda, de maneira simplificada

$$x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k). \tag{3.25}$$

Veja a Figura 8.

Para problemas com um grande número de variáveis a aplicação do método de Newton pode se tornar inviável pois o cálculo da hessiana se torna custoso. Mesmo que o método de Newton tenha um alto custo computacional, é muito utilizado em conjunto com outros métodos, devido a ter uma convergência rápida quando os pontos gerados estão próximos da solução. Veja, por exemplo, a estratégia adotada em [18] em um método Lagrangiano aumentado para problemas não lineares com restrições gerais.

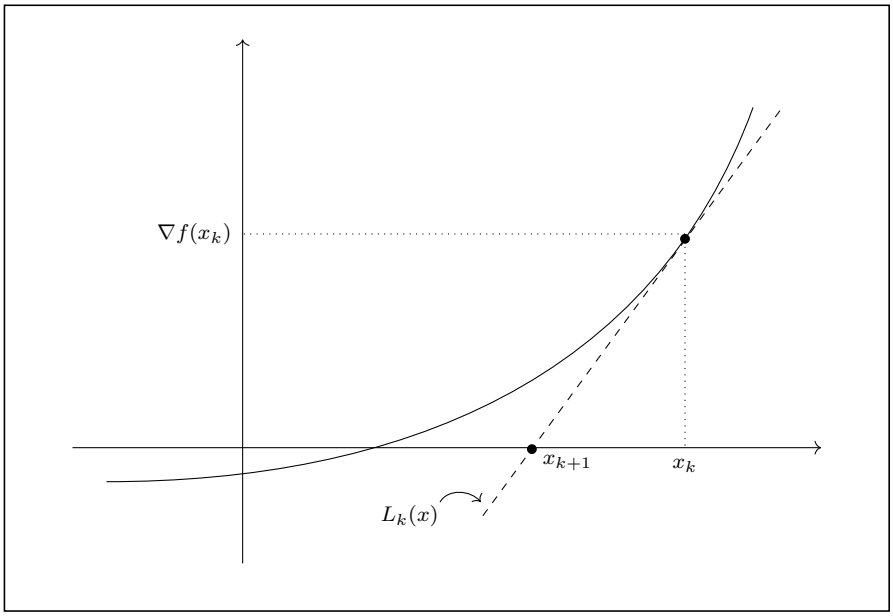


Figura 8: Interpretação geométrica da iteração do método de Newton

Portanto, torna-se interessante o estudo de métodos "baratos" que imitam Newton, os chamados métodos quase-Newton. Tais métodos consistem em trocar o cálculo da $\nabla^2 f(x_k)$ por uma matriz B_k que seja uma boa aproximação para a hessiana ao longo das iterações e facilite a resolução do sistema. Ou seja, em métodos quase-Newton temos o seguinte sistema:

$$\begin{aligned} B_k s_k &= -\nabla f(x_k), \\ x_{k+1} &= x_k + s_k. \end{aligned} \tag{3.26}$$

Ou ainda,

$$x_{k+1} = x_k - B_k^{-1} \nabla f(x_k). \tag{3.27}$$

Uma classe de métodos que se encaixam em estratégias quase-newtonianas são os *métodos secantes*. Tais métodos se baseiam em uma generalização dos clássicos métodos

secantes para o cálculo de zeros de funções de uma variável.

Os métodos secantes consistem em impor que a aproximação linear $L_{k+1}(x)$ interpole os pontos x_{k+1} e x_k. Ou seja,

$$L_{k+1}(x_k) = \nabla f(x_k) \text{ e } L_{k+1}(x_{k+1}) = \nabla f(x_{k+1}).$$

Sendo assim, devemos determinar para quais categorias de matrizes B_k tais imposições são verdadeiras. Logo, considere a aproximação linear de $\nabla f(x)$ na iteração $k+1$,

$$L_{k+1}(x) = \nabla f(x_{k+1}) + B_{k+1}(x - x_{k+1}). \tag{3.28}$$

Repare que a condição $L_{k+1}(x_{k+1}) = \nabla f(x_{k+1})$ é satisfeita pela própria definição da linearização. Já a imposição $L_{k+1}(x_k) = \nabla f(x_k)$, pode ser reescrita como

$$\nabla f(x_{k+1}) + B_{k+1}(x_k - x_{k+1}) = \nabla f(x_k),$$

ou ainda,

$$B_{k+1}s_k = y_k, \tag{3.29}$$

em que $y_k = \nabla f(x_k) - \nabla f(x_{k+1})$. A equação (3.29) é comumente referida como *equação secante*, justamente pela propriedade obtida com sua resolução. Em espaços de dimensões menores, para $n = 1$ ou $n = 2$, pode-se observar a interpretação geométrica de iterações do método secante, observe a Figura 9.

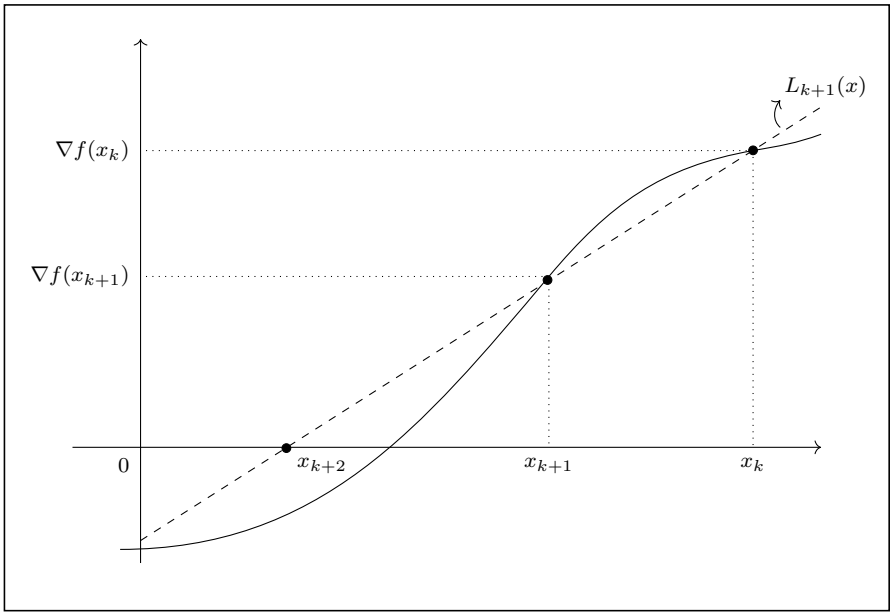


Figura 9: Interpretação geométrica da interação do método secante

Os métodos quase-Newton que não utilizam de uma estratégia de busca linear são

chamados métodos quase-Newton puro, devido a dar sempre o passo completo $\alpha_k = 1$. Mas geralmente é adequado formularmos os métodos quase-Newton na forma

$$x_{k+1} = x_k - \alpha_k B_k^{-1} \nabla f(x_k), \tag{3.30}$$

onde o tamanho do passo $\alpha_k \in (0, 1]$ é obtido por alguma estratégia de busca linear. A escolha adequada de α_k, por exemplo utilizando Armijo, garante convergência global (quando $\alpha_k = 1$ sempre, a convergência é apenas local).

3.4.2 Busca linear não monótoma

Em métodos de busca linear é comum exigirmos que, ao dar o passo, haja um decréscimo suficiente da função objetivo, como abordado na Seção 3.1.1. Tal estratégia, mesmo fornecendo bons resultados, apresenta problemas ao se aproximar da solução visto que exigir um decréscimo da função em toda iteração nos fornece um método lendo próximo da solução.

Nessa seção, abordaremos outra estratégia para aceitação do tamanho do passo, chamada busca linear não monótona, desenvolvida por Grippo, Lampariello e Lucidi [19] e aplicado com sucesso ao método de Newton.

A categoria de métodos de busca não monótona consiste em não exigir um decréscimo da função a cada iteração, mas sim com relação a um histórico do valor funcional em iterações passadas. Nesse caso, considere

$$f_{max} = \max_{0 \leq j \leq m(k)} [f(x_{k-j})],$$

onde $m(k) = \min\{k, M-1\}$ e $M > 0$ inteiro. Na busca linear não monótona, devemos determinar $\bar{\alpha}$ de modo que valha

$$f(x_k + \bar{\alpha} d_k) \leq f_{max} + \eta \bar{\alpha} \nabla f(x_k)^t d_k. \tag{3.31}$$

Ao exigir que o tamanho do passo cumpra (3.31), relaxamos a imposição feita pela condição de Armijo (3.2), e a região de passos aceitáveis aumenta, possibilitando uma maior aceitabilidade dos passos unitários completos.

Para entendermos melhor qual o comportamento da condição não monótona, considere a Figura 10 e compare com a Figura 5. Note que, a região de passos aceitáveis tem a possibilidade de aumentar quando exigimos que o tamanho de passo cumpra (3.31). A busca linear não monótona com *backtracking* e interpolação quadrática é resumido no

Algoritmo 4. Note que a condição de Armijo implica (3.31), pois $f(x_k) \leq f_{max}$. Portanto, o Teorema 3.1 garante que (3.31) seja satisfeita para todo passo suficientemente pequeno, e logo o Algoritmo 4 termina em finitas iterações.

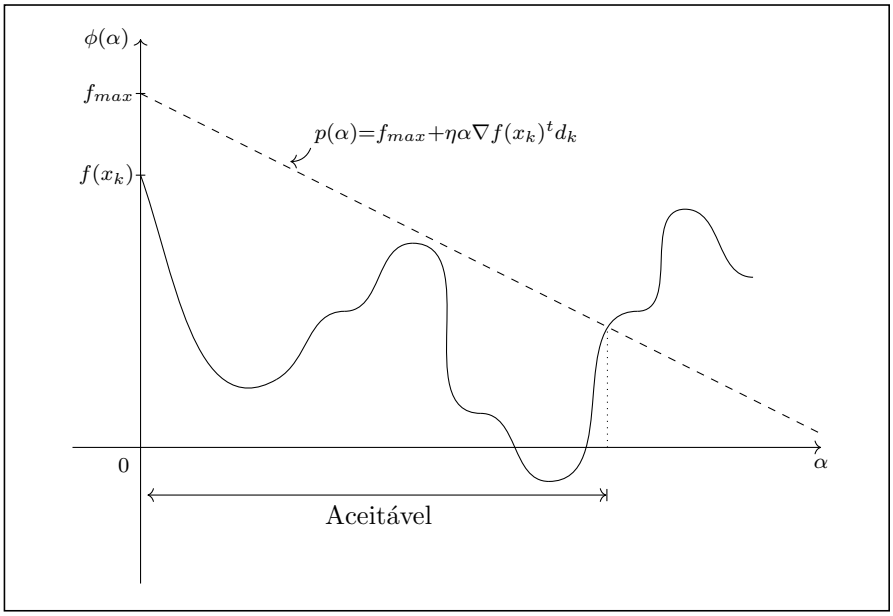


Figura 10: Tamanhos de passo que satisfazem a condição não monótona.

Algoritmo 4 Busca linear não monótona com *backtracking* e interpolação quadrática

Entrada: $\bar{\alpha} > 0$, $0 < \sigma_1 < \sigma_2 < 1$, $\rho \in (0,1)$, $\eta \in (0,1)$, f_{max}, d_k, x_k

Tome $i = 0$ e $\alpha_i = \bar{\alpha}$,
para $f(x_k + \alpha_i d_k) > f_{max} + \eta\ \alpha_i \nabla f(x_k)^t d_k$ **faça**
    se $\alpha_i < \sigma_1$ **então**
        $\alpha_{i+1} = \rho\alpha_i$;
    senão
        Calcule α^* como proposto em (3.4) com $\bar{\alpha} = \alpha_i$;
        se $\alpha^* \in [\sigma_1, \alpha_i\sigma_2]$ **então**
            $\alpha_{i+1} = \alpha^*$,
        senão
            $\alpha_{i+1} = \rho\alpha_i$;
        fim se
    fim se
    $i = i + 1$;
fim para

Saída: $\alpha_k = \alpha_i$

3.4.3 O método

Olhemos para o processo iterativo dos métodos secantes quase-Newton com uma estratégia de busca linear. Nesses métodos, aplicamos o processo iterativo (3.30) onde a

sequência de matrizes $\{B_k\}$ satisfaz a equação secante (3.29). No método SPG assumiremos que a matriz B_k possua uma estrutura simples e aproxime (3.29). De maneira mais específica, impomos que

$$B_k = \sigma_k I, \tag{3.32}$$

sendo $\sigma_k \in \mathbb{R}$. Então de (3.29) temos

$$\sigma_k s_{k-1} = y_{k-1}. \tag{3.33}$$

Observe não haver garantia de que (3.33) possua solução. Logo, estamos interessados em determinar o σ_k que melhor a aproxima em algum sentido. Uma aproximação é obtida a partir da solução do problema de quadrados mínimos

$$\text{Minimizar } \|\sigma s_{k-1} - y_{k-1}\|_2^2 \text{ sujeito a } \sigma \in \mathbb{R},$$

cuja solução

$$\sigma_k = \frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}}$$

é facilmente calculada. Dado $x_0 \in \mathbb{R}^n$ e tomando o *passo espectral* $\lambda_k = 1/\sigma_k$, temos que o método do gradiente espectral para problemas irrestritos é dado pela iteração $x_{k+1} = x_k + \alpha_k d_k$, em que

$$d_k = -\lambda_k \nabla f(x_k).$$

Note que o passo espectral deve ser limitado inferiormente por zero para manter a direção de descida, ao passo que não pode ser muito grande para evitar instabilidades numéricas. Portanto, deve-se acrescentar salvaguardas para λ_k. Sejam λ_{min}, $\lambda_{max} \in \mathbb{R}$ tais que $0 < \lambda_{min} < \lambda_{max} < \infty$. Logo, a cada iteração exigimos que

$$\lambda_k = \max\left\{\lambda_{min}, \min\left\{\frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}, \lambda_{max}\right\}\right\}. \tag{3.34}$$

A salvaguarda (3.34) propõe tomarmos $\lambda_k = \lambda_{\min}$ quando $s_k^t y_k \leq 0$, mas Birgin et al. [3] afirmam que tomarmos $\lambda_k = \lambda_{max}$ é o mais adequado, visto que, daríamos um passo maior na direção de descida d_k e assim contornando o problema naquela região côncava.

Como dito anteriormente, o método do gradiente espectral projetado consiste também de uma projeção assim como a feita para o método do gradiente projetado. Logo, basta tomarmos a direção de busca como a direção projetada da forma

$$d_k = P_\Omega(x_k - \lambda_k \nabla f(x_k)) - x_k. \tag{3.35}$$

No método SPG, exigir que o tamanho de passo α_k satisfaça a condição de Armijo (3.2) não é o ideal. Pois, mesmo nos fornecendo um decréscimo suficiente a cada iteração, faz com que o passo espectral λ_k seja recusado com bastante frequência, levando a resultados práticos ineficientes. Sendo assim, para o método do gradiente espectral projetado, é interessante aplicarmos uma busca linear que convirja e aprove, muitas vezes, o passo espectral. Esse é o caso da busca linear não monótona, abordado na Seção 3.4.2.

O método do gradiente espectral projetado é resumido no Algoritmo 5 e sua convergência abordada na seção seguinte.

Algoritmo 5 Método do gradiente espectral projetado (SPG)

Entrada: $x_0 \in \mathbb{R}^n$, $\lambda_0 \in [\lambda_{min}, \lambda_{max}]$, $\rho \in (0,1), \eta \in (0,1)$, $M \geq 1$ e $\epsilon > 0$.

Tome $k \leftarrow 0$;
Se $x_0 \notin \Omega$ então $x_0 \leftarrow P_\Omega(x_0)$;
enquanto $\|d_k\| > \epsilon$ **faça**
    $d_k = P_\Omega(x_k - \lambda_k \nabla f(x_k)) - x_k$,
    Calcule $f_{max} = \max\{f(x_{k-j});\ 0 \leq j \leq \min\{k, M-1\}\ \}$;
    Tome $\alpha_k = 1$;
    Reduza α_k até que $f(x_k + \alpha_k d_k) \leq f_{max} + \eta \alpha_k \nabla f(x_k)^T d_k$ utilizando *Backtracking* e interpolação quadrática, como proposto no Algoritmo 4. Assumindo que $\bar{\alpha} = \alpha_k$;
    $x_{k+1} = x_k + \alpha_k d_k$;
    $s_k = x_{k+1} - x_k$;
    $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$;
    se $s_k^T y_k \leq 0$ **então**
        $\lambda_{k+1} = \lambda_{max}$
    senão
        $\lambda_{k+1} = \min\{\lambda_{\max}, \max\{\lambda_{\min}, s_k^T s_k / s_k^T y_k\}\}$;
    fim se
    $k = k + 1$;
fim enquanto

3.4.4 Convergência

Nesta seção exibimos os resultados de boa definição e de convergência para o método SPG.

Lema 3.2. *Seja* $g_\lambda(x) = P_\Omega(x - \lambda \nabla f(x)) - x$. *Para todo* $x \in \Omega$ *e* $\lambda \in (0, \lambda_{max}]$ *temos*

$$\nabla f(x)^t g_\lambda(x) \leq -\frac{1}{\lambda}\|g_\lambda(x)\|^2 \leq -\frac{1}{\lambda_{max}}\|g_\lambda(x)\|^2.$$

Demonstração. Segue do Teorema 3.2, notando que $-1/\lambda \leq -1/\lambda_{max}$. □

Teorema 3.4. *O Algoritmo 5 está bem definido, e qualquer ponto de acumulação* x^* *da sequência* $\{x_k\}$ *gerada pelo método é um ponto estacionário do problema* (3.2).

Demonstração. Considere a seguinte notação:

$$g_\lambda(x) = P_\Omega(x - \lambda\nabla f(x)) - x, \quad d_k = g_{\lambda_k}(x_k), \quad m(k) = \min\{k, M-1\}.$$

Se x_k não é um ponto estacionário então pelo Lema 3.2

$$\nabla f(x_k)^t d_k = \nabla f(x_k)^t g_{\lambda_k}(x_k) \leq -\frac{1}{\lambda_{max}}\|g_{\lambda_k}(x_k)\|^2 < 0, \tag{3.36}$$

e a direção de busca d_k é uma direção de descida. Portanto, um tamanho de passo que satisfaça (3.31) será encontrado após um número finito de tentativas, e o algoritmo está bem definido.

Seja $x^* \in \Omega$ um ponto de acumulação da sequência $\{x_k\}$. Renomeie $\{x_k\}$ como uma subsequência que converge a x^*. Consideraremos os dois casos abaixo.

Caso 1. Assuma que $\inf \alpha_k = 0$. Suponha por contradição que x^* não seja um ponto estacionário. Pela continuidade $\nabla f(x)$ e $g_\lambda(x)$, existe $\delta > 0$ tal que

$$\nabla f(x^*)^t \frac{g_\lambda(x^*)}{\|g_\lambda(x^*)\|} < -\delta, \quad \forall\lambda \in [\lambda_{min}, \lambda_{max}],$$

que implica em

$$\nabla f(x_k)^t \frac{g_\lambda(x_k)}{\|g_\lambda(x_k)\|} < -\frac{\delta}{2}, \quad \forall\lambda \in [\lambda_{min}, \lambda_{max}], \tag{3.37}$$

para todo k suficientemente grande. Visto que $\inf \alpha_k = 0$, existe uma subsequência $\{x_k\}_{k\in K}$ com $K \underset{\infty}{\subset} \mathbb{N}$ tal que

$$\lim_{k\in K} \alpha_k = 0.$$

Nesse caso, da forma como α_k é escolhido, existe $\bar{k}$ suficientemente grande tal que para todo $k \geq \bar{k},\ k \in K$, existe ρ_k, $0 < \sigma_1 < \rho_k < \sigma_2$, para o qual $\alpha_k/\rho_k > 0$ falha em satisfazer a condição (3.31). Ou seja,

$$f\left(x_k + \frac{\alpha_k}{\rho_k} d_k\right) > f_{max} + \eta\left(\frac{\alpha_k}{\rho_k}\right)\nabla f(x_k)^t d_k \geq f(x_k) + \eta\left(\frac{\alpha_k}{\rho_k}\right)\nabla f(x_k)^t d_k.$$

Portanto,

$$\frac{f\left(x_k + \frac{\alpha_k}{\rho_k} d_k\right) - f(x_k)}{(\alpha_k/\rho_k)} > \eta\nabla f(x_k)^t d_k.$$

Com a aplicação o Teorema do Valor Médio, essa expressão pode ser reescrita como

$$\nabla f(x_k + t_k d_k)^t d_k > \eta\nabla f(x_k)^t d_k, \tag{3.38}$$

para todo $k \in K, k > \bar{k}$, onde $t_k \in [0, \alpha_k/\rho_k]$ em que $\lim_{k\in K} t_k = 0$. Tomemos uma subsequência conveniente $d_k/\|d_k\|$ cujo limite é d. Dividindo ambos os lados de (3.38) por

$\|d_k\|$ e tomando o limite, obtemos

$$\nabla f(x^*)^t d > \eta \nabla f(x^*)^t d.$$

Daí,

$$(1-\eta)\nabla f(x^*)d \geq 0.$$

Como $(1-\eta) > 0$ e $\nabla f(x_k)^t d_k < 0$, de (3.36) segue que $\nabla f(x^*)^t d = 0$. Pela continuidade e definição de d_k, implica que, para k suficientemente grande naquela subsequência, temos que

$$\nabla f(x_k)^t \frac{g_{\lambda_k}(x_k)}{\|g_{\lambda_k}(x_k)\|} > -\frac{\delta}{2},$$

o que contradiz (3.37). Portanto, x^* é estacionário.

Caso 2. Assuma que $\inf \alpha_k \geq \rho > 0$. Suponha por contradição que x^* não seja um ponto estacionário. Pela continuidade $\nabla f(x)$ e $g_\lambda(x)$, existe $\delta > 0$ tal que $\|g_\lambda(x^*)\| \geq \delta > 0 \ \ \forall \lambda \in [\rho, \lambda_{max}]$. Seja $l(k)$ um inteiro tal que

$$k - m(k) \leq l(k) \leq k. \tag{3.39}$$

Afirmamos que a sequência

$$f(x_{l(k)}) = \max_{0\leq j\leq m(k)} f(x_{k-j})$$

é monótona não crescente. De fato, sabendo que $m(k+1) \leq m(k)+1$ (direto da definição de $m(k)$) temos,

$$\begin{aligned}
f(x_{l(k+1)}) &= \max_{0\leq j\leq m(k+1)} [f(x_{k+1-j})] \\
&\leq \max_{0\leq j\leq m(k)+1} [f(x_{k+1-j})] \\
0 &= \max\left\{ \max_{1\leq j\leq m(k)+1} [f(x_{k+1-j})], f(x_{k+1}) \right\} \\
&= \max\left\{ \max_{0\leq j-1\leq m(k)} [f(x_{k-(j-1)})], f(x_{k+1}) \right\} \\
&= \max[f(x_{l(k)}), f(x_{k+1})] = f(x_{l(k)}).
\end{aligned}$$

Além disso, de (3.31) segue que para $k > M-1$

$$\begin{aligned}
f(x_{l(k)}) &= f(x_{l(k)-1} + \alpha_{l(k)-1} d_{l(k)-1}) \\
&\leq \max_{0\leq j\leq m(l(k)-1)} [f(x_{l(k)-1-j})] + \eta \alpha_{l(k)-1} \nabla f(x_{l(k)-1})^t d_{l(k)-1} \\
&= f(x_{l(l(k)-1)}) + \eta \alpha_{l(k)-1} \nabla f(x_{l(k)-1})^t d_{l(k)-1}.
\end{aligned}$$

Ou seja,

$$f(x_{l(k)}) \leq f(x_{l(l(k)-1)}) + \eta\alpha_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1}. \tag{3.40}$$

Agora, visto que $f(x_k) \leq f(x_0)$ para todo k e $x_k \in \Omega_0 = \{x; f(x) \leq f(x_0)\}$, a sequência $f(x_{l(k)})$ admite limite para $k \to \infty$. Além disso, dado que $\alpha_k > 0$ e $\nabla f(x_k)^t d_k < 0$, segue de (3.40) que

$$\lim_{k\to\infty} \alpha_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1} = 0. \tag{3.41}$$

Do Lema 3.2 segue que $\alpha_k \nabla f(x_k)^t d_k \leq -(\alpha_k/\lambda_{max})\|d_k\|^2$ para todo k. Como $\alpha_k \leq \bar{\alpha}$, temos que (3.41) implica

$$\lim_{k\to\infty} \alpha_{l(k)-1}\|d_{l(k)-1}\| = 0. \tag{3.42}$$

Provaremos agora que $\lim_{k\to\infty} \alpha_k\|d_k\| = 0$. Seja

$$\hat{l}(k) \equiv l(k + M + 2).$$

Primeiramente, mostraremos por indução que para qualquer $j \geq 1$, temos que

$$\lim_{k\to\infty} \alpha_{\hat{l}(k)-j}\|d_{\hat{l}(k)-j}\| = 0 \tag{3.43}$$

e

$$\lim_{k\to\infty} f(x_{\hat{l}(k)-j}) = \lim_{k\to\infty} f(x_{l(k)}). \tag{3.44}$$

Assuma, sem perda de generalidade, que k seja suficientemente grande a ponto de evitar a ocorrência de índices negativos. Se $j = 1$, visto que $\hat{l}(k)$ é uma subsequência de $l(k)$, de (3.42) segue que (3.43) é válido. Isto implica que $\alpha_{\hat{l}(k)-1}\|d_{\hat{l}(k)-1}\| = \|x_{\hat{l}(k)} - x_{\hat{l}(k)-1}\| \to 0$, de modo que (3.44) vale para $j = 1$, visto que f é uniformemente contínua. Agora, assuma que (3.43) e (3.44) sejam verdadeiras para um dado j. Então de (3.40) pode-se escrever

$$f(x_{\hat{l}(x)-j}) \leq f(x_{l(\hat{l}(k)-j-1)}) + \eta\alpha_{\hat{l}(k)-j-1}\nabla f(x_{\hat{l}(k)-j-1})^t d_{\hat{l}(k)-j-1},$$

que tomando o limite para $k \to \infty$, temos de (3.44)

$$\lim_{k\to\infty} \alpha_{\hat{l}(k)-(j+1)}\nabla f(x_{\hat{l}(k)-(j+1)})^t d_{\hat{l}(k)-(j+1)} = 0.$$

Daí, com o uso mesmo argumento que determina (3.42), temos

$$\lim_{k\to\infty} \alpha_{\hat{l}(k)-(j+1)}\|d_{\hat{l}(k)-(j+1)}\| = 0.$$

Isso implica que $\|x_{\hat{l}(k)-j} - x_{\hat{l}(k)-(j+1)}\| \to 0$ e então, de (3.44) e da continuidade de f, temos que

$$\lim_{k\to\infty} f(x_{\hat{l}(k)-(j+1)}) = \lim_{k\to\infty} f(x_{\hat{l}(k)-j}) = \lim_{k\to\infty} f(x_{l(k)})$$

Portanto, (3.43) e (3.44) são válidos para qualquer $j \geq 1$. Agora, para qualquer k temos

$$\begin{aligned} x_{\hat{l}(k)} &= x_{k+1} + \alpha_{k+1}d_{k+1} + \alpha_{k+2}d_{k+2} + \cdots + \alpha_{\hat{l}(k)-1}d_{\hat{l}(k)-1} \\ &= x_{k+1} + \sum_{j=1}^{\hat{l}(k)-k-1} \alpha_{\hat{l}(k)-j}d_{\hat{l}(k)-j}, \end{aligned}$$

ou ainda,

$$x_{k+1} = x_{\hat{l}(k)} - \sum_{j=1}^{\hat{l}(k)-k-1} \alpha_{\hat{l}(k)-j}d_{\hat{l}(k)-j}. \tag{3.45}$$

Por (3.39) temos

$$\hat{l}(k) - k - 1 = l(k + M + 2) - k - 1 \leq M + 1,$$

de modo que (3.45), com (3.43), implica

$$\lim_{k\to\infty} \|x_{k+1} - x_{\hat{l}(k)}\| = 0.$$

Pela continuidade de f temos

$$\lim_{k\to\infty} f(x_k) = \lim_{k\to\infty} f(x_{\hat{l}(k)}) = \lim_{k\to\infty} f(x_{l(k)}) = f(x^*). \tag{3.46}$$

E ainda, para $k > \bar{k}$ suficientemente grande $\|g_{\lambda_k}(x_k)\| \geq \delta/2$. Portanto, usando o Lema 3.2 obtemos

$$f(x_{l(k)}) \leq f(x_{l(l(k)-1)}) - \frac{\eta\rho}{\lambda_{max}}\|g_{\lambda_{l(k)-1}}(x_{l(k)-1})\|^2 \leq f(x_{l(l(k)-1)}) - \frac{\eta\delta^2\rho}{4\lambda_{max}}.$$

Assim, quando $k \to \infty$, temos que $f(x_{l(k)}) \to -\infty$, o que é uma contradição tendo em vista (3.46) e a continuidade de f em x^*. Logo, x^* é estacionário. □

4 Método de direções conjugadas

Nesse capítulo abordaremos o método de direções conjugadas. Tais métodos são muito utilizados para a resolução de problemas de grande porte, devido a usar apenas informações da função e do gradiente e ser mais veloz que os métodos de descida do gradiente, com um custo computacional um pouco maior. Primeiramente, falaremos do caso clássico aplicado à problemas quadráticos, e posteriormente da sua generalização à problemas não quadráticos. Na última seção, exibimos os testes computacionais aplicados a problemas irrestritos e com restrições de caixa. Algumas referências para esse capítulo foram [7, 11, 20, 21]

4.1 Método de direções conjugadas para quadráticas

Apresentaremos agora o método de direções conjugadas para a resolução de problemas da forma

$$\text{Minimizar} \quad f(x) = \frac{1}{2}x^t Ax - b^t x, \tag{4.1}$$

onde $A \in \mathbb{R}^{n\times n}$ simétrica e definida positiva, $b \in \mathbb{R}^n$. Repare que devido à matriz A ser definida positiva a função f possui único minimizador global. De fato, a condição necessária de primeira ordem $\nabla f(x) = 0$ consiste no sistema $Ax = b$, que possui solução única dado que, sendo A definida positiva, A é inversível. Ademais, é possível mostrar via condições de segunda ordem, que tal ponto é minimizador, e não maximizador. Veja, por exemplo [11].

Como foi visto em métodos anteriores, um fator decisivo para que um método tenha uma boa eficiência é uma escolha cuidadosa para as direções de busca. Em problemas da forma (4.1), veremos que uma direção adequada é aquela que satisfaz a propriedade de A-conjugação, descrita a seguir.

Definição 4.1. *Seja $A \in \mathbb{R}^{n\times n}$ uma matriz simétrica e definida positiva. Dizemos que os vetores $d_0, d_1, \ldots, d_k \in \mathbb{R}^n$ não nulos, são A-conjugados se $d_i^t A d_j = 0$ para todos*

$i, j = 0, 1, \ldots, k$ *com* $i \neq j$.

Outra propriedade importante é a independência linear de um conjunto de vetores A-conjugados, abordada no lema a seguir.

Lema 4.1. *Seja* $A \in \mathbb{R}^{n\times n}$ *uma matriz simétrica e definida positiva. Se as direções* $d_0, d_1, \ldots, d_k \in \mathbb{R}^n, k < n-1$, *são A-conjugadas, então o conjunto* $\{d_0, d_1, \ldots, d_k\}$ *é linearmente independente.*

Demonstração. Sejam $\alpha_0, \alpha_1, \ldots, \alpha_k$ números reais tais que

$$\alpha_0 d_0 + \alpha_1 d_1 + \cdots + \alpha_k d_k = 0.$$

Multiplicando toda a equação acima por $d_j^t A, 1 \leq j \leq k$, temos

$$\alpha_j d_j^t A d_j = 0,$$

da onde se segue que $\alpha_j = 0$, pois A é definida positiva. □

O método de direções conjugadas consiste em, dado $x_0 \in \mathbb{R}^n$ e um conjunto de direções $\{d_0, d_1, \ldots, d_{n-1}\}$ com a propriedade de A-conjugação, geramos uma sequência $\{x_k\}$ a partir do seguinte processo iterativo:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{4.2}$$

onde o tamanho do passo α_k pode ser obtido de maneria exata devido à estrutura do problema quadrático. Portanto, tomemos $\alpha_k = \arg\min\{f(x_k + \alpha d_k); \alpha \in \mathbb{R}\}$. Logo, vale

$$\nabla f(x_k + \alpha_k d_k)^t d_k = 0, \tag{4.3}$$

e como

$$\begin{aligned} \nabla f(x_k + \alpha_k d_k) &= A(x_k + \alpha_k d_k) - b \\ &= (Ax_k - b) + \alpha_k A d_k \\ &= \nabla f(x_k) + \alpha_k A d_k, \end{aligned}$$

substituindo em (4.3), obtemos

$$0 = [\nabla f(x_k) + \alpha_k A d_k]^t d_k = \nabla f(x_k)^t d_k + \alpha_k d_k^t A d_k,$$

o que implica em

$$\alpha_k = -\frac{\nabla f(x_k)^t d_k}{d_k^t A d_k}. \tag{4.4}$$

Cabe ressaltar que a busca linear é feita em toda reta, devido às direções de busca nem sempre serem de descida, e a expressão (4.4) está bem definida, pois $d_k^t A d_k > 0$ graças à positividade da matriz A.

Os métodos de direções conjugadas possuem diversas propriedades. Uma das mais interessantes é a sua terminação finita, em no máximo n iterações quando o objetivo é minimizar uma quadrática em $\mathbb{R}^n$. Tal propriedade é abordada no teorema a seguir.

Teorema 4.1. *Para qualquer ponto inicial $x_0 \in \mathbb{R}^n$, o algoritmo básico de direções conjugadas da Equação (4.2) converge para o único x^* (que resolve o sistema $Ax = b$) em n iterações; ou seja, $x_n = x^*$.*

Demonstração. Considere $x^* - x_0 \in \mathbb{R}^n$. Como o conjunto $\{d_0, d_1, \ldots, d_{n-1}\}$ é linearmente independente e gera $\mathbb{R}^n$, existem constantes $\beta_i, i = 0, 1, \ldots, n-1$ tais que

$$x^* - x_0 = \beta_0 d_0 + \beta_1 d_1 + \cdots + \beta_{n-1} d_{n-1}.$$

Agora, multiplicamos ambos os lados dessa equação por $d_k^t A, k = 0, 1, \ldots, n-1$, para obter

$$d_k^t A(x^* - x_0) = \beta_k d_k^t A d_k,$$

onde $d_k A d_i = 0, k \neq i$ devido à propriedade de A-conjugação. Sendo assim,

$$\beta_k = \frac{d_k^t A(x^* - x_0)}{d_k^t A d_k}.$$

Pela definição de x_k, podemos escrever

$$x_k = x_0 + \alpha_0 d_0 + \alpha_1 d_1 + \cdots + \alpha_{k-1} d_{k-1}.$$

Daí,

$$x_k - x_0 = \alpha_0 d_0 + \alpha_1 d_1 + \cdots + \alpha_{k-1} d_{k-1}.$$

Note ainda que $x^* - x_0 = (x^* - x_k) + (x_k - x_0)$ e multiplicando ambos os lados por $d_k^t A$, obtemos

$$\begin{aligned} d_k^t A(x^* - x_0) &= d_k^t A(x^* - x_k) + d_k^t A(x_k - x_0) \\ &= d_k^t (Ax^* - Ax_k) \\ &= d_k^t (b - Ax_k) \\ &= -d_k^t \nabla f(x_k). \end{aligned}$$

Então,

$$\beta_k = -\frac{d_k^t \nabla f(x_k)}{d_k^t A d_k} = \alpha_k.$$

Com isso, temos que

$$x^* = x_0 + \alpha_0 d_0 + \alpha_1 d_1 + \cdots + \alpha_{n-1} d_{n-1} = x_n,$$

o que completa a demonstração. □

Como foi observado e demonstrado no Teorema 4.1, o método de direções conjugadas converge em no máximo n iterações, mas é possível demonstrar uma propriedade ainda mais forte: que o método converge em no máximo r iterações, onde r é o número de autovalores distintos da matriz A. Uma argumentação mais elaborada desta característica é discutida por Nocedal [7].

Caso tenhamos o conjunto de direções conjugadas $\{d_0, d_1, \ldots, d_{n-1}\}$, o método de direções conjugadas é muito eficiente e pode ser aplicado com sucesso à problemas quadráticos com um grande número de variáveis. No entanto, esse conjunto de direções não está disponível previamente. Pelo contrário, o geramos ao longo das iterações a partir de uma combinação linear da direção anterior com a oposta ao do vetor gradiente atual. Portanto, iniciando com $x_0 \in \mathbb{R}^n$ e $d_0 = -\nabla f(x_k)$, construímos indutivamente as direções tomando

$$d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k, \tag{4.5}$$

e determinemos β_k de modo que as direções d_k e d_{k+1} sejam A-conjugadas. Ou seja,

$$d_k^t A d_{k+1} = 0. \tag{4.6}$$

Substituindo (4.5) em (4.6), obtemos

$$0 = d_k^t A(-\nabla f(x_{k+1}) + \beta_k d_k) = -d_k^t A \nabla f(x_{k+1}) + \beta_k d_k^t A d_k.$$

Daí,

$$\beta_k = \frac{d_k^T A \nabla f(x_{k+1})}{d_k^T A d_k}. \tag{4.7}$$

O cálculo de β_k, dada por (4.7), muitas vezes pode ser caro devido aos produtos matriz-vetor. Portanto, apresentamos outras maneiras de se calcular este coeficiente. Uma delas foi proposta em 1969 por Polak e Ribière [22], dada por

$$\beta_k^{PR} = \frac{\nabla f(x_{k+1})^t [\nabla f(x_{k+1}) - \nabla f(x_k)]}{\nabla f(x_k)^t \nabla f(x_k)}. \tag{4.8}$$

Outra é a definida em 1964 por Fletcher e Reeves [23],

$$\beta_k^{FR} = \frac{\nabla f(x_{k+1})^t \nabla f(x_{k+1})}{\nabla f(x_k)^t \nabla f(x_k)}. \tag{4.9}$$

No caso onde a função f é quadrática e o tamanho de passo é dado por (4.4), é possível demonstrar que $\beta_k = \beta_k^{PR} = \beta_k^{FR}$ (Teorema 5.20 de [11]). O método de direções conjugadas clássico é resumido no Algoritmo 6.

O método de direções conjugadas, com a estratégia que gera as direções a cada iteração, é comumente chamado método de gradientes conjugados ou, mais resumidamente método CG (do inglês *Conjugate Gradient*), mesmo que não haja conjugação de gradientes, mas sim de direções. Uma explicação possível para a escolha de tal nomenclatura, se deve ao fato das direções serem a combinação do gradiente com uma direção que exigimos que seja A-conjugada.

Algoritmo 6 Método de direções conjugadas para quadráticas

Dados: $A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n$, $x_0 \in \mathbb{R}^n$, $\epsilon > 0$
Tome $g_0 = \nabla f(x_0)$, $d_0 = -g_0$
para $\|\nabla f(x_k)\|_2 > \epsilon$ **faça**
    $w_k = Ad_k$,
    $z_k = d_k^T w_k$,
    $\alpha_k = -\frac{g_k^T d_k}{z_k}$,
    $x_{k+1} = x_k + \alpha_k d_k$,
    $g_{k+1} = g_k + \alpha_k z_k \quad (g_{k+1} = \nabla f(x_{k+1}))$
    $\beta_k = \frac{w_k^T g_{k+1}}{z_k}$,
    $d_{k+1} = -g_{k+1} + \beta_k d_k$,
    $k \leftarrow k + 1$.
fim para

4.2 Método de direções conjugadas para funções não quadráticas

Na presente seção, apresentaremos uma generalização do método de direções conjugadas para funções não quadráticas, chamado método de gradientes conjugados descendente, ou resumidamente CG-descent (do inglês *Conjugate Gradiente Descent*). O método foi proposto por Hager e Zhang [5] e apresenta ótimos resultados aplicados à problemas irrestritos, em particular para problemas cuja função objetivo são similares à quadráticas.

4.2.1 Busca inexata: condições de Wolfe

Anteriormente, apresentamos a *condição de Armijo*, que permite determinar se um dado tamanho de passo α_k nos fornece um decréscimo suficiente de $f(x)$ na direção de descida d_k. Tal condição é retomada a seguir:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \eta \alpha_k \nabla f(x_k)^t d_k, \tag{4.10}$$

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

A condição (4.10), mesmo nos fornecendo tamanhos de passo eficazes, não é suficiente para que um dado algoritmo efetue um bom progresso, visto que ela é satisfeita para todo α_k suficientemente pequeno como foi observado no Teorema 3.1. Sendo assim, para descartar passos muito pequenos, acrescentamos uma segunda condição, chamada *condição de curvatura*, a qual requer que α_k satisfaça

$$\nabla f(x_k + \alpha_k d_k)^t d_k \geq \xi \nabla f(x_k)^t d_k, \tag{4.11}$$

para alguma constante $\xi \in (\eta, 1)$, onde η é o mesmo de (4.10).

Chamando $\phi(\alpha) = f(x_k + \alpha d_k)$, observe que o lado esquerdo de (4.11) é a derivada $\phi'(\alpha_k)$. Assim a condição de curvatura garante que a inclinação da reta tangente em α_k seja maior que ou igual a ξ vezes a inclinação inicial $\phi'(0)$. Isso faz sentido, visto que, se $\phi'(0)$ for muito negativa, teríamos um indício de que se dermos um passo maior na direção de busca, possivelmente haveria um decréscimo maior de $f(x)$.

Por outra perspectiva, se $\phi'(0)$ for apenas um pouco negativa, é um sinal de que não podemos esperar uma diminuição muito grande da função f na direção proposta. A condição de curvatura é ilustrada na Figura 11. Valores típicos de ξ e η serão melhor discutidos na seção de testes computacionais.

A condição de Armijo com a condição de curvatura são normalmente chamadas *condições de Wolfe*. Na Figura 12 apresentamos os tamanhos de passo aceitos, quando exigimos as condições de Wolfe, dadas a seguir:

$$\begin{aligned} f(x_k + \alpha_k d_k) &\leq f(x_k) + \eta \alpha_k \nabla f(x_k)^t d_k, & (4.12) \\ \nabla f(x_k + \alpha_k d_k)^t d_k &\geq \xi \nabla f(x_k)^t d_k, & (4.13) \end{aligned}$$

com $0 < \eta < \xi < 1$. Pode-se mostrar que se a função f é suave e limitada inferiormente, existe um tamanho de passo que satisfaz as condições de Wolfe (Lema 3.1 de [7]).

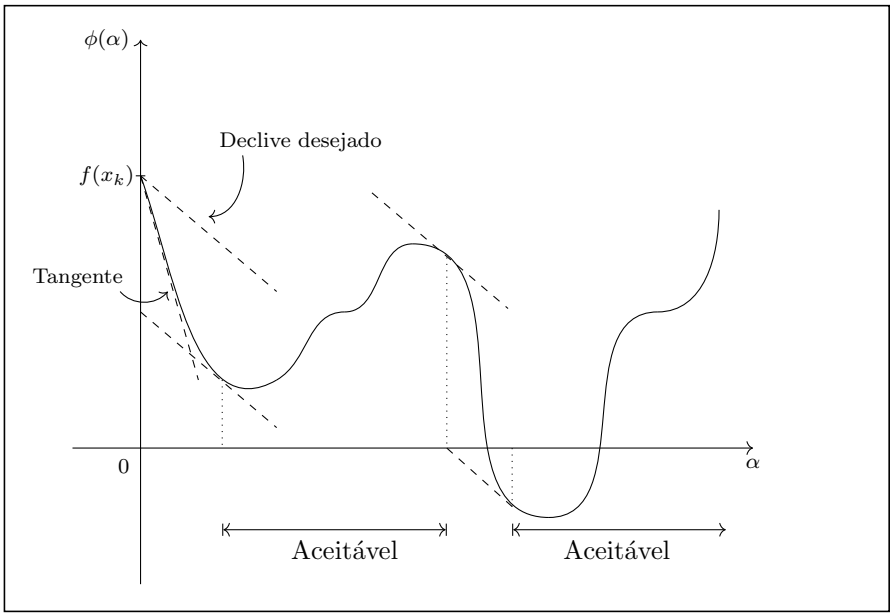


Figura 11: Tamanhos de passo que satisfazem a condição de curvatura.

As condições de Wolfe são essenciais nos métodos que serão abordados posteriormente, visto que tem um papel fundamental para garantir a convergência do método CG-descent e a boa definição do método BFGS, discutido no Capítulo 5.

Considerando $\phi(\alpha) = f(x_k + \alpha d_k)$, as condições de Wolfe podem ser reescritas como

$$\eta\phi'(0) \geq \frac{\phi(\alpha_k) - \phi(0)}{\alpha_k} \quad \text{e} \quad \phi'(\alpha_k) \geq \xi\phi'(0). \tag{4.14}$$

Numericamente, a primeira condição de (4.14) tem dificuldade em ser satisfeita em uma vizinhança do minimizador local, desde que $\phi(\alpha) \approx \phi(0)$ e a operação $\phi(\alpha_k) - \phi(0)$ é relativamente imprecisa. Diante disso, apresentamos as *condições aproximadas de Wolfe*

$$(2\eta - 1)\phi'(0) \geq \phi'(\alpha_k) \quad \text{e} \quad \phi'(\alpha_k) \geq \xi\phi'(0), \tag{4.15}$$

onde $\eta < \min\{(0,5), \xi\}$.

Repare que a desigualdade direita de (4.15) equivale à condição de curvatura e a esquerda a uma aproximação da condição de Armijo. Para verificar isso, basta substituir $\phi(\alpha)$ na primeira condição de (4.14), pela interpolação quadrática $p(\alpha)$ no ponto $(0, \phi(0))$, com $p'(0) = \phi'(0)$ e $p'(\alpha_k) = \phi'(\alpha_k)$, dada a seguir,

$$p(\alpha) = \frac{\phi'(\alpha_k) - \phi'(0)}{2\alpha_k}\alpha^2 + \phi'(0)\alpha + \phi(0). \tag{4.16}$$

A estratégia para determinar $p(\alpha)$ é semelhante à abordada na Seção 3.1.1. Mediante a troca, é possível mostrar que (4.16) nos fornece uma melhor aproximação para a condição

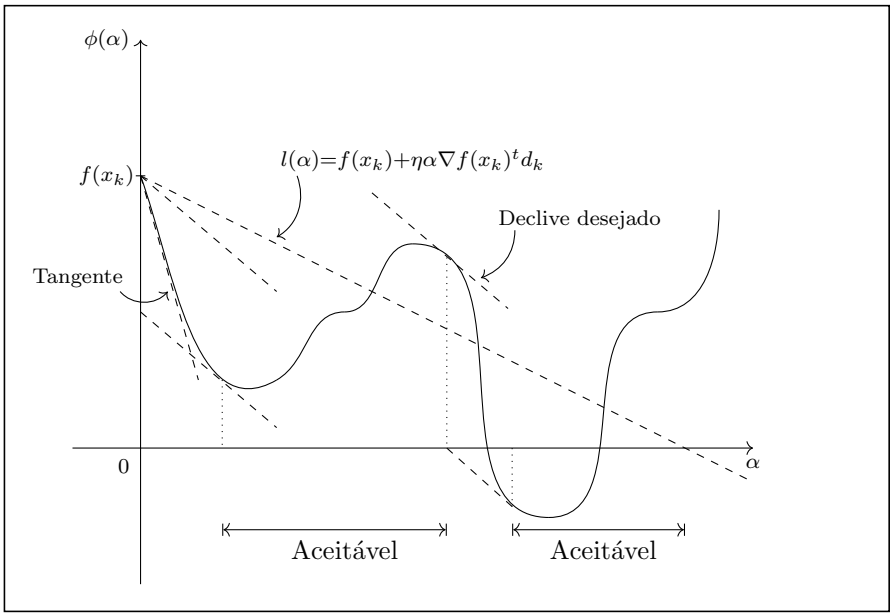


Figura 12: Tamanhos de passo que satisfazem as condições de Wolfe.

de Armijo próximo ao minimizador de $\phi(\alpha)$. Uma discussão mais elaborada é apresentada por Hager e Zhang [5].

Definida as condições de Wolfe e as condições aproximadas de Wolfe, trataremos agora de estratégias de busca linear que as cumpra. Em particular, abordaremos a proposta por Hager e Zhang [5] que chamaremos *busca de Hager-Zhang.*

A busca de Hager-Zhang considera o fator de estabilidade numérica ligado à primeira condição de Wolfe (4.12). Assim, a busca será encerrada quando uma das condições a seguir for verificada:

1) O tamanho do passo satisfaz as condições de Wolfe (4.12)–(4.13).

2) As condições aproximadas de Wolfe (4.15) são satisfeitas sempre que

$$f(x_k + \alpha_k d_k) - f(x_k) < \epsilon_k,$$

onde $\epsilon_k \geq 0$ é uma estimativa para o erro do valor funcional na iteração k. Segundo Hager e Zhang [5], podemos tomar

$$\epsilon_k = \omega|f(x_k)|, \tag{4.17}$$

onde $\omega \in (0, 1)$.

Assim sendo, sempre que possível queremos que o tamanho do passo satisfaça as condições de Wolfe. Em contrapartida, sempre que x_{k+1} e x_k forem muito próximos,

erros numéricos podem fazer com que as condições de Wolfe sejam violadas, nesse caso se verificam as condições aproximadas de Wolfe.

De modo a obter um tamanho de passo α_k que satisfaça 1) ou 2), utilizaremos a estratégia de *Backtracking*, similar a realizada na Seção 3.1.1. Inicialmente, buscaremos um intervalo $[\bar{a}, \bar{b}] \subset [a, b]$, para $a, b \in \mathbb{R}$ dado, que satisfaça as condições de inclinação oposta, isto é,

$$\phi(a) \leq \phi(a) + \epsilon_k,\ \phi'(a) < 0 \text{ e } \phi'(b) \geq 0. \tag{4.18}$$

Considere $c \in \mathbb{R}$ obtido por uma iteração do método secante, aplicado a função ϕ' considerando o intervalo arbitrário $[a, b] \subset \mathbb{R}$, ou seja,

$$c = \text{secante}(a, b) = \frac{a\phi'(b) - b\phi'(a)}{\phi'(b) - \phi'(a)}. \tag{4.19}$$

A expressão (4.19) é facilmente calculada, basta determinar o zero da função linear que interpola os pontos $(a, \phi'(a))$ e $(b, \phi'(b))$. Com isso, dado um intervalo inicial $[a, b]$, $\theta \in (0, 1)$ e $c = \text{secante}(a, b)$ dada por (4.19), aplicamos a estratégia proposta no Algoritmo 7.

Algoritmo 7 Atualização do intervalo

Entrada: $[a, b]$, $\theta \in (0, 1)$ e $c \in \mathbb{R}$;
1: **se** $c \notin (a, b)$ **então**
2: $\bar{a} = a$, $\bar{b} = b$;
3: **senão se** $\phi'(c) \geq 0$ **então**
4: $\bar{a} = a$, $\bar{b} = c$;
5: **senão se** $\phi'(c) < 0$ e $\phi(c) \leq \phi(0) + \epsilon_k$ **então**
6: $\bar{a} = c$, $\bar{b} = b$.
7: **senão se** $\phi'(c) < 0$ e $\phi(c) > \phi(0) + \epsilon_k$ **então**
8: Tome $\hat{a} = a$, $\hat{b} = c$
9: $d = (1 - \theta)\hat{a} + \theta\hat{b}$
10: **se** $\phi'(d) \geq 0$ **então**
11: Tome $\bar{a} = \hat{a}$, $\bar{b} = d$ e finalize;
12: **senão se** $\phi'(d) < 0$ e $\phi(d) \leq \phi(0) + \epsilon_k$ **então**
13: $\hat{a} = d$, vá para a linha 9.
14: **senão se** $\phi'(d) < 0$ e $\phi(d) > \phi(0) + \epsilon_k$ **então**
15: $\bar{b} = d$, vá para a linha 9.
16: **fim se**
17: **fim se**
Saída: $[\bar{a}, \bar{b}]$

Esse algoritmo consiste em considerarmos o comportamento de ϕ' em c, tendo em vista que: se $c \notin (a, b)$ nada podemos afirmar e retornamos o intervalo original $[a, b]$, já para os casos onde $c \in (a, b)$ e satisfaz $\phi'(c) \geq 0$ basta reduzir o lado direto do intervalo

para que a condição de inclinação oposta seja satisfeita. Se $\phi'(c) < 0$ esparece que há um decréscimo quando aumentamos c, mas se $\phi(c) \leq \phi(0) + \epsilon_k$ temos que em c não há um grande decréscimo da função ϕ, logo reduzimos o lado esquerdo do intervalo. Caso em c tenhamos uma derivada negativa, mas agora com $\phi(c) > \phi(0) + \epsilon_k$ teríamos um aumento da função ϕ, não desejado, logo aplica-se uma redução no intervalo, sempre considerando um termo $d \in [\hat{a}, \hat{b}]$. No fim do processo obteremos um intervalo $[\bar{a}, \bar{b}]$ cuja fronteira satisfaz (4.18). Mais informações sobre o Algoritmo 7 é abordado por Hager e Zhang [5].

Conforme explicado por Hager e Zhang [5], se a função ϕ' for convexa ou côncava, a estratégia implementada no Algoritmo 7 atualizará o intervalo $[a, b]$ um lado por vez a cada iteração. Em geral, não sabemos se a função ϕ' é convexa ou côncava, logo deve-se realizar algumas correções.

No Algoritmo 8, tratamos alguns dos casos que podem ocorrer. Se assumirmos que o intervalo dado $[a, b]$ satisfaça as condições de inclinação oposta (4.18), então $c \in [a, b]$. Se $c = B$, então $\phi' \geq 0$ em b e B. Nesse caso, tentamos um passo secante da função ϕ' nos pontos b e B, chamado de $\bar{c}$. Se $\bar{c} \notin [a, b]$ a estratégia do passo secante falha e retorna $[A, B]$. Se $c = A$, então $\phi'(c) < 0$ e $\phi(c) \leq \phi(0) + \epsilon_k$ é satisfeito e ϕ' é negativo em a e A. Neste caso, tentamos um passo secante baseado nos valores de ϕ' em a e A.

Algoritmo 8 Passo secante duplo

Entrada: $[a, b]$, $\theta \in (0, 1)$
1: Tome $c =$ secante(a, b) obtido da Equação (4.19);
2: Obtenha $[A, B]$ da aplicação do Algoritmo 7 com $[a, b]$ e c informados;
3: **se** $c = B$ **então**
4: $\quad \bar{c} =$ secante(b, B) obtido da Equação (4.19);
5: **fim se**
6: **se** $c = A$ **então**
7: $\quad \bar{c} =$ secante(a, A) obtido da Equação (4.19);
8: **fim se**
9: **se** $c = A$ ou $c = B$ **então**
10: $\quad$ Obtenha $[\bar{a}, \bar{b}]$ do Algoritmo 7 dado $[A, B]$ e $\bar{c}$;
11: **senão** $[\bar{a}, \bar{b}] = [A, B]$;
12: **fim se**
Saída: $[\bar{a}, \bar{b}]$

Reforçando que os Algoritmos 7 e 8 são para determinar um intervalo $[\bar{a}, \bar{b}]$ que satisfaçam as condições de inclinação oposta (4.18). Assumindo que a função ϕ é não monótona e um intervalo $[\bar{a}, \bar{b}]$ satisfazendo (4.18) pode ser gerado por avaliações de $\phi(\alpha)$ para várias escolhas de α, a rotina da busca de Hager-Zhang é resumida no Algoritmo 9. No fim do processo obteremos um ponto que satisfaça 1) ou 2).

Algoritmo 9 Busca de Hager-Zhang

Entrada: $[a, b]$, $\theta \in (0, 1)$, $\eta \in (0, 1)$, $\xi \in (\eta, 1)$, $\gamma \in (0, 1)$

Tome $k = 0$, $[a_k, b_k] = [a, b]$
enquanto 1) ou 2) não forem satisfeitas **faça**
    Obtenha $[a, b]$ da aplicação do Algoritmo 8 no intevalo $[a_k, b_k]$;
    se $b - a > \gamma(b_k - a_k)$ **então**
        $c = (a + b)/2$
        Atualize o intervalo $[a, b]$ com o Algoritmo 7 dado $[a, b]$ e c;
    fim se
    $[a_{k+1}, b_{k+1}] = [a, b]$;
    $k = k + 1$;
fim enquanto

4.2.2 O método

O CG-descent é um método para a resolução de problemas da forma (2.1) onde $\Omega = \mathbb{R}^n$, ou seja, é aplicado a problemas de otimização irrestrita. Dado $x_0 \in \mathbb{R}^n$, o método consiste em utilizar o seguinte fórmula de recorrência:

$$x_{k+1} = x_k + \alpha_k d_k \tag{4.20}$$

onde

$$d_{k+1} = -\nabla f(x_{k+1}) + \beta_k^N d_k, \;\; d_0 = -\nabla f(x_0), \tag{4.21}$$

$$\beta_k^N = \frac{1}{d_k^T y_k}\left(y_k - 2d_k \frac{\|y_k\|^2}{d_k^T y_k}\right)^T \nabla f(x_{k+1}), \tag{4.22}$$

em que $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$ e α_k é o tamanho do passo, obtido por uma estratégia de busca linear.

Com o uso das Equações (4.21) e (4.22) é possível mostrar que caso a função objetivo seja quadrática e o passo α_k obtido de maneira exata, o método CG-descent se resumiria ao caso quadrático, discutido na Seção 4.1. De fato, se $\alpha_k = \arg\min\{f(x_k + \alpha d_k);\ \alpha > 0\}$ então $d_k^t \nabla f(x_{k+1}) = 0$ e a Equação (4.22) se resumiria a

$$\beta_k^N = \frac{y_k^t \nabla f(x_{k+1})}{d_k^t y_k} = \frac{\nabla f(x_{k+1})^t [\nabla f(x_{k+1}) - \nabla f(x_k)]}{-d_k^t \nabla f(x_k)},$$

e usando (4.21), obtemos

$$\beta_k^N = \frac{\nabla f(x_{k+1})^t [\nabla f(x_{k+1}) - \nabla f(x_k)]}{\nabla f(x_k)^t \nabla f(x_k)} = \beta_k^{PR}.$$

Segundo Hager e Zhang [5], as expressões (4.21)–(4.22) podem ser obtidas deletando um termo no esquema Perry-Shanno [24, 25]. Além disso, propõem uma melhoria para

o método, limitando o termo β_k^N para garantir a convergência global em problemas com funções não lineares gerais.

Com tal limitação algumas propriedades do método são perdidas, como a velocidade de convergência, principalmente no caso quadrático, mas há uma garantia de que as direções geradas pelo método serão de descida sempre que $d_k^t y_k \neq 0$ (Veja Teorema 1.1, [5]). O ajuste de β_k^N é realizado ajustando o seu limite inferior dinamicamente, para o tornar menor ao longo da convergência:

$$d_{k+1} = -\nabla f(x_{k+1}) + \beta_k^{HZ} d_k, \quad d_0 = -\nabla f(x_0), \tag{4.23}$$

$$\beta_k^{HZ} = \max\{\beta_k^N, \psi_k\}, \quad \psi_k = \frac{-1}{\|d_k\| \min\{\psi, \|\nabla f(x_k)\|\}}, \tag{4.24}$$

onde $\psi > 0$ é uma constante. A condição $d_k^t y_k \neq 0$, pode ser sempre satisfeita quando exigimos as condições de Wolfe (4.12)–(4.13) na busca linear. O método CG-descent é resumido no Algoritmo 10.

Algoritmo 10 Método CG-descent

Entrada: $x_0 \in \mathbb{R}^n$, $\rho \in (0,1)$, $\eta \in (0,1)$, $\xi \in (\eta, 1)$, $\gamma \in (0,1)$, $\theta \in (0,1)$ e uma tolerância $\epsilon > 0$;

1: Tome $k = 0$, $d_0 = -\nabla f(x_0)$;
2: **enquanto** $\|\nabla f(x_k)\| > \epsilon$ **faça**
3: Determine um tamanho de passo $\alpha_k \in (0, 1]$ que satisfaça as condições de Wolfe, como proposto no Algoritmo 9;
4: $x_{k+1} = x_k + \alpha_k d_k$;
5: Tome $\beta_k^{HZ} = \max\{\beta_k^N, \psi_k\}$ utilizando as Equações (4.22) e (4.24)
6: Atualize $d_{k+1} = -\nabla f(x_{k+1}) + \beta_k^{HZ} d_k$,
7: $k \leftarrow k + 1$;
8: **fim enquanto**

Saída: x_k.

A convergência do método CG-descent não será abordada nesse trabalho, mas pode ser encontrada nas referências do capítulo, veja [5].

4.3 Testes numéricos

Na presente seção, faremos uma comparação do comportamento numérico dos métodos apresentados nos dois capítulos anteriores. Todos eles foram implementados na linguagem Julia versão 1.7.0 (30-11-2021), veja [26], e testados no servidor do grupo de pesquisa do projeto FAPES 116/2019, que possui as seguintes especificações: sistema GNU/Linux Ubuntu 20.04.2 LTS, 1 processador Intel(R) Xeon(R) Silver 4114 CPU 2.20 GHz 10

núcleos (20 threads), 160 Gb RAM.

Para a comparação, foram considerados os problemas irrestritos e com restrições de caixa da biblioteca CUTEst [9]. Foram utilizados os perfis de desempenho propostos pela Dolan e Moré [8], gerados com o pacote `BenchmarkProfiles.jl` (`https://github.com/JuliaSmoothOptimizers/BenchmarkProfiles.jl`) na linguagem Julia. Em tais perfis é feito a montagem de um gráfico, para cada método, da fração de problemas resolvidos pelo método dentro de um fator relativo ao melhor algoritmo, podendo ser o tempo, número de iterações, avaliações de função ou gradiente. Esses perfis são muito utilizados na comparação de algoritmos, em especial, de otimização contínua.

Para um melhor entendimento dos perfis de desempenho, suponha que queiramos comparar o desempenho de um conjunto S de n_s algoritmos aplicados a um conjunto P de n_p problemas testes. Fixaremos o tempo de CPU como medida; para outras medidas de desempenho, é análogo. Considere $t_{p,s}$ como o tempo necessário para o algoritmo $s \in S$ resolver o problema $p \in P$. Se o algoritmo s não resolveu o problema p tomaremos $t_{p,s} = \infty$. Assim, definimos o índice de desempenho do algoritmo s no problema p como

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,j} | j \in S\}} \geq 1.$$

Note que $r_{p,s} = \tau$ significa que s levou τ vezes o tempo do algoritmo mais rápido em p. Por exemplo, $r_{p,s} = 3$ significa que s levou o triplo do tempo do algoritmo mais rápido em p, e $r_{p,s} = 1$ diz que o algoritmo s foi o mais rápido ao resolver o problema p. Vale ressaltar que nos casos onde $t_{p,s} = \infty$ tomaremos $r_{p,s} = \infty$ sempre.

Daí, para cada algoritmo s consideramos a função desempenho $\rho_s : [1, \infty) \to [0, 1]$ definida por

$$\rho_s(\tau) = \frac{1}{n_p} \text{card}\{p \in P | r_{p,s} \leq \tau\},$$

onde *card* representa a cardinalidade ou número de elementos do conjunto. Assim, $\rho_s(1)$ é a proporção de problemas onde o algoritmo s resolve no menor tempo. De maneira mais geral, considerando uma medida de desempenho qualquer, $\rho_s(\tau)$ é a porcentagem de problemas que o algoritmo s resolveu em τ vezes o valor da medida de desempenho do algoritmo mais eficiente.

A maneira mais fácil para visualizarmos esse comparativo é traçando um gráfico de $\rho_s(\tau)$, para vários algoritmos. Tal estratégia é utilizada nos testes numéricos posteriores, mas neles consideraremos a escala logarítmica de base 2 (como proposto em [8]) no eixo horizontal para melhor visualização. Além do artigo que propôs os perfis de desempenho [8], o leitor pode consultar a seção 6.3 de [11] para detalhes.

4.3.1 Problemas da coletânea CUTEst

A coletânea CUTEst [9] (do inglês *Constrained and Unconstrained Testing Environment with safe threads*) reúne mais de 1300 problemas, entre lineares e não lineares, restritos e irrestritos. É considerada na literatura a biblioteca padrão para testes de algoritmos de otimização contínua. Nesta seção utilizaremos problemas irrestritos e com restrições de caixa. Declararemos convergência dos métodos quando $\|\nabla f(x_k)\|_\infty < 10^{-7}$ (ou, $\|d_k\| < 10^{-7}$ para os problemas com restrições de caixa) e consideraremos um número de iterações de no máximo 5×10^4.

4.3.1.1 Problemas irrestritos

Nos testes numéricos para problemas irrestritos, foram inicialmente considerados problemas da ordem de 50 a 1000 variáveis da CUTEst. Tais problemas foram carregados no Julia a partir do pacote `CUTEst.jl` (`https://github.com/JuliaSmoothOptimizers/CUTEst.jl`), em uma estrutura onde o cálculo de derivadas e hessianas são automatizados. Para todos os métodos, o ponto inicial foi obtido do próprio arquivo do problema, e caso não o tenha, tomamos como a origem $x_0 = (0, 0, \ldots, 0)^t$.

Na implementação do método do gradiente, usou-se o Algoritmo 2 considerando o tamanho de passo α_k obtido por busca inexata com o uso *backtracking* e interpolação quadrática (Algoritmo 1), com os parâmetros

$$\eta = 10^{-4},\ \rho = (0,5),\ \sigma_1 = 0,1 \ \text{ e } \ \sigma_2 = 0,9.$$

Para o método do Gradiente Espectral Projetado, implementamos, na linguagem Julia, uma tradução da implementação em FORTRAN fornecida pelo projeto TANGO (`https://www.ime.usp.br/~egbirgin/tango/`), cujo esquema básico é apresentado no Algoritmo 5 com $\Omega = \mathbb{R}^n$. Essa implementação traz a busca não monótona do Algoritmo 4, com $M = 100$ e $\eta = 10^{-4}$. Em outras aplicações é comum utilizarmos valores de M entre 2 e 100, mas o melhor valor possível depende do problema. O que pode-se afirmar é que $M = 1$ ou M é o número máximo de iterações, não são boas escolhas, pois a primeira recai na busca inexata clássica (monótona) e o último não nos fornece um controle na diminuição da função objetivo ao longo das iterações. Os outros parâmetros para o SPG foram escolhidos como o padrão na implementação em FORTRAN,

$$\lambda_{min} = 10^{-30},\ \ \lambda_{max} = 10^{30}.$$

O parâmetro $\lambda_0 \in [\lambda_{min}, \lambda_{max}]$ é arbitrário, e pode ser inicializado como afirma Birgin et al [3]:

$$\lambda_0 = \max\left\{\lambda_{min}, \min\left\{\frac{1}{\|P_\Omega(x_0 - \nabla f(x_0)) - x_0\|_\infty}, \lambda_{max}\right\}\right\},$$

assumindo que x_0 é tal que $P_\Omega(x_0 - \nabla f(x_0)) \neq x_0$.

Considerou-se também a adaptação do método de direções conjugadas para não quadráticas CG-descent vista anteriormente. Para o último, a busca linear utilizada foi a busca de Hager-Zhang, cujo esquema básico é apresentado no Algoritmo 9, proposta no artigo referência, implementada a partir do uso do pacote `LineSearches.jl` (`https://github.com/JuliaNLSolvers/LineSearches.jl`) e o método obtido do pacote `Optim.jl` (`https://github.com/JuliaNLSolvers/Optim.jl`), com os parâmetros sugeridos por Hager e Zhang [5].

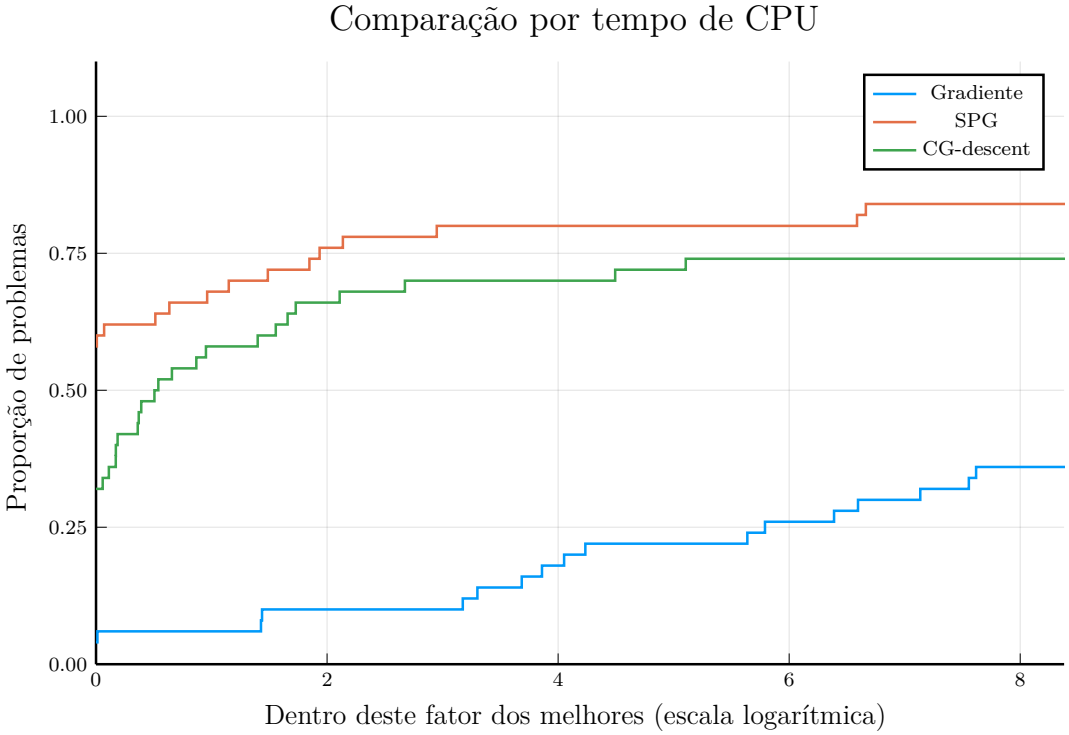


Figura 13: Perfil de desempenho dos métodos do Gradiente, SG e CG-descent com relação ao tempo computacional para problemas irrestritos.

Nas Figuras 13 e 14 apresentamos os perfis de desempenho com relação ao número de iterações e tempo computacional dos métodos do Gradiente, SPG e CG-descent. Podemos observar na Figura 13 que o SPG foi o método mais eficaz e veloz, resolvendo mais de 80% dos problemas, e sendo o mais rápido em $\approx 60\%$ deles. O CG-descent ficou como o segundo melhor, sendo pior que o SPG em eficácia e velocidade, mas superando em muito o clássico método do gradiente, que teve o pior desempenho no conjunto de problemas selecionados.

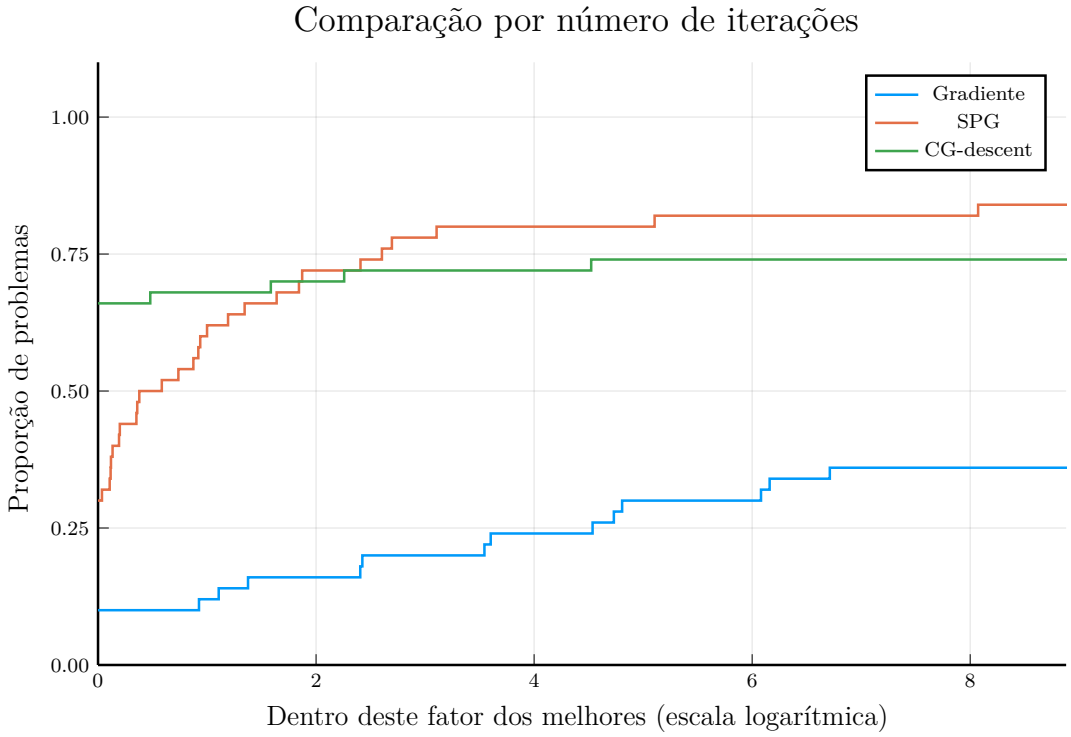


Figura 14: Perfil de desempenho dos métodos Gradiente, SG e CG-descent com relação ao número de iterações para problemas irrestritos.

Na Figura 14, podemos extrair algumas informações importantes: note que o método CG-descent resolve a maior proporção de problemas com poucas iterações, mas perde para o método SPG em relação ao tempo. Logo, concluímos que o CG-descent tem um custo maior por iteração, assim levanto um maior tempo de CPU para encontrar a solução; outro ponto importante são as iterações do SPG, muito baratas computacionalmente. Ou seja, mesmo levando mais iterações para solucionar o problema, acaba por ser um método muito eficiente.

Os resultados dos testes numéricos, utilizados para a análise e construção dos perfis de desempenho, estão presentes no Apêndice A, Tabela 5.

4.3.1.2 Problemas com restrições de caixa

Agora apresentaremos os testes numéricos voltados a problemas com restrições de caixa. Nesses testes aplicaremos o método do gradiente projetado e gradiente espectral projetado, visto que o método CG-descent não é adequado a tais restrições.

Consideramos problemas da ordem de 50 a 1000 variáveis, onde aplicamos o método GP com o uso do Algoritmo 3 e do SPG com o esquema básico do Algoritmo 5. Os parâmetros utilizados foram os mesmos da seção anterior, mas agora com a caixa $\Omega =$

$\{x \in \mathbb{R}^n | u \leq x \leq l\}$.

Na Figura 15 apresentamos o perfil de desempenho com relação ao tempo computacional. Note que, o SPG teve uma eficácia muito superior ao do gradiente projetado, resolvendo cerca de $\approx 70\%$ dos problemas e sendo o mais veloz em todos eles. Já o método do gradiente projetado teve uma eficácia de poucos 6%, o que era de se esperar visto que nos testes para problemas irrestritos houve uma grande diferença entre os dois métodos.

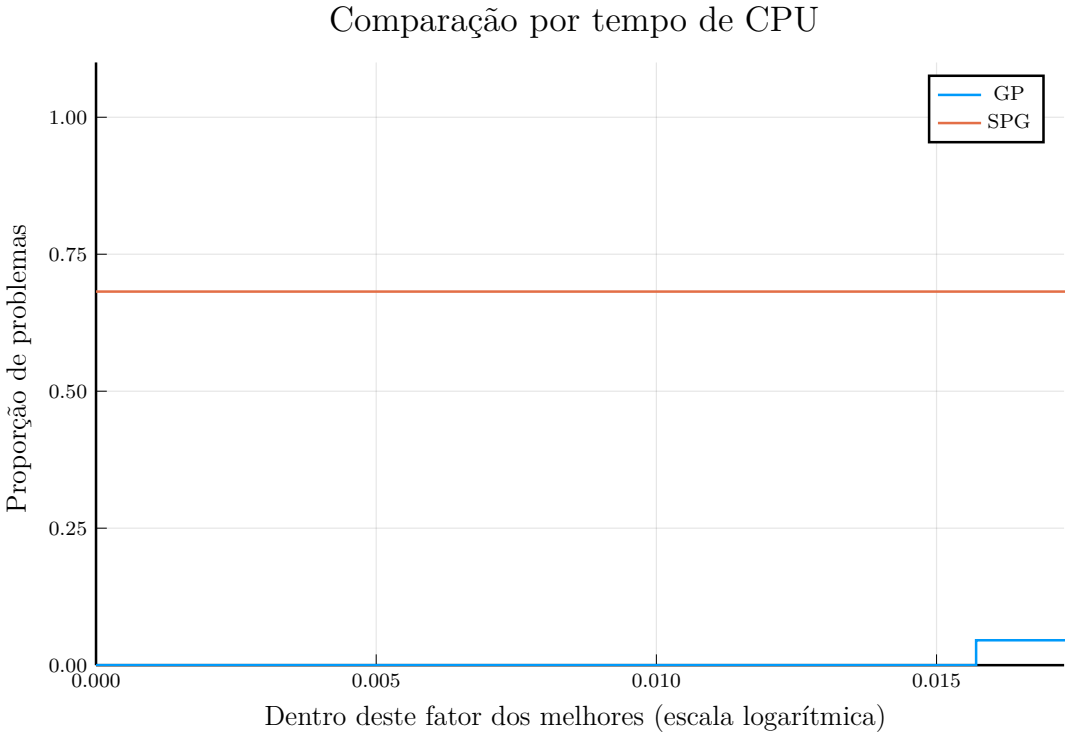


Figura 15: Perfil de desempenho com relação ao tempo computacional dos métodos voltados para problemas com restrições de caixa.

Na Tabela 1 evidenciamos um recorte dos dados obtidos com a aplicação dos métodos para restrições de caixa. Nela, st é a situação de saída (0 – resolvido, 1 – caso contrário), n o número de variáveis, f e $\|d\|_\infty$ são, respectivamente, a aproximação do valor ótimo e a norma do máximo da direção projetada no último iterando, que como vimos anteriormente é uma condição necessária de otimalidade para problemas com restrições convexas.

Repare que em todos os problemas o método do gradiente projetado não conseguiu resolver efetivamente os problemas propostos, não alcançando a tolerância exigida, mas em alguns dos casos se aproximando lentamente da solução. Por outro lado, o método do gradiente espectral projetado os resolveu com um pequeno número de iterações em tempo de CPU muito pequeno, em especial para problemas com um grande número de

variáveis. Isso se deve ao SPG empregar uma ideia quase-Newton, conforme discutido na Seção 3.4.3 , e que o método do gradiente dá passos pequenos próximo da solução, pois limita os passos a 1.

Tabela 1: Recorte dos resultados numéricos obtidos da aplicação dos métodos do Gradiente Projetado e do Gradiente Espectral Projetado.

Problema	n	Método	It	f	$\|d\|_\infty$	st	Tempo(s)
GENROSEB	500	GP	50000	1,59E+03	3,04E+01	1	1,83E+01
		SPG	142	1,59E+03	3,55E-15	0	3,83E-02
DIAGIQT	1000	GP	50000	-3,46E+14	5,00E+07	1	6,84E+01
		SPG	1378	-3,46E+14	9,88E-08	0	3,79E-01
DIAGNQB	1000	GP	50000	-3,33E+15	1,00E+08	1	9,19E+00
		SPG	2	-3,33E+15	0,00E+00	0	5,64E-04
DIAGPQE	1000	GP	50000	-3,74E+00	9,78E-07	1	5,22E+01
		SPG	451	-3,74E+00	7,73E-08	0	1,26E-01
HADAMALS	400	GP	50000	7,17E+03	3,83E+04	1	1,38E+02
		SPG	481	7,16E+03	5,97E-08	0	0.357152
DIAGNQT	1000	GP	50000	-1,67E+15	1,00E+08	1	9,13E+00
		SPG	2	-1,67E+15	0,00E+00	0	5,61E-04
HARKERP2	1000	GP	50000	-5,00E-01	1,00E+00	1	2,21E+05
		SPG	63	-5,00E-01	0,00E+00	0	4,04E-01
HOLMES	180	GP	50000	1,25E+03	6,98E+00	1	1,18E+03
		SPG	88	1,25E+03	6,45E-09	0	4,10E-01
SINEALI	1000	GP	50000	-9,98E+04	8,69E+05	1	4,00E+01
		SPG	119	-7,90E+04	2,25E-08	0	5,35E-02
DIAGIQB	1000	GP	50000	-1,18E+15	5,00E+07	1	2,10E+01
		SPG	422	-1,18E+15	6,41E-08	0	1,15E-01
DIAGNQE	1000	GP	50000	-2,50E+15	1,00E+08	1	9,14E+00
		SPG	2	-2,50E+15	0,00E+00	0	5,62E-04

Os resultados completos dos testes numéricos estão no Apêndice A, Tabela 6.

4.3.2 Um problema "quase-quadrático"

Nesta seção estamos interessados em resolver problemas restritos da forma

$$\text{Minimizar } f(x) = \frac{1}{2}x^t Ax + \frac{\beta}{4}\sum_{i=1}^{n} x_i^4 \text{ sujeito a } x \in S,$$

onde $S = \{x \in \mathbb{R}^n | \|x\|_2 = 1\}$, a matriz A é hermitiana de ordem n e $\beta > 0$ é o parâmetro de regularização. Este problema está relacionado com a discretização da função energia no condensado de Bose-Einstein [27]. Nesse problema, os elementos a_{ij} da matriz A são complexos, mas para prosseguir assumiremos que A seja uma matriz definida positiva

com coeficientes reais. Para transformarmos esse problema em um modelo irrestrito, penalizamos a restrição $x^t x - 1 = 0$, nos levando à função

$$f(x) = \frac{1}{2}x^t Ax + \frac{\beta}{4}\sum_{i=1}^{n} x_i^4 + \frac{\rho}{2}(x^t x - 1)^2,$$

onde ρ é o parâmetro penalizador. A hessiana de f é dada por

$$\nabla^2 f(x) = A + 3\beta \text{diag}(x_i^2) + 4\rho x x^t + 2\rho(x^t x - 1)I. \tag{4.25}$$

Note que, para todo $d \in \mathbb{R}^n/\{0\}$ temos que

$$\begin{aligned} d^t \nabla^2 f(x) d &= d^t A d + 3\beta\ d^t \text{diag}(x_i^2) d + 4\rho\ d^t (x x^t) d + 2\rho(x^t x - 1)\ d^t I d \\ &\geq \lambda_{\min}(A) d^t d + 3\beta \min_i x_i^2 d^t d + 2\rho(x^t x - 1) d^t d. \end{aligned}$$

Daí, $d^t \nabla^2 f(x) d \geq \sigma(x) d^t d$ onde

$$\sigma(x) = \lambda_{\min}(A) + 3\beta \min_i x_i^2 + 2\rho(x^t x - 1).$$

Com isso, podemos observar que quando $\|x\|_2 \geq 1$ temos que o menor autovalor de $\nabla^2 f(x)$ é maior que ou igual a $\lambda_{\min}(A)$, ou seja, a hessiana é definida positiva. Mas não há garantia de que a hessiana permanecerá definida positiva para $\|x\|_2 < 1$ (lembrando que podemos caracterizar a positividade de uma matriz em função dos seus autovalores).

Este problema foi considerado em [28] na comparação entre vários algoritmos tipo gradiente (entre eles, o gradiente espectral), o CG-descent e um novo método tipo gradientes conjugados, não abordado neste trabalho. Os testes, a construção das instâncias e a discussão aqui apresentados seguem basicamente esta referência.

Para os testes numéricos, consideramos $\rho = 2 \times 10^5$ e $\beta = 500$. As matrizes foram obtidas da coletânea Suite Sparse Matrix Collection [10], mantidas pela Universidade da Flórida. Foram selecionadas as matrizes simétricas e definidas positivas do grupo HB com $n \leq 10000$. O ponto inicial foi tomado como $1,1 \times v/\|v\|$, onde v é um autovetor associado ao menor autovalor, calculado pelo método de Lanczos implementado no pacote `Arpack.jl` (`https://github.com/JuliaLinearAlgebra/Arpack.jl`). Tal escolha se deve ao fato de que quanto $\beta = 0$ o problema se resume a um problema de menor autovalor. Ou seja, desejamos que o método inicie próximo à solução, mas não temos garantia que isso ocorrerá devido à presença dos termos quárticos x_i^4 na função objetivo.

Para o experimento consideramos como critério de parada menos exigente

$$\|\nabla f(x_k)\|_\infty \leq 10^{-4}$$

devido à dificuldade numérica em lidar com os termos quárticos. Acrescentamos também um número de iterações de no máximo 10^5, os demais parâmetros foram os mesmos utilizados para os problemas irrestritos. O método do gradiente foi removido da análise, pois apresentou resultados insatisfatórios em testes numéricos anteriores.

De um total de 56 problemas, 12 foram removidos da análise pois nenhum dos métodos os resolveu e 1 deu erro ao calcular o ponto inicial, relacionado ao pacote `Arpack`. A remoção de tais problemas não interfere na análise, visto que nenhum dos métodos foi capaz de os resolver. Os resultados dos testes numéricos podem ser encontrados no Apêndice A, Tabela 7.

Nas Figuras 16 e 17 apresentamos os perfis de desempenho com relação ao tempo computacional e número de avaliações de gradiente, respectivamente, nos problemas cujo menor autovalor $\lambda_{\min}(A) \geq 1$. Olhando para a Figura 16, podemos observar que nas instâncias selecionadas o método CG-descent teve uma eficácia superior ao SPG, resolvendo todos os problemas propostos e sendo o mais veloz em cerca de 40% deles. Assim, observamos que o CG-descent é um pouco mais lento que o SPG, que resolveu $\approx 55\%$ dos problemas no menor tempo, mas se esperar $2^{1,3} \approx 2,45$ vezes o tempo do algoritmo mais rápido o CG-descent se torna o mais eficaz, enquanto a do SPG se estagna.

Na Figura 17, vemos um comportamento semelhante ao perfil anterior: como CG-descent resolveu todos os problemas propostos e destes, cerca de 40% foram solucionados com poucas avaliações de gradiente sendo um pouco a menos que o método SPG, que resolveu cerca de 55% dos problemas e todos com o menor número de avaliações de gradiente.

Agora, olharemos para os perfis desempenhos com relação ao tempo de CPU e número de avaliações de gradiente, em problemas onde $\lambda_{\min}(A) < 1$. Tais perfis podem ser observados nas Figuras 18 e 19 , respectivamente. Repare que nesses casos o SPG foi o melhor em velocidade, resolvendo cerca de 96% dos problemas no menor tempo, e possuindo uma eficácia comparável ao CG-descent, que resolveu todos os problemas, mas para isso precisamos esperar cerca de $113,96 (\approx 2^{6,7})\times$ o tempo do algoritmo mais rápido. Ou seja, mesmo que a eficácia do CG-descent seja maior, no balanço entre tempo e eficácia SPG foi melhor.

Na Figura 19, vemos um comportamento como o da Figura 18: SPG manteve sua eficácia, realizando menos avaliações de gradiente em $\approx 94\%$ dos problemas, superando o CG-descent que resolveu cerca de 10% com o menor número de avaliações de gradiente.

Os resultados obtidos para os diferentes valores de $\lambda_{\min}(A)$ são coerentes visto que

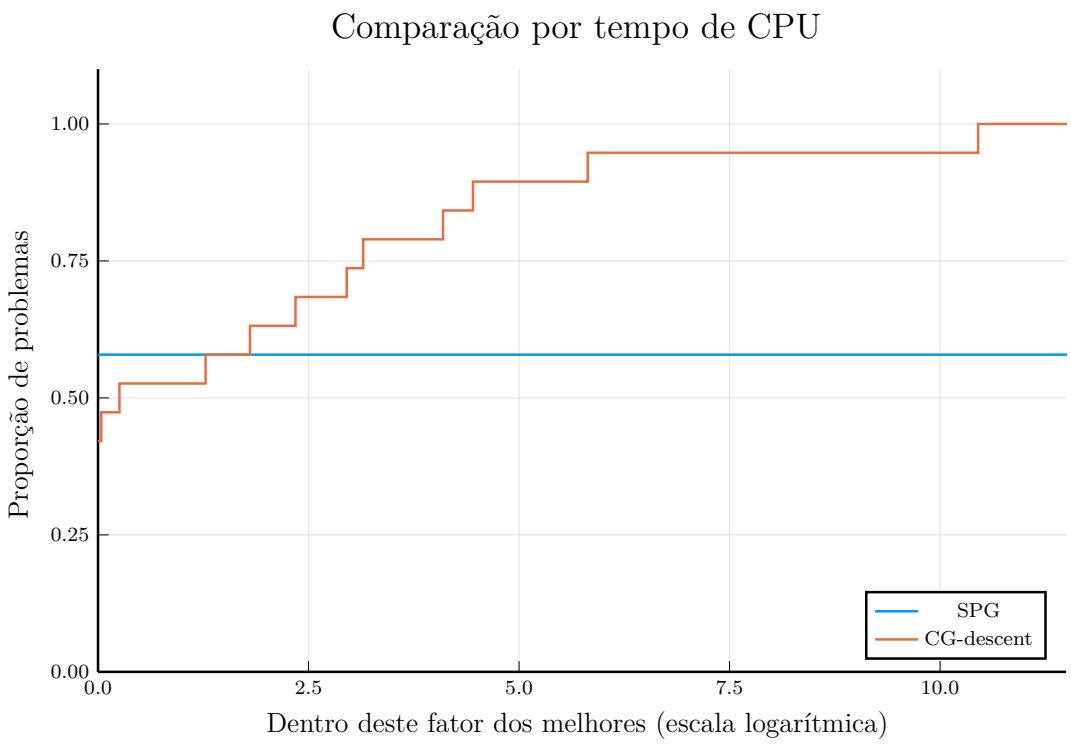


Figura 16: Perfil de desempenho com relação ao tempo computacional dos resultados obtidos da aplicação dos métodos do CG-descent e SPG considerando os problemas em que $\lambda_{\min}(A) \geq 1$.

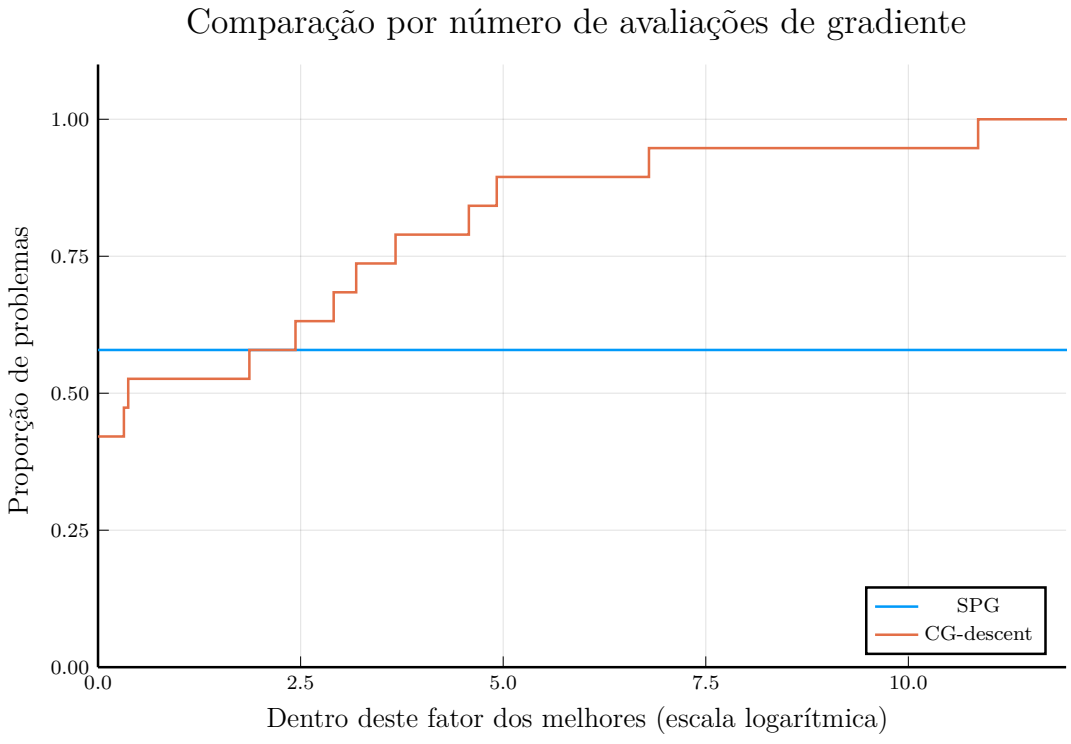


Figura 17: Perfil de desempenho com relação ao número de avaliações de gradiente dos resultados obtidos da aplicação dos métodos do CG-descent e SPG considerando os problemas em que $\lambda_{\min}(A) \geq 1$.

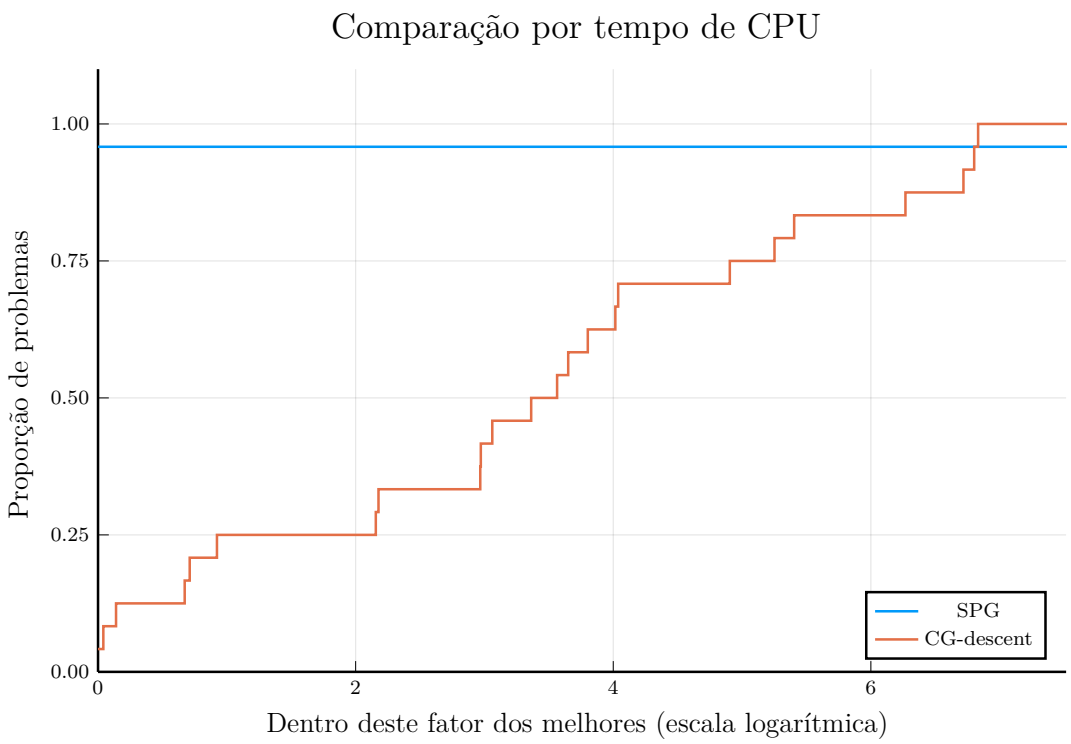


Figura 18: Perfil de desempenho com relação ao tempo computacional dos resultados obtidos da aplicação dos métodos do CG-descent e SPG considerando os problemas em que $\lambda_{\min}(A) < 1$.

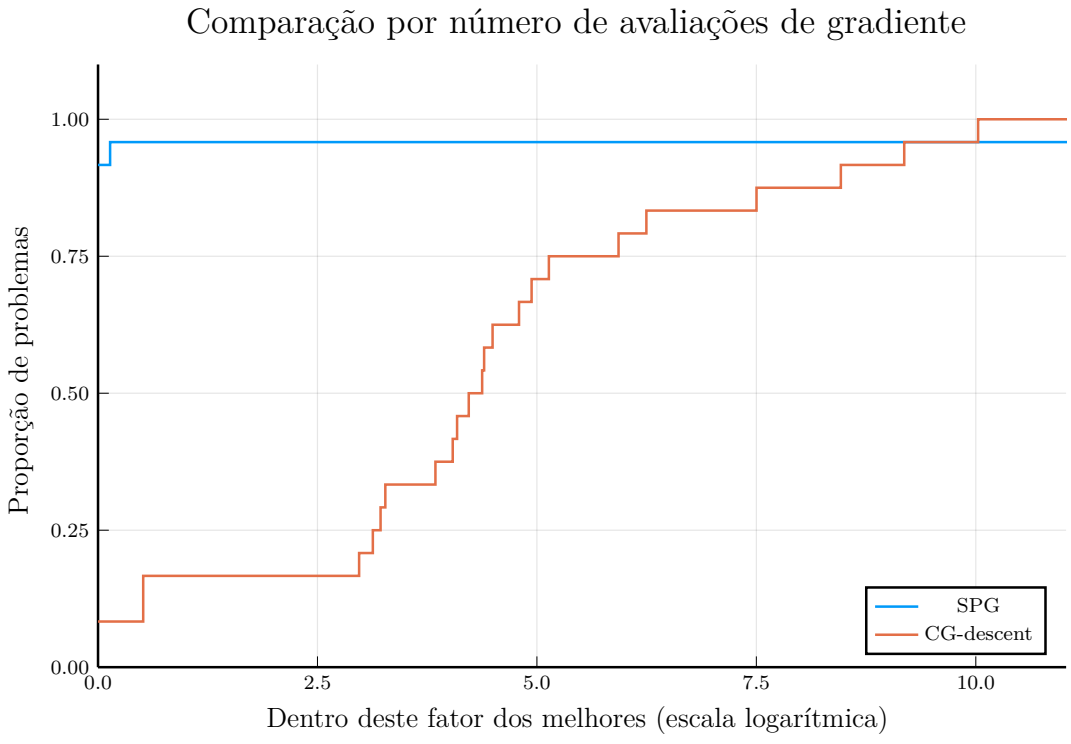


Figura 19: Perfil de desempenho com relação ao número de avaliações de gradiente dos resultados obtidos da aplicação dos métodos do CG-descent e SPG, considerando os problemas em que $\lambda_{\min}(A) < 1$.

CG-descent é uma adaptação do método de direções conjugadas para não quadráticas, ou seja, espera-se que ele seja muito eficiente para funções com um comportamento parecido com quadráticas. De fato, veja que se $\lambda_{\min}(A)$ é grande ($\lambda_{\min}(A) \geq 1$), o termo $\frac{\beta}{4}\sum_{i=1}^{n} x_i^4 + \frac{\rho}{2}(x^t x - 1)^2$ tem menor relevância no valor da função, e logo esperamos que $f(x)$ seja parecida com uma quadrática. Diante dos testes numéricos, isso de fato ocorre: vemos uma grande diferença na eficácia e velocidade dos dois métodos quando são separados com relação a $\lambda_{\min}(A)$.

5 Um método quase-Newton com memória limitada

Os métodos quase-Newton com memória limitada são muito utilizados para a resolução de problemas de grande porte, visto que, em muitos dos casos, o cálculo da hessiana possui um alto custo computacional e impossível de ser armazenada completamente. Tais métodos usam aproximações simples e compactas das matrizes hessianas, ou seja, em vez de armazenar toda a hessiana de ordem $n \times n$, salva-se apenas alguns vetores com comprimento n, que representam as aproximações implicitamente.

Diversos métodos foram desenvolvidos com essa estratégia. Focamos no algoritmo conhecido como BFGS com memória limitada ou, resumidamente, L-BFGS (do inglês *Limited-memory BFGS*) descrito por Liu e Nocedal [29]. Ele é baseado na fórmula de atualização das aproximações das inversas das hessianas do método BFGS. No L-BFGS a ideia principal é usar informações de curvatura de iterações recentes para aproximar a hessiana. As informações de iterações antigas são descartadas de modo a economizar memória e por esperarmos que a hessiana no ponto corrente não guarde relação com as hessianas em pontos distantes. Ao longo desse capítulo foram utilizadas as referências [1, 7, 11, 30, 31].

5.1 O método L-BFGS

Outro método quase-Newton com boas propriedades teóricas e amplamente utilizado em problemas de grande porte é o método BFGS, proposto por Broyden, Fletcher, Goldfarb e Shanno [21]. O BFGS consiste em utilizarmos o esquema (3.30) e gerarmos a matriz B_{k+1} que satisfaça a equação secante (3.29), a partir de B_k e correções de posto 2.

De modo a determinar B_{k+1} com essas características, considere $a, b \in \mathbb{R}$ e $u, v \in \mathbb{R}^n$, logo a matriz B_{k+1} é da forma

$$B_{k+1} = B_k + auu^t + bvv^t$$

e exigimos que $B_{k+1}s_k = y_k$. Daí segue-se que

$$(B_k + auu^t + bvv^t)s_k = y_k.$$

Ou seja,

$$au(u^t s_k) + bv(v^t s_k) = y_k - B_k s_k.$$

Uma possível escolha para satisfazer essa condição é

$$au(u^t s_k) = y_k \text{ e } bv(v^t s_k) = -B_k s_k.$$

Multiplicando a esquerda por s_k^t em ambos os lados, obtemos $a(u^t s_k)^2 = s_k^t y_k$ e $b(v^t s_k)^2 = -s_k^t B_k s_k$. Assim, podemos considerar $a = 1$ e $b = -1$, logo

$$u = \frac{y_k}{u^t s_k} = \frac{y_k}{\sqrt{s_k^t y_k}} \text{ e } v = \frac{B_k s_k}{v^t s_k} = \frac{B_k s_k}{\sqrt{s_k^t B_k s_k}}.$$

Logo, temos a seguinte fórmula secante:

$$B_{k+1} = B_k + \frac{y_k y_k^t}{s_k^t y_k} - \frac{B_k s_k s_k^t B_k}{s_k^t B_k s_k}. \tag{5.1}$$

A escolha (5.1) é conhecida como fórmula BFGS e possui a propriedade de gerar matrizes definidas positivas e simétricas, mantendo as direções resultantes de descida. Tal propriedade é apresentada no teorema a seguir.

Teorema 5.1. *Na fórmula BFGS, se B_k é simétrica, definida positiva e*

$$s_k^t y_k > 0, \tag{5.2}$$

então B_{k+1} também é simétrica e definida positiva.

Demonstração. A simetria de B_{k+1} é direta. Seja $d \in \mathbb{R}^n/\{0\}$. De (5.1) temos que

$$d^t B_{k+1} d = d^t B_k d + \frac{(d^t y_k)^2}{y_k^t s_k} - \frac{(d^t B_k s_k)^2}{s_k^t B_k s_k},$$

onde $d^t B_k d > 0$ e $(d^t y_k)^2/(y_k^t s_k) \geq 0$ por hipótese. Agora, tome

$$\begin{aligned} c &= d^t B_k d - \frac{(d^t B_k s_k)^2}{s_k^t B_k s_k} \\ &= \frac{(s_k^t B_k s_k)(d^t B_k d) - (d^t B_k s_k)^2}{s_k^t B_k s_k}. \end{aligned}$$

Como B_k é simétrica e definida positiva, B_k possui uma decomposição Cholesky única,

ou seja,

$$B_k = LL^t,$$

em que L é uma matriz triangular inferior. Assim,

$$\begin{aligned} s_k^t B_k s_k &= (s_k^t L)(L^t s) = \|L^t s_k\|^2, \\ d^t B_k d &= (d^t L)(L^t d) = \|L^t d\|^2, \\ d^t B_k s_k &= (d^t L)(L^t s_k) = (L^t d)^t (L^t s_k). \end{aligned}$$

Da desigualdade de Cauchy-Schawarz, obtemos

$$(L^t d)^t (L^t s_k) \leq \|L^t d\|^2 \|L^t s_k\|^2.$$

Ou seja,

$$(d^t B_k d)(s_k^t B_k s_k) - (d^t B_k s_k)^2 \geq 0.$$

Portanto, $c \geq 0$. Na verdade, $c = 0$ apenas quando d é múltiplo de s_k, mas neste caso $d^t y_k \neq 0$ e assim $(d^t y_k)^2/(s_k^t y_k) > 0$. Logo, $d^t B_{k+1} d > 0$. □

Quando a função f é convexa, a condição (5.2) é automaticamente satisfeita para quaisquer dois pontos x_k e x_{k+1}. Com isso, as matrizes geradas por (5.1) são definidas positivas pelo Teorema 5.1. No entanto, esta condição não é sempre verdadeira para funções não convexas. Nesses casos, forçaremos (5.2) explicitamente impondo condições sobre a busca linear, utilizada para determinar o tamanho do passo α. De fato, a condição (5.2) é satisfeita se impomos as condições de Wolfe (4.12)–(4.13) na busca linear. Para verificarmos, basta notar que $s_k = x_{k+1} - x_k = \alpha_k d_k$ e por (4.13) obtemos $\nabla f(x_{k+1})^t s_k \geq \xi \nabla f(x_k)^t s_k$, e, portanto

$$\begin{aligned} y_k^t s_k &= \nabla f(x_{k+1})^t s_k - \nabla f(x_k)^t s_k \\ &\geq \xi \nabla f(x_k)^t s_k - \nabla f(x_k)^t s_k \\ &= (\xi - 1) \nabla f(x_k)^t s_k \end{aligned}$$

Assim,

$$y_k^t s_k \geq (\xi - 1) \alpha_k \nabla f(x_k)^t d_k \tag{5.3}$$

Como $\xi < 1$ e d_k é uma direção de descida, o termo à direita de (5.3) é positivo a condição (5.2) é válida.

Uma implementação ingênua dessa variante não é eficiente para problemas de otimização irrestrita com muitas variáveis, visto que ainda teríamos que resolver um

sistema linear. Portanto, geralmente formulamos os métodos quase-Newton na forma

$$x_{k+1} = x_k - \alpha_k H_k \nabla f(x_k)$$

onde a matriz H_k corresponde a B_k^{-1} e α_k o tamanho do passo. Dessa maneira, H_k pode ser interpretada como aproximações das inversas das Hessianas. Se aplicarmos a fórmula de Sherman-Morrison em (5.1) e efetuarmos algumas manipulações algébricas, teremos

$$H_{k+1} = W_k^t H_k W_k + \rho_k s_k s_k^t, \tag{5.4}$$

com

$$\rho_k = \frac{1}{y_k^t s_k}, \quad W_k = I - \rho_k y_k s_k^t \tag{5.5}$$

e

$$s_k = x_{k+1} - x_k = \alpha_k d_k, \quad y_k = \nabla f(x_{k+1}) - \nabla f(x_k). \tag{5.6}$$

Pode ocorrer da matriz H_k ser densa e o custo de seu armazenamento e manipulação ser alto. Para contornar isso, armazenaremos uma versão modificada/aproximada de H_k definida de maneira implícita, realizando o armazenamento de um certo número m de pares de vetores $\{s_i, y_i\}$ utilizadas nas fórmulas (5.4), (5.5) e (5.6). É basicamente nisso que consiste o método L-BFGS.

No método L-BFGS a operação $H_k \nabla f(x_k)$ é obtida a partir da execução de uma sequência de produtos e somas de vetores envolvendo $\nabla f(x_k)$ e os pares $\{s_k, y_k\}$. Após calcularmos a iteração seguinte, o par de vetores mais antigo do conjunto $\{s_i, y_i\}$ é substituído pelo par atual $\{s_k, y_k\}$ obtido de (5.6). Com isso, o conjunto de pares de vetores $\{s_i, y_i\}$ inclui as informações de curvatura das m mais recentes iterações. Em muitos casos, resultados numéricos satisfatórios foram obtidos tomando valores pequenos de m (digamos, entre 3 e 20 [7]).

Agora descreveremos esse processo com mais detalhes. Considere as seguintes informações da iteração k: o iterando atual x_k e o conjunto de pares de vetores $\{s_i, y_i\}$ com $i = k - m, \ldots, k - 1$. Primeiramente, precisaremos de uma aproximação inicial H_k^0 para a inversa da hessiana em x_k, podendo esta variar a cada iteração, como será explicado posteriormente.

Com o uso da estratégia de descarte, analisemos como será o comportamento das matrizes H_k geradas de maneira recursiva pela fórmula (5.4). Para ilustrar o funcionamento das atualizações, suponha que o número de correções armazenadas m seja igual a 2 e

$H_k^0 = H_0$, $\forall k \in \mathbb{N}$. Daí,

$$\begin{aligned} H_1 &= W_0^t H_0 W_0 + \rho_0 s_0 s_0^t, \\ H_2 &= W_1^t H_1 W_1 + \rho_1 s_1 s_1^t \\ &= W_1^t (W_0^t H_0 W_0 + \rho_0 s_0 s_0^t) W_1 + \rho_1 s_1 s_1^t \\ &= (W_1^t W_0^t) H_0 (W_0 W_1) + W_1^t \rho_0 s_0 s_0^t W_1 + \rho_1 s_1 s_1^t. \end{aligned} \tag{5.7}$$

Como $m = 2$, realizamos o descarte da informação mais antiga de (5.7), temos que

$$\bar{H}_2 = W_1^t H_0 W_1 + \rho_1 s_1 s_1^t,$$

e aplicando a atualização, obtemos

$$H_3 = W_2^t W_1^t H_0 W_1 W_2 + W_2^t \rho_1 s_1 s_1^t W_2 + \rho_2 s_2 s_2^t.$$

Similarmente, geramos

$$H_4 = W_3^t W_2^t H_0 W_2 W_3 + W_3^t \rho_2 s_2 s_2^t W_3 + \rho_3 s_3 s_3^t.$$

Assim, de maneira mais geral, para $k \leq m-1$ e H_k^0 qualquer conseguimos a seguinte fórmula de atualização:

$$\begin{aligned} H_k &= (W_{k-1}^t W_{k-2}^t \cdots W_0^t) H_k^0 (W_{k-1} W_{k-2} \cdots W_0) \\ &+ \rho_0 (W_{k-1}^t \cdots W_1^t) s_0 s_0^t (W_1 \cdots W_{k-1}) \\ &+ \cdots \\ &+ \rho_{k-2} (W_{k-1}^t) s_{k-2} s_{k-2}^t (W_{k-1}) \\ &+ \rho_{k-1} s_{k-1} s_{k-1}^t. \end{aligned} \tag{5.8}$$

Agora, para $k > m-1$ obtemos a fórmula de atualização a seguir

$$\begin{aligned} H_k &= (W_{k-1}^t \cdots W_{k-m}^t) H_k^0 (W_{k-m} \cdots W_{k-1}) \\ &+ \rho_{k-m} (W_{k-1}^t \cdots W_{k-m+1}^t) s_{k-m} s_{k-m}^t (W_{k-m+1} \cdots W_{k-1}) \\ &+ \rho_{k-m+1} (W_{k-1}^t \cdots W_{k-m+2}^t) s_{k-m+1} s_{k-m+1}^t (W_{k-m+2} \cdots W_{k-1}) \\ &+ \ldots \\ &+ \rho_{k-1} s_{k-1} s_{k-1}^t. \end{aligned} \tag{5.9}$$

A partir da expressão (5.9) é possível definir uma rotina eficiente para o cálculo do produto $H_k \nabla f(x_k)$, visto que as matrizes W_k têm posto 1 (veja (5.5). A rotina é resumida no Algoritmo 11.

Desconsiderando a multiplicação matriz-vetor $H_k^0 q$, os dois laços de repetição do

Algoritmo 11 Cálculo do produto $H_k \nabla f(x_k)$ eficientemente

$q \leftarrow \nabla f(x_k)$
para $i = k-1, k-2, \ldots, k-m$ **faça**
    $\alpha_i \leftarrow \rho_i s_i^t q$;
    $q \leftarrow q - \alpha_i y_i$;
fim para
$r \leftarrow H_k^0 q$;
para $i = k-m, k-m+1, \ldots, k-1$ **faça**
    $\beta \leftarrow \rho_i y_i^t r$;
    $r \leftarrow r + s_i(\alpha_i - \beta)$
fim para
Saída: resultado do produto $H_k \nabla f(x_k) = r$

Algoritmo 11 requerem $4mn$ multiplicações, mas se H_k^0 for diagonal terá n multiplicações adicionais. Além dessa rotina ter baixo custo computacional, também é possível realizar a substituição da matriz H_k^0 ao longo das iterações, visto que a matriz não está nos laços. Assim, o uso de memória também é baixo.

A matriz inicial H_k^0 é geralmente definida como γI, mas não há uma estratégia geral para a escolha do parâmetro γ. Por exemplo, se γ for grande, de modo que a direção inicial d_0 seja longa, então necessitaremos de muitas avaliações de função até encontrar um valor adequado para α_0. Assim, uma escolha que provou ser eficaz, na prática (veja [7]), é tomar $H_k^0 = \gamma_k I$, onde

$$\gamma_k = \frac{s_{k-1}^t y_{k-1}}{y_{k-1}^t y_{k-1}}. \tag{5.10}$$

Nesse caso, γ_k é chamado fator de escala e estimará o comportamento verdadeiro da hessiana ao longo da direção de busca. Tal escolha nos fornece um bom dimensionamento da direção de busca, resultando na aceitação do passo inicial $\alpha_k = 1$ com mais frequência (um estudo detalhado é realizado em [7]). Cabe ressaltar, que o tamanho de passo é determinado de modo que valha as condições de Wolfe (4.12)–(4.13), para que a atualização do BFGS seja estável.

A convergência global do L-BFGS não será realizada nesse trabalho, mas pode ser encontrada nas referências do capítulo. O método BFGS com memória limitada é resumido no Algoritmo 12.

Algoritmo 12 Método L-BFGS

Entrada: $x_0 \in \mathbb{R}^n$, $m > 0$ inteiro, tolerância $\epsilon > 0$;
1: **para** $\|\nabla f(x_k)\| > \epsilon$ **faça**
2: Escolha H_k^0 como proposto em (5.10);
3: Calcule $d_k \leftarrow -H_k \nabla f(x_k)$ pelo Algoritmo 11;
4: Calcule $x_{k+1} = x_k + \alpha_k d_k$, onde o tamanho de passo α_k é determinado de modo que satisfaça as condições de Wolfe (4.12)–(4.13);
5: **se** $k > m$ **então**
6: Descarte o par de vetores $\{s_{k-m}, y_{k-m}\}$ da memória;
7: **fim se**
8: Calcule e salve $s_k = x_{k+1}x_k,\ y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$;
9: **fim para**

5.2 Testes numéricos

5.2.1 O problema de regressão *lasso*

Em problemas de regressão linear são dados N pontos ou amostras $\{(x_i, y_i)\}_{i=1}^N$, onde $x_i = (x_{i1}, \ldots, x_{ip})$ é um vetor de características de dimensão p, e cada $y_i \in \mathbb{R}$ é a variável de resposta associada. Estamos interessados em aproximar a variável de resposta y_i, em pontos não amostrais, assumindo que os dados possuem uma relação linear, ou seja, que possamos estimar um parâmetro β de modo que

$$y_i \approx \upsilon(x_i) = \sum_{j=1}^{p} \beta_j x_{ij}, \tag{5.11}$$

chamamos $\beta = (\beta_1, \ldots, \beta_p)^t \in \mathbb{R}^p$ de vetor de pesos.

Modelos que utilizam regressão linear estão presentes nas mais diversas áreas. Em particular, na área de finanças com a precificação de ativos [32], na economia com a previsão de gastos de consumo [33], estoque e importações, e no aprendizado de máquina, mais especificamente, no aprendizado de máquina supervisionado [34].

Os parâmetros do modelo são frequentemente determinados pelo método de mínimos quadrados, onde minimizamos a soma do quadrado dos erros. Ou seja,

$$\underset{\beta}{\text{Minimizar}}\ \frac{1}{2}\sum_{i=1}^{n} e_i^2 = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2.$$

Quando $p > n$, a solução encontrada pelo método de mínimos quadrados se torna um problema, pois ela não é única. Outro problema é a falta de esparsidade da solução. Para lidar com isso, introduzimos o *lasso* (do inglês *least absolute shrinkage and selection*

operator) proposto por Tibishirani [35], que impõe restrições sobre os parâmetros β. Assim, temos o seguinte problema:

$$\underset{\beta}{\text{Minimizar}} \ \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 \text{ sujeito a } \|\beta\|_1 \leq t, \tag{5.12}$$

em que $\|\beta\|_1 = \sum_{j=1}^{p}|\beta_j|$ e t é um parâmetro que limita o quão bem podemos ajustar os dados, que ostuma ser especificado por procedimentos externos, como validação cruzada [36].

A norma 1 é preferível em vez da norma euclidiana devido carregar uma propriedade importante, que faz com que muitos parâmetros fiquem com estimativas iguais a zero, reduzindo o número de variáveis independentes. Maiores detalhes dessa propriedade são discutidas por Travor et al. [37]. Problemas da forma (5.12) onde se utiliza a norma euclidiana, ao invés da norma 1, são chamados problemas de regressão *ridge* [38].

O problema de otimização (5.12) pode ser reescrito penalizando a restrição $\|\beta\|_1 - t \leq 0$

$$\underset{\beta}{\text{Minimizar}} \ \frac{1}{2}\|y - X\beta\|_2^2 + \mu\|\beta\|_1, \tag{5.13}$$

onde $y = (y_1, \ldots, y_n)^t$ é o vetor de respostas, X é uma matriz de ordem $n \times p$ o qual cada linha representa o valor das p variáveis dependentes para cada observação e $\mu \geq 0 \in \mathbb{R}$ é o parâmetro penalizador que possui uma relação com t. Note que $t \to \infty$ equivale a $\mu \to 0$.

Nessa seção, discutiremos a eficácia e eficiência dos métodos SPG, CG-descent e L-BFGS, aplicados aos problemas de regressão lasso. Para tal, trabalharemos com a forma (5.13) por conveniência numérica e ser um problema sem restrições. Como o termo $\|\beta\|_1$ não é diferenciável, teremos que realizar uma aproximação do problema lasso, assim, considerando $\delta > 0$ um escalar fixo (veja [39]), resolvemos

$$\underset{\beta}{\text{Minimizar}} \ f(\beta) = \frac{1}{2}\|y - X\beta\|_2^2 + \mu\sum_{i=1}^{p}\sqrt{\beta_i^2 + \delta}. \tag{5.14}$$

Note que (5.14) retorna ao problema original (5.13) quando $\delta \to 0$. O gradiente da função objetivo do problema (5.14) é

$$\nabla f(\beta) = (X^tX)\beta - X^ty + \mu\begin{bmatrix} \beta_1/\sqrt{\beta_1^2 + \delta} \\ \vdots \\ \beta_p/\sqrt{\beta_p^2 + \delta} \end{bmatrix}. \tag{5.15}$$

Tabela 2: Especificações das matrizes selecionadas para os testes numéricos dos problemas de regressão *lasso*. Sendo n, p e nnz, respectivamente, o número de linhas, colunas e não zeros da matriz.

Matriz	Grupo	n	p	nnz
Maragal_1	NYPA	32	14	234
ash219	HB	219	85	438
ash85	HB	85	85	523
ash331	HB	331	104	662
ash608	HB	608	188	1216
abb313	HB	313	176	1557
ash958	HB	958	292	1916
ash292	HB	292	292	2208
Maragal_2	NYPA	555	350	4357
illc1033	HB	1033	320	4719
well1033	HB	1033	320	4732
illc1850	HB	1850	712	8636
well1850	HB	1850	712	8755
Maragal_3	NYPA	1690	860	18391
Maragal_4	NYPA	1964	1034	26719
Maragal_5	NYPA	4654	3320	93091
Maragal_6	NYPA	21255	10152	537694
Delor64K	ANSYS	64719	1785345	652140
landmark	Pereyra	71952	2704	1146848
Maragal_7	NYPA	46845	26564	1200537
Maragal_8	NYPA	33212	75077	1308415
Delor295K	ANSYS	295734	1823928	2401323
Delor338K	ANSYS	343236	887058	4211599
ESOC	Springer	327062	37830	6019939
sls	Bates	1748122	62729	6804304
Rucci1	Rucci	1977885	10990	7791168

Nos testes numéricos, consideramos matrizes X obtidas da coletânea Suite Sparse Matrix Collection [10], a mesma utilizada em experimentos anteriores, mas nesse caso foram consideradas matrizes oriundas de problemas de mínimos quadrados. O vetor de resposta y não é fornecido com as matrizes, assim, tomamos $y = (1, 1, \ldots, 1)^t$. Selecionamos todas as matrizes disponíveis com os mais diversos tamanhos, variando-se de milhares até milhões de não zeros, totalizando 26 matrizes. Suas especificações estão na Tabela 2.

No problema lasso (5.14) utilizamos $\mu = 10^{-3}$ e $\delta = 10^{-6}$, conforme recomendado por Karimi e Vavasis [39]. O ponto inicial foi tomado como $\beta_0 = (1, 1, \ldots, 1)^t$, declaramos convergência quando

$$\|\nabla f(\beta)\|_\infty < 10^{-7}.$$

Fixamos um número de iterações de no máximo 5×10^4. Os parâmetros dos métodos foram os mesmos dos testes anteriores para os problemas da biblioteca CUTEst, ou seja, para o

SPG consideramos na busca não monótona $\eta = 10^{-4}, \rho = (0,5), \sigma_1 = 0,1$ e $\sigma_2 = (0,9)$, outros parâmetros foram $M = 100$, $\lambda_{min} = 10^{-30}, \lambda_{max} = 10^{30}$ e o passo espectral foi inicializado como

$$\lambda_0 = \max\left\{\lambda_{min}, \min\left\{\frac{1}{\|P_\Omega(x_0 - \nabla f(x_0)) - x_0\|_\infty}, \lambda_{max}\right\}\right\},$$

assumindo que x_0 é tal que $P_{\mathbb{R}^n}(x_0 - \nabla f(x_0)) = -\nabla f(x_0) \neq x_0$.

Os métodos CG-descent, Algoritmo 10, e L-BFGS, Algoritmo 12, foram obtidos do pacote `Optim.jl` (`https://github.com/JuliaNLSolvers/Optim.jl`). Na busca linear, utilizamos a busca Hager-Zhang, Algoritmo 9, retirada do pacote `LineSearches.jl` (`https://github.com/JuliaNLSolvers/LineSearches.jl`), com os seguintes parâmetros: $\xi = (0,9)$, $\eta = 10^{-2}$, para o CG-descent, $\eta = 10^{-4}$ para o L-BFGS e

$$\gamma = (0,66), \quad \theta = 0,5 \text{ e } \omega = 10^{-6},$$

lembrando que o último está relacionado com uma estimativa para o erro do valor funcional na Equação (4.17). Além disso, para o método L-BFGS tomamos o parâmetro de armazenamento $m = 2$ e os demais, de ambos os métodos, foram adquiridos das referências [6, 29].

Como faremos uma análise do tempo computacional é interessante realizarmos uma boa estimativa, em particular para problemas onde o tempo de resolução é pequeno e podem ocorrer muitas variações. Assim, seja n o número de execuções realizadas até que a soma dos tempos em cada compilação seja superior a 15 segundos. Logo, uma boa estimativa do tempo é dada pela média, ou seja,

$$\text{tempo} = \frac{t_1 + t_2 + \cdots + t_n}{n},$$

em que t_i $(i = 1, \ldots, n)$ é o tempo de resolução na execução i.

Assim como realizado em testes computacionais anteriores, construímos os perfis de desempenho propostos pela Dolan e Moré [8] usando o pacote `BenchmarkProfiles.jl` (`https://github.com/JuliaSmoothOptimizers/BenchmarkProfiles.jl`) disponibilizado na linguagem Julia, como já discutido e explicado. Na Tabela 8 estão presentes todos os resultados encontrados a partir da aplicação dos métodos aos problemas de regressão *lasso*.

Nas Figuras 20 e 21 apresentamos os perfis de desempenho correspondente ao tempo de CPU e número de avaliações de gradiente, respectivamente. Diante dos perfis, pode-se observar que o método L-BFGS teve a melhor eficácia, resolvendo pouco mais de 75%

dos problemas propostos e sendo mais veloz em certa de 20% deles. Assim, o L-BFGS é melhor que CG-descent em eficácia, mas perdendo em velocidade. Basta notar que CG-descent teve uma eficácia de cerca de 62%, mas sendo o mais veloz em $\approx 38\%$ deles. Dentre os métodos testados, o SPG teve o pior desempenho, com uma eficácia de 50% e sendo o mais veloz em cerca de 16% dos problemas.

Podemos observar da Equação (5.15) que a avaliação de gradiente é cara computacionalmente, visto que teríamos que realizar, ao menos, um produto matriz-matriz (X^tX) e dois produtos matriz-vetor $(X^ty$ e $(X^tX)\beta)$. Assim, é interessante estudarmos a quantidade de avaliações de gradiente realizadas por cada método, para uma boa análise de desempenho, especialmente para problemas grandes. Observando a Figura 21, repare que CG-descent foi o melhor em número de avaliações de gradiente, resolvendo cerca de 55% dos problemas no menor número de avaliações e mantendo a eficácia já comentada. Isso corresponde a mais que o dobro de problemas resolvidos pelo L-BFGS na mesma análise, tornando o CG-descent o melhor quanto à avaliação de gradiente, mas o segundo colocado em eficácia. Note que, no quesito de menor número de avaliações de gradiente, o SPG resolveu cerca de 5% e o L-BFGS em torno de 20% dos problemas apresentados, não sendo tão bons comparados ao CG-descent.

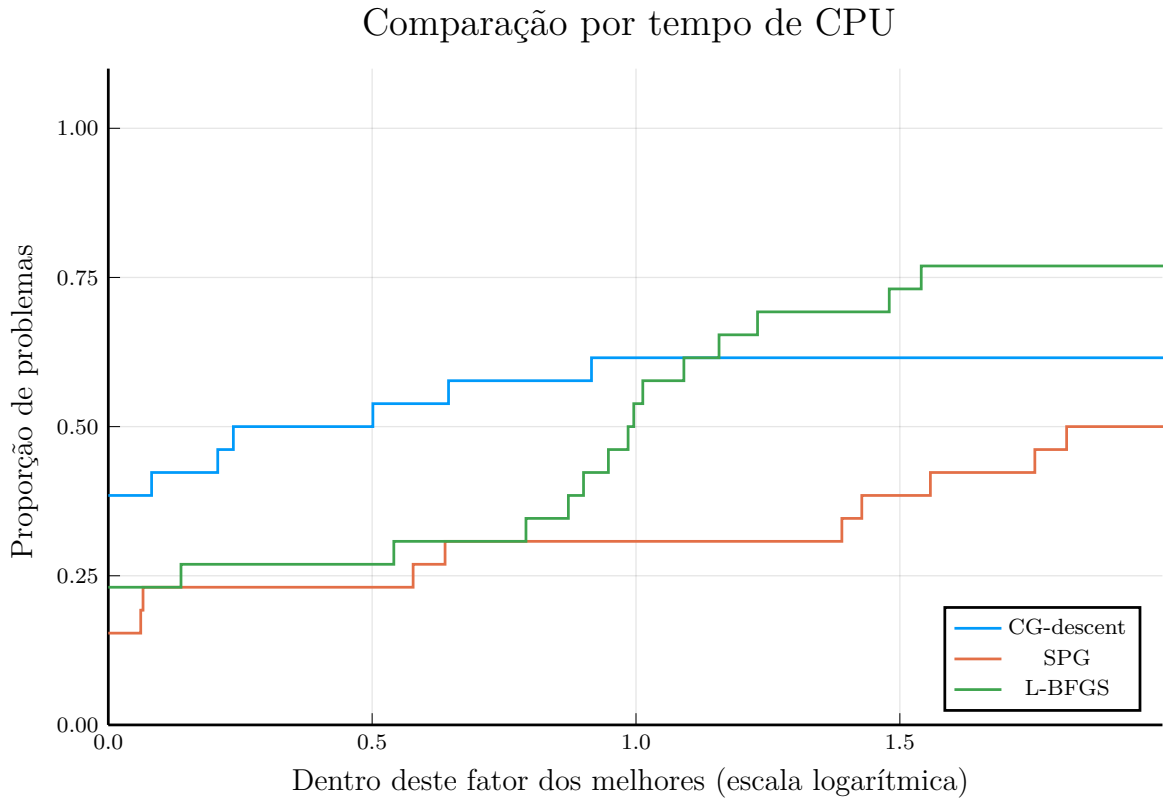


Figura 20: Perfil de desempenho com relação ao tempo computacional dos métodos CG-descent, SPG e L-BFGS aplicados aos problemas de regressão *lasso*.

Outros resultados que podem ser obtidos dos perfis são:

1. Se estivermos dispostos a esperar $2^{1,1} \approx 2,14$ vezes o tempo do algoritmo mais veloz, a eficácia do L-BFGS supera a do CG-descent.

2. A eficácia do L-BFGS supera a do CG-descent se estivermos dispostos a realizar $2^{1,5} \approx 2,8$ vezes mais avaliações de gradiente que o melhor método.

3. O método mais balanceado em termo de eficácia, velocidade e avaliações de gradiente é CG-descent.

4. O L-BFGS e SPG são métodos que requerem mais avaliações de gradiente e tempo computacional.

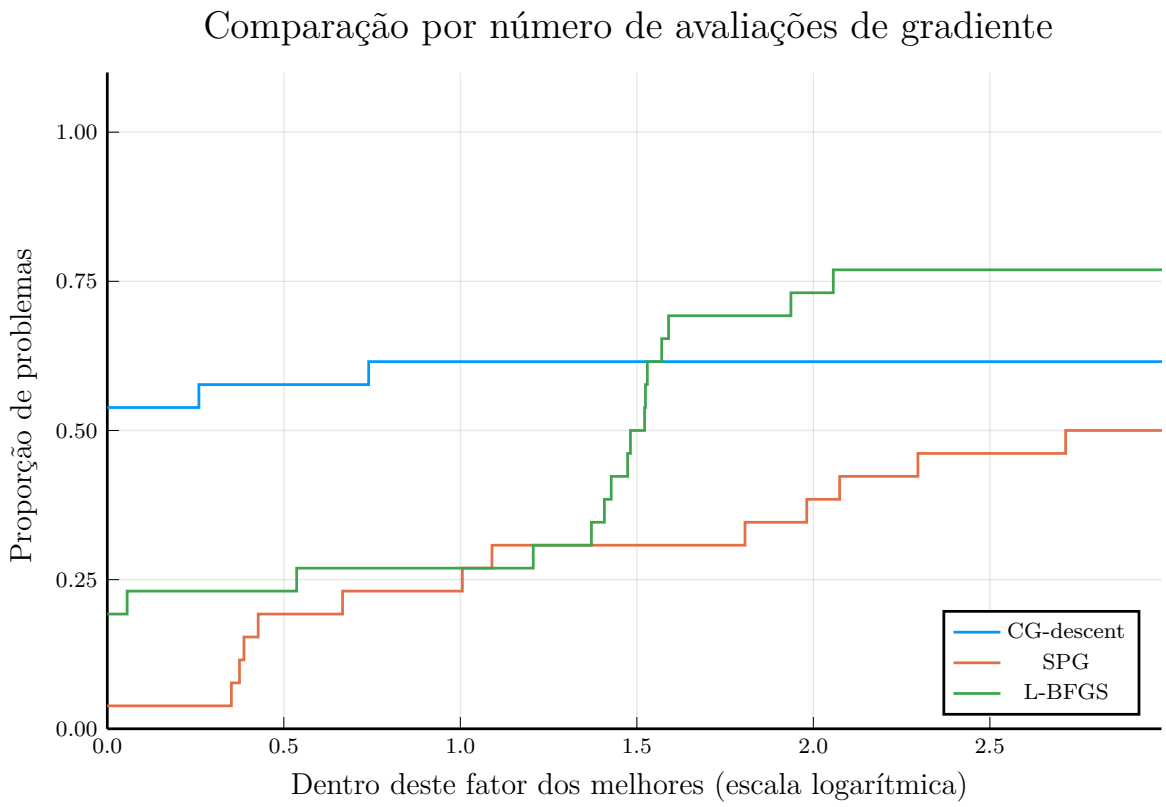


Figura 21: Perfil de desempenho com relação ao número de avaliações de gradiente dos métodos CG-descent, SPG e L-BFGS aplicados aos problemas de regressão *lasso*.

5.2.2 Uma aplicação na biologia celular

O sequenciamento de RNA (ácido ribonucleico) de célula única (scRNA-seq) consiste em uma abordagem genômica para caracterizar os perfis de expressão gênica de células

individuais. O scRNA-seq tem sido utilizado em estudos de biologia celular, pois permite a avaliação de propriedades biológicas fundamentais de populações de células e sistemas biológicos.

No scRNA-seq o RNA é isolado de células individuais e convertido em fragmentos de DNA complementar (cDNA) por uma transcriptase reversa. Em seguida, os fragmentos de cDNA são amplificados para gerar quantidades suficientes de material, e sequenciados com o uso de plataformas de sequenciamento massivo em paralelo [40]. O mapeamento das leituras de sequências permite quantificar a expressão gênica em cada célula, com base no número de leituras atribuídas a cada gene. Com isso, os dados de contagem dos genes podem ser utilizados para identificar e analisar categorias de células e genes altamente variáveis. Uma introdução ao scRNA-seq é abordada por Olsen e Baryawno [41]. Na Figura 22, apresentamos um breve resumo dos passos realizados na estratégia de scRNA-sec aplicadas a um dado tecido.

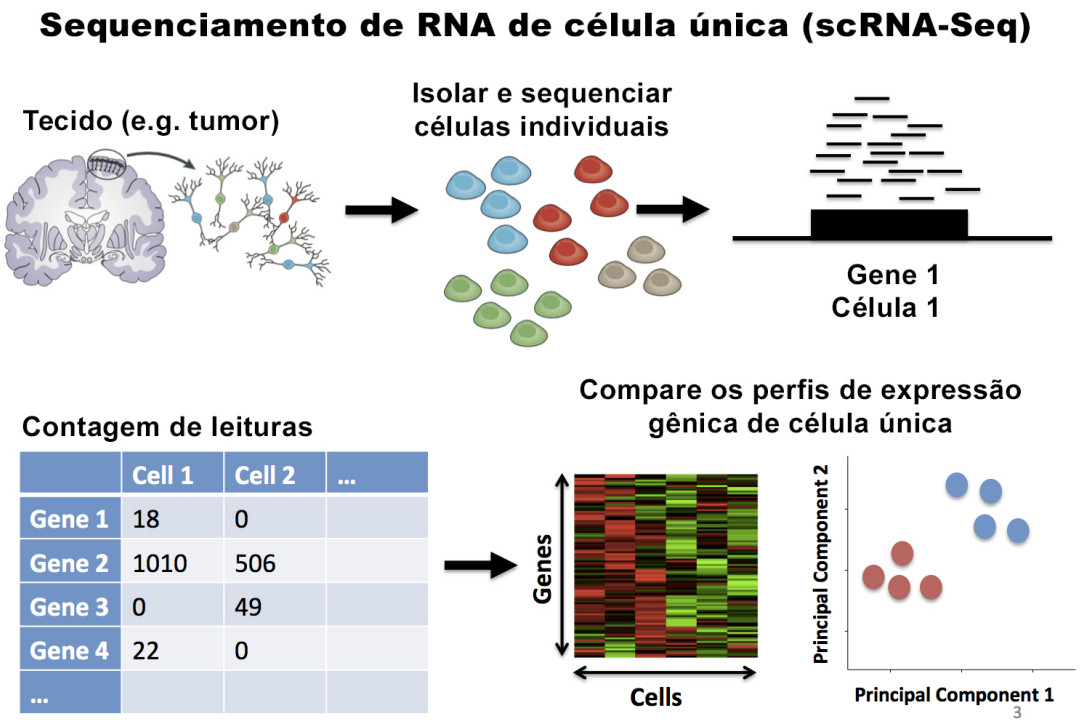


Figura 22: Exemplo das etapas do sequenciamento de RNA de célula única em um tumor cerebral. (Adaptado de `https://learn.gencore.bio.nyu.edu/single-cell-rnaseq`)

No contexto desse trabalho, nosso objetivo é a construção e resolução de um modelo de otimização que classifica dois estágios de uma célula obtida de um camundongo [42]. Para tal, consideramos os dados de scRNA-seq dos estágios G1 e G2M do banco de dados *E-MTAB-2805*, obtido da coletânea Conquer (`http://imlspenticton.uzh.ch:`

3838/conquer/), sendo um conjunto de dados públicos de RNA-seq de célula única, consistente, processadas e prontas para análise. Veja [43]. Nessa coletânea, cada conjunto de dados tem estimativas de contagem e transcrições por milhão (TPM) para genes e transcrições, além de fornecer relatórios de controle de qualidade e análise exploratória.

É dado um conjunto de pares, comumente chamado dados de treinamento, $(x_i, y_i)_{i=1}^n$ onde $x_i = (x_{i1}, \ldots, x_{id}) \in \mathbb{R}^d$ representa os níveis de expressão gênica para os d genes, medidos na i-ésima célula pela estratégia de scRNA-seq. O termo $y_i \in \{0, 1\}$ é a resposta para representar a categoria da i-ésima célula, e denotamos $x_{i,j} \in \mathbb{R}$ como o nível de expressão do gene j na célula i.

De modo a resolver o problema de classificação, consideramos o modelo regressão logística, análogo ao modelo de regressão linear abordado na Seção 5.2.1. Aqui, ao contrário da regressão linear onde $y_i \in \mathbb{R}$, a regressão logística precisa necessita y_i entre 0 e 1. Assim, de modo a limitar o valor a ser previsto dentro desse intervalo, aplicamos o logaritmo $\log(p_i/(1 - p_i))$ onde p_i é a probabilidade do evento i ocorrer. O modelo logístico é então descrito como

$$\log\left(\frac{p_i}{1 - p_i}\right) = x_i\beta, \tag{5.16}$$

ou expresso em termos da resposta

$$y_i \approx p_i = \exp(x_i\beta)[1 + \exp(x_i\beta)]^{-1}. \tag{5.17}$$

A expressão (5.17) é obtida aplicando a função exponencial em ambos os lados da igualdade (5.16) e uma pequena manipulação algébrica.

Visando determinar o parâmetro β da equação (5.17), uma estratégia comum é maximizar o logarítmico da função verossimilhança, que aproxima a probabilidade de que um determinado conjunto de dados seja produzido por uma determinada função logística:

$$l = \sum_{i=1}^{n} y_i \log(p_i) + (1 - y_i) \log(1 - p_i). \tag{5.18}$$

Sabendo que p_i é dado por (5.17) e maximizar l é igual a minimizar $-l$, maximizar (5.18) é equivalente a solucionar

$$\min_{\beta} - \sum_{i=1}^{n} [y_i x_i \beta - \log(1 + \exp(x_i\beta))]. \tag{5.19}$$

Além de utilizarmos o modelo acima também trabalharemos com os modelos regulari-

zados *lasso* o *ridge*, abordados anteriormente. Logo, aplicamos a penalização ao problema (5.19) e obtemos o seguinte problema otimização:

$$\min_{\beta} f(\beta) = -\sum_{i=1}^{n}[y_i x_i \beta - \log(1 + \exp(x_i\beta))] + \mu\Omega(\beta), \tag{5.20}$$

onde $\Omega(\beta)$ é uma penalidade (opcional) para controlar o sobre-ajuste em dimensões altas, e μ é o parâmetro de regularização para controlar o equilíbrio entre *under* e *overfitting*. Escolhas populares para $\Omega(\beta)$ incluem a penalização *ridge* [38]

$$\Omega_{ridge}(\beta) = \|\beta\|_2^2, \tag{5.21}$$

e o *lasso* [35]

$$\Omega_{lasso}(\beta) = \|\beta\|_1. \tag{5.22}$$

A função e o gradiente foram fornecidas manualmente aos métodos. É simples mostrar que o gradiente de $f(\beta)$ é dado por

$$\nabla f(\beta) = -\sum_{i=1}^{n}\left(y_i - \frac{exp(x_i\beta)}{1 + exp(x_i\beta)}\right)x_i^t + \mu\,\frac{\partial\Omega}{\partial\beta}, \tag{5.23}$$

onde $\partial\Omega/\partial\beta$ é a derivada da regularização em relação a β. Repare que o *lasso* (5.22) não é diferenciável, então realizaremos uma estratégia diferente da abordada na seção anterior. Para um melhor entendimento, considere a função $g : \mathbb{R} \to \mathbb{R}$ definida por $g(x) = |x|$ (observe a Figura 23). Para $x < 0$ temos $g'(x) = -1$ e para $x > 0$, $g'(x) = 1$, mas a derivada não está definida em $x = 0$. Assim, é razoável impormos que $g'(0) = 0$ visto que o mínimo da função ocorre em $x = 0$ (em realidade, estamos tomando um *subgradiente* particular de g em $x = 0$). Uma discussão mais elaborada, que trabalha com a ideia de subgradientes, e está fora do escopo desse trabalho, é feita por Bertsekas [44].

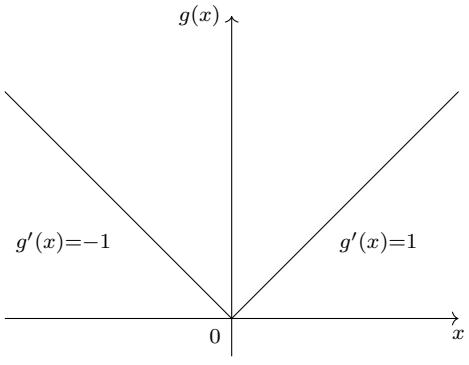


Figura 23: Esboço do gráfico da função $g(x) = |x|$.

Dessa maneira, tomaremos cada entrada da derivada da regularização *lasso* como

$$\left(\frac{\partial \Omega_{lasso}}{\partial \beta}\right)_i = \begin{cases} 1, & \text{se } \beta_i > 0 \\ -1, & \text{se } \beta_i < 0 \\ 0, & \text{caso contrário} \end{cases} \qquad \text{para } i = 1, \ldots, n. \tag{5.24}$$

Nessa seção, estudaremos o desempenho dos métodos SPG, CG-descent e L-BFGS aplicados a um problema de classificação de estágios celulares, ou seja, visamos resolver o problema de otimização (5.20) para diferentes penalizações e ainda analisar a qualidade das soluções encontradas.

Após a remoção de genes com contagens nulas em todas as células, os dados do sequenciamento totalizaram 192 células e 24.612 genes. Para a aplicação no problema de otimização (5.20) realizamos um embaralhamento das informações, para aumentar a variabilidade, e consideramos cerca de 20% das células, totalizando 38 células para o treinamento. O restante é utilizado para testar o modelo gerado, considerando a solução ótima encontrada.

Os parâmetros dos métodos foram os mesmos utilizados para os problemas de regressão *lasso* da seção anterior. O ponto inicial foi tomado como $\beta_0 = (0, 0, \ldots, 0)^t$, $\mu = 10^{-2}$ e declaramos convergência quando

$$\|\nabla f(\beta_k)\|_\infty < 10^{-8},$$

consideramos também um número de iterações de no máximo 200.

Os resultados dos testes numéricos são apresentados na Tabela 3, onde It, nf e ng são, respectivamente, o número de iterações, avaliações de função e avaliações de gradiente. Note que, para o problema em que foi considerado a regressão *lasso*, os testes não atingiram a precisão exigida, visto que, como abordamos anteriormente, realizamos uma aproximação para a derivada, então é comum o método contornar a solução, mas nunca a alcançar efetivamente. Para os problemas em que consideramos a regressão *ridge*, os resultados foram bons, todos os métodos alcançaram a precisão exigida. Dentre eles, o melhor foi o CG-descent resolvendo o problema no menor tempo e com o menor número de avaliações de função e gradiente.

Vimos que os métodos testados se saíram razoavelmente bem considerando as regularizações utilizadas. Analisaremos agora as soluções obtidas do problema de classificação.

Tabela 3: Resultados obtidos da aplicação dos métodos SPG, CG-descent e L-BFGS ao problema de otimização (5.20) com diferentes regularizações, utilizando os dados propostos para o treinamento.

Regularização	Método	It	f	$\|\nabla f\|_\infty$	st	nf	ng	Tempo(s)
Lasso	SPG	200	5,23E-02	5,51E+00	1	1023	201	1,10E+02
	CG-descent	200	1,65E-04	1,07E-02	1	294	142	4,53E+01
	L-BFGS	200	3,02E-04	1,00E-02	1	627	627	1,28E+02
Ridge	SPG	161	2,57E-08	2,74E-09	0	182	162	3,59E+01
	CG-descent	49	2,57E-08	9,12E-09	0	128	71	2,37E+01
	L-BFGS	26	2,57E-08	4,97E-09	0	190	190	5,45E+01

Dado β^* obtido da solução do problema (5.20), a solução para a classificação é da forma

$$y_i = \begin{cases} 1, & \text{se } p_i > 0,5 \\ 0, & \text{caso contrário} \end{cases} \tag{5.25}$$

em que,

$$p_i = \exp(x_i\beta^*)[1 + \exp(x_i\beta^*)]^{-1}. \tag{5.26}$$

A partir disso, construímos uma tabela contendo a porcentagem de acertos de cada método e modelo de regularização considerado. Evidenciamos o quão efetivo é a solução obtida nos dados de treinamento e os dados de teste. Observe a Tabela 4.

Tabela 4: Porcentagem de acertos na classificação considerando os dados de treinamento e testes.

Regularização	Método	Treinamento	Testes
Lasso	SPG	100%	100%
	CG-descent	100%	100%
	L-BFGS	100%	100%
Ridge	SPG	100%	100%
	CG-descent	100%	100%
	L-BFGS	100%	100%

Para entender o porquê desse comportamento, construiremos uma figura da matriz de correlação entre algumas das colunas dos dados considerados. A matriz de correlação nada mais é que uma tabela que exibe os coeficientes de correlação para diferentes variáveis, ou seja, ela representará a correspondência entre todos os possíveis pares de valores em uma determinada tabela de dados. Quanto mais próximo de 1 os valores estiverem, mais correlatos são os dados, e menos relevantes são para o modelo, ou seja, podemos os remover sem haver um grande impacto na solução. Vale notar que cada dado é correlato com ele

mesmo.

Na Figura 24, apresentamos uma ilustração da matriz de correlação entre algumas das colunas dos dados considerados. Note que na maioria dos dados a correlação é baixa, logo essas informações possuem grande importância para o modelo. Esse comportamento se mantém quando aumentamos o número de colunas consideradas. Portanto, devido à abundância de genes, cada informação sobre a categoria da célula possui uma maior relevância. Basta então um pequeno número de dados de treinamento para que o modelo produza bons resultados, como observado.

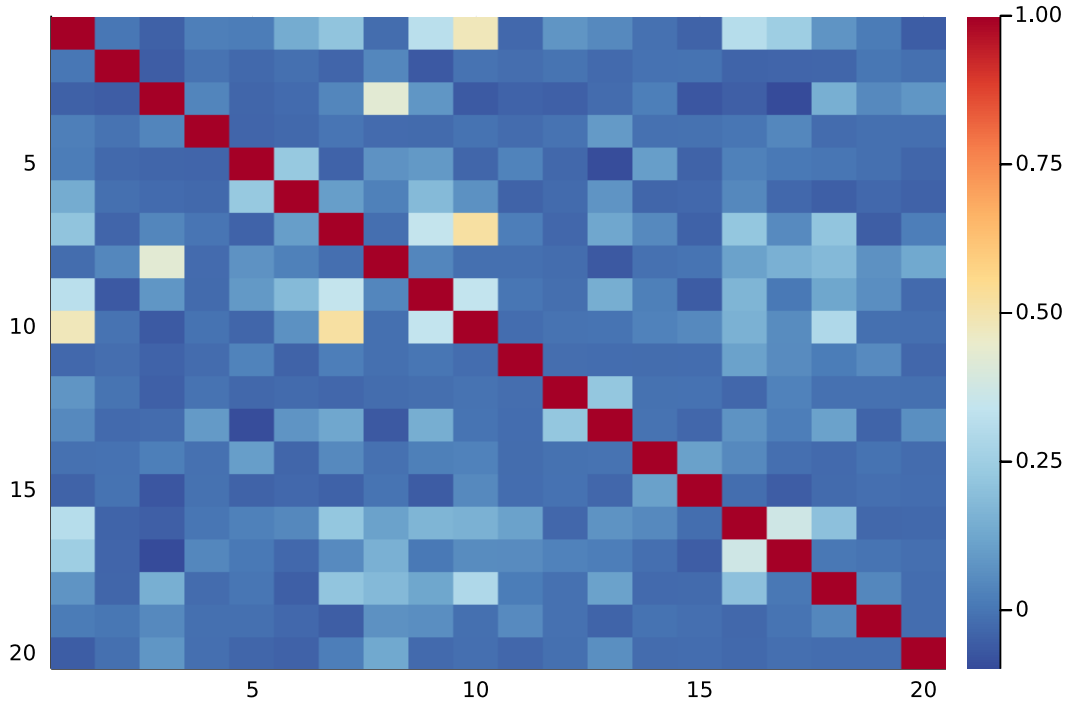


Figura 24: Ilustração da correlação entre algumas das colunas dos dados de scRNA-seq dos estágios celulares G1 e G2M.

6 Conclusões

Nesse trabalho foram considerados, principalmente, quatro métodos aplicados a problemas de otimização irrestrita ou com restrições simples (neste trabalho, *restrições de caixa* $\ell \leq x \leq u$), adequados à larga escala. O estudo de métodos para tais problemas é importante devido à crescente disponibilidade de dados, e o consequente aumento dos modelos de otimização. Foi desenvolvido um estudo teórico básico sobre cada um dos métodos, apresentando a convergência para os métodos de descida tipo gradiente, em particular, para o Gradiente Projetado e o Gradiente Espectral Projetado (do inglês, SPG).

Ao longo da pesquisa, realizamos diversas simulações para averiguar o desempenho de cada método em diferentes categorias de problemas. Conduzimos uma análise de perfis de desempenho em relação ao tempo de resolução, número de iterações e avaliações de gradiente, visando determinar o método mais adequado a cada situação. As implementações foram feitas em Julia, com eventual auxílio de implementações eficientes feitas pela comunidade acadêmica.

Conforme discutido no Capítulo 4, para problemas de otimização irrestrita os métodos SPG e gradientes conjugados de Hager e Zhang [5] (CG-descent) apresentaram ótimos resultados numéricos, tanto em velocidade quanto em eficácia. Com os testes, observamos também que o método tradicional do gradiente é lento, o que está em concordância com a literatura. Assim, o método do Gradiente Espectral Projetado provou-se ser o melhor quando aplicados à problemas irrestritos gerais, cuja função objetivo é, em geral, não linear e com um grande número de variáveis.

Para os problemas com restrições de caixa observamos um comportamento semelhante aos testes irrestritos. O método SPG se manteve com os melhores resultados e o Gradiente Projetado com os piores. Logo, o método SPG mostrou ser muito efetivo quando aplicado à problemas com restrições de caixa.

Visto que o método de gradientes conjugados, originalmente, foi desenvolvido para

minimização de quadráticas, avaliamos o desempenho de sua generalização, o método CG-descent, na minimização de funções que, apesar de não quadráticas, se parecerem de certa forma com quadráticas. Neste caso, percebemos CG-descent é mais eficiente e eficaz que SPG. Os problemas utilizados estão relacionados ao condensado de Bose-Einstein, descritos na Seção 4.3.2. Esses problemas consistem em minimizar a soma de uma função quadrática de hessiana A com um termo não quadrático quártico. Vimos que CG-descent se beneficia de problemas onde $\lambda_{\min}(A)$ é grande, isto é, quando a função a ser minimizada se aproxima de uma quadrática. Com isso, caso saibamos que a estrutura da função objetivo se assemelha a uma quadrática, é interessante aplicarmos o método CG-descent. Caso contrário, assumimos o pior caso possível: problema de otimização não linear com um grande número de variáveis, e o método do gradiente espectral projetado é o recomendado.

No capítulo 5 introduzimos o método L-BFGS, que basicamente é o método BFGS onde se aproxima a matriz da atualização quase-Newton visando pouco uso de memória. Vimos que o custo por iteração é reduzido, pois, o cálculo do produto $H_k \nabla f(x_k)$ é realizado de maneira eficiente e com um baixo uso de memória. Observamos que, para o problema de regressão *lasso* o L-BFGS é o mais eficaz, mas não o mais veloz, perdendo para o CG-descent nesse quesito. Nesses casos, a função objetivo é similar a uma quadrática. Então já era esperado resultados satisfatórios com o método CG-descent, conforme discutido no capítulo 4. O SPG foi pior dentre os três, mas os resultados ainda foram bons. Observamos que em muitas instâncias o método se aproxima da solução, ao custo de mais iterações. Cabe ressaltar, no entanto, que, dos métodos testados, o SPG é aquele com o menor custo por iteração, sendo interessante em muitas aplicações deixar com que o método realize mais iterações.

Além do problema de regressão *lasso*, abordamos uma aplicação na biologia celular, onde dado uma sequência genética de uma célula, obtida pela estratégia de scRNA-seq, prevíamos em qual estágio se adequaria G1 ou G2M. Para tal, consideramos um modelo de regressão logística, onde os parâmetros foram estimados de modo a minimizar $-l$ acrescido de uma regularização, em que l é o logarítmico da função de verossimilhança. Observamos que, devido à aproximação realizada para a derivada do *lasso*, os métodos não alcançaram a precisão exigida, mas ficaram relativamente próximos do minimizador, diferente da regularização *ridge*, que convergiu rapidamente. Após avaliar a solução encontrada em conjunto com a classificação obtida, observamos um comportamento incomum. Devido à alta variabilidade dos dados em um pequeno conjunto de treinamento, o modelo previu perfeitamente em qual categoria cada célula dos dados de teste se encaixa. Com isso,

concluímos que os métodos SPG, CG-descent e L-BFGS podem ser aplicados com sucesso a problemas oriundos de classificação, visto que no modelo considerado observamos um bom desempenho.

APÊNDICE A – Tabelas dos testes numéricos

Considere n o tamanho do problema ou a ordem da matriz, It o número de iterações, f aproximação do valor ótimo, $\|\nabla f\|_\infty$ norma do máximo do gradiente no último ponto calculado, st a situação do problema (0–resolvido, 1–caso contrário), nf é o número de avaliações de função, ng o número de avaliações de gradiente e $\|d\|_\infty$ norma do máximo da direção projetada no último iterando.

Tabela 5: Resultados numéricos obtidos pela aplicação dos métodos do gradiente, SPG e CG-descent para problemas de otimização irrestrita da Seção 4.3.1.

Problema	n	Método	It	f	$\|\nabla f\|_\infty$	st	Tempo(s)
ARGLINA	200	Gradiente	1	2,00E+02	4,94E-14	0	7,50E-02
		SPG	2	2,00E+02	6,89E-14	0	1,46E-01
		CG-descent	1	2,00E+02	8,14E-13	0	1,69E+00
ARGLINB	200	Gradiente	50000	9,96E+01	3,88E+04	1	6,22E+05
		SPG	7889	9,96E+01	8,49E-08	0	1,18E+01
		CG-descent	50000	9,96E+01	2,91E-06	1	3,66E+05
ARGLINC	200	Gradiente	50000	1,01E+05	8,49E+00	1	6,52E+05
		SPG	5613	1,01E+05	3,71E-08	0	6,75E+00
		CG-descent	50000	1,01E+05	1,04E-05	1	4,13E+05
ARGTRIGLS	200	Gradiente	50000	1,20E-08	4,20E-04	1	1,10E+05
		SPG	4133	1,76E-15	8,39E-08	0	2,40E+00
		CG-descent	638	1,75E-16	6,09E-08	0	5,45E-01
BA-L1LS	57	Gradiente	57	6,60E-22	2,21E-08	0	4,10E-02
		SPG	39	4,35E-21	5,90E-08	0	2,83E-03

Tabela 5 – continuação

Problema	n	Método	It	f	$\|\nabla f\|_\infty$	st	Tempo(s)
		CG-descent	30	6,44E-22	3,15E-08	0	8,31E-03
BA-L1SPLS	57	Gradiente	109	3,71E-21	6,44E-08	0	1,00E-01
		SPG	96	1,24E-19	9,96E-08	0	1,02E-02
		CG-descent	42	9,29E-21	8,43E-08	0	1,32E-02
BROWNAL	200	Gradiente	50000	1,47E-09	1,51E-06	1	8,44E+01
		SPG	50000	6,49E+01	3,85E-01	1	2,15E+01
		CG-descent	53	7,14E-12	9,20E-08	0	6,89E-02
CHNROSNB	50	Gradiente	21734	1,34E-14	1,00E-07	0	8,67E-01
		SPG	1703	3,69E-15	9,96E-08	0	1,59E-02
		CG-descent	321	6,16E-16	9,39E-08	0	4,41E-03
CHNRSNBM	50	Gradiente	18025	1,33E-14	9,99E-08	0	7,05E-01
		SPG	1532	1,60E-15	6,55E-08	0	1,43E-02
		CG-descent	252	1,28E-16	6,50E-08	0	3,75E-03
COATING	134	Gradiente	50000	5,85E-01	6,11E-02	1	1,61E+01
		SPG	50000	5,15E-01	1,98E-02	1	3,70E+00
		CG-descent	11473	5,05E-01	9,96E-08	0	1,42E+00
DIAMON2DLS	66	Gradiente	50000	1,06E+04	9,79E+00	1	2,37E+03
		SPG	28	1,47E+04	9,58E-09	0	4,35E-01
		CG-descent	50000	1,21E+04	9,85E-01	1	1,26E+03
DIAMON3DLS	99	Gradiente	50000	1,54E+03	3,99E+05	1	3,72E+03
		SPG	35	1,48E+04	7,07E-08	0	6,24E-01
		CG-descent	50000	7,59E+03	1,19E+00	1	1,94E+03
DMN15102LS	66	Gradiente	50000	1,06E+04	9,79E+00	1	2,38E+03
		SPG	28	1,47E+04	9,58E-09	0	4,37E-01
		CG-descent	50000	1,21E+04	9,85E-01	1	1,27E+03
DMN15103LS	99	Gradiente	50000	1,54E+03	3,99E+05	1	4,49E+03
		SPG	35	1,48E+04	7,07E-08	0	7,18E-01
		CG-descent	50000	7,59E+03	1,19E+00	1	2,19E+03
DMN15332LS	66	Gradiente	50000	1,41E+03	1,47E+01	1	2,23E+03
		SPG	57	1,33E+05	3,59E-08	0	9,22E-01
		CG-descent	50000	1,69E+03	8,34E-01	1	7,20E+05

Tabela 5 – continuação

Problema	n	Método	It	f	$\|\nabla f\|_\infty$	st	Tempo(s)
DMN15333LS	99	Gradiente	50000	3,51E+05	5,69E+01	1	3,92E+03
		SPG	52	1,87E+03	4,64E-08	0	1,05E+00
		CG-descent	50000	1,01E+03	6,32E+01	1	9,63E+05
DMN37142LS	66	Gradiente	50000	2,89E+05	1,67E+00	1	2,23E+03
		SPG	50000	2,13E+05	7,06E-01	1	5,16E+05
		CG-descent	50000	3,64E+05	2,32E-04	1	1,78E+03
DMN37143LS	99	Gradiente	50000	2,91E+05	3,80E+00	1	3,42E+03
		SPG	66	7,87E+03	4,06E-08	0	1,12E+00
		CG-descent	50000	2,63E+05	2,22E+01	1	8,76E+02
EG2	1000	Gradiente	50000	-9,99E+05	1,23E-05	1	1,17E+02
		SPG	5	-9,99E+05	7,34E-14	0	1,14E-03
		CG-descent	3	-9,99E+05	7,35E-14	0	3,60E-03
ERRINROS	50	Gradiente	50000	4,02E+01	4,05E-02	1	1,94E+00
		SPG	50000	3,99E+01	7,94E-05	1	5,36E-01
		CG-descent	12654	3,99E+01	3,15E-08	0	2,02E-01
ERRINRSM	50	Gradiente	50000	3,85E+01	7,30E-02	1	2,07E+00
		SPG	50000	3,78E+01	1,23E-02	1	5,64E-01
		CG-descent	3773	3,77E+01	6,78E-08	0	7,07E-02
EXTROSNB	1000	Gradiente	50000	3,99E+00	2,72E-07	1	7,16E+01
		SPG	50000	3,10E-05	9,22E-05	1	7,56E+00
		CG-descent	28825	3,84E-08	9,92E-08	0	5,69E+00
FLETCHCR	1000	Gradiente	24706	1,33E-14	9,99E-08	0	1,86E+01
		SPG	930	3,93E-15	6,21E-08	0	1,92E-01
		CG-descent	4448	5,76E-15	9,57E-08	0	1,23E+00
GENROSE	500	Gradiente	50000	1,00E+00	4,24E-07	1	1,98E+01
		SPG	3950	1,00E+00	7,49E-08	0	4,64E-01
		CG-descent	1101	1,00E+00	8,35E-08	0	1,66E-01
HYDC20LS	99	Gradiente	50000	4,89E-01	2,11E-01	1	1,63E+01
		SPG	50000	1,72E-01	2,56E-01	1	3,24E+00
		CG-descent	50000	3,10E-03	1,17E-02	1	6,05E+00
INTEQNELS	502	Gradiente	7	3,46E-14	2,98E-08	0	2,73E-02

Tabela 5 – continuação

Problema	n	Método	It	f	$\|\nabla f\|_\infty$	st	Tempo(s)
		SPG	8	7,60E-14	4,58E-08	0	2,71E-02
		CG-descent	7	1,65E-15	8,75E-09	0	2,92E-02
KSSLS	1000	Gradiente	50000	1,31E-17	6,38E-07	1	4,99E+03
		SPG	12	5,04E-22	6,09E-10	0	1,18E-01
		CG-descent	8	2,32E-18	9,62E-08	0	1,13E-01
LUKSAN11LS	100	Gradiente	23042	2,12E-15	9,17E-08	0	1,95E+00
		SPG	3636	1,64E-21	8,10E-10	0	8,70E-02
		CG-descent	993	1,11E-15	8,50E-08	0	3,92E-02
LUKSAN12LS	98	Gradiente	50000	1,85E+03	8,21E-05	1	1,44E+01
		SPG	1013	2,06E+03	4,40E-08	0	3,66E-02
		CG-descent	988	4,25E+03	8,84E-08	0	7,09E-02
LUKSAN13LS	98	Gradiente	50000	2,52E+04	1,19E-04	1	1,65E+01
		SPG	316	2,52E+04	9,80E-08	0	9,39E-03
		CG-descent	165	2,52E+04	5,48E-08	0	1,36E-02
LUKSAN14LS	98	Gradiente	50000	1,25E+05	2,34E-06	1	1,30E+01
		SPG	495	1,25E+05	7,64E-08	0	1,27E-02
		CG-descent	159	1,25E+05	3,74E-08	0	8,19E-03
LUKSAN15LS	100	Gradiente	50000	3,57E+00	3,37E-07	1	1,75E+05
		SPG	34	3,57E+00	2,67E-08	0	1,85E-02
		CG-descent	31	3,57E+00	8,35E-08	0	2,92E-02
LUKSAN16LS	100	Gradiente	50000	3,57E+00	1,58E-07	1	3,88E+01
		SPG	38	3,57E+00	8,76E-08	0	3,11E-03
		CG-descent	35	3,57E+00	4,84E-08	0	1,03E-02
LUKSAN17LS	100	Gradiente	50000	4,93E-01	3,56E-06	1	7,48E+01
		SPG	523	4,93E-01	8,27E-08	0	8,03E-02
		CG-descent	729	4,93E-01	9,28E-08	0	3,46E-01
LUKSAN21LS	100	Gradiente	23045	6,65E-12	9,92E-08	0	8,04E-01
		SPG	2180	3,74E-14	3,88E-08	0	4,85E-02
		CG-descent	1898	6,02E-13	7,93E-08	0	6,36E-02
LUKSAN22LS	100	Gradiente	50000	8,73E+05	1,67E-03	1	6,62E+00
		SPG	50000	8,73E+05	1,08E-04	1	2,10E+00

Tabela 5 – continuação

Problema	n	Método	It	f	$\|\nabla f\|_\infty$	st	Tempo(s)
		CG-descent	50000	8,69E+02	3,50E-06	1	8,47E+00
MANCINO	100	Gradiente	140	1,37E-20	7,61E-08	0	3,88E+00
		SPG	13	1,80E-21	4,64E-08	0	4,63E-02
		CG-descent	12	1,42E-20	9,97E-08	0	5,95E-02
MNISTS0LS	494	Gradiente	1	1,18E+04	0,00E+00	0	2,67E-01
		SPG	1	1,18E+04	5,84E-09	0	9,92E-02
		CG-descent	1	1,18E+04	0,00E+00	0	1,12E-01
MNISTS5LS	494	Gradiente	1	1,08E+04	0,00E+00	0	2,68E-01
		SPG	1	1,08E+04	1,32E-08	0	9,88E-02
		CG-descent	1	1,08E+04	0,00E+00	0	1,11E-01
OSCIPATH	500	Gradiente	28	1,00E+00	3,18E-08	0	1,42E-02
		SPG	14	1,00E+00	3,20E-08	0	1,57E-03
		CG-descent	13	1,00E+00	4,83E-08	0	2,24E-03
PENALTY1	1000	Gradiente	4719	9,69E-03	4,88E-08	0	6,24E-01
		SPG	1549	9,69E-03	1,21E-09	0	1,14E+00
		CG-descent	45	9,69E-03	3,44E-08	0	1,13E-02
PENALTY2	200	Gradiente	50000	4,71E+13	2,60E+01	1	9,12E+01
		SPG	485	4,71E+13	7,20E-08	0	4,95E-02
		CG-descent	378	4,71E+13	8,04E-08	0	1,31E-01
PENALTY3	200	Gradiente	50000	9,98E-04	1,62E-01	1	3,82E+03
		SPG	104	1,00E-03	6,80E-08	0	8,24E-01
		CG-descent	50000	1,00E-03	3,43E-01	1	2,82E+00
QING	100	Gradiente	408	2,51E-16	8,88E-08	0	2,62E-02
		SPG	97	4,52E-17	7,72E-08	0	2,04E-03
		CG-descent	76	3,06E-16	9,30E-08	0	2,32E-03
SENSORS	100	Gradiente	50000	-2,11E+03	1,38E-06	1	6,08E+05
		SPG	241	-2,11E+03	5,42E-08	0	1,26E+00
		CG-descent	28	-2,11E+03	6,92E-08	0	1,63E-01
SPIN2LS	102	Gradiente	20	1,85E-16	8,25E-08	0	1,92E-02
		SPG	5385	1,17E-12	9,66E-08	0	1,84E+00
		CG-descent	60	7,08E-16	9,74E-08	0	3,50E-02

Tabela 5 – continuação

Problema	n	Método	It	f	$\|\nabla f\|_\infty$	st	Tempo(s)
TOINTGOR	50	Gradiente	50000	1,37E+03	3,99E-06	1	7,69E+00
		SPG	275	1,37E+03	9,45E-08	0	3,02E-03
		CG-descent	150	1,37E+03	6,29E-08	0	3,01E-03
TOINTPSP	50	Gradiente	50000	2,26E+02	8,05E-07	1	4,74E+00
		SPG	414	2,26E+02	8,34E-08	0	3,62E-03
		CG-descent	163	2,26E+02	9,70E-08	0	2,54E-03
TOINTQOR	50	Gradiente	196	1,18E+03	6,43E-08	0	1,14E-02
		SPG	70	1,18E+03	5,49E-08	0	6,05E-04
		CG-descent	37	1,18E+03	9,02E-08	0	6,30E-04
VARDIM	200	Gradiente	28	1,72E-20	5,24E-08	0	8,39E-03
		SPG	1	1,48E-31	4,44E-16	0	5,97E-05
		CG-descent	23	5,81E-20	9,64E-08	0	2,05E-03

Tabela 6: Resultados numéricos obtidos da aplicação dos métodos do Gradiente projetado e do Gradiente Espectral Projetado para os problemas com restrições de caixa propostos na Seção 4.3.1.

Problema	n	Método	It	f	$\|d\|_\infty$	st	Tempo(s)
DECONVB	63	GP	50000	1,01E-08	1,24E-05	1	8,12E+00
		SPG	50000	1,18E-08	5,77E-06	1	3,73E+00
POWELLBC	1000	GP	50000	3,10E+05	8,35E+03	1	2,61E+03
		SPG	50000	3,11E+05	2,51E-04	1	2,43E+03
CHEBYQAD	100	GP	50000	9,47E-03	7,69E-05	1	6,32E+05
		SPG	40694	9,47E-03	9,93E-08	0	2,66E+05
PROBPENL	500	GP	50000	3,99E-07	1,72E-07	1	2,78E+04
		SPG	50000	-5,91E-05	5,72E-05	1	1,50E+01
DIAGPQB	1000	GP	50000	-3,49E+05	2,65E+00	1	2,27E+04
		SPG	50000	-8,22E+05	2,42E-01	1	1,40E+01
GENROSEB	500	GP	50000	1,59E+03	3,04E+01	1	1,83E+01
		SPG	142	1,59E+03	3,55E-15	0	3,83E-02
DIAGIQT	1000	GP	50000	-3,46E+14	5,00E+07	1	6,84E+01

Tabela 6 – continuação

Problema	n	Método	It	f	$\|d\|_\infty$	st	Tempo(s)
		SPG	1378	-3,46E+14	9,88E-08	0	3,79E-01
BQPGABIM	50	GP	50000	-3,79E-05	9,74E-02	1	6,88E+00
		SPG	39	-3,79E-05	7,91E-08	0	1,17E-03
DIAGNQB	1000	GP	50000	-3,33E+15	1,00E+08	1	9,19E+00
		SPG	2	-3,33E+15	0,00E+00	0	5,64E-04
DIAGPQE	1000	GP	50000	-3,74E+00	9,78E-07	1	5,22E+01
		SPG	451	-3,74E+00	7,73E-08	0	1,26E-01
HADAMALS	400	GP	50000	7,17E+03	3,83E+04	1	1,38E+02
		SPG	481	7,16E+03	5,97E-08	0	0.357152
DIAGPQT	1000	GP	50000	-9,42E+01	9,03E-01	1	2,28E+01
		SPG	50000	-5,02E+05	1,70E-07	1	1,39E+01
DIAGNQT	1000	GP	50000	-1,67E+15	1,00E+08	1	9,13E+00
		SPG	2	-1,67E+15	0,00E+00	0	5,61E-04
BQPGASIM	50	GP	50000	-5,52E-05	9,59E-02	1	7,09E+00
		SPG	49	-5,52E-05	9,05E-08	0	1,44E-03
CHEBYQADNE	100	GP	0	0,00E+00	0,00E+00	0	9,48E-04
		SPG	0	0,00E+00	0,00E+00	0	9,38E-04
HARKERP2	1000	GP	50000	-5,00E-01	1,00E+00	1	2,21E+05
		SPG	63	-5,00E-01	0,00E+00	0	4,04E-01
HOLMES	180	GP	50000	1,25E+03	6,98E+00	1	1,18E+03
		SPG	88	1,25E+03	6,45E-09	0	4,10E-01
SINEALI	1000	GP	50000	-9,98E+04	8,69E+05	1	4,00E+01
		SPG	119	-7,90E+04	2,25E-08	0	5,35E-02
DIAGIQB	1000	GP	50000	-1,18E+15	5,00E+07	1	2,10E+01
		SPG	422	-1,18E+15	6,41E-08	0	1,15E-01
DIAGIQE	1000	GP	50000	-6,24E+14	4,99E+07	1	6,81E+01
		SPG	50000	-6,24E+14	5,23E+00	1	1,40E+01
DIAGNQE	1000	GP	50000	-2,50E+15	1,00E+08	1	9,14E+00
		SPG	2	-2,50E+15	0,00E+00	0	5,62E-04
QR3DLS	610	GP	50000	2,50E-01	2,87E-02	1	8,92E+01
		SPG	50000	6,31E-05	1,07E-02	1	5,65E+01

Tabela 7: Resultados numéricos obtidos pela aplicação dos métodos do SPG e CG-descent para o problema "quase quadrático" proposto na Seção 4.3.2.

Matriz	n	$\lambda_{\min}(A)$	Método	It	f	$\|\nabla f\|_\infty$	st	nf	ng	Tempo(s)
1138_bus	1138	3,52E-03	CG-descent	1373	1,12E-01	9,48E-05	0	11825	10940	2,71E+00
			SPG	1248	1,12E-01	8,33E-05	0	1463	1249	1,43E+00
494_bus	494	1,24E-02	CG-descent	5917	2,65E-01	8,68E-05	0	114201	110658	3,34E+00
			SPG	4907	2,65E-01	3,07E-05	0	6317	4908	2,06E-01
662_bus	662	5,05E-03	CG-descent	432	1,92E-01	7,72E-05	0	4569	4320	1,62E-01
			SPG	463	1,92E-01	8,54E-05	0	505	464	3,59E-02
685_ bus	685	6,19E-02	CG-descent	1693	2,26E-01	9,93E-05	0	24310	23190	9,69E-01
			SPG	2398	2,26E-01	7,70E-05	0	2667	2399	1,24E-01
bcsstk01	48	3,42E+03	CG-descent	15200	1,74E+03	9,43E-05	0	57081	49558	3,91E-01
			SPG	100000	1,74E+03	8,73E-04	1	13249084	100001	3,25E+01
bcsstk02	66	4,21E+00	CG-descent	605	5,03E+00	7,17E-05	0	24535	24384	4,38E-01
			SPG	218	5,03E+00	7,38E-05	0	224	219	7,77E-03
bcsstk03	112	2,94E+04	CG-descent	55483	1,47E+04	7,43E-05	0	2156632	2147019	2,91E+01
			SPG	100000	1,47E+04	1,42E+01	1	2734331	100001	1,80E+01
bcsstk04	132	4,21E+00	CG-descent	2995	5,03E+00	8,27E-05	0	80073	78773	1,76E+00
			SPG	6181	5,03E+00	9,99E-05	0	7372	6182	1,98E-01
bcsstk05	153	4,34E+02	CG-descent	10347	2,20E+02	9,32E-05	0	268587	264498	5,63E+00
			SPG	35238	2,20E+02	8,54E-05	0	42584	35239	1,11E+00
bcsstk06	420	4,61E+02	CG-descent	10947	2,32E+02	9,19E-05	0	35981	30433	1,40E+00
			SPG	100000	2,32E+02	1,03E-03	1	8156825	100001	1,83E+02
bcsstk07	420	4,61E+02	CG-descent	11492	2,32E+02	9,19E-05	0	37528	31656	1,46E+00
			SPG	100000	2,32E+02	1,14E+00	1	16628834	100001	3,65E+02
bcsstk08	1074	2,95E+03	CG-descent	78107	1,49E+03	9,01E-05	0	464571	427582	3,92E+01
			SPG	100000	1,49E+03	6,57E+00	1	4900983	100001	2,50E+02
bcsstk09	1083	7,10E+03	CG-descent	8494	3,55E+03	8,97E-05	0	463098	463027	3,73E+01
			SPG	248	3,55E+03	7,76E-05	0	255	249	2,67E-02
bcsstk10	1086	8,54E+01	CG-descent	28881	4,46E+01	9,48E-05	0	207540	189983	1,93E+01
			SPG	100000	4,46E+01	2,96E-02	1	122034	100001	1,18E+01
bcsstk11	1473	2,96E+00	CG-descent	100000	2,44E+00	6,40E-01	1	318192	253248	4,26E+01
			SPG	100000	2,46E+00	3,27E-01	1	122542	100001	1,70E+01
bcsstk12	1473	2,96E+00	CG-descent	100000	2,43E+00	1,22E+00	1	318648	254931	4,21E+01
			SPG	100000	2,46E+00	2,04E+00	1	123032	100001	1,70E+01
bcsstk13	2003	2,84E+02	CG-descent	100000	1,53E+02	1,03E+03	1	327691	256654	9,51E+01
			SPG	100000	1,53E+02	5,68E+01	1	120374	100001	3,61E+01
bcsstk14	1806	1,00E+00	CG-descent	100000	4,03E+00	1,76E+01	1	540110	453162	1,27E+02
			SPG	100000	8,62E+00	1,15E+01	1	122766	100001	3,04E+01
bcsstk15	3948	1,00E+00	CG-descent	100000	2,13E+01	1,56E+01	1	1412167	1349831	6,87E+02

Tabela 7 – continuação

Matriz	n	$\lambda_{\min}(A)$	Método	It	f	$\|\nabla f\|_\infty$	st	nf	ng	Tempo(s)
			SPG	100000	3,20E+01	1,06E+03	1	123104	100001	5,84E+01
bcsstk16	4884	1,00E+00	CG-descent	100000	2,24E+00	4,24E+00	1	663470	594217	5,29E+02
			SPG	100000	8,88E+00	8,90E+00	1	123147	100001	1,00E+02
bcsstk19	817	1,43E+03	CG-descent	100000	7,20E+02	4,74E+03	1	5846	3462	1,91E+00
			SPG	100000	1,31E+03	2,38E+03	1	119009	100001	6,33E+00
bcsstk20	485	3,24E+03	CG-descent	100000	1,70E+03	1,26E+08	1	30	26	1,44E+00
			SPG	100000	2,13E+03	4,43E+04	1	126898	100001	4,25E+00
bcsstk21	3600	7,21E+00	CG-descent	38111	4,21E+00	9,88E-05	0	128493	106106	3,02E+01
			SPG	85038	4,21E+00	1,00E-04	0	104645	85039	2,53E+01
bcsstk22	138	5,28E+01	CG-descent	7990	3,43E+01	8,65E-05	0	179035	174697	2,68E+00
			SPG	32265	3,43E+01	6,98E-05	0	41136	32266	7,66E-01
bcsstk23	3134	8,54E+03	CG-descent	100000	4,55E+03	1,08E+09	1	19	16	5,30E+00
			SPG	100000	7,85E+03	4,00E+05	1	120467	100001	3,26E+01
bcsstk24	3562	1,57E+02	CG-descent	100000	7,94E+01	6,92E+02	1	301750	212911	1,34E+02
			SPG	100000	5,86E+02	6,33E+03	1	122794	100001	5,57E+01
bcsstk26	1922	9,54E+02	CG-descent	73749	4,96E+02	9,53E-05	0	305888	267759	5,37E+01
			SPG	100000	4,96E+02	3,43E+01	1	13824924	100001	1,46E+03
bcsstk27	1224	1,44E+02	CG-descent	33413	7,69E+01	9,35E-05	0	857484	841983	1,48E+02
			SPG	92527	7,69E+01	7,51E-05	0	113215	92528	1,91E+01
bcsstk28	4410	8,14E-01	CG-descent	100000	5,06E-01	3,33E-01	1	282519	202619	1,64E+02
			SPG	100000	5,15E-01	7,38E-01	1	124479	100001	7,43E+01
bcsstm02	66	1,97E-02	CG-descent	677	4,17E+01	2,44E-06	0	33202	33168	2,51E-01
			SPG	56	4,17E+01	4,16E-06	0	180	57	2,20E-03
bcsstm05	153	7,31E-02	CG-descent	290	2,09E+01	7,15E-05	0	13786	13758	1,98E-01
			SPG	38	2,09E+01	2,17E-07	0	54	39	1,88E-03
bcsstm06	420	2,20E-03	CG-descent	123	6,25E+00	7,14E-05	0	1495	1414	5,84E-02
			SPG	82	6,25E+00	7,87E-05	0	103	83	7,00E-03
bcsstm07	420	3,30E-01	CG-descent	3382	1,13E+00	9,36E-05	0	101377	99674	3,64E+00
			SPG	1634	1,13E+00	5,79E-05	0	1897	1635	9,56E-02
bcsstm08	1074	1,75E-01	CG-descent	4	1,25E+02	3,10E-10	0	13	10	1,59E-03
			SPG	6	1,25E+02	2,12E-07	0	8	7	9,95E-04
bcsstm09	1083	2,59E-08	CG-descent	543	1,73E-01	8,71E-06	0	4545	4115	2,41E-01
			SPG	249	1,73E-01	2,94E-05	0	576	250	3,07E-02
bcsstm11	1473	2,67E-04	CG-descent	251	1,04E+01	1,79E-05	0	10218	10150	7,06E-01
			SPG	55	1,04E+01	2,80E-05	0	79	56	9,16E-03
bcsstm12	1473	2,12E-05	CG-descent	3554	1,12E-01	9,00E-05	0	29417	26938	3,04E+00
			SPG	1441	1,12E-01	5,25E-05	0	2626	1442	2,42E-01
bcsstm19	817	1,69E+02	CG-descent	1383	1,16E+02	9,92E-05	0	48686	48364	1,79E+00
			SPG	13251	1,16E+02	9,25E-05	0	16193	13252	7,39E-01

Tabela 7 – continuação

Matriz	n	$\lambda_{\min}(A)$	Método	It	f	$\|\nabla f\|_\infty$	st	nf	ng	Tempo(s)
bcsstm20	485	1,87E+02	CG-descent	1308	1,56E+02	7,89E-06	0	73189	73165	1,69E+00
			SPG	2415	1,56E+02	8,93E-05	0	2620	2416	7,74E-02
bcsstm21	3600	6,18E-06	CG-descent	472	1,04E-01	7,55E-06	0	2715	2326	4,91E-01
			SPG	161	1,04E-01	1,79E-05	0	241	162	4,77E-02
bcsstm22	138	1,03E-05	CG-descent	955	6,25E+01	3,30E-06	0	49036	49016	6,73E-01
			SPG	46	6,25E+01	9,15E-09	0	285	47	6,03E-03
bcsstm23	3134	8,74E-03	CG-descent	4	1,25E+02	1,31E-06	0	13	10	4,73E-03
			SPG	10	1,25E+02	6,79E-06	0	12	11	4,30E-03
bcsstm24	3562	5,62E-08	CG-descent	439	3,12E+01	6,93E-05	0	19296	19225	3,37E+00
			SPG	691	3,12E+01	7,70E-05	0	718	692	2,05E-01
bcsstm26	1922	5,78E-06	CG-descent	4	1,25E+02	3,21E-10	0	13	10	2,89E-03
			SPG	6	1,25E+02	2,12E-07	0	8	7	1,77E-03
gr_30_30	900	6,15E-02	CG-descent	203	2,22E-01	2,58E-05	0	1767	1624	1,05E-01
			SPG	76	2,22E-01	8,44E-05	0	114	77	1,02E-01
lund_a	147	8,00E+01	CG-descent	1407	4,49E+01	8,46E-05	0	5492	4746	1,16E-01
			SPG	3665	4,49E+01	9,21E-05	0	4229	3666	1,13E-01
lund_b	147	2,47E-01	CG-descent	1750	3,08E+00	9,50E-05	0	64069	63345	1,26E+00
			SPG	833	3,08E+00	6,22E-05	0	1160	834	2,98E-02
nos1	237	1,23E+02	CG-descent	3470	6,40E+01	9,76E-05	0	14583	13003	3,50E-01
			SPG	100000	6,40E+01	2,58E-04	1	122489	100001	3,80E+00
nos2	957	3,08E+01	CG-descent	100000	1,60E+01	2,03E+00	1	325014	272410	1,65E+01
			SPG	100000	1,60E+01	6,80E-02	1	1174904	100001	3,79E+01
nos3	960	1,83E-02	CG-descent	1109	3,35E-01	7,85E-05	0	23323	22685	1,71E+00
			SPG	644	3,35E-01	9,80E-05	0	684	645	5,71E-02
nos4	100	5,38E-04	CG-descent	185	1,40E+00	4,01E-05	0	3033	2915	4,07E-02
			SPG	94	1,40E+00	4,86E-05	0	134	95	3,44E-03
nos5	468	5,29E+01	CG-descent	6154	3,37E+01	9,69E-05	0	214163	211948	7,44E+00
			SPG	8883	3,37E+01	9,86E-05	0	11517	8884	4,35E-01
nos7	729	4,15E-03	CG-descent	12371	1,74E-01	8,67E-05	0	35105	24555	1,54E+00
			SPG	100000	1,75E-01	9,04E-03	1	122909	100001	5,33E+00
plat1919	1919	7,32E-07	CG-descent	3856	1,96E-02	9,85E-05	0	40878	38165	6,48E+00
			SPG	1837	1,95E-02	4,13E-05	0	3234	1838	4,65E-01
plat362	362	-4,53E-06	CG-descent	475	3,85E-01	1,44E-05	0	2774	2397	9,01E-02
			SPG	304	3,85E-01	7,70E-05	0	682	305	2,02E-02

Tabela 8: Resultados numéricos obtidos pela aplicação dos métodos CG-descent, SPG e L-BFGS para os problemas de regressão *lasso* da Seção 5.2.1.

Matriz	Método	It	f	$\|\nabla f\|_\infty$	st	nf	ng	Tempo(s)
Delor295K	CG_descent	50000	1,07E+02	6,62E-06	1	131029	84680	7,39E+04
	L-BFGS	50000	1,07E+02	1,56E-06	1	150000	150000	1,22E+05
	SPG	50000	1,07E+02	7,35E-05	1	60372	50001	3,54E+04
Delor338K	CG_descent	50000	3,15E+02	1,19E-02	1	104396	55492	3,09E+04
	L-BFGS	50000	3,15E+02	3,67E-03	1	153527	153527	6,00E+04
	SPG	50000	3,16E+02	8,90E-02	1	66085	50001	2,21E+04
Delor64K	CG_descent	39827	5,04E+01	8,88E-08	0	108201	73480	5,13E+04
	L-BFGS	20464	5,04E+01	9,75E-08	0	61387	61387	3,62E+04
	SPG	50000	5,04E+01	1,33E-03	1	58591	50001	3,39E+04
ESOC	CG_descent	50000	1,36E+05	9,94E+07	1	100001	50399	5,29E+03
	L-BFGS	50000	1,29E+05	5,60E+07	1	148848	148848	1,03E+04
	SPG	50000	2,82E+10	1,07E+09	1	61181	50001	3,84E+03
Maragal_1	CG_descent	33	3,44E+00	8,28E-08	0	71	39	5,34E-04
	L-BFGS	33	3,44E+00	2,67E-08	0	90	90	7,77E-04
	SPG	82	3,44E+00	6,27E-08	0	83	83	7,97E-04
Maragal_2	CG_descent	50000	1,33E+02	2,16E-05	1	135086	87915	2,30E+01
	L-BFGS	34702	1,33E+02	7,98E-08	0	103940	103940	2,22E+01
	SPG	50000	1,33E+02	1,90E-03	1	60976	50001	1,18E+01
Maragal_3	CG_descent	50000	3,08E+02	5,10E-07	1	139575	95628	1,01E+02
	L-BFGS	27348	3,08E+02	9,73E-08	0	82016	82016	7,87E+01
	SPG	50000	3,08E+02	5,22E-04	1	60473	50001	4,99E+01
Maragal_4	CG_descent	50000	2,46E+02	4,41E-07	1	139974	94483	1,61E+02
	L-BFGS	22252	2,46E+02	9,88E-08	0	66751	66751	1,05E+02
	SPG	50000	2,46E+02	1,62E-04	1	60473	50001	7,86E+01
Maragal_5	CG_descent	50000	6,34E+02	1,12E-04	1	100001	50002	4,97E+02
	L-BFGS	50000	6,34E+02	9,01E-05	1	149847	149847	1,38E+03
	SPG	50000	6,34E+02	1,26E-01	1	60630	50001	4,28E+02
Maragal_6	CG_descent	50000	3,03E+03	1,29E-06	1	142535	96457	1,25E+04
	L-BFGS	37663	3,03E+03	9,11E-08	0	112983	112983	1,47E+04
	SPG	50000	3,03E+03	1,91E-02	1	60664	50001	6,29E+03
Maragal_7	CG_descent	50000	6,63E+03	1,07E-05	1	111480	61936	2,30E+04
	L-BFGS	50000	6,63E+03	1,69E-05	1	149993	149993	5,47E+04

Tabela 8 – continuação

Matriz	Método	It	f	$\|\nabla f\|_\infty$	st	nf	ng	Tempo(s)
	SPG	50000	6,63E+03	7,79E-02	1	60634	50001	1,78E+04
Maragal_8	CG_descent	50000	4,85E+03	3,35E-03	1	100001	50002	3,60E+04
	L-BFGS	50000	4,85E+03	2,79E-03	1	149993	149993	1,19E+05
	SPG	50000	4,86E+03	3,10E-02	1	60995	50001	3,06E+04
Rucci1	CG_descent	571	5,39E+05	9,29E-08	0	1502	997	1,75E+02
	L-BFGS	482	5,39E+05	9,86E-08	0	1446	1446	1,93E+02
	SPG	2001	5,39E+05	9,99E-08	0	2065	2002	2,73E+02
abb313	CG_descent	371	6,28E-02	5,44E-08	0	743	373	5,55E-02
	L-BFGS	360	6,28E-02	6,01E-08	0	1073	1073	1,11E-01
	SPG	591	6,28E-02	7,04E-08	0	603	592	5,81E-02
ash219	CG_descent	25	4,25E-02	4,84E-08	0	51	27	1,99E-03
	L-BFGS	25	4,25E-02	4,83E-08	0	75	75	3,71E-03
	SPG	34	4,25E-02	5,91E-08	0	35	35	1,27E-03
ash292	CG_descent	1568	1,23E-01	9,84E-08	0	3137	1570	3,49E-01
	L-BFGS	1512	1,23E-01	9,14E-08	0	4533	4533	6,90E-01
	SPG	5490	1,23E-01	9,29E-08	0	6252	5491	9,14E-01
ash331	CG_descent	24	5,20E-02	5,77E-08	0	49	26	2,29E-03
	L-BFGS	23	5,20E-02	5,73E-08	0	69	69	4,22E-03
	SPG	33	5,20E-02	7,58E-08	0	34	34	1,98E-03
ash608	CG_descent	27	9,40E-02	6,19E-08	0	55	29	4,53E-03
	L-BFGS	27	9,40E-02	7,18E-08	0	81	81	8,57E-03
	SPG	38	9,40E-02	9,80E-08	0	39	39	3,84E-03
ash85	CG_descent	1219	9,99E-02	8,59E-08	0	2440	1222	8,76E-02
	L-BFGS	1702	9,99E-02	8,93E-08	0	5083	5083	2,44E-01
	SPG	8020	9,99E-02	4,46E-08	0	9162	8021	3,08E-01
ash958	CG_descent	27	1,46E-01	5,00E-08	0	55	29	6,04E-03
	L-BFGS	26	1,46E-01	8,96E-08	0	78	78	1,15E-02
	SPG	36	1,46E-01	6,45E-08	0	37	37	5,70E-03
illc1033	CG_descent	3341	1,19E-01	9,68E-08	0	6684	3343	9,18E-01
	L-BFGS	2899	1,19E-01	9,53E-08	0	8649	8649	1,59E+00
	SPG	50000	1,19E-01	1,55E-06	1	58901	50001	1,01E+01
illc1850	CG_descent	1807	4,57E-01	9,39E-08	0	3616	1809	1,07E+00
	L-BFGS	2313	4,57E-01	9,82E-08	0	6924	6924	2,52E+00

Tabela 8 – continuação

Matriz	Método	It	f	$\|\nabla f\|_\infty$	st	nf	ng	Tempo(s)
	SPG	50000	4,57E-01	3,03E-07	1	59265	50001	2,16E+01
landmark	CG_descent	1417	2,63E+00	9,85E-08	0	2869	1453	1,95E+01
	L-BFGS	1442	2,63E+00	9,47E-08	0	4315	4315	3,57E+01
	SPG	5736	2,63E+00	8,21E-08	0	6768	5737	5,25E+01
sls	CG_descent	4881	6,27E+01	5,93E-08	0	13512	9888	1,39E+03
	L-BFGS	1967	6,27E+01	6,54E-08	0	6154	6154	7,36E+02
	SPG	5918	6,27E+01	9,88E-08	0	7060	5919	7,68E+02
well1033	CG_descent	225	1,32E-01	9,83E-08	0	451	226	6,18E-02
	L-BFGS	226	1,32E-01	9,88E-08	0	649	649	1,19E-01
	SPG	1109	1,32E-01	9,03E-08	0	1138	1110	2,09E-01
well1850	CG_descent	197	4,72E-01	3,84E-08	0	395	198	1,15E-01
	L-BFGS	199	4,72E-01	9,02E-08	0	596	596	2,15E-01
	SPG	833	4,72E-01	6,30E-08	0	853	834	3,39E-01

Referências Bibliográficas

1 FRIEDLANDER, A. *Elementos de programação não-linear*. [S.l.]: Editora da UNICAMP, 1994.

2 LUENBERGER, D. G.; YE, Y. et al. *Linear and nonlinear programming*. [S.l.]: Springer, 1984.

3 BIRGIN, E. G.; MARTÍNEZ, J. M.; RAYDAN, M. Nonmonotone spectral projected gradient methods on convex sets. *SIAM Journal on Optimization*, SIAM, v. 10, n. 4, p. 1196–1211, 2000.

4 BOTTOU, L. Large-scale machine learning with stochastic gradient descent. In: *Proceedings of COMPSTAT'2010*. [S.l.]: Springer, 2010. p. 177–186.

5 HAGER, W. W.; ZHANG, H. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on optimization*, SIAM, v. 16, n. 1, p. 170–192, 2005.

6 HAGER, W. W.; ZHANG, H. Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent. *ACM Transactions on Mathematical Software (TOMS)*, ACM New York, NY, USA, v. 32, n. 1, p. 113–137, 2006.

7 NOCEDAL, J.; WRIGHT, S. *Numerical optimization*. [S.l.]: Springer Science & Business Media, 2006.

8 DOLAN, E. D.; MORÉ, J. J. Benchmarking optimization software with performance profiles. *Mathematical programming*, Springer, v. 91, n. 2, p. 201–213, 2002.

9 GOULD, N. I.; ORBAN, D.; TOINT, P. L. CUTEst: a Constrained and Unconstrained Testing Environment with safe threads for mathematical optimization. *Computational optimization and applications*, Springer, v. 60, n. 3, p. 545–557, 2015.

10 DAVIS, T. A.; HU, Y. The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, ACM New York, NY, USA, v. 38, n. 1, p. 1–25, 2011.

11 RIBEIRO, A.; KARAS, E. *Otimização contínua : aspectos teóricos e computacionais*. [S.l.]: Cengage Learning, 2013.

12 MARTINEZ, J. M.; SANTOS, S. A. Métodos computacionais de otimização. *Colóquio Brasileiro de Matemática, Apostilas*, Citeseer, v. 20, 1995.

13 GALÁNTAI, A. *Projectors and projection methods*. [S.l.]: Springer Science & Business Media, 2013.

14 GLOWINSKI, R. *Lectures on numerical methods for non-linear variational problems*. [S.l.]: Springer Science & Business Media, 2008.

15 HAGER, W. W.; ZHANG, H. A new active set algorithm for box constrained optimization. *SIAM Journal on Optimization*, SIAM, v. 17, n. 2, p. 526–557, 2006.

16 BIRGIN, E. G.; MARTÍNEZ, J. M. *Practical augmented Lagrangian methods for constrained optimization*. [S.l.]: SIAM, 2014.

17 BIRGIN, E. G.; MARTÍNEZ, J. M. Large-scale active-set box-constrained optimization method with spectral projected gradients. *Computational Optimization and Applications*, Springer, v. 23, n. 1, p. 101–125, 2002.

18 BIRGIN, E. G.; MARTíNEZ, J. M. Improving ultimate convergence of an augmented Lagrangian method. *Optimization Methods and Software*, v. 23, n. 2, p. 177–195, 2008.

19 GRIPPO, L.; LAMPARIELLO, F.; LUCIDI, S. A nonmonotone line search technique for newton's method. *SIAM Journal on Numerical Analysis*, SIAM, v. 23, n. 4, p. 707–716, 1986.

20 CHONG, E. K.; ZAK, S. H. *An introduction to optimization*. [S.l.]: John Wiley & Sons, 2004.

21 FLETCHER, R. *Practical methods of optimization*. [S.l.]: John Wiley & Sons, 2013.

22 POLAK, E.; RIBIERE, G. Note sur la convergence de méthodes de directions conjuguées. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, Dunod, Paris, v. 3, n. R1, p. 35–43, 1969.

23 FLETCHER, R.; REEVES, C. M. Function minimization by conjugate gradients. *The computer journal*, Oxford University Press, v. 7, n. 2, p. 149–154, 1964.

24 PERRY, A. *A class of conjugate gradient algorithms with a two-step variable metric memory*. [S.l.], 1977.

25 SHANNO, D. F. On the convergence of a new conjugate gradient algorithm. *SIAM Journal on Numerical Analysis*, v. 15, n. 6, p. 1247–1257, 1978. Disponível em: <https://doi.org/10.1137/0715085>.

26 BEZANSON, J. et al. Julia: A fresh approach to numerical computing. *SIAM Review*, SIAM, v. 59, n. 1, p. 65–98, 2017. Disponível em: <https://epubs.siam.org/doi/10.1137/141000671>.

27 HU, J. et al. Adaptive quadratically regularized newton method for riemannian optimization. *SIAM Journal on Matrix Analysis and Applications*, SIAM, v. 39, n. 3, p. 1181–1207, 2018.

28 ANDREANI, R. et al. Extension of the delayed weighted gradient method for the minimization of strongly convex functions. *Submetido*, 2021.

29 LIU, D. C.; NOCEDAL, J. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, Springer, v. 45, n. 1, p. 503–528, 1989.

30 NOCEDAL, J. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, v. 35, n. 151, p. 773–782, 1980.

31 JUNIOR, G. P. d. A. Avaliação do lasso e métodos alternativos em modelos de regressão logística. Universidade Federal de São Carlos, 2021.

32 FRENCH, C. W. The treynor capital asset pricing model. *Journal of Investment Management*, California, v. 1, n. 2, p. 60–72, 2003.

33 DEATON, A. et al. *Understanding consumption*. [S.l.]: Oxford University Press, 1992.

34 ALPAYDIN, E. *Introduction to Machine Learning, fourth edition*. MIT Press, 2020. (Adaptive Computation and Machine Learning series). ISBN 9780262043793. Disponível em: <https://books.google.com.br/books?id=tZnSDwAAQBAJ>.

35 TIBSHIRANI, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, Wiley Online Library, v. 58, n. 1, p. 267–288, 1996.

36 REFAEILZADEH, P.; TANG, L.; LIU, H. Cross validation, encyclopedia of database systems (edbs). *Arizona State University, Springer*, v. 6, 2009.

37 HASTIE, T.; TIBSHIRANI, R.; WAINWRIGHT, M. *Statistical learning with sparsity: the lasso and generalizations*. [S.l.]: Chapman and Hall/CRC, 2019.

38 WIERINGEN, W. N. van. *Lecture notes on ridge regression*. 2021.

39 KARIMI, S.; VAVASIS, S. Nonlinear conjugate gradient for smooth convex functions. *arXiv preprint arXiv:2111.11613*, 2021.

40 STEGLE, O.; TEICHMANN, S. A.; MARIONI, J. C. Computational and analytical challenges in single-cell transcriptomics. *Nature Reviews Genetics*, Nature Publishing Group, v. 16, n. 3, p. 133–145, 2015.

41 OLSEN, T. K.; BARYAWNO, N. Introduction to single-cell rna sequencing. *Current protocols in molecular biology*, Wiley Online Library, v. 122, n. 1, p. e57, 2018.

42 KHALFAOUI, B.; VERT, J.-P. Droplasso: A robust variant of lasso for single cell rna-seq data. *arXiv preprint arXiv:1802.09381*, 2018.

43 SONESON, C.; ROBINSON, M. D. Bias, robustness and scalability in differential expression analysis of single-cell rna-seq data. *bioRxiv*, Cold Spring Harbor Laboratory, p. 143289, 2017.

44 BERTSEKAS, D. P. Nonlinear programming. *Journal of the Operational Research Society*, Taylor & Francis, v. 48, n. 3, p. 334–334, 1997.