Universidade Federal Fluminense

LUCAS DE SOUZA KORT CAMP

Machine Learning e otimização restrita aplicados a Manutenções Preventivas

SANTO ANTÔNIO DE PÁDUA

2022

LUCAS DE SOUZA KORT CAMP

Machine Learning e otimização restrita aplicados a Manutenções Preventivas

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia da Universidade Federal Fluminense, como requisito parcial para obtenção do título de Mestre em Modelagem Computacional em Ciência e Tecnologia. Área de Concentração: Modelagem Computacional.

Orientador:

Cecília Toledo Hernandez

Coorientador:

Eliane Da Silva Christo

Universidade Federal Fluminense

SANTO ANTÔNIO DE PÁDUA

2022

Ficha catalográfica automática - SDC/BEM
Gerada com informações fornecidas pelo autor

C186m Camp, Lucas de Souza Kort
Machine Learning e otimização restrita aplicados a Manutenções Preventivas / Lucas de Souza Kort Camp ; Cecília Toledo Hernandez, orientador ; Eliane da Silva Christo, coorientador. Volta Redonda, 2022.
146 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Volta Redonda, 2022.

DOI: http://dx.doi.org/10.22409/PPG-MCCT.2022.m.14062806746

1. Manutenção Industrial. 2. Otimização. 3. Aprendizado de Máquina. 4. Programa de computador. 5. Produção intelectual. I. Hernandez, Cecília Toledo, orientador. II. Christo, Eliane da Silva, coorientador. III. Universidade Federal Fluminense. Escola de Engenharia Industrial e Metalúrgica de Volta Redonda. IV. Título.

CDD -

Bibliotecário responsável: Debora do Nascimento - CRB7/6368

Machine Learning e Otimização restrita aplicados a Manutenções Preventivas

Lucas de Souza Kort Camp

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia da Universidade Federal Fluminense, como requisito parcial para obtenção do título de Mestre em Modelagem Computacional em Ciência e Tecnologia. Área de Concentração: Modelagem Computacional.

Aprovada por:

Prof. Cecília Toledo Hernandez, D.Sc./MCCT-UFF(Presidente)

Prof. Tiago Araújo Neves, D.Sc./MCCT-UFF

Prof. Rafael Alves Bonfim de Queiroz, D.Sc./UFOP

Santo Antônio de Pádua, 25 de abril de 2022.

Ao meu pai, Marcelo, in memoriam.

Agradecimentos

Agradeço a minha família por todo o suporte e incentivo ao decorrer dessa jornada, em especial minha mãe, Ana, e meu tio Márcio, por especial apoio e colaboração. Aos meus avós, por sua sabedoria e ensinamentos.

Agradeço aos meus orientadores, Cecilia Hernandez e Eliane Christo, pelo carinho e toda a ajuda, tão necessária para construir esse estudo.

Agradeço também aos meus professores do programa de mestrado, Gustavo Semaan, Rodolfo Oliveira, Thiago Pereira, Tibério Vale e Wagner Telles por todo ensinamento passado.

Agradeço a Andressa Silva e aos meus demais colegas de curso, que entraram comigo e criaram um ambiente amigável e colaborativo tão logo quanto nos conhecemos.

Agradeço aos professores Rafael Queiroz e Tiago Neves, que participaram da banca de qualificação. Os comentários fornecidos ajudaram a melhorar a qualidade desse trabalho.

Agradeço a comunidade GitHub e Stack Overflow. Boa parte desse estudo só foi possível graças a códigos de computador e estes não seriam possíveis sem a colaboração dessas comunidades e de seus membros.

Resumo

Esse trabalho tem como objetivo geral propor um modelo integrado para agendamento ótimo do intervalo de manutenções preventivas, por meio de previsão de dados de tempo até falha por *Machine Learning* e otimização restrita. Foram gerados bancos de dados sintéticos, que foram usados em treinamento de Redes Neurais Artificiais. A otimização do modelo criado ocorreu por meio do método *Particle Swarm Optimization* no aplicativo Opp!. O ajuste dos dados por Redes Neurais Artificiais apresentou erro absoluto percentual médio de 0,4195%, inferior a estudos comparáveis. A otimização encontrou um intervalo para manutenção ótimo em todas os cenários e restrições aplicadas. Tanto o ajuste por Redes Neurais, quando a própria modelagem de custo de operação e otimização se mostraram poderosas ferramentas para agendamento de operações de manutenções preventivas. O trabalho resultou na construção do aplicativo OCM (Otimizador do Custo de Manutenção).

Abstract

The general objective of this work is to propose an integrated model for optimal scheduling of the preventive maintenance interval through time-to-failure data prediction by Machine Learning and constrained optimization. Synthetic databases have been generated for artificial Neural Networks training. The optimization of the created model took place through the Particle Swarm Optimization method in the application Opp!. The adjustment of data by Artificial Neural Networks showed a mean absolute percentage error of 0.4195%, lower than comparable studies. The optimization found an optimal maintenance interval in all scenarios and applied constraints. The Neural Network adjustment and the operation cost modeling and optimization have proved to be powerful tools for preventive maintenance scheduling. The work resulted in the programming of the OCM (Maintenance Cost Optimizer) application.

Palavras-chave

1. Otimização
2. Machine Learning
3. Distribuição Weibull
4. Manutenção Industrial
5. Manutenção Preventiva

Glossário

A	:	Availability
APE	:	Absolute Percentile Error
ASE	:	Absolute Squared Error
BD	:	Banco de dados
CC	:	Coeficiente de Correlação
CCDF	:	Complementary Cumulative Distribution Function
CD	:	Coeficiente de Determinação
CDF	:	Cumulative Distribution Function
CPU	:	Central Processing Unit
CSV	:	Comma Separator Vector
GB	:	Gigabit(s)
IoF	:	Internet of Things
ISO	:	International Organization for Standardization
MAE	:	Mean Absolute Error
MAPE	:	Mean Absolute Percentile Error
MASE	:	Mean Absolute Squared Error
MP	:	Manutenção Preventiva
MSE	:	Mean Squared Error
MTBF	:	Mean Time Between Failure
MTTF	:	Mean Time to Failure
MTTR	:	Mean Time to Repair
ML	:	Machine Learning
NBR	:	Norma Brasileira
OCM	:	Otimizador do Custo de Manutenção
PDF	:	Probability Density Function
PSO	:	Particle Swarm Optimization
R	:	Reliability
RAM	:	Random Access Memory

Glossário

RL	:	Regressão Linear
RNA	:	Redes Neurais Artificiais
SGD	:	Stochastic Gradient Descendent
TTF	:	Time to Failure

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

No cenário global atual, a competição entre as empresas é acirrada. Devido a crescente população e competição no mercado de hoje, todas as empresas querem ter sucesso em entregar seus produtos rapidamente (DSOUZA et al., 2019). A busca por redução de custos, melhorias de qualidade e confiabilidade de produtos e equipamentos são definidas como a forma de entregar produtos melhores e mais baratos ao consumidor.

Outro detalhe importante é o tempo de produção. Com consumo em massa de certos produtos, atualizações anuais e complexos *supply chains*, empresas e fábricas tentam otimizar seus processos produtivos para atender as altas demandas.

Para que esses sistemas funcionem, unidades produtivas precisam operar com o máximo de eficiência, e equipes de manutenção fazem o possível para manter todos os equipamentos funcionando corretamente. Quebras inesperadas de maquinários, produção de não conformidades e atrasos comprometem toda a cadeia produtiva. Em adição, é vastamente reconhecido que a manutenção de equipamentos produtivos e a qualidade de produtos manufaturados estão fortemente relacionadas (CASSADY et al., 1999).

A Manutenção tem um impacto direto e grande na confiabilidade dos equipamentos (CASSADY et al., 1999). Além disso, permitir que falhas aconteçam também pode levar a situações perigosas, somado aos custosos reparos emergenciais (RUDIN et al., 2012).

Engenheiros industriais continuamente se deparam com desafios da crescente demanda por produtos de alta qualidade. A própria engenharia de produção é caracterizada pela introdução e desenvolvimento de técnicas destinadas a atender a esses desafios de produtividade (CASSADY et al., 1999).

A transformação digital já tem um grande impacto nas técnicas e processos de manufa-

tura e requer estratégias e modelos de manutenção direcionados por dados e informações. Empregar análises preditivas de dados, baseadas em dados históricos e tempo real, permitem fazer previsões de quando uma falha irá ocorrer e recomendar um curso de ação ótimo (LEWIS, 1994).

Técnicas de aprendizado de máquinas ou *Machine Learning* (ML), termo mais utilizado na literatura técnica, levam a introdução de novos e avançados sistemas de manutenção preventiva (MP), realizadas antes do equipamento falhar, o que demonstram ser poderosas ferramentas para melhorar as atividades de manutenção e assegurar altos padrões de qualidade (DSOUZA et al., 2019).

O trabalho proposto por Dsouza et. al (2019) concluiu que a grande vantagem dos estudos de MP é o potencial para gerar redução de custos, o que é prioridade para toda organização existente hoje em dia.

Mas a prática de MP, quando não bem regulada, também pode trazer gastos. Reparar máquinas antes do momento certo e substituir peças em perfeitas condições são alguns dos custos adicionais possíveis e, se imperfeita, pode ainda trazer defeitos anteriormente inexistentes, cenário não abordado nesse trabalho. Para resolver esse problema, Wu et. al. (2011), em seu estudo, utilizam uma série de sistemas de ML e *data-mining* para classificar componentes elétricos da *Consolidated Edison of New York* conforme sua probabilidade de falha, para melhor planejar trabalhos de campo de MP. Dessa forma, uma previsão bem elaborada evita gastos desnecessários.

A grande maioria dos estudos de ML na manutenção industrial, no entanto, foca em manutenção preditiva, em equipamentos altamente sensoriais e conectados, fornecendo grande quantidade de dados de telemetria, o chamado *Internet of Things* (IoF). Os benefícios de uma manutenção preditiva, baseada em informações de sensores, que coleta dados do ambiente e ajuda a determinar a condição do equipamento, permitindo a aplicação de soluções avançadas, reduzindo custo de manutenção e situações perigosas (CIVERCHIA, 2017) tem extrema importância estratégica. Mas grandes parques industriais ainda possuem equipamentos tradicionais. Como a vida útil de um maquinário industrial é longa, os autores consideram interessante trazer benefícios similares a estes a equipamentos tradicionais.

Não sendo possível ou inviável a implantação de sistemas de sensores, a melhor opção é realizar manutenções preventivas por técnicas de previsão de falha com uso de modelos estatísticos. Já foram discutidos os benefícios de tal tipo de manutenção, mas uma correta previsão do momento certo da intervenção é também um fator estratégico.

As análises de manutenções clássicas tentam prever os momentos de falhas por meio de curvas estatísticas, sendo a modelagem *Weibull*, talvez, a mais utilizada delas (LEWIS, 1994). Existem diferentes ferramentas que tem como objetivo ajustar dados experimentais a curvas teóricas, geralmente requerendo grande quantidade de dados. Esse trabalho utiliza ML na tentativa de promover esses ajustes utilizando um pequeno volume de dados, sem perda de qualidade de previsão, mas com ganhos em simplicidade, agilidade e precisão.

O ajuste de dados de TTF a curvas de probabilidade é desejável por permitir aproveitar todas as formulações já desenvolvidas para a manutenção industrial, principalmente confiabilidade e disponibilidade. Essas equação foram utilizadas para modelar o custo de manutenção. Esse modelo foi posteriormente otimizado, com restrições, para permitir encontrar o intervalo ideal entre operações de manutenção.

1.1 Objetivos

Propor um modelo integrado para otimização do intervalo entre manutenções preventivas, por meio de previsão ajustada por ML de dados de tempo de falhas e otimização restrita com parâmetros configuráveis. Para atender esse objetivo, diversos objetivos secundários e seriados precisam de ser atingidos:

1. Ajustar dados de tempo até falha por meio de Redes Neurais Artificiais a distribuição de probabilidade *Weibull*;
2. Calcular parâmetros de manutenção industrial por meio das curvas ajustadas no item (1);
3. Modelar o processo de agendamento de manutenção por meio de uma função objetivo e uma restrição baseados nos parâmetros do item (2);
4. Otimizar a função objetiva restrita do item (3) por meio do *Particle Swarm Optimization* (PSO);
5. Construir um aplicativo para uso das ferramenta desenvolvida nos item (4).

1.2 Estrutura do trabalho

Nos próximos capítulos são introduzidos os principais conceitos utilizados nesse trabalho e nas partes finais, são expostos e comentados os principais resultados, as conclusões e sugestões de melhorias e adições.

- O capítulo 2 apresenta conceitos importantes de Otimização.
- O capítulo 3 apresenta os conceitos e funcionamentos de ML.
- O capítulo 4 apresenta as principais ferramentas e parâmetros utilizados na manutenção industrial.
- O capítulo 5 apresenta a metodologia do estudo.
- O capítulo 6 apresenta, de forma comentada, os principais resultados obtidos.
- O capítulo 7 apresenta as principais conclusões.
- O Anexo contém Tabelas CDF Padronizado
- O Apêndice contém implementações computacionais dos algoritmos utilizados nesse estudo e *link* para o repositório do trabalho no *GitHub*, com implementações computacionais em *Python* e aplicativos para *Microsoft Windows*.

Capítulo 2

Otimização

Esse capítulo apresenta uma breve definição de Otimização, critérios para existência de solução e otimalidade e uma rápida introdução aos Métodos de otimização.

Os métodos usados para a solução de um problema de otimização podem ser, basicamente, de dois grupos: determinísticos ou estocásticos. Dependendo da função objetivo é mais viável a utilização de um tipo de método de otimização em detrimento ao outro. Em muitos trabalhos analisados, encontra-se também o uso de métodos híbridos, onde se aplica em uma parte do algoritmo métodos estocásticos, onde seu uso é mais eficiente, e em outra parte métodos determinísticos.

2.1 Métodos Determinísticos

Algoritmos determinísticos utilizam um método de otimização que produzem uma sequência determinística de possíveis soluções, exigindo o uso de pelo menos a primeira derivada da função objetivo em relação às variáveis de projeto. Nestes algoritmos, a função objetivo e as restrições são dadas como funções matemáticas e relações funcionais. Além disso, a função objetivo deve ser contínua e diferenciável no espaço de busca (BASTOS, 2004).

Os métodos determinísticos são mais eficientes quando utilizados em funções contínuas, convexas e semi-modais. Outras vantagens dos métodos determinísticos, segundo Brandão (2010), é o baixo número de iterações na função objetivo, fazendo com que tenham convergência rápida e baixo custo computacional. Porém, os métodos determinísticos apresentam teoremas que lhes garantem a convergência para uma solução ótima que não é necessariamente a solução ótima global. Como nesses métodos a solução encontrada

é dependente do ponto de partida fornecido, pode-se convergir para um ótimo local, por isso não possuem bom resultado em otimizar funções multimodais, isto é, funções que possuem vários ótimos locais. Destaca-se a definição de ótimo dada por Martínez (2009): Ótimo é o mínimo ou máximo valor para a função objetivo que satisfaz todas as restrições aplicadas.

2.2 Métodos Estocásticos

Algoritmos estocásticos utilizam técnicas de otimização baseadas em algum processo aleatório, guiada por heurísticas. De acordo com Oliveira e Saramago (2005), os métodos estocásticos são aqueles que buscam o ótimo através de regras de probabilidade, operando de maneira aleatória orientada.

Segundo Brandão (2010), os métodos estocásticos podem ser denominados métodos heurísticos. Estes métodos devem ser utilizados quando a função objetivo a ser minimizada ou maximizada pode ser de difícil representação, não-diferenciável, descontínua, não-linear ou multimodal.

2.3 Revisão de otimização

Otimização é o estudo de problemas em que se busca minimizar ou maximizar uma função através da escolha sistemática dos valores de variáveis reais ou inteiras, dentro de um conjunto contido no domínio da função. Dois conceitos importantes para abordar este tema são existência de solução e condição de Otimalidade.

2.3.1 Existência de Solução

Sejam dados um conjunto $D \subset \mathbb{R}^n$ e uma função $f : \Omega \leftarrow \mathbb{R}$ onde $D \subset \Omega$. O problema de achar um minimizador da função objetivo f no conjunto D pode ser escrito como (2.1) (IZMAILOV et al., 2007):

$$\min f(x) \text{ sujeito a } x \in D. \tag{2.1}$$

Definição 2.1 Segundo Izmailov et al., (2007), um ponto $\bar{x} \in D$ é

a. Minimizador global de (2.1), se (2.2).

$$f(\bar{x}) \leq f(x),\ \forall x \in D. \tag{2.2}$$

b. Minimizador local de (2.1), se existe uma vizinhança de tal que (2.3).

$$f(\bar{x}) \leq f(x),\ \forall x \in D \cap U. \tag{2.3}$$

Definição 2.2. Izmailov et al., (2007) diz que $\bar{v} \in [-\infty, \infty]$, definido por (2.4)

$$\bar{v} = \inf\{f(x) | x \in D\}, \tag{2.4}$$

é o valor ótimo do problema (2.1).

Uma função pode admitir vários minimizadores locais, mas o mínimo global do sistema é sempre o mesmo. Por definição, todo minimizador global também é um minimizador local, mas não reciprocamente (IZMAILOV et al., 2007).

2.3.2 Condição de otimalidade

Um problema de otimização pode ser dividido em dois grupos: Funções restritas e irrestritas. Izmailov et al., (2007) diz que o problema (2.1) é irrestrito quando $D = \mathbb{R}^n$ (2.5), e quando $D \neq \mathbb{R}^n$, dizemos que ele é restrito, como em (2.6).

$$\min f(x),\ x \in \mathbb{R}^n, \tag{2.5}$$

$$\min f(x),\ x \in D. \tag{2.6}$$

Esse estudo trabalha com modelagens restritas (item 5.4).

Teorema 2.1. Condição de otimalidade no caso irrestrito (IZMAILOV et al., 2007).

a. Seja $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável no ponto $\bar{x} \in \mathbb{R}^n$. Seja ainda $\bar{x}$ um minimizador local do problema (2.5), a derivada é (2.7).

$$f'(\bar{x}) = 0. \tag{2.7}$$

Se f é duas vezes diferenciável em $\bar{x}$, então além de (2.7), tem-se que a matriz Hessiana de f no ponto $\bar{x}$ é semidefinida positiva, definida por (IZMAILOV et al.,

2007) na Equação (2.8).

$$\langle f''(\bar{x})d, d\rangle \geq 0 \quad \forall d \in \mathbb{R}^n. \tag{2.8}$$

b. Seja $f : \mathbb{R}^n \to \mathbb{R}$ duas vezes diferenciável no ponto $\bar{x} \in \mathbb{R}^n$. Se satisfaz (2.6) e se a matriz Hessiana de f em $\bar{x}$ é definida positiva, se existe $\gamma > 0$ tal que a Equação (2.9) é satisfeita.

$$\langle f''(\bar{x})d, d\rangle \geq \gamma\|d\|^2 \quad \forall d \in \mathbb{R}^n. \tag{2.9}$$

Então $\bar{x}$ é minimizador local estrito do problema (2.5) (IZMAILOV et al., 2007).

A condição (2.7) é uma condição necessária de primeira ordem para o problema expresso na Equação (2.6) , e os pontos que a satisfazem são chamados pontos estacionários deste problema. A combinação de (2.7) e (2.8) compõem a condição necessária de segunda ordem, e a combinação com (2.9) é a condição suficiente de segunda ordem para o problema expresso na Equação (2.6) (IZMAILOV et al., 2007).

2.4 Métodos de otimização

Métodos de otimização são ferramentas que permitem realizar otimizações de forma sistemática. Esse tópico apresenta noções básicas sobre métodos de otimização e algoritmo.

2.4.1 Classificação dos métodos

Seja o problema genérico dado pela Equação (2.10).

$$\min f(x) \quad \text{sujeito a } x \in D = \{x \in \Omega | h(x) = 0, g(x) \leq 0\}, \tag{2.10}$$

onde $\Omega \subset \mathbb{R}^n, f : \mathbb{R}^n \to \mathbb{R}, h : \mathbb{R}^n \to \mathbb{R}^l$ e $g : \mathbb{R}^n \to \mathbb{R}^m$.

Todo método para resolução numérica do problema da Equação (2.10) faz uso de cálculo de valores da função objetivo e das restrições, e em alguns modelos, das derivadas destas funções. Métodos sequenciais, como o utilizado nesse trabalho, geram uma sequência de pontos no $\mathbb{R}^n$, $(x^1, ..., x^k, ...)$, que são chamados de candidatos à solução do

problema. Essa sequência x^k pode não conter todos os pontos em que valores da função e derivadas são calculados, geralmente contendo apenas pontos selecionados.

O processo de geração da nova aproximação $\{x^{k+1}\}$ a partir de $\{x^k\}$ é chamado de iteração do método. A forma como esse processo ocorre, constitui a essência do método. Com frequência, os algoritmos iterativos são descritos como em (2.11).

$$x^{k+1} = \Psi_k(x^k), \quad k = 0, 1, ..., \tag{2.11}$$

onde Ψ_k é o operador com valores em $\mathbb{R}^n$, definido num conjunto $U_k \subset \mathbb{R}^n$ (IZMAILOV et al., 2007).

A Equação (2.11) ilustra um modelo sem memória, ou seja, o operador Ψ_k depende apenas de x^k , mas modelos que consideram outros pontos anteriores (não necessariamente todos) também existem.

2.4.2 Regras de parada

Na análise e avaliação de métodos com convergência assintóticas, a taxa de convergência é extremamente importante, compondo um dos principais indicadores da eficiência de um método iterativo. As estimativas de taxa de convergência fornecem uma informação útil para a comparação qualitativa de métodos diferentes. No entanto, é importante destacar que ela não é a única característica relevante nesse sentido. É indispensável levar em consideração o custo computacional de uma iteração (IZMAILOV et al., 2007).

Mesmo que um método seja de convergência assintótica, nenhum processo computacional pode ser infinito, demandando um critério de parada. As regras de paradas mais confiáveis são justamente baseadas na taxa de convergência. Mas na prática computacional, são utilizadas regras de paradas menos confiáveis do ponto de vista teórico, mas de uma menor complexidade de implementação. Essas regras se baseiam na informação obtida no ponto x^k e, talvez, alguns pontos anteriores. Fixa-se um número pequeno $\varepsilon > 0$ e analisa-se o comportamento de x^k e/ou $f(x^k)$. Quando o passo fica curto e acredita-se não ser possível melhorar a solução, o processo é encerrado. As Equações (2.12) e (2.13) expressam duas regras de parada comumente utilizados (IZMAILOV et al., 2007).

$$\|x^{k+1} - x^k\| \leq \varepsilon, \tag{2.12}$$

ou de forma análoga,

$$\|f(x^{k+1}) - f(x^k)\| \leq \varepsilon. \tag{2.13}$$

Em problemas irrestritos e em quais a função objetivo f é diferenciável, uma regra comumente usada envolve a derivada, como em (2.14) (IZMAILOV et al., 2007):

$$\|f'(x^{k+1})\| \leq \varepsilon. \tag{2.14}$$

É valido notar que, em geral, essas regras não garantem a proximidade do iterando x^{k+1} a uma solução do problema. Mesmo assim, as regras de parada supracitadas são amplamente utilizadas porque alternativas melhores não existem (IZMAILOV et al., 2007).

Ainda é importante considerar as possibilidades computacionais disponíveis. Na prática, para problemas difíceis e de grande complexidade, o método precisa ser interrompido antes mesmo de atingir algum critério de parada, por exaustão do tempo disponível. Nesses casos, a regra de parada é o número máximo de iterações que podem ser realizadas e, naturalmente, a avaliação obtida pode não ser uma boa aproximação da solução do problema, mas é a melhor avaliação permitida com as ferramentas disponíveis (IZMAILOV et al., 2007).

Em geral, a melhor prática é utilizar mais de uma regra de parada, ordenadas por uma hierarquia. A definição dessa hierarquia e de vários dos parâmetros envolvidos é uma questão da arte e da experiência computacional, mais do que da ciência exata (IZMAILOV et al., 2007).

Nesse trabalho, além do número máximo de iterações, foi utilizado a regra de parada expressa na Equação (2.13).

2.5 *Particle Swarm Optimization* (PSO)

O algoritmo PSO é um método de meta-heurístico baseado em vida artificial e na teoria do enxame. Estruturado na técnica de computação evolutiva, avalia mais de uma possível solução por iteração. Essa quantidade de soluções exploradas é chamada população.

De fácil implementação e de baixo custo computacional, o PSO não exige grandes quantidades de memória e velocidade do *Computer Processing Unit* (CPU). Surgiu a partir de estudos dos seus idealizadores do movimento de um rebanho de indivíduos

(pássaros, peixes etc.) e do esforço individual para manter uma distância ótima entre os indivíduos vizinhos (ELBERHART, 1995).

O percurso do PSO foi uma simulação do comportamento social, que era usada para visualizar o movimento de um revoada de pássaros. Diversas versões do modelo de simulação foram desenvolvidas, incorporando conceitos como vizinho mais próximo, velocidade e aceleração por distância, até que foi percebido a possibilidade de usar a simulação como otimizador. Diversos parâmetros foram omitidos e surgiu a primeira versão do PSO (EBERHART et al., 1995).

O algoritmo é inicializado com uma população de candidatos a solução aleatória. Cada indivíduo, denominado aqui de partícula, é ligado a uma velocidade com termos aleatórios $\{0,1\}$, onde $\{a,b\}$ indica escolha aleatória dentro do intervalo (a,b). A velocidade é definida pela Equação (2.15) (TRELEA, 2003):

$$v_j^k = wv_j^{k-1} + \{0,1\}c_1(\check{x} - x_j^{k-1}) + \{0,1\}c_2(\hat{x}^k - x_j^{k-1}), \tag{2.15}$$

onde w, c_1 e c_2 são parâmetros constantes e configuráveis utilizados para balancear o método. E a cada iteração desloca-se sobre o espaço do problema, conforme Equação (2.16).

$$x_j^k = \hat{x}^{k-1} + v_j^k, \tag{2.16}$$

onde $\check{x}$ é a posição do melhor valor obtido para aquela partícula específica e $\hat{x}^k$ é o melhor valor obtido ao longo de toda a execução (TRELEA, 2003). O Algoritmo 1 apresenta o funcionamento do método PSO.

Algoritmo 1 *Particle Swarm Optimization*

Fonte: Autor, baseado no trabalho de Trelea (2013).

1: Recebe os parâmetros: tamanho da população, dimensão do problema, intervalo de busca $[a, b]$, quantidade de iterações K, tolerância E e as constantes c_1, c_2 e w.
2: Criação aleatória dos primeiros candidatos a solução ótima e velocidade
3: Obtém $\hat{x}^k$ dos primeiros candidatos a solução
4: Para a primeira iteração, $\check{x}$ é igual aos primeiros candidatos a solução
5: **para** k=1 até K **faça**
6: **para** j=1 até população // analisar todos os indivíduos da população **faça**
7: $v_j^k = wv_j^{k-1} + \{0,1\}c_1(\check{x} - x_j^{k-1}) + \{0,1\}c_2(\hat{x}^k - x_j^{k-1})$
8: $x_j^k = \hat{x}_j^{k-1} + v_j^k$
9: **se** $f(x_j^k) < f(x_j^{k-1})$ // Testa-se o candidato atual é melhor **então**
10: $x_j^{k-1} = x_j^k$
11: **se** $f(x_j^k) < f(\check{x}_j)$// Testa-se é o melhor resultado geral **então**
12: $\check{x}_j = x_j^k$
13: **fim se**
14: **fim se**
15: **fim para**
16: Escolha do $\hat{x}^k$ pela posição de menor valor da função objetivo
17: **se** $[f(x_j^k) - f(x_j^{k-1})] \leq E$ **então**
18: **fim para** // Interrompe a execução do PSO
19: **fim se**
20: **fim para**

É importante notar a influência que a configuração dos parâmetros representa à performance do método, definida por Trelea (2003) como uma escolha entre testar várias regiões com o objetivo de encontrar um bom ótimo ou concentrar a busca na região de um candidato promissor. Essas escolhas alteram a capacidade de encontrar o ótimo global, mas apesar de estudos recentes, permanecem empíricas (TRELEA, 2003) e baseadas na experiência e prática (IZMAILOV et al., 2007).

Capítulo 3

Machine Learning

Nesse capítulo são introduzidos os principais termos referentes a ML, assim como a matemática por trás do seu funcionamento.

3.1 Introdução

Hoje em dia, ML está em todos os lugares: nos aplicativos de *streaming* de música e vídeo, nas ferramentas de buscas, em aplicativos de tradução, reconhecimento de imagem, veículos autônomos e uma infinidade de campos. Mas ML não é algo novo, estando presente em algumas aplicações especializadas a décadas, sendo a primeira aplicação de uso generalizado lançada na década de 1990, o filtro de spam (GÉRON, 2019).

Como disciplina, ML são modelos criados com o propósito de desenvolver, entender e aplicar programas de computadores capazes de reter conhecimento a partir da experiência, para modelagem, previsão e controle, sendo previsão umas das capacidades mais importante. Previsão também é a capacidade desejada nesse trabalho.

Existem muitos tipos de previsão. A capacidade de previsão pode ser utilizada para entender eventos futuros, como no mercado financeiro, a previsão do tempo ou a provável próxima palavra em teclados de telefones. É possível prever até informações desconhecidas, como se um usuário gostaria ou não de determinado filme ou música, informações dentro de uma imagem etc.

Arthur Samuel definiu ML em 1959 como: "*Machine Learning* é o campo de estudo que dá aos computadores a habilidade de aprender, sem serem explicitamente programados" (GÉRON, 2019, pág. 22).

Destaca-se a habilidade de aprender sem a necessidade de programação direta. Muitos problemas são difíceis e uma programação tradicional resultaria em uma longa lista de regras complexas, complicadas de manter (GÉRON, 2019).

Outros usos para ML são para problemas complexos, de difícil avaliação por métodos tradicionais ou que não tenham equações ou algoritmos definidos. Problemas com cenários fluidos em que novos dados são constantemente acrescentados ou até mesmo, para obter informações sobre problemas complexos com grande quantidade de dados (GÉRON, 2019).

A Figura 3.1 ilustra uma série simples de processos sequenciais para aplicação de ML.

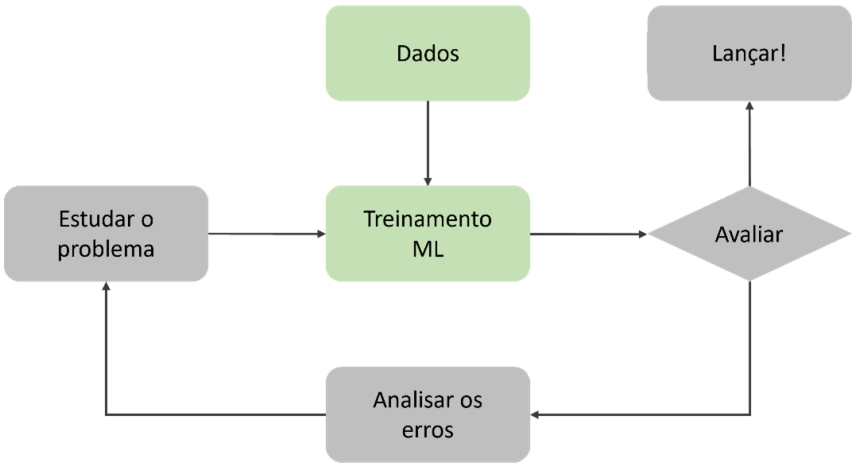

Figura 3.1: Simples processo de utilização de ML.
Fonte: Traduzido e adaptado de Géron, 2019.

A Figura 3.1 ilustra um fluxograma do processo de criação de uma ML. Tudo começa pela obtenção dos dados, que passam pela importante etapa de treinamento. A etapa imediatamente posterior, pode ser dividida em duas outras etapas: verificação dos resultados por meio de comparações com cálculos ou validação.

Uma validação utilizada em ML é a divisão do conjunto de treinamento em dois outros conjuntos menores. Um desses conjuntos participa do treinamento enquanto o outro é separado para validação. Após o treinamento, a rede treinada é executada com os dados de treinamento e os valores obtidos são comparados com os valores reais para esses dados. Esse tipo de validação requer cuidado pois o conjunto de treinamento pode não ser uma boa representação dos dados gerais. Esse trabalho obedeceu a esse processo de validação.

Caso um modelo ainda precise de refinamento, uma análise dos erros encontrados e um

estudo do problema (e resultados) ajudam a redefinir um novo treinamento, que passará por todo o processo novamente. Nesse estudo em particular, o uso de ML é motivado pela dificuldade em programação de métodos tradicionais, que sejam eficientes para o constante fluxo de novos dados.

3.2 Regressão linear (RL)

Aprendizado supervisionado é a forma mais básica e simples do ML. Essa forma de retenção de conhecimento tem como objetivo fornecer uma saída, *output* para cada conjunto de entrada, *input*, baseados em dados fornecidos aos pares para treinamento. O que o algoritmo faz é inferir uma função sobre dados de treinamento rotulados, que constituem o conjunto de treinamento (*training set*).

O conjunto de treinamento, aqui denotado por S_n, é composto por n pares de dados: vetor de características $x^{(i)}$ e rótulo $y^{(i)}$. O $x^{(i)}$ tenta encapsular as características dos dados que é fornecido para o algoritmo. Cada elemento do vetor contém informações sobre determinado objeto de interesse. Um vetor de características típico pode se parecer como (3.1) (BISHOP, 2006).

$$x = [x_1, x_2, ..., x_d] \ \in \mathbb{R}^d. \tag{3.1}$$

As características utilizadas para compor o vetor são escolhidas a partir dos dados disponíveis, da importância da característica e pela complexidade do problema. O número de características abordadas em cada vetor características será denotado por d, assim sendo, $x^{(i)} \in \mathbb{R}^d$.

O rótulo $y^{(i)}$ é a classificação do vetor de características $x^{(i)}$. Para cada $x^{(i)}$ no conjunto de treinamento deverá ser fornecido um rótulo $y^{(i)}$ correspondente. Ao contrário do vetor de características, que são uma descrição do próprio dado, os rótulos são uma classificação desses dados, podendo ser subjetivo. Na RL o rótulo irá assumir valores reais, como em (3.2) (MURPHY, 2012).

$$y^i \in \mathbb{R}. \tag{3.2}$$

Em Camp et al. (2018) e Camp et al. (2019), por exemplo, os autores utilizaram um vetor características contendo dados de Eritrograma, uma posição para cada valor e rótulo binário, para classificar presença ou não de anemias e características particulares

para tipificação.

Assim sendo, o conjunto de treinamento pode ser denotado pela Equação (3.3) (MURPHY, 2012).

$$S_n = \{(x^{(i)}, y^{(i)}), i = 1, 2, 3, ..., n\}, \tag{3.3}$$

onde $x^i \in \mathbb{R}$ e $y^i \in \mathbb{R}$.

Será utilizada a função $f(x, \theta, \theta_0)$ para estimar y, onde o conjunto de parâmetro é definido em (3.4) (MURPHY, 2012).

$$\theta = [\theta_1, \theta_2, \cdot, \theta_d]^i \in \mathbb{R}^d. \tag{3.4}$$

E o termo interceptador é definido em (3.5) (MURPHY, 2012).

$$\theta_0 \in \mathbb{R}. \tag{3.5}$$

A função $f(x, \theta, \theta_0)$ irá receber um vetor de d dimensões e gerará um valor real $\mathbb{R}$, que representa a estimativa de y.

Para RL, a função de regressão será da forma (3.6) (MURPHY, 2012):

$$f(x, \theta, \theta_0) = \sum_{i=1}^{d} \theta_i x_i + \theta_0. \tag{3.6}$$

Em notação vetorial é da forma definida em (3.7) (MURPHY, 2012):

$$f(x, \theta, \theta_0) = \theta \cdot x + \theta_0; \tag{3.7}$$

Isso significa que será usado um hiperplano para estimar os valores dos rótulos $y^{(i)}$ para um dado $x^{(i)}$. Mas apesar de $y^{(i)}$ depender linearmente dos componentes do vetor de características, o poder da RL reside nas diferentes transformações que $x^{(i)}$ pode sofrer para formar uma forte relação linear entre entrada e saída. O rótulo em função do vetor características transformado é definido em (3.8) (BISHOP, 2006).

$$y = \theta_1 x_{novo} + \theta_0, \tag{3.8}$$

onde $x^i_{novo} = f(x^i)$.

O processo de regressão linear é dividido em 3 etapas:

1) Objetivo – Gerar a função objetivo a ser minimizada.

2) Algoritmo de aprendizado – Ferramentas que possibilitaram obter o hiperplano a partir dos dados fornecidos.

3) Regularização – Solucionar o *trade-off* entre super ajustar (*overfitting*) e sob ajustar (*underfitting*)

A primeira etapa do processo, Objetivo, é determinada pela introdução da função objetiva. Isso será feito com a função *Loss function*: $Loss(z_i)$. Muitas formas de perdas podem ser utilizadas, mas no caso da RL a mais utilizada é a perda quadrada, definida em (3.9). Por efeito de simplificação, θ_0 será omitido, sem prejuízo de generalidade (MURPHY, 2012).

$$\text{Loss}z^{(i)} = \frac{z^{(i)_2}}{2}, \quad z^{(i)} = y^{(i)} - \theta.x^{(i)}, \tag{3.9}$$

onde, y é o rótulo correto, e $\theta \cdot x$ é o estimado. A Figura 3.2 ilustra geometricamente esse erro.

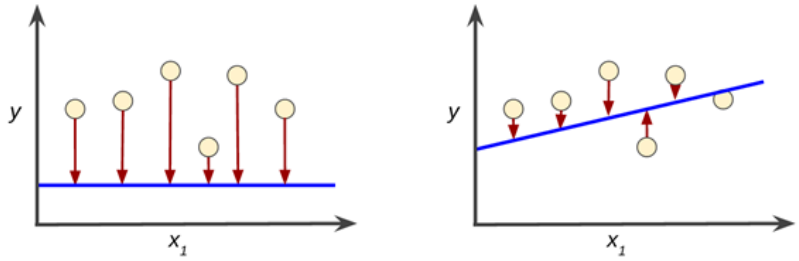

Figura 3.2: Exemplo de perda para vetores característica.
Fonte: Google Developers, 2020.

No Figura 3.2, em ambos os exemplos de gráfico, os círculos representam os valores para $y^{(i)}$, a linha em azul é o hiperplano $\theta \cdot x = 0$ e as linhas verticais são os valores de $z^{(i)}$, para diferentes pontos do conjunto de dados.

A perda quadrada penaliza de forma mais severa pontos distantes do hiperplano e de forma mais suave pontos mais próximos. Dessa forma, a função objetivo pode ser definida como uma média de todas as perdas em (3.10) (BISHOP, 2006).

$$R_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} \frac{(y^{(i)} - \theta \cdot x^{(i)})^2}{2} \tag{3.10}$$

Essa função é chamada *Empirical Risk* e o objetivo é minimizar essa função.

Chama-se a atenção para dois erros comuns.

1) Erros de estrutura – Ocorre quando Regressão Linear não é suficiente para resolver o problema. Nesse caso, a solução se dará por utilizar outras funções mais complexas, como não-lineares.

2) Erros de estimativa – Ocorre quando não há dados suficientes para fazer uma boa estimativa.

Esses dois erros são interligados. Um problema com suficiente dados por exemplo, pode precisar de uma ordem maior para solucionar erros de estrutura e por consequência, requerer mais dados, acarretando erros de estimativa. Existirá, portanto, um *trade-off* que deverá ser balanceado de acordo com cada modelo.

Para melhorar a generalização do modelo, introduz-se o termo regulador $\frac{1}{2}\|\theta\|^2$. Sua função é contrapor o *empirical risk* e auxiliar para que o modelo definido na Equação (3.11) (BISHOP, 2006) tenha bons resultados para dados novos.

$$J(\theta, \lambda) = R_n(\theta) + \frac{\lambda}{2}\|\theta\|^2, \tag{3.11}$$

onde λ é chamado de hiper parâmetro e auxilia no balanceamento da função objetiva (3.11), priorizando um termo em relação ao outro na otimização.

Para minimizar a função objetivo (3.11) será utilizado o algoritmo Gradiente Descendente Estocástico, ou do inglês *Stochastic Gradient Descendent (SGD)*.

3.2.1 Gradiente Descendente

Seja uma função convexa $J(\theta)$ tal que $\theta \in \mathbb{R}$, a derivada dessa função no ponto $\theta = \theta_0$ é a inclinação da linha tangente a esse ponto, formulada em (3.12). A magnitude da derivada é definida pela inclinação da curva. Quanto maior a inclinação da curva, maior a derivada e vice-versa (GUIDORIZZI, 2014).

$$tan\alpha = \frac{d}{d\theta}J(\theta). \tag{3.12}$$

A Figura 3.3 ilustra a derivada de $J(\theta)$ para várias posições.

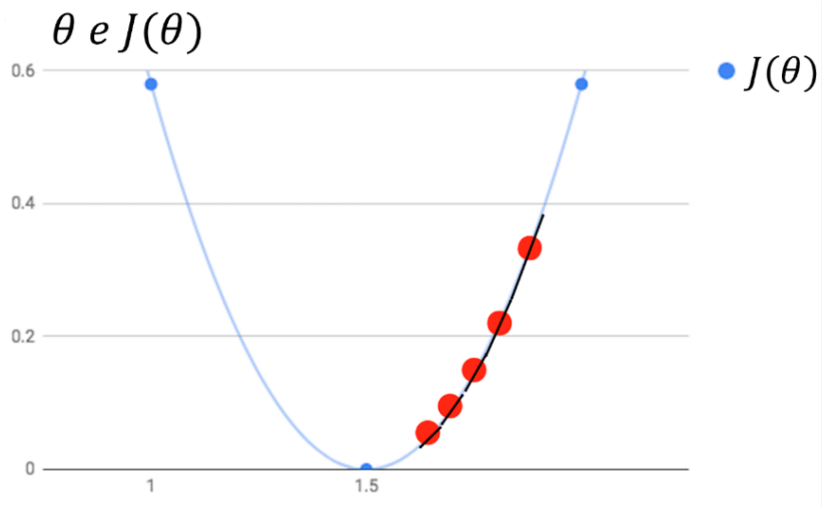

Figura 3.3: Gradiente de $J(\theta)$
Fonte: Adaptado de Rushton, 2018.

Observa-se pela Figura 3.3 que à medida que se aproxima do vale, a intensidade da inclinação da reta tangente e da derivada são reduzidas. No ponto mínimo, a derivada (3.12) é nula.

Considere agora um vetor unitário $\hat{\theta}$ na direção das abcissas. A derivada em $\hat{\theta}$ pode ser escrita como (3.13) (GUIDORIZZI, 2014).

$$-\frac{d}{d\theta}J(\theta)\hat{\theta}. \tag{3.13}$$

O vetor terá a magnitude da derivada da função objetiva e a direção de $\hat{\theta}$ quando a derivada for negativa e a direção contrária quando a derivada for positiva, sempre apontando para a posição em que a derivada é zero, ou seja, para o ponto de mínimo.

Supondo agora um algoritmo para encontrar esse ponto de mínimo, a atualização é dada por (3.14) (BISHOP, 2006):

$$\theta^{k+1} = \theta^k - \eta\frac{d}{d\theta}J(\theta), \tag{3.14}$$

onde o termo η é chamado de taxa de aprendizado.

De forma mais genérica, pode-se escrever (3.14) como (3.15).

$$\vec{\theta}^{k+1} = \vec{\theta}^k - \eta\vec{\nabla}J(\vec{\theta}). \tag{3.15}$$

Sendo $\vec{\nabla}J(\vec{\theta})$ definido em (3.16).

$$\vec{\nabla} J(\vec{\theta}) = \begin{bmatrix} \frac{\partial}{\partial \theta_1} J(\vec{\theta}) \\ \frac{\partial}{\partial \theta_2} J(\vec{\theta}) \\ \vdots \\ \frac{\partial}{\partial \theta_d} J(\vec{\theta}) \end{bmatrix}. \tag{3.16}$$

Voltando a (3.15), a atualização a cada iteração por Gradiente Descendente pode ser computacionalmente muito intensa para grandes bancos de dados, da ordem de $n \times d$ cálculos do gradiente, que é uma operação computacionalmente custosa. Nesses casos utiliza-se SGD.

3.2.2 Gradiente Descendente Estocástico

Essa técnica é muito utilizada para resolver problemas de ML de grande escala. O objetivo do método é resolver problemas do tipo formulado em (3.17) (MURPHY, 2012).

$$min \frac{1}{n} \sum_{i=1}^{n} f_i(\theta). \tag{3.17}$$

O método do SGD escolhe aleatoriamente um ponto a cada iteração para calcular o gradiente.

Retornando a (3.16), a posição a ser calculada a derivada é escolhida aleatoriamente como em (3.18).

$$i_k = \{1, 2, 3, ..., n\}, \tag{3.18}$$

onde {} representa escolha aleatória entre intervalo indicado, segundo notação do Autor (2021).

Uma formulação apresentada para taxa de aprendizado é definida por (3.19) (MURPHY, 2012).

$$\eta_k = \frac{1}{1+k}. \tag{3.19}$$

O método SGD é extremamente volátil quando se aproxima do ponto mínimo e por esse motivo, as taxas de aprendizado devem decrescer com o avanço das iterações. Nesse trabalho, foi utilizada uma outra taxa de aprendizado, definida em (5.5). O otimizador

utilizado para cálculo do SGD foi o Adam (KINGMMA, 2015).

3.2.3 Bias-Variância *Trade-off*

Existem formas de avaliar se um estimador é bom ou ruim. Segundo Briscoe (2011), *trade-off* Bias-Variância é referente a qualidade em que a independência de associações é feita dos dados de treinamento.

O objetivo de ML é justamente encontrar um modelo que tenha um bom desempenho de previsão em novos dados, não utilizados no treinamento (BELKIN, 2019). Para isso, é necessário um máximo de generalização dos dados de treinamento, aumentando a precisão de futuras classificações, mas contrário a intuição, isso não ocorre maximizando o aprendizado dos dados de treinamento (BRISCOE, 2011).

Um modelo que aprende de forma muito detalhada os dados de treinamento, de forma inevitável, irá incluir muitos padrões particulares do conjunto, que não se repetirão em outros conjuntos de dados. Por consequência, o modelo terá uma performance ruim com novos dados. Esse evento chama-se *overfitting*. De forma extremamente oposta, um treinamento com altos erros em relação aos dados de treinamento, irá acarretar o não aprendizado de tendências reais e desejáveis e performará mal. Esse evento é definido como *underfitting*.

Uma prática convencionada em ML é controlar a capacidade da função baseada no *trade-off* Bias-Variância (BELKIN, 2019). Se o modelo é muito complexo, e nesse caso, complexidade está relacionada a capacidade de aprendizado, minimizar o *empirical risk* pode resultar em má performance de novos dados, enquanto um modelo muito simples pode acarretar *underfitting*. A Figura 3.4 ilustra a capacidade de generalização ao longo de crescente complexidade de um modelo genérico.

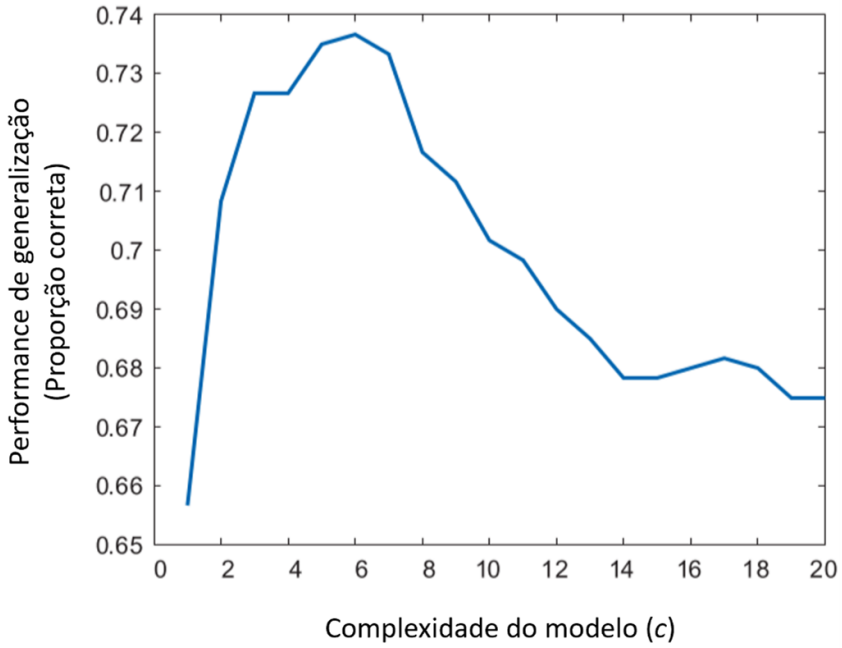

Figura 3.4: Gráfico Generalização vs. Complexidade.
Fonte: Adaptada de Briscoe, 2011.

A complexidade dada pelo inverso do hiperparâmetro: $1/\lambda$. Como λ, complexidade é uma forma de ajustar o *Empirical Risk*. Como ilustrado na Figura 3.4, a qualidade de generalização cresce com a complexidade do modelo até um certo nível, a partir do qual, aumentar a complexidade não resulta em ganho de qualidade, mas em manutenção, ou ainda perda de qualidade.

3.3 Redes Neurais Artificiais

Uma das desvantagens dos modelos apresentados até então é a otimização não convexa durante o treinamento. Uma alternativa é utilizar formas paramétricas para a função base, nas quais os parâmetros são adaptados durante o treinamento. O modelo mais bem sucedido no contexto de reconhecimento de padrão é a RNA (Redes Neurais Artificiais) *feedforward*. Para muitas aplicações, o modelo resultante pode ser muito compacto e mais rápido de avaliar que os métodos anteriores, para um mesmo nível de generalização (BISHOP, 2006).

O termo *neural network* tem sua origem na tentativa de encontrar representação matemática para processamento de informações de sistema biológicos (BISHOP, 2006). Inspirados pelo modelo neural do cérebro humano, engenheiros criaram a ideia de um

cérebro artificial para máquinas, RNA. As RNA são divididas nas seguintes partes:

- Nódulo: Representado por uma circunferência, nódulos (node) recebem um ou mais entradas, e emitem uma ou mais saídas. Para cada node existe uma função associada. Essa função atua nas entradas, gerando as saídas.
- Camada: Um conjunto de nódulos de mesmo nível definem uma camada (Layer).

Toda rede neural possui uma camada de entrada (*input layer*), uma camada de saída (*output layer*) e pelo menos uma camada oculta (*hidden layer*). Esse último tipo, são todas as camadas entre as camadas de entrada e de saída. A Figura 3.5 ilustra esquematicamente uma rede neural simples.

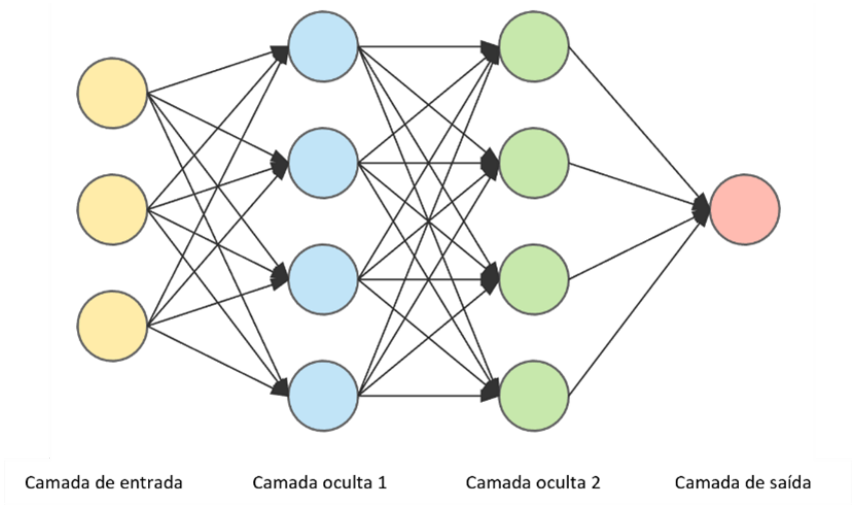

Figura 3.5: Exemplo: Estrutura de uma RNA.
Fonte: Adaptada de Dertat, 2017.

A Figura 3.5 ilustra a estrutura de uma RNA simples. Os nódulos amarelos compõem a camada de entrada, o módulo laranja compõe a camada de saída. Todas as camadas entre essas duas, com nódulos azuis e verdes, compõem as diferentes camadas ocultas. A camada de entrada tem o mesmo tamanho que o vetor característico.

A camada de entrada fornece dados para o nódulo de saída. Esses dados são representados pela letra z. O nódulo de saída irá atuar em z para fornecer um simples dado de saída $f(z)$, que normalmente é balanceado pela soma de diferentes componentes de entrada, também balanceados pelo correspondente peso, como definido em (3.20) (BISHOP, 2006).

$$z = w_0 + x \cdot w = w_0 + \sum_{i=1}^{d} x_i \cdot w_i, \tag{3.20}$$

sendo d, o número de nódulos da camada de entrada.

A Figura 3.6 ilustra a estrutura esquemática do balanceamento dos nódulos por peso.

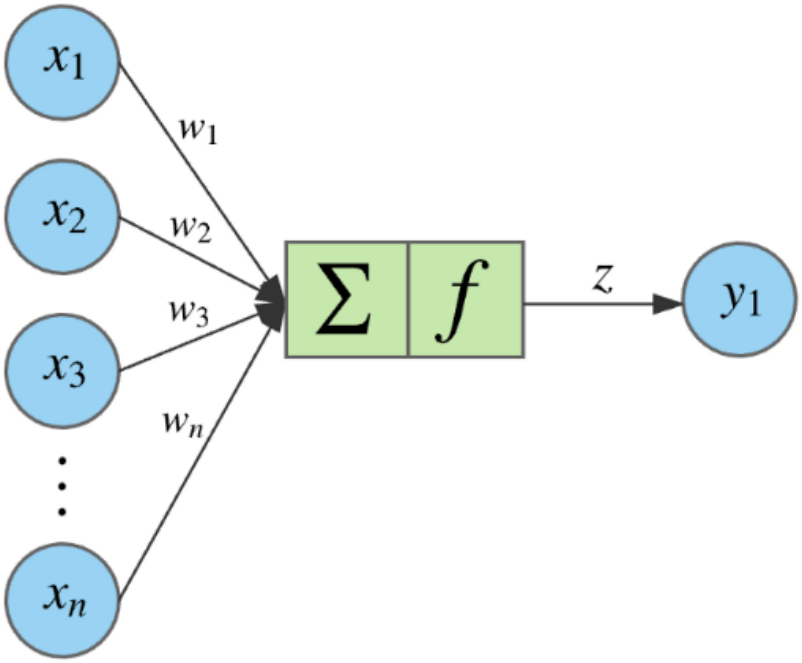

Figura 3.6: Figura 3.6 - Ilustração do peso em RNA.
Fonte: Dertat, 2017.

A Figura 3.6 ilustra um nódulo. A função de ativação é aplicada no somatório das entradas, somado ao bias.

3.3.1 Função de ativação

Cada nódulo tem uma função associada. Essa função recebe o nome de função de ativação. Algumas das mais utilizadas são a Linear, ReLU, Sigmoid e Tanh. Essas funções realizam uma transformação nos dados de entrada, com o objetivo melhor ajustar os dados. A Figura 3.7 ilustra as principais funções de ativação.

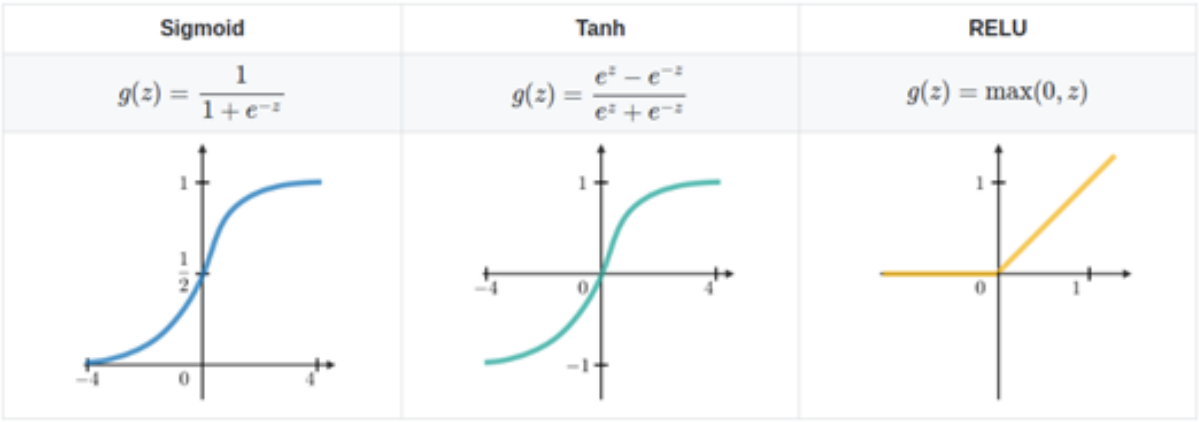

Figura 3.7: Principais funções de ativação
Fonte: Kanani, 2019.

Como pode ser visto na Figura 3.7, cada função de ativação tem um comportamento e imagem diferentes. A equação do hiperplano é definida em (3.21) (BISHOP, 2006).

$$z_j^l = x_1 w_{11} + \cdots + x_n w_{nj} + w_{0j}. \tag{3.21}$$

Onde z_j^l representa a entrada para o nódulo j, da camada l, para n entradas e w_{ij} é peso referente ao nódulo j para o nódulo i.

A Equação (3.21) é a equação do hiperplano, já apresentado, no espaço $x_1 - x_i$ com vetor normal $w = [w_{1j}, \cdots, w_{nj}]^T$ e termo interceptador w_{0j}. Em notação vetorial, tem-se (3.22) (BISHOP, 2006).

$$z_j^{il} = x^i \cdot w + w_{0j}. \tag{3.22}$$

A distância do i-ésimo ponto, na camada l, nódulo j da linha k é definida por (3.23) (BISHOP, 2006).

$$\frac{|z_j^{il}|}{\|w\|}. \tag{3.23}$$

E função de ativação (3.24) (BISHOP, 2006).

$$f_j^l = f(z_j^l). \tag{3.24}$$

Em alguns casos, a aplicaçãp da transformação pela função de ativação é a solução para erros de estimativa (item 3.2). Para os resultados expressos no capítulo 6, a função

de ativação utilizada foi a *sigmoid*.

3.3.2 *Backpropagation*

Seja o bias e o resultado da função de ativação, respectivamente, do j-ésimo nódulo, da l-ésima camada, definida pela notação do autor (3.25).

$$b_j^l, a_j^l \tag{3.25}$$

Tem-se (3.26) (BISHOP, 2006).

$$a_j^l = f\left(\sum_k w_{jk}^l a_k^{l-1} + b_j^l\right), \tag{3.26}$$

onde a é uma saída real, $f()$ uma função de ativação e k é o número de nódulos da camada anterior.

Segundo Bishop (2006), o bias pode ser absorvido pelo conjunto de parâmetros peso ao adicionar uma entrada x_0, cujo valor é fixado em $x_0 = 1$. A Equação (3.26) fica então na forma (3.27) (BISHOP, 2006).

$$a_j^l = f\left(\sum_{i=0}^{k} w_{jk}^l a_k^{l-1}\right). \tag{3.27}$$

É desejável minimizar o erro ao quadrado, definido em (3.28) (BISHOP, 2006).

$$\frac{1}{2n}\sum_{i=1}^{n}(y^i - f_L^i)^2, \tag{3.28}$$

onde n é o número de vetores características em treinamento, y^i é o valor verdadeiro e f_L^i é o valor previsto. Aplicando novamente SGD para minimizar a função (3.28), a atualização da posição torna-se (3.29) (BISHOP, 2006).

$$\vec{w}_{k+1} = \vec{w}_k - \eta_k \left[\vec{\nabla}\frac{(y - f_L(x, \vec{w}))^2}{2}\right]. \tag{3.29}$$

A saída da rede neural, na camada de saída, é a_j^L, sendo $j = 1$ na maioria dos casos e a derivada de interesse é (3.30) (BISHOP, 2006):

$$\frac{\partial a_j^L}{\partial w_{jk}^l}, \frac{\partial a_j^L}{\partial b_j^l}. \tag{3.30}$$

Enquanto a_j^L é uma função de valor real, w e b são valores a serem aprendidos.

Para obter as derivadas parciais de (3.30), faz-se uso da regra da cadeia, sendo sempre necessário avaliar as saídas da camada oculta anterior, até que todas tenham sido avaliadas e por isso o nome *Backpropagation*.

Capítulo 4

Manutenção

Nesse capítulo são apresentados os principais conceitos e cálculos matemáticos usados para definição de indicadores importantes para a manutenção industrial, assim como definição e cálculos base para a criação de programas de manutenção.

4.1 Introdução

Com o advento da economia globalizada, observou-se um aumento da demanda por produtos e sistemas de melhor desempenho a custos competitivos (FOGLIATTO et al., 2011). Devido a isso, a indústria passou a olhar mais atentamente às falhas dos equipamentos e os custos provenientes destas, o que resultou em uma crescente ênfase em confiabilidade.

Segundo Lewis (1994), a "confiabilidade de um item corresponde à sua probabilidade de desempenhar adequadamente o seu propósito especificado, por um determinado período de tempo e sob condições ambientais predeterminadas."

Em sua definição, Lewis (1994) chama a atenção para quatro termos importantes: item, desempenho, período de tempo e condições predeterminadas. Sua distinção é importante pois o estudo da confiabilidade está atrelado a um item, que pode ser desde um componente de um equipamento, ou um sistema, passando por uma máquina ou até mesmo toda uma linha de montagem.

É importante também a definição de desempenho. O modelo matemático mais simples usado para representar a condição de um item é o binário: ou o item está em estado de funcionamento ou falha (FOGLIATTO et al., 2011).

O período de tempo merece atenção, pois confiabilidade é definida em função deste, o qual também pode ser definido em unidades não temporais, como ciclos (FOGLIATTO et al., 2011).

E por último, condições ambientais predeterminadas devem ser levadas em consideração, devido a variação de desempenho e vida útil que ambientes de diferentes umidades, temperaturas e demandas influenciam.

Em razão disto, em um cenário industrial em que *just-in-time* é amplamente usado, a confiabilidade dos equipamentos produtivos tem sua importância ainda elevada, tornando um elemento essencial do cronograma de manutenção, devido a importante conexão existente entre manutenção e confiabilidade (FOGLIATTO et al., 2011).

4.2 Funções Distribuição de Probabilidade

Grande parte dos cálculos referentes a parâmetros utilizados na Manutenção Industrial envolvem estimativas de eventos possíveis de acontecer. Nesse cenário, para embasar as análises de confiabilidade, objeto central desse trabalho, é necessário considerar como as probabilidades de falha dependem de uma variedade de outras variáveis, que são contínuas. Tais variáveis são assim chamadas, contínuas aleatórias, porque não podem ser descritas com exatidão, mas apenas com probabilidade de que assumirão certo valor, com alguma margem. Por esse motivo, uma revisão da estatística envolvida se mostra importante.

Seja $\mathbf{x}$ uma variável aleatória contínua e x os valores que $\mathbf{x}$ pode assumir. As propriedades de variáveis aleatórias são especificadas em termos de probabilidades. Dois tipos particulares de probabilidades são mais frequentemente utilizados para descrever variáveis aleatórias. Esses são definidos pelas Equações (4.1) e (4.2) (LEWIS, 1994).

$$F(x) = P\{\mathbf{x} \leq x\}. \tag{4.1}$$

A Equação (4.1) é a probabilidade de que $\mathbf{x}$ tenha um valor menor ou igual a x, também conhecido como função de distribuição acumulativa, ou do inglês *cumulative distribution function (CDF).*

$$f(x)\Delta x = P\{x \leq \mathbf{x} \leq x + \Delta x\}. \tag{4.2}$$

A Equação (4.2) representa a probabilidade de $\mathbf{x}$ assumir valores entre x, $x + \Delta x$, quando Δx tende a ser muito pequeno. A função $f(x)$ é chamada de densidade de probabilidade,

ou do inglês, *probability density function (PDF)*.

Essas duas funções são relacionadas. Se $\mathbf{x}$ pode assumir qualquer valor no intervalo $-\infty \leq \mathbf{x} \leq +\infty$, o CDF é apenas a integral do PDF sobre todo $\mathbf{x} \leq x$, como posto em (4.3) (LEWIS, 1994).

$$F(x) = \int_{-\infty}^{x} f(x')dx'. \tag{4.3}$$

E alternativamente (4.4).

$$f(x) = \frac{d}{dx}F(x). \tag{4.4}$$

Como $\mathbf{x}$ precisa ser um valor entre $(-\infty, \infty)$, tem-se (4.5) (LEWIS, 1994).

$$P\{-\infty \leq \mathbf{x} \leq \infty\} = 1. \tag{4.5}$$

E a condição de normalização (4.6) (LEWIS, 1994).

$$\int_{-\infty}^{\infty} f(x)dx = 1. \tag{4.6}$$

Então, para $\mathbf{x} = \infty$ em (4.3), encontra-se a condição (4.7) para CDF (LEWIS, 1994).

$$F(\infty) = 1. \tag{4.7}$$

A Figura 4.1 contém gráficos de PDF e CDF.

Uma outra função que é frequentemente utilizada é a função distribuição acumulativa complementar, ou do inglês, *complementary cumulative distribution function (CCDF)*. Segundo Lewis (1994), CCDF é definida por (4.8).

$$\check{F}(x) = P\{\mathbf{x} > x\} \tag{4.8}$$

Pela definição de $f(x)$ e a Equação (4.6), obtém-se (4.9) (LEWIS, 1994).

$$\check{F}(x) = \int_{x}^{\infty} f(x')dx' = 1 - \int_{-\infty}^{x} f(x')dx' \tag{4.9}$$

Combinando com (4.3), chega-se a (4.10) (LEWIS, 1994).

$$\check{F}(x) = 1 - F(x) \tag{4.10}$$

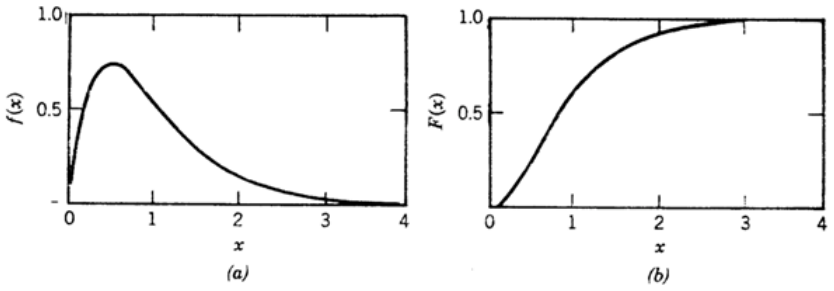

Figura 4.1: Relação entre PDF e CDF para variável aleatória genérica.
Fonte: Lewis, 1994.

O gráfico a) da Figura 4.1 ilustra uma curva PDF e o gráfico b) da Figura 4.1 ilustra uma curva CFD.

4.3 Qualidade e confiabilidade

Qualidade e confiabilidade são conceitos frequentemente confundidos, mas existe um importante diferença entre eles. Enquanto confiabilidade incorpora unidades de tempo, qualidade é uma descrição estática (FOGLIATTO et al., 2011).

A qualidade está associada à capacidade de projetar produtos que incorporem características e atributos otimizados para atender a necessidade do usuário e à redução da variabilidade nas características de desempenho.

4.3.1 Principais conceitos associados a confiabilidade

Os principais conceitos associados à confiabilidade são: qualidade, disponibilidade, manutenibilidade, segurança e confiança. A seguir estes são definidos com base na norma NBR ISO-8402 (1994) e 5462(1994).

Qualidade é o cumprimento de especificações de projeto e manufatura com menor variabilidade possível.

Disponibilidade é a capacidade de um item desempenhar seu propósito de forma adequada em um determinado instante de tempo ou período predeterminado. Em equipamentos não reparáveis, disponibilidade é equivalente a confiabilidade. Nos equipamentos reparáveis, os possíveis estados são funcionando ou em manutenção. A disponibilidade, tal qual definida por Lewis (1994) está definida na Equação (4.11).

$$A = \frac{MTTF}{MTTF + MTTR}, \tag{4.11}$$

onde A (*availability*) denota disponibilidade média, MTTF (*Mean time to failure*) é o tempo médio para falha e MTTR (*Mean time to repair*) é o tempo médio para reparo (LEWIS, 1994). Destaca-se que nesse trabalho, uma equação adaptada pelos autores foi utilizada. Esta adaptação está definida na Equação (5.14).

Manutenibilidade é a capacidade de um item ser mantido ou recolocado em condições de executar seu propósito adequadamente, quando submetidos a manutenção com recursos e procedimentos padrões. Os sistemas objeto desse estudo possuem essa propriedade.

Segurança é definida com a ausência de condições que possam causar morte, dano ou doença a pessoas, bem como perda ou dano de equipamentos ou propriedade.

Confiança é utilizado para designar um coletivo que inclui a disponibilidade, seus fatores determinantes. Os termos confiança e confiabilidade podem ser considerados como análogos, mas esse primeiro é bem mais amplo, não estritamente associado a probabilidades.

4.3.2 Medida de confiabilidade

Nessa seção são apresentadas as diversas medidas de confiabilidade. A notação usada é comum a literatura especializada. Nota-se que a partir de qualquer uma das medidas apresentadas, as demais podem ser derivadas.

4.3.2.1 Tempo até falha

O tempo decorrido desde o momento em que a unidade é colocada em operação até sua primeira falha é definido como tempo até falha, ou do inglês, *time to failure (TTF)*. Convenciona-se o início da operação como $t = 0$. Por estar sujeito a variações aleatórias, o TTF é definido como uma variável aleatória T.

Como dito na seção 4.1, o sistema matemático mais simples para descrição do funcionamento de um item é o binário. A partir destes, convenciona-se o estado de uma máquina de acordo com a Equação (4.12) (FOGLIATTO, 2011).

$$X(t) = \begin{cases} 0, \text{se não operante em } t \\ 1, \text{se operante em t} \end{cases} \tag{4.12}$$

A Figura 4.2 ilustra graficamente a imagem de (4.12).

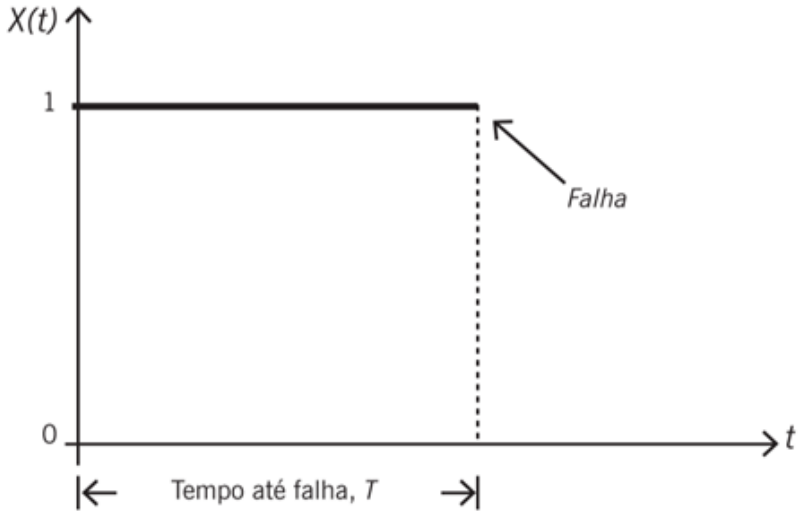

Figura 4.2: Função de estado X(t).
Fonte: Fogliatto, 2011.

A Figura 4.2 ilustra o status binário de um item. Uma função constante e descontínua.

A função de distribuição de T é dada pela Equação (4.13) (FOGLIATTO, 2011).

$$F(t) = P(T \leq t) = \int_0^t f(u)du, t > 0, \tag{4.13}$$

onde $F(t)$ denota a probabilidade de falha em um tempo menor ou igual a t. Devido a isso, é comunalmente referido como probabilidade de falha. A função $f(t)$ é a densidade de probabilidade de T.

Para valores muito pequenos de Δt (LEWIS, 1994), tem-se (4.14).

$$P(t \leq T \leq t + \Delta t) \approx f(t)\Delta t. \tag{4.14}$$

Como $F(t)$ e $f(t)$ são probabilidades, ambas devem ser maiores ou iguais a zero, para qualquer valor de t.

4.3.2.2 Função de confiabilidade

A função da confiabilidade é informar a probabilidade da unidade apresentar sucesso na operação no intervalo de tempo $(0, t)$ e ainda estar funcionando no tempo t. Essa função é definida como probabilidade acumulada de sucesso, definida pela Equação (4.15)

(FOGLIATTO, 2011).

$$R(t) = \frac{n_s(t)}{n_s(t) + n_f(t)}. \tag{4.15}$$

Sendo n_0 o número total de unidades em teste, $n_s(t)$ o número total de unidades sobreviventes no tempo t e $n_f(t)$ o número total de unidades que falharam no tempo t, tal que $n_0 = n_f(t) + n_s(t)$.

Considerando a variável aleatória T, a função de confiabilidade pode ser expressa pela Equação (4.16) (FOGLIATTO, 2011).

$$R(t) = P(T > t). \tag{4.16}$$

Desde que um sistema que não falhou para $T \leq t$, falhará em algum T. Logo, tem-se (4.17) (FOGLIATTO, 2011).

$$R(t) = 1 - F(t). \tag{4.17}$$

Como definido pela Equação (4.10), sendo, portanto, a confiabilidade CCDF da probabilidade de falha $F(t)$.

4.3.2.3 Função de risco

A função de risco, também conhecida como taxa de falha, pode ser interpretada como a quantidade de risco associada a uma unidade de tempo t. É a medida de confiabilidade mais difundida na prática (FOGLIATTO et al., 2011).

Seja a probabilidade de falha definida pela Equação (4.18) (FOGLIATTO, 2011).

$$P(t \leq T \leq t + \Delta t) = \int_t^{t+\Delta t} f(u)du = R(t) - R(t + \Delta t). \tag{4.18}$$

Condicionando que até o tempo t o item ainda não falhou, tem-se (4.19) (FOGLIATTO, 2011).

$$P(t \leq T \leq t + \Delta t | T \geq t) = \frac{P(t \leq T \leq t + \Delta t)}{P(T \geq t)} = \frac{R(t) - R(t + \Delta t)}{R(t)}. \tag{4.19}$$

A taxa de falha média é obtida dividindo-se (4.19) por Δt. Quando $\Delta t \to 0$, é obtido a taxa de falha instantânea, dada pela Equação (4.20) (FOGLIATTO, 2011).

$$h(t) = \lim_{\Delta t \to 0} \frac{R(t) - R(t + \Delta t)}{R(t)\Delta t} = \frac{f(t)}{R(t)},\ t \geq 0. \tag{4.20}$$

Sendo necessário atender as condições (FOGLIATTO, 2011):

(i) $\int_0^\infty h(t)dt = \infty$.

(ii) $h(t) \geq 0 | \forall t \geq 0$.

Integrando-se a taxa de falha sobre um período de tempo, obtém-se a função acumulada de risco, definido na Equação (4.21) (FOGLIATTO, 2011).

$$H(t) = \int_0^t h(u)du, t \geq 0. \tag{4.21}$$

A forma mais útil de expressar confiabilidade é em função da taxa de risco. Para tal, elimina-se $f(t)$ para obter a taxa de falha em função de confiabilidade, definido em (4.22) (FOGLIATTO, 2011).

$$h(t) = -\frac{1}{R(t)}\frac{d}{dt}R(t). \tag{4.22}$$

Multiplicando (4.22) por dt, tem-se (4.23).

$$h(t)dt = -\frac{dR(t)}{R(t)}. \tag{4.23}$$

Integrando no intervalo $(0, t)$, obtém-se (4.24).

$$\int_0^t h(t')dt' = -ln[R(t)]. \tag{4.24}$$

Como pela condição de definição, $R(0) = 1$, a confiabilidade pode ser escrita na forma (4.25) (FOGLIATTO, 2011).

$$R(t) = exp\left[-\int_0^t h(t')dt'\right]. \tag{4.25}$$

A forma da taxa de falha é um indicativo da maneira como uma unidade envelhece. Produtos manufaturados costumam apresentar uma taxa de falha dada pela ocorrência

sucessiva de três classificações: (i) taxa de falha decrescente, (ii) taxa de falha estacionária e (iii) taxa de falha crescente.

A taxa de falha pode ser formalmente dividida em seis modelos de risco: (i) constante, (ii) crescente, (iii) decrescente, (iv) curva da banheira *piecewise* linear, (v) função de potência e (vi) exponencial. O uso combinado de todos os modelos permite representar a quase totalidade dos mecanismos de riscos existentes na prática. A Figura 4.3 ilustra a curva da banheira (FOGLIATTO, 2011).

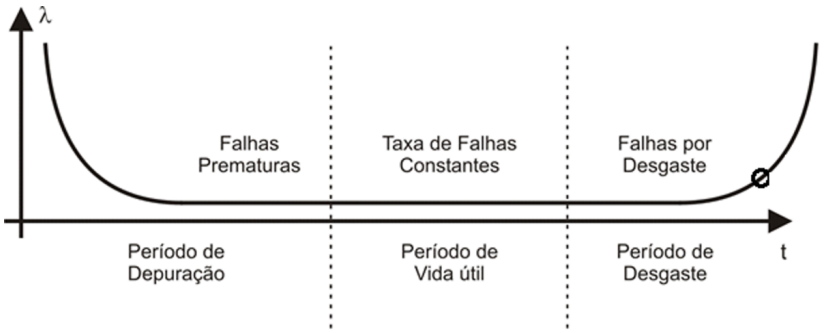

Figura 4.3: Curva da banheira.
Fonte: Sellito, 2005.

Figura 4.3 ilustra uma genérica curva da banheira, como é conhecido o gráfico da taxa de falha ao longo do tempo. Cada divisão do gráfico ilustra uma fase de falha do item. Falhas do tipo mortalidade infantil em geral ocorrem por deficiências no processo de manufatura. As falhas durante a vida útil do item ocorrem devido a condições extremas no ambiente de operação do produto. A deterioração do item frequentemente leva a desgastes concentrados no final da vida útil do produto, na fase de envelhecimento (FOGLIATTO, 2011).

A seleção de materiais mais duráveis na etapa de projeto, assim como práticas de manutenção adequadas e controle de fatores ambientais, são algumas das alternativas que podem ser utilizadas para amenizar a intensidade do envelhecimento (FOGLIATTO, 2011).

4.3.2.4 Tempo médio até falha

O tempo médio até falha é provavelmente o parâmetro mais usado para caracterizar confiabilidade (LEWIS, 1994), MTTF é definido pela Equação (4.26) (FOGLIATTO, 2011).

$$MTTF = E(T) = \int_0^\infty tf(t)dt. \tag{4.26}$$

Ou alternativamente, pode ser definido como (4.27) (FOGLIATTO, 2011).

$$MTTF = -\int_0^\infty t\frac{dR}{dt}dt = -[tR(t)]_0^\infty + \int_0^\infty R(t)dt. \tag{4.27}$$

Em $t = 0$, $tR(t)$ desaparece e para intervalos de tempo muito grandes (∞), $R(t) \to 0$. Logo, tem-se (4.28) (LEWIS, 1994).

$$MTTF = \int_0^\infty R(t)dt. \tag{4.28}$$

O MTTF então é encontrado com base na integral da função confiabilidade ao longo do tempo.

4.3.2.5 Função de vida Residual média

A função de vida residual média corresponde à vida remanescente esperada da unidade, dado que ela sobreviveu até o tempo t. É definido por (4.29) (FOGLIATTO, 2011).

$$L(t) = E[T - t|T \leq t], t \geq 0. \tag{4.29}$$

O cálculo da expectativa é dado por (4.30) (FOGLIATTO, 2011):

$$L(t) = \int_0^\infty u\frac{f(u)}{R(t)}du - t = \frac{1}{R(t)}\int_t^\infty uf(u)du - t. \tag{4.30}$$

Usualmente, a função de vida residual média é uma medida de confiabilidade de menor utilização prática (FOGLIATTO, 2011) e por isso não foi utilizada nesse trabalho.

4.4 Falha de componentes e modos de falha

É possível, desde que as falhas sejam independentes, generalizar e tratar a confiabilidade do sistema em termos das falhas de mecanismos e componentes (LEWIS, 1994).

4.4.1 Modos de taxas de falha

Quando se trata de falhas de componentes, é possível analisar a confiabilidade de um sistema em termos dos seus componentes, desde que suas respectivas falhas sejam independentes um dos outros. Isso quer dizer que, a falha de um componente não influencia a falha do outro (LEWIS, 1994).

Seja $h(t) = \lambda$ a função taxa de falhas constante. Para um sistema com M diferentes modos de falha, a confiabilidade é definida pela Equação (4.31) (LEWIS, 1994).

$$R(t) = P\{X_1 \cap X_2 \cap \cdots \cap X_M\}, \tag{4.31}$$

onde X_i é o evento em que a i-ésima falha não ocorra antes do tempo t.

De forma análoga, pode-se escrever a confiabilidade do sistema com produto de todos as probabilidades de sobrevivência, definido pela Equação (4.32) (LEWIS et al., 1994).

$$R(t) = \Pi_i R_i(t). \tag{4.32}$$

Similarmente, aproveitando as derivações anteriores para taxa de falha, tem-se (4.33).

$$\lambda(t) = \sum_i \lambda_i(t), \tag{4.33}$$

onde, para obter a taxa de falha do sistema, somam-se as taxas de falha dos componentes desse sistema. A notação λ utilizada nesse capítulo não tem qualquer relação com seu uso no capítulo 3.

4.4.2 Taxa de falha dependente do tempo

Conforme explicitado pela curva da banheira, sistemas e componentes passam por diferentes etapas durante o clico de vida útil. Essas etapas são explicitadas pela taxa de falha. Apesar das taxas de falhas serem amplamente utilizadas para descrever os fenômenos da confiabilidade, esses são definidos pela suposição de que a taxa em que os sistemas falham não está relacionada com sua idade.

A aproximação para taxa de falha constante é muitas vezes bem adequada, até quando o sistema ou seus componentes demonstram moderado sinais de falhas por imaturidade ou efeitos do envelhecimento. Muitos desses sinais podem ser limitados por controles de

qualidade em manutenção e reposição cíclica de partes e componentes (LEWIS, 1994). Até mesmo quando a taxa de falha varia com o tempo, é possível usar taxas de falhas constantes que envolvam a curva, mas, nesses casos, será moderadamente pessimista (LEWIS, 1994).

Há uma variedade de situações em que o explícito tratamento de falhas precoces ou efeitos da idade ou ambos, requerem o uso de taxa de falha dependentes do tempo. Isso é feito ao considerar a influência de que um tempo de operação acumulado T_0 na probabilidade que um dispositivo possa sobreviver por um tempo adicional t.

Seja $R(t|T_0)$, definido pela Equação (4.34), a confiabilidade de um dispositivo que tenha operado por um tempo T_0.

$$R(t|T_0) = P\{\mathbf{t}' > T_0 + t|\mathbf{t}' > T_0\}. \tag{4.34}$$

Seja um evento X, dependente de um evento Y. Define-se a probabilidade condicional do evento X, dado o evento Y, como $P\{X|Y\}$. Se a probabilidade do evento Y correr foi maior que zero, tem-se (4.35) (LEWIS, 1994).

$$P\{X|Y\} = \frac{P\{X \cap Y\}}{P\{Y\}}. \tag{4.35}$$

Dada a condição (4.35), pode-se escrever (4.34) como (4.36) (LEWIS, 1994).

$$P\{\mathbf{t}' > T_0 + t|\mathbf{t}' > T_0\} = \frac{P\{(\mathbf{t}' > T_0 + t) \cap (\mathbf{t}' > T_0)\}}{P\{\mathbf{t}' > T_0 + t\}}. \tag{4.36}$$

Como $(\mathbf{t}' > T_0 + t) \cap (\mathbf{t}' > T_0) = \mathbf{t}' > T_0 + t$, tem-se (4.37) (LEWIS, 1994).

$$R(t|T_0) = \frac{P\{\mathbf{t}' > T_0 + t\}}{P\{\mathbf{t}' > T_0\}}. \tag{4.37}$$

A confiabilidade de um item novo é dada pela Equação (4.38) (LEWIS, 1994).

$$R(t) = R(t|T_0) = P\{\mathbf{t}' > t\}. \tag{4.38}$$

E tem-se (4.39) (LEWIS, 1994).

$$R(t|T_0) = \frac{R(t + T_0)}{R(T_0)}. \tag{4.39}$$

Pela Equação (4.25), obtém-se (4.40) (LEWIS, 1994):

$$R(t|T_0) = exp\left[-\int_{T_0}^{t+T_0} \lambda(t')dt'\right]. \tag{4.40}$$

Para saber se o tempo de operação anterior T_0 tem influência positiva ou negativa na confiabilidade do dispositivo, deriva-se $R(t|T_0)$ com respeito ao tempo T_0, conforme (4.41) (LEWIS, 1994).

$$\frac{\partial}{\partial T_0} R(t|T_0) = -[\lambda(T_0) - \lambda(T_0 + t)]R(t|T_0). \tag{4.41}$$

O aumento de T_0 só apresenta melhorias na confiabilidade se a taxa de falha decresce. Se a taxa de falha crescer, o tempo de serviço apenas adiciona a deterioração do dispositivo e a confiabilidade da vida útil cai (LEWIS, 1994).

Para modelar de forma mais precisa falhas precoces e de envelhecimento é preciso recorrer a distribuições de tempo para falhas específicas. A distribuição *Weibull* é a mais universalmente utilizada (LEWIS, 1994) e com ela é possível modelar falhas precoces, falhas aleatórias e efeitos do envelhecimento.

Uma outra distribuição importante é a *Lognormal*, amplamente utilizada em engenharia de confiabilidade para descrever falhas causadas por fadiga, incertezas nas taxas de falha e uma variedade de outros fenômenos (LEWIS, 1994).

As distribuições de probabilidade são o meio pelo qual se estimam as funções de densidade apresentadas.

4.4.2.1 Distribuição *Weibull*

A distribuição *Weibull* é uma das mais usadas em cálculos de confiabilidade, podendo modelar de forma apropriada uma grande variedade de comportamentos de taxa de falhas. Incluindo taxas de falha constante e taxas de falhas precoces e de envelhecimento (LEWIS, 1994). Essa distribuição é bem flexível quanto a aplicabilidade, sendo utilizada em uma grande variedade de fenômenos.

A CDF da *Weibull* é definida por (4.42) (LEWIS, 1994).

$$F(x) = 1 - exp\left[-\left(\frac{x}{\theta}\right)^{\gamma}\right], 0 \leq x \leq \infty, \tag{4.42}$$

onde θ é o parâmetro de escala e γ é o parâmetro de forma. O PDF da *Weibull*, definido

em (4.43), é encontrado pela derivação indicada na Equação (4.4).

$$f(x) = \frac{\gamma}{\theta}\left(\frac{x}{\theta}\right)^{\gamma-1} exp\left[-\left(\frac{x}{\theta}\right)^{\gamma}\right], 0 \leq x \leq \infty. \tag{4.43}$$

A Figura 4.4 ilustra o PDF para diferentes valores de γ.

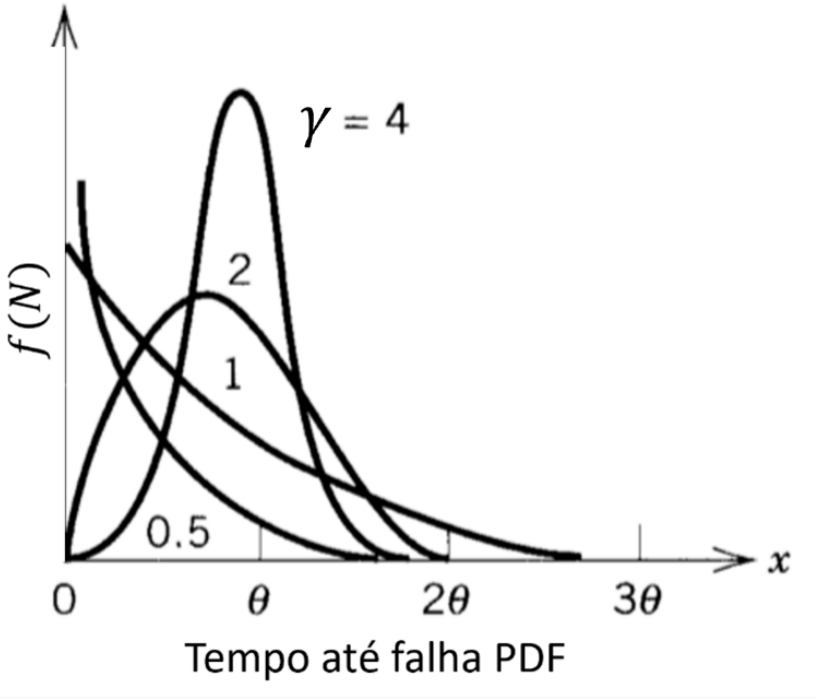

Figura 4.4: A distribuição *Weibull*.
Fonte: Adaptada de Lewis, 1994.

Pode ser observado na Figura 4.4 a influência do parâmetro γ, parâmetro de forma na curva da função, enquanto o parâmetro de escala influencia a amplitude da curva.

Considerado agora um sistema em cadeia, onde em cada componente, um evento é descrito por variáveis aleatórias $x_1, x_2, x_3, \cdots, x_n$. O evento para a cadeia é também uma variável aleatória, $\mathbf{y}$, e a probabilidade de $\mathbf{y} > y$ é dado por (4.44) (LEWIS, 1994).

$$P\{\mathbf{y} > y\} = P\{\mathbf{x_1} > y \cap \mathbf{x_2} > y \cap \mathbf{x_3} > y \cap \cdots \cap \mathbf{x_n} > y\}. \tag{4.44}$$

Se o evento nos componentes for independente, (4.44) pode ser reescrito como (4.45) (LEWIS, 1994).

$$P\{\mathbf{y} > y\} = P\{\mathbf{x_1} > y\}P\{\mathbf{x_2} > y\}P\{\mathbf{x_3} > y\}\cdots P\{\mathbf{x_n} > y\}. \tag{4.45}$$

Se todos os eventos forem governados por uma distribuição idêntica, a probabilidade pode ser expressa em uma único CDF, definido em (4.46).

$$P\{\mathbf{x_i} > y\} = 1 - P\{\mathbf{x_i} \leq y\} = 1 - F_x(y). \tag{4.46}$$

De mesma forma, CDF para y pode ser escrito como $F_y(y) = 1 - P\{\mathbf{y} > y\}$. A Equação (4.44) torna-se (4.47) (LEWIS, 1994).

$$F_y(y) = 1 - [1 - F_x(y)]^n. \tag{4.47}$$

Se o evento em cadeia for governado pela distribuição *Weibull* (LEWIS, 1994), obtém-se (4.48).

$$F_x(x) = 1 - exp\left[-\left(\frac{x}{\theta}\right)^{\gamma}\right]. \tag{4.48}$$

Combinando as duas equações (LEWIS, 1994), tem-se (4.49).

$$F_y(y) = 1 - exp\left[-\left(\frac{y}{\theta}\right)^{\gamma}\right]^N = 1 - \exp^{-N\left(\frac{y}{\theta}\right)^{\gamma}}. \tag{4.49}$$

O evento para a cadeia, expresso como distribuição *Weibull*, é definido pela Equação (4.50) (LEWIS, 1994).

$$F_y(y) = 1 - exp\left[-\left(\frac{y}{\theta'}\right)^{\gamma}\right], \tag{4.50}$$

onde se tem (4.51) (LEWIS, 1994).

$$\theta' = N^{-\frac{1}{\gamma}}\theta. \tag{4.51}$$

Até quando a distribuição não é explicitamente conhecida, mas a falha do mecanismo surge de diferentes defeitos concorrentes, a distribuição *Weibull* é muitas vezes um bom ajuste empírico para os dados coletados (LEWIS, 1994).

A distribuição *Weibull* assume que taxa de falha está na forma da Equação (4.52) (LEWIS, 1994).

$$\lambda(t) = \frac{m}{\theta}\left(\frac{t}{\theta}\right)^{\gamma-1}. \tag{4.52}$$

A partir dessa taxa de falha, é possível obter o PDF, definido por (4.53) (LEWIS, 1994).

$$f(x) = \frac{\gamma}{\theta}\left(\frac{t}{\theta}\right)^{\gamma-1} \exp\left[-\left(\frac{t}{\theta}\right)^{\gamma}\right]. \tag{4.53}$$

Integrando a variável de tempo de 0 a t, obtém-se o CDF definido pela Equação (4.54).

$$F(t) = 1 - \exp\left[-\left(\frac{t}{\theta}\right)^{\gamma}\right]. \tag{4.54}$$

A Figura 4.5 ilustra as propriedades de $\lambda(t), f(t)$ e $R(t)$ para diferentes valores de γ, representado na figura por m. Quando $m = 1$, obtém-se a distribuição exponencial correspondente a uma taxa de falha constante. Para $m < 1$, a taxa de falhas é decrescente e para $m > 1$, a taxa de falhas é crescente, tipicamente efeitos de envelhecimento (LEWIS, 1994).

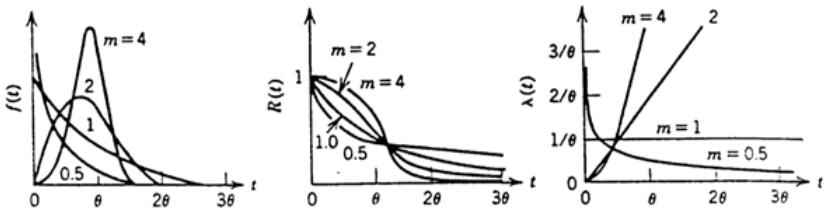

Figura 4.5: Função de distribuição de probabilidades, confiabilidade e taxa de falhas na distribuição *Weibull*.

Fonte: Lewis, 1994.

Observa-se na Figura 4.5 a influência do parâmetro de forma m na curva da função PDF (gráfico da esquerda), a curva de confiabilidade (gráfico central), e a taxa de falha (gráfico da direita).

4.4.2.2 Distribuição *Lognormal*

Segundo Lewis (1994), o desgaste de uma máquina pode ser proporcional ao produto da magnitude das demandas feitas sobre esse sistema. Nesse cenário, a distribuição *Lognormal* apresenta um ajuste melhor aos dados.

Seja x uma variável que pode ser expressa pela soma de variáveis aleatórias $x_i, i = 1, 2, \cdots, n$ onde não exista um dominante, então x pode ser descrito como uma distribuição normal, mesmo que x_i seja descrito por uma distribuição não-normal e que não necessariamente seja a mesma distribuição para diferentes i (LEWIS, 1994).

Uma segunda frequência consiste em uma variável aleatória y, produto de variáveis

aleatórias y_i, definida pela Equação (4.55) (LEWIS, 1994).

$$y = y_1 y_2 \cdots y_N. \tag{4.55}$$

Aplicando o logaritmo natural em (4.55), tem-se (4.56).

$$x = \ln y. \tag{4.56}$$

Então, x é distribuído normalmente e y é distribuído lognormalmente.

Para obter a distribuição *Lognormal*, parte-se da distribuição normal para x, definida por (4.57) (LEWIS, 1994).

$$f_x(x) = \frac{1}{\sqrt{2\pi}\sigma_x} \exp\left[-\frac{1}{2\sigma^2}(x - \mu_x)^2\right], \tag{4.57}$$

onde μ_x é a média dos valores de x e σ_x^2 é a variância da distribuição em x.

Para obter o PDF em y, transforma-se a distribuição, de acordo com (4.58) (LEWIS, 1994).

$$f_y(y) = f_x(x)\left|\frac{dx}{dy}\right|. \tag{4.58}$$

Notando a definição (4.59).

$$\frac{dx}{dy} = \frac{d}{dy} lny = \frac{1}{y}. \tag{4.59}$$

Substituindo $x = lny$ em (4.57), obtém-se (4.60) (LEWIS, 1994).

$$f_y(y) = \frac{1}{\sqrt{2\pi}\sigma_x y} \exp\left\{-\frac{1}{2\sigma^2}[ln(y) - \mu_x]^2\right\}. \tag{4.60}$$

O CDF correspondente é obtido pela integral de y com um limite inferior $y = 0$, segundo Lewis (1994), definido pela Equação (4.61).

$$F_y(y) = \Phi\left[\frac{1}{\sigma_x}[ln(y) - \mu_x]\right], \tag{4.61}$$

onde Φ é o valor da distribuição normal padronizada.

A Figura 4.6 contém os gráficos PDF e CDF para a distribuição *Lognormal*.

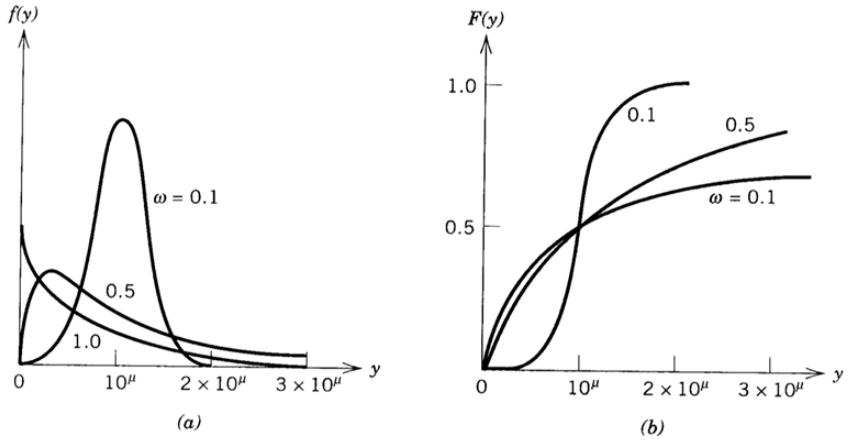

Figura 4.6: Distribuição *Lognormal*: PDF (a) e CDF (b).
Fonte: Lewis, 1994.

Pode ser observado na Figura 4.6 a influência do parâmetro nas diferentes curvas da função, tanto para PDF, na Figura a), quanto CDF, na Figura b).

Como indicado no item 4.4.2, a distribuição normal é muito útil para descrever o processo de envelhecimento, quando é possível indicar um tempo para falha com uma certa incerteza. A função *Lognormal* é uma distribuição relacionada, útil para descrever distribuição de falha para uma variedade de situações (LEWIS, 1994).

O PDF para tempo de falha é definido na Equação (4.62), segundo Fogliatto (2011).

$$f(t) = \frac{1}{\sqrt{2\pi}\sigma t} exp\left\{-\frac{1}{2\sigma^2}[ln(t) - \mu]^2\right\}, \tag{4.62}$$

e o CDF correspondente é dado por Fogliatto (2011) na Equação (4.63).

$$F(t) = \Phi\left(\frac{ln(t) - \mu}{\sigma}\right), \tag{4.63}$$

onde Φ pode ser calculado pela função definida na Equação (4.64), adaptada pelos autores da aproximação de Byrc (2002) ou pelas tabelas do Anexo 1.

$$\Phi(z) = 1 - \frac{(4-\pi)z + \sqrt{2\pi}(\pi - 2)}{(4-\pi)z^2\sqrt{2\pi} + 2\pi z + 2\sqrt{2\pi}(\pi - 2)} exp - \frac{z^2}{2}. \tag{4.64}$$

Nos testes realizados pelos autores observou-se que a Equação (4.64) não fornecia boas estimativas para o intervalo $z < -1$. Por esse motivo, aplicou-se a regra binária expressa

na Equação (4.65) (Autor, 2021).

$$\Phi(z') = \begin{cases} \Phi(z), & \text{se } z \geq 0 \\ 1 - \Phi(|z|), & \text{se } z < 0 \end{cases} \tag{4.65}$$

Os resultados obtidos pela regra exposta na Equação (4.65) foram comparados diversas vezes com a Tabela do CDF normal padronizado (Anexo 1), sendo sempre muito próximos. Destaca-se que valores entre dois intervalos na Tabela do CDF normal padronizado foram ajustados por regressão linear simples nas comparações.

4.4.3 Substituições

Até o momento, foram consideradas distribuições de tempos de falhas para sistemas novos em $t = 0$. Na maioria das situações, falhas não são consideradas o fim da vida útil. Os sistemas passam por manutenções, são reparados ou substituídos e continuam a operar (LEWIS, 1994).

Para modelar essas situações, utiliza-se a aproximação da taxa de falha constante. Nesse caso, a taxa de falha é dada em função do tempo médio entre falhas, do inglês *mean time between failure MTBF*, ao invés de MTTF. Os dois termos são o mesmo número, considerando reparos perfeitos (LEWIS, 1994).

4.5 Estimativa de parâmetros e tempos até falha

A definição mais usual de confiabilidade de um item é dada em termos de sua probabilidade de sobrevivência até um tempo t de interesse. A determinação de tal probabilidade é possível através de modelagens dos tempos até a falha da unidade em estudo. A modelagem dos tempos até falha é, portanto, central em estudos de confiabilidade (FOGLIATTO et al., 2011).

Conforme a Equação (4.17), conhecendo-se $f(t)$ é possível determinar a confiabilidade $R(t)$ do item para qualquer tempo t, além de outras medidas de interesse da seção 4.3.2. Os parâmetros da função que caracteriza uma determinada unidade são estimados utilizando informações de TTF.

4.5.1 Métodos de estimação de parâmetros

Para uma amostra aleatória completa de tempos até falha $T_1, \cdots, T_n$ obtida de uma população de interesse, tal que T_i são variáveis aleatórias independentes que seguem uma mesma distribuição de probabilidade, utilizam-se as informações da amostra para estimar o vetor de parâmetros θ da distribuição de forma a desenvolver um estimador $\hat{\Theta}$ para θ (FOGLIATTO, 2011).

Existem diversos métodos de estimação: (i) dos momentos, (ii) dos mínimos quadrados e o mais utilizado (iii) da máxima verossimilhança. Independentemente do método, deseja-se que retenham as seguintes propriedades (FOGLIATTO, 2011):

a) Não-tendencioso – não subestimar ou superestimar o valor real do parâmetro;

b) Consistente – convergir rapidamente para o valor real do parâmetro à medida que o tamanho da mostra aumenta;

c) Eficiente – apresentar a menor variância dentro os estimadores usados;

d) Suficiente – utilizar toda a informação acerca do parâmetro que a amostra possui.

Um dos métodos estimativa dos parâmetros da distribuição *Weibull* é a função da verossimilhança, definida na Equação (4.66), como apresentado por Fogliatto (2011). Os parâmetros θ e λ são obtidos de forma iterativa.

$$L(\theta, \gamma) = \frac{\gamma}{\theta} \Pi_{i=1}^{n} t_i^{\gamma-1} exp\left(-\frac{1}{\theta}\sum_{i=1}^{n} t_i^{\gamma}\right). \tag{4.66}$$

Para a distribuição *Lognormal*, os estimadores de verossimilhança são dados de forma direta pelas equações (4.67) e (4.68) (FOGLIATTO, 2011).

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} lnt_i, \tag{4.67}$$

$$\hat{\sigma}^2 = \frac{1}{n}\left\{\sum_{i=1}^{n}(\ln t_i)^2 - \left[\frac{(\sum_{i=1}^{n}\ln t_i)^2}{n}\right]\right\}. \tag{4.68}$$

A distribuição *Lognormal* não é centrada em μ, como é o caso de uma distribuição normal (FOGLIATTO, 2011).

4.5.2 Verificação do ajuste de dados a funções de distribuições de probabilidade

Após estimados os parâmetros das funções de distribuição de probabilidade, é necessário verificar qual distribuição tem mais aderência aos dados coletados. Existem duas formas mais comuns de verificar o ajuste de dados a distribuições hipotetizadas: (i) Gráfica, através de histogramas de frequência e papéis de probabilidade, e (ii) analítica, através de testes de aderência (FOGLIATTO et al., 2011).

Os testes analíticos de aderência mais utilizados são o do qui-quadrado e o de *Kolmogorov-Smirnov*. Ambos os testes apresentam a estrutura de um teste de hipóteses, em que a hipótese nula H_0 é de que os dados sigam uma determinada distribuição hipotetizadas (FOGLIATTO, 2011).

O teste do qui-quadrado é um teste paramétrico, com estatística de teste seguindo uma distribuição do qui-quadrado, caso H_0 seja verdadeira. A ideia é calcular a soma dos quadrados das diferenças entre frequências esperadas e frequências empíricas observadas em diferentes intervalos de classe. Se a soma ultrapassar um determinado valor tabelado, rejeita-se H_0 (FOGLIATTO, 2011).

O teste de *Kolmogorov-Smirnov* é implementado de maneira análoga, entretanto considerando frequências acumuladas ao invés de frequências absolutas, utilizando melhor a informação contida nas amostras. Esse teste é mais adequado em situações com poucos dados amostrais (FOGLIATTO, 2011).

Nesse trabalho serão utilizados os coeficientes de correlação e determinação apresentados no item 4.5.3. por facilidade de implementação.

4.5.3 Coeficiente de correlação e determinação

A análise de Correlação é uma ferramenta importante para as diferentes áreas do conhecimento, tendo grande utilidade como uma das etapas de utilização de outras técnicas de análise (LIRA, 2006).

O método usualmente utilizado para medir a correlação entre duas variáveis é o coeficiente de correlação (CC) de *Pearson*, também conhecido como CC do Momento Produto (LIRA, 2006).

CC é definido por (4.69) (ASUERO, 2007).

$$R = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \tag{4.69}$$

onde n é o número de observações da amostra, $\bar{\hat{y}}$ é a média aritmética do valor previsto de y, $\hat{y}$ é a estimativa de y e $\bar{y}$ é a média aritmética de y.

Uma forma alternativa de interpretar CC é por meio do Coeficiente de Determinação (CD). O método do CD, uma derivação do CC, é uma ferramenta importante e útil para determinar a qualidade do ajuste de regressões lineares aos dados, definido pela Equação (4.70) (ASUERO, 2007).

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{4.70}$$

Como assume valores no intervalo $0 \leq R^2 \leq 1$ é muito comum expressar R^2 multiplicado pela centena. Nesse caso, fornece uma porcentagem (LIRA, 2006).

Segundo Barret (1974), R^2 mede a qualidade dos ajustes, usado para comparar modelos de previsão. Para um banco de dados (BD), maiores valores de R^2 refletem um aumento da precisão de previsão por equações de regressão. Um dos objetivos do trabalho, detalhado no Capítulo 5.

Capítulo 5

Metodologia

Nesse capítulo é apresentada a metodologia utilizada nesse estudo. A metodologia foi dividida em três partes:

1. Geração dos dados e Treinamento das redes neurais.
2. Cálculo de parâmetros industriais, modelagem da função objetivo e otimização.
3. Construção do aplicativo.

A Figura 5.1 apresenta um fluxograma das etapas.

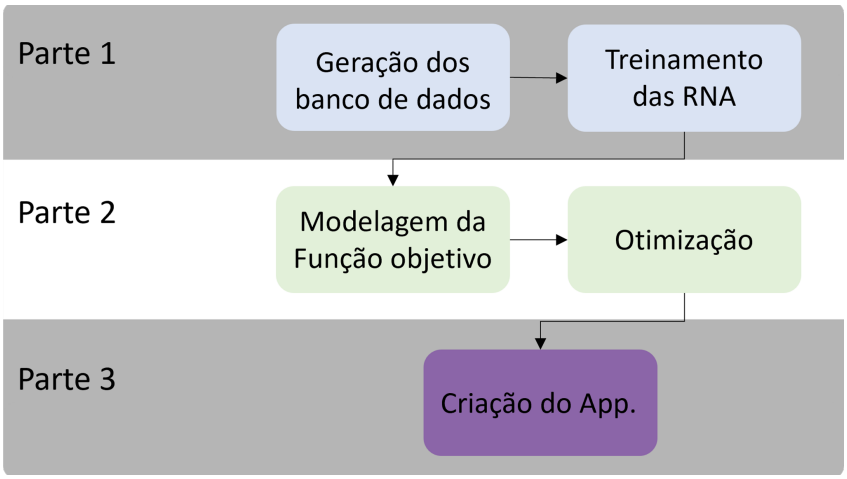

Figura 5.1: Divisão em partes - Fluxograma
Fonte: Autor,2021.

Em (1), dados sintéticos são criados conforme o item 5.1 e o treinamento ocorre em sequência, como o exposto no item 5.2. A parte (1) também possui uma subdivisão:

Função de distribuição *Weibull* e Função de distribuição *Lognormal*. Os processos são apresentados para a distribuição *Weibull*. Devido a existência de métodos diretos para ajustes da distribuição *Lognormal*, esta não participou do treinamento.

Na parte (2), com os dados ajustados por RNA, calcula-se os principais parâmetros da manutenção industrial. Estes são então usados para modelar os processos de MP. Uma função objetiva de custo de operação é apresentada com base nesses parâmetros e esse modelo é resolvido por otimização restrita, de forma a recomendar ações ótimas dependente das prioridades configuradas (CAMP, 2021).

Por fim, em (3), um aplicativo de computador é proposto, de forma a desempenhar os processos anteriores de forma simplificada e acessível.

Todos os processos foram executados em um ambiente *Windows* 11 Pro *build* 22000.71, com processador Intel Corei7-1065G7, 16GB de memória RAM, e armazenamento suficiente, plataforma Anaconda 4.10.1 e linguagem de programação *Python* 3.8.10.

5.1 Geração dos bancos de dados

Os conjuntos de dados utilizados nos treinamentos são todos sintéticos, gerados por meio de um algoritmo criado pelo autor e disponível no Apêndice A (i).

Cada BD possui mil pares de vetores característica x e rótulo y, com cinco elementos TTF cada vetor característica, distribuídos conforme uma das duas distribuições de probabilidades consideradas. O vetor de rótulos y são compostos pelos parâmetros θ e γ de cada vetores característica *Weibull* e μ e σ para vetores de características *Lognormal*.

O tamanho de cada vetor características foi influenciado pela obra de Liu (1995), onde o autor afirma que não é incomum ter disponível apenas uma pequena quantidade de dados de amostras de confiabilidade, quando testando um sistema.

Cinco bancos de dados foram criados de forma direta, e um sexto, criado com a junção dos cinco anteriores. Somado a isso, antes de serem exportados para arquivos de planilhas, a ordem dos vetores i foi alterada par-a-par aleatoriamente, de forma que vetores características e rótulos de mesmo índice foram movidos para um mesmo novo índice. Dessa forma, o aprendizado não ocorrerá com os dados em qualquer tipo de ordem (crescente ou decrescente).

O processo se repetiu para ambas as distribuições de probabilidade. Destes cinco mil vetores características, quatro mil foram utilizados no treinamento e os mil restantes

foram utilizados na validação da RNA.

Retornando ao item 3.2, dado pela Equação (5.1) (BISHOP, 2006).

$$S_n = \{(x^{(i)}, y^{(i)}), i = 1, 2, 3, \cdots, n\}. \tag{5.1}$$

Cada posição i de (5.1) é formada por dois pares diferentes x^i, y^i. Devido a aleatoriedade na criação dos vetores características, é improvável, apesar de não verificado, que existam dois pares iguais em todos os bancos de dados. .

Todos eles estão disponíveis para download em formato .xlsx no Apêndice B (i) (Testados no *Microsoft Excel* 2019).

5.1.1 Distribuição *Weibull*

O processo de geração de cada vetor característica seguiu a Equação (5.2), definida em *Numpy* (2021).

$$x_j^i = -\theta \ln z_j^{i\frac{1}{\gamma}}, \tag{5.2}$$

onde z é uma sequência aleatória e uniformemente distribuída no intervalo $[0, 1)$, de mesmo tamanho do vetor de características, como apresentado na Equação (3.1). E x^i é uma sequência de tempos resultante, segundo a distribuição *Weibull*.

Os parâmetros θ e γ, ambos de tamanho n, são gerados segundo regras definidas pelo autor nas Equações (5.3) e (5.4). Esse formato foi inspirado no trabalho de Liu (1995) para garantir uma distribuição uniforme dos parâmetros ao longo do BD.

$$\gamma^i = 0,5 + 0,5 \left[\frac{n}{150}\right], \tag{5.3}$$

e

$$\theta^i = 10^{\{1;3.5\}}, \tag{5.4}$$

onde [ ] significa truncamento sem casa decimal e $\{a; b\}$ significa escolha aleatória no intervalo (a, b).

Os valores de θ foram convertidos em escala logarítmica para treinamento, devido a grande diferença de valores entre θ e γ.

5.1.2 Distribuição *Lognormal*

O processo de geração de cada vetor característica segundo a distribuição *Lognormal* foi gerado a partir da função *random.lognormal* da biblioteca *Numpy* (Numpy, 2021).

Os parâmetros μ e σ foram escolhidos aleatoriamente por meio da função *random* da biblioteca *Numpy* de acordo com os intervalos: $3 \leq \mu \leq 8$ e $3e-2 \leq \sigma \leq 5e-1$. Esse intervalo representa valores comumente encontrado em outros trabalhos.

Foram gerados cinco BD por meio do algoritmo de geração (Apêndice A.ii) e estes, convergidos em um único BD, cuja ordem dos indexes foi alterada aleatoriamente. Para os resultados do capítulo 6, foram usados os três primeiros vetores características segundo essa nova ordenação. Todos os dados gerados estão disponíveis no Apêndice B.

O ajuste dos parâmetros da distribuição *Lognormal* seguiu as Equações (4.67) e (4.68). As curvas CDF utilizaram uma aproximação para CDF normal padronizado disposto na Equação (4.64) e (4.65). Os resultados dessa última passaram por um teste de sanidade, composto por comparações aproximadas por regressão linear dos intervalos da tabela CDF normal padronizadas, disponíveis no Anexo 1.

5.2 Treinamento das Redes Neurais

O processo de treinamento das redes neurais utilizou a plataforma *TensorFlow* versão 2.5.0 em conjunto com Keras versão 2.5.0, fazendo uso da tecnologia Intel *Deep Learning Boost, Advanced Vector Extensions 512 e Vector Neural Network Instruction.*

Diversos procedimentos e configurações foram utilizadas em treinamento. No capítulo 6 está retratado o conjunto que apresentou maior sucesso.

O modelo criado utilizou o otimizador Adam (KINGMA, 2015), baseado em SGD, e treinamento em *Batches*, não fornecendo todos os dados simultaneamente. Foram utilizados diferentes valores para taxa de aprendizado, sendo essa dinâmica, segundo o método Decaimento Exponencial, definido pela Equação (5.5) (*TensorFlow*, 2021).

$$\eta = \frac{\eta_0}{1 + \frac{d_r s}{d_s}}, \tag{5.5}$$

onde η é a taxa de aprendizado instantânea, η_0 é a taxa de aprendizado inicial, d_r é o *decay rate*, taxa de decremento, s é o step, etapa do treinamento e d_s é o *decay steps*, decremento da etapa de treinamento.

Nesse caso, uma etapa é uma atualização por SGD de um *batch* enquanto uma iteração completa de todos os dados é chamada *epoch*, composto de diversas etapas. Cada *epoch* é igual ao tamanho do conjunto de treinamento (item 3.2), dividido pelo tamanho do *batch*.

Em todos os treinamentos, o erro (Equação 3.28) fora substituído por *Mean Absolute Percentile Error (MAPE)*. Isso ocorreu pelos baixos valores dos parâmetros objetivo, com diferenças muito menores que um, $E \ll 1$.

Foram treinadas ao todo 38 redes neurais nesse trabalho, todas com configurações e erros de treinamento e validação diferentes. A Rede Neural RNA *Weibull* 38 foi a que atingiu o requesito de erro máximo e por isso foi utilizada nesse trabalho. Um diário técnico do treinamento está disponivel no repositório do trabalho hospedado no *GitHub* em http://guthub.com/lukekort/OCM. O diário contém a configuração, resultados e redes de todos os 38 treinamentos.

5.3 Parâmetros da manutenção Industrial e função objetiva

O parâmetro industrial confiabilidade, definido pela Equação (4.17), foi calculado conforme as equações (4.53) para a Distribuição *Weibull* e (4.63) para a Distribuição *Lognormal*.

O modelo de custo de operação utilizado nesse trabalho é uma adaptação dos autores do modelo extraído do trabalho de Ghosh e Roy (2009), com adição de elementos e termos, uma generalização do proposto por Lewis (1994) em seu livro.

Seja C_m o custo de uma ocorrência de MP, C_r um reparo não programado e C_{inc} o custo por paralização e outras perdas financeiras devido a quebra da unidade produtiva, o custo total de operação durante um intervalo de tempo, desde o início da operação até o momento de sua primeira intervenção de manutenção $T_m(1)$, é definido originalmente segundo Ghosh e Roy (2009) por (5.6).

$$C_t^{0\to1} = C_m^{0\to1} R[T_m(1)] + C_r^{0\to1}[1 - R[T_m(1)]]. \tag{5.6}$$

Os autores adicionaram o termo C_{inc} a Equação (5.6), obtendo (5.7).

$$C_t^{0\to1} = C_m^{0\to1} R[T_m(1)] + (C_r^{0\to1} + C_{inc})[1 - R[T_m(1)]]. \tag{5.7}$$

Seja $R(t|T_0)$ a confiabilidade de um dispositivo que tenha operado por um tempo T_0, como definido pela Equação (4.34). A confiabilidade é definida pela Equação (5.8) (LEWIS, 1994).

$$R(t|T_0) = \frac{R(t+T_0)}{R(t_0)}. \tag{5.8}$$

O custo de operação entre a primeira e segunda intervenção de manutenção é definido pela Equação (5.9), adaptada do trabalho de Ghosh e Roy (2009).

$$C_t^{1\to 2} = C_m^{1\to 2}\left[\frac{R[T_m(2)]}{R[T_m(1)}\right] + (C_r^{1\to 2} + C_{inc})\left[1 - \frac{R[T_m(2)]}{R[T_m(1)}\right]. \tag{5.9}$$

Assumindo N como o número total de intervenções de manutenção durante o tempo de serviço T_{ser}, tem-se a Equação (5.10), adaptada do trabalho de Ghosh e Roy (2009).

$$C(T) = \sum_{j=1}^{N}\left[C_m\frac{R(jT)}{R[(j-1)T]}\right] + (C_r + C_{inc})\left[1 - \frac{R(jT)}{R[(j-1)T]}\right], \tag{5.10}$$

onde o intervalo T é dado pela Equação (5.11) (GHOSH, ROY, 2009).

$$T = \frac{T_{ser}}{N}. \tag{5.11}$$

Nesse trabalho é considerada a simplificação de que as operação de manutenção são sempre perfeitas e que a unidade volta a operar com confiabilidade máxima (estado "Tão bom quanto novo") após cada ocorrência de manutenção (item 4.4.3) e que estas operações são sempre perfeitas, restaurando a confiabilidade do equipamento, sem adicionar outros defeitos. Também não serão considerados quaisquer mudanças em custos de intervenção ou acidente ao logo da vida útil do sistema (5.11). Logo, a modelagem da estimativa do custo de operação pode ser dada pela Equação (5.12).

$$C_T = \frac{T_{ser}}{T}\{C_m R(T) + (C_r + C_{inc}[1 - R(t)])\}. \tag{5.12}$$

Devido a uma dificuldade em representar o correto número de quebras não programadas e consequentemente, manutenções corretivas, os autores apresentam também uma modificação a multiplicação do segundo termo. A nova Equação é dada por (5.13).

$$C_T = \frac{T_{ser}}{T}C_m R(T) + \frac{T_{ser}}{MTBF}(C_r + C_{inc})[1 - R(t)]. \tag{5.13}$$

A Figura 5.2 ilustra as diferenças de confiabilidade ao longo do tempo para um equipamento com e sem MP.

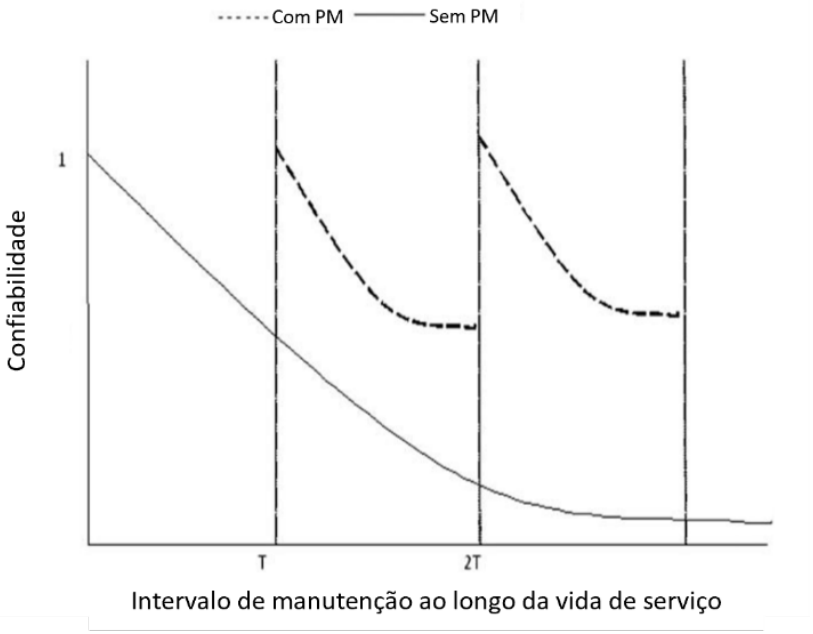

Figura 5.2: Confiabilidade vs. Tempo com e sem MP
Fonte: Adaptada de Ghosh e Roy (2009).

A Figura 5.2 ilustra a confiabilidade para um equipamento com e sem MP. É perceptível a degradação da confiabilidade ao longo do tempo, sendo que em um cenário com MP ativa, em intervalos de tempo T, essa confiabilidade é restaurada ao estado de novo, enquanto em um cenário sem MP, a confiabilidade degrada de forma contínua.

O autor também modificou a formulação de disponibilidade original exposta na Equação (4.11) para considerar dois tempos de operação de manutenção diferentes, o tempo de manutenção t_m referente a MP programada e o tempo de reparo t_r referente ao tempo de manutenção corretiva emergencial, sendo este último com duração maior, devido as dificuldades impostas pela situação emergencial.

A nova formulação, como definida pelos autores, é dada pela Equação (5.14).

$$A(t) = \frac{t}{t + R(t)T_m + (1 - R(t))T_r}, \tag{5.14}$$

onde t no cenário estudado por esse trabalho representa o intervalo entre cada operação de MP programada.

Destaca-se que a Equação (5.14) não calcula a disponibilidade em um intervalo t

separado, mas em média, dentre vários intervalos t ao longo da vida útil do equipamento t_{ser}, espera-se que (5.14) forneça uma boa estimativa da disponibilidade média.

5.4 Otimização

Esse estudo trabalhou com estrutura de um único plano de manutenção, com o objetivo de otimizar os intervalos entre MP. O modelo matemático consiste em otimizar a função de custo da Equação (5.13), restrita pela função de disponibilidade da Equação (5.14) e/ou função de confiabilidade por meio do método de otimização PSO, apresentado no item 2.5.

Para o processo de otimização, foi utilizado o aplicativo de otimização Opp! v1.11, desenvolvido pelo Autor (2021). A Figura 5.3 apresenta uma captura de tela da interface do programa.

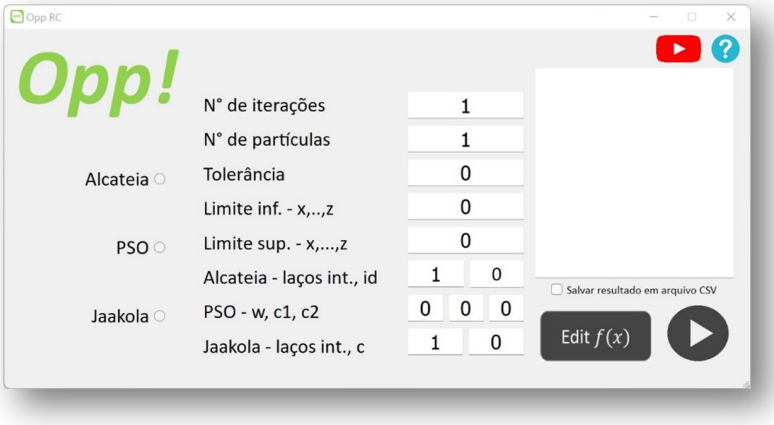

Figura 5.3: Interface do aplicativo Opp!
Fonte: Autor,2021.

A Figura 5.3 contém uma captura de tela da interface central do programa. O aplicativo Opp! foi desenvolvido pelo autor anteriormente a esse trabalho, como ferramenta de otimização. O aplicativo está disponível no Apêndice A ix.

Para os cálculos de otimização, sendo todos os indicadores industriais funções $f(t)$ e a modelagem do custo função $f(T)$, o autor considerou a igualdade (5.15).

$$t = T. \tag{5.15}$$

A implementação computacional para introdução da função (5.13) e (5.14) no aplicativo Opp está disponível no Apêndice A (ii). Nessa implementação computacional existe a possibilidade de restringir a função objetiva também pela função de confiabilidade (4.17).

Destaca-se que em diversos testes, os autores encontraram inúmeras situações em que não foi possível encontrar um intervalo que atendia a ambas as restrições, apenas em casos muito específicos. Somado a isso, tanto a função objetiva, quanto a função disponibilidade tem em suas imagens o tempo e a confiabilidade. Por esses motivos, esse trabalho prioriza a otimização restrita por (5.13) e apresenta apenas um cenário de otimização restrita por (4.17).

5.5 Construção do aplicativo

Para facilitar a execução dos muitos processos realizados nesse trabalho, um aplicativo de computador foi construído. O aplicativo, chamado de OCM, foi escrito na linguagem *Python* 3.8.10, utilizou a biblioteca Qt 5.9.7 para criação da interface gráfica e foi incorporado por meio do *Pyinstaller 4.3*. A Figura 5.4. contém um fluxograma do funcionamento do aplicativo.

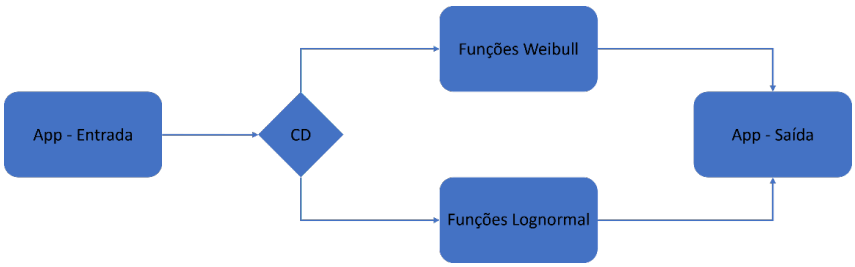

Figura 5.4: Estrutura do aplicativo OCM
Fonte: Autor,2021.

A estrutura do aplicativo OCM é exposta na Figura 5.4. O usuário entra com os dados na interface do aplicativo que utiliza uma RNA já treinada para fazer o ajuste Weibull e equações diretas para fazer o ajuste Lognormal. O CD é calculado por meio da Equação 4.69 para auxiliar na escolha da melhor função de distribuição (a função com maior índice de determinação provavelmente fornecerá o melhor ajuste). Os resultados das operações são retornados na interface do aplicativo. A otimização acontece pela *engine* do aplicativo Opp!, integrada ao código da aplicação OCM. O OCM, apesar de usar parte do código do OPP, é uma aplicação completamente independente.

O aplicativo OCM está disponível para download. O endereço se encontra no Apêndice A iv.

Capítulo 6

Resultados e comentários

Nesse capítulo são apresentados os principais resultados do estudo.

6.1 Treinamento das Redes Neurais Artificiais - *Weibull*

Os treinamentos das RNAs foram realizados conforme o item 5.2. Diversas RNAs foram treinadas, com diferentes configurações, destacando-se a distribuição *Weibull*: RNA *Weibull* 38 por apresentar os menores erros de treinamento e validação. A Tabela 6.1 apresenta todos os parâmetros de configuração de treinamento das RNA.

Tabela 6.1: Parâmetros de configuração de treinamento: RNA *Weibull* 38
Fonte: Autor, 2021.

Parâmetro	Valor
Tamanho do BD	5000
Proporção Treinamento/Total	80%
Tamanho da 1 camada oculta	20
Tamanho da 2 camada oculta	20
Número de epoch	10000
Tamanho do Batch	10
η_0	1e-3
d_r	1e-2
d_s	1000
Função de erro	MAPE

Devido ao rearranjo da ordem das distribuições nos BD, não é possível avaliar qual porcentagem de cada BD participou do treinamento, sendo 80% a porcentagem utilizada total. Não se espera que a proporção se repita dentro dos BD menores. A Figura 6.1 apresenta os erros por *epoch* ao longo do treinamento.

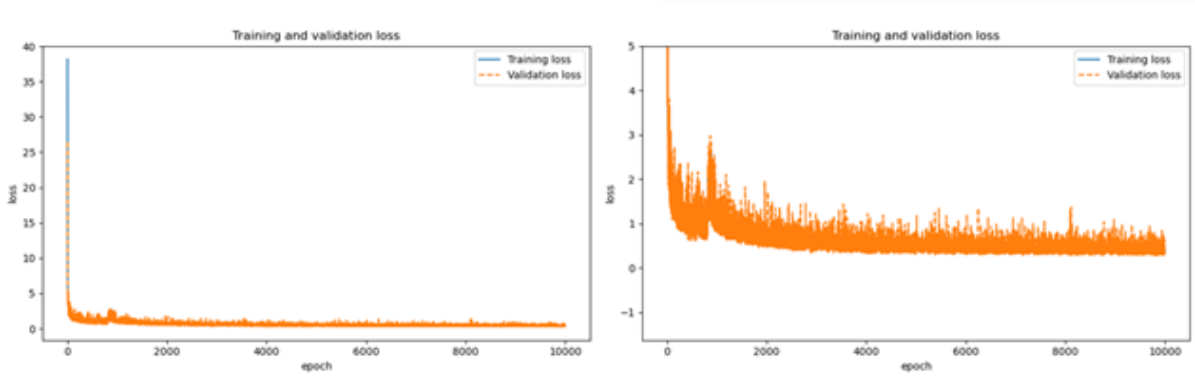

Figura 6.1: Erro por epoch - Treinamento RNA *Weibull* 38
Fonte: Autor,2021

Os gráficos da Figura 6.1 ilustram o decaimento do erro de treinamento ao longo da execução dos *epochs*. O gráfico esquerdo da Figura 6.1 apresenta os erros ao longo de todo o treinamento, com limites do eixo vertical ajustados automaticamente, e o gráfico da direita apresenta erros inferiores a 5%.

Ao final do treinamento, o erro de treinamento foi de 0,3867% enquanto para os dados de validação foi de 0,4663%. A proximidade de erros para ambos os dados sugere um bom ajuste do modelo. A Figura 6.2 contém gráfico do erro para todo o BD.

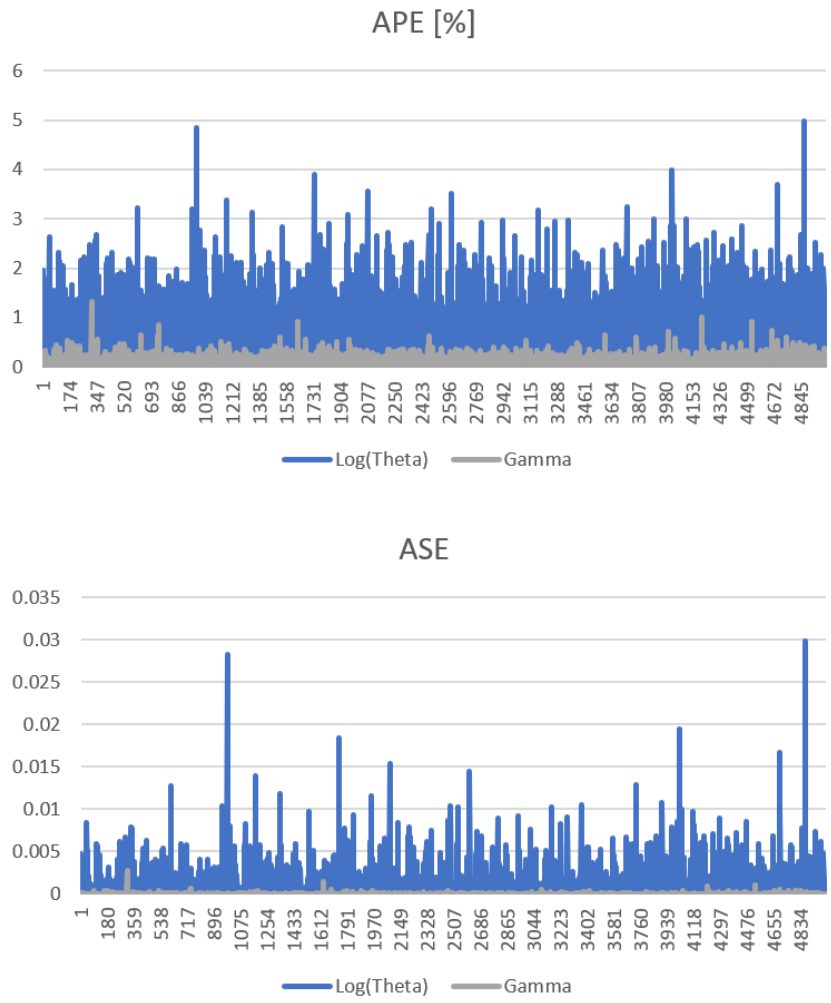

Figura 6.2: Ajuste do BD Total pelo treinamento RNA *Weibull* 38
Fonte: Autor,2021.

Pela Figura 6.2, a maioria dos dados modelados apresentou erros inferiores a 2% ou em ASE, inferiores a 0,005. Na média, esse BD apresentou erro MAPE de 0,4195% e MASE de 0,3115e-3. O MAPE para os dados novos foi de 0,4263% e MASE de 0,3540e-3, confirmando um bom ajuste. Principalmente quando comparado ao trabalho de Liu (1995), na qual, em trabalho similar, o autor encontrou um erro máximo de 10,2% para o $\log(\theta)$ e MAPE de 2,7%. Liu (1995) utilizou um BD com 600 distribuições para treinamento e 24 (proporção de treinamento de 96,15%) para teste. Como nesse trabalho, Liu também utilizou 5 valores de TTF vetor.

Para efeitos de comparação, a Tabela 6.2 apresenta as distribuições nas quais a avaliação pelo modelo RNA *Weibull* 38 dos parâmetros da função apresentou os maiores erros.

Tabela 6.2: Distribuições com maiores erros na avaliação por RNA *Weibull* 38

Fonte: Autor, 2021.

N	T1 [h]	T2 [h]	T3 [h]	T4 [h]	T5 [h]	$\log(\theta)$	γ
1	25,4872	1952,9150	3381,7910	3794,4110	7173,5769	3,4742	1,0000
2	25,2078	1931,5090	3344,7230	3752,8210	7094,9483	3,4695	1,0000
3	97,1280	2441,6100	2812,6070	5333,5710	6164,2170	3,4959	1,0000
4	1037,1370	2492,5130	2667,3670	3172,4460	8429,0507	3,4748	1,0000
5	384,1971	396,3390	3140,6190	3832,3930	4222,9175	3,4936	1,5000
6	22,5895	1730,8850	2997,3100	3363,0190	6358,0032	3,4218	1,0000
7	379,7368	391,7377	3104,1580	3787,9010	4173,8919	3,4885	1,5000
8	134,6986	141,1342	3148,1470	4243,6260	4908,5253	3,4929	1,0000
9	535,7499	1287,5480	1377,8720	1638,7780	4354,1620	3,1879	1,0000
10	547,4190	1315,5921	1407,8831	1674,4724	4449,0001	3,1973	1,0000

Na Tabela 6.2, T1 a T5 contém os tempos TTF de cada um dos 10 vetores características e $Log(\theta)$ e γ apresentam os valores reais dos parâmetros.

A Tabela 6.3 apresenta os parâmetros $Log(\theta)$ e γ dos vetores características contidas na Tabela 6.2, mas agora ajustados pela RNA *Weibull* 38 e os respectivos MAPE e MASE.

Tabela 6.3: Ajuste pela RNA *Weibull* 38 das distribuições com maiores erros

Fonte: Autor, 2021.

N	$Log(\theta) - Modelo$	γ - Modelo	MAPE [%]	MASE [e-2]
1	3,3014	1,0010	2,5382	1,4936
2	3,3014	1,0010	2,4727	1,4121
3	3,3562	0,9995	2,0223	0,9752
4	3,3391	1,0010	2,0025	0,9205
5	3,3644	1,4997	1,8596	0,8346
6	3,3014	1,0010	1,8107	0,7253
7	3,3644	1,4997	1,7897	0,7705
8	3,3747	1,0011	1,7502	0,6992
9	3,2900	1,0010	1,6532	0,5216
10	3,2993	1,0010	1,6462	0,5202

Os dados contidos na Tabela 6.3 mostram que dentre os 10 maiores erros, apresentaram MAPE entre um mínimo de 1,6462% e um máximo de 2,5382%. Para avaliar esses erros, essas mesmas 10 distribuições foram apresentadas ao programa ProConf 2000

(2009). Os dados foram ajustados segundo o modelo de Estimativa da Verossimilhança Máxima e a Tabela 6.4 apresenta os resultados encontrados.

Tabela 6.4: Piores ajustes da RNA *Weibull* 38 avaliados pelo ProConf 2000
Fonte: Autor, 2021.

N	$Log(\theta)$-ProConf	γ - ProConf	MAPE [%]	MASE [e-2]
1	3,4930	0,8615	7,1944	0,9766
2	3,4882	0,8615	7,1947	0,9766
3	3,5410	1,1225	6,7697	0,8519
4	3,6008	1,5295	28,2887	14,8129
5	3,4047	1,2125	10,8548	4,5275
6	3,4405	0,8615	7,1985	0,9766
7	3,3997	1,2125	10,8567	4,5275
8	3,3597	0,8850	7,6570	1,5486
9	3,3140	1,5295	28,4520	14,8129
10	3,3233	1,5295	28,4462	14,8129

Pelos dados da Tabela 6.4, as mesmas distribuições de TTF, quando avaliadas por Estimativa da Verossimilhança Máxima pelo software ProConf 2000, apresentam MAPE e MASE muito superiores ao modelo RNA *Weibull* 38, estando estes entre 6,7697% e 28,4520% para MAPE e 0,8519 e 14,8129 para MASE. Esses dados reforçam a dificuldade em ajustar curvas a distribuições com poucos dados (TTF) e reafirmam a possibilidade de utilização de modelos RNA para esse fim, como já defendido pelos autores Liu (1995) e indiretamente Assis et al (2020) em suas obras. A Tabela 6.5 contém os dados de tempo das dez (10) distribuições mais bem avaliadas pela RNA *Weibull* 38.

Tabela 6.5: Distribuições com menores erros na avaliação por RNA *Weibull* 38

Fonte: Autor, 2021.

N	T1 [h]	T2 [h]	T3 [h]	T4 [h]	T5 [h]	$Log(\theta)$	γ
1	262,5005	953,3728	1008,872	1303,165	1380,84	3,0225	2,5000
2	261,7424	950,6196	1005,958	1299,402	1376,853	3,0213	2,5000
3	387,0979	2195,661	2734,958	2863,846	3694,76	3,4150	2,5000
4	277,4936	1007,826	1066,495	1377,597	1459,709	3,0467	2,5000
5	277,2894	1007,085	1065,71	1376,584	1458,635	3,0463	2,5000
6	260,5678	946,3537	1001,444	1293,571	1370,674	3,0193	2,5000
7	273,1693	992,1209	1049,875	1356,13	1436,962	3,0398	2,5000
8	6,262418	26,59882	31,94113	33,19068	41,04056	1,4860	3,0000
9	51,67782	52,89794	249,8331	290,0622	311,9594	2,3951	2,0000
10	258,1593	937,606	992,187	1281,614	1358,004	3,0153	2,5000

Na Tabela 6.5, os parâmetros $log(\theta)$ e γ são os parâmetros reais. A Tabela 6.6 apresenta os parâmetros ajustados pela RNA *Weibull* 38 e os respectivos MAPE e MASE.

Tabela 6.6: Avaliação pela RNA *Weibull* 38 das distribuições com menores erros

Fonte: Autor, 2021.

N	$Log(\theta)$-Modelo	γ - Modelo	MAPE [e-3%]	MASE [e-8]
1	3,0225	2,4998	4,8901	2,4738
2	3,0212	2,4998	5,7924	2,8036
3	3,4148	2,5002	6,5527	3,9902
4	3,0468	2,4998	6,5844	3,2696
5	3,0465	2,4998	6,7200	3,4185
6	3,0192	2,4998	7,3287	3,9907
7	3,0401	2,4998	8,3816	5,6392
8	1,4862	3,0001	8,5232	2,6059
9	2,3953	2,0002	10,7545	5,5949
10	3,0149	2,4998	10,9889	10,0435

Pelos dados da Tabela 6.6, os dez (10) menores MAPE ficam entre 4,8901e-3% e 10,9889e-3%. Recomenda-se conferir os dados completos no Apêndice C. Destaca-se também que as distribuições 7 e 8 são as únicas que não participaram do treinamento. A Tabela 6.7 apresenta os resultados do ajuste por RNA *Weibull* 38 de cada um dos bancos de dados separados.

Tabela 6.7: Modelagem para bancos de dados segundo RNA *Weibull* 38

Fonte: Autor, 2021.

BD	MAPE [%]	MASE [e-3]
1	0,3805	0,2815
2	0,4948	0,3511
3	0,3102	0,2081
4	0,4445	0,2878
5	0,4678	0,4288

Pelos dados da Tabela 6.7, a avaliação de todos os bancos de dados apresentou desempenho simular em ambos os tipos de erro.

6.2 Ajuste a distribuição *Lognormal*

O ajuste a curva *Lognormal* ocorreu conforme definido em 5.3. Os TTF e os respectivos parâmetros estão dispostos na Tabela 6.8.

Tabela 6.8: Ajuste a distribuição *Lognormal*

Fonte: Autor, 2021.

N	T1 [h]	T2 [h]	T3 [h]	T4 [h]	T5 [h]	μ	σ
4319	175,4322	285,7774	412,8521	430,0948	763,0729	5,9094	0,4862
4095	17,37235	21,46993	25,78649	27,77582	28,66188	3,1702	0,1868
1890	232,1389	247,7431	262,2271	270,326	271,6088	5,5466	0,0595

Na Tabela 6.8, os índices da primeira coluna seguem a numeração do BD (disponível no Apêndice B).

Os parâmetros μ e σ são os parâmetros ajustados. Devido ao pequeno tamanho de cada amostra, existe uma divergência quanto aos valores esperados. Essa divergência é causada pela geração dos dados. Quanto menor o conjunto de dados gerados, maior será a diferença para os valores esperados para esses parâmetros. Devido a isso, essa diferença não constitui um erro e por isso não há comparação entre os parâmetros esperados e os encontrados.

6.3 Indicadores de manutenção industrial - *Weibull*

Com base nos dados dos parâmetros da distribuição θ e γ ajustados pela RNA *Weibull* 38, foram calculados os indicadores de manutenção industrial considerados nesse trabalho.

O custo total de manutenção estimado para uma vida de serviço foi calculado com a Equação (5.13), os dados da Tabela 6.5 e com os custos tabelados de operação e indidente, dispostos na Tabela 6.9.

Tabela 6.9: Valores para os parâmetros de custo

Fonte: Autor, 2021.

Parâmetro	Descrição	Valor
C_{inc}	Custo por incidente	R$ 10.000,00
C_m	Custo por reparo programado	R$ 1.000,00
C_r	Custo por reparo não programado	R$ 2.500,00
T_{ser}	Tempo total de serviço esperado	10 anos (87600 horas)

Os parâmetros de custo contidos na Tabela 6.9 são fictícios e são usados apenas para construção do problema. A Figura 6.3 contém os gráficos de custo estimado dos vetores características 8, 9 e 10 da Tabela 6.5, calculados com os parâmetros de custo da Tabela 6.9. Esses vetores características foram escolhidos por terem as maiores diferenças de intervalos de TTF entre elas.

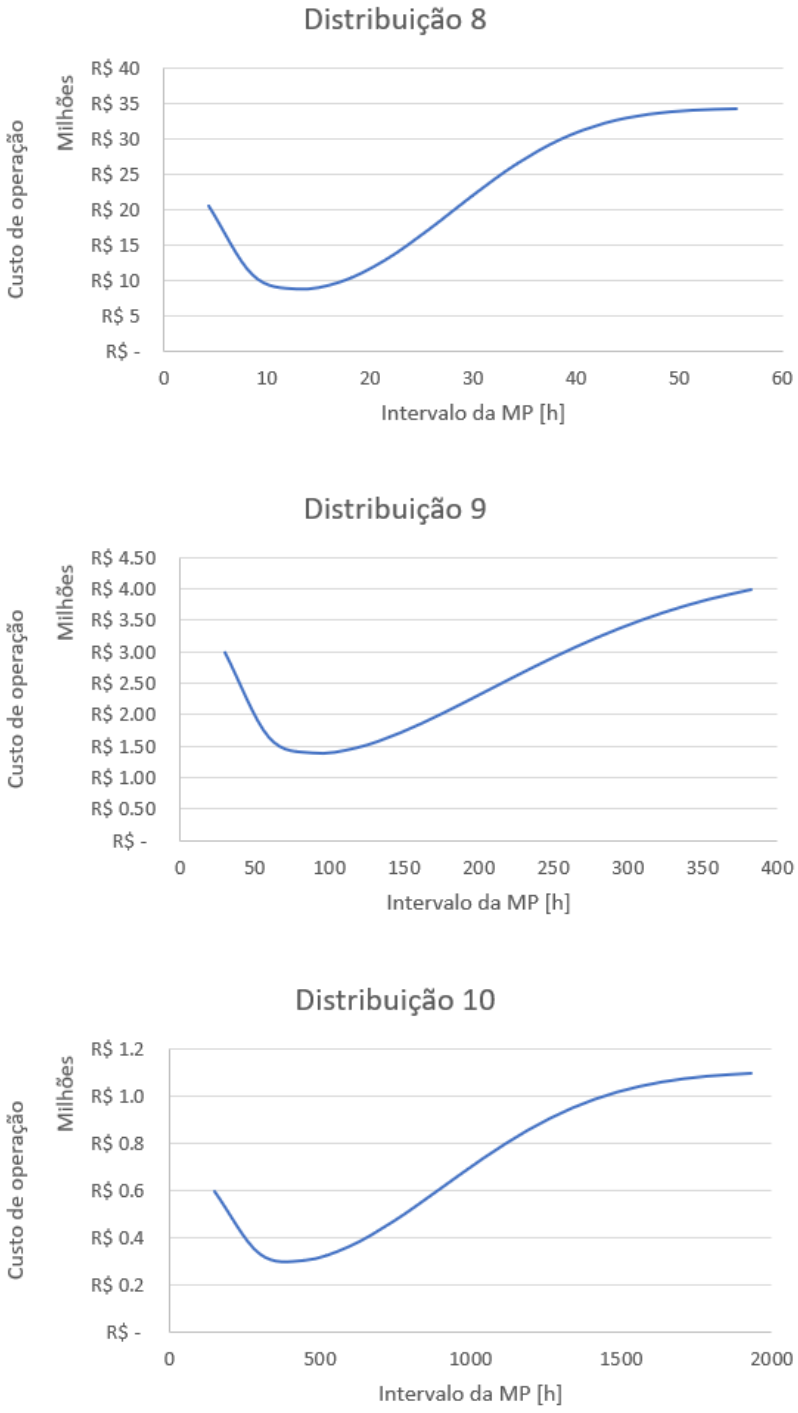

Figura 6.3: Custo total estimado - distribuição 8, 9 e 10.
Fonte: Autor,2021.

A Figura 6.3 ilustra a modelagem da estimativa de custo para uma vida de serviço de um equipamento genérico e fictício, com dados de reparo e manutenção fictícias.

É possível observar que existe uma região de vale, onde os intervalos de MP implicam em uma estimativa de custo ótimo, ou seja, mínima. Isso se deve ao fato da Equação

(5.12) modelar um maior custo para reparos antecedido por quebras não programadas. Quanto maior a confiabilidade, menores as chances do equipamento quebrar entre uma operação de manutenção e outra, evitando custos de reparos e de incidente, que são mais elevados (Tabela 6.9), ao mesmo tempo que, um intervalo muito pequeno entre operações de manutenção implicam muitas operações de MP, simulando trocas desnecessárias.

Para a modelagem da disponibilidade, Equação (5.14), a Tabela 6.10 contém os dados de tempo de reparo, tempo necessário para reparar um equipamento que saiu do estado operando inesperadamente e tempo de manutenção, operação de MP programada. Devido ao caráter não programado, os tempos de reparo foram modelados com uma penalização sobre os tempos de manutenção. Ambos são fictícios e não representa uma operação ou equipamento real.

Tabela 6.10: Tempos de manutenção e reparo - distribuição 8, 9 e 10
Fonte: Autor, 2021.

Distribuição	T_m [h]	T_r [h]
8	0,5	2
9	6	12
10	4	20

Os tempos de manutenção e reparo da Tabela 6.10 foram calculados de acordo com MTTF das respectivas distribuições, não superando 10% desse valor, para cada um dos vetores características. A Figura 6.4 contém os gráficos da disponibilidade pelo intervalo de manutenção.

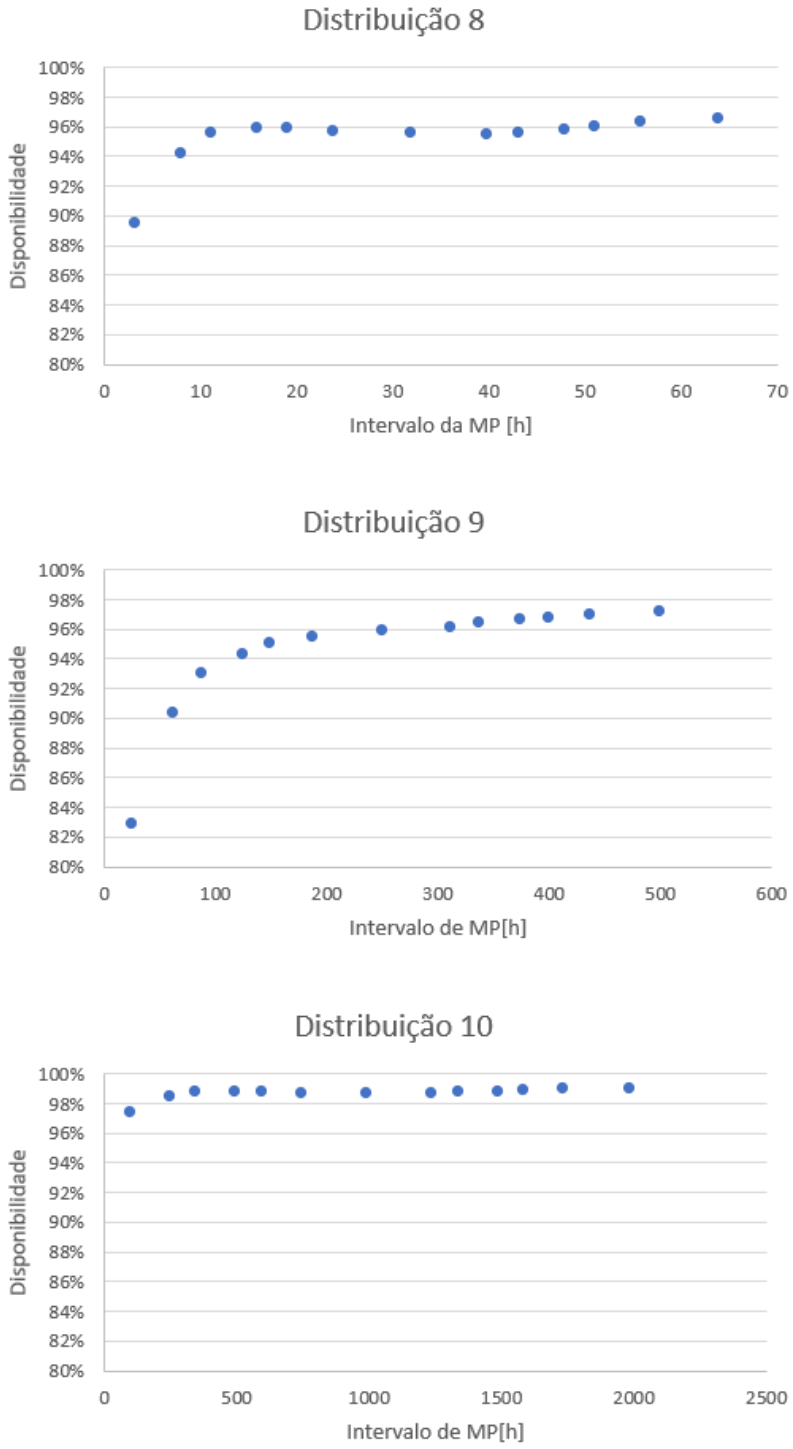

Figura 6.4: Disponibilidade - distribuição 8, 9 e 10.
Fonte: Autor,2021.

Destaca-se nos gráficos da Figura 6.4, que a disponibilidade está sendo calculada no intervalo entre duas operações de MP e não em relação a vida de serviço do equipamento.

Observa-se que a disponibilidade cresce com o aumento do intervalo entre manuten-

ções, mas é possível observar uma pequena depressão ao centro dos gráficos distribuição 8 e 10. Esses vales ocorrem pela relação entre t_m e t_r.

A Figura 6.5 contém os gráficos de confiabilidade para as distribuições 8, 9 e 10 da Tabela 6.5.

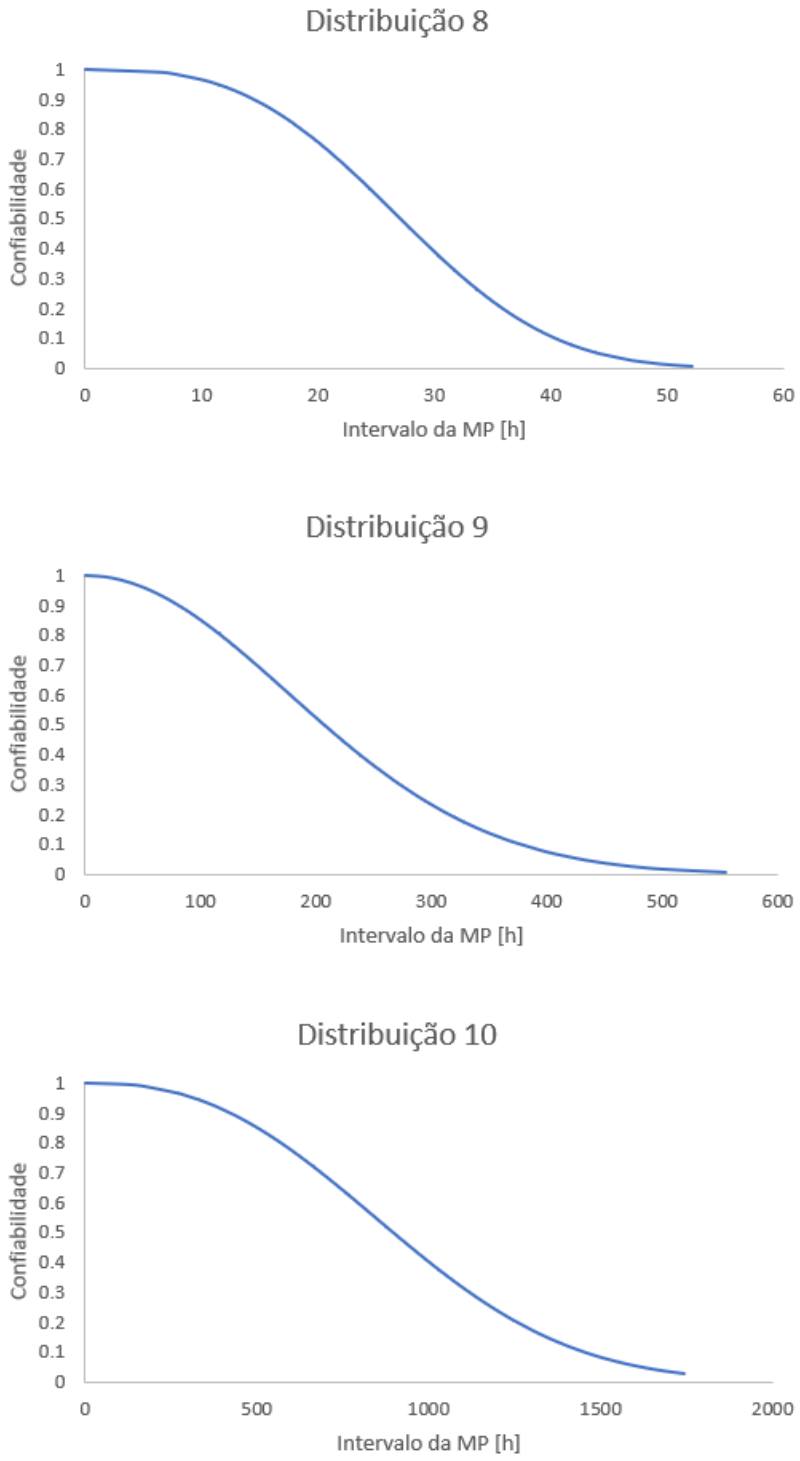

Figura 6.5: Confiabilidade - distribuição 8, 9 e 10.
Fonte: Autor,2021.

Os gráficos da Figura 6.5 representam a confiabilidade pelo intervalo de operação de MP. Os dados foram calculados por meio dos parâmetros θ e λ estimados, da Tabela 6.5.

6.4 Indicadores de manutenção industrial – *Lognormal*

Com base nos dados da Tabela 6.8, foram calculados os indicadores de manutenção industriais para as curvas de distribuição *Lognormal*, de modo análogo ao item 6.3. As funções de custo e disponibilidade foram realizadas com as mesmas modelagens do item 6.3. Os custos totais de operação também seguiram a Tabela 6.10.

As Figuras 6.6 contém os gráficos para os custos de operação para os vetores 4319, 4095 e 1890.

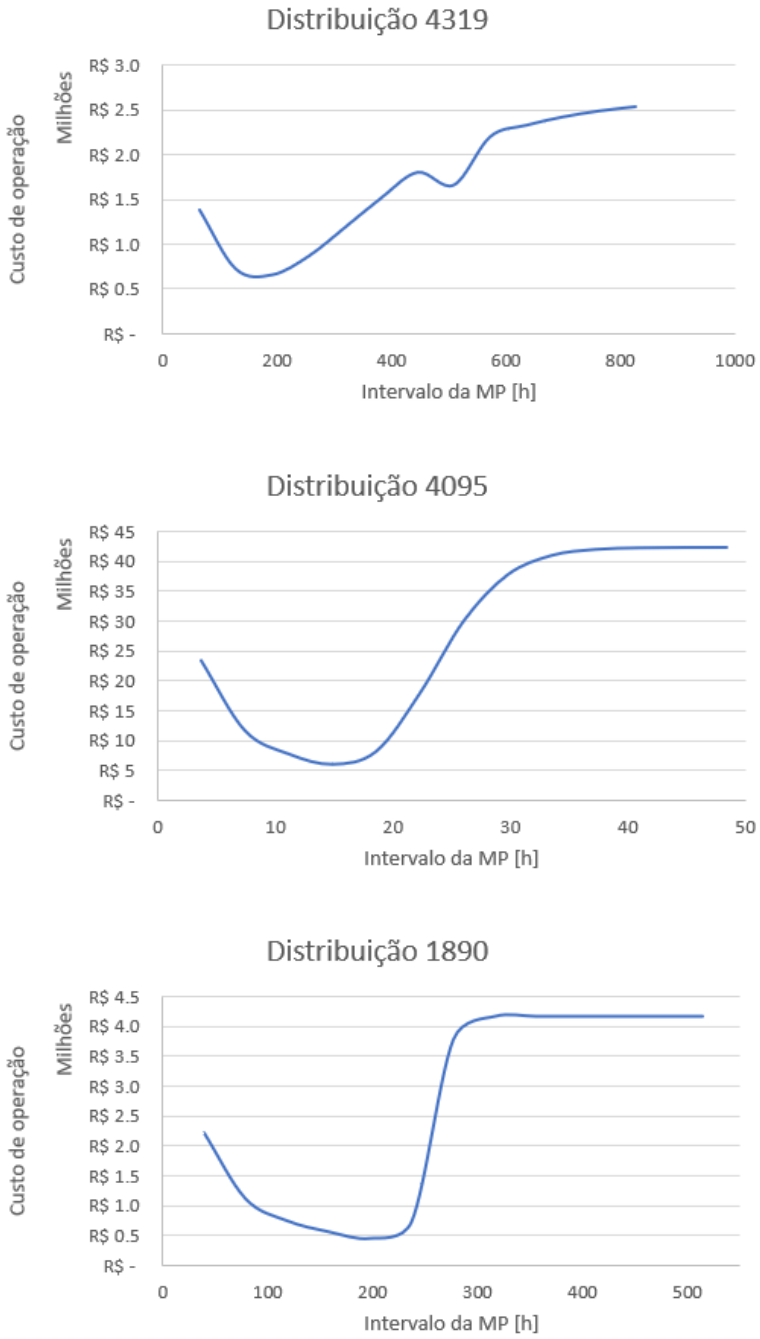

Figura 6.6: Custo total estimado - distribuição 4319, 4095 e 1890.
Fonte: Autor,2021.

A Figura 6.6 ilustra a modelagem da estimativa de custo para uma vida de serviço de um equipamento genérico e fictício, com dados de reparo e manutenção fictícias. É possível perceber em todos os gráficos que existe uma região de ótimo global, na qual o intervalo ente MP acarretará o menor custo de operação ao longo da vida de serviço do sistema, muito simular ao expresso nos gráficos da Figura 6.3, para a distribuição *Weibull*.

Nota-se também, três comportamentos diferentes. Um destaque que se faz é quanto ao gráfico da distribuição 4319. Na região nas proximidades do intervalo de 500 horas, existe um pequeno vale, indicando um mínimo local, como definido na Equação (2.3).

A distribuição 4095 assume um formato mais parecido com os gráficos da distribuição *Weibull*, da Figura 6.3. O gráfico da distribuição 1890 se parece com o da distribuição 4095, mas com um crescimento do custo a direita mais intenso.

A modelagem da disponibilidade fez uso dos dados da Tabela 6.11.

Tabela 6.11: Tempos de manutenção e reparo - distribuição 4319, 4095 e 1890
Fonte: Autor, 2021.

Distribuição	T_m [h]	T_r [h]
4319	3	5
4095	0,5	2
1890	2	4

Os tempos de manutenção e reparo da Tabela 6.11 foram calculados com os MTTF das respectivas distribuições, nunca sendo superiores a 10% desse valor. A Figura 6.7 contém os gráficos da disponibilidade pelo intervalo de MP.

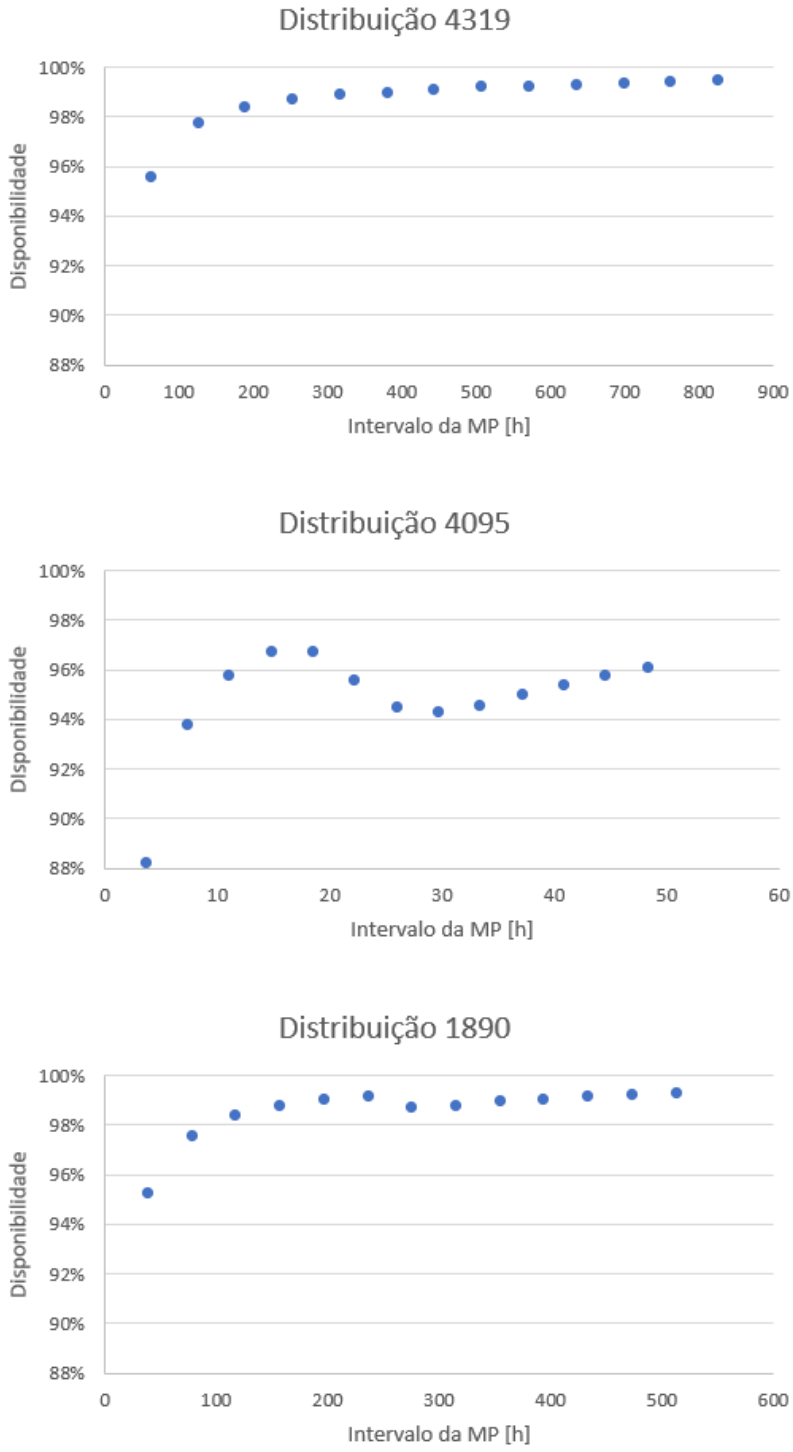

Figura 6.7: Disponibilidade - distribuição 4319, 4095 e 1890.
Fonte: Autor,2021.

De modo similar aos gráficos da Figura 6.4, os gráficos da Figura 6.7 mostram um crescimento da disponibilidade com o intervalo de MP, sendo que nos gráficos das distribuições 4095 e 1890, existem depressões nas regiões próximas a 30 horas e 300 horas,

respectivamente. Isso não foi observado para o gráfico da distribuição 4319.

A Figura 6.8 contém os gráficos de confiabilidade para as distribuições 4319, 4095 e 1890.

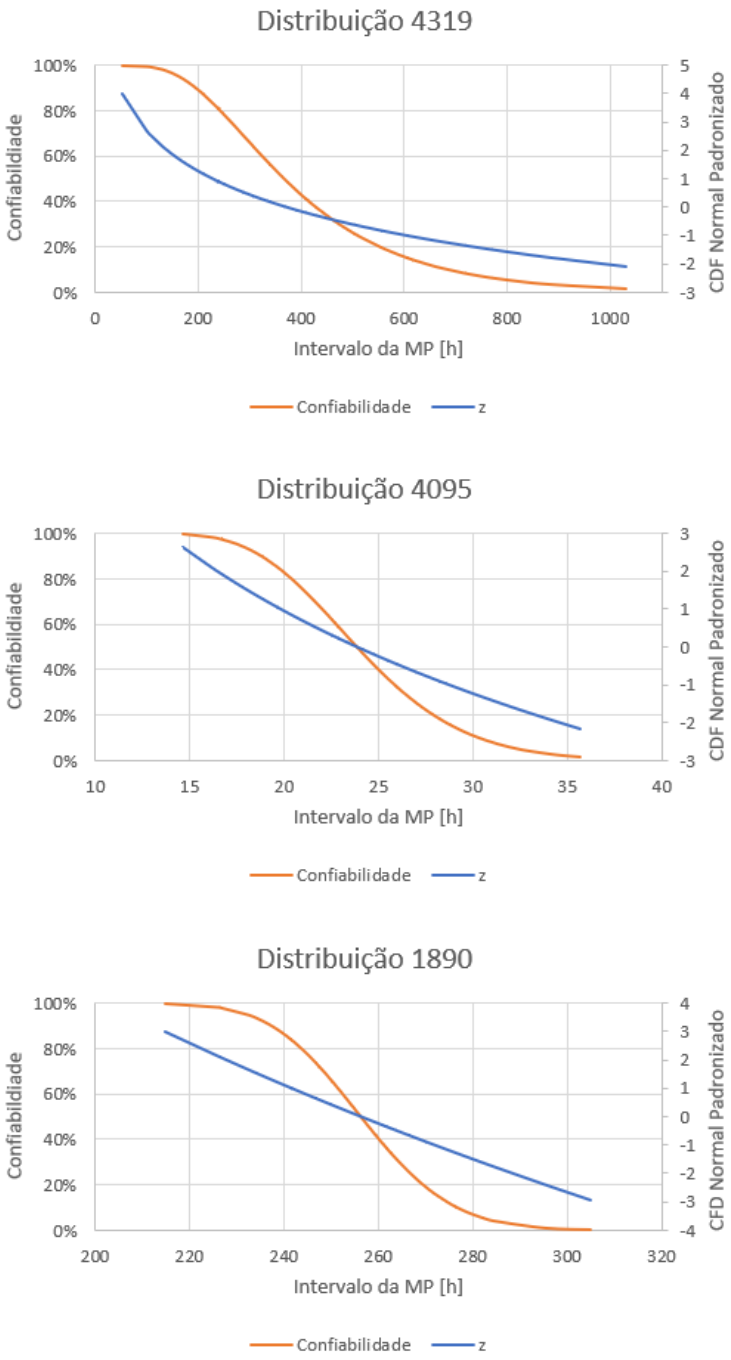

Figura 6.8: Confiabilidade - distribuições 4319, 4095 e 1890.
Fonte: Autor,2021.

Os gráficos da Figura 6.8 representam a confiabilidade para um dado intervalo de operação de MP. No eixo vertical direito, estão expostos os valores para a CDF normal

padronizado e no eixo vertical esquerdo estão expostos os valores para a confiabilidade.

Para um dado valor de z, com a interpolação dos dados da tabela do Anexo 1, é possível encontrar um valor aproximado para a confiabilidade. Os valores do gráfico da Figura 6.8 foram obtidos pela aproximação da Equação (4.64) e a condição (4.65). Como era esperado, e similar ao observado nos gráficos da Figura 6.5, houve um rápido declínio da confiabilidade com o aumento do intervalo da MP.

6.5 Otimização

A otimização da função objetiva 5.13 ocorreu por meio do aplicativo Opp! (item 5.4). Diversos parâmetros de configuração foram utilizados. Os parâmetros que obtiveram os melhores resultados estão dispostos na Tabela 6.12. Duas regras de paradas foram utilizadas, número de iterações máximo e tolerância, como definido pela Equação (2.12).

Tabela 6.12: Valores dos parâmetros de configuração PSO
Fonte: Autor, 2021.

Parâmetro	Valor
Número de iterações	100
Número de partículas	600
Tolerância	1E-40
Limite de busca Distribuição 8	(0,270)
Limite de busca Distribuição 9	(0,1910)
Limite de busca Distribuição 10	(0,9650)
Limite de busca Distribuição 4319	(0, 4130)
Limite de busca Distribuição 4095	(0, 240)
Limite de busca Distribuição 1890	(0, 2560)
PSO w, c_1 e c_2 (2.21)	0,5; 1,2; 1,2

Os dados dispostos na Tabela 6.12 foram utilizados para configurar o PSO nas operações de otimização. Estes não foram escolhidos com o objetivo de apresentarem menor tempo de execução ou menor custo computacional. Foram escolhidos mediantes testes, de forma a confiavelmente apresentar os resultados ótimos globais em todas as execuções, não necessitando configuração individual para cada caso analisado, com exceção do parâmetro Limite de Busca.

Na grande maioria dos casos, essa configuração utilizou recursos de forma exagerada,

mas o tempo de execução médio foi de 0,8188, 0,8822 e 0,9169 segundos para as distribuições *Weibull* 8, 9 e 10, respectivamente e 1,2586, 1,1265 e 1,3103 para as distribuições *Lognormal* 4319, 4095 e 1890, respectivamente.

O aumento da média de tempos entre uma mesma distribuição é reflexo do gradativo aumento do espaço de busca, estes foram escolhidos pelo truncamento sem casa decimal do MTTF, multiplicado por uma dezena. O aumento do tempo de execução médio entre as distribuições *Weibull* e *Lognormal* se devem as funções envolvidas na otimização. O cálculo da aproximação do CDF Normal padronizado (Equação 5.64 e 5.65) se mostrou mais intenso que o cálculo da confiabilidade CCDF *Weibull* (Equação 4.10 e 4.54).

Em todos os casos, o algoritmo de otimização sugeriu tempo do intervalo de MP t com as devidas casas decimais, mas a função objetiva e de restrição avaliaram apenas o truncamento deste valor, sem casa decimal. Todos os cálculos de indicadores de manutenção industrial utilizaram tempos inteiros.

A Tabela 6.13 contém os resultados encontrados para a Distribuição 8.

Tabela 6.13: Resultados - Otimização - Distribuição 8

Fonte: Autor, 2021.

Restrição	Custo [R$]	Intervalo [h]	Confiabilidade	Disponibilidade	Tempo [s]
99,0000%	39.378.967,9974	201,8592	0,0000%	99,0148%	0,6670
98,0000%	39.378.967,9974	98,3202	0,0000%	98,0000%	0,6962
97,0000%	39.376.263,4510	65,2117	0,0071%	97,0151%	0,7832
96,0000%	38.576.751,0185	48,6184	2,1362%	96,0616%	0,7873
95,0000%	9.139.157,7578	13,4015	92,6450%	95,5157%	0,8578
90,0000%	9.139.157,7578	13,0689	92,6450%	95,5157%	0,8767
85,0000%	9.139.157,7578	13,0273	92,6450%	95,5157%	0,9412
80,0000%	9.139.157,7578	13,4919	92,6450%	95,5157%	0,9412

A análise dos resultados na Tabela 6.13, e principalmente da Figura 6.9, mostram que a função custo de operação é descontínua no intervalo de tempo considerado. Esse fato confirma que a escolha de um método de otimização estocástico (item 2.2) e sequencial (item 2.41.) foi uma boa escolha.

Destaca-se também que, entre a restrição de disponibilidade de 80,0% e 95,0%, é indicado como ótimo, o mesmo intervalo de tempo, 13 horas, e consequentemente mesma confiabilidade e custo de operação. Mesmo quando foi exigido uma confiabilidade menor, de apenas 80,0%, o método indicou um intervalo ótimo que representa uma disponibilidade

de 95,5%. Isso indica que este intervalo é a posição do ótimo local e global para a função, e é justificado pela confiabilidade alta, de 92,6%. Vale relembrar que a função de custo de operação (5.13) é dependente da confiabilidade.

Ao elevar a restrição para 96,0%, um incremento na disponibilidade menor que 0,5%, o intervalo de menor custo torna-se 48 horas, e o custo de operação tem um aumento significativo de 422,1%. Isso se deve novamente a confiabilidade, que desloca de 92,6% para 2,1%, afetando diretamente o custo.

Um novo incremento na restrição de disponibilidade mínima para 97,0% aumenta o intervalo de MP para 65 horas e elevam o custo de operação em 2,1%. A disponibilidade média passa a ser 97,0%.

Definindo a disponibilidade mínima em 98,0% faz com que o custo de operação atinja o seu valor máximo. Isso ocorre porque a confiabilidade atinge o seu valor mínimo. Os tempos indicados pela otimização já não mais fazem sentido. Qualquer que seja o intervalo entre MP com a confiabilidade em 0,0%, espera-se que o sistema entre em estado não operante a cada MTTF e os intervalos de MP, muito superiores, não trarão mudança no custo e operação.

O tempo de execução também foi decrescente durante as execuções, gráfico 2 da Figura 6.9, mas não existe um motivo particular para o efeito.

O resultado aponta que o menor custo possível, mediante as configurações e valores fornecidos, é referente ao intervalo entre MP de 13 horas. Intervenções de manutenção em intervalos superiores apresentam o maior risco de quebra antes a execução da manutenção, o que implica custos de manutenção emergencial, mais caras, com custos extras de incidente e mais demoradas.

De forma inversa, valores inferiores ao ótimo trazem custos de manutenções provavelmente desnecessárias. Para o melhor desempenho financeiro da operação do equipamento, o intervalo de manutenção ótimo é o indicado. Essa mesma avaliação vale para todos os outros seis casos estudados nesse trabalho. A Figura 6.9 contém os gráficos dos resultados para a distribuição 8, contidos na Tabela 6.13.

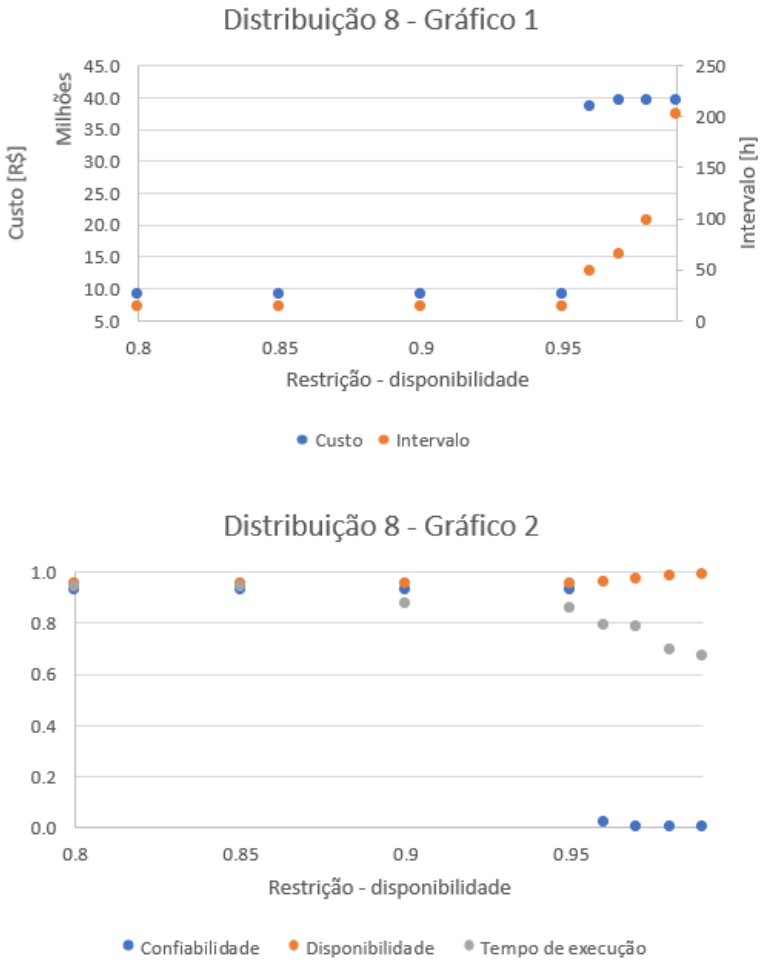

Figura 6.9: Resultados - Otimização - Distribuição 8.
Fonte: Autor,2021.

O gráfico 1 da Figura 6.9 ilustra melhor a descontinuidade do custo mediante um pequeno incremento na restrição do intervalo entre MP. A justificativa é o decréscimo da confiabilidade, que pode ser observado no gráfico 2, atingindo valor próximo ao limite mínimo.

A Tabela 6.14 contém os resultados encontrados para a Distribuição 9.

Tabela 6.14: Resultados- Otimização - Distribuição 9

Fonte: Autor, 2021.

Restrição	Custo [R$]	Intervalo [h]	Confiabilidade	Disponibilidade	Tempo [s]
99,0000%	5.724.409,9057	1188,534144	0,0000%	99,0000%	0,6837
98,0000%	5.703.393,6755	587,4775053	0,3770%	98,0004%	0,8427
97,0000%	5.097.751,7019	366,9625817	11,4248%	97,0013%	0,9433
96,0000%	3.374.136,3316	225,1069955	44,0532%	96,0075%	0,9274
95,0000%	2.138.897,6976	149,812993	69,8047%	95,0184%	0,9087
94,0000%	1.681.704,5290	111,9595794	81,9154%	94,0000%	0,8834
93,0000%	1.557.252,4861	90,90271204	87,7098%	93,0354%	0,8847
92,0000%	1.548.162,6022	83,18792259	89,4465%	92,5996%	0,9175
91,0000%	1.548.162,6022	83,88205244	89,4465%	92,5996%	0,8932
90,0000%	1.548.162,6022	83,50489943	89,4465%	92,5996%	0,9096
85,0000%	1.548.162,6022	83,90316632	89,4465%	92,5996%	0,8966
80,0000%	1.548.162,6022	83,20742507	89,4465%	92,5996%	0,8966

Pelos dados da Tabela 6.14, observa-se que os resultados da distribuição 9 tiveram um comportamento bem diferente da distribuição 8 e por isso uma maior quantidade de restrições foram testadas. Parte do comportamento diferente se deve ao incremento mais suave no intervalo entre MP indicado para cada aumento da restrição da disponibilidade mínima. Esse efeito foi causado principalmente pela menor diferença entre tempos de reparo e tempos de manutenção (Tabela 6.10).

Entre as restrições de 80,0% e 92,0%, foi encontrado um intervalo ótimo de 83 horas, disponibilidade média de 92,6% e confiabilidade de 89,4%, acompanhada de um custo de R$1.548.162,60.

Uma mudança na restrição para 93,0% provocou um pequeno incremento no intervalo de MP, para 90 horas, um decremento na confiabilidade para 87,7% e um aumento no custo de operação de 5,9%.

Uma nova alteração na restrição para 94% provocou um incremento maior no intervalo de MP, para 111 horas, um decremento maior na confiabilidade, para 81,9% e um aumento maior no custo de operação de 8,0%. Destaca-se que a magnitude do decremento da confiabilidade segue o incremento do intervalo de manutenção.

Novos incrementos na restrição de disponibilidade provocaram incrementos gradativamente maiores no intervalo de MP, decrementos gradativamente maiores na confiabilidade,

até 11,4%, e um aumento gradativamente maior nos custos de operação.

Para a restrição de 98%, o custo chegou próximo ao seu valor máximo, devido a confiabilidade estar próxima do seu limite mínimo. Com a restrição de 99,0%, a confiabilidade assume valor nulo e o custo de operação atinge seu limite máximo, de R$572.409,91.

A Figura 6.10 contém os gráficos dos resultados para a distribuição 9.

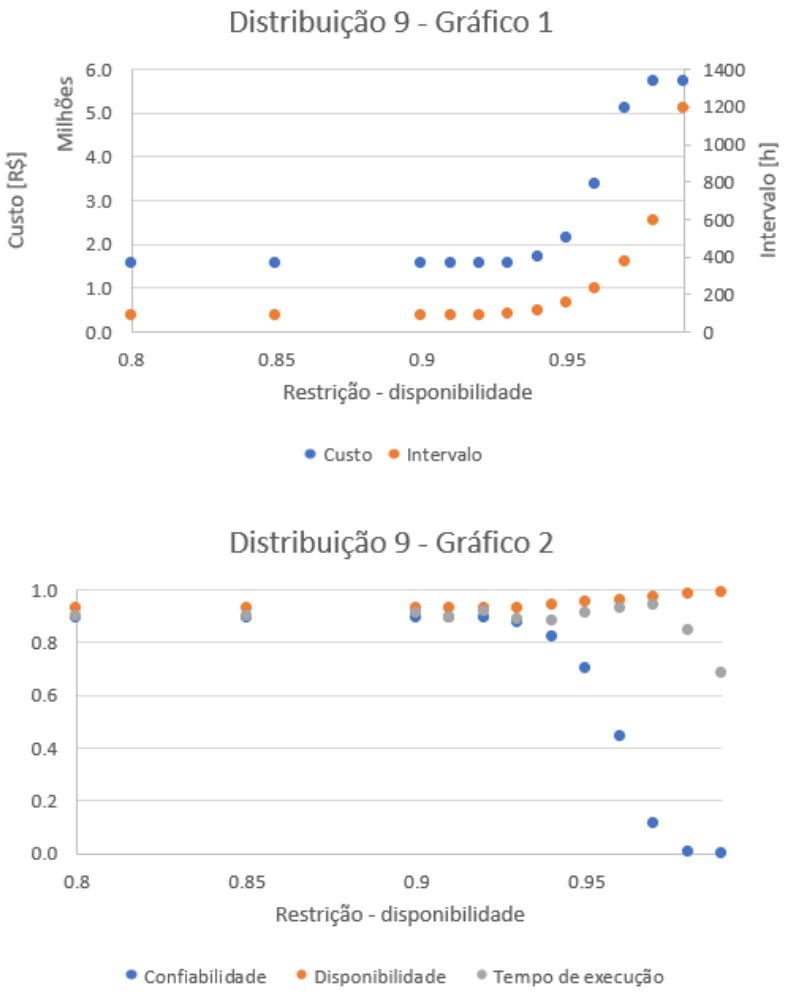

Figura 6.10: Resultados - Otimização - Distribuição 9.
Fonte: Autor,2021.

Os gráficos da Figura 6.10 auxiliam o entendimento do comportamento mais suave no incremento dos custos de operação. O aumento na restrição de disponibilidade mínima gerou um incremento menor no intervalo entre MP quanto comparados aos resultados da distribuição 8, o que promoveu uma redução menos acentuada na confiabilidade e um aumento no custo de operação menor. A confiabilidade é o fator de maior influência na modelagem do custo.

A observância do gráfico 2 da Figura 6.10 também mostra que a confiabilidade reduziu de forma vultosa entre a restrição de disponibilidade de 94,0% (confiabilidade de 87,7%) e disponibilidade de 99,0% (confiabilidade de 0,0%). Mas a distribuição 8 teve uma redução mais acentuada para um intervalo de restrição menor: disponibilidade de 95,0% (confiabilidade de 92,6%) para disponibilidade de 97,0% (confiabilidade de aproximadamente 0,0%). Esse decremento mais lento da confiabilidade se traduziu em aumento mais gradativo do custo de operação.

A Tabela 6.15 contém os resultados encontrados para a Distribuição 9. Tabela 6.15

Tabela 6.15: Resultados - Otimização - Distribuição 10

Fonte: Autor, 2021.

Restrição	Custo [R$]	Intervalo [h]	Confiabilidade	Disponibilidade	Tempo [s]
99,0000%	1.126.754,7206	1970,0630	0,6751%	99,0003%	0,9470
98,0000%	300.148,1570	403,1180	90,9704%	98,6670%	0,9067
97,0000%	300.148,1570	403,1771	90,9704%	98,6670%	0,8847
96,0000%	300.148,1570	403,7641	90,9704%	98,6670%	0,9023
95,0000%	300.148,1570	403,5565	90,9704%	98,6670%	0,9220
90,0000%	300.148,1570	403,4365	90,9704%	98,6670%	0,8833
85,0000%	300.148,1570	403,1649	90,9704%	98,6670%	0,9786
80,0000%	300.148,1570	403,1335	90,9704%	98,6670%	0,9110

Pelos dados da Tabela 6.15, pode-se observar que a distribuição 10 apresentou um comportamento parecido com a distribuição 8. Observa-se que entre restrição de disponibilidade mínima de 80,0% a 98,0%, o intervalo para MP indicado é o mesmo, 403 horas, com disponibilidade média de 98,7% e confiabilidade de 91,0%. Um incremento dessa restrição eleva o intervalo de MP para 1970 horas, reduzindo a confiabilidade para 0,7% e elevando o custo de operação em 375,4%.

Esse aumento significativo no intervalo de MP é explicado pelo gráfico da Figura 6.14. A curva de disponibilidade por intervalo de tempo é ainda menos inclinada que o da distribuição 8, necessitando de um incremento muito grande de tempo para um pequeno ganho de disponibilidade.

Como já visto anteriormente, esse cenário tem grande efeito na confiabilidade, que influencia fortemente o aumento do custo de operação.

A Figura 6.11 contém os gráficos dos resultados para a distribuição 10.

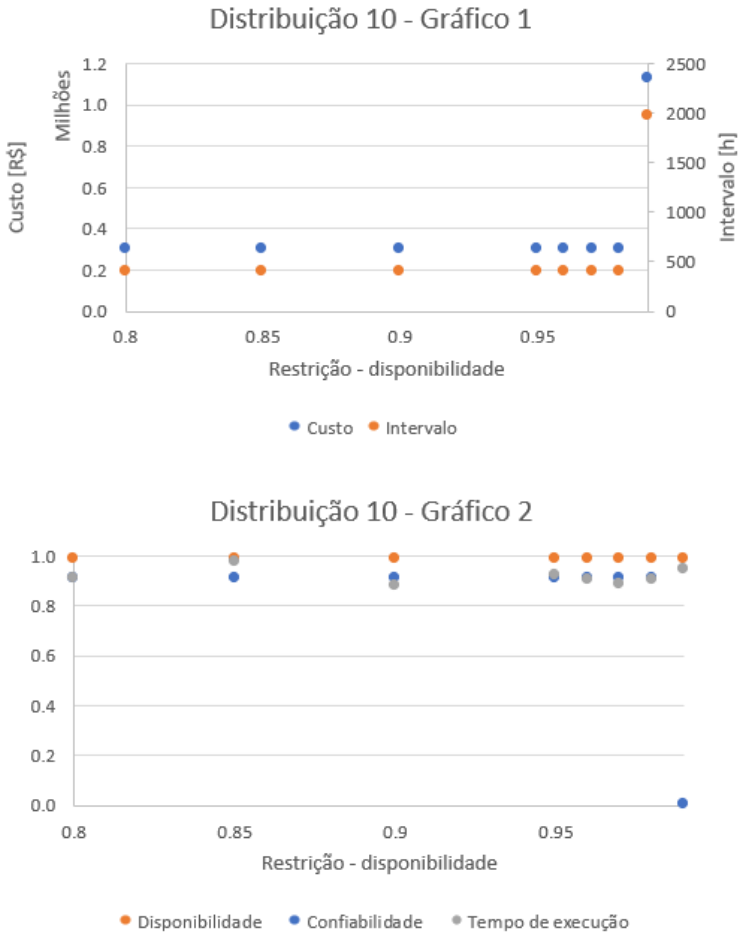

Figura 6.11: Resultados - Otimização - Distribuição 10.
Fonte: Autor,2021.

O gráfico 1 da Figura 6.11 ilustra o comportamento do custo de operação e do intervalo de MP pela restrição de disponibilidade. Pode ser observado o comportamento comentado anteriormente. Não existe qualquer mudança no intervalo de MP indicado pelo método, e consequentemente do custo de operação, até a restrição de 98,0%, sendo a confiabilidade nesse intervalo de 98,7% Para uma confiabilidade mínima de 99,0%, foi necessário um intervalo de MP de 488,8%, com um custo de operação 375,4% maior.

O gráfico 2 da Figura 6.11 ilustra a explicação para esse aumento expressivo do custo de operação. No mesmo intervalo, a confiabilidade cai de 91,0% para 0,7%, muito próxima do seu limite mínimo.

Destaca-se que, diferente das outras duas distribuições, apesar de próxima do limite, a restrição de 99,0% na disponibilidade ainda não representa o custo de operação máximo, que ocorre quando a confiabilidade é nula. Esse custo máximo é de R$ 1.103.622,60, 0,8%

maior que o custo esperado para uma restrição de 99,0%. Em uma última análise, aplicou-se a restrição de confiabilidade, estando essa limitada ao valor mínimo de 95,0%, valor usualmente encontrado na literatura especializada. A Tabela 6.16 contém os resultados encontrados para todas as distribuições.

Tabela 6.16: Resultados - Otimização - Restrição 95% de confiabilidade
Fonte: Autor, 2021.

Distribuição	8	9	10
Custo mínimo [R$]	9.139.157,7578	1.548.162,6022	300.148,1570
Custo restrito [R$]	9.384.437,6882	1.770.206,7742	320.755,6656
Intervalo [h]	11,6303	56,9166	315,4340
Confiabilidade	95,4774%	95,0501%	95,0164%
Disponibilidade	95,0912%	89,8920%	98,4999%
Tempo de execução [s]	0,6636	0,6744	0,6103

Pode-se perceber pela Tabela 6.16 que o custo de operação da otimização restrita ficou muito próximo do custo ótimo global mínimo. Isso era esperado, devido a relação de custo de operação com a confiabilidade, já bastante comentado. Ainda assim, a diferença é creditada a possíveis intervenções desnecessárias, simulando trocas de componentes ainda em bom estado.

A comparação entre as Tabelas 6.13, 6.14 e 6.15 com a Tabela 6.16 mostra pequena divergência entre os intervalos de operação de manutenção, e consequentemente dos demais índices. E apenas a distribuição 9 apresentou disponibilidade menor que 90%, estando muito próxima, em 89,9%.

Quanto a redução no tempo de execução, é creditado a retirada da função de disponibilidade do modelo. Enquanto confiabilidade é fundamental e sempre calculada, pois as demais funções a tomam como base, o mesmo não pode ser dito da função disponibilidade.

A Figura 6.12 ilustra graficamente os resultados da Tabela 6.16.

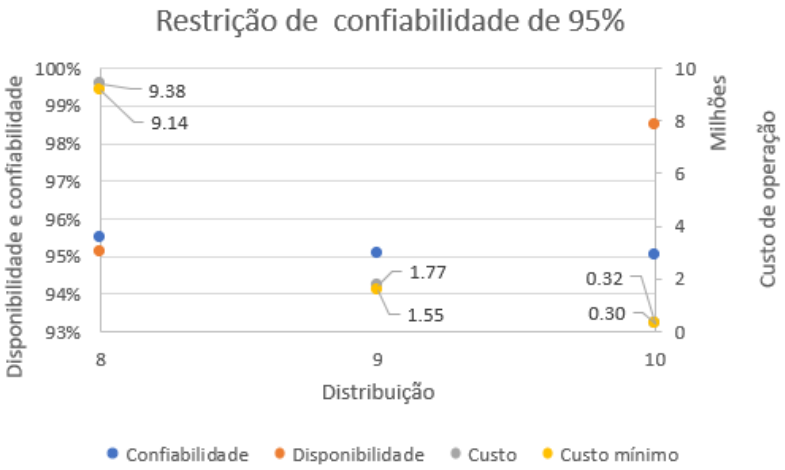

Figura 6.12: Resultados - Restrição 95% de confiabilidade – Distribuição 8,9 e 10.
Fonte: Autor,2021.

No gráfico da Figura 6.12, o eixo das coordenadas esquerdo apresenta as porcentagens para confiabilidade e disponibilidade, enquanto o eixo vertical da direita apresenta os valores em milhões de reais para custo mínimo restrito e custo mínimo irrestrito, ao longo de toda vida de serviço. A observância do gráfico torna a aproximação dos resultados para custo operacional restrito e mínimo ainda mais perceptível.

A Tabela 6.17 contém os resultados encontrados para a Distribuição 4319.

Tabela 6.17: Resultados - Otimização - Distribuição 4319
Fonte: Autor, 2021.

Restrição	Custo [R$]	Intervalo [h]	Confiabilidade	Disponibilidade	Tempo [s]
99,0000%	1.669.483,2780	416,9019	40,1571%	99,0012%	1,2126
98,0000%	635.452,1135	169,2012	94,5023%	98,1930%	1,1852
97,0000%	635.452,1135	169,8005	94,5023%	98,1930%	1,2899
96,0000%	635.452,1135	169,4694	94,5023%	98,1930%	1,2899
95,0000%	635.452,1135	169,9536	94,5023%	98,1930%	1,2488
90,0000%	635.452,1135	169,0302	94,5023%	98,1930%	1,1983
85,0000%	635.452,1135	169,2831	94,5023%	98,1930%	1,3357
80,0000%	635.452,1135	169,3211	94,5023%	98,1930%	1,3086

Pela análise da Tabela 6.17 é possível observar que o custo de operação esperado mínimo foi encontrado para uma disponibilidade média de 98,2%, para restrições de 80,0% a 98,0%, como também pode ser visto no gráfico Distribuição 4319 da Figura 6.6. A confiabilidade referente ao dado intervalo também é alta, o que, com já foi comentado,

era esperado. O tempo de execução permaneceu praticamente constante, não indicando uma maior dificuldade em encontrar o mínimo ótimo para qualquer restrição, apesar do aumento do espaço de busca.

A Figura 6.13 contém os gráficos dos resultados para a distribuição 4319 contidos na Tabela 6.17.

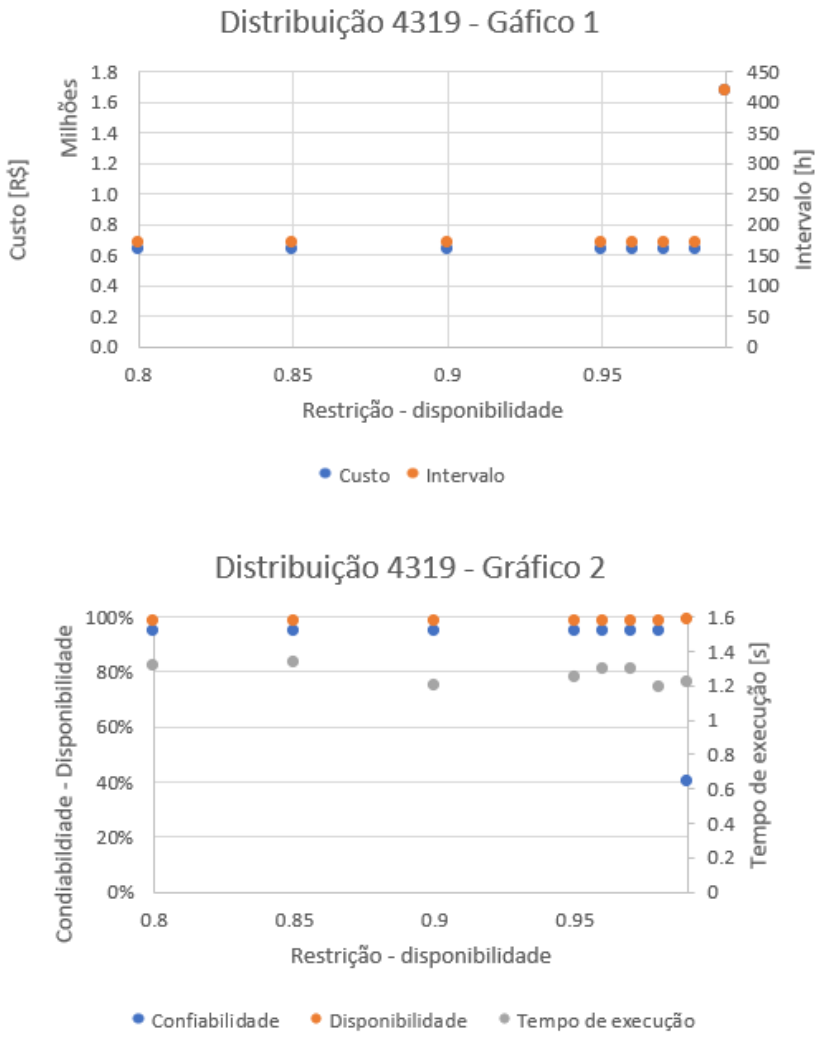

Figura 6.13: Resultados - Otimização - Distribuição 4319.
Fonte: Autor,2021.

O gráfico 1 da Figura 6.13 torna mais visível o fenômeno observado na Tabela 6.17. Para qualquer restrição de disponibilidade, exceto a de 99,0%, o intervalo da MP ótimo é o mesmo, 169 horas, indicando que é a posição do ótimo local e global.

O gráfico 2 da Figura 6.13 mostra que para uma restrição de 99,0%, onde os custos de operação disparam, a confiabilidade cai. Esse evento também era esperado, devido a

proporcionalidade inversa dessas duas grandezas.

Ainda no gráfico 2, a disponibilidade demonstra um pequeno ganho, para grandes perdas de confiabilidade e elevação dos custos de operação. Todavia, ressalta-se que os custos consideram a Tabela 6.9 e devem ser julgados cenário a cenário.

A Tabela 6.18 contém os resultados encontrados para a Distribuição 4095.

Tabela 6.18: Resultados - Otimização - Distribuição 4095

Fonte: Autor, 2021.

Restrição	Custo [R$]	Intervalo [h]	Confiabilidade	Disponibilidade	Tempo [s]
99,0000%	45.223.089,6830	236,8350	0,0000%	99,1597%	0,7381
98,0000%	45.223.089,6830	98,2800	0,0000%	98,0000%	0,9006
97,0000%	45.223.087,9571	65,2147	0,0000%	97,0149%	1,0525
96,0000%	6.107.572,2693	15,8590	99,3206%	96,7106%	1,1449
95,0000%	6.107.572,2693	15,8246	99,3206%	96,7106%	1,1747
94,0000%	6.107.572,2693	15,8056	99,3206%	96,7106%	1,1159
93,0000%	6.107.572,2693	15,2629	99,3206%	96,7106%	1,2358
92,0000%	6.107.572,2693	15,9969	99,3206%	96,7106%	1,2198
91,0000%	6.107.572,2693	15,5143	99,3206%	96,7106%	1,2015
90,0000%	6.107.572,2693	15,2176	99,3206%	96,7106%	1,2451
85,0000%	6.107.572,2693	15,0157	99,3206%	96,7106%	1,2784
80,0000%	6.107.572,2693	15,6253	99,3206%	96,7106%	1,2104

Pela Tabela 6.18, foi observado na distribuição 4095 que entre as restrições de 97,0% e 99,0%, a confiabilidade foi nula, mas para a restrição de 96,0%, observou-se que um intervalo de 15 horas e confiabilidade próxima ao valor máximo, com um custo de operação esperado ótimo global.

Isso se deve principalmente ao fato de que para atingir uma disponibilidade de 97,0%, 0,3% maior, houve um expressivo aumento do intervalo de MP de 433,3%, devido aos tempos de manutenção e reparo da Tabela 6.11.

A Figura 6.14 contém os gráficos dos resultados para a distribuição 4095

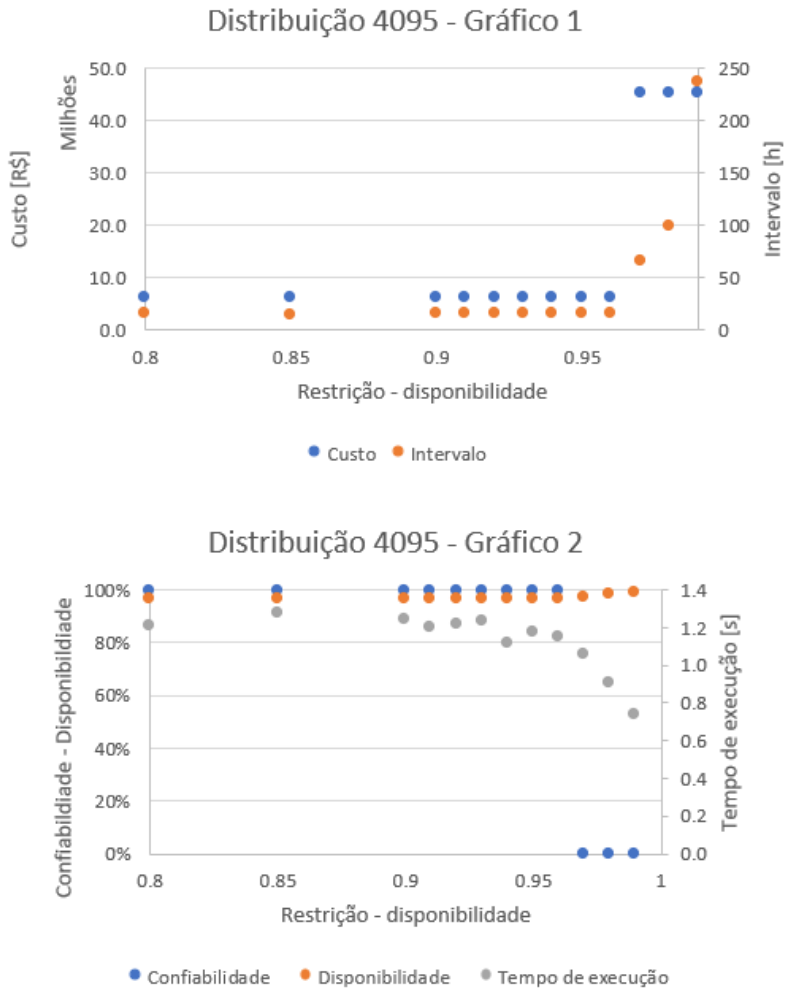

Figura 6.14: Resultados - Otimização - Distribuição 4095.
Fonte: Autor,2021.

No gráfico 1 da Figura 6.14 pode-se observar que os custos de operação não acompanham o intervalo da MP, contrário ao gráfico 1 da Figura 6.15. Para um aumento gradativo da restrição de disponibilidade, os custos de operação, para as restrições consideradas, assumem apenas duas posições: valor máximo e mínimo global.

O efeito anterior é explicado pelo gráfico 2. A modelagem de custo é restrita a operação e não considera perdas de não-produção, além das manutenções corretivas. Ao elevar a restrição de 96,0% para 97,0%, o intervalo da MP sobe consideravelmente, entrando na região de confiabilidade nula e elevando o custo de operação estimado ao valor máximo. A partir de então, não importa qual seja o intervalo de MP, o custo de produção associado ao equipamento será máximo.

A Tabela 6.19 contém os resultados encontrados para a Distribuição 1890.

Tabela 6.19: Resultados - Otimização - Distribuição 1890

Fonte: Autor, 2021.

Restrição	Custo [R$]	Intervalo [h]	Confiabilidade	Disponibilidade	Tempo [s]
99,0000%	413.349,2843	216,7745	99,7980%	99,0807%	1,1599
98,0000%	413.349,2843	216,9221	99,7980%	99,0807%	1,3267
97,0000%	413.349,2843	216,5595	99,7980%	99,0807%	1,3517
96,0000%	413.349,2843	216,2443	99,7980%	99,0807%	1,3551
95,0000%	413.349,2843	216,4479	99,7980%	99,0807%	1,2554
90,0000%	413.349,2843	216,3936	99,7980%	99,0807%	1,3052
85,0000%	413.349,2843	216,6627	99,7980%	99,0807%	1,3380
80,0000%	413.349,2843	216,0402	99,7980%	99,0807%	1,3909

A distribuição 1890, cuja resultados estão expostos na Tabela 6.19, é um caso especial. Uma alta confiabilidade e uma alta disponibilidade ocorrem ao mesmo intervalo de MP – esse é o cenário ideal. Para as configurações estudadas, o custo de operação mínimo global ocorre independente de qual seja a restrição. A baixa variância da distribuição permite alta confiabilidade para intervalos de MP elevados, próximos ao MTTF.

A Figura 6.15 contém os gráficos dos resultados para a distribuição 1890.

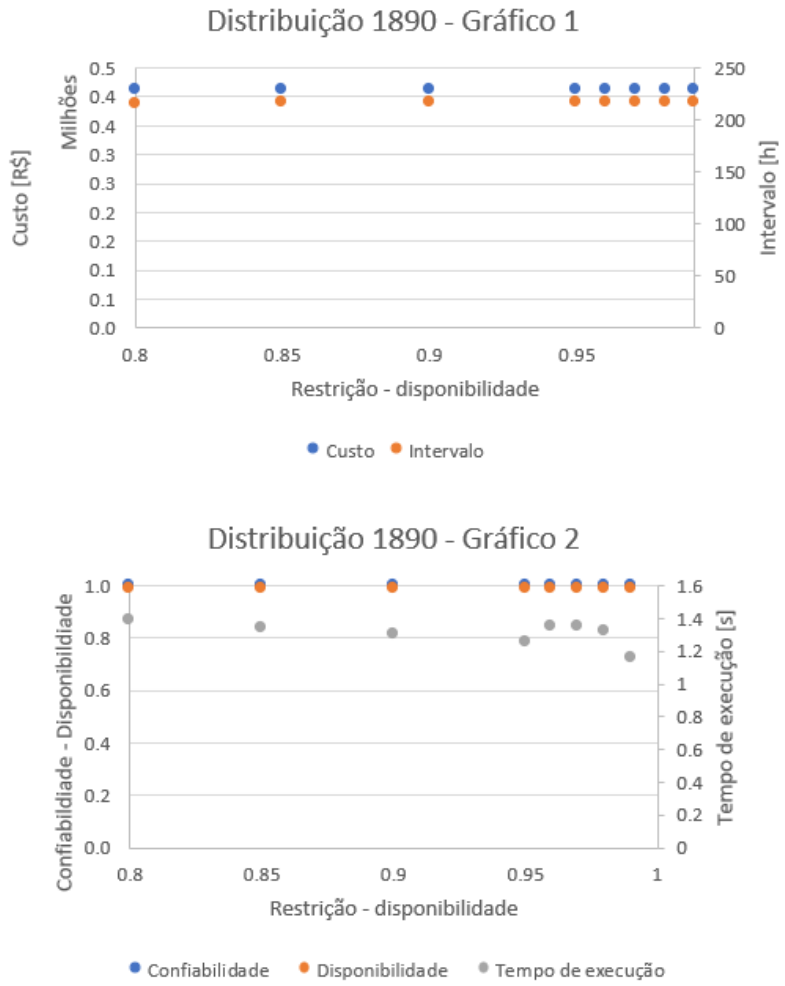

Figura 6.15: Resultados - Otimização - Distribuição 1890.
Fonte: Autor,2021.

O gráfico 1 da Figura 6.15, mostra graficamente que para qualquer restrição de disponibilidade, o intervalo da MP, e consequentemente o custo de operação, não sofrem alteração.

O gráfico 2 reafirma a observação anterior. Não há alteração na confiabilidade e disponibilidade para qualquer que seja a restrição aplicada. O tempo de execução permanece muito próximo entre as restrições.

Semelhante a comparação feita para as distribuições *Weibull*, em uma última análise das distribuições *Lognormal*, aplicou-se a restrição de confiabilidade mínima de 95,0%. A Tabela 6.15 contém os resultados obtidos.

Tabela 6.20: Resultados - Otimização - Restrição 95% de confiabilidade

Fonte: Autor, 2021.

Distribuição	4319	4095	1890
Custo mínimo [R$]	635.452,1135	6.107.572,2693	413.349,2843
Custo restrito [R$]	636.196,9206	6.107.572,2693	413.349,2843
Intervalo [h]	165,1275	15,1848	216,0833
Confiabilidade	95,0279%	99,3206%	99,7980%
Disponibilidade	98,1562%	96,7106%	99,0807%
Tempo de execução [s]	0,7492	0,9109	1,0342

Pelos dados da Tabela 6.20, observa-se que apenas a distribuição 4319 apresentou um custo de operação restrito a 95,0% de confiabilidade maior que o mínimo global, encontrado a um intervalo 4 horas maior (aumento de 2,4%) e uma confiabilidade de 94,5% (redução de 0,5%).

Esse fato é interessante, especialmente porque mostra que apesar da modelagem do custo priorizar a confiabilidade, ainda considera o número de paradas. Isso constitui um *trade-off* entre manter uma confiabilidade alta e baixa probabilidade de quebra e necessidade de manutenções corretivas e um elevado número de intervenções de MP.

À medida que a confiabilidade é reduzida, um maior número de paradas emergenciais para reparos, e mais caras, são prováveis. Ao mesmo passo, um aumento da confiabilidade requer um intervalo de MP menor e aumenta o número de operação de MP. Essa última, apesar de mais barata, ainda contribui para o custo de operação estimado total. Uma situação complexa de ser resolvida e uma boa defesa para o uso de algoritmos de otimização.

Nos casos estudados, pela configuração do método PSO da Tabela 6.12, até 600 possibilidades foram verificadas por iteração, em um total de 60000 (sessenta mil) testes de cenários, em tempos que, em muitas das distribuições, foram inferiores a 1 segundo.

Destaca-se ainda que muitos algoritmos de otimização, PSO incluso, apesar de partirem de uma posição aleatória no espaço de busca, possuem mecanismos internos para que os próximos candidatos a solução estudados tenham resultados melhores que os candidatos anteriores.

A Figura 6.16 ilustra graficamente os resultados da Tabela 6.20.

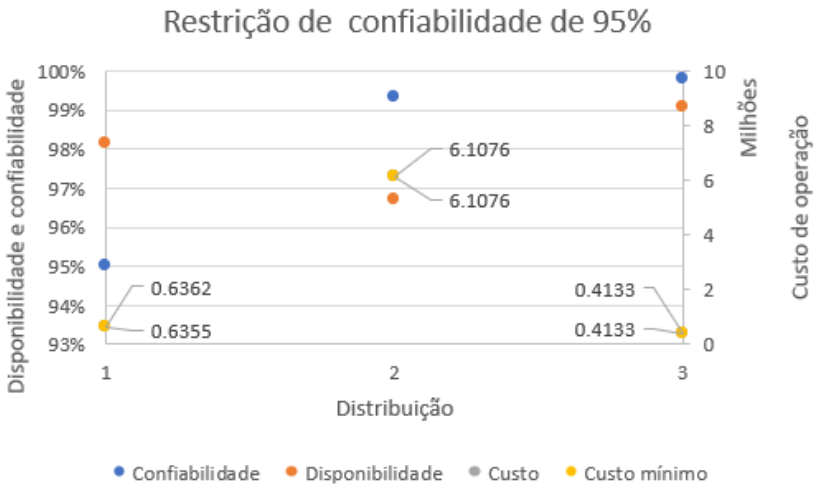

Figura 6.16: Resultados - Restrição 95% de confiabilidade – Distribuição 4319 (1), 4095 (2) e 1890 (3).

Fonte: Autor,2021.

De modo geral, os seis casos apresentaram uma clara região de mínimo global (Equação 2.2), como pode ser visto mais claramente nos gráficos das Figuras 6.3 e 6.6. Ao aplicar restrições, deslocou-se o resultado da otimização lateralmente, aumentando os custos de operação, mas ainda indicando o menor custo que respeitasse cada restrição.

O aumento da disponibilidade também foi acompanhado de maiores custos de operação. Mas vale destacar que o agendamento de manutenções não é tão direto como otimizar a função custo. É preciso respeitar cronogramas de produção por exemplo. Nesses casos, as equipes de manutenção e produção devem trabalhar com o *trade-off* custo/disponibilidade. O aplicativo apresentado no item 6.6 compõe uma ferramenta que auxiliará nesse processo.

Houve casos em que foi observado disponibilidade médias superiores aos das restrições, dentro de um certo intervalo, indicando que aquele intervalo de MP era também o ótimo global.

Esses resultados, que contrariam os achados de Ghosh (2009), sugerem que a modificação da Equação (5.13) foi uma boa escolha e que o uso do custo mínimo em otimização de MP baseado em confiabilidade e disponibilidade tem provável utilidade prática e, se validado por experimentos, pode ser utilizado para agendamento de operações de MP em sistemas reais.

Por fim, ressalta-se que todos os resultados foram obtidos em um computador em

formato de notebook. Não foi usado em nenhuma etapa desse estudo, incluindo treinamento e ajuste por RNA, qualquer tipo de equipamento especial, *workstation ou cloud computing*, tornando o estudo e a obtenção de resultados similares, em cenários diversos, completamente acessível.

6.6 Utilização do aplicativo

A utilização do aplicativo é direta e cada aba representa uma etapa, conforme fluxograma da Figura 6.17.

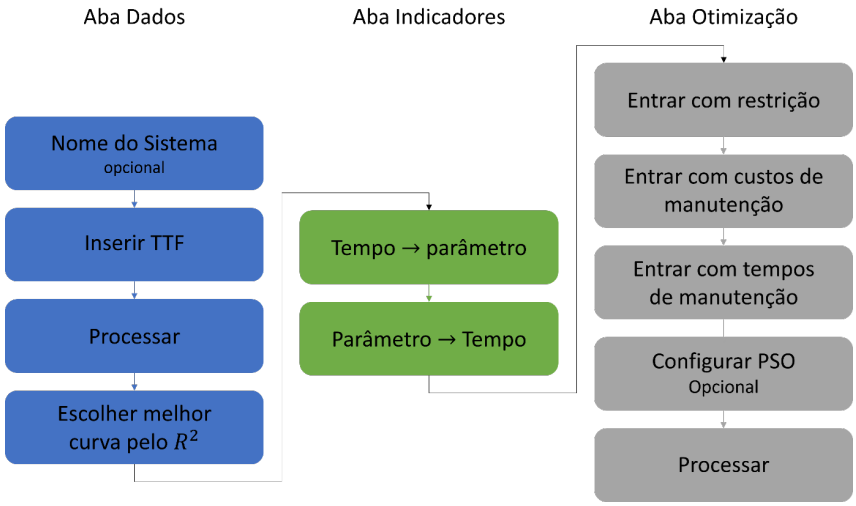

Figura 6.17: Fluxograma do funcionamento da aplicação OCM.
Fonte: Autor,2021.

A aba Dados recebe os dados básicos do equipamento em estudo. O nome do equipamento, opcional e não necessário e os tempos até falha TTF. Ao clicar em processar, o algoritmo irá utilizar a versão de treinamento da Rede Neural disponível, já treinada e ajustará os dados às curvas *Weibull* e *Lognormal*. O ajuste *Weibull* ocorrerá pela rede neural e o ajuste a função *Lognormal* ocorrerá de modo direto, pelas Equações (4.67) e (4.68) O aplicativo irá exibir os valores de ajuste para os parâmetros de cada distribuição e fornecerá também o coeficiente de determinação, para auxiliar na escolha da distribuição de melhor ajuste. Destaca-se que a escolha da distribuição e opcional e o usuário pode ignorar a recomendação do programa.

Na aba Indicadores é possível fazer cálculos da confiabilidade e taxa de falha para um determinado tempo ou o tempo associado a uma determinada confiabilidade ou taxa de falha. O aplicativo respeita a curva de distribuição escolhida na aba Dados. Na aba

Otimização ocorre a otimização do modelo restrito de custo de operação do equipamento. O algoritmo recebe as restrições, que podem ser nulas (sem restrição), os custos de manutenção, os tempos de reparo e vida de serviço e configurações do PSO (Opcional, o aplicativo já vem com uma configuração padrão). Ao clicar em processar, o aplicativo irá utilizar os parâmetros calculados na aba Dados, os dados introduzidos na aba Otimização e irá otimizar a função objetiva. Ao terminar o processo, irá exibir o tempo ideal entre operações de manutenção e o custo associado a esse intervalo. Plotar gráficos da execução e parâmetros também é possível e opcional.

Para exemplificar a utilização do aplicativo desenvolvido durante esse trabalho foi utilizado dados extraídos de Silveira (2012). Estes dados estão dispostos na Tabela 6.21.

Tabela 6.21: Dados TTF – Utilização do aplicativo
Fonte: Autor, 2021.

i	TTF [min]	TTF [dias)]
1	40000	27,7778
2	80000	55,5556
3	120000	83,3333
4	160000	111,1111
5	200000	138,8889
6	240000	166,6667
7	280000	194,4444
8	320000	222,2222
9	360000	250,0000
10	400000	277,7778

Os dados originais utilizados por Silveira (2012) estão dispostos na unidade de tempo minutos. O autor fez a conversão para dias com o objetivo de trabalhar com números menores. Essa preferência é justificada pelos valores utilizados no treinamento das Redes Neurais (itens 5.1.1 e 5.1.2), pequenos, quando comparados aos dados em minutos da Tabela 6.21.

A Tabela 6.22 contém os custos considerados por Silveira (2012).

Tabela 6.22: Custos de manutenção - utilização do aplicativo

Fonte: Autor, 2021.

Parâmetro	Descrição	Valor
C_{inc}	Custo por incidente	R$ 0,00
C_m	Custo por reparo programado	R$ 404,00
C_r	Custo por reparo não programado	R$ 2.324,00
T_{ser}	Tempo total de serviço esperado	10 anos (87600 horas)

Silveira (2012) não considerou custos de incidente e não explicitou o tempo total de serviço esperados, sendo este valor o mesmo já utilizado em todos os cálculos de indicadores industriais desse capítulo (Tabela 6.19). O tempo de reparo para ambas as operações de manutenção será o mesmo, 4 horas. Silveira (2012) ressalta que não é típico um cenário real apresentar valores de manutenção corretiva e preventiva iguais, semelhante às considerações feitas pelo autor.

A introdução dos dados no aplicativo ocorre na aba Dados e recebe os valores de TTF e um nome (opcional). A Tabela 6.21 contém dez dados de TTF, mas o aplicativo precisa de apenas de cinco, e por isso foi utilizado os TFF com índices ímpares, arredondados para uma casa decimal.

A Figura 6.18 ilustra o aplicativo configurado e com os dados já processados.

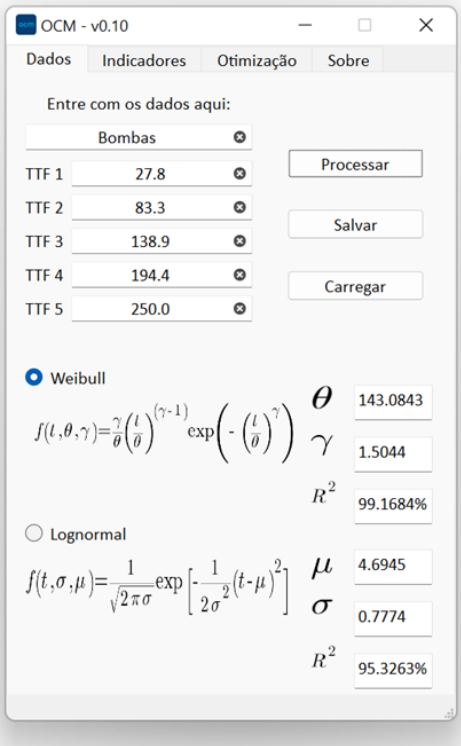

Figura 6.18: Configuração do aplicativo e resultados - Dados.
Fonte: Autor,2021.

Os resultados apresentados pelo aplicativo incluem os parâmetros de ajuste das duas distribuições de probabilidade e o coeficiente de determinação (item 4.5.4) para cada uma delas. Para os dados da Tabela 6.21, o ajuste para a distribuição *Weibull* apresentou R^2 de 99,1684%, indicando que o ajuste pela distribuição *Weibull* é provavelmente melhor que o ajuste pela distribuição *Lognormal*, R^2 de 95,3263% - aquela função foi selecionada.

A aba Indicadores calcula a confiabilidade e taxa de falha para um dado tempo ou vice-versa. A Figura 6.19 ilustra a configuração do aplicativo e resultados.

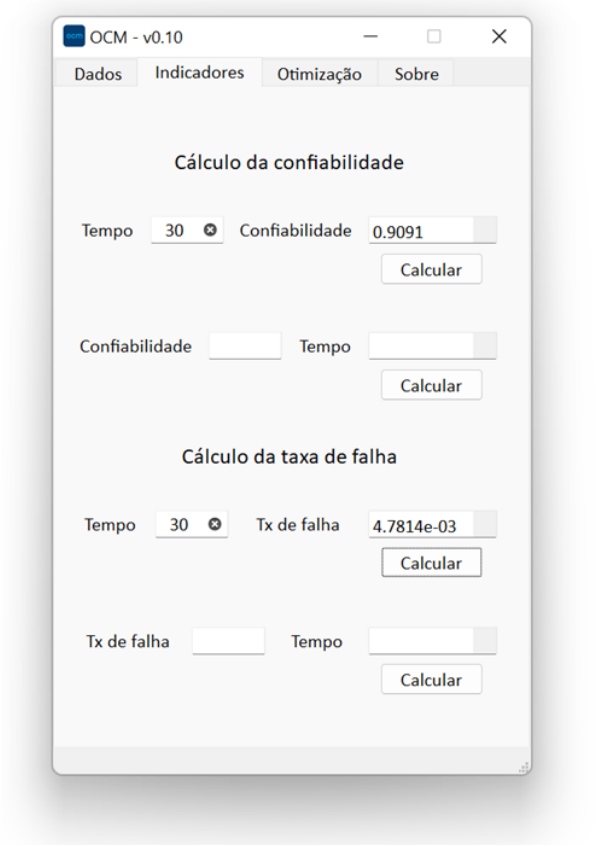

Figura 6.19: Configuração do aplicativo e resultados - Indicadores.
Fonte: Autor,2021.

Para um dado tempo de 30 dias, a confiabilidade para a bomba hidráulica em estudo é de 90,91% e a taxa de falha é de 4,7814E-3.

A aba Otimização calcula o intervalo de manutenção ótimo para os dados fornecidos. Foram utilizados os dados das Tabelas 6.21 e 6.22 e a configuração padrão do método PSO trazida pelo aplicativo. Nenhuma restrição foi utilizada. O limite máximo para intervalo de manutenção, e limite de busca do PSO, foi de 300 dias.

A Figura 6.20 ilustra a configuração do aplicativo e resultados.

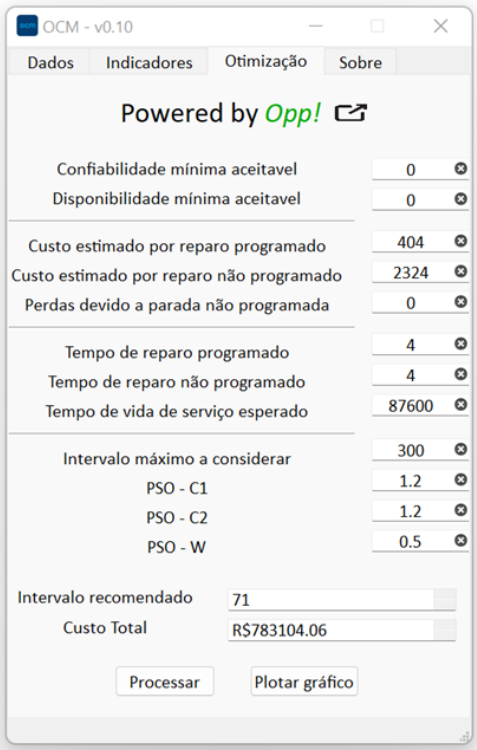

Figura 6.20: Configuração do aplicativo e resultados - Resultados.
Fonte: Autor,2021.

O intervalo de manutenção ótimo indicado pelo aplicativo é de 71 dias. O aplicativo não considera tempos fracionários, apenas inteiros, apesar do ajuste às distribuições de probabilidade considerarem casas decimais. Como não foram utilizadas restrições de confiabilidade, o custo indicado de R$783.104,06 para os dez anos de serviço do equipamento é o menor possível, ótimo global, segundo a modelagem da Equação (5.13).

Esse mesmo intervalo de manutenção indicado pelo programa, quando introduzido na aba Indicadores, indicou uma confiabilidade de 70,5769% e uma taxa de falha de 0,0074.

Os cálculos realizados pelo aplicativo são os mesmos introduzidos nos demais itens desse trabalho. A principal motivação do aplicativo é facilitar e agilizar estes mesmos cálculos e procedimentos. O acesso ao aplicativo está disponível no Apêndice D.

Capítulo 7

Conclusões e Trabalhos Futuros

A partir da análise dos resultados do capítulo 6 é possível afirmar que o estudo cumpriu com cada um dos seus objetivos (item 1.1). Cada uma das três partes do estudo apresentou resultados satisfatórios e dentro do esperado. A utilização de RNA para ajuste de BD (item 5.2) obteve sucesso em fazer o ajuste de dados a distribuição *Weibull*, apresentando erros inferiores a estudos comparáveis e ferramentas clássicas.

Existem recomendações para aperfeiçoamento da previsão, mas com base nos dados objetos de estudo é possível afirmar que RNA são uma boa ferramenta para ajuste a distribuições de probabilidade, mais simples e muitas vezes mais eficiente que os métodos tradicionais (itens 5.4 e 6.2), sobretudo quando a quantidade de dados é pequena.

O modelo com RNA se mostrou uma ferramenta mais eficiente, conseguindo fazer um bom ajuste a curvas de distribuição de probabilidade com poucos dados de TTF, quando os métodos tradicionais enfrentam dificuldades no mesmo cenário, dependendo de mais dados para um ajuste mais preciso. Essa maior capacidade de ajuste com menos dados deve ser destacada devido a dificuldade de coletar dados e equipamentos em campo.

Um atrativo adicional a utilização de RNA para o ajuste de distribuição de probabilidades foi a possibilidade de aproveitar todo o ferramental matemático já existente para cálculo dos indicadores de manutenção industrial. Ajustados os parâmetros para cada vetor característica, foi possível calcular de forma direta, todos os indicadores de interesse.

A otimização, por fim, o objetivo principal do trabalho, obteve ótimos resultados e capacidade de indicação do intervalo de MP ótimo para uma grande quantidade de cenários e restrições diferentes. Em todos eles, apresentou a melhor alternativa para agendamento de MP, baseada no limite mínimo de disponibilidade. Ambas as funções

atreladas a confiabilidade.

Com o aplicativo construído no estudo, os procedimentos realizados nesse trabalho se tornaram uma tarefa simples e rápida. A utilização desse aplicativo, inclusive, permitirá uma validação mais rápida, em estudos futuros, dos resultados aqui encontrados.

As modelagens e estimativas feitas aqui, porém, são teóricas, mas existe um desejo dos autores em validar os resultados encontrados por meio de experimentos e medições de campo.

Existe também um grande interesse e recomendação para estudos futuros, para realizar o treinamento de RNA, similar ao realizado aqui, com dados reais de TTF. Como comentado no item 6.1, ao utilizar dados sintéticos, a RNA está absorvendo como reais, particularidades dos algoritmos de criação dos dados, o que constitui uma limitação ao estudo. Dados de TTF distantes das margens do intervalo dos dados utilizados no treinamento também podem ter um ajuste de menor qualidade.

Das duas possíveis soluções para esse problema, criar e treinar um número muito elevado de BD por meio de outros processos de criação de dados sintéticos ou utilizar dados reais, coletados em campo, o último é considerado o mais interessante.

Somado a isso, foi utilizado nesse trabalho as distribuições *Weibull* e *Lognormal* com ajuste em dois parâmetros. Uma atualização dessas funções para três parâmetros e a substituição das simplificações de manutenções sempre perfeitas e retorno do sistema a confiabilidade máxima a cada MP, podem tornar o estudo ainda mais próximo de um cenário de MP real e permitir uma previsão do descomissionamento de equipamento.

Uma quarta e última colocação feita pelos autores é a introdução de mais um termo na Equação objetiva (5.13) para incluir sinais de sensores de componentes em equipamentos, formando uma equação híbrida, com componentes da MP clássica e telemetria da indústria 4.0.

Referências

[1] ASSIS, E. M., FIGUEIRÔA FILHO, C. L. S, LIMA, G. A. C., COSTA, L. A. N., SALLES, G. M. O. Machine Learning and q-Weibull Applied to Reliability Analysis in Hydropower Sector. IEEE, vol 8, p. 203331-203346.

[2] ASUERO, A. G., SAYAGO, A., GONZÁLEZ, A. G. (2007). The Correlation Coefficient: An Overview, Critical Reviews in Analytical Chemistry, 36:1, 41-49.

[3] BARRET, J. P. (1974). The coefficient of Determination – Some Limitations. The American Statistician, 28:1, 19-20.

[4] BASTOS, E. A. (2004). Otimização de Seções Retangulares de Concreto Armado Submetidas à Flexo-Compressão Oblíqua Utilizando Algoritmos Genéticos. Dissertação (Mestrado em Engenharia Civil) – Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2004.

[5] BELKIN, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proceedings of the National Academy of Sciences, 201903070. doi:10.1073/pnas.1903070116.

[6] BISHOP, C. M. (2006). Pattern Recognition and Machine Learning. Springer. Cambridge. Reino Unido.

[7] BRANDÃO, M. A. (2010). Estudo de Alguns Métodos Determinísticos de Otimização Irrestrita. Dissertação (Mestrado em Matemática) – Faculdade de Matemática, Universidade Federal de Uberlândia, Uberlândia.

[8] BRISCOE, E., & FELDMAN, J. (2011). Conceptual complexity and the bias/variance tradeoff. Cognition, 118(1), 2–16.

[9] BYRC, W. (2002). A uniform approximation to the right normal Tail integral. Applied Mathematics and Computation. V 123, p 365-374.

[10] CAMP, L. S. K., CAMP, M. E. K., CORRÂ Jr., C. A., CORRÊA, R. A. P. (2018). Diagnóstico assistido por inteligência computacional para identificação de anemias.

ERMAC, São Mateus.

[11] CAMP, L. S. K., CAMP, M. E. K., SILVA, A. A. M., FIGUEIRA, H. M., CORRÂ Jr., C. A., CORRÊA, R. A. P. (2019). Utilização de redes neurais artificiais na classificação de anemias. ENMC, Juiz de Fora.

[12] CAMP, L.S.K, HERNANDEZ, C. T. (2021). AHP e ANP combinados com métodos de otimização restrita na tomada de decisão multicritério para manutenção preventiva. Revista de Engenharia e Tecnologia. Ainda não publicado.

[13] CASSADY, C.R., BOWDEN, R., LIEW, L., POHL, E. (1999). Combining preventive maintenance and statistical process control: a preliminary investigation. IIE Transactions. V. 23, p. 471-478.

[14] CIVERCHIA, F., BOCCHINO, S., SALVADORI, C., ROSSI, E., MAGGIANI, L., PETRACCA, M. (2017). Industrial Internet of Things Monitoring Solutions for Advanced Predictive Maintenance Applications. Journal of Industrial Information Integration. Doi: 10.1016/j.jii.2017.02.003.

[15] DERTAT, A. (2017). Applied Deep Learning – Part 1: Artificial Neural Networks. Towards data science. Disponível em https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6, acessado em 17/08/2021.

[16] DSOUZA, J., VELAN, S. Preventive Maintenance for Fault Detection in Transfer Nodes using Machine Learning. In: International Conference on Computational Intelligence and Knowledge Economy, 12, 2019, Amity University Dubai, UAE.

[17] EBERHART, R., KENNEDY, J. (1995). A new optimizer using particle swarm theory. MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science.

[18] FOGLIATTO, F.S., RIBEIRO, J.L.D. Confiabilidade e manutenção industrial. 1 ed. Elsevier Editora LTDA, 2011.

[19] GÉRON, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly. Ed. 2.

[20] GHOSH, D., ROY, S. (2009). Maintenance optimization using probabilistic cost-benefit analysis. Journal of Loss Prevention in the Process Industries. 22. 403-407.

[21] GOOGLE DEVELOPERS. (2020). Descending into ML: Training and Loss. Machine Learning Crash Course. Disponivel em https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss, acessado em 11/07/2021.

[22] GUIDORIZZI, H. L. (2012). Um curso de Cálculo – Volume 2. Grupo Editorial Nacional. Ed. 5. Rio de Janeiro, Brasil.

[23] HUANG, J., CHANG, Q., ARINEZ, J. Deep Reinforcement Leaning based Preventive Maintenance Policy for Serial Production Lines. Expert Systems with Applications. V.160, 113701, ISSN 0957-4174, DOI: 10.1016/j.eswa.2020.113701, 2020.

[24] International Organization for Standardization (ISO). General Information on ISO.

[25] IZMAILOV, A., Solodov, M. (2007), Otimização – Volume 2: Métodos Computacionais. IMPA, Rio de Janeiro.

[26] KANANI, B. (2019). Activations Functions in Neural Network. Machine Learning Tutorias. Disponível em https://studymachinelearning.com/activation-functions-in-neural-network/, acessado em 17/08/2021.

[27] KINGMA, D. P., BA, J. L. (2015). Adam: A Method for Stochastic Optimization. ICLR.

[28] LE, J. (2018). Support Vector Machines in R. data camp. Disponível em https://www.datacamp.com/community/tutorials/support-vector-machines-r, acessado em 11/07/2021.

[29] LEWIS, E.E. Introduction to Reliability Engineering. 7.ed. Illinois: John Wiley & Sons, Inc. 1994.

[30] LIU, M. C., KUO, W., SASTRI, T. (1995). An exploratory study of a neural network approach for reliability data analysis. Quality and reliability engineering international, vol. 11, p. 107-112.

[31] LIRA, S. A., CHAVES NETO, A. (2006). Coeficientes de correlação para variáveis ordinais e dicotômicas derivados do coeficiente linear de Pearson. RECIE, Uberlândia, v.15, n.1/2, p/ 45-53.

[32] MARTINEZ, L. C. C. (2009). Otimização dos Circuitos de Refrigerante nos Trocadores de Calor de Sistemas de Refrigeração por Compressão de Vapor. Dissertação (Doutorado em Engenharia Mecânica. Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro.

[33] MISRA, R. (2019). Support Vector Machines – Soft Margin Formulation and Kernel Trick. Towards data science. Disponível em (https://towardsdatascience.com/support-vector-machines-soft-margin-formulation-and-kernel-trick-4c9729dc8efe), acessado em 08/07/2021.

[34] MURPHY, k. P. (2012). Machine Learning: A Probabilistic Perspective. The MIT Press. Massachusetts, EUA.

[35] NBR 5462. Confiabilidade e mantenabilidade. Rio de Janeiro: ABNT – Associação Brasileira de normas técnicas, 1994. 37p.

[36] NEMETH, T., ANSARI, F., SIHN, W., HASLHOFER, B., SCHINDLER, A. PriMa-X: A reference model for realizing prescriptive maintenance and assessing its maturity enhanced by machine Learning. In: CIRP Conference on Manufacturing Systems. 2018, p. 1039-1044.

[37] NUMPY. (2021). The Numpy community. API Reference. Disponível em https://numpy.org/doc/stable/reference. Acessado em: 05/08/2021.

[38] OLIVEIRA, G. T.; SARAMAGO, S. F. Estratégias de evolução diferencial aplicadas a problemas de otimização restritos. 15º POSMEC – Simpósio do Programa de Pós-Graduação em Engenharia Mecânica., 2005.

[39] ProConf 2000 – Confiabilidade de Componentes. CH Tech Desenvolvimento de Sistemas Ltda.

[40] RUDIN, C., WALTZ, D., ANDERSON, R.N., BOULANGER, A., SALLEB-AOISSI, A., CHOW, M. DUTTA, H., GROSS, P., HUANG, B., IEROME, S., ISSAC, D.F., KRESSNER, A., PASSONNEU, R., PASSONNEAU, A.R., WU, L. Machine Learning for the New York City Power Grid. IEEE Transactions on Pattern Analysis and Machine Intelligence. V.35, N.2, 2012.

[41] RUSHTON, I. (2018). Machine Learning Pt.1 – Linear Regression and Gradient Descent. Develo. Disponível em https://www.develodesign.co.uk/news/machine-learning-pt-1-fundamentals-and-introduction-to-gradient-descent/, acessado em 09/07/2021.

[42] PYTHON Software Foundation. (2021). Python 3.9.6 documentations. Disponível em https://docs.python.org/3. Acessado em: 28/07/2021.

[43] SELLITTO, M. A. (2005). Formulação estratégica da manutenção industrial com base na confiabilidade dos equipamentos. Revista Produção, v/ 15, n.1, p. 44-59.

[44] SILVEIRA, C. B. (2012). Confiabilidade e disponibilidade de máquinas: um exemplo prático. Confiabilidade Operacional, Manutenção Industrial. Citisystems. Disponível em https://www.citisystems.com.br/confiabilidade-disponibilidade-maquinas/2. Acessado em: 29/09/2021.

[45] STEINBRUCH, A., WINTERLE, P. (1995). Geometria Analítica. Person Maron Books. 1° Ed.

[46] TENSORFLOW. (2021). TensorFlow Core v2.5.0. Disponível em https://www.tensorflow.org/api_docs. Acessado em: 05/08/2021.

[47] TRELEA, I. C. (2003). The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters, 85(6), 317–325.

[48] WU, L., RUDIN, C., KAISER, G., ANDERSON, R. (2011). Data Quality Assurance and Performance Measurement of Data Mining for Preventive Maintenance of Power Grid. ACM 978-1-4503-0842-7.

Anexo 1 - CDF Normal Padronizado

$\Phi(z)$: Standard Normal CDF

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
− .0	.5000	.4960	.4920	.4880	.4840	.4801	.4761	.4721	.4681	.4641
− .1	.4602	.4562	.4522	.4483	.4443	.4404	.4364	.4325	.4286	.4247
− .2	.4207	.4168	.4129	.4090	.4052	.4013	.3974	.3936	.3897	.3859
− .3	.3821	.3783	.3745	.3707	.3669	.3632	.3594	.3557	.3520	.3483
− .4	.3446	.3409	.3372	.3336	.3300	.3264	.3228	.3192	.3156	.3121
− .5	.3085	.3050	.3015	.2981	.2946	.2912	.2877	.2343	.2810	.2776
− .6	.2743	.2709	.2676	.2643	.2611	.2578	.2546	.2514	.2483	.2451
− .7	.2420	.2389	.2358	.2327	.2297	.2266	.2236	.2206	.2177	.2148
− .8	.2119	.2090	.2061	.2033	.2005	.1977	.1949	.1922	.1894	.1867
− .9	.1841	.1814	.1788	.1762	.1736	.1711	.1685	.1660	.1635	.1611
−1.0	.1587	.1562	.1539	.1515	.1492	.1469	.1446	.1423	.1401	.1379
−1.1	.1357	.1335	.1314	.1292	.1271	.1251	.1230	.1210	.1190	.1170
−1.2	.1131	.1131	.1113	.1093	.1075	.1056	.1038	.1020	.1003	.09853
−1.3	.09680	.09510	.09342	.09176	.09012	.08851	.08691	.08534	.08379	.08226
−1.4	.08076	.07927	.07780	.07636	.07493	.07353	.07215	.07078	.06944	.06811
−1.5	.06681	.06552	.06426	.06301	.06178	.06057	.05938	.05821	.05705	.05592
−1.6	.05480	.05370	.05262	.05155	.05050	.04947	.04846	.04746	.04648	.04551
−1.7	.04457	.04363	.04272	.04182	.04093	.04006	.03920	.03836	.03754	.03673
−1.8	.03593	.03515	.03438	.03362	.03288	.03216	.03144	.03074	.03005	.02938
−1.9	.02872	.02807	.02743	.02680	.02619	.02559	.02500	.02442	.02385	.02330
−2.0	.02275	.02222	.02169	.02118	.02068	.02018	.01970	.01923	.01876	.01831
−2.1	.01786	.01743	.01700	.01659	.01618	.01578	.01539	.01500	.01463	.01426
−2.2	.01390	.01355	.01321	.01287	.01255	.01222	.01191	.01160	.01130	.01101
−2.3	.01072	.01044	.01017	$.0^{3}9903$	$.0^{3}9642$	$.0^{3}9387$	$.0^{3}9137$	$.0^{3}8894$	$.0^{3}8656$	$.0^{3}8424$
−2.4	$.0^{3}8198$	$.0^{3}7976$	$.0^{3}7760$	$.0^{3}7549$	$.0^{3}7344$	$.0^{3}7143$	$.0^{3}6947$	$.0^{3}6756$	$.0^{3}6569$	$.0^{3}6387$
−2.5	$.0^{3}6210$	$.0^{3}6037$	$.0^{3}5868$	$.0^{3}5703$	$.0^{3}5543$	$.0^{3}5386$	$.0^{3}5234$	$.0^{3}5085$	$.0^{3}4940$	$.0^{3}4799$
−2.6	$.0^{3}4661$	$.0^{3}4527$	$.0^{3}4396$	$.0^{3}4269$	$.0^{3}4145$	$.0^{3}4025$	$.0^{3}3907$	$.0^{3}3793$	$.0^{3}3681$	$.0^{3}3573$
−2.7	$.0^{3}3467$	$.0^{3}3364$	$.0^{3}3264$	$.0^{3}3167$	$.0^{3}3072$	$.0^{3}2980$	$.0^{3}2890$	$.0^{3}2803$	$.0^{3}2718$	$.0^{3}2635$
−2.8	$.0^{3}2555$	$.0^{3}2477$	$.0^{3}2401$	$.0^{3}2327$	$.0^{3}2256$	$.0^{3}2186$	$.0^{3}2118$	$.0^{3}2052$	$.0^{3}1988$	$.0^{3}1926$
−2.9	$.0^{3}1866$	$.0^{3}1807$	$.0^{3}1750$	$.0^{3}1695$	$.0^{3}1641$	$.0^{3}1589$	$.0^{3}1538$	$.0^{3}1489$	$.0^{3}1441$	$.0^{3}1395$
−3.0	$.0^{3}1350$	$.0^{3}1306$	$.0^{3}1264$	$.0^{3}1223$	$.0^{3}1183$	$.0^{3}1144$	$.0^{3}1107$	$.0^{3}1070$	$.0^{3}1035$	$.0^{3}1001$
−3.1	$.0^{3}9676$	$.0^{3}9354$	$.0^{3}9043$	$.0^{3}8740$	$.0^{3}8447$	$.0^{3}8164$	$.0^{3}7888$	$.0^{3}7622$	$.0^{3}7364$	$.0^{3}7114$
−3.2	$.0^{3}6871$	$.0^{3}6637$	$.0^{3}6410$	$.0^{3}6190$	$.0^{3}5976$	$.0^{3}5770$	$.0^{3}5571$	$.0^{3}5377$	$.0^{3}5190$	$.0^{3}5009$
−3.3	$.0^{3}4834$	$.0^{3}4663$	$.0^{3}4501$	$.0^{3}4342$	$.0^{3}4189$	$.0^{3}4041$	$.0^{3}3897$	$.0^{3}3758$	$.0^{3}3624$	$.0^{3}3495$
−3.4	$.0^{3}3369$	$.0^{3}3248$	$.0^{3}3131$	$.0^{3}3018$	$.0^{3}2909$	$.0^{3}2803$	$.0^{3}2701$	$.0^{3}2602$	$.0^{3}2507$	$.0^{3}2415$
−3.5	$.0^{3}2326$	$.0^{3}2241$	$.0^{3}2158$	$.0^{3}2078$	$.0^{3}2001$	$.0^{3}1926$	$.0^{3}1854$	$.0^{3}1785$	$.0^{3}1718$	$.0^{3}1653$
−3.6	$.0^{3}1591$	$.0^{3}1531$	$.0^{3}1473$	$.0^{3}1417$	$.0^{3}1363$	$.0^{3}1311$	$.0^{3}1261$	$.0^{3}1213$	$.0^{3}1166$	$.0^{3}1121$
−3.7	$.0^{3}1078$	$.0^{3}1036$	$.0^{4}9961$	$.0^{4}9574$	$.0^{4}9201$	$.0^{4}8842$	$.0^{4}8496$	$.0^{4}8162$	$.0^{4}7841$	$.0^{4}7532$
−3.8	$.0^{4}7235$	$.0^{4}6948$	$.0^{4}6673$	$.0^{4}6407$	$.0^{4}6152$	$.0^{4}5906$	$.0^{4}5669$	$.0^{4}5442$	$.0^{4}5223$	$.0^{4}5012$
−3.9	$.0^{4}4810$	$.0^{4}4615$	$.0^{4}4427$	$.0^{4}4247$	$.0^{4}4074$	$.0^{4}3908$	$.0^{4}3747$	$.0^{4}3594$	$.0^{4}3446$	$.0^{4}3304$
−4.0	$.0^{4}3167$	$.0^{4}3036$	$.0^{4}2910$	$.0^{4}2789$	$.0^{4}2673$	$.0^{4}2561$	$.0^{4}2454$	$.0^{4}2351$	$.0^{4}2242$	$.0^{4}2157$
−4.1	$.0^{4}2066$	$.0^{4}1978$	$.0^{4}1894$	$.0^{4}1814$	$.0^{4}1737$	$.0^{4}1662$	$.0^{4}1591$	$.0^{4}1523$	$.0^{4}1458$	$.0^{4}1395$
−4.2	$.0^{4}1335$	$.0^{4}1277$	$.0^{4}1222$	$.0^{4}1168$	$.0^{4}1118$	$.0^{4}1069$	$.0^{4}1022$	$.0^{5}9774$	$.0^{5}9345$	$.0^{5}8934$
−4.3	$.0^{5}8540$	$.0^{5}8163$	$.0^{5}7801$	$.0^{5}7455$	$.0^{5}7124$	$.0^{5}6807$	$.0^{5}6503$	$.0^{5}6212$	$.0^{5}5934$	$.0^{5}5668$
−4.4	$.0^{5}5413$	$.0^{5}5169$	$.0^{5}4935$	$.0^{5}4712$	$.0^{5}4498$	$.0^{5}4294$	$.0^{5}4098$	$.0^{5}3911$	$.0^{5}3732$	$.0^{5}3561$
−4.5	$.0^{5}3398$	$.0^{5}3241$	$.0^{5}3092$	$.0^{5}2949$	$.0^{5}2813$	$.0^{5}2682$	$.0^{5}2558$	$.0^{5}2439$	$.0^{5}2325$	$.0^{5}2216$
−4.6	$.0^{5}2112$	$.0^{5}2013$	$.0^{5}1919$	$.0^{5}1828$	$.0^{5}1742$	$.0^{5}1660$	$.0^{5}1581$	$.0^{5}1506$	$.0^{5}1434$	$.0^{5}1366$
−4.7	$.0^{5}1301$	$.0^{5}1239$	$.0^{5}1179$	$.0^{5}1123$	$.0^{5}1069$	$.0^{5}1017$	$.0^{6}9680$	$.0^{6}9211$	$.0^{6}8765$	$.0^{6}8339$
−4.8	$.0^{6}7933$	$.0^{6}7547$	$.0^{6}7178$	$.0^{6}6827$	$.0^{6}6492$	$.0^{6}6173$	$.0^{6}5869$	$.0^{6}5580$	$.0^{6}5304$	$.0^{6}5042$
−4.9	$.0^{6}4792$	$.0^{6}4554$	$.0^{6}4327$	$.0^{6}4111$	$.0^{6}3906$	$.0^{6}3711$	$.0^{6}3525$	$.0^{6}3348$	$.0^{6}3179$	$.0^{6}3019$

Figura i.0.1 - Anexo 1 - CDF normal padronizado

Fonte: Lewis (1994), pág. 415.

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
.0	.5000	.5040	.5080	.5120	.5160	.5199	.5239	.5279	.5319	.5359
.1	.5398	.5438	.5478	.5517	.5557	.5596	.5636	.5675	.5714	.5753
.2	.5793	.5832	.5871	.5910	.5948	.5987	.6026	.6064	.6103	.6141
.3	.6179	.6217	.6255	.6293	.6331	.6368	.6406	.6443	.6480	.6517
.4	.6554	.6591	.6628	.6664	.6700	.6736	.6772	.6808	.6844	.6879
.5	.6915	.6950	.6985	.7019	.7054	.7088	.7123	.7157	.7190	.7224
.6	.7257	.7291	.7324	.7359	.7389	.7422	.7454	.7486	.7517	.7549
.7	.7580	.7611	.7642	.7673	.7703	.7734	.7764	.7794	.7823	.7852
.8	.7881	.7910	.7939	.7967	.7995	.8023	.8051	.8078	.8106	.8133
.9	.8159	.8186	.8212	.8238	.8264	.8289	.8315	.8340	.8365	.8389
1.0	.8413	.8438	.8461	.8485	.8508	.8531	.8554	.8577	.8599	.8621
1.1	.8643	.8665	.8686	.8708	.8729	.8749	.8770	.8790	.8810	.8830
1.2	.8849	.8869	.8888	.8907	.8925	.8944	.8962	.8980	.8997	.90147
1.3	.90320	.90490	.90658	.90824	.90988	.91149	.91309	.91466	.91621	.91774
1.4	.91924	.92073	.92220	.92364	.92507	.92647	.92785	.92922	.93056	.93189
1.5	.93319	.93448	.93574	.93699	.93822	.93943	.94062	.94179	.94295	.94408
1.6	.94520	.94630	.94738	.94845	.94950	.95053	.95154	.95254	.95352	.95449
1.7	.95543	.95637	.95728	.95818	.95907	.95994	.96080	.96164	.96246	.96327
1.8	.96407	.96485	.96562	.96638	.96712	.96784	.96856	.96926	.96995	.97062
1.9	.97128	.97193	.97257	.97320	.97381	.97441	.97500	.97558	.97615	.97670
2.0	.97725	.97778	.97831	.97882	.97932	.97982	.98030	.98077	.98124	.98169
2.1	.98214	.98257	.98300	.98341	.98382	.98422	.98461	.98500	.98537	.98574
2.2	.98610	.98645	.98679	.98713	.98745	.98778	.98809	.98840	.98870	.98899
2.3	.98928	.98956	.98983	.$9^2$0097	.$9^2$0358	.$9^2$0613	.$9^2$0863	.$9^2$1106	.$9^2$1344	.$9^2$1576
2.4	.$9^2$1802	.$9^2$2024	.$9^2$2240	.$9^2$2451	.$9^2$2656	.$9^2$2857	.$9^2$3053	.$9^2$3244	.$9^2$3431	.$9^2$3613
2.5	.$9^2$3790	.$9^2$3963	.$9^2$4132	.$9^2$4297	.$9^2$4457	.$9^2$4614	.$9^2$4766	.$9^2$4915	.$0^2$5060	.$0^2$5201
2.6	.$9^2$5339	.$9^2$5473	.$9^2$5604	.$9^2$5731	.$9^2$5855	.$9^2$5975	.$9^2$6093	.$9^2$6207	.$0^2$6319	.$0^2$6427
2.7	.$9^2$6533	.$9^2$6636	.$9^2$6736	.$9^2$6833	.$9^2$6928	.$9^2$7020	.$9^2$7110	.$9^2$7197	.$0^2$7282	.$0^2$7365
2.8	.$9^2$7445	.$9^2$7523	.$9^2$7599	.$9^2$7673	.$9^2$7744	.$9^2$7814	.$9^2$7882	.$9^2$7948	.$0^2$8012	.$0^2$8074
2.9	.$9^2$8134	.$9^2$8193	.$9^2$8250	.$9^2$8305	.$9^2$8359	.$9^2$8411	.$9^2$8462	.$9^2$8511	.$0^2$8559	.$0^2$8605
3.0	.$9^2$8650	.$9^2$8694	.$9^2$8736	.$9^2$8777	.$9^2$8817	.$9^2$8856	.$9^2$8893	.$9^2$8930	.$9^2$8965	.$9^2$8999
3.1	.$9^3$0324	.$9^3$0646	.$9^3$0957	.$9^3$1260	.$9^3$1553	.$9^3$1836	.$9^3$2112	.$9^3$2378	.$9^3$2636	.$9^3$2886
3.2	.$9^3$3129	.$9^3$3363	.$9^3$3590	.$9^3$3810	.$9^3$4024	.$9^3$4230	.$9^3$4429	.$9^3$4623	.$9^3$4810	.$9^3$4991
3.3	.$9^3$5166	.$9^3$5335	.$9^3$5499	.$9^3$5658	.$9^3$5811	.$9^3$5959	.$9^3$6103	.$9^3$6242	.$9^3$6376	.$9^3$6505
3.4	.$9^3$6631	.$9^3$6752	.$9^3$6869	.$9^3$6982	.$9^3$7091	.$9^3$7197	.$9^3$7299	.$9^3$7398	.$9^3$7493	.$9^3$7585
3.5	.$9^3$7674	.$9^3$7759	.$9^3$7842	.$9^3$7922	.$9^3$7999	.$9^3$8074	.$9^3$8146	.$9^3$8215	.$9^3$8282	.$9^3$8347
3.6	.$9^3$8409	.$9^3$8469	.$9^3$8527	.$9^3$8583	.$9^3$8637	.$9^3$8689	.$9^3$8739	.$9^3$8787	.$9^3$8834	.$9^3$8879
3.7	.$9^3$8922	.$9^3$8964	.$9^4$0039	.$9^4$0426	.$9^4$0799	.$9^4$1158	.$9^4$1504	.$9^4$1838	.$9^4$2159	.$9^4$2468
3.8	.$9^4$2765	.$9^4$3052	.$9^4$3327	.$9^4$3593	.$9^4$3848	.$9^4$4094	.$9^4$4331	.$9^4$4558	.$9^4$4777	.$9^4$4988
3.9	.$9^4$5190	.$9^4$5385	.$9^4$5573	.$9^4$5753	.$9^4$5926	.$9^4$6092	.$9^4$6253	.$9^4$6406	.$9^4$6554	.$9^4$6696
4.0	.$9^4$6833	.$9^4$6964	.$9^4$7090	.$9^4$7211	.$9^4$7327	.$9^4$7439	.$9^4$7546	.$9^4$7649	.$9^4$7748	.$9^4$7843
4.1	.$9^4$7934	.$9^4$8022	.$9^4$8106	.$9^4$8186	.$9^4$8263	.$9^4$8338	.$9^4$8409	.$9^4$8477	.$9^4$8542	.$9^4$8605
4.2	.$9^4$8665	.$9^4$8723	.$9^4$8778	.$9^4$8832	.$9^4$8882	.$9^4$8931	.$9^4$8978	.$9^5$0226	.$9^5$0655	.$9^5$1066
4.3	.$9^5$1460	.$9^5$1837	.$9^5$2199	.$9^5$2545	.$9^5$2876	.$9^5$3193	.$9^5$3497	.$9^5$3788	.$9^5$4066	.$9^5$4332
4.4	.$9^5$4587	.$9^5$4831	.$9^5$5065	.$9^5$5288	.$9^5$5502	.$9^5$5706	.$9^5$5902	.$9^5$6089	.$9^5$6268	.$9^5$6439
4.5	.$9^5$6602	.$9^5$6759	.$9^5$6908	.$9^5$7051	.$9^5$7187	.$9^5$7318	.$9^5$7442	.$9^5$7561	.$9^5$7675	.$9^5$7784
4.6	.$9^5$7888	.$9^5$7987	.$9^5$8081	.$9^5$8172	.$9^5$8258	.$9^5$8340	.$9^5$8419	.$9^5$8494	.$9^5$8566	.$9^5$8634
4.7	.$9^5$8699	.$9^5$8761	.$9^5$8821	.$9^5$8877	.$9^5$8931	.$9^5$8983	.$9^6$0320	.$9^6$0789	.$9^6$1235	.$9^6$1661
4.8	.$9^6$2067	.$9^6$2453	.$9^6$2822	.$9^6$3173	.$9^6$3508	.$9^6$3827	.$9^6$4131	.$9^6$4420	.$9^6$4696	.$9^6$4958
4.9	.$9^6$5208	.$9^6$5446	.$9^6$5673	.$9^6$5889	.$9^6$6094	.$9^6$6289	.$9^6$6475	.$9^6$6652	.$9^6$6821	.$9^6$6981

Figura i.0.2- Anexo 1 - CDF normal padronizado

Fonte: Lewis (1994), pág. 416.

Apêndice

A. Implementações computacionais em Python

i. Repositório de funções

Functions need in other scripts - Lucas Kort (Jun. 23, 2021)

from tkinter.constants import UNITS

import tensorflow as tf

import pandas as pd

import matplotlib.pyplot as plt

from tensorflow.keras.layers import Input, Dense, Lambda

from tensorflow.keras.models import Model

from tensorflow.keras.optimizers import Adam

import tkinter as tk #hide tk window from tkinter import filedialog #get a file dialog window

def ml_setup():

#Banco de dados - com extensão

training_data_file_name = 'data/data_5_total.xlsx'

file_path = 'treinamentos/set 5/Teste 31-5/Teste 31-5' #carregar treinamento salvo

teste_data_file_name = 'data/data_5_5.xlsx' #Válido apenas para previsão

#proporção dos dados para treinamento/dados de validação

training_ratio = 0.8

#número de epochs

epochs = 10000

#quantidade de dados por epoch

batch_size = 10

#Função de erro - escolher

#loss_function = tf.keras.losses.MeanSquaredError()

loss_function = tf.keras.losses.MeanAbsolutePercentageError()

#taxa de aprendizado - dinâmica

lr_rate = tf.keras.optimizers.schedules.InverseTimeDecay(#taxa de aprendizado - dinâmica

#initial_learning_rate / (1 + decay_rate * step / decay_step)

initial_learning_rate=1e-3,

decay_steps=100000,

decay_rate=0.01

)

opt = Adam(

learning_rate = lr_rate,

amsgrad = True) #otimizador

return training_data_file_name, teste_data_file_name, file_path, training_ratio, epochs, batch_size, loss_function, lr_rate, opt

def weibull_layer(x):

"

Lambda function for generating weibull parameters

theta and gamma from a Dense(2) output.

Assumes tensorflow 2 backend.

Usage

outputs = Dense(2)(final_layer)

distribution_outputs = Lambda(weibull_layer)(outputs)

Parameters

x : tf.Tensor

output tensor of Dense layer

Returns

out_tensor : tf.Tensor

"

Get the number of dimensions of the input

num_dims = len(x.get_shape())

Separate the parameters

gamma,theta = tf.unstack(x, num=2, axis=-1)

Add one dimension to make the right shape

theta = tf.expand_dims(theta, -1)

gamma = tf.expand_dims(gamma, -1)

Apply a softplus to make positive

theta = tf.keras.activations.softplus(theta)

gamma = tf.keras.activations.softplus(gamma)

#print('theta',theta.get_shape(),' ngama',gamma.get_shape()) #teste de qualidade apenas

Join back together again

out_tensor = tf.concat((gamma,theta), axis=num_dims-1)

return out_tensor

def model_creator(loss_function, opt):

Define inputs with predefined shape

inputs = Input(shape=(5,))

Build network with some predefined architecture

#Layer1 = Dense(units=20, activation = 'sigmoid')

Layer2 = Dense(units=20, activation = 'sigmoid')

```
Layer3 = Dense(units=20, activation = 'sigmoid')
#output1 = Layer1(inputs)
# output2 = Layer2(output1)
last_output = Layer3(inputs)
# Predict the parameters of a negative binomial distribution
outputs = Dense(2)(last_output)
#distribution_outputs = Lambda(weibull_layer)(outputs)
# Construct model
model = Model(
  inputs=inputs,
  outputs=outputs
  )
model.compile(
loss = loss_function,
  optimizer = opt
  )
return model
def plot_loss_val(history):
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(loss) + 1)
plt.plot(epochs, loss, '-', label='Training loss')
plt.plot(epochs, val_loss, '–', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('epoch')
plt.ylabel('loss')
```

```
plt.legend()

plt.show()

def save_file(model):

try:

tk.Tk().withdraw()

export_file_path = filedialog.asksaveasfilename() #local de salvamento sem extensão

#Salvar modelo para reuso

#model.save(export_file_path)

model.save_weights(export_file_path)

except:

print('File not saved!')

def get_data(file_name,training_ratio):

time = pd.read_excel(

file_name,

sheet_name= 'Tempos Weibull',

index_col=0 #não tem nenhuma columa com nome para o index

)

parameters = pd.read_excel(

file_name,

sheet_name= 'Parâmetros',

index_col=0 #não tem nenhuma columa com nome para o index

)

header = []

vector_n,vector_size = time.shape #tamanho da última dimensão

for i in range(vector_size): #cabeçalho para os tempos
```

header.append('T' + str(i+1))

Divisão dos dados de acordo com a divisão

lim_training = round(vector_n*training_ratio)

train_x = time.loc[range(lim_training),header].values.tolist()

train_y =
parameters.loc[range(lim_training),['Log(Theta)','Gamma']].values.tolist()

val_x = time.loc[range(lim_training,vector_n),header].values.tolist()

val_y =
parameters.loc[range(lim_training,vector_n),['Log(Theta)','Gamma']].values.tolist()

Sanity check de algumas amostras

test_x = time.loc[range(vector_n-10,vector_n),header].values.tolist()

test_y =
parameters.loc[range(vector_n-10,vector_n),['Log(Theta)','Gamma']].values.tolist()

return train_x, train_y, val_x, val_y, test_x, test_y

def results_check(pred_params,test_y):

#vector_n,vector_size = test_y.shape tamanho da última dimensão

vector_n = len(test_y)

MSE_test = 0

for i in range(vector_n):

MSE_test += (pred_params[i] - test_y[i])**2

MSE_test = MSE_test/vector_n

print(' nSanity check n n#MSE n Theta|Gamma: n', MSE_test,' n n#Valores pelo modelo n Theta|Gamma: n',pred_params,' n n#Valores reais n Theta|Gamma:')

for i in range(10):

print(test_y[i])

def predict_val(test_x):

#Carregar parâmetros

file_name, test_file_name, file_path, training_ratio, epochs, batch_size,
loss_function, lr_rate, opt = ml_setup()

#Criar modelo

model = model_creator(loss_function,opt)

#Restaurar último treinamento

checkpoint = tf.train.Checkpoint(model)

checkpoint.restore(file_path).expect_partial()

#Prever resultados

pred_params = model.predict(test_x)

return pred_params

ii.Iniciar Treinamento

Inicial data training - Lucas kort (Jun. 23, 2021)

import tensorflow as tf

from function_repository import model_creator, plot_loss_val, save_file, get_data,
results_check, ml_setup

#Carregar parâmetros

file_name, test_file_name, file_path, training_ratio, epochs, batch_size,
loss_function, lr_rate, opt = ml_setup()

#Criar modelo

model = model_creator(loss_function,opt)

#Obter dados

train_x, train_y, val_x, val_y, test_x, test_y = get_data(file_name,training_ratio)

Treinamento do modelo

history = model.fit(train_x, train_y, batch_size = batch_size, epochs = epochs,
validation_data = (val_x, val_y))

#Prever resultados

pred_params = model.predict(test_x)

#Teste de sanidade

results_check(pred_params,test_y)

#Plotar loss e val

plot_loss_val(history)

#Salvar treinamento

save_file(model)

iii. Continuar Treinamento

Resumes training from selected previously saved weights - Lucas kort (Jun. 23, 2021)

import tensorflow as tf

from function_repository import model_creator, plot_loss_val, save_file, get_data, results_check, ml_setup

#Carregar parâmetros

file_name, test_file_name, file_path, training_ratio, epochs, batch_size, loss_function, lr_rate, opt = ml_setup()

#Criar modelo

model = model_creator(loss_function,opt)

#Obter dados

train_x, train_y, val_x, val_y, test_x, test_y = get_data(file_name,training_ratio)

#Restaurar último treinamento

checkpoint = tf.train.Checkpoint(model)

checkpoint.restore(file_path).expect_partial()

Treinamento do modelo

history = model.fit(train_x, train_y, batch_size = batch_size, epochs = epochs, validation_data = (val_x, val_y))

#Prever resultados

pred_params = model.predict(test_x)

#Teste de sanidade

results_check(pred_params,test_y)

#Plotar loss e val

plot_loss_val(history)

#Salvar treinamento

save_file(model)

iv. Ajustar dados por Machine Learning

import tensorflow as tf

from function_repository import model_creator, training_data

def predict_val():

#Carregar parâmetros

file_name, file_path, training_ratio, epochs, batch_size, loss_function, lr_rate, opt = ml_setup()

#Obter dados

train_x, train_y, val_x, val_y, test_x, test_y = training_data(file_name,training_ratio)

#Criar modelo

model = model_creator(loss_function,opt)

#Restaurar último treinamento

checkpoint = tf.train.Checkpoint(model)

checkpoint.restore(file_path).expect_partial()

#Prever resultados

pred_params = model.predict(test_x)

return pred_params

v. Função para o aplicativo Opp

```
# Objectives and constraints functions (Aug. 04, 2021)

import math

import numpy as np

def objective(var_o): #objetive functions Weibull

#confiabilidade

gamma_ =

theta =

t = int(math.floor(var_o.copy()))

r_t = math.exp(-((t/theta)**(gamma_)))

#custo

c_m = 1000

c_r = 2500

c_inc = 10000

t_ser = 87600

mttf =

c_t = (t_ser/t)*c_m*r_t + (t_ser/mttf)*(c_r+c_inc)*(1-r_t)

#função objetivo

y = c_t

return y

def constraints(var_c): #constraint functions

#confiabilidade

gamma_ =

theta =

lim = #limite

t = int(math.floor(var_c.copy()))
```

r_t = math.exp(-((t/theta)**(gamma_)))

#disponibilidade

t_m = #tempo de reparo

t_r = #tempo de manutenção

a_t = t/(t + r_t*t_m + (1-r_t)*t_r)

#Substituir r_t por a_t para usar função de confiabilidade como restrição

#constraint functions 1 to n

if (r_t >= lim): #test conditions 1 to n

return True #all conditions has been met

else:

return False #one or mor$_t$*conditionhasn'tbeenmet*

def objective(var_o): #objetive functions *Lognormal*

#confiabilidade

mu = 5.9093828021596

sigma = 0.486238331177103

t = int(math.floor(var_o.copy()))

z = (mu - math.log(var_o))/sigma

termo_1 = ((4-math.pi)*abs(abs(z)) + math.sqrt(2*math.pi)*(math.pi-2))

termo_2 = (((4-
math.pi)*math.sqrt(2*math.pi)*abs(z)**2)+(2*math.pi*abs(z))+(2*math.sqrt(2*math.pi)*(math.pi
2)))

termo_3 = math.exp(-(abs(z)**2)/2)

if z < 0:

r_t = 1-((termo_1/termo_2)*termo_3)

else:

r_t = 1 - (1-((termo_1/termo_2)*termo_3))

#custo

```
c_m = 1000

c_r = 2500

c_inc = 10000

t_ser = 87600

mttf = 413

c_t = (t_ser/t)*c_m*r_t + (t_ser/mttf)*(c_r+c_inc)*(1-r_t)

#função objetivo

y = c_t

return y

def constraints(var_c): #constraint functions

#confiabilidade

mu = 5.9093828021596

sigma = 0.486238331177103

t = int(math.floor(var_c.copy()))

z = (mu - math.log(var_c))/sigma

termo_1 = ((4-math.pi)*abs(abs(z)) + math.sqrt(2*math.pi)*(math.pi-2))

termo_2 = (((4-math.pi)*math.sqrt(2*math.pi)*abs(z)**2)+
(2*math.pi*abs(z))+(2*math.sqrt(2*math.pi)*(math.pi-2)))

termo_3 = math.exp(-(abs(z)**2)/2)

if z < 0:

r_t = 1-((termo_1/termo_2)*termo_3)

else:

r_t = 1 - (1-((termo_1/termo_2)*termo_3))

#disponibilidade

t_m = 3 #tempo de reparo

t_r = 5 #tempo de manutenção
```

a_t = t/(t + r_t*t_m + (1-r_t)*t_r)

#Substituir r_t por a_t para usar função de confiabilidade como restrição

#constraint functions 1 to n

if (a_t >= 0.99): #test conditions 1 to n

return True #all conditions has been met

else:

return False one or more condition hasn't been met

vi. Geração de Dados Weibull

Generates random data in specified shape - Lucas kort (Jun. 23, 2021)

import numpy as np

import pandas as pd

import random as rand

import math

import tkinter as tk #hide tk window

from tkinter import filedialog #get a file dialog window

#Configuração dos dados de saída

record_size = 5 #tamanho de cada vetor de dados

record_total_n = 1000 #número de vetores por dataset

a = 1 #limite inferior de intervalo log(theta)

b = 3.5 #limite superior de intervalo log(theta)

gamma = 0.5 #inicialização de parâmetro gamma (Gamma mínimo)

gamma_plus = 0.5 #incremento de gamma

gamma_each = 150 #incrementar gamma a cada x conjunto de dados

#

#função de densidade weibull

```
def weibull(theta,gamma,x,x_size):
weibull_x = np.zeros(x_size)
for i in range(x_size):
weibull_x[i] = theta*(-math.log(x[i]))**(1/gamma)
return weibul l_x
#salvar em arquivo do excel
def export_xlsx(parameters,time_weibull,record_size,index_col):
index = range(index_col) #Index na ordem crescente
rand_index = np.random.permutation(index) #tornar a ordem do index aleatória
#Primeiro aleatório, depois crescente, para planilha ficar na ordem crescente
df_parameters=pd.DataFrame(
parameters,
index=rand_index,
columns=['Theta','Log(Theta)','Gamma']
)
header = []
for i in range(record_size): #cabeçalho para os tempos
header.append('T' + str(i+1))
df_time_weibull=pd.DataFrame(
time_weibull,
index=rand_index,
columns=header
)
new_index = np.random.permutation(index) #tornar a ordem dos dados aleatória
root = tk.Tk()
root.withdraw()
```

export_file_path = filedialog.asksaveasfilename(defaultextension ='.xlsx') #local de salvamento + extensão .xlsx

with pd.ExcelWriter(export_file_path) as writer:#escrever em mais de uma planilha ao mesmo tempo

df_parameters.reindex(index).to_excel(writer,sheet_name = 'Parâmetros')

df_time_weibull.reindex(index).to_excel(writer,sheet_name = 'Tempos Weibull')

#Inicialização de variáveis

parameters = np.zeros((record_total_n,3))

time_weibull = np.zeros((record_total_n,record_size))

x = np.random.default_rng().uniform(0,1,record_size)

#geração automática de dados Weibull

for i in range(record_total_n):

log_theta = a+(b-a)*rand.random()

theta = 10**log_theta

if (i%gamma_each==0):

gamma += gamma_plus

time_weibull[i,:] = np.sort(weibull(theta,gamma,x,record_size).copy())

parameters[i,:] = [theta,log_theta,gamma]

#Exportar dados para arquivo do excel

export_xlsx(parameters,time_weibull,record_size,record_total_n)

vii. Geração de dados - *Lognormal*

Generates random data in specified shape - Lucas kort (Jun. 23, 2021)

import numpy as np

import pandas as pd

import random as rand

```
import tkinter as tk #hide tk window
from tkinter import filedialog #get a file dialog window
#Configuração dos dados de saída
record_size = 5 #tamanho de cada vetor de dados
record_total_n = 1000 #número de vetores por dataset
mu_a = 3 #limite inferior de intervalo log(theta)
mu_b = 8 #limite superior de intervalo log(theta)
sigma_a = 0.03 #incremento de sigma
sigma_b = 0.5 #incrementar sigma a cada x conjunto de dados
#salvar em arquivo do excel
def export_xlsx(parameters,time_weibull,record_size,index_col):
index = range(index_col) #Index na ordem crescente
rand_index = np.random.permutation(index) #tornar a ordem do index aleatória
#Primeiro aleatório, depois crescente, para planilha ficar na ordem crescente
df_parameters=pd.DataFrame(
parameters,
index=rand_index,
columns=['Mu','Sigma']
)
header = []
for i in range(record_size): #cabeçalho para os tempos
header.append('T' + str(i+1))
df_time_weibull=pd.DataFrame(
time_weibull,
index=rand_index,
columns=header
```

)

new_index = np.random.permutation(index) #tornar a ordem dos dados aleatória

root = tk.Tk()

root.withdraw()

export_file_path = filedialog.asksaveasfilename(defaultextension ='.xlsx') #local de salvamento + extensão .xlsx

with pd.ExcelWriter(export_file_path) as writer: escrever em mais de uma planilha ao mesmo tempo

df_parameters.reindex(index).to_excel(writer,sheet_name = 'Parâmetros')

df_time_weibull.reindex(index).to_excel(writer,sheet_name = 'Tempos Lognormal')

#Inicialização de variáveis

parameters = np.zeros((record_total_n,2))

time_weibull = np.zeros((record_total_n,record_size))

x = np.random.default_rng().uniform(0,1,record_size)

#geração automática de dados Weibull

for i in range(record_total_n):

mu = mu_a+(mu_b-mu_a)*rand.random()

sigma = sigma_a+(sigma_b-sigma_a)*rand.random()

time_weibull[i,:] = np.sort(np.random.lognormal(mu,sigma,record_size))

parameters[i,:] = [mu,sigma]

#Exportar dados para arquivo do excel

export_xlsx(parameters,time_weibull,record_size,record_total_n)

viii. Testar modelo

import math

#confiabilidade

var_c = 236.83501593

#confiabilidade

mu = 3.17022358779506

sigma = 0.186847401773389

t = int(math.floor(var_c))

z = (mu - math.log(t))/sigma

termo_1 = ((4-math.pi)*abs(abs(z)) + math.sqrt(2*math.pi)*(math.pi-2))

termo_2 =(((4-math.pi)*math.sqrt(2*math.pi)*abs(z)**2)+
(2*math.pi*abs(z))+(2*math.sqrt(2*math.pi)*(math.pi-2)))

termo_3 = math.exp(-(abs(z)**2)/2)

if z > 0:

r_t = 1-((termo_1/termo_2)*termo_3)

else:

r_t = 1 - (1-((termo_1/termo_2)*termo_3))

#disponibilidade

t_m = 0.5 tempo de reparo

t_r = 2 tempo de manutenção

a_t = t/(t + r_t*t_m + (1-r_t)*t_r)

#custo

c_m = 1000

c_r = 2500

c_inc = 10000

t_ser = 87600

mttf = 24.21329475

c_t = (t_ser/t)*c_m*r_t + (t_ser/mttf)*(c_r+c_inc)*(1-r_t)

print('r_t:',r_t,' na_t:',a_t,' nc_t:',c_t,' nt:',t)

ix. Aplicativo para otimização Opp

Disponível em: https://github.com/LukeKort/Opp/releases

Figura A.A.1 - QR Code - Aplicativo de otimização

B. Dados utilizados

i. Bando de dados gerados (Apêndice A) e utilizados nos treinamentos

Disponível em: https://github.com/LukeKort/reliability_ml/tree/main/data

Figura B.1 - QR Code - BD gerados

C. Resultados

ii. Planilhas em Excel com todos os dados de resultados

Disponível em: https://github.com/LukeKort/reliability_ml/tree/main/resultados

Figura C.1 - QR Code - Planilhas de resultados

D. Aplicativo OCM

Disponível em: https://github.com/LukeKort/reliability_ml

Figura D.1 - QR Code – Aplicativo OCM