O método do gradiente espectral projetado e variantes para minimização com restrições convexas

Edital:	**Edital Piic 2022/2023**
Grande Área do Conhecimento (CNPq):	**Ciências Exatas e da Terra**
Área do Conhecimento (CNPq):	**Matemática / Matemática Aplicada**
Título do Projeto:	**Métodos Computacionais em Otimização (9403/2019)**
Título do Subprojeto:	**O método do gradiente espectral projetado e variantes para minimização com restrições convexas**
Professor Orientador:	**Leonardo Delarmelina Secchin**
Estudante:	**Pedro Henrique Fischer Ferreira**

Resumo

Uma das estratégias mais básicas para resolução de problemas de otimização com restrições convexas é vista no método do gradiente projetado. Este método é baseado na ideia de que, em uma função continuamente diferenciável, a direção de maior decréscimo é a contrária à de seu gradiente. Ainda que seja utilizado em implementações práticas, a minimização iterativa através de passos de gradiente pode se tornar significativamente lenta conforme o método aproxima-se da solução. Técnicas de aceleração incluem a estratégia de controle de tamanho de passo. Em especial, o método do gradiente espectral projetado usa um cálculo simples e barato do tamanho do passo que usa informações de segunda ordem provindas da equação secante, em que também se baseiam métodos Quasi-Newton de grande sucesso. O método do gradiente espectral projetado, tal como formalizado por Raydan em 1997 e Birgin, Martínez e Raydan em 2000, também faz uso da técnica de interpolação quadrática alinhada ao uso de uma busca linear inexata não-monótona para acelerar o cálculo do passo. Este método se mostrou muito eficiente para minimização de funções gerais, incluindo as de grande porte, com eficácia muito acima do método do gradiente tradicional, despertando a atenção da comunidade. Com isso, diversas variantes foram propostas na literatura, sendo objeto de pesquisa ainda nos dias atuais. Nesta pesquisa, foram considerados o método do gradiente espectral projetado e suas principais variantes, comparando-os do ponto de vista teórico e numérico, e exibindo os resultados por perfis de desempenho. Além disso, também é exibida uma adaptação do método do gradiente estocástico, amplamente utilizado em *machine learning*, que desfruta da técnica do gradiente espectral para o cálculo automático da taxa de aprendizagem.

Palavras-chave: Programação Não-Linear. Método do Gradiente e Variantes. Gradiente Espectral. Busca Linear Não-Monótona.

1 Introdução

Nesta pesquisa, serão considerados problemas de otimização com restrições convexas da forma

$$\min_{x\in\Omega} f(x), \tag{1}$$

onde $f : \mathbb{R}^n \to \mathbb{R}$ é uma função continuamente derivável e $\Omega \subset \mathbb{R}^n$ é um conjunto fechado, convexo e não vazio. O objetivo deste projeto é estudar o método do gradiente espectral projetado (do inglês, SPG) e suas variantes para resolver problemas do tipo (1). Estes métodos são extensões do método do gradiente projetado e são diferenciados pela construção de seus passos. No SPG, por exemplo, a construção do passo espectral agrega informações importantes da equação secante, no estilo Quasi-Newton, o que o dá vantagem em relação ao gradiente projetado.

No método SPG, tem-se a associação de duas estratégias: a busca linear não-monótona, desenvolvida por Grippo, Lampariello e Lucidi [1] para o método de Newton, e a escolha do tamanho do passo utilizando uma aproximação da equação secante, no estilo Quasi-Newtoniano, introduzida por Barzilai e Borwein e analisada por Raydan [2]. Birgin, Martínez e Raydan [3] descrevem a utilização de interpolação quadrática no cálculo do tamanho do passo, associada ao uso de salvaguardas, i.e., o passo da interpolação é rejeitado caso não esteja em um intervalo pré-especificado, dando lugar à uma estratégia denominada *backtracking*. Neste projeto, são consideradas duas formas de implementação do intervalo das salvaguardas: $[0.1t, 0.9t]$ e $[0.1, 0.9t]$. O método SPG foi implementado em ambas as versões e testado sobre problemas da biblioteca `CUTEst`, o que resultou na necessidade de comparação dos resultados obtidos. Isto levantou a importância de estudar um tópico relacionado à comparação de performance de algoritmos, introduzido a seguir.

Os perfis de desempenho, propostos por Dolan e Moré [4], são uma poderosa ferramenta para avaliar e comparar o desempenho de algoritmos em relação a uma determinada medida, tais como tempo de execução, número de iterações, avaliações de função etc. A construção e definição detalhada de tais perfis é feita adiante neste relatório. Em suma, a comparação foi feita com base no número de avaliações de f. Este critério é adequado pois reflete diretamente a eficiência do algoritmo de busca linear, ou seja, ter menos avaliações de f implica que a busca linear foi mais eficiente. Após a realização dos testes com o SPG, foram implementadas e testadas ainda algumas de suas variantes, incluindo o método de Barzilai-Borwein Adaptativo (ABB), sua versão alternativa ABBmin e o método de Gradientes Conjugados para Barzilai-Borwein (Dai-Kou). Mais detalhes sobre essas variantes podem ser encontrados em [5–7].

Além disso, no decorrer da pesquisa foram estudadas versões estocásticas do SPG, propostas em [8]. Em métodos tipo gradiente aplicados à *machine learning*, um dos parâmetros de maior importância é a taxa de aprendizado, que é tamanho do passo. Em consequência disto, diversas técnicas vêm sendo desenvolvidas para otimizar esse parâmetro. Uma prática comum é utilizar uma taxa de aprendizado decrescente, ou ajustar manualmente uma taxa fixa, o que pode consumir muito tempo na prática. Desse modo, foi proposta em [8] uma adaptação no cálculo da taxa de aprendizagem, que possui relação direta com o método SPG, obtendo resultados promissores.

2 Objetivos

O objetivo geral deste projeto é comparar os métodos do gradiente espectral projetado e algumas de suas variantes. Os objetivos específicos são: estudo/revisão dos conceitos fundamentais de otimização sem restrições e dos métodos do gradiente e Quasi-Newton secantes; estudo dos principais artigos científicos sobre o tema, em especial, [5–7,9]; implementação e testes numéricos a fim de comparar os diferentes métodos considerados na pesquisa.

No decorrer da pesquisa objetivou-se ainda o estudo do método de Barzilai-Borwein aplicado ao aprendizado de máquina conforme proposto em [8].

3 Embasamento Teórico

Nesta seção, são apresentados os conceitos fundamentais do método do gradiente espectral projetado e de suas variantes, além da prova de convergência do SPG. Também apresenta-se o conceito básico do método do gradiente estocástico para *machine learning* e sua adaptação com a taxa de aprendizado tipo SPG.

3.1 Método do Gradiente Espectral Projetado

Considere um problema do tipo (1). Estamos interessados em garantir a viabilidade dos pontos gerados, ou seja, garantir que x_{k+1} seja viável. Para isso, projetamos sobre o conjunto Ω. A projeção de um ponto $y \in \mathbb{R}^n$ em Ω é o ponto $P_\Omega(y) \in \Omega$ mais próximo à y, ou seja, $x^* = P_\Omega(y)$ é a solução do problema

$$\min_x \tfrac{1}{2}\|x-y\|^2 \text{ s.a. } x \in \Omega. \tag{2}$$

O teorema a seguir garante a existência e unicidade da projeção em Ω. Ele decorre diretamente do fato do problema acima ser convexo com função objetivo estritamente convexa.

Teorema 3.1. *Se $\Omega \neq \emptyset$ é convexo e fechado, então $P_\Omega(y)$ está bem definida, ou seja, a projeção existe e é única para cada $y \in \mathbb{R}^n$.*

Neste trabalho, consideramos Ω um conjunto que chamamos de "caixa", definido como $\Omega = \{x \in \mathbb{R}^n; l_i \leq x_i \leq u_i, \forall i\}$. Assim, a projeção sobre Ω é dada por $[P_\Omega(y)]_i = \max\{l_i, \min\{u_i, y_i\}\}$, para $i = 1, \ldots, n$. O critério de parada é a solução KKT para o problema (2), que pode ser facilmente identificado como $P_\Omega(x^* - \nabla f(x^*)) - x^* = 0$. Desse modo, a direção tomada no problema é $d_k = P_\Omega(x_k - g(x_k)) - x_k$, onde $g(x_k) = \nabla f(x_k)$. Este critério funciona para parar tanto na borda de Ω, quanto em seu interior.

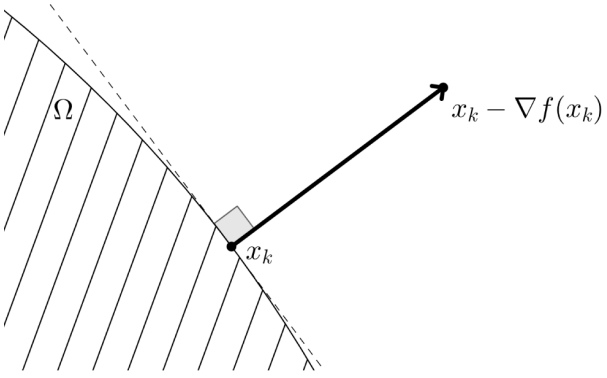

Figura 1: Projeção de $x_k - \nabla f(x_k)$ sobre o conjunto Ω igual a x_k. Fonte: Produção do próprio autor.

Uma iteração típica dos métodos Quasi-Newton para resolver (1) é da forma $x_{k+1} = x_k - B_k^{-1} g(x_k)$, onde B_k é uma aproximação da matriz Hessiana de f no iterando x_k. Além disso, B_k deve satisfazer a equação secante, dada por $B_k s_k = y_k$, onde $s_k = x_k - x_{k-1}$ e $y_k = g(x_k) - g(x_{k-1})$, para $k \geq 1$. Neste trabalho, a escolha de B_k é dada por

$B_k = \sigma_k I$, $\sigma_k > 0$, que, como não satisfaz a equação secante sempre, tomamos σ_k como solução de

$$\min_{\sigma_k} \tfrac{1}{2}\|\sigma_k s_k - y_k\|^2 \tag{3}$$

(ou seja, minimizamos o resíduo da equação secante). Isto nos leva a escolher $\sigma_k = (s_k^t y_k)/(s_k^t s_k)$, $s_k \neq 0$. Daí, $B_k^{-1} = \frac{1}{\sigma_k} I$. Desta forma, definimos o *Passo Espectral* como λ_k, dado por

$$\lambda_k = \frac{1}{\sigma_k} = \frac{s_k^t s_k}{s_k^t y_k}, \tag{4}$$

onde $s_{k-1}^t y_{k-1} > 0$. Caso $s_{k-1}^t y_{k-1} \leq 0$, (4) será negativo ou não definido e, neste caso, tomamos $\lambda_k = \lambda_{\max}$ onde $\lambda_{\max} > 0$ é um parâmetro. O passo $x_{k+1} = x_k - t_k \lambda_k g(x_k)$, com $\lambda_k \in [\lambda_{\min}, \lambda_{\max}]$ ($\lambda_{\min} > 0$ é parâmetro), converge globalmente se $t_k > 0$ é calculado por uma busca linear inexata tipo Armijo. Porém, se $t_k < 1$ estamos descartando o passo espectral λ_k, que contém informações valiosas da equação secante. Em outras palavras, gostaríamos de $t_k = 1$ com frequência, mesmo que f aumente. Para isso, $t_k > 0$ deve satisfazer a condição de Armijo "relaxada":

$$f(x_k - t_k \lambda_k g(x_k)) \leq f_{\max} - t_k \eta \lambda_k \|g(x_k)\|^2, \tag{5}$$

onde $f_{\max} = \max\{f(x_k), f(x_{k-1}), \ldots, f(x_{k-M})\}$, para $M \geq 1$ (note que a condição de Armijo usual é com $f_{\max} = f(x_k)$, o que força f decrescer sempre). Desse modo, a busca linear feita com (5) é não-monótona, pois permite que f cresça eventualmente. Este efeito pode ser visualizado na Figura 2.

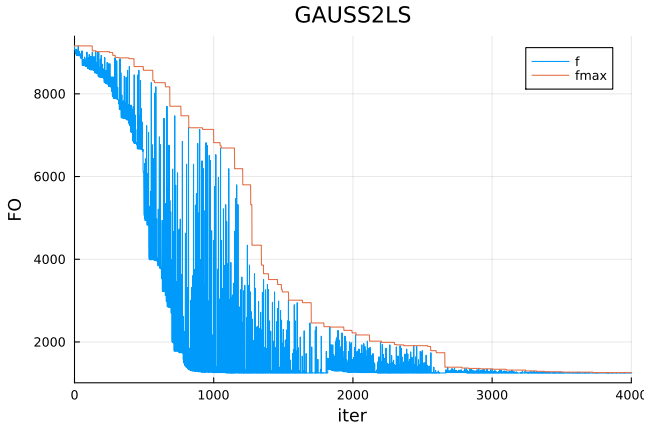

Figura 2: SPG executado no problema GAUSS2LS da `CUTEst`. Com a busca linear não-monótona, o valor da função objetivo oscila ao longo dos iterandos. Mesmo assim, o problema é resolvido. Fonte: Produção do próprio autor.

O Algoritmo a seguir representa o funcionamento do método do gradiente espectral projetado.

Algoritmo 1 Método do gradiente espectral projetado

Entrada: Ponto inicial $x_0 \in \Omega$, inteiro $M \geq 1$, $\lambda_{\max} > \lambda_{\min} > 0$, $\lambda_0 \in [\lambda_{\min}, \lambda_{\max}]$, parâmetros da salvaguarda $0 < \sigma_1 < \sigma_2 < 1$, parâmetro de decréscimo $\eta \in (0,1)$, critério de parada $\varepsilon > 0$.

enquanto $\|P(x_k - g(x_k)) - x_k\| > \varepsilon$ **faça**
  $d_k = P(x_k - \lambda_k g(x_k)) - x_k, \quad t_k = 1.$
  $x_{novo} = x_k + t_k d_k.$
  enquanto $f(x_{novo}) > f_{\max} + \eta t_k g(x_k)^t d_k$ **faça**
    Compute $t_{quad} = -\frac{1}{2} t_k^2 g(x_k)^t d_k / (f(x_{novo}) - f(x_k) - t_k g(x_k)^t d_k)$, como descrito em [3].
    se $t_{quad} \in [\sigma_1 t_k, \sigma_2 t_k]$ **então**
      $t_k \leftarrow t_{quad}.$
    senão
      $t_k \leftarrow t_k/2.$
    fim se
    $x_{k+1} = x_k - t_k \lambda_k g(x_k).$
  fim enquanto
  $s_k = x_{k+1} - x_k, \quad y_k = g(x_{k+1}) - g(x_k)).$
  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
fim enquanto

3.2 Convergência do Método

Os resultados a seguir provam o bom comportamento do Algoritmo 1.

Lema 3.2. *Para todo $x \in \Omega$ e $\lambda \in (0, \lambda_{\max}]$ temos*

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

Demonstração: Sejam $v_k = x_k - \lambda \nabla f(x_k)$ e $p_k = P_\Omega(v_k) = d_k + x_k$. Como Ω é convexo e fechado, temos $(x - p)^t (v - p) \leq 0$. Note que $(x_k - p_k)^t = (x_k - (d_k + x_k))^t = -(d_k)^t$ e $v_k - p_k = x_k - \lambda \nabla f(x_k) - (d_k + x_k) = -\lambda \nabla f(x_k) - d_k$. Daí,

$$(x_k - p_k)^t (v_k - p_k) = -(d_k)^t(-\lambda \nabla f(x_k) - d_k) \Rightarrow \lambda \nabla f(x_k)^t d_k + \|d_k\|^2 \leq 0 \Rightarrow \nabla f(x_k)^t d_k \leq -\frac{1}{\lambda}\|d_k\|^2.$$

Observando que $-\frac{1}{\lambda} \leq -\frac{1}{\lambda_{\max}}$, temos $\nabla f(x_k)^t d_k \leq -\frac{1}{\lambda}\|d_k\|^2 \leq -\frac{1}{\lambda_{\max}}\|d_k\|^2$. □

O próximo teorema foi adaptado de [9].

Teorema 3.3. *O Algoritmo 1 está bem definido e qualquer ponto de acumulação da sequência $\{x_k\}$ gerada pelo método é um ponto estacionário.*

Demonstração: Seja $p_\lambda(x) = P_\Omega(x - \lambda\nabla f(x)) - x$, $d_k = p_{\lambda_k}(x_k)$, $m(k) = \min\{k, M-1\}$. Se x_k não é um ponto estacionário, então pelo lema (3.2),

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

e a direção de busca é uma direção de descida. Portanto, um tamanho de passo satisfazendo (5) será encontrado em um número finito de tentativas e o método está bem definido. Seja $x^* \in \Omega$ um ponto de acumulação de $\{x_k\}$ e renomeie $\{x_k\}$ como uma subsequência que converge para x^*. Consideraremos os dois casos a seguir.

CASO 1: Assuma que $\inf t_k = 0$. Suponha, por contradição, que x^* não seja um ponto estacionário. Pela continuidade e compacidade, existe $\delta > 0$ tal que, para todo $\lambda \in [\lambda_{\min}, \lambda_{\max}]$,

$$\nabla f(x^*)^t \frac{p_\lambda(x^*)}{\|p_\lambda(x^*)\|} \leq -\delta \Rightarrow \nabla f(x_k)^t \frac{p_\lambda(x_k)}{\|p_\lambda(x_k)\|} \leq -\frac{\delta}{2}, \tag{6}$$

e todo k suficientemente grande em $\{x_k\}$. Como $\inf t_k = 0$, existe uma subsequência $\{x_k\}_K$ tal que $\lim_{k\in K} t_k = 0$. Neste caso, da forma que t_k é escolhido em (5), existe um índice $\bar{k}$ suficientemente grande tal que para todo $k \geq \bar{k}, k \in K$, existe $\bar{t_k} \in [\alpha_1, \alpha_2]$ que falha em satisfazer (5), i.e., $f(x_k + \bar{t_k} d_k) > f_{\max} + \eta \bar{t_k} \nabla f(x_k)^t d_k \geq f(x_k) + \eta \bar{t_k} \nabla f(x_k)^t d_k$. Daí,

$$\frac{f(x_k + \bar{t_k} d_k) - f(x_k)}{\bar{t_k}} > \eta \nabla f(x_k)^t d_k. \tag{7}$$

Seja $\varphi(t) = f(x_k + t d_k)$. Pelo teorema do valor médio, existe um $t_k \in (0, \bar{t_k})$, tal que

$$\varphi'(t_k) = \frac{\varphi(\bar{t_k}) - \varphi(0)}{\bar{t_k} - 0} \Rightarrow \nabla f(x_k + t_k d_k)^t d_k = \frac{f(x_k + \bar{t_k} d_k) - f(x_k)}{\bar{t_k}}.$$

Então, podemos reescrever (7) como

$$\nabla f(x_k + t_k d_k)^t d_k > \eta \nabla f(x_k)^t d_k, \tag{8}$$

para todo $k \in K, k \geq \bar{k}$, onde $k \to \infty \Rightarrow t_k \to 0$. Tome uma subsequência conveniente tal que $d_k/\|d_k\|$ converge para d. Dividindo ambos os lados de (8) por $\|d_k\|$ e passando o limite, $\nabla f(x^*)^t d > \eta \nabla f(x^*)^t d \Rightarrow (1-\eta)\nabla f(x^*)^t d \geq 0$. De fato, a sequência $\{\|d_k\|\}_K$ é limitada e, assim, $t_k \|d_k\| \to 0$. Como $(1-\eta) > 0$ e $\nabla f(x^*)^t d < 0$ para todo k, segue que $\nabla f(x^*)' d = 0$.

Pela continuidade e definição de d_k, isto implica que para k suficientemente grande naquela subsequência temos

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

o que contradiz (6). Logo, x^* é estacionário.

CASO 2: Assuma que $\inf t_k \geq \rho > 0$. Suponha, por contradição, que x^* não seja um ponto estacionário. Portanto, $\|p_t(x^*)\| > 0$ para todo $t \in (0, \lambda_{\max}]$. Por continuidade e compacidade, existe $\delta > 0$ tal que $\|p_t(x^*)\| \geq \delta > 0$, para todo $t \in [\rho, \lambda_{\max}]$. Seja $l(k)$ um inteiro tal que

$$k - m(k) \leq l(k) < k, \quad f(x_{l(k)}) = \max_{0 \leq j \leq m(k)} [f(x_{k-j})], \tag{9}$$

onde $\{f(x_{l(k)})\}$ é uma sequência monótona não crescente. De fato, sabendo que $m(k+1) \leq m(k)+1$, temos

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

Além disso, de (5), para $k > M$, obtemos

$$\begin{aligned} f(x_{l(k)}) &= f(x_{l(k)-1} + t_{l(k)-1}d_{l(k)-1}) \\ &\leq \max_{0\leq j\leq m(l(k)-1)}[f(x_{l(k)-1-j})] + \eta t_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1} \\ &= f(x_{l(l(k)-1)} + \eta t_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1}. \end{aligned} \tag{10}$$

Agora, já que $f(x_k) \leq f(x_0)$ para todo k, $\{x_k\} \subset \Omega$ tal que $\{f(x_{l(k)})\}$ admite limite para $k \to \infty$. Dado que $t_k > 0$ e $\nabla f(x_k)^t d_k < 0$, segue de (10) que

$$\lim_{k\to\infty} t_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1} = 0 \tag{11}$$

Do lema (3.2), temos $t_k\nabla f(x_k)^t d_k \leq -\frac{t_k}{\bar{\lambda}}\|d_k\|^2 \leq -\frac{t_k}{\lambda_{\max}}\|d_k\|^2$, para todo k, e como $t_k < \bar{t}$, (11) implica

$$\lim_{k\to\infty} t_{l(k)-1}\|d_{l(k)-1}\| = 0. \tag{12}$$

Provaremos agora que $\lim_{k\to\infty} t_k\|d_k\| = 0$. Seja $\hat{l}(k) \equiv l(k+M+2)$. Provaremos por indução que dado qualquer $j \geq 1$, temos

$$\lim_{k\to\infty} t_{\hat{l}(k)-j}\|d_{\hat{l}(k)-j}\| = 0 \tag{13}$$

e

$$\lim_{k\to\infty} f(x_{\hat{l}(k)-j}) = \lim_{k\to\infty} f(x_{l(k)}). \tag{14}$$

(Aqui e na sequência assumimos, sem perda de generalidade, que o índice k é suficientemente grande para evitar a ocorrência de índices negativos, i.e., $k \geq j-1$). Se $j = 1$, como $\{\hat{l}(k)\} \subset \{l(k)\}$, de (12), segue que (13) é válido. Isto implica que $\|x_{\hat{l}(k)} - x_{\hat{l}(k)-1}\| \to 0$, de modo que (14) valha para $j = 1$, já que $f(x)$ é uniformemente contínua em Ω. Assuma agora que (13) e (14) valem para um dado j. Então, por (10),

$$f(x_{\hat{l}(k)-j}) \leq f(x_{l(\hat{l}(k)-j-1)}) + \eta t_{\hat{l}(k)-j-1}\nabla f(x_{\hat{l}(k)-j-1})d_{\hat{l}(k)-j-1}.$$

Tomando limite para $k \to \infty$, de (14) temos $\lim_{k\to\infty} t_{\hat{l}(k)-(j+1)}\nabla f(x_{\hat{l}(k)-(j+1)})d_{\hat{l}(k)-(j+1)} = 0$. Usando os mesmo argumentos que resultaram (12), $\lim_{k\to\infty} t_{\hat{l}(k)-(j+1)}\|d_{\hat{l}(k)-(j+1)}\| = 0$. Isso implica que $\|x_{\hat{l}(k)-j} - x_{\hat{l}(k)-(j+1)}\| \to 0$, tal que de (14) e da continuidade uniforme de f em Ω, temos $\lim_{k\to\infty} f(x_{\hat{l}(k)-(j+1)}) = \lim_{k\to\infty} f(x_{\hat{l}(k)-j}) = \lim_{k\to\infty} f(x_{l(k)})$. Concluímos então que (13) e (14) valem para todo $j \geq 1$. Agora, para qualquer k, temos $x_{\hat{l}(k)} = x_{k+1} + \sum_{j=1}^{\hat{l}(k)-k-1} t_{\hat{l}(k)-j}d_{\hat{l}(k)-j}$, ou seja,

$$x_{k+1} = x_{\hat{l}(k)} - \sum_{j=1}^{\hat{l}(k)-k-1} t_{\hat{l}(k)-j}d_{\hat{l}(k)-j}. \tag{15}$$

Por (9), temos $\hat{l}(k) - k - 1 = l(k+M+2) - k - 1 \leq M+1$, de modo que (15), por (13), implica $\lim_{k\to\infty}\|x_{k+1} -$

$x_{\hat{l}(k)}\| = 0$. Como $\{f(x_l(k))\}$ admite limite, segue, da continuidade de f em Ω, que

$$\lim_{k\to\infty} f(x_k) = \lim_{k\to\infty} f(x_{\hat{l}(k)}) = \lim_{k\to\infty} f(x_{l(k)}) = f(x^*). \tag{16}$$

Por continuidade, para $k > \bar{k}$ suficientemente grande, $\|p_{\lambda_k}(x_k)\| \geq \frac{\delta}{2}$. Usando (10) e o lema (3.2), obtemos

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

Quando $k \to \infty$, claramente $f(x_{l(k)}) \to -\infty$, que é uma contradição, tendo em vista (16) e a continuidade de f em x^*. Portanto, x^* é estacionário. □

3.3 Variantes do Método SPG

Ao resolver o problema (3), obtemos $\lambda_k = \lambda_k^{BB1} = (s_k^t s_k)/(s_k^t y_k)$. Contudo, outra escolha de λ_k pode ser obtida resolvendo

$$\min_{\sigma_k} \tfrac{1}{2}\|\sigma_k^{-1} y_k - s_k\|^2, \tag{17}$$

o que nos leva a escolher $\lambda_k = \lambda_k^{BB2} = (s_k^t y_k)/(y_k^t y_k)$. De acordo com [5], se a relação $\lambda_k^{BB2}/\lambda_k^{BB1}$ for menor que uma constante $\kappa \in (0,1)$, houve uma pequena redução em $\|g(x_k)\|_2$. Portanto devemos escolher um passo mais controlado, dado por λ_k^{BB2}. Caso contrário, escolhe-se λ_k^{BB1} para um passo mais agressivo. Deste modo, no método ABB, λ_k é escolhido da seguinte maneira:

$$\lambda_k = \begin{cases} \lambda_k^{BB2}, & \text{se } \lambda_k^{BB2}/\lambda_k^{BB1} < \kappa, \\ \lambda_k^{BB1}, & \text{caso contrário.} \end{cases} \tag{18}$$

Com o objetivo de melhorar esta escolha de λ_k, no método denominado de ABBmin, temos a seguinte adaptação de (18):

$$\lambda_k = \begin{cases} \min\{\lambda_j^{BB2}; j = \max\{1,k-m\},\ldots,k\}, & \text{se } \lambda_k^{BB2}/\lambda_k^{BB1} < \kappa, \\ \lambda_k^{BB1}, & \text{caso contrário.} \end{cases}$$

Para mais detalhes sobre o porquê desta escolha, consulte [6]. Por fim, no método de gradientes conjugados desenvolvido por Dai e Kou, a direção d_k é escolhida da seguinte maneira (para simplificar a notação, considere $g_k = g(x_k)$):

$$d_k = \mu_k g_k + \nu_k s_{k-1},$$

onde

$$\mu_k = \frac{1}{\Delta_k}\left(g_k^t y_{k-1} g_k^t s_{k-1} - s_{k-1}^t y_{k-1} g_k^t g_k\right), \quad \nu_k = \frac{1}{\Delta_k}\left(g_k^t y_{k-1} g_k^t g_k - \rho_k g_k^t s_{k-1}\right), \tag{19}$$

e

$$\Delta_k = \rho_k s_{k-1}^t y_{k-1} - \left(g_k^t y_{k-1}\right)^2 > 0. \tag{20}$$

De fato, (19) e (20) são obtidos resolvendo um problema de minimização de uma função quadrática aproximada - veja [7] para detalhes. A relação deste método com o SPG está na construção de ρ_k, que carrega informações do passo espectral. Para incorporar esta ideia, é proposto aproximar a Hessiana B_k por $(1/\lambda_k^{BB1})I$ ou $(1/\lambda_k^{BB2})I$ para

a estimativa $\rho_k \approx g_k^t B_k g_k$. Isto leva as seguintes escolhas de ρ_k:

$$\rho_k^{BB1} = \frac{s_{k-1}^t y_{k-1}}{s_{k-1}^t s_{k-1}} g_k^t g_k \tag{21}$$

e

$$\rho_k^{BB2} = \frac{y_{k-1}^t y_{k-1}}{s_{k-1}^t y_{k-1}} g_k^t g_k. \tag{22}$$

Um ponto importante a ser destacado sobre a fórmula (22) é que, se $s_{k-1}^t y_{k-1} > 0$, a relação (20) é sempre satisfeita, a menos de quando y_{k-1} e g_k são colineares. Estudos numéricos apontam que (21) tem uma performance superior a (22). Apesar disso, é introduzido um parâmetro $\omega_k \geq 1$ em (22), obtendo

$$\rho_k^{BB3} = \omega_k \frac{y_{k-1}^t y_{k-1}}{s_{k-1}^t y_{k-1}} g_k^t g_k. \tag{23}$$

Também por experimentos numéricos, é visto que a escolha de ρ_k como em (23) com $\omega_k = 3/2$ performa melhor que ρ_k como em (21). Portanto, esta foi a opção adotada neste trabalho.

3.4 Adaptação do Método SPG para *Machine Learning*

Recentemente, o aprendizado de máquina emergiu como uma das principais ferramentas computacionais. As redes neurais vêm sendo utilizadas para os mais variados problemas, dentre eles, podemos citar a classificação de imagens, reconhecimento de ações, segmentação semântica, etc. Nesse contexto, o aprendizado de máquina possibilita que redes neurais artificiais identifiquem padrões complexos em uma grande quantidade de dados.

Uma vez que a arquitetura de uma rede neural foi construída para resolver um problema específico, a próxima tarefa é ensinar/treinar os pesos da rede. No então chamado aprendizado supervisionado, os pesos são ajustados com base em dados de entrada (amostras de treinamento) tal que o erro R entre a saída da rede e a saída real seja minimizado. Mais especificamente, denotaremos como (W,b) os conjuntos de pesos e vieses de uma rede neural. Considere um conjunto de treinamento $\{(x_i,y_i)\}_{i=1}^M$ contendo M amostras, onde x_i denota o dado de entrada e y_i denota o dado de saída real. Devemos definir uma função de predição h, cuja qualidade é medida contando o número de respostas erradas, i.e., $h(x_i) \neq y_i$. O processo de aprendizado, então, estima os melhores valores para (W,b) que minimizam a seguinte função de custo:

$$\min_{W,b} R_M(W,b) = \frac{1}{M} \sum_{i=1}^{M} L_i(W,b). \tag{24}$$

onde $L_i(W,b) = L(h(x_i;W,b);y_i)$. Usaremos $L = (h-y)^2$, chamada de função de perda. Uma escolha comum para h é

$$h(x;W,b) = W^t x + b, \tag{25}$$

por funcionar bem em vários problemas. Usaremos (24) e (25) neste trabalho. Para avaliar $h(x)$, usamos a técnica comumente chamada de *feedfoward*, que basicamente é uma aplicação sucessiva de funções (composição) nos dados de entrada.

Um método efetivo e eficiente para encontrar uma boa solução para (24) é essencial para o sucesso do aprendizado. Contudo, geralmente o problema (24) é de larga-escala não-suave e não-convexo. Por isso, métodos de primeira ordem, tal como o método de descida do gradiente, são preferencialmente escolhidos. Entre eles, o método de descida do gradiente estocástico (SGD) é de longe o mais predominante no treinamento de redes neurais. Ao invés de utilizar o gradiente completo de todas as amostras de treinamento, os métodos clássicos de SGD usam apenas o gradiente de uma amostra pequena a cada passo. A abordagem que geralmente é utilizada é a de mini-lotes, que usa uma pequena porção das amostras de treinamento para uma estimativa do gradiente. Assim, o passo do gradiente com mini-lote segue como:

$$(W,b)_{k+1} = (W,b)_k - \frac{\lambda_k}{|B_k|} \sum_{i \in B_k} \nabla_{(W,b)} L_i((W,b)_k), \tag{26}$$

onde $(W,b)_k$ é a estimativa na iteração k, B_k denota o índice do conjunto das amostras escolhidas aleatoriamente do conjunto de dados de treinamento na iteração k, $|B_k|$ denota a cardinalidade do conjunto B_k, e o valor $\lambda_k > 0$ é chamado de *taxa de aprendizado*. A estratégia dos mini-lotes é uma das práticas mais importantes do treinamento da rede. Na prática, $\nabla_{(W,b)} L_i(W,b)$ é calculado usando uma técnica chamada *backpropagation* (ou retro-propagação). Quando treinando uma rede neural, a taxa de aprendizado λ_k é indiscutivelmente um dos hiper-parâmetros mais importantes para atingir boa performance, portanto requer uma calibração rigorosa.

O método do gradiente estocástico é o mais utilizado para treinamento de redes neurais. Nele, temos um problema do tipo

$$\min_x f(x), \tag{27}$$

onde f é convexa. A iteração do método é dada por $x_{k+1} = x_k - t_k g_k$, onde $t_k > 0$ é o tamanho do passo e g_k é uma estimativa aleatória de $\nabla f(x_k)$. Note que a estimativa de g_k deve ser feita de maneira uniforme pois, deste modo, garantimos que cada amostra tem probabilidade igual (i.e., $P(i) = \frac{1}{M}, \forall i$) de ser selecionada e não teremos viés na escolha. Desse modo, x_{k+1} está condicionado apenas à escolha de x_k e g_k. A seguir, no Algoritmo 2, está o pseudo-código do método em questão.

Algoritmo 2 Método do gradiente estocástico

Entrada: máximo de épocas J, passos por época K, pesos iniciais $(W,b)_{1,1}$, taxa de aprendizado λ_1, fator de decrescimento $t \in (0,1)$.

para $j \leftarrow 1$ até J **faça**
  para $k \leftarrow 1$ até K **faça**
    Construa um lote $B_{j,k}$ aleatoriamente de maneira uniforme
    $(W,b)_{j,k+1} \leftarrow (W,b)_{j,k} - \lambda_j \nabla L_{B_{j,k}}((W,b)_{j,k})$
  fim para
  $\lambda_{j+1} \leftarrow t\lambda_j$
  $(W,b)_{j+1,1} \leftarrow (W,b)_{j,K}$
fim para

Seja $\nabla L_B(W,b)$ o gradiente do mini-lote usado no treinamento de redes neurais:

$$\nabla L_B(W,b) = \frac{1}{|B|}\sum_{i\in B}\nabla_{(W,b)}L_i(W,b). \tag{28}$$

Seja j o índice da época e k o índice do passo dentro de cada *época*, de 1 a K. Ao final da época, os dados são atualizados da seguinte maneira:

$$(W,b)_{j,1} = (W,b)_{j-1,K}, \qquad (W,b)_{j,k+1} = (W,b)_{j,k} - \lambda_j \nabla L_{B_{j,k}}((W,b)_{j,k}), \tag{29}$$

para $k = 1,2,\ldots,K$ e $j = 1,2,\ldots,J$. O gradiente para estimar λ_j é definido do seguinte modo:

$$g_{j,k+1} = (1-\beta)g_{j,k} + \beta \nabla L_{B_k}((W,b)_{j,k}), \tag{30}$$

para $k = 1,2,\ldots,K$ e $g_{j,1} = 0$, onde β é uma constante pré-definida entre $[0,1]$, que controla e suaviza o decaimento exponencial, efeito este que é chamado de momento. Assim, definimos a diferença dos gradientes de duas épocas por $y_j = g_{j,K} - g_{j-1,K}$. Já a diferença entre os pontos de duas épocas é definido como a diferença das duas últimas amostras das épocas normalizada pelas iterações: $s_j = \frac{1}{K}((W,b)_{j,K} - (W,b)_{j-1,K})$.

Desse modo, temos a taxa de aprendizado adaptativa:

$$\lambda_{j+1} = \frac{s_j^t s_j}{|s_j^t y_j|}. \tag{31}$$

Observe que tomamos o módulo de $s_j^t y_j$, pois como estamos tratando apenas com uma amostra do gradiente dos dados, este produto pode ser negativo, inviabilizando o passo. É demonstrado em [10] que, se uma hipótese usual de "não enviesamento" vale e se e a taxa de aprendizado satisfaz

$$\sum_{j=1}^{\infty}\lambda_j = \infty \quad \text{e} \quad \sum_{j=1}^{\infty}(\lambda_j)^2 < \infty, \tag{32}$$

então $\lim_{j\to\infty}\nabla L((W,b)_j) = 0$, i.e, o método converge caso λ_j satisfaça (32). Sendo assim, para garantir convergência do método com passo espectral, uma condição suficiente é deixar a sequência de taxas de aprendizado satisfazer (32). Portanto, em [8] é proposta a seguinte salvaguarda para λ_j:

$$\lambda_j = \begin{cases} \lambda_j, \text{ se } \lambda_j \in [\tau_{\min}/j, \tau_{\max}/j] \\ \tau_0/j, \text{ caso contrário,} \end{cases} \tag{33}$$

onde j é o índice da época, $\tau_{\min}$, $\tau_{\max}$ e τ_0 são constantes pré-definidas. Claramente, a taxa de aprendizado em (33) satisfaz a condição (32). Veja o Algoritmo 3 para a descrição do método SPG para taxa de aprendizado adaptativa.

Algoritmo 3 Taxa de aprendizado adaptativa baseada no método SPG

Entrada: máximo de épocas J, passos por época K, tamanho do lote $|B|$, parâmetro de peso $\beta \in (0,1]$, pesos iniciais $(W,b)_{1,1}$, taxa de aprendizado $\lambda_1 = \lambda_2$, τ_0, $\tau_{\min}$ e $\tau_{\max}$.

para $j \leftarrow 1$ até J **faça**
  se $j > 2$ **então**
    $s_j \leftarrow \frac{1}{K}\left((W,b)_{j,K} - (W,b)_{j-1,K}\right), \quad y_j \leftarrow g_{j,K} - g_{j-1,K}$
    $\lambda_j = \begin{cases} \bar{\lambda} = (s_j^t s_j)/(|s_j^t y_j|), \text{ se } \bar{\lambda} \in [\tau_{\min}/j, \tau_{\max}/j] \\ \tau_0/j, \text{ caso contrário.} \end{cases}$
  fim se
  $g_{j,1} \leftarrow 0$
  para $k \leftarrow 1$ até K **faça**
    Construa um lote $B_{j,k}$ aleatoriamente de maneira uniforme
    $(W,b)_{j,k+1} \leftarrow (W,b)_{j,k} - \lambda_j \nabla L_{B_{j,k}}((W,b)_{j,k})$
    $g_{j,k+1} \leftarrow (1-\beta) g_{j,k} + \beta \nabla L_{B_{j,k}}((W,b)_{j,k})$
  fim para
  $(W,b)_{j+1,1} \leftarrow (W,b)_{j,K}$
fim para

4 Metodologia

A metodologia utilizada para o desenvolvimento deste trabalho consistiu em revisar conceitos fundamentais de otimização irrestrita e métodos do gradiente e Quasi-Newton secantes, utilizando notas de aula digitais, o livro de Karas e Ribeiro [11] e outros materiais de apoio. Foram realizadas reuniões semanais com o orientador para acompanhar o desenvolvimento da pesquisa. Para introduzir os principais artigos científicos sobre o tema, foram considerados [5–7, 9]. Com base nestes artigos e na documentação do Julia, os códigos das buscas lineares e dos quatro algoritmos foram desenvolvidos e testados preliminarmente com problemas irrestritos selecionados da biblioteca `CUTEst`. Após a análise dos códigos, os testes numéricos finais da primeira parte do estudo foram realizados no servidor de otimização disponibilizado pela UFES (FAPES 116/2019) com problemas irrestritos e restritos da biblioteca `CUTEst`. O tratamento da informação incluiu a análise dos resultados dos testes e a avaliação da eficiência dos algoritmos.

Foram selecionados 441 problemas restritos da biblioteca `CUTEst`, cujo número máximo de restrições ordinárias ($h(x) = 0$ e $g(x) \leq 0$) é igual a zero. Isto é, dentre os problemas restritos, somente aqueles com restrições de caixa ($l \leq x \leq u$) foram considerados. Contudo, devido a erros fatais na interface, os seguintes problemas foram removidos durante a análise: BLEACHNG, PRICE4, BA-L16LS, BA-L21LS, BA-L49LS, BA-L52LS, BA-L73LS, JIMACK e RAYBENDS. Assim, o procedimento estatístico realizado neste trabalho consistiu na construção dos perfis de desempenho gerados com base nos 432 problemas restantes.

Para a seção de *machine learning*, o *dataset* utilizado para os testes foi o MNIST (fornecido pelo pacote MLDatasets.jl). Este *dataset* possui 60 000 dados de treinamento e 10 000 dados de teste. Tais dados são imagens de

28×28 pixels, em escala de cinza, que representam os números de 0 a 9. No artigo referência [8], são apresentados os valores dos parâmetros iniciais. O problema tratado, portanto, é o de classificar imagens de entrada entre 0 e 9.

5 Resultados e Discussão

Um dos objetivos da pesquisa era comparar o desempenho de duas versões da busca linear não-monótona e avaliar seu comportamento em diferentes problemas. A versão denominada busca original, apresenta o intervalo de salvaguarda definido como $[0.1, 0.9t]$ (usada na implementação oficial do SPG, fornecido pelo projeto TANGO em Fortran). Já na versão denominada busca modificada, o intervalo é definido como $[0.1t, 0.9t]$.

Análises anteriores (consulte [3]) mostraram que, na busca original, quando a interpolação tende a rejeitar 90% do intervalo de busca original ($[0,1]$), era considerado que sua previsão não era confiável e logo a estratégia conservadora de *backtracking* era usada. Este procedimento se mostrou mais eficiente do que a busca modificada. Contudo, nossos resultados indicaram o oposto. A busca modificada, mais "frouxa" (por tender a aceitar o passo de interpolação mais vezes) devido à presença do fator no extremo esquerdo do intervalo da salvaguarda, apresentou um desempenho ligeiramente superior quando testada sobre a `CUTEst`. É provável que essa diferença seja resultado das diferenças na linguagem de programação e computador utilizados nos testes. De qualquer forma, o ganho é marginal.

A Figura 3 apresenta a comparação dos resultados obtidos nas duas versões da busca linear não-monótona (à esquerda) e o perfil de desempenho dos quatro algoritmos, baseado no número de avaliações de f, para os problemas avaliados (à direita). A escala do eixo τ é logarítmica para melhor visualização próximo de $\tau = 1$.

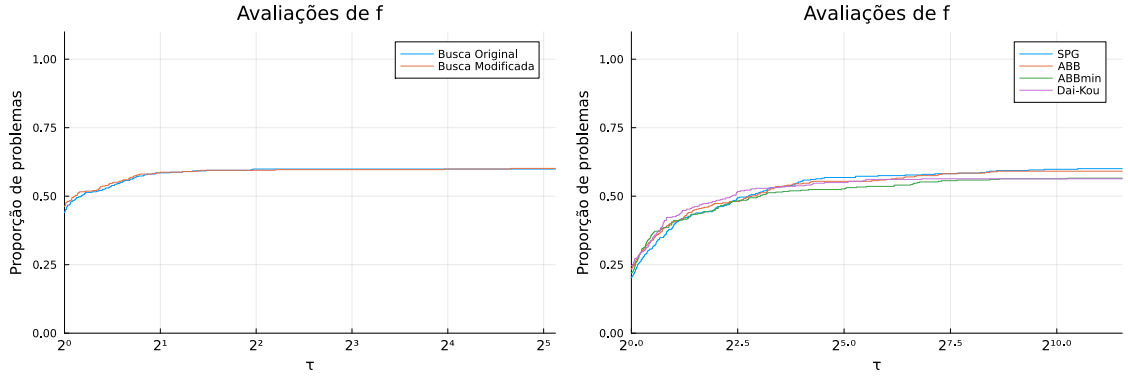

Figura 3: Comparação entre as buscas original e modificada (à esquerda) e comparação de desempenho dos algoritmos SPG, ABB, ABBmin e Dai-Kou (à direita). Fonte: Produção do próprio autor.

Do total de 432 problemas testados, a busca original resolveu 260, enquanto a busca modificada resolveu 261. Isso demonstra a semelhança de robustez de ambas buscas (i.e., resolveram praticamente a mesma quantidade de problemas). Contudo, a busca modificada foi mais eficiente nos testes. Em virtude disso, esta foi selecionada para utilização nos algoritmos da Figura 3 (à direita).

Em relação aos outros algoritmos avaliados, Dai-Kou apresentou maior eficácia em aproximadamente 50% dos problemas avaliados, mas apresentou a menor robustez, ou seja, foi o método que resolveu menos problemas. Por

outro lado, SPG apresentou pouca eficiência, mas foi o algoritmo mais robusto (resolveu a maior quantidade de problemas). Já os algoritmos ABB e ABBmin apresentam eficácia semelhante, mas o ABB foi consideravelmente mais robusto. Este é um destaque negativo para o método ABBmin, já que seu objetivo era melhorar seu predecessor, ABB.

Agora, abordaremos os resultados comparativos entre os métodos do gradiente estocástico (denominado SGD) e o método SPG para aprendizado adaptativo (denominado de BB), com e sem momento. Os parâmetros de referência são os seguintes: $\beta = 4/K$, $\tau_0 = 1$, $\tau_{\min} = 0.00001$ e $\tau_{\max} = 3$. O tamanho do mini-lote é 128 e $\lambda_1 = 0.1$. Finalmente, o número de neurônios da camada interna (ou camada oculta) é 20. Serão feitas comparações entre quatro algoritmos principais: SGD sem momento, SGD com momento, BB sem momento e BB com momento. Foram realizadas cadeias de 5 testes numéricos para cada algoritmo, registrando seus dados em um *DataFrame*.

Os dados exibidos a seguir mostram os resultados de perda e a porcentagem de acertos sobre os dados de teste da melhor rodada de cada algoritmo.

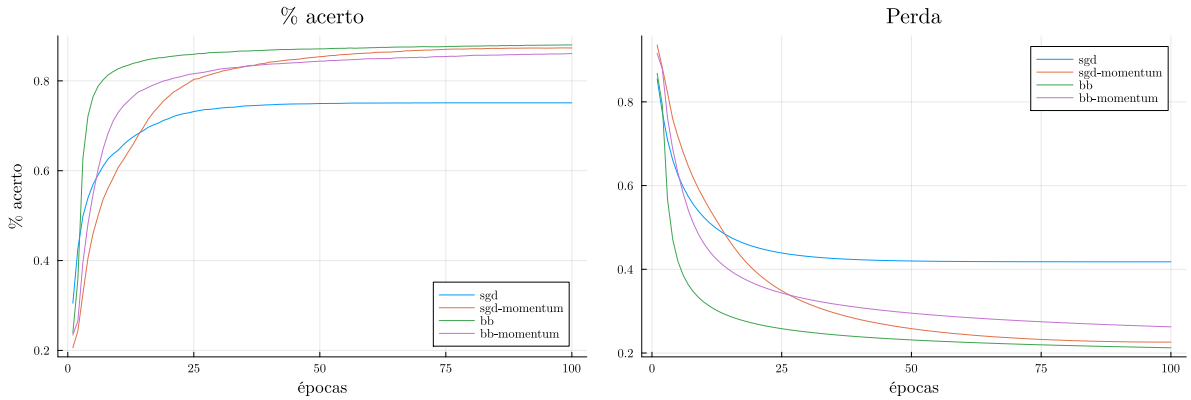

Figura 4: Porcentagem de acertos (à esquerda) e Risco Empírico (à direita).

Na figura, fica nítido que o método SGD sem momento é muito inferior. Os demais obtiveram resultados próximos, embora o método BB tenha se sobressaído. Além disso, nota-se uma clara melhora no desempenho dos algoritmos com momento, visto a performance do SGD com tal.

6 Conclusões

Neste trabalho, foi realizada a implementação do método do Gradiente Espectral Projetado (SPG) e de algumas de suas variantes, tais como o método de Barzilai-Borwein Adaptativo (ABB), sua versão alternativa ABBmin e o método de Gradientes Conjugados tipo Barzilai-Borwein (Dai-Kou). No método SPG, foram comparadas duas formas de implementação da salvaguardas, onde destas, a busca denominada modificada se sobressaiu. Após os testes com as diferentes salvaguardas, foram realizados testes numéricos entre os quatro algoritmos citados. Baseado nos resultados destes testes, concluímos que o método SPG é, no que diz respeito ao número de avaliações de função, menos eficiente comparado aos outros, porém ligeiramente mais robusto (i.e., resolveu mais problemas). Já com Dai-Kou é o contrário: é superior em eficácia, mas menos robusto, ficando atrás de todos os algoritmos nesse quesito. Além disso, ABB e ABBmin apresentam resultados semelhantes em eficiência, mas ABB é mais

robusto.

Finalmente, foi realizada a implementação do algoritmo do Gradiente Estocástico (SGD) para aprendizado de máquina, assim como uma versão alternativa que utiliza a ideia de Barzilai-Borwein (BB) para automatizar o processo do cálculo da taxa de aprendizagem. Nos testes realizados, a diferença nos resultados é nítida quando é utilizada a taxa de aprendizado utilizando a ideia de BB no algoritmo.

Este trabalho de iniciação científica servirá de base para o trabalho de conclusão de curso, onde pretende-se realizar mais testes no contexto do aprendizado de máquina.

Agradecimentos

O autor gostaria de agradecer todo o suporte e comentários valiosos recebidos pelo Orientador. Este trabalho teve o apoio da FAPES.

Referências Bibliográficas

[1] L. Grippo, F. Lampariello, and S. Lucidi, "A nonmonotone line search technique for Newton's method," *SIAM Journal of Numerical Analysis*, vol. 4, pp. 299–312, 2008.

[2] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, pp. 141–148, 1988.

[3] E. G. Birgin, J. M. Martínez, and M. Raydan, "Algorithm 813: SPG—software for convex-constrained optimization," *ACM Transactions on Mathematical Software*, vol. 27, pp. 340–349, 2001.

[4] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, pp. 201–213, 2002.

[5] B. Zhou, L. Gao, and Y.-H. Dai, "Gradient methods with adaptive step-sizes," *Computational Optimization and Applications*, vol. 35, pp. 69–86, 2006.

[6] G. Frassoldati, L. Zanni, and G. Zanghirati, "New adaptive stepsize selections in gradient methods," *Journal of Industrial and Management Optimization*, vol. 4, pp. 299–312, 2008.

[7] Y.-H. Dai and C.-X. Kou, "A Barzilai-Borwein conjugate gradient method," *Science China Mathematics*, vol. 59, pp. 1511–1524, 2016.

[8] J. Liang, Y. Xu, C. Bao, Y. Quan, and H. Ji, "Barzilai–Borwein-based adaptive learning rate for deep learning," *Pattern Recognition Letters*, vol. 128, pp. 197–203, 2019.

[9] E. G. Birgin, J. M. Martínez, and M. Raydan, "Nonmonotone spectral projected gradient methods on convex sets," *SIAM Journal on Optimization*, vol. 10, no. 4, pp. 1196–1211, 2000.

[10] A. Beck, *First-Order Methods in Optimization*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2017.

[11] E. W. Karas and A. A. Ribeiro, *Otimização Contínua - Aspectos Teóricos e Computacionais*. Cengage, 2014.