INSTITUTO FEDERAL DA PARAÍBA
CAMPUS CAJAZEIRAS
CURSO DE LICENCIATURA EM MATEMÁTICA

JOSÉ JORGE DE SOUZA SILVA

O ALGORITMO DE RETROPROPAGAÇÃO

CAJAZEIRAS
2022

JOSÉ JORGE DE SOUZA SILVA

O ALGORITMO DE RETROPROPAGAÇÃO

Monografia apresentada junto ao **Curso de Licenciatura em Matemática** do **Instituto Federal da Paraíba**, como requisito à obtenção do título de **Licenciado em Matemática**.

Orientador:
Prof. Dr. Vinicius Martins Teodosio Rocha.

Cajazeiras
2022

JOSÉ JORGE DE SOUZA SILVA

O ALGORITMO DE RETROPROPAGAÇÃO

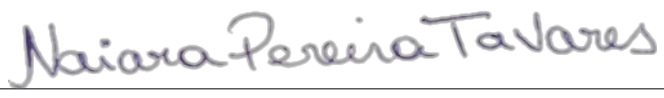
Monografia apresentada ao programa de **Curso de Licenciatura em Matemática** do **Instituto Federal da Paraíba**, como requisito à obtenção do título de **Licenciado em Matemática**.

Data de aprovação: 06/05/2022

Banca Examinadora:

Prof. Dr. Vinicius Martins Teodosio Rocha
Instituto Federal da Paraíba - IFPB

Profa. Ma. Kissia Carvalho
Instituto Federal da Paraíba - IFPB



Profa. Esp. Naiara Pereira Tavares
Instituto Federal da Paraíba - IFPB

IFPB / Campus Cajazeiras
Coordenação de Biblioteca
Biblioteca Prof. Ribamar da Silva
Catalogação na fonte: Suellen Conceição Ribeiro CRB-2218

S586a Silva, José Jorge de Souza

O algoritmo de retropropagação / José Jorge de Souza Silva. – Cajazeiras/PB: IFPB, 2022.

67f.:il.

Trabalho de Conclusão de Curso (Graduação em Matemática) - Instituto Federal de Educação, Ciência e Tecnologia da Paraíba-IFPB, Campus Cajazeiras. Cajazeiras, 2022.

Orientador(a): Prof. Dr. Vinicius Martins Teodosio Rocha.

1. Matemática. 2. Algoritmo. 3. Redes Neurais. 4. Retropropagação. 5. Cálculo.

I. Silva, José Jorge de Souza. II. Titulo.

CDU: 51 S586a

Dedico esse trabalho a mim, por todo esforço, coragem e persistência durante toda a minha trajetória.

AGRADECIMENTOS

Agradeço à mim mesmo, por me manter firme perante a tantos desafios.

Agradeço à minha mãe, Vanda Lúcia de Souza Silva, por ter me dado o apoio inicial necessário.

Agradeço aos amigos que fiz no curso, por terem me proporcionado vivências tão especiais.

Agradeço à minha amiga Larissa Soares, por tantas emoções vivenciadas e pela relação de parceria e irmandade, tanto nos momentos de diversão quanto nos de aflição e dificuldade.

Agradeço ao professor Vinicius Martins Teodosio Rocha pela orientação, e às professoras Kissia Carvalho e Naiara Pereira, por terem aceito o convite para compor a banca de avaliação desse trabalho.

Agradeço ao Instituto Federal da Paraíba (IFPB) - campus Cajazeiras, lugar onde passei um período marcante da minha vida no qual amadureci e aprendi muito, não só em relação ao curso, com os excelentes professores, mas também enquanto ser humano.

Por fim, agradeço a todos que contribuíram direta ou indiretamente para que eu chegasse até aqui.

"No fim tudo dá certo, e se não deu certo é porque ainda não chegou ao fim."

Fernando Sabino

RESUMO

O crescente campo da inteligência artificial pode ser visto como uma área de estudos que tenta simular a inteligência humana em máquinas. Variados modelos são utilizados nesse propósito, destacam-se entre eles as redes neurais artificiais que, inspiradas nas redes neurais biológicas, buscam reproduzir a aprendizagem por meio do reconhecimento de padrões. Nelas se insere o algoritmo de retropropagação, uma ferramenta balizada por conceitos do cálculo, para fazê-las aprenderem. Desse modo, considerando a forma pouco atenciosa como a parte matemática do algoritmo é comumente abordada pelas referências disponíveis, decidimos por seguir com essa temática. Assim, nos norteamos a partir do seguinte questionamento: Como o Cálculo Diferencial torna o aprendizado do algoritmo de retropropagação possível? Destarte, o presente trabalho tem como objetivo geral descrever o algoritmo de retropropagação. Para isso, realizamos uma pesquisa bibliográfica, com uma abordagem qualitativa e de natureza básica na qual, para contextualizar o algoritmo, apresentamos a construção de uma rede neural e explicamos o seu funcionamento, em seguida exploramos os conceitos matemáticos utilizados nele como derivadas e regra da cadeia e, por fim, o expomos por meio de um exemplo. Outrossim, concluímos que o método de aproximação numérica gradiente descendente é a base que possibilita a aprendizagem desse algoritmo.

Palavras-chave: Retropropagação; Redes Neurais; Perceptron Multicamadas; Cálculo.

ABSTRACT

The expanding field of artificial intelligence can be thought as the field of study that aims to simulate human intelligence in machines. A number of models are applied to this purpose, we highlight the artificial neural networks, that gets inspiration from the biological neural network and seeks to reproduce the process of learning through behavior recognition. Included in the neural networks is the algorithm of backpropagation, a tool based on calculus concepts that allows those networks to learn. Therefore, taking in consideration the not so thoughtful way the mathematics concepts used in the algorithm are approached in the classical available references, we decided to explore this theme. We parted from the following question: How does the Differential Calculus makes the learning process of the backpropagation algorithm possible? In this sense, we overtook a bibliographical research, with a qualitative approach along which, in order to put the algorithm in context, we present the construction of neurals networks and explain their functioning, next we explore the mentioned math concepts, as derivatives and the chain rules and, at last, we expose the whole operation through an example. Lastly, we concluded that the gradient descent method is the basis that allows the learning process of the algorithm.

Keywords: Backpropagation; Neural Networks; Multilayer Perceptron; Calculus.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

INTRODUÇÃO

A Inteligência Artificial (IA) pode ser vista como um campo de estudo que tenta simular a inteligência humana em máquinas via *softwares*. Ela se insere na realidade da sociedade em diferentes situações e a maioria, por não ter conhecimento sobre o assunto, não percebe. Além de a utilizarmos, contribuimos para seu desenvolvimento ao fornecer dados diariamente ao utilizarmos a internet.

Nesse contexto estão inseridas as Redes Neurais Artificiais (RNAs), modelos de IA inspirados no cérebro humano com vasta aplicação em problemas de reconhecimento de padrões e previsões. A rede neural, por sí só, é um escopo. Para funcionar, ela necessita de um algoritmo para treiná-la a fazer determinada tarefa, um dos mais populares é o de retropropagação (*backpropagation*).

A base teórica de matemática que fundamenta esse algoritmo é visto nas disciplinas de Cálculo Diferencial e Integral no curso de Licenciatura em Matemática, em específico, regra da cadeia e vetor gradiente. Tendo em vista a forma como o tema é tratado na literatura disponível – onde o embasamento teórico que justifica a viabilidade do algoritmo é pouco explorado – e meu gosto pela disciplina Cálculo, unido a afinidade pela área de Matemática Computacional, decidimos por descrever o algoritmo de retropropagação de forma sistemática

Sendo assim, fomos norteados sob a perspcetiva da seguinte questão: **Como o Cálculo Diferencial torna o aprendizado do algoritmo de retropropagação possível?** Para respondê-la, em primeiro momento realizamos consultas de artigos e livros sobre o tema, para compreendermos o funcionamento de tal algoritmo e verificarmos nossa hipótese de que o vetor gradiente é a base de sua aprendizagem. Essas consultas foram realizadas por meio da base de dados Google Acadêmico e de forma livre no site de buscas Google.

Inicialmente realizamos leituras sobre a história das redes neurais, desde seu surgimento até o desenvolvimento do algoritmo de retropropagação. Em seguida, estudamos alguns conceitos de cálculo necessários para conseguir abstrair as equações e o algoritmo como um todo.

Posto isso, o obgetivo geral desse trabalho é descrever o algoritmo de retropropagação, e está atrelado aos seguintes objetivos específicos, contextualizar as redes neurais, apresentar o gradiente descendente enquanto método de otimização, descrever o perceptron de Rosenblatt e o multicamadas, e por fim, estudar a aprendizagem por retropropagação

como aplicação do gradiente descendente.

Nesse sentido, o presente trabalho trata-se de uma pesquisa bibliográfica com abordagem qualitativa, já que os resultados obtidos não podem ser quantificados (RODRIGUES et al., 2007). Quanto à finalidade, trata-se de uma pesquisa básica, já que objetiva "adquirir conhecimentos novos que contribuam para o avanço da ciência, sem que haja uma aplicação prática prevista" (FONTELLES et al., 2009, p.5).

Assim, esperamos que o trabalho contribua para a difusão do conhecimento sobre o algoritmo de retropropagação, dada a ausência de referências que abordem seu cálculo de forma detalhada. Ademais, nas pesquisas realizadas para a aquisição de embasamento, percebemos que grande parte das referências encontradas são em língua estrangeira, inglês. Desse modo, almejamos que a pesquisa possa tornar o conhecimento mais acessível para aqueles que pretendem começar os estudos nessa área e, também, proporcionar aos alunos de matemática a visão da aplicação de conceitos vistos nas disciplinas de cálculo em um problema real no contexto de uma área na qual estão, mesmo que inconscientemente, imersos.

Destarte, iniciamos o presente Trabalho de Conclusão de Curso (TCC) com uma contextualização das RNAs para situar o leitor no tema do trabalho, apresentamos o histórico, descrevemos o perceptron de Rosenblatt e, principalmente, o perceptron multicamadas. Em seguida, discutimos os conceitos matemáticos que respaldam a funcionalidade de algoritmo de retropropagação. Finalizamos apresentando o algoritmo de retropropagação, o cálculo que ele executa, e um exemplo aplicando os conceitos vistos.

1 REDES NEURAIS

A Inteligência Artificial (IA) teve seu surgimento na década de 50 e, embora não exista uma definição precisa, pode ser entendida como um ramo da ciência da computação que utiliza variadas técnicas e modelos de sistemas computacionais para desenvolver, em máquinas, a capacidade de pensar de forma inteligente para solucionar determinados problemas (SICHMAN, 2021).

A IA possui uma vasta aplicação que se estende da realização de diagnósticos médicos ao monitoramento de lavouras. Atualmente, visando a melhoria da experiência dos seus usuários, empresas de grande porte, como as proprietárias de serviços de streaming, realizam investimentos massivos nessa área. A sociedade está imersa nesse meio, desfrutando e contribuindo para sua melhoria ao fornecer dados que são úteis para o aprendizado dos algoritmos.

Nesse contexto estão inseridas as Redes Neurais Artificiais (RNAs) ou, simplismente, Redes Neurais (RN) como uma técnica de IA que tenta simular o aprendizado do cérebro humano por meio da extração e do reconhecimento de padrões a partir de grandes bases de dados. Esse modelo é constituído por unidades de processamento, os neurônios artificiais ou nodos, distribuídos em paralelo por meio de camadas[1] que estão interligadas por conexões unidirecionais, conhecidas como pesos sinápticos (BRAGA et al., 2007).

Uma RN recebe um padrão de entrada e produz uma saída, seu processo de aprendizagem consiste em encontrar os valores ideais para seus pesos que produzam uma resposta eficiente. Para isso, é necessário a utilização de um algoritmo que realize as correções necessárias nos seus valores, como o de retropropagação. Nesse contexto, existem duas formas de aprendizado para RNAs, o aprendizado supervisionado:

> um supervisor externo fornece à RNA a saída desejada em relação a um padrão de entrada. Com isso, é possível comparar a saída da RNA com a saída desejada, obtendo-se o erro referente à resposta atual (FLECK et al., 2016, p.52);

e o não supervisionado: não há uma saída desejada, a rede deve aprender a detectar padrões relevantes apenas com os dados de entrada.

Uma aplicação popular das redes neurais, através do aprendizado supervisionado, é o reconhecimento de dígitos manuscritos. Resumidamente, a rede é treinada sob uma

[1] Em redes neurais, camadas são conjuntos de neurônios que não se conectam. Eles estão distribuídos em paralelo, onde a execução de procedimentos lógicos se dá de forma sequencial de acordo com seus índices.

grande quantidade de exemplos de treino já classificados, neste caso, imagens de dígitos escritos a mão. Essas imagens são convertidas em dados numéricos e são apresentadas a rede, esta deve reconhecer qual o dígito presente na imagem. Essa aplicação é descrita em Nielsen (2015) que utiliza a retropropagação como algoritmo de treino.

Assim, nesse capítulo será realizada uma apresentação das redes neurais bem como o contexto histórico de sua criação e desenvolvimento. Enfatizaremos o perceptron de Rosenblatt e, posteriormente, o perceptron multicamadas, descrevendo aspectos como arquitetura e aprendizagem.

1.1 UM BREVE HISTÓRICO DAS RNAS

Podemos dizer que as discussões acerca das redes neurais tiveram sua primeira ascensão com o trabalho de McCulloch e Pitts (1943), os autores propuseram um modelo de neurônio artificial, ao realizarem uma discussão complexa sobre o sistema nervoso humano, com o intuito de modelar o aprendizado biológico. Posteriormente, Donald Heeb (1949), mostrou que o conceito de plasticidade era aplicável nas redes neurais. Ou seja, por meio da experiência envolvendo o processo de adaptação dos pesos sinápticos, era possível fazê-las aprenderem (BRAGA et al., 2007).

Esse fato levou, anos depois, a ideia de perceptron proposta por Rosenblatt (1958), uma topologia de rede com um algoritmo de aprendizado. Acrescido de sinapses ajustáveis, seu modelo poderia aprender a realizar classificações binárias em conjuntos de dados que fossem linearmente separáveis.

Os estudos acerca das redes neurais possuem variações de popularidade ao longo do tempo. O perceptron foi responsável por um grande aumento de pesquisas na área devido a euforia causada pela descoberta. Porém, sua limitação aos dados linearmente separáveis foi fruto de grandes críticas. Por exemplo, ele não consegue resolver o problema simples da função das portas lógicas do Ou Exclusivo (XOR): $f(0,0) = 0, f(1,1) = 0, f(0,1) = 1, f(1,0) = 1$.

> A solução para o problema XOR já era conhecida. Bastava acrescentar mais uma camada de neurônios na rede (uma camada escondida). O que faltava era um algoritmo que fosse capaz de treinar os pesos dessa rede multi-camada para que pudesse classificar corretamente problemas mais complexos (RAUBER, 2005, p.4).

Houve, então, a maior queda de popularidade e os estudos acerca das redes neurais entraram em declínio. Anos depois, "em 1980, a segunda onda de pesquisas em redes neurais emergiram em grande parte por meio do movimento conhecido como conexionismo ou processamento paralelo distribuido" (GOODFELLOW et al., 2016, p.16).

Nesse contexto surgiu o algoritmo de retropropagação descrito em Rumelhart et al. (1985). Os autores apresentaram o método de propagação de erro na resolução de alguns problemas como o do XOR, que foi solucionado com a inclusão de uma camada oculta composta por dois neurônios. Os autores destacam que seus resultados não garantem uma solução geral para todos os problemas solucionáveis, mas que a técnica da propagação de erro proporciona uma solução para quase todos os casos.

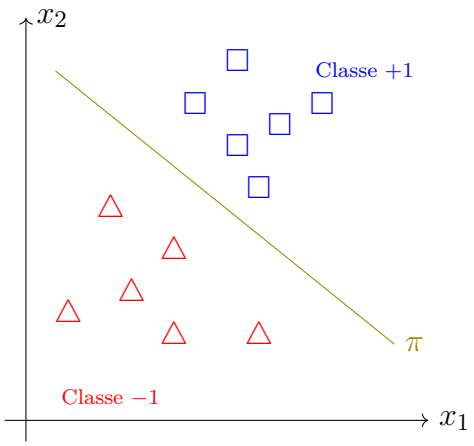
1.2 O PERCEPTRON

O perceptron, proposto por Rosenblatt, é uma arquitetura de rede neural com uma camada de entrada, composta por um número n de neurônios e uma camada de saída que possui apenas um. Como já mecionado, o perceptron funciona em problemas de classificação binária sob conjuntos de dados linearmente separáveis. Vejamos a Definição 1.2.1.

Definição 1.2.1 Um hiperplano π em $\mathbb{R}^n$ é um conjunto definido pela equação algébrica $w_0 + w_1x_1 + \ldots w_nx_n = 0$ com $w_0, \ldots, w_n \in \mathbb{R}$.

Nesse sentido, entendemos por conjunto linearmente separável um conjunto que possui dois subconjuntos representados por duas classes distintas que podem ser separados por um hiperplano. No caso de duas entradas podemos associar o hiperplano a uma reta e o conceito de separação pode ser compreendido como os lados opostos dela, observemos a Figura 1.1. O caso de três entradas é semelhante, no entanto, o hiperplano passa a ser um plano. Em dimensões maiores o conceito se torna mais abstrato já que não podemos associá-lo a uma interpretação geométrica, assim, podemos entendê-lo como uma estrutura linear de n dimensões que divide o $\mathbb{R}^n$ em duas partes.

Figura 1.1 – Hiperplano em $\mathbb{R}^2$.



Fonte: Elaborado pelo autor.

O algoritmo do perceptron se inicia com um vetor de pesos $w = (w_0, w_1, \ldots, w_n)$ de valores aleatórios e realiza uma soma ponderada com um vetor de entrada $\overline{x} = (1, x_1, \ldots, x_n)$ que está associado a um exemplo de treino. Desse modo, $a = w_0 \cdot 1 + w_1 x_1 + \cdots + w_n x_n$, fazendo $1 = x_0$, compactamos a soma em

$$a = \sum_{i=0}^{n} w_i x_i. \tag{1}$$

A razão pela qual primeiro valor do vetor de entrada é fixo, igual a 1, se deve ao fato de que, desse modo, o hiperplano se desloca da origem e tem mais chances de conseguir realizar a classificação dos dados corretamente. É fácil perceber isso quando associamos ao hiperplano em $\mathbb{R}^2$, ou seja, quando a rede neural possui duas entradas[2].

A constante de valor $w_0 x_0$ é chamada de *bias* e, na literatura, também é denotada por b, tornando $a = \sum_{i=1}^{n} w_i x_i + b$. Com o objetivo de generalizar o conteúdo para o tópico da rede neural multicamadas, que será visto na seção posterior, optamos por fixar a notação como mostrado em (1).

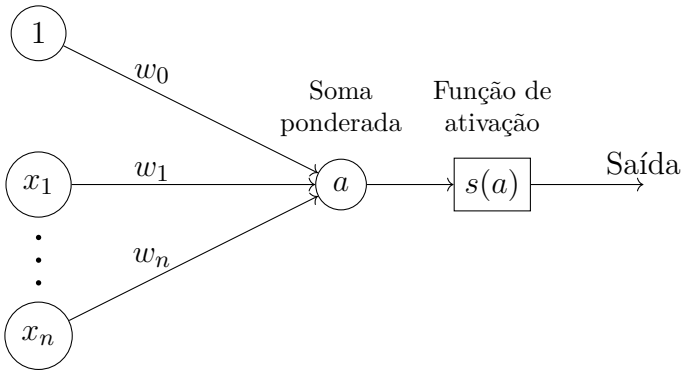
Após computada, a passa por uma função de ativação. Nas redes neurais em geral, essa função é essencial para a funcionalidade dos algoritmos de aprendizagem. Ela serve para reduzir a não-linearidade das respostas dos neurônios de saída e, assim, padronizá-las.

A função de ativação utilizada pelo perceptron é a função sinal

$$s(u) = \begin{cases} 1, & \text{se } u > 0 \\ -1, & \text{se } u < 0 \end{cases}, \ u \in \mathbb{R}.$$

A Figura 1.2 ilustra a dinâmica do algoritmo.

Figura 1.2 – Perceptron de Rosenblatt.



Fonte: Elaborado pelo autor.

O treinamento é realizado sob conjunto previamente classificado $D = \{\overline{x}_i, y_i\}_{i=1}^{m}$ onde $\overline{x}_i$ e y_i são, respectivamente, o vetor de entrada do i-ésimo exemplo de treino e a sua

[2] Contamos o número de entrada excluindo a de valor 1, já que ela é fixa.

classe, sendo y_i igual a $+1$ ou -1. A aprendizagem do perceptron é online, isso quer dizer que a adaptação dos pesos acontece sempre que $\overline{x}$ é classificado incorretamente. Assim, o vetor de pesos é atualizado seguindo a seguinte regra

$$w_i(t+1) = w_i(t) + y_i\overline{x}_i$$

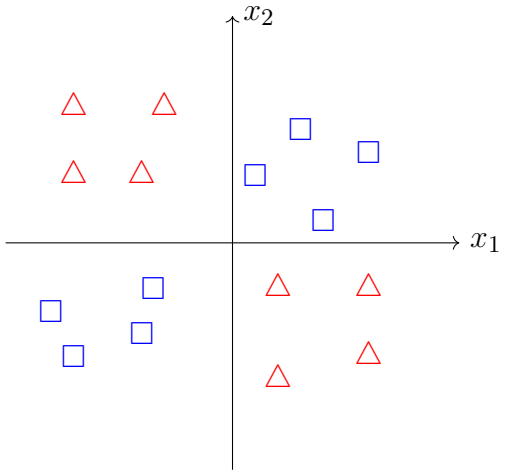
onde t é a iteração — a interpretação geométrica no caso de duas dimensões é descrito em Silva et al. (2021).

Dizemos que o perceptron converge quando todo o conjunto D é classificado corretamente. Sempre que D for linearmente separável, a convergência é garantida pelo Teorema 1.2.1. É importante frizar que, geralmente, o hiperplano encontrado após uma única passagem dos exemplos de treino não é o ideal, assim, é necessário percorrer D mais vezes, essa passagem é chamada de época.

Teorema 1.2.1 (Teorema da Convergência do Perceptron) Seja $X = X_+ \cup X_- \subset \mathbb{R}^n$ finito com X_+ e X_- linearmente separáveis. Então Perceptron de Rosenblatt converge (no sentido de não haver mais atualizações) após uma quantidade $n_{\max}$ de iterações, onde $n_{\max}$ é uma constante dependente do comprimento máximo dos vetores em X e do máximo das margens (distância mínima entre um hiperplano separador de X_+ e X_- e os pontos de X).

Como citado anteriormente, o problema XOR não pode ser resolvido pelo perceptron de camada única dado a não separabilidade linear dos seus dados. A Figura 1.3 ilustra o problema em uma perspectiva mais ampla que pode ser entendida da seguinte forma: se $(x_1 > 0 \wedge x_2 > 0) \vee (x_1 < 0 \wedge x_2 < 0)$, o ponto pertence a classe $+1$, caso contrário, pertence a -1.

Figura 1.3 – Problema XOR.



Fonte: Elaborado pelo autor.

Como podemos notar, não há um hiperplano que faça o algoritmo convergir. Nesse contexto, introduzimos o Perceptron Multicamadas (MLP) do inglês *Multilayer Perceptron*.

1.3 PERCEPTRON MULTICAMADAS (MLP)

Como visto na seção anterior, o perceptron de camada única se limita a dados linearmente separáveis. Assim, a rede MLP surge para superar essa limitação. Esse modelo possui mais de uma camada, que estão diretamente conectadas, com mais de um neurônio. Haykin (1999) destaca três pontos que caracterizam esse modelo: a implementação de funções de ativação não-lineares e diferenciáveis; a existência de camadas ocultas entre a de entrada e a de saída; e o elevado nível de conectividade entre os neurônios por meio dos pesos sinápticos. O autor também destaca que os neurônios das camadas ocultas tem por finalidade descobrir ou detectar características que possuem maior impacto no processo de treinamento.

Uma rede MLP possui um alto poder de processamento. Em suma, ela é capaz de extrair um conhecimento próprio a partir de um conjunto de dados de treinamento e generalizá-lo, aplicando em dados ainda não vistos, para realizar tarefas complexas. Tal conhecimento é armazenado nas conexões internas, os pesos sinápticos (HAYKIN, 1999).

Os dados passam pela camada de entrada e são processados através das camadas ocultas, camada à camada, em direção à de saída, cuja a ativação dos neurônios é a resposta da rede. Para todos os neurônios, excluindo os da cama de entrada e os *bias*, a rede realiza o processo de soma ponderada como no perceptron. No entanto, a função de ativação aplicada a soma nas redes MLP é diferenciável.

Desse modo, denotando $w_{ij}^{(l)}$ como o o peso de conexão entre o i-ésimo neurônio da camada $(l-1)$ e o j-ésimo neurônio da camada l, a ativação (ou saída) de j é computada de acordo com a equação

$$a_j^{(l)} = \sigma\left(\sum_{i=0}^{m} w_{ij}^{(l)} a_i^{(l-1)}\right), \tag{2}$$

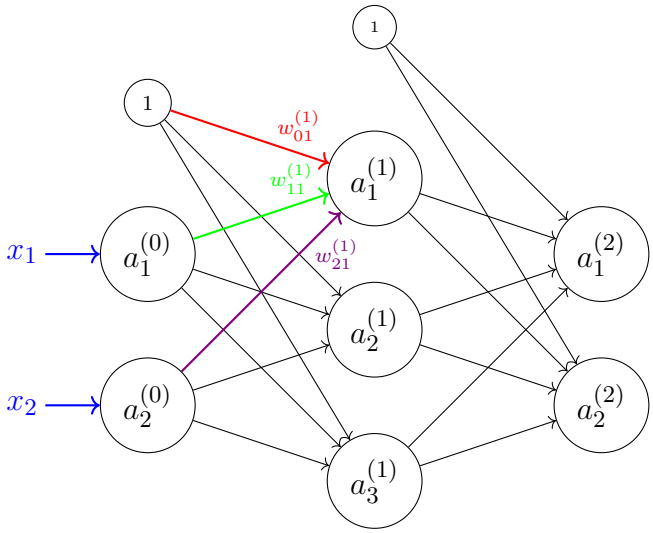
onde σ é a função de ativação e m é a quantidade de neurônios na camada j excluindo o *bias*. Este, está indexado como $a_0^{(l)}$.

Fazendo $\sum_{i=0}^{m} w_{ij}^{(l)} a_i^{(l-1)} = s_j^{(l)}$, (2) se torna

$$a_j^{(l)} = \sigma(s_j^{(l)}).$$

Vejamos a Figura 1.4.

Figura 1.4 – Modelo rede MLP com duas entradas (x_1, x_2), uma camada oculta de três neurônios e a de saída com dois.



Fonte: Elaborado pelo autor.

Ao desenvolver $s_j^{(l)}$, notamos que o somatório pode ser escrito como como um produto entre a matriz transposta dos pesos sinápticos e a matriz coluna das ativações, ou seja

$$s_j^{(l)} = \begin{bmatrix} w_{0j}^{(l)} & w_{1j}^{(l)} & \dots & w_{mj}^{(l)} \end{bmatrix} \begin{bmatrix} a_0^{(l-1)} \\ a_1^{(l-1)} \\ \vdots \\ a_m^{(l-1)} \end{bmatrix}.$$

Com isso, podemos comprimir o cálculo para todas as ativações de uma camada na equação

$$A^{(l)} = \sigma\left(A^{(l-1)}W + W_0\right) \tag{3}$$

sendo A a matriz linha das ativações cuja a camada de referência é indicada pelo expoente, W e W_0 a matriz dos pesos sinápticos e a matriz dos *bias* de conexão entre as ativações das camadas $(l-1)$ e l, respectivamente; e σ agora se aplica a vetores.

Usando a rede da Figura 1.4 como exemplo, verifiquemos que $A^{(1)} = \sigma\left(A^{(0)}W + W_0\right)$. Temos que

$$\begin{aligned} A^{(0)}W + W_0 &= \begin{bmatrix} a_1^{(0)} & a_2^{(0)} \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{bmatrix} + \begin{bmatrix} w_{01} & w_{02} + w_{03} \end{bmatrix} \\ &= \begin{bmatrix} a_1^{(0)} w_{11} + a_2^{(0)} w_{21} & a_1^{(0)} w_{12} + a_2^{(0)} w_{22} & a_1^{(0)} w_{13} + a_2^{(0)} w_{23} \end{bmatrix} + \begin{bmatrix} w_{01} & w_{02} & w_{03} \end{bmatrix} \\ &= \begin{bmatrix} a_1^{(0)} w_{11} + a_2^{(0)} w_{21} + w_{01} & a_1^{(0)} w_{12} + a_2^{(0)} w_{22} + w_{02} & a_1^{(0)} w_{13} + a_2^{(0)} w_{23} + w_{03} \end{bmatrix} \\ &= \begin{bmatrix} s_1^{(1)} & s_2^{(1)} & s_3^{(1)} \end{bmatrix}; \end{aligned}$$

assim,

$$\sigma\left(A^{(0)}W + W_0\right) = \begin{bmatrix} \sigma(s_1^{(1)}) & \sigma(s_2^{(1)}) & \sigma(s_3^{(1)}) \end{bmatrix} = \begin{bmatrix} a_1^{(1)} & a_2^{(1)} & a_3^{(1)} \end{bmatrix} = A^{(1)}.$$

É importante ressaltar que a função de ativação σ não é necessariamente a mesma para todos os neurônios.

Assim, o processo de aprendizado ocorre por meio dos ajustes sucessivos nos pesos sinápticos (que inicialmente são escolhidos aleatoriamente) à passagem de cada exemplo de treino. Na retropropagação, estes ajustes são realizados com base no erro que a rede apresenta, ou seja, na comparação entre a resposta da rede e aquela que se espera que ela consiga atingir ou melhor se aproximar.

2 PRÉ-REQUISITOS

Este capítulo destina-se à discussão de tópicos do Cálculo Diferencial e Integral que embasam o cálculo do algoritmo de retropropagação. Iniciaremos por função de várias variáveis, entendimento necessário para o estudo de limites e derivadas parciais na seção posterior. Adotaremos como referência para esses conteúdos Thomas et al. (2012),da Seção 2.1 à Subseção 2.3.2, e Stewart (2013) para as subseções posteriores. Por fim, encerraremos o capítulo abordando o gradiente descendente, seguindo de acordo com Deisenroth et al. (2020). As demonstrações dos teoremas apresentados nesse capítulo encontram-se presentes nas referências citadas. Optamos por não trascrevê-las aqui, pois, não têm enfase no que se refere aos objetivos do trabalho.

2.1 FUNÇÃO DE VÁRIAS VARIÁVEIS

Funções de várias váriáveis são amplamente utilizadas na modelagem de problemas reais em várias áreas como engenharia, física, bolsa de valores, problemas de otimização, entre outras. Vejamos a Definição 2.1.1.

Definição 2.1.1 Suponha que D seja um conjunto de n-uplas de valores reais $(x_1, x_2, \ldots, x_n)$. Uma função de valores reais f em D é uma regra que associa um único número real $w = f(x_1, x_2, \ldots, x_n)$ a cada elemento em D. O conjunto D é o domínio da função. O conjunto de valores de w assumidos por f é a imagem da função. O símbolo w é a variável dependente de f, e dizemos que f é uma função de n variáveis independentes x_1 a x_n. Também chamamos os x_j de variáveis de entrada da função, e denominamos w a variável de saída da função.

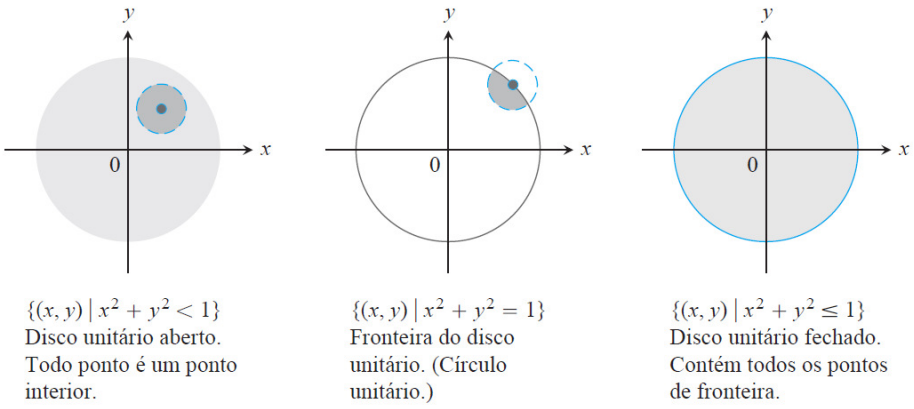
Exemplo 2.1.1 São exemplos de funções de várias variáveis:

1. $f(x, y, z) = x^2 + 2y + xyz$;
2. $P(L, K) = bL\ \alpha K^{1-\alpha}$ (Função de Produção de Cobb-Douglas, onde b e α são parâmetros fixos);
3. $f(x, t) = \cos(1,7 \times 10^{-2}t - 0,2x)e^{-0,2x}$ (função da temperatura abaixo da superfície da Terra dependente da profundidade x abaixo da superfície e da época do ano t).

Em funções de duas variáveis, o domínio é representado por uma região no plano xy. Essa região pode ser caracterizada como aberta ou fechada. Consequentemente, os pontos desse domínio também tem uma classificação, podem ser interiores ou de fronteira.

Definição 2.1.2 Um ponto (x_0, y_0) em uma região (conjunto) R no plano xy é um ponto interior de R se é o centro de um disco de raio positivo que está inteiramente em R. Um ponto (x_0, y_0) é um ponto de fronteira de R se todo disco centrado em (x_0, y_0) contém ao mesmo tempo pontos que estão em R e fora de R. (O ponto de fronteira propriamente dito não precisa pertencer a R.) Os pontos interiores de uma região, como um conjunto, formam o interior da região. Os pontos de fronteira da região formam sua fronteira. Uma região é aberta se consiste inteiramente em pontos interiores. Uma região é fechada se contém todos os seus pontos de fronteira (Figura 2.1).

Figura 2.1 – Pontos interiores e pontos de fronteira do disco unitário no plano.



Fonte: Thomas (2012).

2.2 LIMITES E CONTINUIDADE

Seja f uma função de duas variáveis e L um número real fixado. Dizemos que f se aproxima de um limite L quando, para todos os pontos (x, y) de seu domínio, suficientemente próximos de um ponto (x_0, y_0), temos $f(x, y)$ próximo a L. Notemos que (x_0, y_0) está no domínio de f, uma região de um plano. Assim, (x, y) pode se aproximar de (x_0, y_0) por uma direção qualquer. Para que o limite exista, o valor de L deve ser o mesmo independentemente da direção tomada. Vejamos a definição formal.

Definição 2.2.1 Dizemos que uma função $f(x, y)$ se aproxima do limite L à medida que (x, y) se aproxima de (x_0, y_0) e escrevemos

$$\lim_{(x,y)\to(x_0,y_0)} f(x, y) = L$$

se, para todo $\epsilon > 0$, existir um número $\delta > 0$ correspondente tal que, para todo (x, y) no domínio de f

$$|f(x, y) - L| < \epsilon \qquad \text{sempre que} \qquad 0 < \sqrt{(x - x_0)^2 + (y - y_0)^2} < \delta.$$

As propriedades dos limites de funções de uma variável também se aplicam aqui. Ademais, a continuidade também é definida de forma semelhante.

Definição 2.2.2 Uma função $f(x,y)$ é contínua no ponto (x_0,y_0) se

1. f for definida em (x_0,y_0);
2. $\lim\limits_{(x,y)\to(x_0,y_0)} f(x,y)$ existe;
3. $\lim\limits_{(x,y)\to(x_0,y_0)} f(x,y) = f(x_0,y_0)$.

Uma função é contínua se for contínua em todos os pontos do seu domínio.

As Definições 2.1.1 e 2.2.1, são estendidas para funções de n variáveis.

2.3 DERIVADAS PARCIAIS

Quando trabalhamos com derivadas com funções de várias variáveis, mudamos a nomenclatura para derivadas parciais, pois, só podemos derivar em relação a uma variável por vez, as demais se tornam constantes. Para funções de duas variáveis, temos as sequintes definições.

Definição 2.3.1 A derivada parcial de $f(x,y)$ em relação a x no ponto (x_0,y_0) é

$$\left.\frac{\partial f}{\partial x}\right|_{(x_0,y_0)} = \lim_{h\to 0}\frac{f(x_0+h,y_0)-f(x_0,y_0)}{h},$$

desde que o limite exista.

Analogamente, temos

Definição 2.3.2 A derivada parcial de $f(x,y)$ em relação a y no ponto (x_0,y_0) é

$$\left.\frac{\partial f}{\partial y}\right|_{(x_0,y_0)} = \lim_{h\to 0}\frac{f(x_0,y_0+h)-f(x_0,y_0)}{h},$$

desde que o limite exista.

O conceito de diferenciabilidade também se aplica à derivadas parciais. Uma função f é diferenciável quando sua derivada existe em todo o seu domínio. Sendo assim, a diferenciabilidade implica na continuidade das derivadas parciais. Nesse trabalho abordaremos apenas funções diferenciáveis. Para uma discussão mais detalhada e formal, sugerimos a leitura de Thomas et al. (2012, p. 233).

Exemplo 2.3.1 Seja $f(x,y) = \dfrac{x}{x^2+y^2}$. Calcule $\partial f/\partial x$ e $\partial f/\partial y$.

Solução.

$$\begin{aligned}\frac{\partial f}{\partial x} &= \frac{(x^2+y^2)-2xx}{(x^2+y^2)^2} \\ &= \frac{y^2-x^2}{(x^2+y^2)^2};\end{aligned}$$

$$\frac{\partial f}{\partial y} = \frac{-2yx}{(x^2+y^2)^2}.$$

Para funções com mais variáveis, seguimos da mesma forma.

Exemplo 2.3.2 Calcule $\partial f/\partial x$, $\partial f/\partial y$ e $\partial f/\partial z$, onde $f(x,y,z) = x - \sqrt{y^2+z^2}$.

Solução.

$$\frac{\partial f}{\partial x} = 1;$$

$$\begin{aligned}\frac{\partial f}{\partial y} &= -\frac{1}{2}(y^2+z^2)^{-\frac{1}{2}}2y \\ &= -\frac{y}{\sqrt{y^2+z^2}};\end{aligned}$$

$$\begin{aligned}\frac{\partial f}{\partial z} &= -\frac{1}{2}(y^2+z^2)^{-\frac{1}{2}}2z \\ &= -\frac{z}{\sqrt{y^2+z^2}}.\end{aligned}$$

2.3.1 Derivadas de ordem superior

Ao derivarmos uma função $f(x,y)$ duas vezes, encontramos suas derivadas de segunda ordem. Elas são denotadas por

$$\frac{\partial f}{\partial x^2}, \frac{\partial f}{\partial y^2}, \frac{\partial f}{\partial y \partial x}, \text{ ou } \frac{\partial f}{\partial x \partial y}.$$

A ordem de derivação é da direita para esquerda, por exemplo,

$$\frac{\partial f}{\partial y \partial x} = \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x}\right).$$

Enfatizamos que essa discussão se estende para funções de três ou mais variáveis. Não há um limite para a ordem das derivadas, desde que elas existam.

Exemplo 2.3.3 Seja $f(x,y) = \cos(x) + 2xy$. Calcule

$$\frac{\partial f}{\partial x^2}, \frac{\partial f}{\partial y^2}, \frac{\partial f}{\partial y \partial x}, \text{ e } \frac{\partial f}{\partial x \partial y}$$

.

Solução. Primeiro calculamos as derivadas de primeira ordem:

$$\frac{\partial f}{\partial x} = -\operatorname{sen}(x) + 2y \text{ e } \frac{\partial f}{\partial y} = 2x.$$

Assim, temos que,

$$\begin{aligned}
\frac{\partial f}{\partial x^2} &= \frac{\partial}{\partial x}(-\operatorname{sen}(x) + 2y) = -\cos(x); \\
\frac{\partial f}{\partial y^2} &= \frac{\partial}{\partial y}(2x) = 0; \\
\frac{\partial f}{\partial y \partial x} &= \frac{\partial}{\partial y}(-\operatorname{sen}(x) + 2y) = 2; \\
\frac{\partial f}{\partial x \partial y} &= \frac{\partial}{\partial x}(2x) = 2.
\end{aligned}$$

As derivadas $\partial f/\partial y \partial x$ e $\partial f/\partial x \partial y$ são chamadas de derivadas mistas. No exemplo elas são iguais e isso acontece sempre que f, $\frac{\partial f}{\partial x^2}, \frac{\partial f}{\partial y^2}, \frac{\partial f}{\partial y \partial x}$, e $\frac{\partial f}{\partial x \partial y}$ forem contínuas.

2.3.2 Regra da cadeia

Em funções de várias variáveis, a depender delas, a regra da cadeia pode assumir variadas formas. Seguindo o fluxo das subseções anteriores, consideremos a função $w = f(x,y)$, com $x = x(t)$ e $y = y(t)$ onde x, y são variáveis intermediárias e t é independente. A fórmula para esse caso é dada pelo teorema a seguir.

Teorema 2.3.1 Se $w = f(x,y)$ é diferenciável e se $x = x(t), y = y(t)$ forem funções deriváveis de t, então a composta $w = f(x(t),y(t))$ será uma função derivável de t e

$$\frac{dw}{dt} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt}.$$

Exemplo 2.3.4 Seja $w = x^2 + y^2$ com $x = \cos(t), y = \text{sen}(t)$. Calcule dw/dt em $t = \pi$.

Solução.

$$\begin{aligned}\frac{dw}{dt} &= \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial w}{\partial y}\frac{dy}{dt} \\ &= 2x(-\text{sen}(t)) + 2y\cos(t) \\ &= 2\cos(t)(-\text{sen}(t)) + 2\text{sen}(t)\cos(t);\end{aligned}$$

logo,

$$\begin{aligned}\frac{dw}{dt}(\pi) &= 2\cos(\pi)(-\text{sen}(\pi)) + 2\text{sen}(\pi)\cos(\pi) \\ &= 2(-1)0 + 2\cdot 0(-1) \\ &= 0.\end{aligned}$$

O processo é análogo para funções de mais variáveis intermediárias e uma independente. No entanto, quando a função possui mais de uma variável independente, o processo sofre uma alteração. Vejamos o teorema 2.3.2.

Teorema 2.3.2 Suponha que $w = f(x,y,z)$, $x = g(r,s)$, $y = h(r,s)$ e $z = k(r,s)$. Se todas as quatro funções forem diferenciáveis, então w terá derivadas parciais em relação a r e s, dadas pelas fórmulas

$$\begin{aligned}\frac{\partial w}{\partial r} &= \frac{\partial w}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial w}{\partial y}\frac{\partial y}{\partial r} + \frac{\partial w}{\partial z}\frac{\partial z}{\partial r} \\ \frac{\partial w}{\partial s} &= \frac{\partial w}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial w}{\partial y}\frac{\partial y}{\partial s} + \frac{\partial w}{\partial z}\frac{\partial z}{\partial s}.\end{aligned}$$

Exemplo 2.3.5 Sendo $u = \dfrac{p-q}{q-r}$, $p = x+y+z$, $q = x-y+z$, $r = x+y-z$. Encontre $\dfrac{\partial u}{\partial x}$.

Solução. Devemos calcular

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial p}\frac{\partial p}{\partial x} + \frac{\partial u}{\partial q}\frac{\partial q}{\partial x} + \frac{\partial u}{\partial r}\frac{\partial r}{\partial x}.$$

$$\begin{aligned}\frac{\partial u}{\partial p}\frac{\partial p}{\partial x} &= \frac{(q-r)}{(q-r)^2}\\ &= \frac{1}{(q-r)}\\ &= \frac{1}{(x-y+z-x-y+z)}\\ &= \frac{1}{2(z-y)};\end{aligned}$$

$$\begin{aligned}\frac{\partial u}{\partial q}\frac{\partial q}{\partial x} &= \frac{-1(q-r)-(p-q)}{(q-r)^2}\cdot 1\\ &= \frac{-q+r-p+q}{(q-r)^2}\\ &= \frac{x+y-z-x-y-z}{(x-y+z-x-y+z)^2}\\ &= -\frac{2z}{4(z-y)^2}\\ &= -\frac{z}{2(z-y)^2};\end{aligned}$$

$$\begin{aligned}\frac{\partial u}{\partial r}\frac{\partial r}{\partial x} &= \frac{-(-1)(p-q)}{(q-r)^2}\\ &= \frac{p-q}{(q-r)^2}\\ &= \frac{x+y+z-x+y-z}{4(z-y)^2}\\ &= \frac{2y}{4(z-y)^2}\\ &= \frac{y}{2(z-y)^2}.\end{aligned}$$

Portanto,

$$\begin{aligned}\frac{\partial u}{\partial x} &= \frac{1}{2(z-y)}-\frac{z}{2(z-y)^2}+\frac{y}{2(z-y)^2}\\ &= \frac{z-y-z+y}{2(z-y)^2}\\ &= \frac{0}{2(z-y)^2}\\ &= 0.\end{aligned}$$

No geral, a quantidade de parcelas da soma no cálculo da derivada dependerá da quantidade de variáveis da função. Já a quantidade de derivadas de cada parcela dependerá da forma que as variáveis independentes estão na composição das funções das variáveis intermediárias. Stewart (2013) apresenta o caso mais geral da regra da cadeia considerando a existência de uma variável independente.

Teorema 2.3.3 Suponha que u seja uma função diferenciável de n variáveis $x_1, x_2, \ldots, x_n$ onde cada x_j é uma função diferenciável de m variáveis $t_1, t_2, \ldots, t_m$. Então u é uma função de $t_1, t_2, \ldots, t_m$ e

$$\frac{\partial u}{\partial t_i} = \frac{\partial u}{\partial x_1}\frac{\partial x_1}{\partial t_i} + \frac{\partial u}{\partial x_2}\frac{\partial x_2}{\partial t_i} + \cdots + \frac{\partial u}{\partial x_n}\frac{\partial x_n}{\partial t_i}$$

para cada $i = 1, 2, \ldots, m$.

Vejamos como funciona a regra da cadeia em uma função com mais de uma composição.

Exemplo 2.3.6 Considere a função $f(x,y) = x^2 + y^2$, onde $x = u^2 + v$, $y = uv + v^2$ com $u = \cos(t)$ e $v = \operatorname{sen}(t)$. Calcule $\dfrac{\partial f}{\partial t}$.

Solução. Temos que

$$\begin{aligned} \frac{\partial f}{\partial t} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial t} \\ &= 2x\frac{\partial x}{\partial t} + 2y\frac{\partial y}{\partial t}. \end{aligned} \tag{4}$$

Para encontrar as derivadas restantes, devemos perceber que u e v dependem de t. Então, aplicamos a regra da cadeia novamente e encontramos que

$$\begin{aligned} \frac{\partial x}{\partial t} &= \frac{\partial x}{\partial u}\frac{\partial u}{\partial t} + \frac{\partial x}{\partial v}\frac{\partial v}{\partial t} \\ &= -2u\operatorname{sen}(t) + \cos(t) \\ &= -2\cos(t)\operatorname{sen}(t) + \cos(t); \end{aligned} \tag{5}$$

e, também,

$$\begin{aligned}\frac{\partial y}{\partial t} &= \frac{\partial y}{\partial u}\frac{\partial u}{\partial t}+\frac{\partial y}{\partial v}\frac{\partial v}{\partial t}\\ &= -v\,\mathrm{sen}(t)+(u+2v)\cos(t)\\ &= -\mathrm{sen}(t)\,\mathrm{sen}(t)+(\cos(t)+2\,\mathrm{sen}(t))\cos(t)\\ &= -\mathrm{sen}^2(t)+\cos^2(t)+2\,\mathrm{sen}(t)\cos(t)\\ &= \cos^2(x)-1+\cos^2(t)+2\,\mathrm{sen}(t)\cos(t)\\ &= 2\cos^2(t)+2\,\mathrm{sen}(t)\cos(t)-1. \qquad (6)\end{aligned}$$

Substituindo (5) e (6) em (4), temos que

$$\begin{aligned}\frac{\partial f}{\partial t} &= -2\cos(t)\,\mathrm{sen}(t)+\cos(t)+2\cos^2(t)+2\,\mathrm{sen}(t)\cos(t)-1\\ &= 2\cos^2(t)+\cos(t)-1.\end{aligned}$$

2.3.3 Derivada direcional e vetor gradiente

A derivada direcional calcula a taxa de variação de uma função de várias variáveis em uma direção específica determinada por um vetor, vejamos a definição a seguir para funções de duas variáveis.

Definição 2.3.3 A derivada de f em $P_0(x_0,y_0)$ na direção do vetor unitário $u=ai+bj$ é o número

$$D_u f=\lim_{h\longrightarrow 0}\frac{f(x_0+ha,y_0+hb)-f(x_0,y_0)}{h},$$

contanto que o limite exista.

O cálculo da derivada direcional mostrado na Definição 2.3.3 é simplificado pelo seguinte teorema.

Teorema 2.3.4 Se f é uma função diferenciável de x e y, então f tem derivada direcional na direção de qualquer vetor $u=ai+bj$ e

$$D_u f(x,y)=\frac{\partial f}{\partial x}a+\frac{\partial f}{\partial y}b.$$

Exemplo 2.3.7 Determine a derivada direcional da função $f(x,y)=x^4-x^2y^3$ no ponto $(2,1)$ na direção do vetor $u=i+3j$.

Solução. Temos que

$$\frac{\partial f}{\partial x} = 4x^3-2xy^3;$$

$$\frac{\partial f}{\partial y} = -3x^2y^2.$$

Como $\|u\| = \sqrt{10}$, o vetor unitário de u é

$$v = \frac{1}{\sqrt{10}}i + \frac{3}{\sqrt{10}}j.$$

Assim, $D_uf(x,y) = (4x^3 - 2xy^3)\frac{1}{\sqrt{10}} + (-3x^2y^2)\frac{3}{\sqrt{10}}$, logo,

$$\begin{aligned} D_uf(2,1) &= (4\cdot 2^3 - 2\cdot 2\cdot 1^3)\frac{1}{\sqrt{10}} + (-3\cdot 2^2\cdot 1^2)\frac{3}{\sqrt{10}} \\ &= -\frac{4\sqrt{10}}{5}. \end{aligned}$$

Facilmente percebemos que a equação do Teorema 2.3.4 pode ser escrita como o produto escalar de dois vetores:

$$\begin{aligned} D_uf(x,y) &= \frac{\partial f}{\partial x}a + \frac{\partial f}{\partial y}b \\ &= \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)\cdot(a,b) \\ &= \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)\cdot u. \end{aligned}$$

O primeiro vetor possui aplicações que vão além do cálculo da derivada direcional, ele possui nome e notação própria.

Definição 2.3.4 Se f é uma função de duas variáveis x e y, então o gradiente de f é a função vetorial ∇f definida por

$$\nabla f(x,y) = \frac{\partial f}{\partial x}i + \frac{\partial f}{\partial y}j.$$

Assim, podemos reescrever a fórmula para derivada direcional como

$$D_uf(x,y) = \nabla f(x,y)\cdot u. \tag{7}$$

Exemplo 2.3.8 Determine a derivada direcional da função $f(x,y) = x^2 + xy$ no ponto $(1,1)$ na direção do vetor $u = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$.

Solução.

$$\nabla f(x,y) = (2x+y, x) \Rightarrow \nabla f(1,1) = (3,1).$$

O vetor u é unitário, assim

$$D_u f(1,1) = (3,1) \cdot \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}} \right) = \frac{3}{\sqrt{2}} + \frac{1}{\sqrt{2}} = \frac{4}{\sqrt{2}} = 2\sqrt{2}.$$

Lembremos que

$$\cos\theta = \frac{\nabla f \cdot u}{\|\nabla f\| \|u\|}.$$

Logo, $\nabla f \cdot u = \|\nabla f\| \|u\| \cos\theta$. Assim, da Equação (7) temos $D_u f(x,y) = \|\nabla f\| \|u\| \cos\theta$. Como u é unitário $\|u\| = 1$, assim

$$D_u f(x,y) = \|\nabla f\| \cos\theta.$$

θ é o ângulo entre ∇f e u. Quando $\theta = 0$, $\cos\theta$ assume valor máximo 1. Portanto, o valor máximo de $D_u f$ é $\|\nabla f\|$ sempre que $\theta = 0$, quando u tem a mesma direção de ∇f. Por outro lado, $\cos\theta$ assume valor mínimo igual a -1 quando $\theta = \pi$, ou seja, o valor mínimo de $D_u f$ é $-\|\nabla f\|$, quando $D_u f$ tem direção oposta a de u.

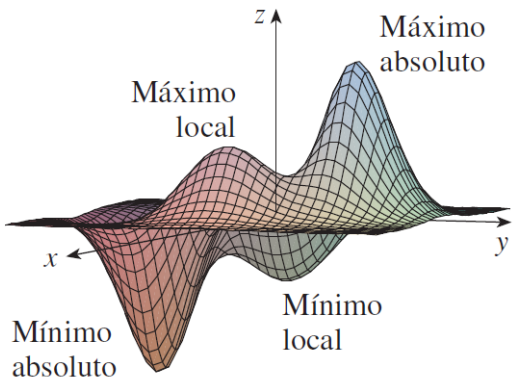
Em outras palavras, o gradiente mostra a direção na qual a função cresce de forma mais rápida e o seu inverso indica o sentido que ela decresce de forma mais rápida.

Ressaltamos que os conceitos discutidos nessa subseção aplicam-se a funções de mais variáveis seguindo o mesmo raciocínio.

2.3.4 Valores máximo e mínimo

Uma das principais aplicações da derivada em funções de uma variável é a determinação dos valores extremos da função. Isso se estende à derivadas parciais. Aqui abordaremos, inicialmente, essa aplicação em funções de duas variáveis para explorar a noção intuitiva através da geometria do assunto. Assim, observemos o gráfico de uma função f na Figura 2.2 com alguns picos e vales.

Figura 2.2 – Máximos e Mínimos.



Fonte: Stewart (2013).

Em dois pontos (a,b), f assume um valor máximo local, onde $f(a,b)$ é maior que os valores de $f(x,y)$ próximos. O maior dos dois valores de máximo local, é chamado de máximo absoluto ou global. Da mesma forma, f possui dois valores de mínimo local, o menor deles é chamado de mínimo absoluto ou global, onde $f(a,b)$ é menor que todos os valores de $f(x,y)$ próximos.

Definição 2.3.5 Uma função de duas variáveis tem um máximo local em (a,b) se $f(x,y) \leq f(a,b)$ quando (x,y) está próximo de (a,b). O número $f(a,b)$ é chamado de valor máximo local. Se $f(x,y) \geq f(a,b)$ quando (x,y) está próximo de (a,b), então f tem um mínimo local em (a,b) e $f(a,b)$ é um valor mínimo local.

Se as desigualdades da Definição 2.3.5 forem válidas para todos os pontos do domínio de f, então f tem máximo ou mínimo global.

Teorema 2.3.5 Se f tem um máximo ou mínimo local em (a,b) e as derivadas parciais de primeira ordem de f existem nesses pontos, então

$$\frac{\partial f}{\partial x}(a,b) = 0 \text{ e } \frac{\partial f}{\partial y}(a,b) = 0.$$

Um ponto (a,b) é chamado de crítico quando satisfaz o Teorema 2.3.5. Precisamos saber que valor extremo a função assume em cada ponto crítico. Para isso, vejamos o teste a seguir.

Teste da Segunda Derivada Suponha que as segundas derivadas parciais de f sejam contínuas em uma bola aberta com centro no ponto crítico (a,b). Seja

$$D = D(a,b) = \frac{\partial f}{\partial x^2}(a,b)\frac{\partial f}{\partial y^2}(a,b) - \left(\frac{\partial f}{\partial y \partial x}\right)^2$$

(a) Se $D > 0$ e $\dfrac{\partial f}{\partial x^2}(a,b) > 0$, então $f(a,b)$ é um mínimo local.

(b) Se $D > 0$ e $\dfrac{\partial f}{\partial x^2}(a,b) < 0$, então $f(a,b)$ é um máximo local.

(c) Se $D < 0$, então $f(a,b)$ não é mínimo nem máximo local. Nesse caso, o chamamos de ponto de sela.

(d) Se $D = 0$, nada podemos afirmar sobre $f(a,b)$.

Exemplo 2.3.9 Determine os valores de máximos e mínimos e pontos de sela de $f(x,y) = x^3 - 12xy + 8y^3$.

Solução. Primeiramente devemos encontrar os pontos críticos.

$$\begin{aligned} \frac{\partial f}{\partial x} &= 3x^2 - 12y; \\ \frac{\partial f}{\partial y} &= -12x + 24y^2. \end{aligned}$$

Resolvendo o sistema de equações

$$\begin{cases} 3x^2 - 12y &= 0 \\ -12x + 24y^2 &= 0 \end{cases},$$

encontramos os pontos críticos $(0,0)$ e $(2,-1)$. Temos que

$$\begin{aligned} \frac{\partial f}{\partial x^2} = 6x \quad &\Rightarrow \quad \frac{\partial f}{\partial x^2}(0,0) = 0; \\ \frac{\partial f}{\partial y^2} = 48y \quad &\Rightarrow \quad \frac{\partial f}{\partial y^2}(0,0) = 0; \\ \frac{\partial f}{\partial y \partial x} = -12 \quad &\Rightarrow \quad \frac{\partial f}{\partial y \partial x}(0,0) = -12 \end{aligned}$$

logo,

$$D(0,0) = 0 \cdot 0 - (-12)^2 = -144 < 0.$$

Concluímos que $(0,0)$ é um ponto de sela.

Por outro lado,

$$\begin{aligned} \frac{\partial f}{\partial x^2}(2,1) = 12; \\ \frac{\partial f}{\partial y^2}(2,1) = 41; \\ \frac{\partial f}{\partial y \partial x}(2,1) = -12; \end{aligned}$$

logo,

$$D(2,1) = 12 \cdot 41 - (-12)^2 = 348 > 0.$$

Concluímos que $f(2,1) = -8$ é um mínimo.

Quando a função é contínua e está restrita a um domínio fechado, um teorema garante a existência de valores extremos globais.

Teorema 2.3.6 Se f é contínua em um conjunto fechado e limitado D em $\mathbb{R}^2$, então, f assume um valor máximo global $f(x_1, y_2)$ e um valor mínimo global $f(x_2, y_2)$ em alguns pontos (x_1, y_1) e (x_2, y_2) de D.

Para encontrarmos os valores máximo e mínimo globais de uma função f contínua, seguimos como visto no Exemplo 2.3.9. Além disso, precisamos determinar os valores extremos de f na fronteira do seu domínio. Assim, o maior dos valores encontrados será o máximo global e o menor o mínimo global.

Exemplo 2.3.10 Determine os valores máximo e mínimo globais de $f(x,y) = x^4 + y^4 - 4xy + 2$ no conjunto $D = \{(x,y) | 0 \leq x \leq 3, 0 \leq y \leq 2\}$.

Solução.

Temos que, $\dfrac{\partial f}{\partial x} = 4x^3 - 4y$, $\dfrac{\partial f}{\partial y} = 4y^3 - 4x$. Resolvendo

$$\begin{cases} 4x^3 - 4y &= 0 \\ 4y^3 - 4x &= 0 \end{cases},$$

encontramos os pontos críticos $(0,0)$, $(1,1)$, $(-1,-1)$.

Calculemos, agora, as derivadas de segunda ordem:

$$\frac{\partial f}{\partial x^2} = 12x^2, \ \frac{\partial f}{\partial y^2} = 12y^2, \ \frac{\partial f}{\partial y \partial x} = -4.$$

Assim,

(I) $\dfrac{\partial f}{\partial x^2}(0,0) = 0$, $\dfrac{\partial f}{\partial y^2}(0,0) = 0$, $\dfrac{\partial f}{\partial y \partial x}(0,0) = -4$;

$$D(0,0) = 0 \cdot 0 - (-4)^2 = -16,$$

logo, $(0,0)$ é ponto de sela.

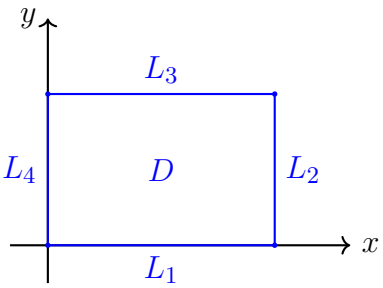
(II) $\dfrac{\partial f}{\partial x^2}(1,1) = \dfrac{\partial f}{\partial y^2} = 12$, $\dfrac{\partial f}{\partial y \partial x}(1,1) = -4$;

$$D(1,1) = 12 \cdot 12 - (-4)^2 = 128,$$

logo, $f(1,1) = 0$ é um valor mínimo.

Precisamos encontrar, também, os valores extremos de f na fronteira de D que está representada na Figura 2.3.

Figura 2.3 – Fronteira de D.



Fonte: Elaborado pelo autor.

Em L_1 temos $y = 0 \Rightarrow f(0,y) = x^4 + 2$ que possui $f(3,0) = 83$ e $f(0,0) = 2$ como valor máximo e mínimo, respectivamente.

Em L_2 temos $x = 3 \Rightarrow f(3,y) = 83 + y^4 - 12y$ que possui $f(3,0) = 83$ e $f(3, \sqrt[3]{3}) \cong 70,02$ como valor máximo e mínimo, respectivamente.

Em L_4 temos $x = 0 \Rightarrow f(0,y) = y^4 + 2$ que possui $f(0,2) = 18$ e $f(0,0) = 2$ como valor máximo e mínimo, respectivamente.

Em L_3 temos $y = 2 \Rightarrow f(x,2) = x^4 - 8x + 18$ que possui $f(3,2) = 75$ e $f(\sqrt[3]{2},2) \cong 10,44$ como valor máximo e mínimo, respectivamente.

Assim, concluímos que $f(1,1) = 0$ é o mínimo global e $f(3,0) = 83$ é o máximo global.

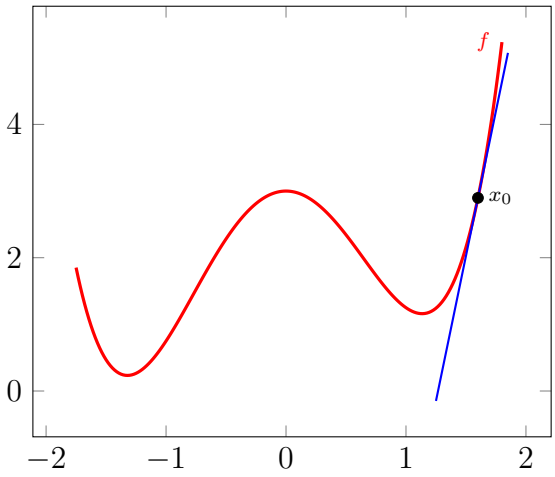
2.4 GRADIENTE DESCENDENTE

Os problemas de otimização se resumem a encontrar valores extremos de funções que, normalmente, modelam o custo de um determinado problema prático. Em geral, busca-se minimizá-las, ou seja, encontrar o valor de mínimo que proporcione a maior redução do custo. Porém, esses problemas envolvem funções de múltiplas variáveis que tornam essa tarefa árdua se feita de forma usual. Nesse contexto se insere o conceito de Gradiente Descendente como uma poderosa ferramenta para se chegar, ou se aproximar, da solução desejada.

Considere a função $f(x) = x^4 + \frac{1}{4}x^3 - 3x^2 + 3$. Observando a Figura 2.4 vemos que f possui dois valores de mínimo: um local e outro global. O objetivo é encontrá-los, porém, de uma forma não-analítica, sem precisarmos encontrar seus pontos críticos e realizar os testes de derivada. Para isso, primeiramente, tomemos um valor aleatório para x_0 e calculemos $f'(x_0)$.

Sendo $x_0 = 1,6$ e $f'(x) = 4x^3 + \frac{3}{4}x^2 - 6x$, temos que $f'(1,6) = 8,704$. Como $f'(1,6) > 0$, a reta tangente a f em $x_0 = 1,6$ é crescente, ou seja, se pegarmos valores à direita, próximos a x_0, encontraremos valores maiores para f e não é o que queremos.

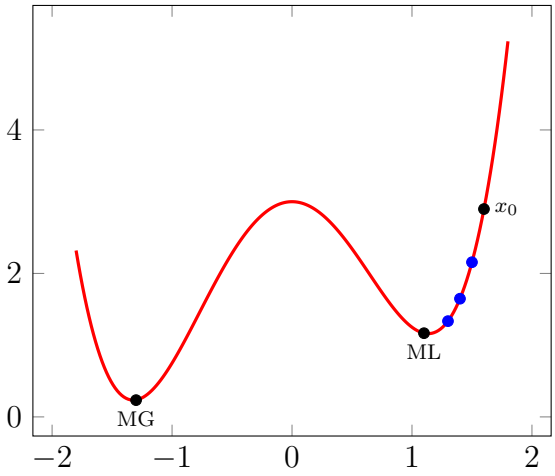
Figura 2.4 – Tangente a f em x_0.



Fonte: Elaborado pelo autor.

Assim, seguindo a direção contrária do gradiente, podemos chegar ao nosso objetivo dando pequenos "passos" proporcionais ao subtraírmos pequenas unidades de x_0, como ilustrado na Figura 2.5. Perceba que o valor inicial de x_0 influencia no resultado pois, com a escolha do exemplo, os passos nos levam para um mínimo local $x_0 \cong 1,2$. No entanto, f assume seu menor valor no mínimo global, assim, o melhor resultado seria $x_0 \cong -1,4$.

Figura 2.5 – Caminho em direção a um mínimo. Mínimo Global (ML), Mínimo Local (ML).



Fonte: Elaborado pelo autor.

Essa breve discussão teve por objetivo introduzir o conceito, por meio da vizualização em $\mathbb{R}$, do gradiente descendente. Ademais, anteriormente usamos o termo gradiente no seu sentido de direção relacionado ao sinal da derivada já que a função do exemplo possui apenas uma variável. A partir de agora, será tratado, de fato, como um vetor.

Destarte, seja $f:\mathbb{R}^n \to \mathbb{R}$ uma função diferenciável. O gradiente descendente é um algorítmo para minimizar f de forma iterativa. Para isso, aplica-se em f uma enupla $(x_1,\ldots,x_n)$ de valores arbitrários que, em seguida, sofrem pequenos ajustes tal que

$$x_i(t+1) = x_i(t) - \eta \nabla f(x_i(t)),\ i = 1,\ldots,n,$$

onde $\nabla f(x_i(t))$ é o gradiente de f em x_i na iteração t, e η é uma constante positiva chamada taxa de aprendizado. Sendo assim, nesses ajustes seguimos a direção contrária do vetor gradiente, de forma que a correção aplicada em x_i é inversamente proporcional ao gradiente de f em relação ao própio, consequentemente, fazendo com que a sequência $f(x_1(t),\ldots,x_n(t)) \geq f(x_1(t+1),\ldots,x_n(t+1))$ convirja para um mínimo local.

Esse processo também é conhecido como descida do gradiente, pois, associado ao caso onde f possui duas variáveis e seu gráfico é uma superfície, sua interpretação geométrica é, de fato, uma descida em direção a algum vale (mínimo). As iterações também são chamadas de passos de aprendizagem.

A taxa de aprendizado deve ser um valor pequeno, geralmente $\eta \in (0,1)$, pois ela representa o tamanho da variação que x_i irá sofrer. Haykin (1999) enfatiza que η influencia diretamente o comportamento da convergência: quanto menor a taxa de aprendizado, mais devagar o método convergirá para uma solução adequada e a tragetória traçada por u é suave. A depender da função, para valores altos, η pode fazer com que a convergência ocorra mais rápido, porém, a trajetória de u será um caminho com oscilações, em alguns casos, podendo fazer com que o caminho desvie da solução ou que, simplesmente, não convirja para nenhuma.

Exemplo 2.4.1 Sendo $f(x_1,x_2) = x_1^2 + x_2^2$, encontre, aproximadamente, o mínimo de f utilizando o gradiente descendente.

Solução. Escolhamos $\eta = 0,15$ e iniciemos com $(x_1,x_2) = (1,2)$, assim, $f(1,2) = 5$. Temos que,

$$\frac{\partial f}{\partial x_1} = 2x_1,\ \frac{\partial f}{\partial x_2} = 2x_2 \Rightarrow \nabla f = (2x_1, 2x_2).$$

Assim,

$$\begin{aligned} x_1(1) &= x_1(0) - \eta \nabla f(x_1(0)) \\ &= 1 - 0,15 \cdot 2 \cdot 1 \\ &= 0,7; \end{aligned}$$

$$\begin{aligned} x_2(1) &= x_2(0) - \eta \nabla f(x_2(0)) \\ &= 2 - 0,15 \cdot 2 \cdot 2 \\ &= 1,4. \end{aligned}$$

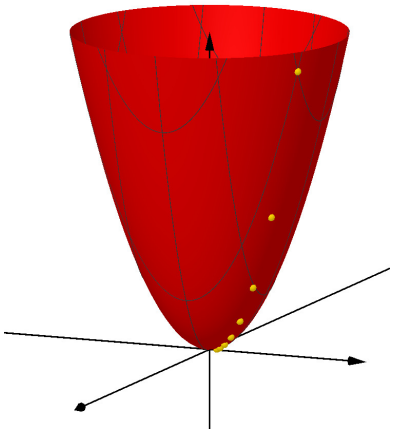
Note que $f(0,7,1,4) = 2,45 < f(1,2)$. Repetindo o processo sucesssivas vezes, percebemos que o mínimo converge para $(0,0)$. A Tabela 2.1 mostra o resultado de algumas iterações que estão ilustradas na Figura 2.6.

Tabela 2.1 – Descida do gradiente para $\eta = 0,15$.

t	x_1	x_2	$f(x_1,x_2)$
0	1	2	5
1	0,7	1,4	2,45
2	0,49	0,98	1,2005
3	0,343	0,686	0,5882
4	0,2401	0,4802	0,2882
5	0,1680	0,3361	0,1412
6	0,1176	0,2352	0,0691
7	0,0823	0,1647	0,0339

Fonte: Elaborado pelo autor.

Figura 2.6 – Descida do gradiente.



Fonte: Elaborado pelo autor.

Exemplo 2.4.2 Aplique o método do gradiente descendente em $f(x_1,x_2) = \text{sen}(x_1) + \sqrt{x_1^2+x_2^2+1}$.

Solução. Temos que

$$\begin{aligned}\frac{\partial f}{\partial x_1} &= \cos(x_1) + \frac{1}{2}(x_1^2+x_2^2+1)^{-\frac{1}{2}}2x_1 \\ &= \cos(x_1) + x_1\frac{1}{\sqrt{x_1^2+x_2^2+1}} \\ &= \cos(x_1) + x_1\frac{\sqrt{x_2^2+x_2^2+1}}{x_1^2+x_2^2+1}.\end{aligned}$$

De forma semelhante, encontramos que

$$\frac{\partial f}{\partial x_2} = x_2 \frac{\sqrt{x_2^2+x_2^2+1}}{x_1^2+x_2^2+1},$$

logo,

$$\nabla f = \left(\cos(x_1) + x_1 \frac{\sqrt{x_2^2+x_2^2+1}}{x_1^2+x_2^2+1}, x_2 \frac{\sqrt{x_2^2+x_2^2+1}}{x_1^2+x_2^2+1}\right).$$

Tomemos $(x_1(0), x_2(0)) = (2,-4)$ e $\eta = 0,2$. Temos que,

$$\begin{aligned} f(2,-4) &= \text{sen}(2) + \sqrt{2^2+(-4)^2+1} \\ &\cong 5.491873121781522. \end{aligned}$$

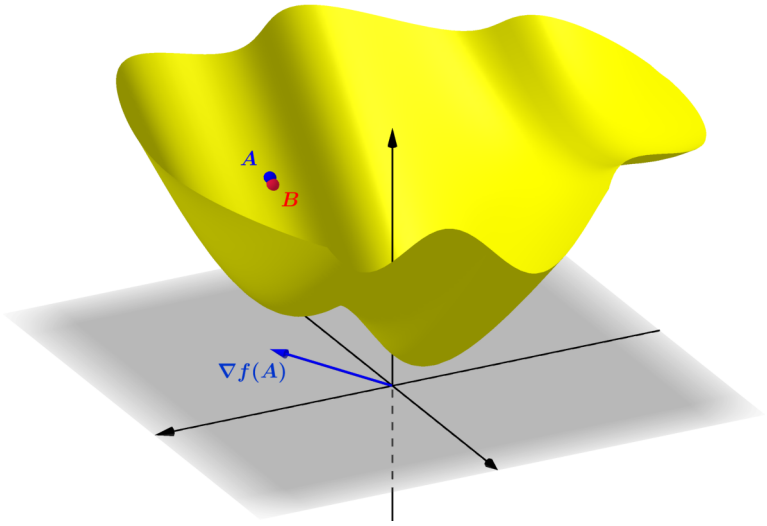
Além disso, $\nabla f(x_1(0), x_2(0)) \cong (0,020288943924842318, -0,8728715609439694)$. Assim,

$$\begin{aligned} x_1(1) &= x_1(0) - \eta \nabla f(x_1(0)) \\ &\cong 2 - 0,2 \cdot 0,020288943924842318 \\ &\cong 1,9959422112; \end{aligned}$$

$$\begin{aligned} x_2(1) &= x_2(0) - \eta \nabla f(x_2(0)) \\ &\cong -4 - 0,2 \cdot (-0,8728715609439694) \\ &\cong -3,8254256878. \end{aligned}$$

Após essa primeira iteração, encontramos que $f(x_1(1), x_2(1)) \cong 5,3401621316$, vejamos a Figura 2.7.

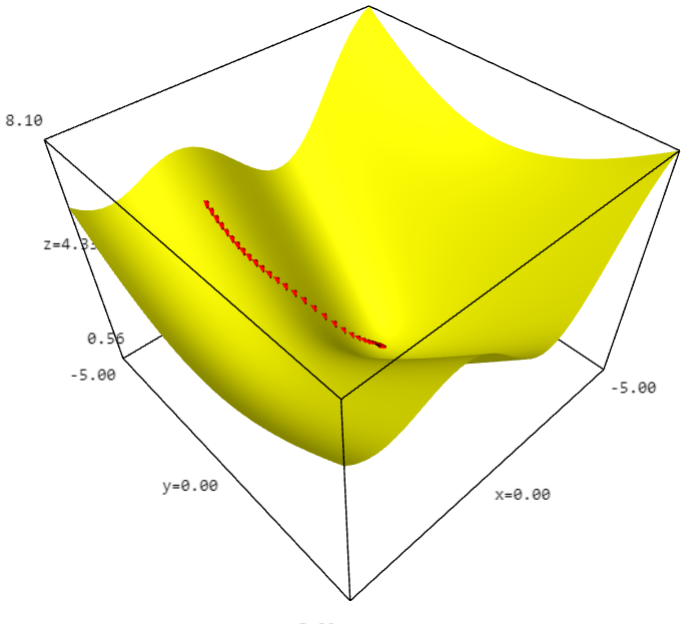
Figura 2.7 – Geometria da primeira iteração, sendo $A(x_1(0), x_2(0))$ e $B(x_1(1), x_2(1))$.



Fonte: Elaborado pelo autor.

Depois de 88 atualizações encontramos $x_1 = -0,8603335863$ e $x_2 = -0,0000111829$, que resultam no mínimo global aproximadamente igual a $0,5610963382$. A Figura 2.8 ilustra todas as iterações, estas estão presentes na tabela do Apêndice A.1.

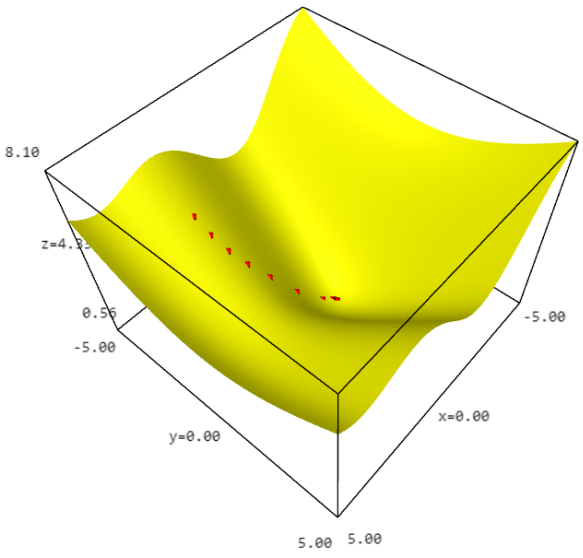
Figura 2.8 – Iterações com $\eta = 0,2$ e escolha inicial $(2,-4)$.



Fonte: Elaborado pelo autor.

Se tomarmos $\eta = 0,7$, neste caso, a convergência para o mínimo acontece mais rápida sendo necessárias 22 iterações (ver Apêndice A.2) para chegar ao resultado alcançado anteriormente. Vejamos a Figura 2.9.

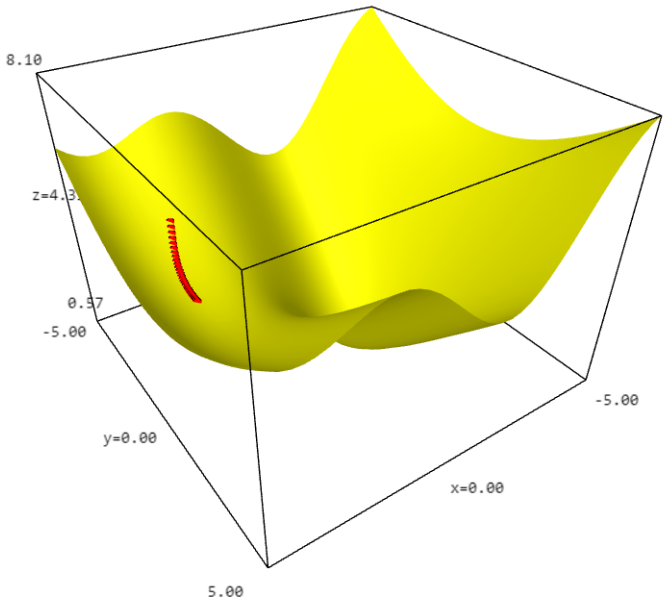
Figura 2.9 – Iterações com $\eta = 0,7$ e escolha inicial $(2,-4)$.



Fonte: Elaborado pelo autor.

Outro fator que pode influenciar no resultado, como comentado no início dessa seção, é a escolha de $(x_1(0), x_2(0))$. Para esse mesmo exemplo, tomemos $(x_1(0), x_2(0)) = (3, -3)$ e escolhamos a taxa de aprendizado inicial $\eta = 0,2$. A Figura 2.10 mostra que a escolha dos valores iniciais proporcionou um resultado insatisfatório, pois, realizando 70 iterações (ver Apêndice A.3), ficamos presos em um mínimo local aproximadamente igual a $3,28$.

Figura 2.10 – Iterações com $\eta = 0,2$ e escolha inicial $(3, -3)$.



Fonte: Elaborado pelo autor.

No Exemplo 2.4.1, poderiamos afirmar que $(0, 0)$ é o ponto de mínimo da função apenas observando sua lei de formação, diferentemente do Exemplo 2.4.2. As redes neurais são modeladas matematicamente por uma composição de funções de muitas variáveis, minimizá-las encontrando pontos críticos é inviável. Os exemplos tiveram como objetivo facilitar a compreensão do método que se aplica ao algoritmo de retropropagação que veremos no capítulo a seguir.

3 RETROPROPAGAÇÃO

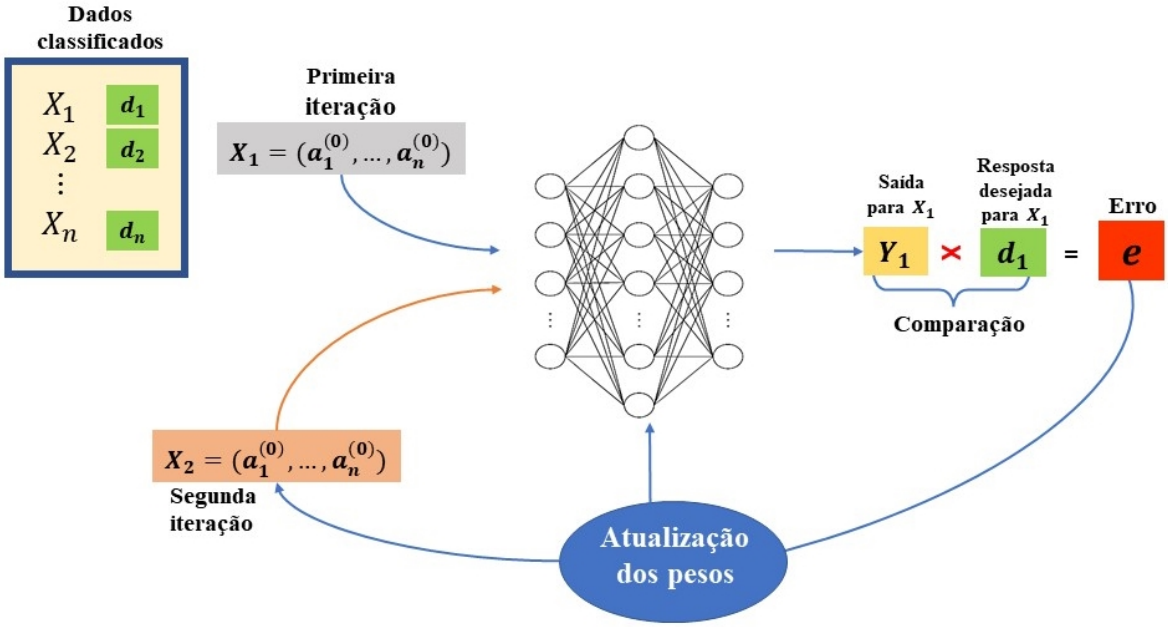
Neste capítulo será abordado o algoritmo da retropropagação. Iniciaremos por uma abordagem mais intuitiva e seguiremos para definição de alguns conceitos que consideramos como preliminares. Posteriormente, encontraremos as fórmulas necessárias para a atualização dos pesos sinápticos que se dividem em dois casos: quando o peso se conecta a um neurônio de saída e quando ele se conecta a um neurônio oculto. Por fim, mostraremos um exemplo do algoritmo.

3.1 UMA ABORDAGEM INICIAL DA RETROPROPAGAÇÃO

Já sabemos que para uma rede aprende treinando com um conjunto de dados chamados de exemplos de treino. Esse treinamento consiste em encontrar os valores ideais para seus pesos que produzam uma saída adequada correspondente para cada exemplo.

No aprendizado supervisionado, esses exemplos já possuem uma resposta que se espera que a rede melhor se aproxime. Como os pesos iniciais são aleatórios, é improvável que eles sejam os ideais, logo, precisam ser atualizados. O exemplo de treino é passado para a rede e produz uma saída que é comparada com sua resposta desejada e, assim, define-se um erro. Com base nesse erro, todos os pesos são atualizados e o próximo exemplo é processado pela rede. Vejamos a esquematização desse processo.

Figura 3.1 – Retropropagação.



Fonte: Elaborado pelo autor.

O algoritmo realiza os passos ilustrados na Figura 3.1 sucessivamente até que todos os dados sejam processados, isto é, após X_2, segue-se com $X_3, \ldots, X_n$.

3.2 CONCEITOS PRELIMINARES

Seja

$$\mathcal{T} = \{x(n), d(n)\}_{n=1}^{N}$$

um conjunto com N exemplos de treino onde $x(n)$ e $d(n)$ são, respectivamente, os vetores de entrada e de resposta desejada do exemplo n. Sendo $a_j^{(L)}(n)$ o sinal produzido pelo j-ésimo neurônio da camada de saída L, definimos o seu sinal de erro como

$$e_j(n) = d_j(n) - a_j^{(L)}(n). \tag{8}$$

Desse modo, definimos como a energia de erro instantâneo do neurônio j como

$$\mathscr{E}_j(n) = \frac{1}{2} e_j(n)^2.$$

E, assim, denotamos o erro total da rede como

$$\begin{aligned} \mathscr{E}(n) &= \sum_{j=1}^{C} \mathscr{E}_j(n) \\ &= \sum_{j=1}^{C} \frac{1}{2} e_j(n)^2 \\ &= \frac{1}{2} \sum_{j=1}^{C} e_j(n)^2, \end{aligned} \tag{9}$$

onde C é o número de neurônios da camada de saída.

A retropropagação se divide em duas fases: a fase para frente (*foward*), vista na Seção 1.3; e a fase para trás (*backward*), nela calculamos as derivadas da função de erro em relação aos pesos e os atualizamos utilizando o gradiente descendente, para minimizar $\mathscr{E}(n)$, realizando ajustes seguindo a regra

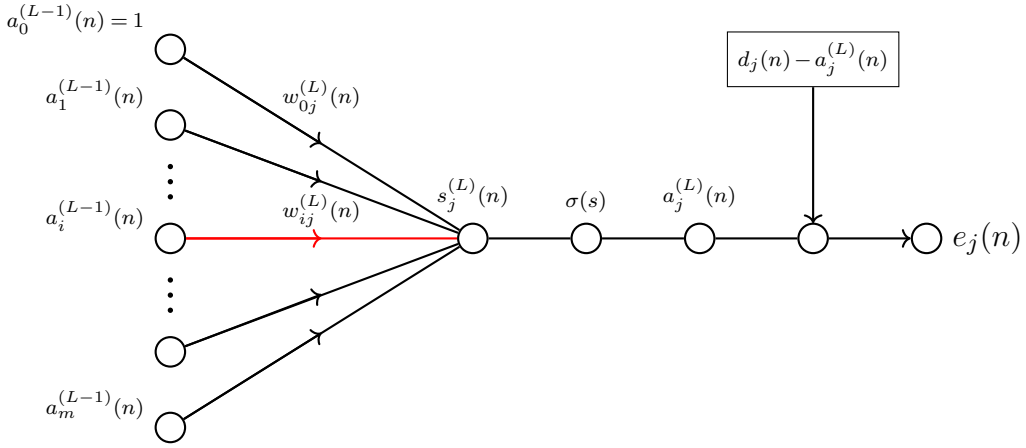
$$w_{ij}^{(l)}(n+1) = w_{ij}^{(l)}(n) - \eta \nabla \mathscr{E}(w_{ij}^{(l)}(n)). \tag{10}$$

Isso faz sentido, visto que cada peso influencia $\mathscr{E}(n)$ e a derivada nos informa justamente o tamanho dessa influência. Assim, a cada iteração é subtraido um valor proporcional ao gradiente da função de erro em relação ao peso correspondente, de modo que, com os valores dos pesos atualizados, a função se aproxima de um mínimo, tal como discutido na Seção 2.4.

3.3 GRADIENTE EM RELAÇÃO A UM NEURÔNIO DE SAÍDA

O esquema da Figura 3.2 ilustra o j-ésimo neurônio de saída da rede e suas conexões com a camada anterior.

Figura 3.2 – Neurônio de saída j.



Fonte: Adaptado de Haykin (1999).

Sua soma ponderada é dada por

$$s_j^{(L)}(n) = \sum_{i=0}^{m} w_{ij}^{(L)}(n) a_i^{(L-1)}(n) \tag{11}$$

onde m é o número de neurônios da camada $(L-1)$ excluindo o *bias*. Após computada, a soma de (11) passa pela função de ativação produzindo, assim, a ativação de j

$$a_j^{(L)}(n) = \sigma(s_j^{(L)}(n)). \tag{12}$$

Para ajustarmos $w_{ij}^{(L)}(n)$ precisamos encontrar $\nabla \mathscr{E}(w_{ij}^{(L)}(n))$, que corresponde a derivada parcial

$$\frac{\partial \mathscr{E}(n)}{\partial w_{ij}^{(L)}(n)} = \frac{\partial \mathscr{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{a_j^{(L)}(n)} \frac{\partial a_j^{(L)}(n)}{\partial s_j^{(L)}(n)} \frac{\partial s_j^{(L)}(n)}{\partial w_{ij}^{(L)}(n)}. \tag{13}$$

O que estamos calculando em (13) é uma das coordenadas do vetor gradiente da função de erro com relação a entrada, os pesos, referentes ao exemplo de treino n. No geral, a retropropagação calcula

$$\nabla \mathscr{E} = \left(\frac{\partial \mathscr{E}}{\partial w_1}, \frac{\partial \mathscr{E}}{\partial w_2}, \ldots, \frac{\partial \mathscr{E}}{\partial w_k} \right) \tag{14}$$

sendo k o número total de pesos da rede.

De volta a (13), temos que, diferenciando (9),

$$\begin{aligned}\frac{\partial \mathscr{E}(n)}{\partial e_j(n)} &= \frac{1}{2}(2e_j(n)) \\ &= e_j(n). \end{aligned} \tag{15}$$

Da Equação (8), concluimos que

$$\frac{\partial e_j(n)}{a_j^{(L)}(n)} = -1. \tag{16}$$

A próxima parcela corresponde a $s_j^{(L)}(n)$,

$$\frac{\partial a_j^{(L)}(n)}{\partial s_j^{(L)}(n)} = \sigma'(s_j^{(L)}(n)). \tag{17}$$

A derivada de (17) dependerá da função de ativação aplicada. Como última parcela temos

$$\frac{\partial s_j^{(L)}(n)}{\partial w_{ij}^{(L)}(n)} = a_i^{(L-1)}(n). \tag{18}$$

Assim, concluimos que

$$\frac{\partial \mathscr{E}(n)}{\partial w_{ij}^{(L)}(n)} = -e_j(n)\sigma'(s_j^{(L)}(n))a_i^{(L-1)}(n). \tag{19}$$

Por fim, reescrevemos (19) como

$$\frac{\partial \mathscr{E}(n)}{\partial w_{ij}^{(L)}(n)} = \delta_j^{(L)}(n)a_i^{(L-1)}(n), \tag{20}$$

onde $\delta_j^{(L)}(n) = e_j(n)\sigma'(s_j^{(L)}(n))$ é definido como gradiente local de j.

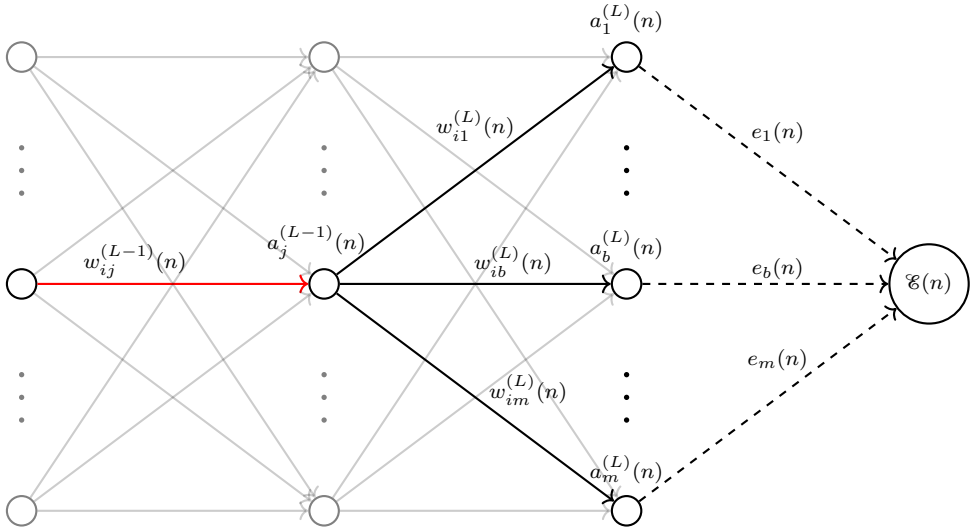
Nessa seção, abordamos a correção de um peso de conexão a um neurônio da camada de saída. Esse caso é o mais fácil, visto que há uma resposta desejada para ele e podemos calcular o erro correspondente. O caso mais complexo é quando o peso se conecta a um neurônio de uma camada oculta, não há uma resposta para ele e acessá-los não é uma tarefa muito simples. Por outro lado, ele também é responsável pela resposta produzida pelos neurônios de saída. Assim, utilizamos os gradientes locais dos neurônios das camadas posteriores para realizar a correção.

3.4 GRADIENTE EM RELAÇÃO A UM NEURÔNIO OCULTO

Nesse caso, queremos o gradiente em relação a um peso que se conecta a um neurônio de uma camada oculta. Para desenvolvermos as equações, consideremos que

ele esteja na camada $(L-1)$. Vejamos o esquema da Figura (3.3), não há uma resposta desejada para o neurônio. No entanto, ele está diretamente conectado a todos os neurônios de saída. Desse modo, para acessá-lo, devemos agir de forma recursiva (de frente para trás) considerando os erros de todos os neurônios de saída.

Figura 3.3 – Esquema para compreensão do gradiente em relação a $w_{ij}^{(L-1)}$.



Fonte: Elaborado pelo autor.

Precisamos encontrar $\nabla\mathscr{E}(w_{ij}^{(L-1)}(n))$, utilizando a regra da cadeia

$$\frac{\partial\mathscr{E}(n)}{\partial w_{ij}^{(L-1)}(n)} = \frac{\partial\mathscr{E}(n)}{\partial a_j^{(L-1)}(n)}\frac{\partial a_j^{(L-1)}(n)}{\partial s_j^{(L-1)}(n)}\frac{\partial s_j^{(L-1)}(n)}{\partial w_{ij}^{(L-1)}(n)}. \tag{21}$$

Entretanto, não temos a primeira derivada já que não há resposta desejada para neurônios de camadas ocultas, consequentemente, não é possível definir um erro para ele. Assim, usamos mais uma vez o conceito de regra da cadeia para conseguir essa derivada. Como mostrado na Figura (3.3), $w_{ij}^{(L-1)}(n)$ está conectado a m neurônios de saída. Assim, conseguimos essa derivada da seguinte forma:

$$\begin{aligned}
\frac{\partial\mathscr{E}(n)}{\partial a_j^{(L-1)}(n)} &= \sum_{k=1}^{m}\left(\frac{\partial\mathscr{E}(n)}{\partial e_k(n)}\frac{\partial e_k(n)}{\partial a_k^{(L)}(n)}\frac{\partial a_k^{(L)}(n)}{\partial s_j^{(L)}(n)}\frac{\partial s_j^{(L)}(n)}{\partial a_j^{(L-1)}(n)}\right) \\
&= \sum_{k=1}^{m}\left(e_k(n)(-1)\sigma'(s_j^{(L)}(n))w_{jk}^{(L)}(n)\right) \\
&= -\sum_{k=1}^{m} e_k(n)\sigma'(s_j^{(L)}(n))w_{jk}^{(L)}(n) \\
&= -\sum_{k=1}^{m}\delta_k^{(L)}(n)w_{jk}^{(L)}(n)
\end{aligned} \tag{22}$$

Associando essa derivada parcial ao que foi discutido na subseção Regra da Cadeia, percebemos que $a_j^{(L-1)}(n)$ é uma variável independente e as demais são intermediárias. Ademais, a Equação (22) nos mostra que utilizamos a soma de todos os gradientes locais, da camada imediatamente posterior a do peso, para conseguir o gradiente da função de erro em relação a ele. Substituindo (22) em (21), temos

$$\begin{aligned}\frac{\partial \mathscr{E}(n)}{\partial w_{ij}^{(L-1)}(n)} &= -\sum_{k=1}^{m} \delta_k^{(L)}(n) w_{jk}^{(L)}(n) \frac{\partial a_j^{(L-1)}(n)}{\partial s_j^{(L-1)}(n)} \frac{\partial s_j^{(L-1)}(n)}{\partial w_{ij}^{(L-1)}(n)} \\ &= -\frac{\partial a_j^{(L-1)}(n)}{\partial s_j^{(L-1)}(n)} \frac{\partial s_j^{(L-1)}(n)}{\partial w_{ij}^{(L-1)}(n)} \sum_{k=1}^{m} \delta_k^{(L)}(n) w_{jk}^{(L)}(n) \\ &= -\sigma'(s_j^{(L-1)}(n)) a_i^{(L-2)}(n) \sum_{k=1}^{m} \delta_k^{(L)}(n) w_{jk}^{(L)}(n). \end{aligned} \tag{23}$$

Desse modo, o cálculo do gradiente em relação a um peso qualquer é definido por

$$\nabla \mathscr{E}(w_{ij}^{(l)}(n)) = \begin{cases} \delta_j^{(L)}(n) a_i^{(L-1)}(n), & \text{se } j \in L \\ \sigma'(s_j^{(l)}(n)) a_i^{(l-1)}(n) \sum_{k=1}^{m} \delta_k^{(l+1)}(n) w_{jk}^{(l+1)}(n), & \text{se } j \notin L \end{cases}, \tag{24}$$

onde m agora passa a representar o número de neurônios da camada $(l+1)$ excluindo o *bias*.

3.5 FUNÇÃO DE ATIVAÇÃO SIGMOIDE

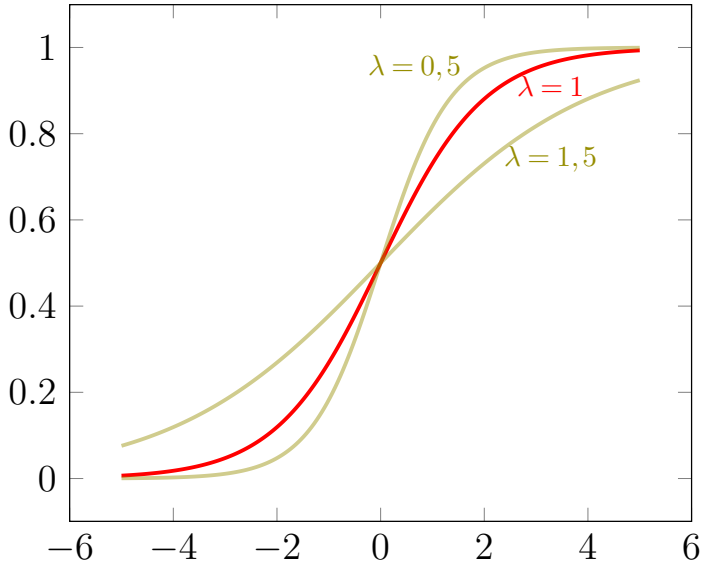
Para trabalharmos com a retropropagação é necessário termos conhecimento sobre a derivada da(s) função(ões) de ativação utilizada(s). Logo, os requisitos para essas funções são a diferenciabilidade e, consequentemente, continuidade. As funções sigmoidais são comumente utilizadas em redes MLP, elas recebem esse nome por conta do seus gráficos que se assemelham a letra S.

Um exemplo é a função sigmóide, ou logística, definida por

$$\sigma(x) = \frac{1}{1+e^{-(\lambda x)}}, \quad \lambda > 0.$$

A constante λ é um parâmetro para ajustar a inclinação da curva como ilustrado na Figura 3.4.

Figura 3.4 – Função sigmoide.



Fonte: Elaborado pelo autor.

Assim sendo, temos que

$$\begin{aligned}
\sigma'(x) &= -\frac{(-\lambda e^{(-\lambda x)})}{(1+e^{(-\lambda x)})^2} \\
&= \frac{\lambda e^{(-\lambda x)}}{(1+e^{(-\lambda x)})^2} \\
&= \frac{\lambda e^{(-\lambda x)}}{(1+e^{(-\lambda x)})(1+e^{(-\lambda x)})} \\
&= \frac{1}{1+e^{(-\lambda)}}\lambda\left(\frac{e^{(-\lambda x)}}{1+e^{(-\lambda x)}}\right) \\
&= \sigma(x)\lambda\left(\frac{e^{(-\lambda x)}+1-1}{1+e^{(-\lambda x)}}\right) \\
&= \lambda\sigma(x)\left(\frac{e^{(-\lambda x)}+1}{1+e^{(-\lambda x)}} - \frac{1}{1+e^{(-\lambda x)}}\right) \\
&= \lambda\sigma(x)(1-\sigma(x)). \qquad (25)
\end{aligned}$$

Esse resultado será útil para a seção posterior.

3.6 EFETIVIDADE DO ALGORITMO DE RETROPROPAGAÇÃO

O algoritmo de retropropagação é computacionalmente eficiente dada as operações que ele efetua, porém, também possui limitações. Como visto no Exemplo 2.4.2, alguns fatores podem influenciar negativamente no desempenho do algoritmo, como a existência de mínimos locais que podem fazer com que ele estacione e não encontre soluções melhores, podendo fazer com que cada pequena mudança nos pesos influencie negativamente na função de erro. Outrossim, por ser um método estocástico, a convergência tende a acontecer

lentamente uma vez que o vetor gradiente também pode apontar para direção errada a depender do vetor de pesos (HAYKIN, 1999).

Nessa seção, buscamos ilustrar, por meio de um exemplo, como a atualização dos pesos faz com que a saída para um dado de treino específico se aproxime da sua respectiva resposta desejada.

Seja $x(1) = (0,5;\ 0,2)$ e $d(1) = (1,0)$. Para esta simulação utilizaremos a rede da Seção 1.3. Assim, a iniciaremos com os pesos da Tabela 3.1.

Tabela 3.1 – Pesos iniciais.

$l=1$		$l=2$	
w_{01}	1	w_{01}	-1
w_{02}	$0,1$	w_{02}	$0,3$
w_{03}	1	w_{11}	$1,2$
w_{11}	$0,5$	w_{12}	$0,5$
w_{12}	2	w_{21}	1
w_{13}	$1,5$	w_{22}	$0,4$
w_{21}	$0,9$	w_{31}	$-0,8$
w_{22}	$0,2$	w_{32}	1
w_{23}	$0,3$		

Fonte: Elaborado pelo autor.

Calculando $A^{(1)}$:

$$\begin{aligned} AW + W_0 &= \begin{bmatrix} 0,5 & 0,2 \end{bmatrix} \begin{bmatrix} 0,5 & 2 & 1,5 \\ 0,9 & 0,2 & 0,3 \end{bmatrix} + \begin{bmatrix} 1 & 0,1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0,43 & 1,04 & 0,81 \end{bmatrix} + \begin{bmatrix} 1 & 0,1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1,43 & 1,14 & 1,81 \end{bmatrix}. \end{aligned} \tag{26}$$

Adotaremos a função de ativação sigmoide com $\lambda = 1$ para esse exemplo, assim,

$$\begin{aligned} A^{(1)} &= \sigma\left(\begin{bmatrix} 1,43 & 1,14 & 1,81 \end{bmatrix}\right) \\ &= \begin{bmatrix} \sigma(1,43) & \sigma(1,14) & \sigma(1,81) \end{bmatrix} \\ &= \begin{bmatrix} 0,80690132 \\ 0,75767964 \\ 0,85936187 \end{bmatrix}^T \end{aligned}$$

A notação de matriz transposta tem como único fim a melhoria da estética.

Calculando $A^{(2)}$:

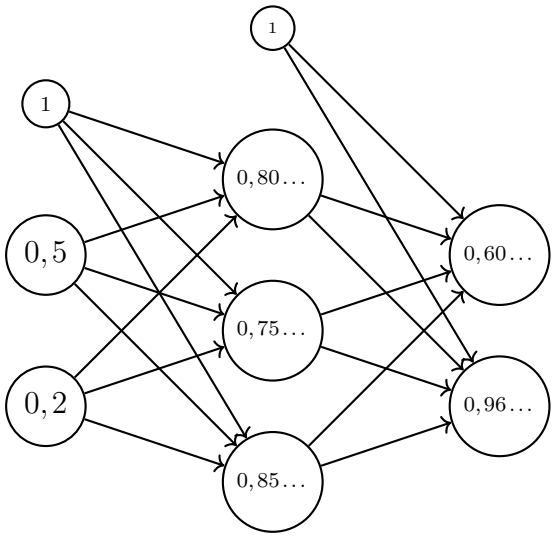
$$\begin{aligned} A^{(1)}W + A_0 &= \begin{bmatrix} 0,80690132 \\ 0,75767964 \\ 0,85936187 \end{bmatrix}^T \begin{bmatrix} 1,2 & 0,5 \\ 1 & 0,4 \\ -0,8 & 1 \end{bmatrix} + \begin{bmatrix} -1 & 0,3 \end{bmatrix} \\ &= \begin{bmatrix} 0,408 & 3,281 \end{bmatrix}. \end{aligned}$$

Assim,

$$\begin{aligned} A^{(2)} &= \sigma\left(\begin{bmatrix} 0,408 & 3,281 \end{bmatrix}\right) \\ &= \begin{bmatrix} \sigma(0,408) & \sigma(3,281) \end{bmatrix} \\ &= \begin{bmatrix} 0,60060822 & 0,96377122 \end{bmatrix}. \end{aligned}$$

As ativações estão ilustradas na Figura 3.5.

Figura 3.5 – Ativações na rede.



Fonte: Elaborado pelo autor.

Calculando $\mathscr{E}(1)$: O erro é definido por

$$\begin{aligned} \mathscr{E}(1) &= \frac{1}{2}\sum_{j=1}^{2} e_j(1)^2 \\ &= \frac{1}{2}[e_1(1)^2 + e_2(1)^2] \\ &= \frac{1}{2}[(1-0,60060822)^2 + (0-0,96377122)^2] \\ &= 0,5441843792159284. \end{aligned}$$

Convém ressaltar que $e_1(1) = 0,39939178$ e $e_2(1) = -0,96377122$. Ademais, na Seção 3.2, a indexação n referia-se simuntaneamente ao exemplo de treino e a iteração de atualização dos pesos. Isso não faz sentido aqui, pois, diferente dos problemas reais que possuem grandes quantidades de exemplos de treino, estamos trabalhando com apenas um. Assim, no processo de atualização dos pesos, a indexação destes irá se referir especificamente a iteração.

Atualizando $w_{11}^{(2)}$:

O peso está na camada de saída, logo

$$\nabla\mathscr{E}(w_{11}^{(2)}(1)) = \delta_1^{(2)}(1)a_1^{(1)}(1).$$

Mas,

$$\begin{aligned}\delta_1^{(2)}(1) &= e_1(1)\sigma'(s_1^{(2)}(1)) \\ &= e_1(1)\sigma(s_1^{(2)}(1))(1-\sigma(s_1^{(2)}(1))) \\ &= 0,39939178 \cdot 0,60060822(1-0,60060822) \\ &= 0,09580529583868609.\end{aligned}$$

Assim,

$$\begin{aligned}\nabla\mathscr{E}(w_{11}^{(2)}(1)) &= 0,09580529583868609 \cdot 0,80690132 \\ &= 0,07730541967522632;\end{aligned}$$

portanto, tomando $\eta = 0,5$ para esse exemplo,

$$\begin{aligned}w_{11}^{(2)}(2) &= w_{11}^{(2)}(1) - \eta\nabla\mathscr{E}(w_{11}^{(2)}(1)) \\ &= 1,2 - 0,5 \cdot 0,07730541967522632 \\ &= 1,1613472901623867.\end{aligned}$$

Os demais pesos da camada são atualizados de forma análoga, vejamos a Tabela 3.2.

Tabela 3.2 – Pesos atualizados.

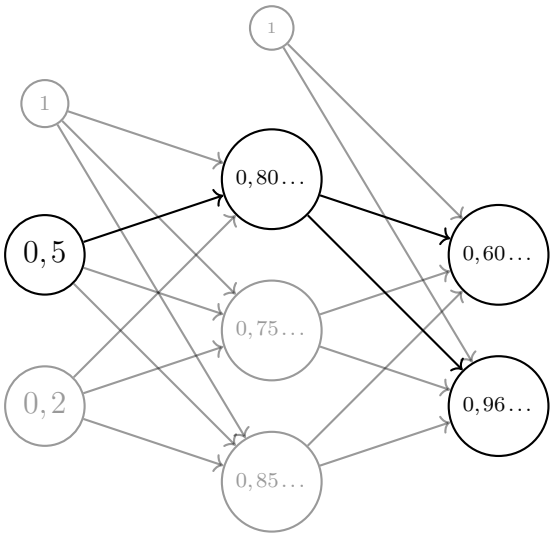
$l = 2$	
w_{01}	$-1,0386527098376133$
w_{02}	$1,1613472901623867$
w_{11}	$0,9613472901623868$
w_{12}	$-0,8386527098376132$
w_{21}	$0,31682564108039435$
w_{22}	$0,5135766319976164$
w_{31}	$0,41274844567656244$
w_{32}	$1,0144593143827965$

Fonte: Elaborado pelo autor.

Atualizando $w_{11}^{(1)}$:

O peso está em uma camada oculta. Para absorvermos melhor a equação, primeiramente analisemos a Figura 3.6. O peso $w_{11}^{(1)}$ se conecta diretamente com a saída da rede. Ao contrário do cálculo anterior, precisamos considerar dois gradientes locais: um pra cada neurônio de saída.

Figura 3.6 – Acesso a $w_{11}^{(1)}$.



Fonte: Elaborado pelo autor.

Temos que,

$$\begin{aligned}\nabla\mathscr{E}(w_{11}^{(1)}(1)) &= \sigma'(s_1^{(1)}(1))a_1^{(0)}(1)\sum_{k=1}^{2}\delta_k^{(2)}(1)w_{1k}^{(2)}(1)\\ &= 0,80690132(1-0,80690132)0,5\left(\delta_1^{(2)}(1)w_{11}^{(2)}(1)+\delta_2^{(2)}(1)w_{12}^{(2)}(1)\right)\\ &= 0,07790578989112878\left(0,09580529583868609\cdot 1,2+\delta_2^{(2)}(1)\cdot 0,5\right)\\ &= 0,07790578989112878\left(0,1149663550064233+\delta_2^{(2)}(1)\cdot 0,5\right) \qquad (27)\end{aligned}$$

Mas,

$$\begin{aligned}\delta_2^{(2)}(1) &= e_2(1)\sigma'(s_2^{(2)}(1))\\ &= e_2(1)\sigma(s_2^{(2)}(1))(1-\sigma(s_2^{(2)}(1)))\\ &= -0,96377122\cdot 0,96377122(1-0,96377122)\\ &= -0,03365128216078873. \qquad (28)\end{aligned}$$

Substituindo (29) em (28), encontramos que $\nabla\mathscr{E}(w_{11}^{(1)}(1)) = 0,007645729838886587$. Por fim, realizamos a atualização

$$\begin{aligned}w_{11}^{(1)}(2) &= 0,5-0,5\cdot 0,007645729838886587\\ &= 0,4961771350805567.\end{aligned}$$

Os pesos da camada $l=1$ após a primeira atualização estão na Tabela 3.3.

Tabela 3.3 – Pesos atualizados.

$l=1$	
w_{01}	$0,9923542701611134$
w_{02}	$0,09244069938723522$
w_{03}	$1,0066651056135527$
w_{11}	$0,4961771350805567$
w_{12}	$1,9962203496936175$
w_{13}	$1,5033325528067765$
w_{21}	$0,8984708540322227$
w_{22}	$0,19848813987744707$
w_{23}	$0,30133302112271054$

Fonte: Elaborado pelo autor.

Refazendo o cálculo das ativações, encontramos $A^{(2)} = \begin{bmatrix}0,56681681 & 0,83149713\end{bmatrix}$. Assim,

$$\begin{aligned}
\mathscr{E}(2) &= \frac{1}{2}\sum_{j=1}^{2} e_j(2)^2 \\
&= \frac{1}{2}[e_1(e)^2 + e_2(2)^2] \\
&= \frac{1}{2}[(1-0,56681681)^2 + (1-0,83149713)^2] \\
&= 0,4395175766484065.
\end{aligned}$$

Comparando o erro da segunda iteração com o da primeira, vemos que houve uma redução significativa. Esse processo, aplicado a conjuntos com muitos dados, tende a encontrar os pesos adequados. Normalmente, o número de iterações, para que a resposta da rede melhor se aproxime da desejada, é expressivo e pode variar de acordo com alguns fatores, dentre eles, a escolha inicial dos pesos.

CONSIDERAÇÕES FINAIS

Entendemos que o desenvolvimento deste trabalho possibilita uma visão holística do algoritmo de retropropagação, no que se refere aos cálculos que ele executa. Sendo assim, com relação à questão norteadora: Como o Cálculo Diferencial torna o aprendizado do algoritmo de retropropagação possível? Constatamos que o conceito de gradiente descendente é a base do funcionamento deste, uma vez que, através do vetor gradiente, realiza os ajustes interativos nos pesos sinápticos na direção correta e, consequentemente, minimiza a função de erro da rede. Assim, concluímos que o objetivo do trabalho foi alcançado.

No desenrolar da produção desse TCC me deparei com algumas dificuldades, como a leitura de textos em língua inglesa e a adequação do tema, da forma que é abordado por Haykin (1999), aos objetivos do trabalho. Enfatizo que a discussão realizada no trabalho é inicial, visto que o nível de complexidade do assunto pode se elevar. Ademais, as discussões realizadas sobre taxa de aprendizado e função de ativação foram um tanto superficiais, e assim, podem ser melhor aprofundadas em trabalhos posteriores. Nessa perspectiva, também tenho o interesse de trabalhar com o algoritmo de retropropagação em algum problema prático dentro de suas variadas possibilidades de aplicação.

Do ponto de vista pessoal, posso afirmar que o estudo sobre o tema possibilitou-me um entendimento mais aprofundado dos tópicos de matemática, tanto os que foram trabalhados nas disciplinas de cálculo, quanto o gradiente descendente, conteúdo não discutido na minha graduação e ausente nos livros adotados pelos professores. Além disso, no decorrer da pesquisa aprendi a programar, dado que parte dos cálculos desse trabalho foram realizados utilizando a linguagem de programação Python.

Por fim, enxergo que o trabalho pode contribuir para o ensino de cálculo em caráter interdisciplinar, onde os discentes podem ver outra aplicação dos conteúdos estudados. Outra contribuição se refere ao campo da IA, visto que pode servir de fundamento para outras pesquisas.

REFERÊNCIAS

BRAGA, A. P.; CARVALHO, A. P. L. F.; LUDEMIR, T. B. **Redes Neurais Artificiais**: Teoria e aplicações. 2. ed. Rio de Janeiro: LTC, 2007.

DEISENROTH, M. P.; FAISAL, A. A.; ONG, C. S. **Mathematics for machine learning**. [S.l.]: Cambridge University Press, 2020.

FLECK, L.; TAVARES, M. H. F.; EYNG, E.; HELMANN, A.; ANDRADE, M. d. M. Redes neurais artificiais: Princípios básicos. **Revista Eletrônica Científica Inovação e Tecnologia**, v. 1, n. 13, p. 47–57, 2016.

FONTELLES, M. J.; SIMÕES, M. G.; FARIAS, S. H.; FONTELLES, R. G. S. Metodologia da pesquisa científica: diretrizes para a elaboração de um protocolo de pesquisa. **Revista paraense de medicina**, v. 23, n. 3, p. 1–8, 2009.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. Disponível em: <http://www.deeplearningbook.org>. Acesso em 12 de dez. 2021.

HAYKIN, S. **Neural Networks and Learning Machines**. 3. ed. Hamilton: Pearson, 1999.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943.

NIELSEN, M. A. **Neural networks and deep learning**. São Francisco: Determination press, 2015. v. 25. Disponível em: <http://neuralnetworksanddeeplearning.com/>. Acesso em 19 de jan. 2022.

RAUBER, T. W. Redes neurais artificiais. **Universidade Federal do Espírito Santo**, v. 29, 2005.

RODRIGUES, W. C. et al. Metodologia científica. **Faetec/IST. Paracambi**, p. 2–20, 2007.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. 1985.

SICHMAN, J. S. Inteligência artificial e sociedade: avanços e riscos. **Estudos Avançados**, SciELO Brasil, v. 35, p. 37–50, 2021.

SILVA, J. J. S.; SOUSA, L. S.; ROCHA, V. M. T. A geometria do perceptron de rosenblatt. **Proceeding Series of the Brazilian Society of Computational and Applied Mathematics**, Sociedade de Matemática Aplicada e Computacional, São Carlos, v. 8, 2021.

STEWART, J. **Cálculo**. 7. ed. São Paulo: Cengage Learning, 2013. v. 1.

THOMAS, G. B.; WEIR, M. D.; HASS, J. **Cálculo**. 12. ed. São Paulo: Pearson Education do Brasil, 2012. v. 2.

APÊNDICE A –

A.1

t	x_1	x_2	$f(x_1, x_2)$
0	2	-4	5,4918731218
1	1,9959422112	-3,8254256878	5,3401621316
2	1,9883060799	-3,6526884040	5,1914247191
3	1,9764335103	-3,4818952119	5,0455791121
4	1,9595674196	-3,3131467389	4,9024154046
5	1,9368337019	-3,1465330321	4,7615454014
6	1,9072188590	-2,9821284523	4,6223345090
7	1,8695428302	-2,8199854413	4,4838088184
8	1,8224272528	-2,6601269721	4,3445284591
9	1,7642609897	-2,5025374837	4,2024169210
10	1,6931680760	-2,3471521476	4,0545378109
11	1,6069896648	-2,1938445079	3,8968214960
12	1,5033033525	-2,0424131102	3,7237775570
13	1,3795234861	-1,8925692074	3,5283103725
14	1,2331571326	-1,7439311521	3,3019227604
15	1,0623285562	-1,5960390719	3,0358704225
16	0,8667051299	-1,4484202580	2,7241122565
17	0,6488837427	-1,3007663168	2,3686791906
18	0,4159783079	-1,1533190557	1,9862300015
19	0,1804499260	-1,0075272279	1,6104415657
20	-0,0415233382	-0,8667097848	1,2824636830
21	-0,2350784186	-0,7357843933	1,0306631736
22	-0,3923693899	-0,6193243380	0,8575874632
23	-0,5138834134	-0,5194305992	0,7469380466
24	-0,6050670903	-0,4355500582	0,6785038874
25	-0,6725412241	-0,3657123808	0,6364118000
26	-0,7221849568	-0,3076345711	0,6102689115
27	-0,7586435053	-0,2592375191	0,5937581996
28	-0,7854162643	-0,2187852489	0,5831306641
29	-0,8050887145	-0,1848716341	0,5761604370
30	-0,8195569090	-0,1563652129	0,5715096319
31	-0,8302080643	-0,1323524912	0,5683594777
32	-0,8380569448	-0,1120908064	0,5661985543
33	-0,8438463929	-0,0949717387	0,5647006944

34	-0,8481207593	-0,0804932980	0,5636536913
35	-0,8512793888	-0,0682387868	0,5629169483
36	-0,8536155714	-0,0578605885	0,5623958127
37	-0,8553449370	-0,0490675703	0,5620256890
38	-0,8566261784	-0,0416151529	0,5617619936
39	-0,8575761984	-0,0352973780	0,5615736707
40	-0,8582811942	-0,0299404947	0,5614389284
41	-0,8588047792	-0,0253977202	0,5613423865
42	-0,8591939409	-0,0215449241	0,5612731404
43	-0,8594834155	-0,0182770499	0,5612234319
44	-0,8596989039	-0,0155051306	0,5611877263
45	-0,8598594372	-0,0131537910	0,5611620667
46	-0,8599791196	-0,0111591497	0,5611436199
47	-0,8600684117	-0,0094670524	0,5611303548
48	-0,8601350783	-0,0080315821	0,5611208137
49	-0,8601848877	-0,0068138007	0,5611139501
50	-0,8602221283	-0,0057806843	0,5611090120
51	-0,8602499904	-0,0049042231	0,5611054589
52	-0,8602708499	-0,0041606583	0,5611029021
53	-0,8602864768	-0,0035298360	0,5611010622
54	-0,8602981912	-0,0029946599	0,5610997381
55	-0,8603069781	-0,0025406266	0,5610987852
56	-0,8603135730	-0,0021554327	0,5610980994
57	-0,8603185257	-0,0018286404	0,5610976058
58	-0,8603222472	-0,0015513947	0,5610972505
59	-0,8603250451	-0,0013161834	0,5610969948
60	-0,8603271498	-0,0011166334	0,5610968108
61	-0,8603287338	-0,0009473379	0,5610966784
62	-0,8603299265	-0,0008037098	0,5610965830
63	-0,8603308251	-0,0006818575	0,5610965144
64	-0,8603315024	-0,0005784796	0,5610964650
65	-0,8603320131	-0,0004907750	0,5610964295
66	-0,8603323983	-0,0004163676	0,5610964039
67	-0,8603326891	-0,0003532412	0,5610963855
68	-0,8603329086	-0,0002996856	0,5610963722
69	-0,8603330744	-0,0002542496	0,5610963627
70	-0,8603331997	-0,0002157023	0,5610963558
71	-0,8603332944	-0,0001829992	0,5610963509

72	-0,8603333660	-0,0001552543	0,5610963473
73	-0,8603334201	-0,0001317159	0,5610963448
74	-0,8603334611	-0,0001117462	0,5610963429
75	-0,8603334921	-0,0000948041	0,5610963416
76	-0,8603335156	-0,0000804307	0,5610963406
77	-0,8603335334	-0,0000682364	0,5610963400
78	-0,8603335468	-0,0000578910	0,5610963395
79	-0,8603335570	-0,0000491140	0,5610963391
80	-0,8603335647	-0,0000416677	0,5610963388
81	-0,8603335706	-0,0000353504	0,5610963387
82	-0,8603335751	-0,0000299908	0,5610963385
83	-0,8603335784	-0,0000254439	0,5610963384
84	-0,8603335810	-0,0000215863	0,5610963384
85	-0,8603335829	-0,0000183135	0,5610963383
86	-0,8603335844	-0,0000155370	0,5610963383
87	-0,8603335855	-0,0000131814	0,5610963383
88	-0,8603335863	-0,0000111829	0,5610963382

A.2

t	x_1	x_2	$f(x_1, x_2)$
0	2,0000000000	-4,0000000000	5,4918731218
1	1,9857977393	-3,3889899073	4,9683422748
2	1,9250805144	-2,8037048459	4,4828486332
3	1,7877901709	-2,2500746359	4,0194144250
4	1,5272220667	-1,7324532291	3,5157552706
5	1,0719458746	-1,2505860609	2,8050564173
6	0,3476449636	-0,7962813998	1,6654189480
7	-0,4941777077	-0,3755200740	0,7026487409
8	-0,8165147196	-0,1521779025	0,5711803376
9	-0,8561697220	-0,0702324250	0,5629788075
10	-0,8602954246	-0,0329404042	0,5615075570
11	-0,8604695816	-0,0154659560	0,5611870024
12	-0,8603873206	-0,0072601840	0,5611163180
13	-0,8603493376	-0,0034077853	0,5611007400
14	-0,8603377017	-0,0015994910	0,5610973079
15	-0,8603346008	-0,0007507361	0,5610965518
16	-0,8603338293	-0,0003523642	0,5610963853
17	-0,8603336448	-0,0001653849	0,5610963486
18	-0,8603336018	-0,0000776247	0,5610963405
19	-0,8603335919	-0,0000364338	0,5610963387
20	-0,8603335897	-0,0000171005	0,5610963383
21	-0,8603335892	-0,0000080262	0,5610963382

A.3

t	x_1	x_2	$f(x_1, x_2)$
0	2,0000000000	-4,0000000000	4,5000189516
1	3,0603490590	-2,8623505597	4,3891453226
2	3,1176115889	-2,7294649287	4,2865461635
3	3,1712754998	-2,6013982180	4,1922003577
4	3,2209568683	-2,4781640697	4,1059144463
5	3,2664058730	-2,3597388184	4,0273504996
6	3,3075026700	-2,2460668623	3,9560623123
7	3,3442452102	-2,1370665616	3,8915338483
8	3,3767316305	-2,0326360771	3,8332149105
9	3,4051399131	-1,9326587248	3,7805507663
10	3,4297071468	-1,8370075931	3,7330042967
11	3,4507101291	-1,7455493234	3,6900707171
12	3,4684484073	-1,6581470638	3,6512858669
13	3,4832302866	-1,5746626813	3,6162294951
14	3,4953618923	-1,4949583505	3,5845250241
15	3,5051390872	-1,4188976463	3,5558370914
16	3,5128418754	-1,3463462623	3,5298678904
17	3,5187308622	-1,2771724578	3,5063530411
18	3,5230453362	-1,2112473183	3,4850574685
19	3,5260025784	-1,1484448935	3,4657715648
20	3,5277980617	-1,0886422612	3,4483077748
21	3,5286062630	-1,0317195491	3,4324976444
22	3,5285818717	-0,9775599371	3,4181893216
23	3,5278612296	-0,9260496529	3,4052454640
24	3,5265638810	-0,8770779688	3,3935414970
25	3,5247941461	-0,8305372016	3,3829641654
26	3,5226426598	-0,7863227182	3,3734103235
27	3,5201878369	-0,7443329431	3,3647859159
28	3,5174972410	-0,7044693687	3,3570051110
29	3,5146288452	-0,6666365644	3,3499895535
30	3,5116321800	-0,6307421828	3,3436677122
31	3,5085493700	-0,5966969622	3,3379743029
32	3,5054160630	-0,5644147227	3,3328497711
33	3,5022622588	-0,5338123553	3,3282398245

34	3,4991130445	-0,5048098039	3,3240950061
35	3,4959892447	-0,4773300392	3,3203703022
36	3,4929079949	-0,4512990247	3,3170247804
37	3,4898832448	-0,4266456752	3,3140212550
38	3,4869262011	-0,4033018084	3,3113259761
39	3,4840457132	-0,3812020894	3,3089083417
40	3,4812486125	-0,3602839699	3,3067406299
41	3,4785400061	-0,3404876217	3,3047977511
42	3,4759235347	-0,3217558664	3,3030570180
43	3,4734015951	-0,3040341005	3,3014979336
44	3,4709755352	-0,2872702187	3,3001019947
45	3,4686458215	-0,2714145338	3,2988525109
46	3,4664121855	-0,2564196960	3,2977344386
47	3,4642737492	-0,2422406108	3,2967342278
48	3,4622291347	-0,2288343560	3,2958396822
49	3,4602765570	-0,2161600995	3,2950398307
50	3,4584139055	-0,2041790173	3,2943248105
51	3,4566388127	-0,1928542119	3,2936857597
52	3,4549487136	-0,1821506324	3,2931147205
53	3,4533408964	-0,1720349959	3,2926045501
54	3,4518125452	-0,1624757108	3,2921488401
55	3,4503607767	-0,1534428010	3,2917418441
56	3,4489826708	-0,1449078335	3,2913784110
57	3,4476752969	-0,1368438473	3,2910539251
58	3,4464357346	-0,1292252846	3,2907642521
59	3,4452610926	-0,1220279243	3,2905056902
60	3,4441485226	-0,1152288182	3,2902749257
61	3,4430952316	-0,1088062294	3,2900689934
62	3,4420984914	-0,1027395725	3,2898852406
63	3,4411556462	-0,0970093571	3,2897212947
64	3,4402641183	-0,0915971328	3,2895750344
65	3,4394214123	-0,0864854372	3,2894445632
66	3,4386251184	-0,0816577455	3,2893281861
67	3,4378729143	-0,0770984228	3,2892243887
68	3,4371625661	-0,0727926783	3,2891318179
69	3,4364919285	-0,0687265217	3,2890492649
70	3,4358589449	-0,0648867217	3,2889756503

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Cajazeiras
Rua José Antônio da Silva, 300, Jardim Oásis, CEP 58.900-000, Cajazeiras (PB)
CNPJ: 10.783.898/0005-07 - Telefone: (83) 3532-4100

Documento Digitalizado Restrito

TCC

Assunto: TCC
Assinado por: Jorge Souza
Tipo do Documento: Relatório
Situação: Finalizado
Nível de Acesso: Restrito
Hipótese Legal: Informação Pessoal (Art. 31 da Lei no 12.527/2011)
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- Jose Jorge de Souza Silva, ALUNO (201812020037) DE LICENCIATURA EM MATEMÁTICA - CAJAZEIRAS, em 20/05/2022 16:13:41.

Este documento foi armazenado no SUAP em 20/05/2022. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 523448
Código de Autenticação: 801dc5d432