Universidade Federal do Espírito Santo

Centro Universitário Norte do Espírito Santo

Curso de Bacharelado em Matemática Industrial

Pedro Henrique Fischer Ferreira

O MÉTODO DO GRADIENTE ESPECTRAL PROJETADO E APLICAÇÕES AO APRENDIZADO DE MÁQUINA SUPERVISIONADO

São Mateus

2024

Pedro Henrique Fischer Ferreira

O MÉTODO DO GRADIENTE ESPECTRAL PROJETADO E APLICAÇÕES AO APRENDIZADO DE MÁQUINA SUPERVISIONADO

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 Industrial.

São Mateus

2024

Pedro Henrique Fischer Ferreira

O MÉTODO DO GRADIENTE ESPECTRAL PROJETADO E APLICAÇÕES AO APRENDIZADO DE MÁQUINA SUPERVISIONADO

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 Industrial.

Aprovada em 01 de Outubro de 2024.

Comissão Examinadora

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

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

Profa. Ms. Karyne Alves Zampirolli
Universidade Federal do Espírito Santo

Agradecimentos

Agradeço, primeiramente, à Deus, por ter me dado forças para concluir essa jornada até o fim.

À minha mãe, Marinete, e avó, Olinda, que me apoiaram mesmo de tão longe.

Aos meus amigos João Pedro, Douglas e Johny, por estarem comigo desde o início.

Ao meu orientador, prof. Leonardo D. Secchin, que não desistiu de mim e incentivou a conclusão do trabalho em meio a dificuldades pessoais.

À todos os professores que transmitiram seus conhecimentos de maneira primorosa e que foram fundamentais no meu desenvolvimento acadêmico.

À minha mulher, Érica, por todo o apoio e cobrança que me incentivaram a concluir este trabalho.

Ao meu "irmão", Gabriel, pois, não fosse ele, eu não estaria onde estou hoje.

Resumo

Neste trabalho, são apresentados métodos computacionais para resolução de problemas de otimização irrestrita e com restrições de caixa. A principal direção tomada é a abordagem do método do Gradiente Espectral Projetado, do inglês, *Spectral Projected Gradient* (SPG), que baseia-se na ideia de que, dada uma função continuamente diferenciável, a direção de maior decréscimo é a contrária à de seu gradiente. Tomando nota de que a minimização iterativa através de passos de gradiente pode tornar-se 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 SPG usa um cálculo simples e barato do tamanho do passo que desfruta de informações de segunda ordem provindas da equação secante, em que também são baseados métodos Quasi-Newton de grande sucesso. Além disso, no SPG estudado emprega-se uma busca linear não-monótona com interpolação quadrática para acelerar o cálculo do passo. Estes fatores tornam esse método muito eficiente para minimização de funções gerais, incluindo as de grande porte, com eficácia muito acima do método do gradiente tradicional. Como consequência, diversas variantes foram e vêm sendo desenvolvidas na literatura. Três delas são abordadas: os métodos *Adaptive Barzilai-Borwein* (ABB), seu sucessor ABBmin e o método de Gradientes Conjugados de Dai-Kou. O principal objetivo deste trabalho é realizar uma comparação abrangente entre esses métodos, tanto do ponto de vista teórico como numérico, exibindo os resultados por perfis de desempenho. Ademais, diante da grande visibilidade do método SPG, versões estocásticas do mesmo vêm sendo desenvolvidas para resolver problemas da área de *machine learning*. Como objetivo adicional, este trabalho explora esse tema, apresentando a fundamentação teórica e a motivação para o uso dessa técnica. Os resultados obtidos a partir de testes práticos em um *dataset* amplamente reconhecido na literatura são promissores, oferecendo fortes incentivos para a aplicação dessa abordagem em diferentes áreas.

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

Lista de Figuras

Sumário

Introdução

O desenvolvimento de métodos computacionais eficientes para a resolução de problemas de otimização tem sido um foco constante da comunidade acadêmica, especialmente à medida que esses problemas se tornam mais complexos com os avanços tecnológicos e a crescente disponibilidade de dados. Métodos que exigem cálculos de derivadas de segunda ordem, por serem de alto custo computacional, em geral são impraticáveis. Assim, a criação de métodos eficientes para esses desafios é de extrema importância.

Os métodos de descida pelo gradiente são conhecidos por seu baixo custo computacional e simplicidade. Entre esses métodos, destaca-se o Gradiente Espectral Projetado, do inglês, *Spectral Projected Gradient* (SPG) [1], que se notabiliza por empregar uma estratégia Quasi-Newtoniana na atualização do passo, mantendo um custo computacional reduzido por iteração. Sua eficiência e simplicidade atraíram grande interesse na comunidade acadêmica, impulsionando o desenvolvimento de novos métodos.

Entre esses, destaca-se o método *Adaptive Barzilai-Borwein* (ABB) [2], que alterna entre dois passos Quasi-Newtonianos com base na qualidade do iterando, preservando as características de simplicidade e baixo custo computacional do método original. Um método relacionado é o ABBmin [3], que utiliza um histórico de passos anteriores para otimizar a escolha dos passos Quasi-Newtonianos. Além disso, o método de gradientes conjugados proposto por Dai e Kou [4] combina as ideias clássicas de gradientes conjugados com o conceito do SPG.

Outra aplicação para métodos de otimização pode ser vista na área de aprendizado de máquina, onde o desenvolvimento de métodos eficazes desempenha um papel crucial no avanço dessa área. Dentro deste contexto, o uso do método SPG surge como uma abordagem promissora, combinando elementos da otimização com técnicas adaptativas para lidar com a dificuldade do cálculo da taxa de aprendizagem. Assim, são explorados conceitos como redes neurais, momento e probabilidade para apresentar uma versão estocástica do SPG voltada para o aprendizado de máquina supervisionado (treinamento de redes neurais). Especificadamente, é considerada a versão estocástica do SPG descrita em [5].

O principal objetivo desta pesquisa é comparar, tanto do ponto de vista teórico quanto numérico, os métodos abordados, avaliando seu desempenho em problemas consolidados da literatura. Além disso, busca-se realizar uma análise comparativa entre o método estocástico mais básico, o Gradiente Estocástico, do inglês, *Stochastic Gradient Descent* (SGD), e a versão estocástica do SPG, de modo a observar o comportamento dessas variações em diferentes cenários. Estas comparações são feitas por meio de *perfis de desempenho*.

Os perfis de desempenho são uma abordagem visual simples proposta por Dolan e Moré [6]. Tais perfis facilitam a comparação do desempenho dos métodos em problemas selecionados, permitindo determinar a eficiência de cada um com base em alguma medida de interesse, por exemplo, o tempo de execução, o número de iterações ou o número de avaliações da função objetivo.

Antes de adentrar nos detalhes dos métodos que serão apresentados, é essencial conhecer e compreender alguns conceitos básicos de otimização. No Capítulo 1, é discutido o problema de otimização alvo, algumas condições para otimalidade, alguns dos fundamentos da otimização convexa, métodos de gradiente e projeção sobre conjuntos convexos, fornecendo a estrutura teórica necessária para a compreensão dos métodos.

Uma análise detalhada dos métodos de otimização abordados neste trabalho é realizada no Capítulo 2, com ênfase no método do Gradiente Espectral Projetado, que servirá de base para os outros métodos. São exploradas a teoria e ideias desses métodos, justificando a escolha deles para este trabalho.

No Capítulo 3, são apresentados elementos do aprendizado de máquina supervisionado, começando com uma introdução às redes neurais e ao método do gradiente incremental. Em seguida, é discutido o método do gradiente estocástico para treinamento de redes neurais e, por fim, aborda-se os métodos com momento. Este capítulo estabelece a base teórica necessária para a aplicação dos conceitos no capítulo seguinte.

No Capítulo 4, apresenta-se uma versão estocástica do método do gradiente espectral para o treinamento de redes neurais desenvolvida em [5].

No Capítulo 5, são exibidos os resultados de testes numéricos realizados sobre problemas selecionados da biblioteca CUTEst [7] (do inglês, *Constrained and Unconstrained Testing Environment with safe threads*), uma biblioteca padrão de problemas-teste muito utilizada na literatura. Também são avaliados o desempenho e a eficácia do método estocástico proposto por meio de experimentos numéricos sobre o *dataset* MNIST [8]. Este capítulo fornece uma análise prática dos resultados obtidos e sua aplicabilidade no

treinamento de redes neurais.

Por fim, no Capítulo de Conclusões, são descritos os principais resultados obtidos neste trabalho.

1 Conceitos Fundamentais

Neste capítulo, serão apresentados alguns conceitos básicos em otimização contínua, a saber, minimizadores locais e globais, convexidade, projeção e condições de otimalidade. Algumas referências para este capítulo são [9–12].

1.1 O problema de otimização

Neste trabalho, será considerado o seguinte problema de otimização:

$$\begin{aligned} \text{minimizar} \quad & f(x), \\ \text{sujeito a} \quad & x \in \Omega. \end{aligned} \tag{1.1}$$

A função $f : \mathbb{R}^n \to \mathbb{R}$, considerada continuamente diferenciável ao longo do trabalho, é chamada *função objetivo* e o conjunto $\Omega \subset \mathbb{R}^n$ é denominado *conjunto factível*. Os casos de interesse são $\Omega = \mathbb{R}^n$ e o conjunto denominado como "caixa", representado pelo conjunto a seguir:

$$\Omega = \{x \in \mathbb{R}^n \mid l \leq x \leq u\}, \tag{1.2}$$

onde l e u são vetores. Em outras palavras, a "caixa" $l \leq x \leq u$, é o conjunto de todos os pontos $x \in \mathbb{R}^n$ cujas componentes satisfazem $l_i \leq x_i \leq u_i$, para todo $i = 1, \ldots, n$.

Os pontos de Ω serão os *pontos factíveis* de (1.1). As soluções $x_* \in \Omega$ de (1.1) e seus respectivos valores funcionais, $f(x_*)$, serão chamadas de, respectivamente, *minimizadores* e *mínimos* do problema.

Definição 1.1. *Dizemos que o ponto x_* é* minimizador local *de* (1.1) *se existe $\delta > 0$, tal que $f(x_*) \leq f(x)$ para todo $x \in \Omega$, com $\|x - x_*\| \leq \delta$. Se $f(x_*) \leq f(x)$ para todo $x \in \Omega$, dizemos que o ponto x_* é* minimizador global *de* (1.1).

Pelo Teorema de Bolzano-Weierstrass [9], o problema (1.1) possui solução se o conjunto Ω for compacto, isto é, toda sequência $\{x_k\} \in \Omega$ admite uma subsequência convergente. Em particular, uma caixa definida por limites finitos (ou seja, $-\infty < l_i \leq u_i < \infty$ para

todo i), é um conjunto compacto. Portanto, sendo Ω como em (1.2), a função f do problema (1.1) admitirá minimizador global. No caso $\Omega = \mathbb{R}^n$, um minimizador estará garantido caso f seja, por exemplo, coerciva, isto é, quando $f(x) \to \infty$ quando $\|x\| \to \infty$.

Tendo definido o problema de otimização geral e o caso particular da "caixa", vamos agora explorar o conceito de convexidade, que desempenha um papel fundamental na otimização, especialmente na garantia da existência e unicidade de soluções ótimas.

1.2 Conjuntos e funções convexas

A convexidade é uma propriedade fundamental em otimização, pois, como veremos, garante que qualquer minimizador local de uma função convexa em um conjunto convexo é também um minimizador global. Essa propriedade é explorada em diversos algoritmos de otimização. As definições e teoremas a seguir têm como referência [9, 10].

Definição 1.2. *Um conjunto $C \subset \mathbb{R}^n$ é dito* convexo *se para quaisquer $x, y \in C$ e $t \in [0,1]$, qualquer ponto z no segmento de reta que liga x e y, dado por $z = tx + (1-t)y$, estiver em C.*

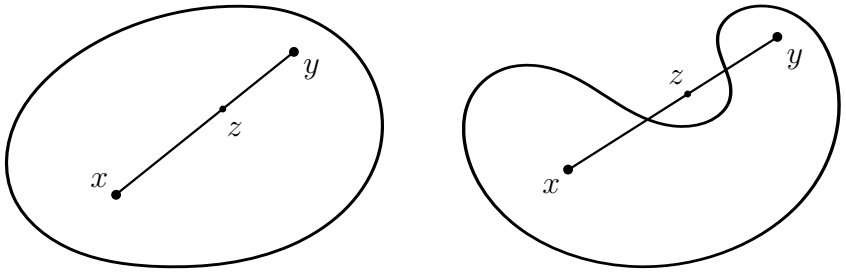

Figura 1: Conjuntos convexo e não convexo.

A convexidade de um conjunto é uma propriedade fundamental para definir funções convexas, como veremos a seguir.

Definição 1.3. *Uma função $f : C \to \mathbb{R}$ definida em um conjunto convexo $C \subset \mathbb{R}^n$ é dita* convexa *se, para quaisquer $x, y \in C$ e $t \in [0,1]$,*

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

Se a desigualdade na Definição 1.3 for estrita para todos $x \neq y$, dizemos que f é *estritamente convexa*. Uma propriedade importante das funções convexas é que seus minimizadores locais são também globais, conforme demonstrado no teorema a seguir.

Teorema 1.1. *Seja $f : C \to \mathbb{R}$ uma função convexa definida em um conjunto convexo $C \subset \mathbb{R}^n$. Se $x_* \in C$ é minimizador local de f, então x_* é minimizador global de f.*

Demonstração. Seja $x_* \in C$ minimizador local de f. Suponha, por contradição, que x_* não seja minimizador global. Assim, existe $y \in C$ tal que $f(y) < f(x_*)$. Como C é convexo, temos $\bar{x} = tx_* + (1-t)y \in C$, para todo $t \in [0,1]$. Contudo, para $t \in (0,1]$,

$$f(\bar{x}) = f(tx_* + (1-t)y) \leq tf(x_*) + (1-t)f(y) < tf(x_*) + (1-t)f(x_*) = f(x_*),$$

ou seja, $f(\bar{x}) < f(x_*)$ para todo $t \in (0,1]$, contrariando a minimalidade de x_*. □

Neste trabalho, estamos considerando que a função f do problema (1.1) é continuamente derivável e, portanto, suas derivadas parciais existem e também são contínuas. Quando a função é diferenciável a convexidade pode ser caracterizada de forma mais simples, como veremos no próximo teorema.

Teorema 1.2. *Seja $f : C \to \mathbb{R}$ uma função diferenciável definida em um conjunto convexo $C \subset \mathbb{R}^n$. Então, f é convexa se, e somente se, para todo $x, y \in C$,*

$$f(y) \geq f(x) + \nabla f(x)^t(y - x).$$

Demonstração. Pode ser encontrada em [9]. □

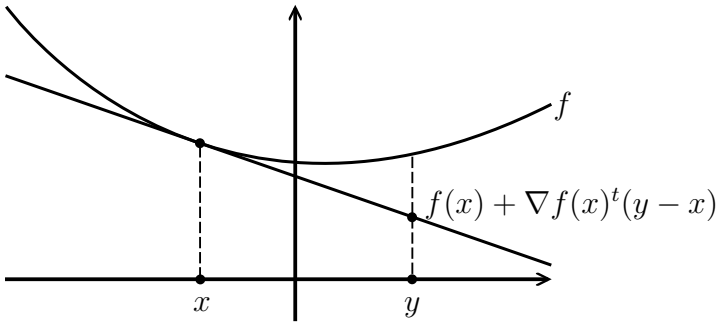

Figura 2: Interpretação geométrica do Teorema 1.2.

O Teorema 1.2 diz que a função f está sempre acima de suas aproximações lineares.

Definição 1.4. *Seja $A \in \mathbb{R}^{n\times n}$ uma matriz quadrada. Dizemos que A é definida positiva quando $x^tAx > 0$, para todo $x \in \mathbb{R}^n\backslash\{0\}$. Tal propriedade é denotada por $A \succ 0$. Se $x^tAx \geq 0$, para todo $x \in \mathbb{R}^n$, A é dita semidefinida positiva, fato este denotado por $A \succeq 0$.*

Para funções duas vezes diferenciáveis, a convexidade pode ser caracterizada em termos da matriz Hessiana, como dito no teorema a seguir.

Teorema 1.3. *Seja $f : C \to \mathbb{R}$ uma função duas vezes diferenciável definida em um conjunto convexo aberto $C \subset \mathbb{R}^n$. Então, f é convexa se, e somente se, a matriz Hessiana de f, denotada por $\nabla^2 f(x)$, for semidefinida positiva para todo $x \in C$.*

A convexidade é uma propriedade importante para lidar com problemas de otimização com restrições. Uma ferramenta útil para resolver tais problemas é a projeção ortogonal, que será apresentada a seguir.

1.3 Projeção sobre conjuntos convexos

A projeção ortogonal é uma ferramenta poderosa em otimização, especialmente quando lidamos com problemas que envolvem restrições em conjuntos convexos. A ideia central é encontrar o ponto de um conjunto que seja mais próximo de um ponto dado. Para formalizar esse conceito, considere o seguinte *problema de projeção* de z em Ω:

$$\begin{aligned} \underset{x}{\text{minimizar}} \quad & \rho(x) = \tfrac{1}{2}\|x - z\|^2, \\ \text{sujeito a} \quad & x \in \Omega. \end{aligned} \tag{1.3}$$

Dado um conjunto $\Omega \subset \mathbb{R}^n$ e um ponto $z \in \mathbb{R}^n$, o problema de encontrar o ponto em Ω mais próximo de z nem sempre possui solução ou, quando possui, pode não ser única. No entanto, se o conjunto Ω for convexo e fechado, garantimos algumas propriedades importantes.

Teorema 1.4 (Bolzano-Weierstrass)**.** *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função contínua definida em um conjunto compacto e não vazio $\Omega \subset \mathbb{R}^n$. Então, existe minimizador global de f em Ω.*

Demonstração. Pode ser encontrada em [9]. □

Teorema 1.5. *Seja $\Omega \subset \mathbb{R}^n$ um conjunto não vazio, convexo e fechado, e seja $z \in \mathbb{R}^n$. Então, a projeção de z em Ω, denotada por $P_\Omega(z)$, existe e é única.*

Demonstração. Consideraremos Ω como em (1.2). Como Ω é não vazio, o problema (1.3) é viável, ou seja, existe pelo menos um ponto $x \in \Omega$. Além disso, $\rho(x)$ é uma quadrática estritamente convexa, pois $\nabla^2 \rho(x) = I \succ 0$. Assim, pelo Teorema 1.4, existe um minimizador global x_* para o problema (1.3).

Agora suponha, por contradição, que $\bar{x}$ e $\tilde{x}$ sejam minimizadores distintos de (1.3).

Então, pela convexidade de Ω, $\hat{x} = \frac{1}{2}\bar{x} + \frac{1}{2}\tilde{x} \in \Omega$. Pela convexidade estrita de ρ, temos

$$\rho(\hat{x}) = \rho\left(\frac{1}{2}\bar{x} + \frac{1}{2}\tilde{x}\right) < \frac{1}{2}\rho(\bar{x}) + \frac{1}{2}\rho(\tilde{x}) \leq \frac{1}{2}\rho(x_*) + \frac{1}{2}\rho(x_*) = \rho(x_*),$$

o que contradiz o fato de x_* ser um minimizador global. Portanto, a projeção $P_\Omega(z)$ existe e é única. □

Uma das vantagens de trabalhar com restrições de caixa é a facilidade de calcular a projeção ortogonal em relação a esse tipo de conjunto. De fato, a projeção de $y \in \mathbb{R}^n$ sobre Ω é dada por $[P_\Omega(y)]_i = \max\{l_i, \min\{u_i, y_i\}\}$, para $i = 1, \ldots, n$, ou seja,

$$[P_\Omega(y)]_i = \begin{cases} l_i, & \text{se } y_i \leq l_i, \\ y_i, & \text{se } l_i \leq y_i \leq u_i, \\ u_i, & \text{se } y_i \geq u_i. \end{cases}$$

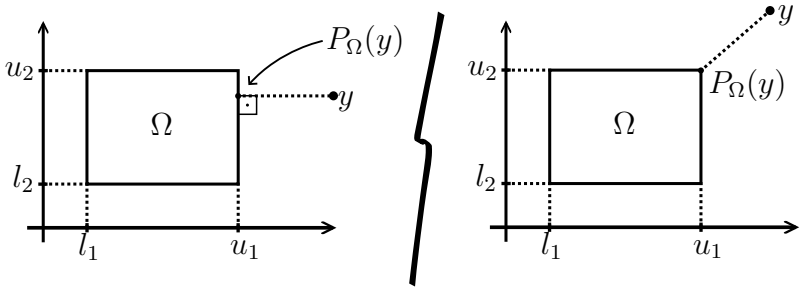

Figura 3: Exemplo visual da projeção de $y \in \mathbb{R}^2$ sobre Ω.

Teorema 1.6. *Seja $\Omega \subset \mathbb{R}^n$ um conjunto convexo fechado. Então, $p = P_\Omega(y)$ se, e somente se, para todo $x \in \Omega$,*

$$(x - p)^t(y - p) \leq 0.$$

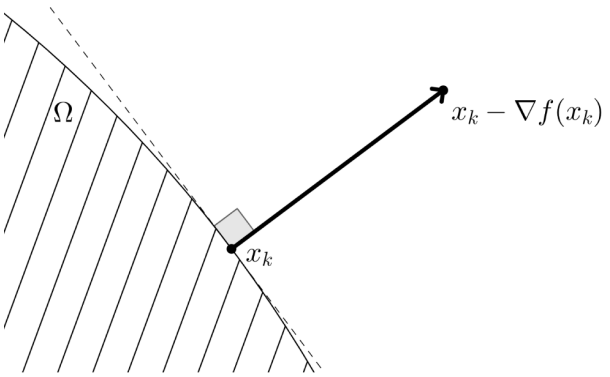

Figura 4: Projeção de $x_k - \nabla f(x_k)$ sobre o conjunto Ω igual a x_k.

Agora que entendemos o conceito de projeção e sua importância em problemas com restrições, podemos explorar as condições de otimalidade, que nos fornecem critérios para identificar soluções do problema de otimização.

1.4 Condições de otimalidade de primeira ordem

Em problemas com restrições, as condições de otimalidade envolvem não apenas o gradiente da função objetivo, mas também informações sobre a geometria do conjunto factível. *Condições necessárias* devem, obrigatoriamente, serem satisfeitas por minimizadores.

Teorema 1.7 (Condição necessária de primeira ordem com restrições)**.** *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função diferenciável e $\Omega \subset \mathbb{R}^n$ um conjunto convexo e fechado. Se x_* é um minimizador local de f em Ω, então para todo $x \in \Omega$,*

$$\nabla f(x_*)^t(x - x_*) \geq 0. \tag{1.4}$$

Demonstração. Pode ser encontrada em [12]. □

O Teorema 1.7 é crucial para o desenvolvimento de algoritmos de otimização com restrições, pois fornece um critério para determinar se um ponto é um possível minimizador local. Segundo este teorema, a condição necessária é que o gradiente da função objetivo em x_* deve formar um ângulo obtuso ou reto com qualquer vetor que aponta de x_* para um ponto factível $x \in \Omega$. Em outras palavras, o gradiente aponta para fora do conjunto factível, indicando que não é possível melhorar o valor da função objetivo movendo-se em qualquer direção factível a partir de x_*.

Além disso, esta condição pode ser interpretada em termos da projeção do gradiente negativo sobre o conjunto factível Ω. Se x_* for um minimizador local, então a projeção do vetor $-\nabla f(x_*)$ sobre Ω deve coincidir com o próprio ponto x_*. Isso sugere que o gradiente negativo aponta para dentro do conjunto factível, e que a única forma de se mover em uma direção factível a partir de x_* é aumentar o valor da função objetivo.

Definição 1.5. *Dizemos que um ponto $x_* \in \Omega$ é um ponto* estacionário *ou* crítico *para o problema* (1.1)*, se satisfaz a condição* (1.4)*.*

Teorema 1.8. *Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função diferenciável e $\Omega \subset \mathbb{R}^n$ um conjunto convexo e fechado. Se $x_* \in \Omega$ é minimizador local de f em Ω, então, para todo $\lambda > 0$,*

$$P_\Omega(x_* - \lambda \nabla f(x_*)) - x_* = 0.$$

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

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

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 $\lambda > 0$, temos

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

Finalmente, pelo Teorema 1.6, temos

$$P_\Omega(x_* - \lambda\nabla f(x_*)) = x_* \Leftrightarrow P_\Omega(x_* - \lambda\nabla f(x_*)) - x_* = 0,$$

completando a demonstração. □

2 Métodos para minimização irrestrita e sobre conjuntos convexos

Neste capítulo, serão apresentados os métodos de interesse para o estudo. Algumas referências para os métodos citados aqui são [1–4, 13–15].

2.1 Método do gradiente espectral projetado

Métodos tipo gradiente têm sua iteração definida pela expressão

$$x_{k+1} = x_k + t_k d_k,$$

onde $t_k > 0$ é o tamanho do passo e d_k é a direção de descida para f a partir do ponto x_k, geralmente dada por $d_k = -\nabla f(x_k)$.

O método do gradiente espectral vem da ideia de aproximar a iteração do método de Newton pela chamada equação secante, a mesma ideia empregada em métodos Quasi-Newton. Uma iteração típica destes métodos para resolver (1.1) é da forma

$$x_{k+1} = x_k - B_k^{-1} \nabla f(x_k), \tag{2.1}$$

onde B_k é uma aproximação da matriz Hessiana de f no iterando x_k. Esta aproximação visa assemelhar-se ao método de Newton, aproveitando a sua boa convergência local e evitando o custo computacional do cálculo da Hessiana.

Dados x_k e x_{k+1}, a aproximação de primeira ordem do gradiente de f em torno de x_k fornece

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

Desta aproximação, obtemos

$$\nabla^2 f(x_k) s_k \approx y_k,$$

onde $s_k = x_{k+1} - x_k$ e $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$, para $k \geq 1$. A equação secante é obtida trocando a Hessiana por uma matriz B_{k+1} e exigindo a igualdade, ou seja,

$$B_{k+1}s_k = y_k. \tag{2.2}$$

A equação (2.2) admite várias soluções e, por este motivo, há vários métodos Quasi-Newton secantes, tais como o método BFGS (Broyden-Fletcher-Goldfarb-Shanno) ou o método DFP (Davidon-Fletcher-Powell) [12]. A escolha de B_{k+1} para o método do gradiente espectral é dada por:

$$B_{k+1} = \sigma_{k+1}I,$$

onde $\sigma_{k+1} > 0$ e I é a matriz identidade. Como esta escolha nem sempre garante uma solução de (2.2), escolhemos σ_{k+1} que melhor aproxima a equação secante resolvendo o seguinte problema:

$$\underset{\sigma_{k+1}}{\text{minimizar}} \quad \tfrac{1}{2}\|\sigma_{k+1}s_k - y_k\|^2. \tag{2.3}$$

O problema (2.3) é convexo e pode ser resolvido anulando a derivada da função objetivo:

$$0 = \frac{d}{d\sigma_{k+1}}\left(\tfrac{1}{2}\|\sigma_{k+1}s_k - y_k\|^2\right) = \left(\sigma_{k+1}s_k - y_k\right)^t s_k.$$

Ou seja, minimizamos o resíduo de (2.2), nos levando a escolher

$$\sigma_{k+1} = \frac{s_k^t y_k}{s_k^t s_k},$$

sempre que $s_k \neq 0$. Sendo $B_{k+1} = \sigma_{k+1}I$, temos $B_{k+1}^{-1} = \frac{1}{\sigma_{k+1}}I$ e definimos λ_k como *passo espectral*, dado por:

$$\lambda_k = \frac{1}{\sigma_k} = \frac{s_{k-1}^t s_{k-1}}{s_{k-1}^t y_{k-1}}, \tag{2.4}$$

desde que $s_{k-1}^t y_{k-1} > 0$. Para manter certa estabilidade numérica e garantir convergência teórica do método, são definidos parâmetros limitantes para a variável λ_k, dados por $0 < \lambda_{\min} < \lambda_k < \lambda_{\max}$. Adotamos $\lambda_{\min} = 10^{-30}$ e $\lambda_{\max} = 10^{30}$, como em [1]. Quando (2.4) é negativo ou mesmo não definido, tomamos $\lambda_k = \lambda_{\max}$. Esta escolha tem como objetivo tentar "andar" ao máximo na direção $-\nabla f(x_k)$, deixando todo o ajuste para o tamanho do passo, realizado pela a *busca linear*.

Nessa etapa, o tamanho do passo é determinado de modo que a função objetivo diminua, ou, pelo menos, que o valor da função seja controlado para garantir a convergência ao ótimo no limite. Em uma busca linear *inexata*, o passo calculado não minimiza a função f na direção desejada, mas garante uma redução "satisfatória" de

f. Em geral, o passo deve satisfazer a *condição de Armijo*, e técnicas como *backtracking* e *interpolação quadrática* são comumente empregadas para ajustar o tamanho do passo adequadamente.

A condição de Armijo para (2.1) resulta em

$$f(x_k + t_k d_k) \leq f(x_k) + \eta t_k \nabla f(x_k)^t d_k,$$

onde $d_k = -\lambda_k \nabla f(x_k)$ e $\eta \in (0, 1)$ é um parâmetro. Esta condição força o decréscimo de f proporcional à $\eta \nabla f(x_k)^t d_k < 0$ em toda iteração.

O método com iteração $x_{k+1} = x_k - t_k \lambda_k \nabla f(x_k)$, com $\lambda_k \in [\lambda_{\min}, \lambda_{\max}]$, 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 que $t_k = 1$ com frequência, mesmo que eventualmente f aumente. Para isso, $t_k > 0$ deve satisfazer a condição de Armijo "relaxada":

$$f(x_k - t_k d_k) \leq f_{\max} + \eta t_k \nabla f(x_k)^t d_k, \tag{2.5}$$

onde $f_{\max} = \max\{f(x_k), f(x_{k-1}), \ldots, f(x_{k-M})\}$, para $M \geq 1$. Observe que a condição de Armijo usual é com $f_{\max} = f(x_k)$, o que força f decrescer sempre. Nesta nova condição, forçamos f a decrescer em relação às M últimas iterações, e não mais apenas em relação a anterior. Desse modo, a busca linear feita com (2.5) é não-monótona, pois permite que f cresça eventualmente. Este efeito pode ser visualizado na Figura 5.

Observe que com a busca linear não-monótona, o valor da função objetivo oscila ao longo dos iterandos e, mesmo assim, o problema é resolvido.

A seguir, o método do gradiente espectral projetado é sintetizado no Algoritmo 1 e é discutida sua boa definição e convergência global na subseção seguinte.

2.1.1 Boa definição e convergência global

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

Lema 2.1. *Temos, para todo $x \in \Omega$ e $\lambda \in (0, \lambda_{\max}]$,*

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

Demonstração. Sejam $v_k = x_k - \lambda \nabla f(x_k)$ e $p_k = P_\Omega(v_k) = d_k + x_k$. Pelo Teorema 1.6,

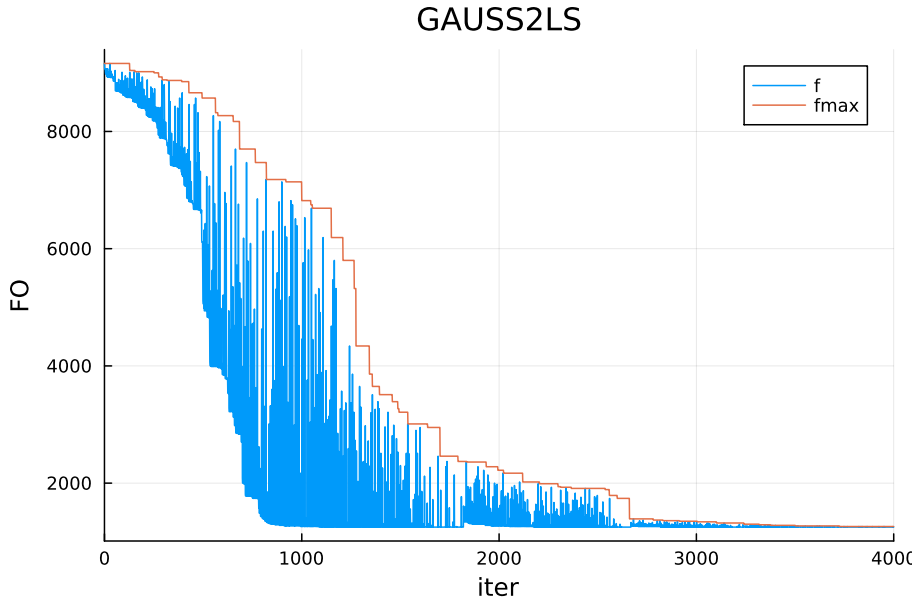

Figura 5: Efeito do uso da busca linear inexata não monótona no valor da função objetivo; problema GAUSS2LS da CUTEst.

como Ω é convexo e fechado, temos $(x_k - p_k)^t(v_k - p_k) \leq 0$. Observando que

$$(x_k - p_k)^t = (x_k - (d_k + x_k))^t = -(d_k)^t$$

e

$$v_k - p_k = x_k - \lambda \nabla f(x_k) - (d_k + x_k) = -\lambda \nabla f(x_k) - d_k,$$

obtemos

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

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 [1].

Teorema 2.1. *O Algoritmo 1 está bem definido e qualquer ponto de acumulação da sequência $\{x_k\}$ gerado pelo método é um ponto estacionário.*

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

Algoritmo 1 Método do Gradiente Espectral Projetado (SPG)

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

enquanto $\|P(x_k - \nabla f(x_k)) - x_k\| > \epsilon$ **faça**

$t_k = 1$

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

$x_{novo} = x_k + t_k d_k$

enquanto $f(x_{novo}) > f_{\max} + \eta t_k \nabla f(x_k)^t d_k$ **faça**

Compute $t_{quad} = -\frac{1}{2} t_k^2 \nabla f(x_k)^t d_k / (f(x_{novo}) - f(x_k) - t_k \nabla f(x_k)^t d_k)$, como descrito em [14]

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

$x_{k+1} = x_k - t_k \lambda_k \nabla f(x_k)$

fim enquanto

$s_k = x_{k+1} - x_k$

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

se $(s_k^t y_k) \leq 0$ **então**

$\lambda_{k+1} = \lambda_{\max}$

senão

$\lambda_{k+1} = \min\left\{\lambda_{\max}, \max\left\{\lambda_{\min}, \frac{s_k^t s_k}{s_k^t y_k}\right\}\right\}$

fim se

fim enquanto

Se x_k não é um ponto estacionário, então pelo Lema 2.1,

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

e a direção de busca é uma direção de descida. Portanto, um tamanho de passo satisfazendo (2.5) 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_*. Considere 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 de Ω, existe $\delta > 0$ tal que, para todo $\lambda \in [\lambda_{\min}, \lambda_{\max}]$,

$$\nabla f(x_*)^t \frac{p_\lambda(x_*)}{\|p_\lambda(x_*)\|} \leq -\delta \Rightarrow \nabla f(x_k)^t \frac{p_\lambda(x_k)}{\|p_\lambda(x_k)\|} \leq -\frac{\delta}{2}, \tag{2.6}$$

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 (2.5), 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 (2.5), 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,

$$\frac{f(x_k + \bar{t_k} d_k) - f(x_k)}{\bar{t_k}} > \eta \nabla f(x_k)^t d_k. \tag{2.7}$$

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 (2.7) como

$$\nabla f(x_k + t_k d_k)^t d_k > \eta \nabla f(x_k)^t d_k, \tag{2.8}$$

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 (2.8) 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 $\nabla f(x_*)^t d = 0$. 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 (2.6). 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 $t \in (0, \lambda_{\max}]$. Pela continuidade e compacidade de Ω, existe $\delta > 0$ tal que $\|p_t(x_*)\| \geq \delta > 0$, para todo $t \in [\rho, \lambda_{\max}]$. Agora, considere que $l(k)$ seja um inteiro tal que

$$k - m(k) \leq l(k) < k, \quad f(x_{l(k)}) = \max_{0 \leq j \leq m(k)} [f(x_{k-j})], \tag{2.9}$$

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 (2.5), para $k > M$, obtemos

$$\begin{aligned} f(x_{l(k)}) &= f(x_{l(k)-1} + t_{l(k)-1}d_{l(k)-1}) \\ &\leq \max_{0\leq j\leq m(l(k)-1)}[f(x_{l(k)-1-j})] + \eta t_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1} \\ &= f(x_{l(l(k)-1)} + \eta t_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1}. \end{aligned} \tag{2.10}$$

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 (2.10) que

$$\lim_{k\to\infty} t_{l(k)-1}\nabla f(x_{l(k)-1})^t d_{l(k)-1} = 0. \tag{2.11}$$

Do Lema 2.1, temos

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

para todo k e, como $t_k < \bar{t}$, (2.11) implica que

$$\lim_{k\to\infty} t_{l(k)-1}\|d_{l(k)-1}\| = 0. \tag{2.12}$$

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

$$\lim_{k\to\infty} t_{\hat{l}(k)-j}\|d_{\hat{l}(k)-j}\| = 0 \tag{2.13}$$

e

$$\lim_{k\to\infty} f(x_{\hat{l}(k)-j}) = \lim_{k\to\infty} f(x_{l(k)}). \tag{2.14}$$

(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 (2.12), segue que (2.13) é válido. Isto implica que $\|x_{\hat{l}(k)} - x_{\hat{l}(k)-1}\| \to 0$, de modo que (2.14) valha para $j = 1$, já que $f(x)$ é uniformemente contínua em Ω. Assuma agora que (2.13) e (2.14) valem para um dado j. Então, por (2.10),

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

Tomando limite para $k \to \infty$, de (2.14) temos

$$\lim_{k\to\infty} t_{\hat{l}(k)-(j+1)} \nabla f(x_{\hat{l}(k)-(j+1)}) d_{\hat{l}(k)-(j+1)} = 0,$$

e usando os mesmo argumentos que resultaram (2.12),

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

Isto implica que $\|x_{\hat{l}(k)-j} - x_{\hat{l}(k)-(j+1)}\| \to 0$, tal que de (2.14) 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 (2.13) e (2.14) 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,

$$x_{k+1} = x_{\hat{l}(k)} - \sum_{j=1}^{\hat{l}(k)-k-1} t_{\hat{l}(k)-j} d_{\hat{l}(k)-j}. \tag{2.15}$$

Por (2.9), temos $\hat{l}(k) - k - 1 = l(k + M + 2) - k - 1 \leq M + 1$, de modo que (2.15), por (2.13), implica que $\lim_{k\to\infty} \|x_{k+1} - x_{\hat{l}(k)}\| = 0$. Como $\{f(x_l(k))\}$ admite limite, segue, da continuidade de f em Ω, que

$$\lim_{k\to\infty} f(x_k) = \lim_{k\to\infty} f(x_{\hat{l}(k)}) = \lim_{k\to\infty} f(x_{l(k)}) = f(x_*). \tag{2.16}$$

Por continuidade, para $k > \bar{k}$ suficientemente grande, temos $\|p_{\lambda_k}(x_k)\| \geq \frac{\delta}{2}$. Usando (2.10) e o Lema 2.1, 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 (2.16) e a continuidade de f em x_*. Portanto, x_* é estacionário. □

2.2 Método *Adaptive Barzilai-Borwein* (ABB)

Ao resolver o problema (2.3), obtemos a seguinte expressão para o passo λ_k:

$$\lambda_k^{BB1} = \frac{s_{k-1}^t s_{k-1}}{s_{k-1}^t y_{k-1}}. \tag{2.17}$$

Outra alternativa para determinar λ_k é resolvendo o problema:

$$\underset{\sigma_k}{\text{minimizar}} \quad \tfrac{1}{2}\|\sigma_{k+1}^{-1} y_k - s_k\|^2.$$

Da mesma forma que resolvemos (2.3), definimos

$$\lambda_k^{BB2} = \frac{s_{k-1}^t y_{k-1}}{y_{k-1}^t y_{k-1}}, \tag{2.18}$$

desde que $y_{k-1} \neq 0$.

De acordo com [2], a razão $\lambda_k^{BB2}/\lambda_k^{BB1}$ ser pequena, por exemplo menor que um parâmetro $\kappa \in (0,1)$, indica que $\|\nabla f(x_k)\|_2$ reduz menos usando (2.17) que (2.18). Portanto, deve-se optar pelo passo mais controlado λ_k^{BB2}. Caso contrário, escolhe-se λ_k^{BB1} para um passo mais agressivo. Assim, no método ABB, a escolha de λ_k é feita da seguinte forma:

$$\lambda_k = \begin{cases} \lambda_k^{BB2}, & \text{se } \lambda_k^{BB2}/\lambda_k^{BB1} < \kappa, \\ \lambda_k^{BB1}, & \text{caso contrário.} \end{cases} \tag{2.19}$$

2.3 Método ABBmin

Em [3] são exploradas diversas estratégias para a seleção de tamanhos de passo. Uma dessas abordagens é o método *Adaptive Cyclic Barzilai-Borwein* (ACBB), que introduz uma escolha adaptativa para o comprimento do ciclo, também conhecido como "memória". Testes indicaram que tanto o ACBB quanto o ABB são eficazes e frequentemente superam outras abordagens semelhantes ao método Barzilai-Borwein. O método ABB, em particular, tem um desempenho superior em problemas quadráticos grandes e mal condicionados.

Buscando melhorar a seleção do passo λ_k conforme estabelecido em (2.19), no método nomeado de ABBmin, é proposta a seguinte adaptação:

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

Esta abordagem utiliza a ideia de "memória", permitindo a reutilização do mesmo

passo em iterações consecutivas, semelhante ao método ACBB. De acordo com [3], o método ABBmin se revela bem adequado para problemas gerais de otimização não linear, assim como o método ABB padrão.

2.4 Método de gradientes conjugados de Dai-Kou

No método de gradientes conjugados desenvolvido por Dai e Kou, a direção d_k é escolhida da seguinte maneira (para simplificar a notação, considere $g_k = \nabla f(x_k)$):

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

onde

$$\mu_k = \frac{1}{\Delta_k}\left(g_k^t y_{k-1} g_k^t s_{k-1} - s_{k-1}^t y_{k-1} g_k^t g_k\right), \quad \nu_k = \frac{1}{\Delta_k}\left(g_k^t y_{k-1} g_k^t g_k - \rho_k g_k^t s_{k-1}\right), \quad (2.20)$$

e

$$\Delta_k = \rho_k s_{k-1}^t y_{k-1} - \left(g_k^t y_{k-1}\right)^2 > 0. \quad (2.21)$$

De fato, (2.20) e (2.21) são obtidos resolvendo um problema de minimização de uma função quadrática aproximada - veja [4] 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:

$$\rho_k^{BB1} = \frac{s_{k-1}^t y_{k-1}}{s_{k-1}^t s_{k-1}} g_k^t g_k \quad (2.22)$$

e

$$\rho_k^{BB2} = \frac{y_{k-1}^t y_{k-1}}{s_{k-1}^t y_{k-1}} g_k^t g_k. \quad (2.23)$$

Um ponto importante a ser destacado sobre a fórmula (2.23) é que, se $s_{k-1}^t y_{k-1} > 0$, a relação (2.21) é sempre satisfeita, a menos de quando y_{k-1} e g_k são colineares. Estudos numéricos realizados em [4], apontam que (2.22) tem uma performance superior a (2.23). Apesar disso, é introduzido um parâmetro $\omega_k \geq 1$ em (2.23), obtendo

$$\rho_k^{BB3} = \omega_k \frac{y_{k-1}^t y_{k-1}}{s_{k-1}^t y_{k-1}} g_k^t g_k. \quad (2.24)$$

Também por experimentos numéricos, é visto que a escolha de ρ_k como em (2.24) com $\omega_k = 3/2$ performa melhor que ρ_k como em (2.22). Portanto, esta foi a opção adotada

neste trabalho.

3 Aprendizado de máquina supervisionado

Na última década, o aprendizado de máquina tornou-se uma ferramenta fundamental no cotidiano, sendo aplicado em diversos problemas como classificação de imagens, reconhecimento de padrões e segmentação semântica. Os modelos de redes neurais permitem a identificação de padrões complexos em grandes volumes de dados. Neste contexto, exploraremos os principais conceitos relacionados ao aprendizado de máquina supervisionado, com o objetivo de conectá-los à ideia do método do gradiente espectral abordada nos capítulos anteriores. As principais referências deste capítulo são [5, 16].

3.1 Preliminares

O aprendizado de máquina supervisionado foca no desenvolvimento de algoritmos que permitem aos computadores aprender a partir de dados rotulados fornecidos pelo usuário. Em particular, consideramos o cenário onde o modelo é treinado com um conjunto de dados composto por entradas e saídas esperadas, denotado como $\{(x_i, y_i)\}_{i=1}^{m}$, onde x_i representa o dado de entrada, y_i a saída real correspondente e m o número total de amostras.

A otimização desempenha um papel crucial na calibragem dos parâmetros do modelo, conhecidos como pesos e vieses. Denotamos (W, b) como o conjunto de parâmetros da rede neural, com W e b representando, respectivamente, os pesos e vieses. Para ajustar esses parâmetros, utilizamos uma *função de predição* h, cuja eficácia é avaliada pela quantidade de previsões incorretas, ou seja, $h(x_i) \neq y_i$. Por simplicidade, considere por hora que

$$h(x; W, b) = W^t x + b, \tag{3.1}$$

onde W e b são otimizados para que a predição $h(x_i; W_*, b_*)$ seja o mais próximo possível de y_i.

Um processo comum para isso é usar a chamada *função de perda*. Também conhecida como *função de custo*, é uma medida que quantifica a diferença entre as predições feitas por um modelo de aprendizado de máquina e os valores reais esperados. Temos como exemplo a função de perda logarítmica, $l(h; y) = \log(1 + e^{-hy})$, com $h \in [-1, 1]$. Entretanto, neste trabalho usamos a função de erro quadrático, dada por

$$l(h; y) = (h - y)^2,$$

com h tal qual (3.1). Desse modo, o problema ser resolvido é dada por

$$\underset{W,b}{\text{minimizar}} \quad R_m(W, b) = \frac{1}{m} \sum_{i=1}^{m} l(h(x_i; W, b); y_i). \tag{3.2}$$

$R_m(W, b)$ é chamado *risco empírico*.

3.1.1 Redes neurais

Redes neurais são modelos computacionais inspirados na estrutura e no funcionamento do cérebro humano, projetados para reconhecer padrões, aprender com dados e tomar decisões. Elas são compostas por camadas interconectadas que simulam a forma como os neurônios biológicos se comunicam. Na prática, temos uma camada de entrada de dados, uma camada intermediária e uma camada de saída de dados.

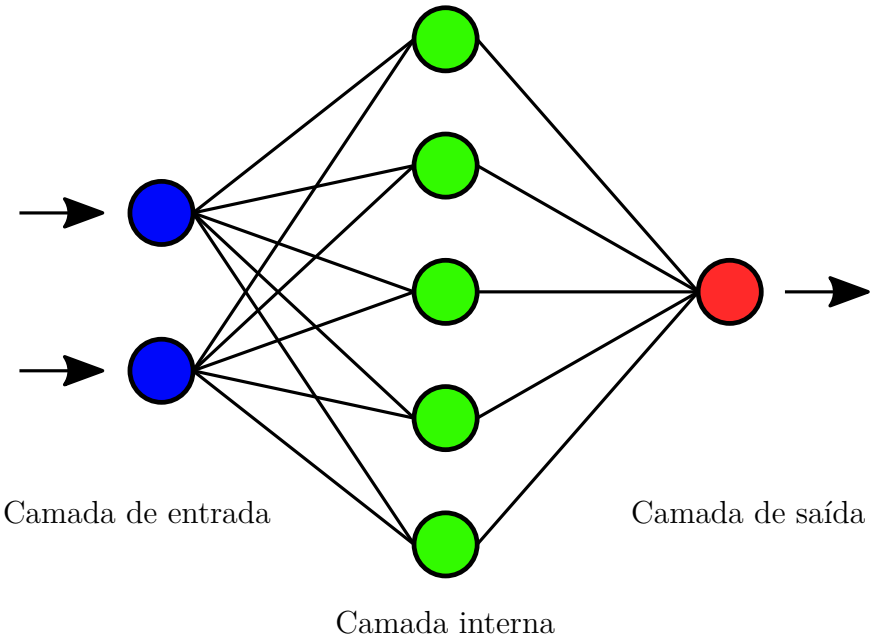

Figura 6: Diagrama simplificado de uma rede neural.

A seguir, apresentamos como é feita essa relação.

Dada uma amostra $x_i \in \mathbb{R}^m$, escrevemos h como uma aplicação sucessiva de funções, ou seja,

$$h(x) = f_L(f_{L-1}(\cdots f_1(x)\cdots)),$$

onde L é o número de camadas da rede neural. No procedimento que chamamos de *feedfoward*, para avaliar $h(x)$, avaliamos a sequência

$$f_1 = f_1(x), \quad f_2 = f_2(f_1), \quad \ldots, \quad h(x) = f_L(f_{L-1}).$$

Para avaliar $\nabla h(x)$, usamos a regra da cadeia. Desse modo, suponha que $f_i : \mathbb{R} \to \mathbb{R}$, para todo i. Assim, temos o procedimento descrito como *backpropagation*, dado por

$$\nabla h(x) = f_L'(f_{L-1}) \cdot f_{L-1}'(f_{L-2}) \cdot \ldots \cdot f_1'(x).$$

A concatenação das f_i's pode ser vista por meio de um grafo. Por sua inspiração em modelos para neurônios, tal grafo é chamado de *rede neural*. Os nós, junto com seu "mecanismo de controle do fluxo", são chamados de *neurônios*.

É comum que cada f_j seja a composta de uma aplicação afim com uma função não-linear a, chamada de *função de ativação*, ou seja,

$$x_i^{(j)} = a(W_j x_i^{(j-1)} + b_j),$$

para todo i. Algumas escolhas recorrentes para a são $a(z) = 1/(1 - e^{-z})$ (Sigmoid), que usamos neste trabalho, e $a(z) = \max\{0, z\}$ (ReLU). As funções de ativação têm o intuito de ajustar o fluxo na rede neural.

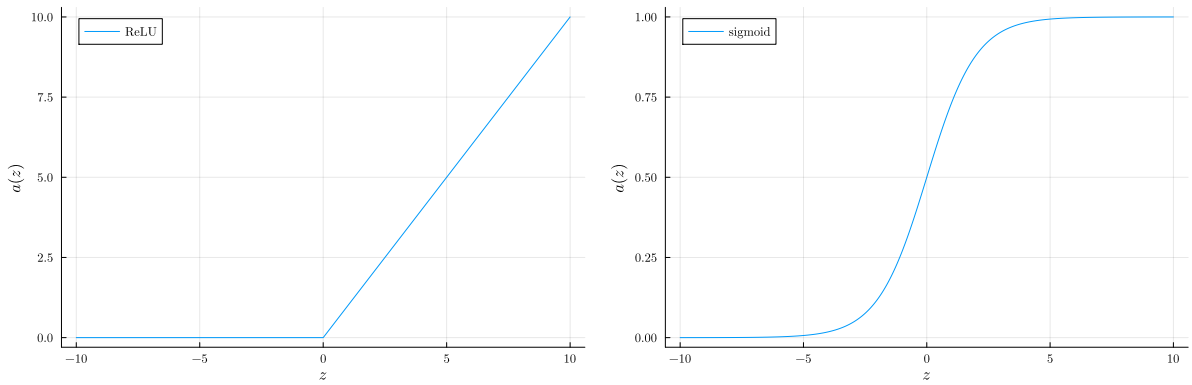

Figura 7: Funções de ativação ReLU (à esquerda) e Sigmoid (à direita).

3.1.2 Método do gradiente incremental

A fim de apresentar os métodos mais utilizados no treinamento de redes neurais, é instrutivo começar com o método do gradiente incremental. É importante ressaltar que este método não é aplicado com sucesso ao treinamento de redes neurais. No entanto, ele contém um ingrediente fundamental para o contexto: o de minimizar uma função definida por uma soma tomando, a cada passo, gradientes de apenas alguns termos da soma.

A soma em $R_m(W, b)$ envolve muitos termos devido ao grande número de dados no treinamento, tornando o cálculo de ∇R_m caro. A solução é substituir ∇R_m pela soma de alguns poucos gradientes a cada iteração, o que leva ao método do gradiente incremental. Nesse método, temos um problema do tipo

$$\begin{aligned} \text{minimizar} \quad & f(x) = \sum_{i=1}^{m} f_i(x), \\ \text{sujeito a} \quad & x \in \mathbb{R}^n. \end{aligned}$$

Como m é muito grande, avaliamos ∇f_i para algum i por iteração e, a cada iteração, escolhemos um $j = 0, \ldots, m-1$, dando um passo na direção $-\nabla f_{j+1}(x_{k,j})$, onde $x_{k,j}$ é o ponto corrente.

Na k-ésima iteração do método, calculamos o passo $t_k > 0$, iniciamos com o iterando externo x_k e atualizamos o ponto corrente dando um passo de tamanho t_k na direção de cada gradiente ∇f_j separadamente. Mais especificamente, a iteração k consiste em

$$x_{k,0} = x_k, \quad x_{k,j+1} = x_{k,j} - t_k \nabla f_{j+1}(x_{k,j}), \quad j = 0, \ldots, m-1. \tag{3.3}$$

Note que o passo da iteração interna (em j) é realizado sobre o ponto imediatamente anterior, ou seja, $x_{k,j+1}$ aproveita $x_{k,j}$. Essa é a diferença com o gradiente tradicional, que visa acelerar a convergência a um custo não maior. O laço que percorre todos os gradientes ∇f_{j+1}, $j = 0, \ldots, m-1$, é chamado de ciclo. Note que t_k é constante para cada ciclo.

O Algoritmo 2 apresenta o funcionamento descrito acima e, em seguida, discutimos sua convergência para funções convexas.

Algoritmo 2 Esquema de gradiente incremental

Entrada: Ponto inicial $x_0 \in \mathbb{R}^n$, número máximo de iterações K.

1: **para** $k \leftarrow 0$ até K **faça**
2: Calcule $t_k > 0$
3: **para** $j \leftarrow 0$ até $m-1$ **faça**
4: $x_{k,0} \leftarrow x_k$
5: $x_{k,j+1} \leftarrow x_{k,j} - t_k \nabla f_{j+1}(x_{k,j})$
6: **fim para**
7: $x_{k+1} \leftarrow x_{k,m}$
8: **fim para**

Seja $\{x_k\}$ a sequência gerada pelo método. Considere a hipótese abaixo.

Hipótese 3.1 (limitação do gradiente)**.** *Existe $L > 0$ tal que $\|\nabla f_{j+1}(x_{k,j})\| \leq L$, para todo j, k.*

O lema e o teorema a seguir foram adaptados de [16].

Lema 3.1. *Suponha f_i convexa para todo i e a Hipótese 3.1 válida. Temos, para todo $y \in \mathbb{R}^n$,*

$$\|x_{k+1} - y\|^2 \leq \|x_k - y\|^2 - 2t_k(f(x_k) - f(y)) + t_k^2 m^2 L^2.$$

Demonstração. Para cada $j = 0, \ldots, m-1$, temos

$$\begin{aligned}
\|x_{k,j+1} - y\|^2 &= \|x_{k,j} - t_k \nabla f_{j+1}(x_{k,j}) - y\|^2 \\
&= \|x_{k,j} - y\|^2 - 2t_k \nabla f_{j+1}(x_{k,j})^t (x_{k,j} - y) + t_k^2 \|\nabla f_{j+1}(x_{k,j})\|^2 \\
&\underset{(*)}{\leq} \|x_{k,j} - y\|^2 - 2t_k(f_{j+1}(x_{k,j}) - f_{j+1}(y)) + t_k^2 L^2,
\end{aligned}$$

onde $(*)$ segue da convexidade de f_{j+1} e da Hipótese 3.1.

Somando para $j = 0, \ldots, m-1$, usando as identidades $x_{k,0} = x_k$ e $x_{k,m} = x_{k+1}$ e cancelando termos de ambos os lados da desigualdade, obtemos

$$\|x_{k+1} - y\|^2 \leq \|x_k - y\|^2 - 2t_k \sum_{j=0}^{m-1} (f_{j+1}(x_{k,j}) - f_{j+1}(y)) + t_k^2 m L^2.$$

Como $\sum_{j=0}^{m-1} f_{j+1}(z) = f(z)$, temos

$$\|x_{k+1} - y\|^2 \leq \|x_k - y\|^2 - 2t_k \left(f(x_k) - f(y) + \sum_{j=0}^{m-1} (f_{j+1}(x_{k,j}) - f_{j+1}(x_k)) \right) + t_k^2 m L^2.$$

Como f_{j+1} é convexa, temos

$$f_{j+1}(x_{k,j}) - f_{j+1}(x_k) \geq \nabla f_{j+1}(x_k)^t (x_{k,j} - x_k),$$

daí,

$$\begin{aligned}\|x_{k+1} - y\|^2 &\leq \|x_k - y\|^2 - 2t_k(f(x_k) - f(y)) + 2t_k \sum_{j=0}^{m-1} -\nabla f_{j+1}(x_k)^t (x_{k,j} - x_k) + t_k^2 m L^2 \\ &\leq \|x_k - y\|^2 - 2t_k(f(x_k) - f(y)) + 2t_k L \sum_{j=0}^{m-1} \|x_{k,j} - x_k\| + t_k^2 m L^2. \qquad (3.4)\end{aligned}$$

Agora, veja que

$$\|x_{k,1} - x_k\| = \|x_{k,0} - t_k \nabla f_1(x_{k,0}) - x_k\| \leq t_k \|\nabla f_1(x_{k,0})\| \leq t_k L.$$

Similarmente,

$$\|x_{k,2} - x_k\| = \|x_{k,1} - t_k \nabla f_2(x_{k,1}) - x_k\| \leq \|x_{k,1} - x_k\| + t_k \|\nabla f_2(x_{k,1})\| \leq 2t_k L.$$

Em geral,

$$\|x_{k,j} - x_k\| \leq j t_k L, \quad j = 0, \ldots, m-1,$$

e podemos continuar (3.4) deduzindo que

$$\|x_{k+1} - y\|^2 \leq \|x_k - y\|^2 - 2t_k(f(x_k) - f(y)) + 2t_k^2 L^2 \sum_{j=0}^{m-1} j + t_k^2 m L^2.$$

Como $\sum_{j=0}^{m-1} j = \frac{m(m-1)}{2}$, obtemos

$$\|x_{k+1} - y\|^2 \leq \|x_k - y\|^2 - 2t_k(f(x_k) - f(y)) + t_k^2 m^2 L^2,$$

como queríamos provar. □

Daqui em diante, denotaremos o melhor valor objetivo encontrado por um método até a iteração k como

$$f_k = \min_{0 \leq j \leq k} f(x_j).$$

Note que $\{f_k\}$ é não crescente e, logo, $\lim_{k \to \infty} f_k$ existe ou é $-\infty$.

Teorema 3.1. *Suponha f_i convexa para todo i, a Hipótese 3.1 válida e que $f(x) = \sum_{i=1}^m f_i(x)$ admita minimizador x_*. Seja $\{t_k\}$ a sequência de passos do Algoritmo 2.*

(a) Se

$$t_k \to 0^+, \quad \sum_{k=0}^{\infty} t_k = \infty \quad e \quad \sum_{k=0}^{\infty} t_k^2 < \infty,$$

então,

$$\lim_{k\to\infty} f_k = f(x_*).$$

(b) Se $t_k = t > 0$ para todo k, então,

$$\lim_{k\to\infty} f_k \leq f(x_*) + \frac{tm^2L^2}{2},$$

onde L é a constante dada na Hipótese 3.1.

Demonstração. (a) Pelo Lema 3.1 (com $y = x_*$), temos

$$\begin{aligned}
\|x_{k+1} - x_*\|^2 &\leq \|x_k - x_*\|^2 - 2t_k(f(x_k) - f(x_*)) + t_k^2 m^2 L^2 \\
&\leq \|x_{k-1} - x_*\|^2 - 2t_{k-1}(f(x_{k-1}) - f(x_*)) + t_{k-1}^2 m^2 L^2 \\
&- 2t_k(f(x_k) - f(x_*)) + t_k^2 m^2 L^2 \\
&\vdots \\
&\leq \|x_0 - x_*\|^2 - 2\sum_{j=0}^{k} t_j(f(x_j) - f(x_*)) + m^2L^2 \sum_{j=0}^{k} t_j^2.
\end{aligned}$$

Isto implica que

$$\begin{aligned}
\left(2\sum_{j=0}^{k} t_j\right)(f_k - f(x_*)) &\leq 2\sum_{j=0}^{k} t_j(f(x_j) - f(x_*)) \\
&\leq \|x^0 - x_*\|^2 + m^2L^2 \sum_{j=0}^{k} t_j^2.
\end{aligned}$$

Daí,

$$f_k - f(x_*) \leq \frac{\|x_0 - x_*\|^2 + m^2L^2 \sum_{j=0}^{k} t_j^2}{2\sum_{j=0}^{k} t_j}.$$

De fato, das condições impostas sobre $\{t_k\}$ e fazendo $k \to \infty$, segue que

$$\begin{aligned}
&\lim_{k\to\infty} f_k - f(x_*) \leq \lim_{k\to\infty} \frac{\|x_0 - x_*\|^2 + m^2L^2 \sum_{j=0}^{k} t_j^2}{2\sum_{j=0}^{k} t_i} \\
\Rightarrow &\lim_{k\to\infty} f_k - f(x_*) \leq 0 \\
\Rightarrow &\lim_{k\to\infty} f_k = f(x_*).
\end{aligned}$$

(b) Novamente, temos, pelo Lema 3.1,

$$\|x_{k+1} - x_*\|^2 \leq \|x_k - x_*\|^2 - 2t_k(f(x_k) - f(x_*)) + t_k^2 m^2 L^2, \tag{3.5}$$

para todo k.

Suponha, por contradição, que

$$\lim_{k\to\infty} f_k > f(x_*) + \frac{tm^2L^2}{2} + 2\varepsilon, \tag{3.6}$$

para algum $\varepsilon > 0$. Além disso, para todo $k \gg 1$, temos

$$f_k \geq \lim_{k\to\infty} f_k - \varepsilon. \tag{3.7}$$

Somando (3.6) e (3.7) obtemos, para um certo k_0,

$$f_k - f(x_*) \geq \frac{tm^2L^2}{2} + \varepsilon, \quad \forall k \geq k_0.$$

Assim, da inequação (3.5) com $t_k = t$ temos

$$\begin{aligned}\|x_{k+1} - x_*\|^2 &\leq \|x_k - x_*\|^2 - 2t\left(\frac{tm^2L^2}{2} + \varepsilon\right) + t^2m^2L^2 \\ &\leq \|x_k - x_*\|^2 - 2t\varepsilon, \quad \forall k \geq k_0.\end{aligned}$$

Aplicando essa desigualdade sucessivas vezes, obtemos

$$\|x_{k+1} - x_*\|^2 \leq \|x_{k_0} - x_*\|^2 - 2(k + 1 - k_0)t\varepsilon.$$

Tomando $k \gg k_0$, obtemos uma contradição, pois $-2(k + 1 - k_0)t\varepsilon \to -\infty$ e $\|x_{k+1} - x_*\|^2 \geq 0$. Em outras palavras, tomando $\varepsilon = 0$, vale

$$\lim_{k\to\infty} f_k \leq f(x_*) + \frac{tm^2L^2}{2},$$

como queríamos provar. □

Na próxima seção apresentamos a "versão estocástica" do método do gradiente incremental, que é o principal método utilizado para treinamento de redes neurais. Basicamente, a diferença é que na versão estocástica escolhemos a variável do *loop* interno usada para dar o passo de maneira aleatória (em verdade, o método apresentado na seção seguinte trabalha com blocos de variáveis – os chamados mini-lotes – ao invés de uma única variável por iteração interna). Esse detalhe é fundamental para o sucesso do método

no treinamento de redes neurais. É comum considerar neste método o passo constante, ou seja, $t_k = t > 0$, para todo k. Contudo, a "versão estocástica" do SPG descrita no próximo capítulo usa o passo decrescente, isto é, $\{t_k\}$, tal que

$$t \to 0^+, \quad \sum_{k=0}^{\infty} t_k = \infty \quad \text{e} \quad \sum_{k=0}^{\infty} t_k^2 < \infty. \tag{3.8}$$

Note que no teorema anterior os dois casos são cobertos.

3.2 Método do gradiente estocástico para treinamento de redes neurais

Um método eficaz e eficiente para encontrar uma solução adequada para o problema (3.2) é fundamental para o sucesso no aprendizado. No entanto, esse problema geralmente é de larga escala, não suave e não convexo, o que torna os métodos de primeira ordem, como o método de descida do gradiente, as escolhas preferenciais. Dentre esses métodos, o de descida do gradiente estocástico (do inglês, *Stochastic Gradient Descent*, ou SGD) destaca-se como o mais amplamente utilizado no treinamento de redes neurais.

Nele, temos um problema do tipo

$$\underset{x}{\text{minimizar}} \quad f(x),$$

onde $f : \mathbb{R}^n \to \mathbb{R}$ é uma função convexa. Neste problema, não temos acesso direto a ∇f, uma vez que, em muitos casos, seu cálculo completo é computacionalmente dispendioso.

A iteração do método SGD tradicional é dada por

$$x_{k+1} = x_k - t_k g_k,$$

onde $t_k = t > 0$, constante, é o tamanho do passo e g_k é uma estimativa *aleatória* de $\nabla f(x_k)$. A seleção aleatória adotada por este método o diferencia do gradiente incremental e justifica a denominação "estocástico" (vide linha 4 do Algoritmo 3). Como consequência, os resultados de convergência serão expressos em termos probabilísticos, em contraste com o Lema 3.1 e o Teorema 3.1 apresentados na seção anterior. Adicionalmente, é crucial destacar que a estimativa de g_k deve ser realizada de maneira uniforme e independente, garantindo que cada amostra tenha a mesma probabilidade (isto é, $\mathbb{P}(i) = 1/m$ para todo i) de ser selecionada, evitando, assim, qualquer viés no processo de escolha.

Conforme mencionado anteriormente, em vez de utilizar o gradiente completo das

m amostras de treinamento, os métodos clássicos de SGD calculam o gradiente com base em uma pequena amostra a cada iteração. Esse procedimento pode apresentar ineficiências, devido à alta variância e instabilidade no cálculo do passo, resultando em uma convergência mais lenta e menos precisa. Para mitigar esses efeitos, uma abordagem amplamente adotada é a de *mini-lotes*, que consiste em usar uma porção reduzida $|B_k|$ das amostras de treinamento para obter uma estimativa do gradiente. Nessa abordagem, B_k denota o índice do conjunto das amostras escolhidas aleatoriamente do conjunto de dados de treinamento na iteração k, e $|B_k|$ denota a cardinalidade de B_k.

Pode-se observar que $|B| = m$ (um único lote) refere-se ao método de gradiente clássico, uma vez que calcula-se $\nabla f(x_k)$ em sua totalidade; $|B| = 1$ corresponde ao método SGD tradicional; e, quando $1 < |B| \ll m$, utiliza-se a estratégia de mini-lotes, a qual resulta em iterações mais eficientes em comparação com as demais abordagens, sendo, por essa razão, a escolhida para este trabalho. Ressalta-se que o ajuste do valor de $|B|$ é determinado de forma empírica.

Desse modo, a iteração do método SGD com mini-lotes é dada por

$$x_{k,0} = x_k, \quad x_{k,j+1} = x_{k,j} - \frac{t_k}{|B_k|}\nabla f_{j+1}(x_{k,j}), \quad j = 0, \ldots, m-1. \tag{3.9}$$

Observe a diferença entre a iteração (3.9) e a iteração (3.3) do método de gradiente incremental, que se deve à aleatoriedade da porção $|B_k|$. Neste contexto, o ciclo que percorre todos os gradientes é denominado *época*. Na prática, o cálculo de $\nabla f(x)$ é realizado por meio da técnica de *backpropagation*, conforme discutido na Subseção 3.1.1.

O Algoritmo 3 ilustra o funcionamento desse esquema, seguido da apresentação dos resultados de convergência do método aplicados a funções convexas.

Algoritmo 3 Método do gradiente estocástico com mini-lotes

Entrada: Ponto inicial $x_0 \in \mathbb{R}^n$, número máximo de épocas K, número de passos por época J, tamanho do mini-lote $|B|$.

1: **para** $k \leftarrow 0$ até K **faça**
2: Calcule $t_k > 0$
3: **para** $j \leftarrow 0$ até $J-1$ **faça**
4: Construa um lote $B_{k,j}$ aleatoriamente de tamanho $|B|$ e de maneira uniforme
5: $x_{k,0} \leftarrow x_k$
6: $x_{k,j+1} \leftarrow x_{k,j} - \frac{t_k}{|B_{k,j}|}\nabla f_{j+1}(x_{k,j})$
7: **fim para**
8: $x_{k+1,0} \leftarrow x_{k,J}$
9: **fim para**

Considere f_j convexa, para $j = 1, \ldots, m$, e $\{x_k\}$ como a sequência gerada pelo

método. Ao longo desta seção, $\mathbb{E}(z|x)$ representará o *valor esperado* da variável aleatória z, condicionado a x.

As hipóteses e teoremas apresentados a seguir foram adaptados de [16]. Em especial, a Hipótese 3.3 corresponde ao análogo da Hipótese 3.1 para o contexto estocástico.

Hipótese 3.2 (não enviesamento)**.** *Para todo $k \geq 0$, $\mathbb{E}(g_k|x_k) = \nabla f(x_k)$.*

Hipótese 3.3 (limitação do gradiente)**.** *Existe $L > 0$ tal que, para todo $k \geq 0$, $\mathbb{E}(\|g_k\|^2|x_k) \leq L^2$.*

Teorema 3.2. *Suponha f convexa, Hipóteses 3.2 e 3.3 válidas e que f admita minimizador x_*. Seja $\{t_k\}$ a sequência de passos do Algoritmo 3.*

1. *Se*
$$t_k \to 0^+, \quad \sum_{k=0}^{\infty} t_k = \infty \quad e \quad \sum_{k=0}^{\infty} t_k^2 < \infty,$$
então
$$\lim_{k\to\infty} \mathbb{E}(f_k) = f(x_*).$$

2. *Se $t_k = t > 0$ para todo k, então*
$$\lim_{k\to\infty} \mathbb{E}(f_k) \leq f(x_*) + \frac{tm^2L^2}{2},$$
onde L é a constante dada na Hipótese 3.3.

Observe que o Teorema 3.2 constitui uma versão probabilística do Teorema 3.1, sendo sua demonstração inteiramente análoga àquela apresentada na seção anterior. Para mais detalhes, consulte [16].

3.3 Métodos com momento

Os métodos com *momento* são técnicas desenvolvidas para acelerar a convergência de algoritmos de otimização, sendo amplamente aplicadas em problemas de aprendizado de máquina e redes neurais [17]. O conceito é inspirado na física, onde um objeto em movimento tende a continuar na mesma direção, a menos que uma força externa o desvie.

Por exemplo, considere um corpo que se move de um ponto x_{k-1} para outro x_k com uma velocidade v_k. Ao atingir o ponto x_k, o corpo não muda bruscamente sua direção, uma vez que há uma inércia que o mantém seguindo na direção $x_k - x_{k-1}$. Em contrapartida, a iteração clássica $x_{k+1} = x_k - t_k g_k$ representa uma mudança abrupta na direção de $-g_k$.

O momento, portanto, é introduzido para suavizar esse processo de atualização, acumulando uma "velocidade" que leva em conta tanto o gradiente atual quanto a direção do gradiente anterior. Em vez de usar diretamente g_k, a atualização passa a ser feita com um "vetor de velocidade" que combina g_k com a velocidade anterior: $x_{k+1} = x_k - t_k v_k$, onde $v_k = (1-\beta)v_{k-1} + \beta g_k$, com $\beta \in [0,1]$ sendo um parâmetro e $v_0 = 0$ a velocidade inicial. Assim, o momento acelera as atualizações em direções estáveis e reduz oscilações em direções que variam rapidamente.

Em muitos casos, o uso do momento no algoritmo permite escapar de mínimos locais rasos e de regiões planas da função de perda, atingindo o ótimo global de forma mais rápida, como dito em [17]. Essa abordagem será aplicada na versão estocástica do método do gradiente espectral para treinamento de redes neurais, discutido no próximo capítulo.

4 Uma versão estocástica do método do gradiente espectral para o treinamento de redes neurais

Ao treinar uma rede neural, a taxa de aprendizado é, sem dúvida, um dos parâmetros mais críticos para alcançar uma boa performance, exigindo, portanto, uma calibração rigorosa. Nesta seção, associamos o conceito do passo espectral à taxa de aprendizado, com o objetivo de calibrar esse parâmetro dinamicamente. Para tanto, a partir deste ponto, a taxa de aprendizado t_k será renomeada como λ_k. O método apresentado foi proposto em [5].

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

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

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

$$(W, b)_{k,0} = (W, b)_{k-1,J}, \qquad (W, b)_{k,j+1} = (W, b)_{k,j} - \lambda_k \nabla L_{B_{k,j}}((W, b)_{k,j}),$$

para $j = 1, 2, \ldots, J$ e $k = 0, 1, \ldots$. O gradiente para estimar λ_k é definido do seguinte modo:

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

para $j = 0, 1, \ldots, J - 1$ e $g_{k,0} = 0$, onde β é uma constante pré-definida no intervalo $[0, 1]$, que controla e suaviza o decaimento exponencial (efeito descrito na Seção 3.3). A diferença dos gradientes entre duas épocas é então definida por:

$$y_k = g_{k,J} - g_{k-1,J}.$$

A diferença entre os pontos de duas épocas, normalizada pelas iterações, é dada pela

diferença entre as duas últimas amostras de cada época:

$$s_k = J^{-1}((W,b)_{k,J} - (W,b)_{k-1,J}).$$

A versão estocástica do SPG para minimizar somas grandes de funções necessita lidar com lotes, assim como no método de gradiente estocástico. No SPG tradicional, o cálculo do passo espectral λ_k é feito sobre o gradiente completo da função objetivo (vide Algoritmo 1). Os autores de [5] sugerem, portanto, que os passos espectrais sejam calculados para cada lote individualmente:

$$\lambda_{k+1} = \frac{s_k^t s_k}{|s_k^t y_k|}.$$

É importante observar que tomamos o valor absoluto de $s_k^t y_k$, uma vez que, ao lidarmos apenas com uma amostra do gradiente dos dados, esse produto pode ser negativo, o que inviabilizaria o passo.

Conforme mencionado na Seção 3.2, se a hipótese usual de não enviesamento (Hipótese 3.2) for válida e se a taxa de aprendizado λ_k satisfizer a condição (3.8), então

$$\lim_{k\to\infty} \nabla L((W,b)_k) = 0,$$

indicando que o método converge.

Desse modo, para assegurar a convergência do método com passo espectral, uma condição suficiente é que a sequência de taxas de aprendizado atenda à condição (3.8). Portanto, [5] propõe a seguinte salvaguarda para λ_k:

$$\bar{\lambda}_k = \begin{cases} \lambda_k, & \text{se } \lambda_k \in \left[\frac{\tau_{\min}}{k+1}, \frac{\tau_{\max}}{k+1}\right], \\ \frac{\tau_0}{k+1}, & \text{caso contrário,} \end{cases} \tag{4.1}$$

onde k é o índice da época, e $\tau_{\min}$, $\tau_{\max}$ e τ_0 são constantes pré-definidas. Evidentemente, a taxa de aprendizado λ_k determinada pela expressão (4.1) satisfaz a condição (3.8) necessária para a convergência. O método SPG com taxa de aprendizado adaptativa é descrito no Algoritmo 4.

Algoritmo 4 Método do gradiente espectral estocástico

Entrada: máximo de épocas K, passos por época J, tamanho do mini-lote $|B|$, parâmetro de peso $\beta \in (0, 1]$, pesos iniciais $(W, b)_{0,0}$, taxa de aprendizado $\lambda_0 = \lambda_1$, τ_0, $\tau_{\min}$ e $\tau_{\max}$.

1: **para** $k \leftarrow 0$ até $K - 1$ **faça**
2: **se** $k > 1$ **então**
3: $y_{k-1} \leftarrow g_{k-1,J} - g_{k-2,J}$
4: $s_{k-1} \leftarrow \frac{1}{J}\left((W, b)_{k-1,J} - (W, b)_{k-2,J}\right)$
5: $\lambda_k \leftarrow \frac{s_{k-1}^t s_{k-1}}{|s_{k-1}^t y_{k-1}|}$
6: $\bar{\lambda}_k = \begin{cases} \lambda_k & \text{se } \lambda_k \in \left[\frac{\tau_{\min}}{k+1}, \frac{\tau_{\max}}{k+1}\right], \\ \frac{\tau_0}{k+1}, & \text{caso contrário.} \end{cases}$
7: **fim se**
8: $g_{k,0} \leftarrow 0$
9: **para** $j \leftarrow 0$ até $J - 1$ **faça**
10: Construa um lote $B_{k,j}$ aleatoriamente de tamanho $|B|$ e de maneira uniforme
11: $(W, b)_{k,j+1} \leftarrow (W, b)_{k,j} - \bar{\lambda}_k \nabla L_{B_{k,j}}((W, b)_{k,j})$
12: $g_{k,j+1} \leftarrow (1 - \beta) g_{k,j} + \beta \nabla L_{B_{k,j}}((W, b)_{k,j})$
13: **fim para**
14: $(W, b)_{k+1,0} \leftarrow (W, b)_{k,J}$
15: **fim para**

Exceto pela aplicação do momento no Algoritmo 4 (linha 12), as salvaguardas para a taxa de aprendizado calculadas na linha 6 do Algoritmo 4 garantem que $\bar{\lambda}_k$ satisfaz as condições do passo decrescente estabelecidas no item (a) do Teorema 3.2. Portanto, a convergência do método é análoga àquela descrita no Teorema 3.2. Para mais detalhes, consulte [5].

5 Testes numéricos

Neste capítulo apresentamos uma comparação do desempenho dos diferentes métodos citados neste trabalho, além de exibir os resultados comparativos entre os métodos do gradiente estocástico (denominado SGD) e o método SPG estocástico (denominado de BB), com e sem momento.

Neste trabalho, todos os algoritmos foram implementados em linguagem Julia [18] e os experimentos computacionais foram realizados 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® Xeon® Silver 4114 CPU @ 2.20 GHz 10 núcleos (20 threads), 160 Gb RAM.

Algoritmos podem ser comparados por diversas métricas, que aqui supomos ser positiva e obedecer ao princípio "menor é melhor". O *tempo de execução* é uma métrica útil para comparar algoritmos de diferentes naturezas, mas requer cautela na análise do tempo de CPU dado que paralelismo, processos concorrentes no processador e oscilações naturais no tempo de execução afetam sua acurácia. O *número de iterações* é útil apenas quando o custo computacional por iteração é similar, como em algoritmos de pontos interiores primal-dual, seguidor de caminhos e preditor-corretor. Em contrapartida, métodos como o gradiente e Newton, cujas iterações possuem custos distintos, não devem ser comparados diretamente por essa métrica, uma vez que o gradiente pode demandar mais iterações, mas ainda assim ser mais rápido. Por fim, o *número de avaliações da função objetivo* é uma métrica adequada à algoritmos de baixo custo para resolução de problemas de grande porte, pois o custo de avaliar funções ou gradientes é significativo. Por isso, a última é a métrica escolhida. No contexto deste trabalho, essa medida é apropriada pois reflete diretamente a eficiência do algoritmo de busca linear: menos avaliações da função indica uma busca linear mais eficiente.

Uma forma fácil e intuitiva de comparar diferentes objetos de estudo é de maneira visual. Neste contexto, entram os *perfis de desempenho* propostos por Dolan e Moré [6]. Tais perfis são muito utilizados em comparações de algoritmos, especialmente de

otimização contínua.

Para este trabalho, foram selecionados da biblioteca CUTEst [7] problemas irrestritos e com limitantes nas variáveis (restrições de "caixa"), totalizando 441 problemas. Por motivos de incompatibilidade ou mal funcionamento, foram removidos 9 problemas dos testes, sendo eles: BLEACHNG; PRICE4; BA-L16LS; BA-L21LS; BA-L49LS; BA-L52LS; BA-L73LS; JIMACK; RAYBENDS. Portanto, os testes foram realizados com os 432 problemas restantes.

Em todos os métodos, o ponto inicial x_0 foi obtido diretamente dos dados do problema e, caso não disponível, definimos como a origem, ou seja, $x_0 = (0, 0, \ldots, 0) \in \mathbb{R}^n$. Definimos os limites $\lambda_{\max} = 10^{30}$ e $\lambda_{\min} = 10^{-30}$, assim como considerado em [1], para o passo espectral λ_k nos métodos SPG, ABB e ABBmin. A convergência dos métodos é declarada pelo critério de parada $\|\nabla f(x_k)\|_\infty < 10^{-6}$ e é considerado um número máximo de iterações de 10^4.

Para a busca linear, tomando como referência [14], utilizamos os seguintes parâmetros:

$$M = 100, \quad \sigma_1 = 0.1, \quad \sigma_2 = 0.9, \quad \eta = 10^{-4}.$$

No método ABB, foi considerado $\kappa = 0.5$, seguindo [2]. Para o método ABBmin, consideramos $\kappa = 0.8$, de acordo com [3], e $m = 10$. No método de Dai-Kou, usamos ρ_k^{BB3} como descrito em (2.24).

O parâmetro $\lambda_0 \in [\lambda_{\min}, \lambda_{\max}]$ é arbitrário, e é inicializado como sugerido em [1]:

$$\lambda_0 = \begin{cases} \min\left\{\lambda_{\max}, \max\left\{\lambda_{\min}, \frac{1}{\|\nabla f(x_k)\|_\infty}\right\}\right\}, & \text{se } \|\nabla f(x_k)\|_\infty > 0, \\ \lambda_{\max}, & \text{caso contrário.} \end{cases}$$

A Tabela 1 traz um recorte dos 50 maiores problemas considerados, onde **n** é o número de variáveis do problema, **it** é o número de iterações, **st** é o status do problema (0: resolvido; 1: não resolvido) e **nf** é o número de avaliações de f.

Visando uma comparação global dos métodos em todos os problemas considerados, a Figura 8 apresenta o perfil de desempenho dos quatro algoritmos, baseado no número de avaliações de f, para os problemas avaliados. A escala do eixo τ é logarítmica para melhor visualização próximo de $\tau = 1$.

		ABB			ABBmin			Dai-Kou			SPG		
Problema	**n**	**it**	**st**	**nf**	**it**	**st**	**nf**	**it**	**st**	**nf**	**it**	**st**	**nf**
YATP1CLS	123200	46	0	47	18	0	19	89	0	105	34	0	37
YATP1LS	123200	46	0	47	18	0	19	89	0	105	34	0	37
YATP2CLS	123200	11	0	120	11	0	120	2000	0	2303	11	0	120
YATP2LS	123200	11	0	120	11	0	120	2030	0	2317	11	0	120
CYCLIC3LS	100002	885	0	1131	50000	1	50367	50000	1	50150	8251	0	9737
DEGTRID	100001	104	0	105	123	0	124	155	0	157	135	0	136
DEGTRID2	100001	4	0	5	17	0	18	1	0	2	4	0	5
INDEFM	100000	50000	1	5258345	50000	1	59558	7733	0	10460	6487	0	95844
OSCIGRAD	100000	85	0	86	90	0	91	243	0	274	84	0	85
BOXPOWER	20000	2547	0	2770	2167	0	2223	640	0	699	492	0	628
MODBEALE	20000	50000	1	50471	480	0	484	50000	1	50523	4705	0	5604
BOX	10000	32	0	53	22	0	37	73	0	112	44	0	146
DIXON3DQ	10000	11350	0	11433	9789	0	9841	13033	0	13035	19429	0	23761
POWER	10000	1021	0	1033	785	0	788	452	0	493	1762	0	1892
SPARSQUR	10000	30	0	31	30	0	31	65	0	81	30	0	31
FMINSRF2	5625	863	0	868	1150	0	1152	449	0	450	1111	0	1142
FMINSURF	5625	1708	0	1713	1455	0	1462	626	0	627	2232	0	2347
NCB20	5010	10660	0	13653	1366	0	2762	1407	0	1427	11418	0	14764
ARWHEAD	5000	3	0	4	3	0	4	12	0	28	3	0	4
BDEXP	5000	15	0	16	15	0	16	20	0	21	15	0	16
BDQRTIC	5000	66	0	67	85	0	86	801	0	824	73	0	74
BROYDN3DLS	5000	91	0	92	37	0	38	79	0	87	103	0	104
BROYDN7D	5000	2501	0	13266	3523	0	16811	1865	0	1869	2653	0	2794
BROYDNBDLS	5000	42	0	44	181	0	182	64	0	74	73	0	78
BRYBND	5000	42	0	44	181	0	182	64	0	74	73	0	78
CRAGGLVY	5000	297	0	410	177	0	178	159	0	174	721	0	1563
DQDRTIC	5000	27	0	28	15	0	16	55	0	63	26	0	27
DQRTIC	5000	49	0	50	49	0	50	53	0	80	49	0	50
ENGVAL1	5000	34	0	35	31	0	32	42	0	52	30	0	31
FLETBV3M	5000	343	0	2714	6677	0	43955	1038	0	2079	206	0	296
FLETCBV2	5000	0	0	1	0	0	1	0	0	1	0	0	1
FREUROTH	5000	143	0	147	41	0	45	121	0	133	222	0	340
GENHUMPS	5000	6747	0	47333	6242	0	53798	669	0	746	20765	0	41515
LIARWHD	5000	46	0	48	44	0	45	85	0	110	50	0	77
MOREBV	5000	91	0	104	118	0	131	175	0	178	89	0	111
NCB20B	5000	4462	0	5371	6422	0	7688	5461	0	5477	7166	0	9101
NONCVXU2	5000	50000	1	55731	38725	0	47443	11199	0	11202	50000	1	59868
NONDIA	5000	21	0	22	23	0	24	49	0	68	10	0	13
NONDQUAR	5000	3623	0	3659	4911	0	4951	4010	0	4046	5257	0	6620
NONSCOMP	5000	49	0	50	46	0	47	59	0	66	78	0	92
POWELLSG	5000	197	0	198	337	0	341	176	0	185	134	0	135
QUARTC	5000	49	0	50	49	0	50	53	0	80	49	0	50
SCHMVETT	5000	58	0	60	61	0	63	57	0	59	76	0	78
SINQUAD	5000	193	0	635	114	0	555	144	1	3122	171	0	521
SPARSINE	5000	17177	0	17356	13773	0	13861	50000	1	50006	50000	1	62064
SROSENBR	5000	26	0	28	17	0	19	42	0	51	36	0	38
SSBRYBND	5000	50000	1	51517	50000	1	50230	12049	0	12089	50000	1	58704
TESTQUAD	5000	23803	0	24043	13535	0	13638	4107	0	4126	50000	1	61883
TOINTGSS	5000	24	0	25	19	0	20	2	0	4	26	0	27
TQUARTIC	5000	29	0	133	26	0	126	77	0	92	496	0	1391

Tabela 1: Recorte dos resultados numéricos obtidos nos testes com os quatro métodos.

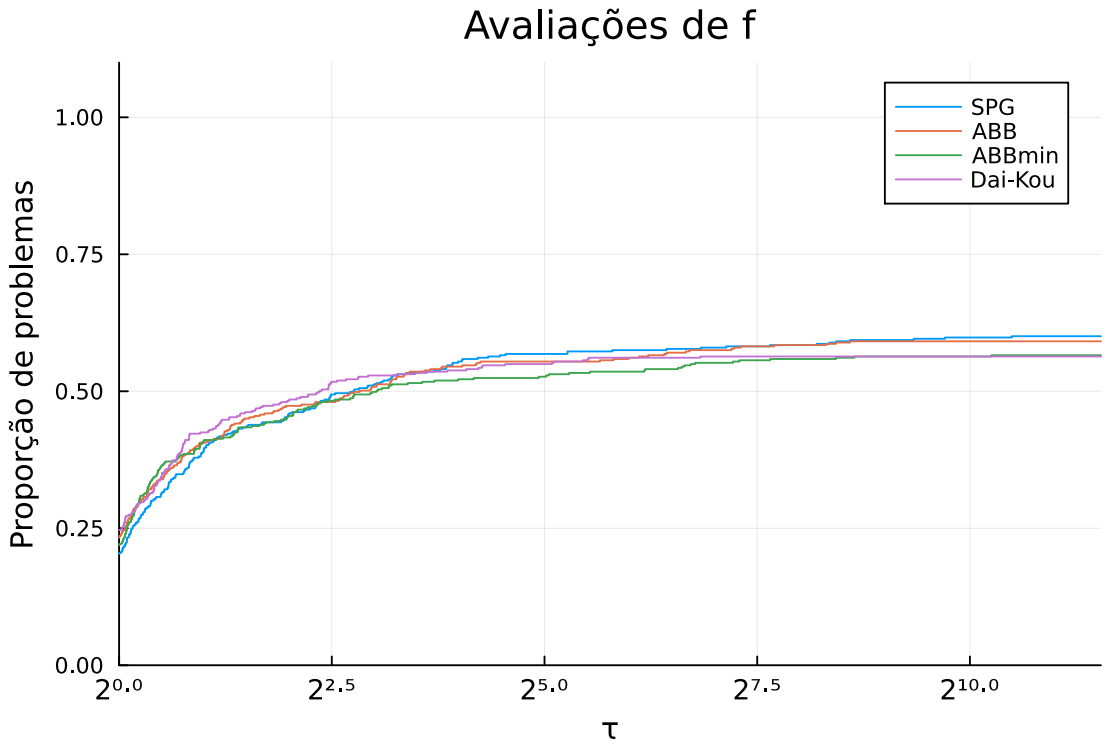

Figura 8: Comparação de desempenho dos algoritmos SPG, ABB, ABBmin e Dai-Kou nos 432 problemas considerados.

Tome como nota que, para as comparações realizadas neste trabalho, um método é considerado "robusto" quando resolve muitos problemas, e considerado "eficaz" quando resolve problemas com rapidez.

Nesse contexto, do total de 432 problemas analisados, o método SPG resolveu 259, sendo o mais robusto dentre os quatro métodos apresentados. No entanto, sua eficiência foi inferior comparado aos demais. Os métodos ABB e ABBmin apresentaram eficácia semelhante, contudo, ABB foi mais robusto, resolvendo 255 problemas contra 244 do ABBmin. Este é um destaque negativo para o método ABBmin, já que seu objetivo era melhorar seu predecessor, ABB. Por último, o método Dai-Kou se destacou como o mais eficaz, embora tenha sido menos robusto, resolvendo 243 problemas.

O conjunto de dados utilizado para os testes foi o *dataset* MNIST [8] (do inglês, *Modified National Institute of Standards and Technology*). Este conjunto é amplamente utilizado em testes de algoritmos para aprendizado de máquina supervisionado. Ele contém 70 mil imagens em escala de cinza de dígitos manuscritos, numerados de 0 a 9, sendo 60 mil dessas imagens destinadas ao treinamento de modelos e 10 mil reservadas para testes. Cada imagem tem uma resolução de 28×28 pixels, o que resulta em uma matriz de 784 valores de intensidade que variam de 0 (preto) a 255 (branco), representando

o nível de preenchimento de cada pixel. Na Figura 9 são exibidos exemplos das imagens contidas no MNIST.

Figura 9: Exemplos de dígitos presentes no MNIST.

O MNIST tornou-se um dos conjuntos de dados mais populares devido à sua simplicidade e à facilidade com que modelos podem ser treinados sobre ele, o que o transforma em uma excelente base para iniciantes e pesquisadores na área de inteligência artificial e aprendizado profundo. Ele também oferece um desafio balanceado entre complexidade e viabilidade de solução, já que, embora os dígitos sejam simples, suas variações manuscritas geram uma variedade de formas e tamanhos, tornando o conjunto adequado para testar algoritmos de classificação como redes neurais convolucionais (CNNs) e outros modelos baseados em aprendizado supervisionado.

Para o método proposto, os parâmetros de referência são os seguintes:

$$\beta = 4/K, \quad \tau_0 = 1, \quad \tau_{\min} = 10^{-5}, \quad \tau_{\max} = 3,$$

onde β e τ_0 são dados em [5]. Os parâmetros $\tau_{\min}$ e $\tau_{\max}$ foram obtidos a partir de testes de calibragem. O tamanho do mini-lote foi definido como $|B| = 128$ e a taxa de aprendizado inicial $\lambda_0 = 0.1$, seguindo [5]. Como este conjunto de dados é relativamente simples, temos uma camada de dados de entrada, uma camada para os dados de saída e apenas uma camada interna, ou seja, $L = 3$. Portanto, temos 784 neurônios para a camada de entrada, 20 neurônios para a camada interna e 10 neurônios para a camada de saída.

Foram implementados quatro algoritmos diferentes: SGD com e sem momento, e BB com e sem momento. Foram realizadas cadeias de 5 testes numéricos para cada algoritmo, registrando seus dados em um *DataFrame*. Com os dados, exibimos os resultados de perda e a porcentagem de acertos sobre a melhor rodada de cada algoritmo.

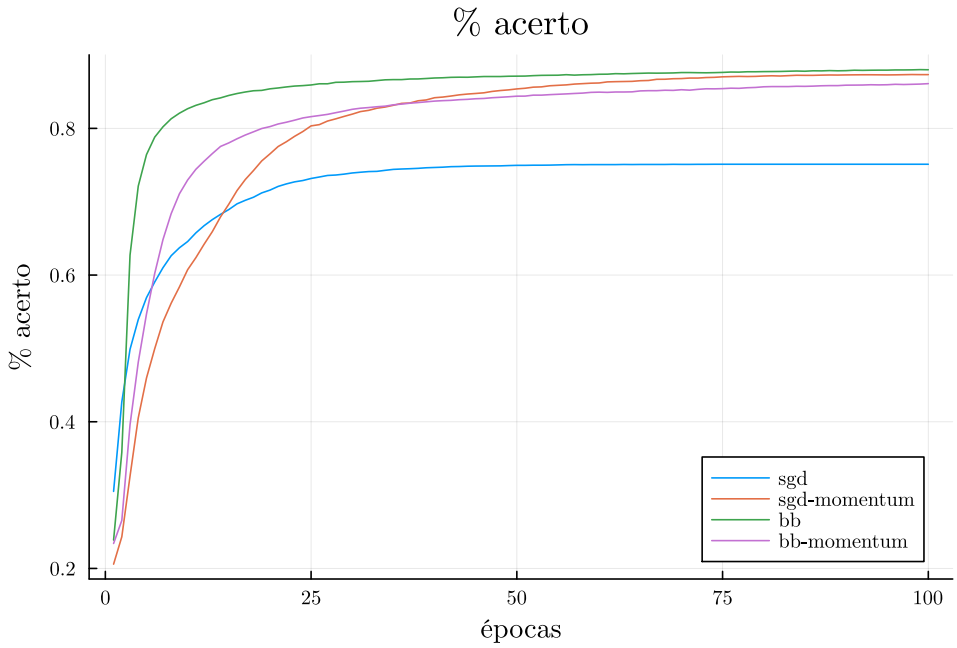

Figura 10: Porcentagem de acertos.

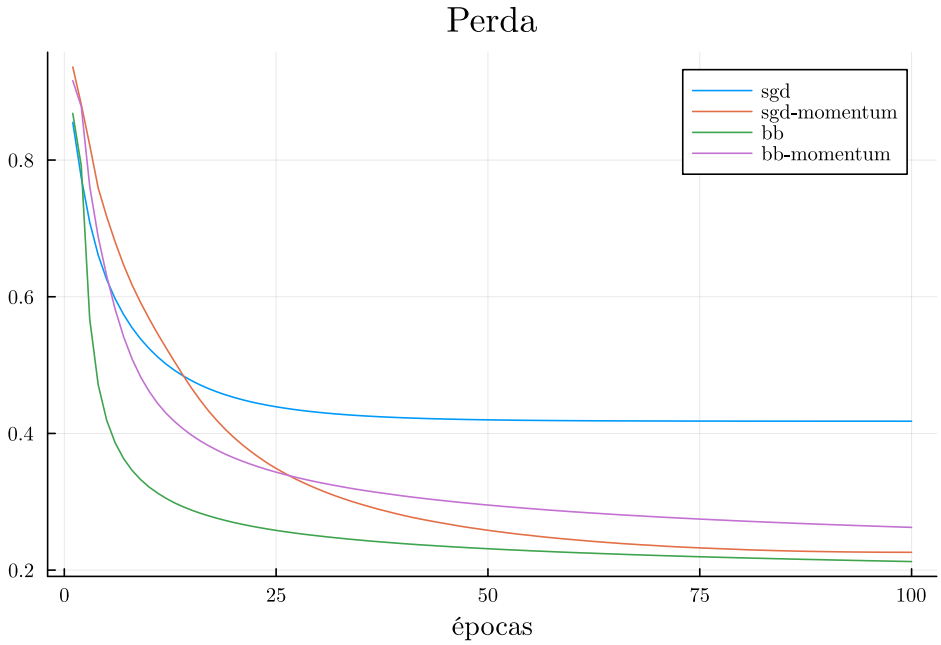

Figura 11: Risco Empírico.

Nas Figuras 10 e 11, é possível observar de forma clara a superioridade dos métodos

que utilizam momento em comparação ao SGD sem momento. Este último apresentou um desempenho significativamente inferior, confirmando a ineficiência do método em alcançar os mesmos níveis de precisão e convergência que os demais algoritmos. Embora todos os métodos com momento tenham exibido resultados semelhantes, o método BB se sobressaiu, mostrando uma melhor performance no geral.

Além disso, a análise revela uma notável melhora no desempenho geral dos algoritmos ao incorporar o uso de momento, com destaque para o método SGD, que apresentou uma evolução expressiva na convergência. Esse comportamento ressalta a importância do uso de técnicas que aceleram a convergência e estabilizam o processo de otimização em problemas de aprendizagem.

Conclusões

Neste trabalho, foram considerados métodos para minimização irrestrita e com restrições convexas simples, com foco no método do gradiente espectral projetado (SPG). Foi desenvolvido um estudo teórico básico do método, apresentando suas características de boa definição e convergência global. Discutiu-se ainda métodos derivados da ideia do passo espectral presente no SPG, que buscam melhores resultados numéricos, como é o caso dos métodos *Adaptive Barzilai-Borwein* (ABB), uma versão alternativa denominada ABBmin e o método de gradientes conjugados tipo Barzilai-Borwein (Dai-Kou).

Ao longo da pesquisa, foram realizados testes numéricos com os algoritmos apresentados. Os testes indicaram que o método SPG é, no que diz respeito ao número de avaliações de função, menos eficiente quando comparado aos outros, porém, ligeiramente mais robusto, ou seja, foi o que resolveu mais problemas. Já com o método Dai-Kou, temos o caso contrário: é superior em eficácia, mas menos robusto, ficando atrás de todos os algoritmos neste quesito. Ademais, os métodos ABB e ABBmin apresentam resultados semelhantes em eficiência, mas ABB é mais robusto.

Apresentou-se uma aplicação da ideia espectral no treinamento de redes neurais recentemente sugerida na literatura [5]. A ideia de passos espectrais, consolidada com o trabalho sobre o SPG [1], tem sido cada vez mais empregada em otimização e, com o avanço da inteligência artificial, mostra-se também útil. Assim, este trabalho busca evidenciar essa vertente inovadora e promissora no contexto do aprendizado de máquina supervisionado. Para tanto, foram resgatados alguns tópicos preliminares, como o conceito de redes neurais, os métodos do gradiente incremental e do gradiente estocástico, e também a ideia geral de métodos com momento.

Testes foram realizados sobre um problema clássico da literatura em treinamento de redes neurais, apresentando resultados promissores frente à técnicas consolidadas no ramo.

Referências Bibliográficas

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

2 ZHOU, B.; GAO, L.; DAI, Y.-H. Gradient methods with adaptive step-sizes. *Computational Optimization and Applications*, v. 35, p. 69–86, 2006.

3 FRASSOLDATI, G.; ZANNI, L.; ZANGHIRATI, G. New adaptive stepsize selections in gradient methods. *Journal of Industrial and Management Optimization*, v. 4, p. 299–312, 2008.

4 DAI, Y.-H.; KOU, C.-X. A Barzilai-Borwein conjugate gradient method. *Science China Mathematics*, v. 59, p. 1511–1524, 2016.

5 LIANG, J. et al. Barzilai–Borwein-based adaptive learning rate for deep learning. *Pattern Recognition Letters*, v. 128, p. 197–203, 2019.

6 DOLAN, E. D.; MORé, J. J. Benchmarking optimization software with performance profiles. *Mathematical Programming*, v. 91, p. 201–213, 2002.

7 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*, v. 60, n. 3, p. 545–557, 2015.

8 DENG, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, IEEE, v. 29, n. 6, p. 141–142, 2012.

9 MARTíNEZ, J. M.; SANTOS, S. A. *Métodos Computacionais de Otimização*. [S.l.]: IMECC - UNICAMP, 1995.

10 KARAS, E. W.; RIBEIRO, A. A. *Um Curso de Otimização*. [S.l.: s.n.], 2010.

11 KARAS, E. W.; RIBEIRO, A. A. *Otimização Contínua - Aspectos Teóricos e Computacionais*. [S.l.]: Cengage, 2014.

12 NOCEDAL, J.; WRIGHT, S. J. *Numerical Optimization*. [S.l.]: Springer Series in Operations Research, 1999.

13 BARZILAI, J.; BORWEIN, J. M. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, v. 8, p. 141–148, 1988.

14 BIRGIN, E. G.; MARTíNEZ, J. M.; RAYDAN, M. Algorithm 813: SPG—software for convex-constrained optimization. *ACM Transactions on Mathematical Software*, v. 27, p. 340–349, 2001.

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

16 BECK, A. *First-Order Methods in Optimization*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2017. Disponível em: <https://epubs.siam.org/doi/abs/10.1137/1.9781611974997>.

17 SUTSKEVER, I. et al. On the importance of initialization and momentum in deep learning. In: DASGUPTA, S.; MCALLESTER, D. (Ed.). *Proceedings of the 30th International Conference on Machine Learning*. Atlanta, Georgia, USA: PMLR, 2013. (Proceedings of Machine Learning Research, 3), p. 1139–1147. Disponível em: <https://proceedings.mlr.press/v28/sutskever13.html>.

18 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>.