UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE TECNOLOGIA E GEOCIÊNCIAS

DEPARTAMENTO DE ENGENHARIA CIVIL E AMBIENTAL

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA CIVIL

ADRIANO DAYVSON MARQUES FERREIRA

ANÁLISE MULTIRESOLUÇÃO E REDES NEURAIS PROFUNDAS PARA AVALIAÇÃO DE DUTOS CORROÍDOS

Recife

2022

ADRIANO DAYVSON MARQUES FERREIRA

ANÁLISE MULTIRESOLUÇÃO E REDES NEURAIS PROFUNDAS PARA AVALIAÇÃO DE DUTOS CORROÍDOS

Tese apresentada ao Programa de PósGraduação em Engenharia Civil da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de doutor em Engenharia Civil. Área de concentração: Estruturas.

Orientadora: Profª. Drª. Silvana Maria Bastos Afonso da Silva
Coorientador: Prof. Dr. Ramiro Brito Willmersdorf

Recife
2022

Catalogação na fonte
Bibliotecário Gabriel Luz CRB-4 / 2222

F383a Ferreira, Adriano Dayvson Marques.
Análise multiresolução e redes neurais profundas para avaliação de dutos corroídos / Adriano Dayvson Marques Ferreira. 2022.
159 f: il., tabs.

Orientadora: Profa. Dra. Silvana Maria Bastos Afonso da Silva.
Coorientador: Prof. Dr. Ramiro Brito Willmersdorf.
Tese (Doutorado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia Civil, Recife, 2022.
Inclui referências.

1. Engenharia civil. 2. Dutos corroídos. 3. Análise multiresolução. 4. Transformada wavelet discreta. 5. Redes neurais profundas. 6. Método dos elementos finitos. I. Silva, Silvana Maria Bastos Afonso da (Orientadora). II. Willmersdorf, Ramiro Brito (Coorientador). III. Título.

UFPE

624 CDD (22. ed.) BCTG / 2022 - 381

ADRIANO DAYVSON MARQUES FERREIRA

ANÁLISE MULTIRESOLUÇÃO E REDES NEURAIS PROFUNDAS PARA AVALIAÇÃO DE DUTOS CORROÍDOS

Tese apresentada ao Programa de PósGraduação em Engenharia Civil da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de doutor em Engenharia Civil. Área de concentração: Estruturas.

Aprovado em 30 de Agosto de 2022

Orientadora: Profª. Drª. Silvana Maria Bastos Afonso da Silva
Coorientador: Prof. Dr. Ramiro Brito Willmersdorf

BANCA EXAMINADORA

participação por videoconferência
Prof. Dr. Renato de Siqueira Motta (examinador interno)
Universidade Federal de Pernambuco

participação por videoconferência
Prof. Dr. Breno Pinheiro Jacob (examinador externo)
Universidade Federal do Rio de Janeiro

participação por videoconferência
Prof. Dr. José Ângelo Peixoto da Costa (examinador externo)
Instituto Federal de Educação, Ciência e Tecnologia de Pernambuco

participação por videoconferência
Prof. Dr. Paulo Fernando Silva Sousa (examinador externo)
Universidade Federal de Pernambuco

participação por videoconferência
Prof. Dr. Bernardo Horowitz (examinador interno)
Universidade Federal de Pernambuco

Dedico este trabalho à minha famĺia.

AGRADECIMENTOS

Agradeço à minha família que sempre me deu muito apoio, em especial a minha mãe Inalva e irmãos, Anderson e Cybelle, por terem me ajudado ao longo da vida e unindo forças para que eu continuasse avançando. Um agradecimento especial à minha esposa Amélia, por ter me dado muito suporte e amor nesta fase de pesquisa e escrita e para o meu filho amado Miguel por estar sempre sorrindo e trazendo a alegria necessária nos momentos mais difícieis.

Agradeço a professora Silvana Bastos e ao professor Ramiro Willmersdorf por me guiarem nesse caminho científico e terem me colocado no projeto o qual me trouxe até aqui. Aos amigos da UFPE, que sempre juntos discutem e resolvem vários problemas além de transmitir conhecimentos, auxiliando a realização do trabalho. E agradeço a todos em geral que de alguma forma mesmo que direta ou indiretamente contribuíram para a conclusão deste trabalho.

"Todas as vitórias ocultam uma abdicação."

(Simone de Beauvoir)

RESUMO

A avaliação da integridade de dutos corroídos é considerada uma tarefa vital na indústria de petróleo e gás. O objetivo desta pesquisa é desenvolver um sistema eficiente, capaz de prever com precisão a pressão de ruptura de dutos corroídos, com perfis de corrosão complexos por meio de modelos híbridos, usando análise multiresolução, simulações numéricas e metamodelos. O trabalho aborda a parametrização de geometrias reais de corrosão e sua utilização como entrada para um sistema de redes neurais que pode prever com velocidade e precisão a pressão de ruptura. O mapa de corrosão é obtido a partir de inspeções ultrassônicas e os dados são utilizados tanto na forma de um perfil *river bottom* quanto na forma de mapeamento tridimensional. O método dos elementos finitos (MEF) é usado para avaliar a pressão de ruptura. Rotinas criadas para gerar automaticamente modelos de elementos finitos (EF) axissimétricos e tridimensionais são utilizadas e as pressões de falha são obtidas através de análise não linear. Os modelos de EF e o procedimento de análise são validados com testes experimentais e em comparação a métodos de avaliação semi-empíricos. A transformada *wavelet* discreta é realizada para a parametrização das espessuras remanescentes e como um banco de filtros para reduzir a quantidade de dados que descreve o defeito. Os coeficientes obtidos a partir da transformada *wavelet* discreta e as propriedades dos materiais dos dutos são utilizados como parâmetros de entrada para alimentar a rede neural profunda. Modelos sintéticos axissimétricos e tridimensionais com estatísticas semelhantes aos perfis reais de corrosão são criados e submetidos à análise não linear via MEF. As respectivas pressões de falha obtidas a partir dos defeitos sintéticos são usadas para treinar uma rede neural capaz de prever a pressão de ruptura dos dutos com defeitos no formato de perfil *river bottom* e uma rede capaz de prever a pressão de defeitos tridimensionais complexos. Os resultados obtidos com as redes neurais profundas são muito precisos para todos os casos apresentados neste trabalho, tanto na utilização de modelos axissimétricos quanto na utilização de modelos tridimensionais.

Palavras-chave: dutos corroídos; análise multiresolução; transformada wavelet discreta; redes neurais profundas; método dos elementos finitos.

ABSTRACT

The structural integrity of corroded pipelines is considered a vital task in the oil and gas industry. This research aims to develop an efficient system to accurately predict the burst pressure of corroded pipelines with complex corrosion profiles through hybrid models using multiresolution analysis, numerical analysis, and metamodels. These work addresses the parametrization of real corrosion shapes and its use as input to a neural network system that can accurately predict the burst pressure quickly. The corrosion map is obtained from ultrasonic inspections and the data is used both in the form of a river bottom profile and in the form of a three-dimensional mapping. The finite element method (FEM) is used to evaluate the burst pressure. Scripts to automatically generate axisymmetric and three-dimensional finite element (FE) models are used and failure pressures are obtained with non-linear analysis. The FE models and analysis procedure are validated against experimental tests and are compared with semi-empirical assessment methods. A discrete wavelet transform is performed for the parametrization of the remaining thicknesses and as a filter bank to reduce the amount of data do describes the defect. The coefficients obtained from the wavelet transform and the material properties of the pipelines are used as inputs to feed a deep neural network. Axisymetric and three-dimensional synthetic models that have similar statistics to real corrosion profiles are created and submitted to non-linear FEM analysis. The respective failure pressures obtained from the synthetic defects are used to train a neural network to predict the burst pressure of the pipelines with river bottom profile and to train a neural network to predict the burst pressure of pipelines with real corrosion defects. The results obtained with the deep neural networks are very precise for all the cases presented in this work, both in the use of axisymmetric models and in the use of three-dimensional models.

Keywords: corroded pipelines; multiresolution analysis; discrete wavelet transform; deep neural network; finite element method.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O objetivo desta pesquisa é desenvolver um sistema eficiente, capaz de prever com precisão a pressão de ruptura de dutos corroídos, com perfis de corrosão complexos por meio de modelos híbridos, usando análise multiresolução, simulações numéricas e metamodelos.

1.1 MOTIVAÇÃO

A avaliação da segurança operacional de dutos corroídos é uma tarefa muito importante nas indústrias de petróleo e gás devido ao custo muito alto da consequência de falhas, que podem causar danos ao meio ambiente e colocar em risco a segurança e a vida da população em casos de acidentes.

A malha terrestre de dutos no Brasil se estende por milhares de quilômetros e é um dos meios mais seguros para transporte de petróleo e gás e um dos que mais preocupam as grandes organizações. Além da malha terrestre há também os dutos submarinos responsáveis pelo transporte do óleo da linha de produção submarina a uma unidade estacionária de produção (UEP).

Tanto os dutos submarinos quanto os dutos terrestre sofrem problemas devido à corrosão que exigem uma resposta rápida quanto a sua integridade, pois podem causar acidentes graves com consequências ambientais e risco a vida humana. Um dos métodos comprovadamente mais eficientes para avaliar a integridade dos mesmos é o Método dos Elementos Finitos (CABRAL et al., 2017), porém exige tempo e experiência do engenheiro para gerar modelos corretos e gerar análises adequadas.

Modelar dutos com defeitos considerando a geometria mais próxima ao encontrado em campo, defeitos reais, tornam esse trabalho de modelagem manual mais complexo e demorado, podendo ser inviável realizar na maioria dos casos, devido ao excesso de geometrias necessárias para uma representação fiel da corrosão. Uma aproximação simplificada ao perfil de corrosão tridimensional é chamada *river bottom* ((CABRAL et al., 2022)). Este perfil é formado pelos valores mínimos de espessura remanescente ao longo do comprimento longitudinal do defeito. A aproximação é utilizada na indústria junto a métodos semi-empíricas para avaliação de integridade com resultados menos conservadores em relação aos métodos empíricos simplificados. Devido a irregularidade da superfície do defeito e da quantidade de medições de espessura, a modelagem por Elementos Finitos (EF) manual destes defeitos também exige muitas horas de trabalho de um profissional especialista, encarecendo a solução para aplicação industrial.

Modelar um duto com uma corrosão real complexa, que se baseia diretamente

nos dados obtidos através de inspeção ultrassônica, com elementos finitos é uma tarefa difícil e demorada devido ao alto número de geometrias que devem ser incluídas, além de que, a malha final pode ficar altamente distorcida. Neste trabalho, rotinas desenvolvidas na linguagem de programação *python*(PYTHON, 2011) são criadas para automatizar a geração dos modelos de elementos finitos de dutos corroídos com uma malha suave, sem distorções.

A fim de otimizar os custos com manutenção nos dutos, a indústria moderna tem investido cada vez mais em sistemas *digital twin*, que são cópias digitais de uma entidade física, onde é possível monitorar e simular em tempo real a entidade em questão. Alguns destes sistemas utilizam inteligência artificial para o fornecimento de uma resposta rápida a problemas que venham acontecer durante operação.

1.2 OBJETIVOS

Este trabalho propõe um sistema híbrido que utiliza redes neurais profundas para predição da pressão de falha de dutos com corrosão complexa. Os defeitos de geometria real são parametrizados utilizando transformadas *wavelet*, através da filtragem das curvas de alta frequências. Como entrada para treinar a rede neural são utilizados resultados de análises via Método dos Elementos Finitos (MEF), os parâmetros do defeito de corrosão obtidos via transformada *wavelet* discreta e as propriedades do material do duto. O objetivo final deste trabalho é produzir um sistema que possa prever a pressão de ruptura de dutos com defeitos complexos de forma precisa e rápida.

1.3 METODOLOGIA

A metodologia para alcançar o objetivo proposto consiste basicamente em treinar uma rede neural profunda que foi viabilizada através da parametrização dos defeitos. Os principais objetivos são:

- Desenvolvimento de um código computacional que gera o modelo geométrico e sua malha de EF de dutos com defeitos de geometria complexa axissimétrica e tridimensional a partir de dados de espessuras remanescentes obtidos através de alguma inspeção no campo. Estes códigos foram implementados em Python utilizando apenas bibliotecas matemáticas e de dados.

- Desenvolver procedimentos de análise não linear utilizando o software livre code_aster (ASTER, 2019) e rotinas para gerar automaticamente os arquivos necessários para execução no cluster de computadores do grupo de pesquisa PADMEC.

- Validar os geradores de malhas e o procedimento de análise comparando os resultados das simulações via MEF com testes experimentais e avaliações através de métodos semi-empíricos.
- Utilizar transformada discreta *wavelet* para parametrizar os dados de perda de espessura e como banco de filtros para redução da quantidade de parâmetros.
- Testar qual tipo de *wavelet* pode ser utilizada para melhor representação dos perfis reais de corrosão axissimétricos e complexos.
- Gerar modelos sintéticos baseados nas estatísticas de perfis reais de corrosão, para que estes possam ser utilizados como dados substitutos e alimentar as redes neurais artificial. Também são consideradas as propriedades dos materiais que os dutos foram construídos.
- Treinar redes neurais profundas para que as mesmas possam prever com precisão a pressão de ruptura de dutos com modelos tridimensionais com defeitos reais.

1.4 ORGANIZAÇÃO DO TRABALHO

Este trabalho está dividido em oito capítulos e a bibliografia, além de conter apêndices e anexos. Na presente seção são apresentadas algumas das características gerais desta tese, além de descrever sua organização.

Os capítulos 1 e 2 aprsentam respectivamente a introdução e uma revisão bibliográfica com literaturas relacionadas ao tema.

O capítulo 3 apresenta as ferramentas criadas para gerar automaticamente modelos de elementos finitos axissimétricos e tridimensionais de dutos com defeitos de corrosão tipo *river bottom* e defeitos de geometria complexa.

No capítulo 4 são apresentados alguns aspectos sobre a realização da análise não linear, como o critério de convergência e as condições de contorno e carregamento que foram utilizadas além de características de implementação do programa code_aster.

Capítulo 5 apresenta a análise multiresolução com foco em transformada discreta *wavelets*. Uma das seções apresenta os resultados das análises não lineares realizadas utilizando a transformada com diversas famílias de *wavelets* e a parametrização de dados proposta.

O capítulo 6 mostra aspectos de inteligência artificial e de redes neurais profundas apresentando no final o algoritmo de treinamento de uma rede neural profunda.

No capítulo 7 são apresentados a validação dos modelos e dos procedimentos de análise, a validação da análise da transformada *wavelet* aplicada a dutos corroídos e os resultados obtidos da predição da pressão de ruptura para os casos estudados.

Finalizando, no capítulo 8 são apresentadas algumas conclusões e observações referentes às ferramentas criadas e os resultados da pressão de falha obtidos utilizando as redes neurais. Também são citados alguns trabalhos futuros que podem ser desenvolvidos a partir do tema.

Os apêndices A e B apresentam a teoria e equações matemáticas da transformada *wavelet* discreta e de redes neurais artificiais.

Nos anexos A e B estão incluídos um exemplo do arquivo de malha gerado pelas ferramentas e um exemplo do arquivo de configuração de análise do code_aster.

2 REVISÃO BIBLIOGRÁFICA

A literatura relacionada à avaliação estrutural de dutos corroídos é bastante ampla. Esta possui métodos de avaliação que incluem desde métodos analíticos, chamados de semi empíricos, análise via método dos elementos finitos, análise de confiabilidade e utilização de modelos substitutos. Alguns trabalhos, como (XU et al., 2017) e (SILVA; GUERREIRO; LOULA, 2007), já foram publicados utilizando modelos híbridos que utilizam elementos finitos e modelos substitutos na área de dutos corroídos, porém o presente trabalho se diferencia pela utilização de modelos sintéticos e aplicação em modelos de corrosão com geometria complexa bidimensional e tridimensional. Além disto, a utilização de análise multiresolução para parametrizar e filtrar os pontos de espessura remanescente também é inovadora. Os trabalhos que envolvem análise multiresolução em dutos corroídos normalmente são utilizados na área de inspeção para a verificação de anomalias presentes, como apresentado por (LOWE; EREN, 2002) e para identificar o tipo de corrosão tal como apresentado no trabalho de (WANG et al., 2013).

Neste capítulo também será comentado um artigo que trata de novos conceitos aplicados à quarta revolução industrial, também chamada de indústria 4.0. Um deles é conhecido como *Digital Twin* ((IYER, 2018)), que representa sistemas de monitoramento que utilizam cópias computacionais do sistema real. Alguns deles utilizam sistemas de aprendizado de máquina, que é um dos temas abordados nesta tese, e que pode ser aplicado a essas novas tecnologias. Também será descrito o trabalho de (LIU et al., 2017) sobre inspeção que utiliza transformada *wavelet* para descobrir a presença de trincas em dutos enterrados. Na última subseção serão apresentados trabalhos que utilizam modelos substitutos como *MT5 Tree* e redes neurais aplicados à avaliação de integridade de dutos corroídos com defeito(s) idealizado(s). Além destes, outros trabalhos relacionados à integridade de dutos serão apresentados.

2.1 DIGITAL TWIN

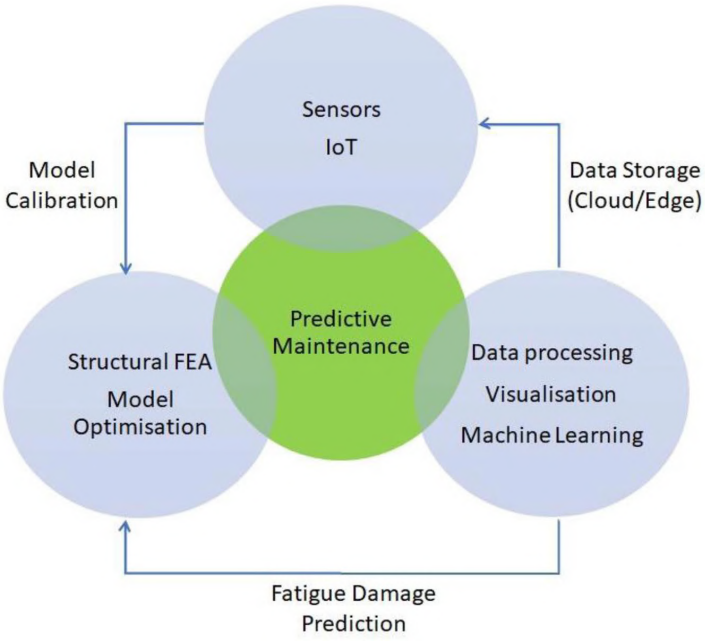
O primeiro trabalho aqui apresentado foi publicado por (KISHAWY; GABBAR, 2010), este trabalho apresenta uma revisão e análise de práticas de gerenciamento e técnicas de inspeção relacionadas à integridade da tubulação usando critérios de pressão interna. Para ajudar os engenheiros a manter a segurança da tubulação, o Manual de avaliação de defeitos em tubulações, (COSHAM; HOPKINS, 2002), apresenta procedimentos técnicos para avaliar tubulações com defeitos (corrosão, amassados, goivas, defeitos de solda, etc.).

Nos últimos anos, devido ao preço mais baixo do petróleo, os operadores de campo de petróleo e gás demonstraram interesse em sistemas *digital twin*, gêmeos digitais em

português, porque podem otimizar a estratégia de manutenção, aumentando a confiabilidade e reduzindo os custos financeiros que podem chegar a vários milhões de dólares (PIPELINE...,).

(BHOWMIK, 2019) publicou um trabalho apresentando um projeto conceitual de um sistema *digital twin* para dutos submarinos (*risers*). O projeto inclui o uso de técnicas de aprendizado de máquina e modelos de elementos finitos (FE) para criar, calibrar e prever mudanças no comportamento da tubulação. No geral o sistema de gêmeos digitais proposto é constituído de 3 componentes principais mostrados na Figura 1.

Figura 1 – Componentes conceituais do gêmeo digital.

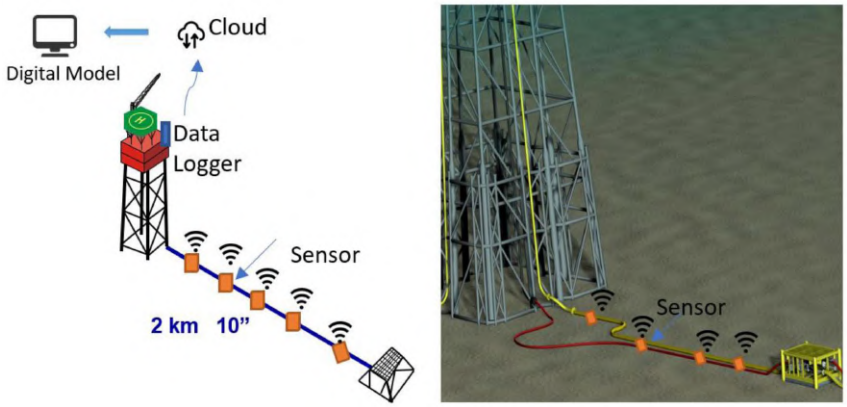


Fonte: Adaptado de Bhowmik (2019).

Um grande desafio para a implantação deste tipo de sistemas em dutos submarinos é a transmissão dos dados. Este trabalho propõe a instalação de *wireless data loggers* para a aquisição dos dados, onde estes são conectados a sistemas IoT que transmitem os dados armazenados na nuvem para análises, visualização e geração de relatórios. Um arranjo proposto dos sensores e transmissão de dados pode ser observado na Figura 2.

O trabalho é concluído com um sistema de predição de fadiga, que pode melhorar a programação das manutenções reduzindo o custo.

Figura 2 – Arranjo de sensores e transmissão de dados.



Fonte: Adaptado de Bhowmik (2019).

2.2 MÉTODOS SEMI-EMPÍRICOS

Nos programas de integridade de dutos é muito importante que os operadores tenham o máximo possível de informações sobre a atividade de corrosão em sua tubulação. A inspeção de dutos (ILI)que utilizam tecnologia ultrassônica, pode fornecer dados como um perfil *river bottom* (rbp), que é uma representação bidimensional detalhada da espessura da parede remanescente ao longo da tubulação. O rbp é uma projeção dos valores mínimos em toda a largura circunferencial. Com esses dados, podemos determinar a geometria da corrosão formada pelas profundidades circunferenciais dos picos, a espessura mínima restante da parede e o comprimento total do defeito. O trabalho de (CABRAL et al., 2022) apresentou um gerador de modelo automático para avaliação de dutos com defeitos com perfil *river bottom* utilizando o método dos elementos finitos. Os autores confrontaram os resultados com as pressões de falha de modelos tridimensionais e o erro entre os dois foi pequeno.

Métodos semi-empíricos avançados utilizados para o cálculo da pressão de ruptura consideram os perfis rbp da perda de material. Esses métodos produzem resultados mais precisos e menos conservadores quando comparados aos métodos de avaliação de defeitos simples (por exemplo, B31G (ASME, 2009), RSTRENG 0,85dL (KIEFNER; VIETH, 1989)) que consideram apenas o comprimento máximo e a profundidade máxima da área de corrosão. Os métodos avançados de avaliação que usam o rbp para calcular a pressão de falha de dutos corroídos são o *Effective Area* RSTRENG (KIEFNER; VIETH, 1989), Assessment of Complex Shape PART A e DNV-RP-F101 Assessment of Complex Shape PART B (DNV, 2017).

A avaliação da acurácia de métodos analíticos que são comumente utilizados na indústria foi realizado por (Abdalla Filho et al., 2014) utilizando elementos finitos. Este artigo propôs um método analítico para avaliação de dutos com defeitos com geometria

tipo *pit*.

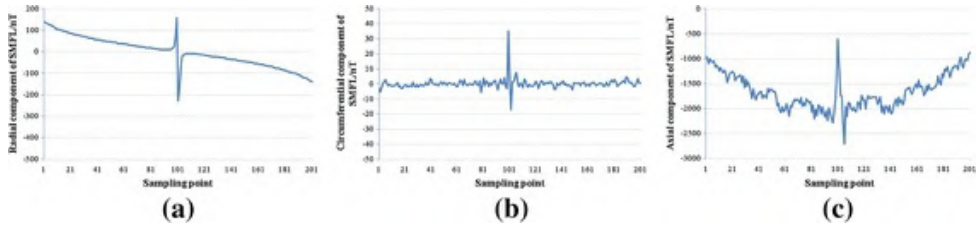
2.3 ANÁLISE MULTIRESOLUÇÃO

O número de medições de espessura de um defeito de corrosão pode variar entre dezenas e milhares de pontos. Nesta tese, a análise multiresolução é utilizada para parametrizar e reduzir o tamanho dos dados necessários para descrever a geometria do defeito, diminuindo a dimensão do espaço para treinamento e consulta das redes neurais. Atualmente, a análise multiresolução é usada em sistemas de detecção de corrosão para verificar anomalias e corrosão oculta, conforme mostrado no trabalho de (SILVA; GOUYON; LEPOUTRE, 2003).

A transformada *wavelet* discreta é também uma ferramenta muito poderosa para monitorar e identificar parâmetros de trinca utilizando redes neurais alimentadas por um sinal de teste de memória magnética do metal conforme publicado no trabalho de (LIU et al., 2017).

No trabalho publicado por (SONG et al., 2017), os autores realizam uma análise acoplada magneto-estrutural, que tem característica não linear, utilizando o método dos elementos finitos para investigar as três componentes do campo magnético de um duto submetido a uma carga magnética. A Figura 3 apresenta resultados da distribuição do *spontaneous magnetic flux leakage* (SMFL), que em português pode ser traduzido como vazamento espontâneo do fluxo magnético, obtido de uma análise via MEF.

Figura 3 – Componentes do campo SMFL do duto nas direções: a) radial, b) circunferencial e c) axial.

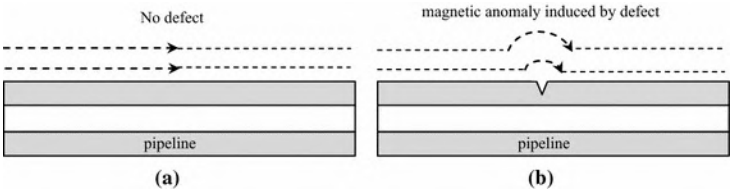


Fonte: Adaptado de Song et al. (2017).

Como os componentes do campo apresentaram picos onde o uma falha foi inserida no modelo MEF, os pesquisadores realizaram um teste experimental em um duto que possua duas trincas, conforme apresentado na Figura 4.

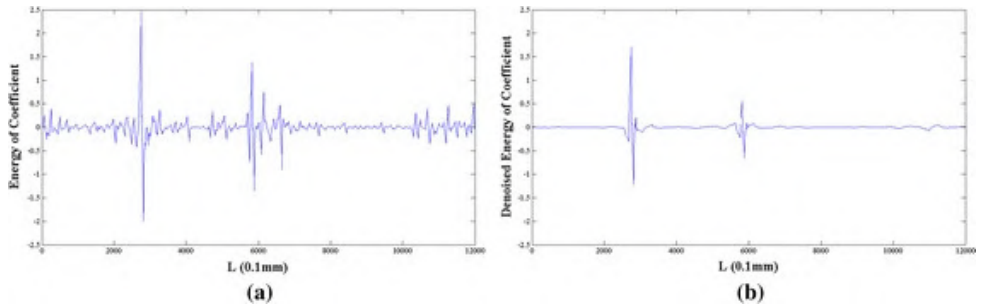
Transformada *wavelet* foi então utilizada para analisar, decompor e filtrar o sinal magnético obtido. A energia dos coeficientes foi calculada e uma remoção do ruído foi realizada. Os resultados apresentados na Figura 5 mostram os picos nos coeficientes na localização das trincas do experimento.

Figura 4 – Inspeção magnética de duto enterrado.



Fonte: Adaptado de Song et al. (2017).

Figura 5 – a) Coeficiente energético dos níveis 7 e 9 e b) Resultado da remoção de ruído.



Fonte: Adaptado de Song et al. (2017).

2.4 ANÁLISES VIA MEF

Análises usando o método de elementos finitos (MEF) demonstraram ser uma maneira poderosa e eficiente para a avaliação estrutural de dutos com defeitos causados por corrosão. Vários estudos demonstraram a eficiência de análises de elementos finitos de dutos com defeitos idealizados e múltiplos defeitos idealizados, como apresentado nos trabalhos de (CHOI et al., 2003), (ANDRADE et al., 2008), (CABRAL, 2007), (PATIL, 2012), (MOTTA et al., 2017) e (BRUÈRE et al., 2019). A análise de dutos com defeitos complexos de corrosão demonstrou ser muito precisa nos trabalhos de (SOUZA, 2008), (FERREIRA, 2011) quando comparada com resultados de testes experimentais publicados por (SOUZA et al.,) nesta e nas seguintes referências (SOUZA, 2003; SOUZA et al., 2004; SOUZA et al., 2005).

No trabalho desenvolvido por (PIMENTEL et al., 2020) foi apresentada uma ferramenta capaz de gerar automaticamente modelos de elementos finitos tridimensionais com defeitos de corrosão complexa que são construídos utilizando dados de inspeção ultrassônica. Os resultados encontrados utilizando a ferramenta desenvolvida foram comparados a métodos semi-empíricos e a resultados experimentais. A diferença entre o resultado da análise não linear e os resultados experimentais ficou entre 0.87 e 15.4%. Os resultados dos métodos analíticos foram mais conservadores em todos os casos estudados.

O trabalho de (FERREIRA et al., 2021) apresenta um procedimento para geração

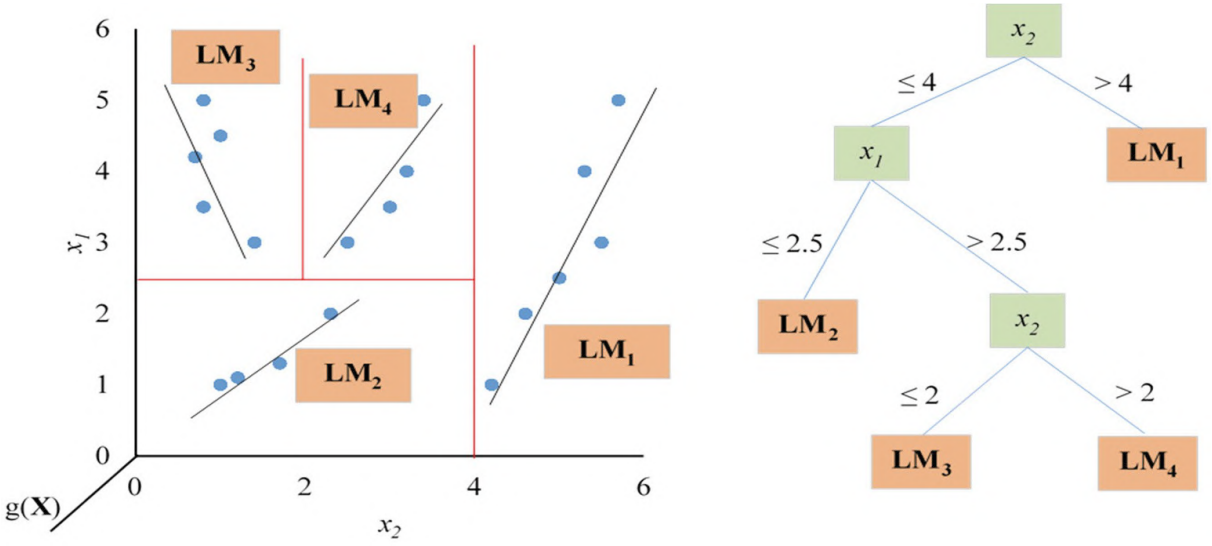
automática de modelos sintéticos de dutos com defeitos complexos. Um ajuste estatístico é realizado no mapa de corrosões utilizando a distribuição Valor Extremo Generalizada e uma função de correlação espacial do tipo exponencial é aplicada aos pontos aleatórios. Após isso uma análise estatística utilizando método de Monte Carlo é utilizada e a distribuição de falha é comparada com os parâmetros espessura mínima e média do defeito. No trabalho de (BAO; ZHOU, 2021) modelos aleatórios são gerados utilizando uma distribuição lognormal deslocada e uma função de correlação espacial do tipo exponencial. Os modelos são comparados com casos reais e bons resultados foram encontrados.

2.5 EMPREGANDO METAMODELOS

Alguns trabalhos utilizam metamodelos para avaliação da integridade estrutural de dutos corroídos. O trabalho publicado por (SEGHIER et al., 2018) realiza uma análise de confiabilidade de dutos que transportam gás com defeito de corrosão na superfície externa do duto. O método utilizado para a análise de confiabilidade é o Monte Carlo (MC). Este método é bastante adequado para problemas altamente não lineares, pois não apresenta instabilidades, porém, pode ter um custo computacional alto em problemas com baixa probabilidade de falha. Com o objetivo da redução deste custo, os autores propõem utilizar o metamodelo junto ao MC para calcular a probabilidade de falha em dutos corroídos fabricados com material API 5LX60.

A estrutura básica do modelo *M5 Tree* pode ser observada na Figura 6 e os autores concluem que a utilização do metamodelo resultou em uma melhora no desempenho do cálculo da probabilidade de falha de dutos com defeitos idealizados.

Figura 6 – Estrutura do M5 Tree: a) dados em sub-regiões; b) modelo linear para sub-regiões.



Fonte: Adaptado de Seghier et al. (2018).

Redes neurais artificiais (RNA) foram utilizadas para modelar o crescimento da corrosão e prever a taxa de crescimento em dutos, conforme publicado nos trabalhos de (SUPRIYATMAN et al., 2012) e de (DIN et al., 2015).

Os próximos trabalhos aqui descritos são similares em termos de utilizarem metamodelos e EF. Estes utilizam análises via métodos dos elementos finitos e redes neurais artificiais para a criação de metamodelos. Os metamodelos são responsáveis por prever a pressão de falha dos dutos. A grande vantagem em utilizar estes metamodelos é excluir a necessidade de gerar modelos e malhas de elementos finitos e evitar a realização da análise tridimensional não linear, que pode ter um custo computacional bastante alto.

(HAN; SHEN; DAI, 1996) utilizaram com sucesso uma rede neural artificial para prever o momento fletor que leva a falha de dutos corroídos submetidos à pressão interna e carga térmica. Estes dutos possuem trincas que atravessam totalmente a espessura do mesmo e trincas que possuem profundidade que atravessam parcialmente a espessura do duto. Os dados utilizados para o treinamento da rede vieram de experimentos e os resultados são comparados com o método da ASME (ASME. . . , 2010) e com um modelo analisado via MEF. Como conclusão os autores conseguiram prever com precisão o momento de falha utilizando a rede neural.

Redes neurais foram aplicadas no trabalho de (HAN; HAN; LIU, 1999) para prever a pressão interna de falha de dutos com defeitos idealizados. Setenta modelos experimentais publicados por (KIEFNER et al., 1973) foram utilizados para treinar a rede. Os resultados encontrados foram satisfatórios sendo menos conservadores que o encontrado utilizando normas. A rede neural também foi utilizada para analisar a sensibilidade dos parâmetros do duto e do defeito e os autores apresentaram a influência destes na pressão de falha.

O trabalho publicado por (SILVA; GUERREIRO; LOULA, 2007), apresentou um rede neural artificial, que foi utilizada pra prever regras de interação entre dois defeitos em dutos corroídos. Os casos estudados possuem dois defeitos idealizados, alinhados longitudinalmente ou circunferencialmente. A ideia foi treinar uma rede neural para prever a relação entre a pressão de falha de um defeito isolado e dois defeitos alinhados, pois defeitos isolados são mais simples de calcular, então, sabendo-se a relação entre as pressões fica fácil prever a pressão de falha de dutos com defeitos alinhados.

Neste trabalho os autores propuseram uma equação para o cálculo da pressão de falha em dutos com defeitos simples com comprimento de $80mm$ e largura de $32mm$. A mesma foi validada com os próprios resultados obtidos por elementos finitos e com o cálculo adotado pela norma Det Norske Veritas (DNV), que pode ser encontrado na referência (DNV, 2017).

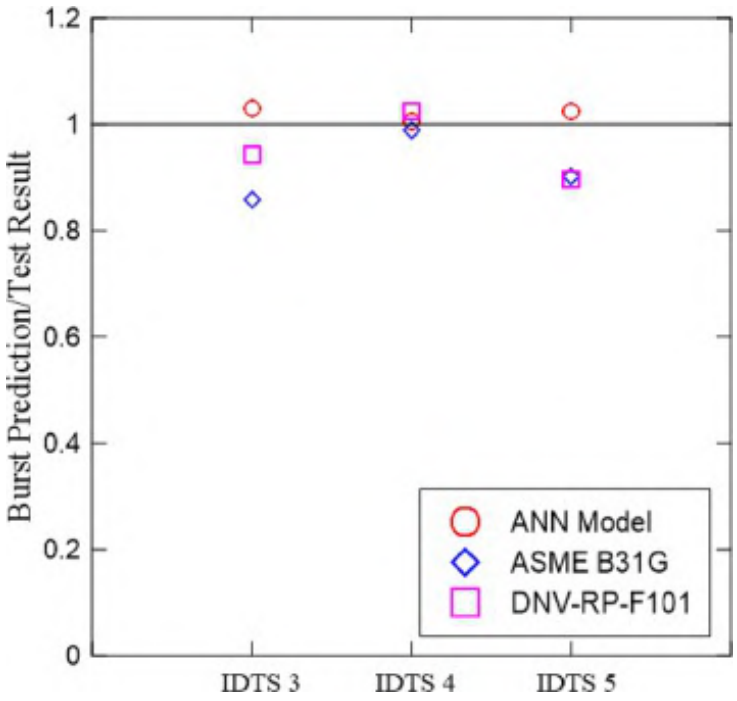
Com os resultados da análise MEF, uma rede neural com 3 camadas foi proposta. A primeira camada com 2 neurônios, a camada intermediária com 3 neurônios e a camada

de saída com 1 neurônio. Os autores conseguiram predizer as regras de interação entre defeitos, porém concluíram que mais estudos são necessários para generalização destas regras para outros casos.

O artigo publicado por (XU et al., 2017), também utilizou redes neurais artificiais para a predição de falha em dutos corroídos. Neste artigo foram considerados múltiplos defeitos idealizados com posições arbitrárias no duto. Para o estudo 5 modelos experimentais foram utilizados para validação, estes são nomeados da seguinte forma: IDST3, IDST4, IDST5, IDST6 e IDST7. O material de fabricação desses dutos foi o aço estrutural API 5LX80.

O experimento destes espécimes e solução via modelos de elementos finitos validados já haviam sido apresentados nos trabalhos desenvolvidos por (BENJAMIN et al., 2005) ,(ANDRADE et al., 2008), (ANDRADE et al., 2006) e (MOTTA et al., 2017). Os resultados obtidos pela predição da rede foram confrontados com resultados obtidos por normas como a B31G e a DNV-RP-F101. Um gráfico com os três resultados relacionando a pressão predita com o resultado experimental é apresentado na Figura 7.

Figura 7 – Resultado da predição pela Rede Neural.



Fonte: Adaptado de Xu et al. (2017).

Os autores concluíram a aplicação com sucesso da rede neural para a predição de falha. O erro relativo foi menor que 2% e foi menor que os encontrados utilizando os métodos semi-empíricos das normas. Vale salientar que os resultados, em todos os casos, são não conservadores, pois ficaram um pouco acima da unidade, e para garantir a

segurança operacional dos dutos corroídos, um coeficiente de segurança adequado deve ser aplicado.

No trabalho de (KUMAR; KARUPPANAN; OVINIS, 2022) foi aplicado redes neurais para predição de dutos com defeitos alinhados circunferencialmente. Os autores conseguiram prever com sucesso a falha de dutos construídos de material da classe API 5LX80 quando comparados a falhas obtidas via MEF e obtendo um ajuste do R^2 de 0.99.

O trabalho aqui desenvolvido se diferencia pela utilização de um sistema híbrido constituído por modelos de EF, transformada *wavelet* discreta e redes neurais para a predição rápida e precisa da pressão de falha de dutos com defeitos de geometria complexa. Esta tese desenvolveu modelos híbridos capazes de avaliar tanto defeitos axissimétricos *river bottom* quanto defeitos tridimensionais, considerando a geometria do duto e diversos tipos de materiais.

3 GERAÇÃO E DISCRETIZAÇÃO DE MODELOS

Conforme apresentado no capítulo 2, o método dos elementos finitos, se utilizado corretamente, é capaz de alcançar resultados muito próximos aos encontrados nos experimentos de laboratório. O modelo híbrido proposto nesta tese utilizará simulações numéricas via MEF para criar os modelos sintéticos que serão utilizados para treinamento, teste e validação da rede neural.

Este capítulo apresenta algumas características da geração e da discretização do modelo de elementos finitos. Neste trabalho foram utilizados modelos axissimétricos e tridimensionais. A aproximação axissimétrica é uma prática comum na indústria e para o caso de modelos de elementos finitos, representam um menor custo computacional em relação a modelos tridimensionais. O resultado da pressão de falha calculada via MEF para esses modelos tende a ser conservadora em relação aos resultados encontrados com modelos tridimensionais, pois apresentam uma corrosão igual ao longo de toda circunferência do duto.

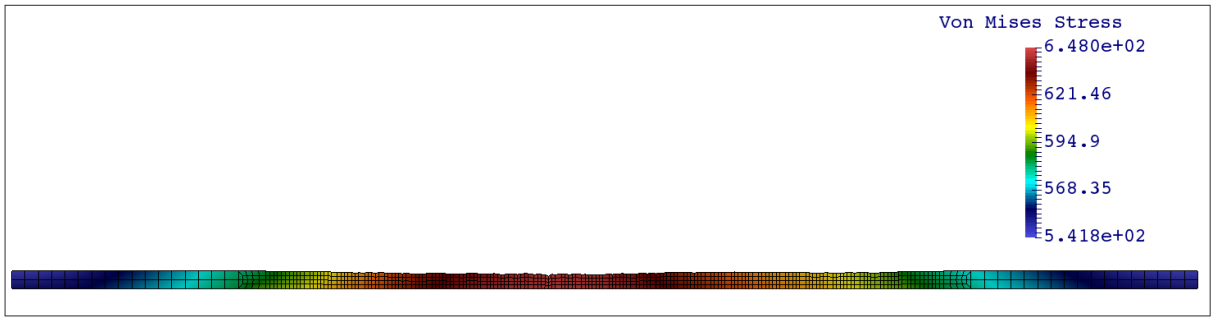
Como diversos modelos foram necessários para a realização dos estudos deste trabalho, rotinas em Python foram criadas para a geração automática de modelos de elementos finitos. Para os modelos axissimétricos as rotinas podem gerar tanto defeitos idealizados quanto com perfil *river-bottom*. As malhas possuem uma região de transição dos elementos ao longo da espessura, diminuindo a densidade da malha e o custo computacional para a simulação. As condições de contorno e carregamento também são definidas e automaticamente aplicadas, podendo assim, representar diversas condições reais aos quais os dutos podem ser submetidos. Nos modelos tridimensionais, nenhuma região de transição foi realizada, pois a implementação destas custaria um tempo que inviabilizaria a conclusão deste trabalho dentro do cronograma. A aplicação das condições de contorno e carregamento também são aplicadas automaticamente, permitindo avaliar dutos em condições de experimento que possuem efeito de tampa e dutos com restrição de deslocamento longitudinal, como no caso de dutos enterrados.

As rotinas geradas permitem escrever automaticamente as coordenadas dos nós, a definição dos elementos e as condições de contorno e carregamento em arquivos prontos para serem executados em dois possíveis *solvers* de elementos finitos. Estes programas são o code_aster (ASTER, 2019) e o *Calculix* (DHONDT, 2011), ambos são pacotes de código aberto com rotinas para solução de problemas termomecânicos e estruturais utilizando o MEF. Neste trabalho todas as análises foram realizadas utilizando o code_aster.

A Figura 8 apresenta um modelo axissimétrico com defeito de corrosão no formato *river-bottom* simulado utilizando um utilizando o pacote code_aster. Todos os processos de

geração e o procedimento da análise não linear são realizados automaticamente utilizando as rotinas em Python.

Figura 8 – Resultado de uma análise de um modelo axissimétrico com defeito *river bottom* utilizando o programa code_aster.



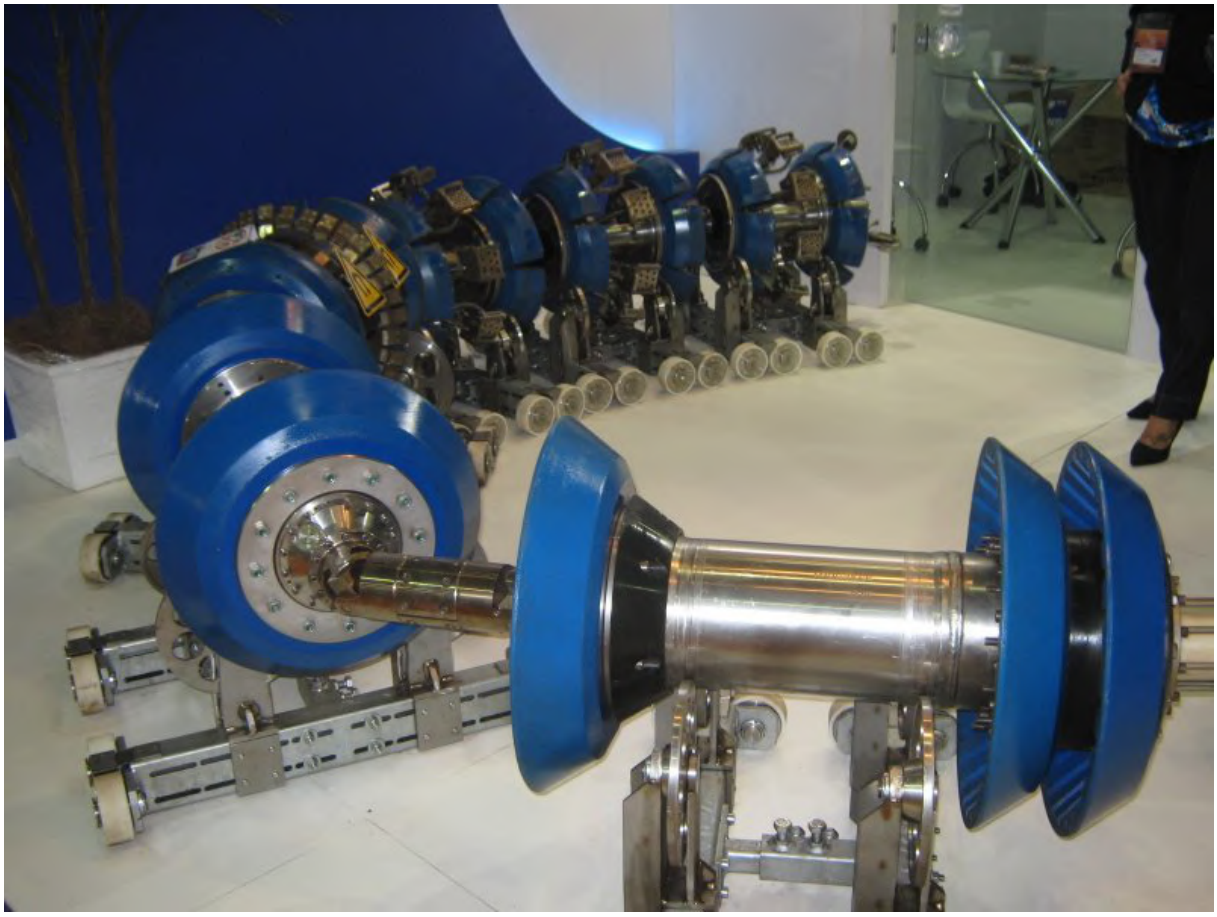
Fonte: O autor (2021).

Este trabalho se divide em duas partes: A primeira focou na utilização de modelos axissimétricos com defeitos tipo *river bottom*, pois estes geram resultados mais próximos aos experimentais em relação aos modelos idealizados e são mais econômicos que modelos tridimensionais. Na segunda parte modelos tridimensionais foram testados pois estes representam uma geometria mais próxima aos casos que temos resultados experimentais. O *solver* utilizado para obter os resultados da pressão de ruptura foi o code_aster, portanto, as seções a seguir darão ênfase a estes temas.

3.1 MODELOS COM PERFIL DE CORROSÃO *RIVER BOTTOM*

O pig de inspeção ultrassônico, Figura 9, fornece uma grade das medições de espessura remanescentes, alinhadas ao longo das direções circunferencial e longitudinal. Um perfil *river bottom* é um perfil bidimensional da superfície de corrosão tridimensional, obtida pela seleção da espessura remanescente mínima em cada linha circunferencial da grade de corrosão completa.

Figura 9 – Ilustração de um pig de inspeção ultrassônica.



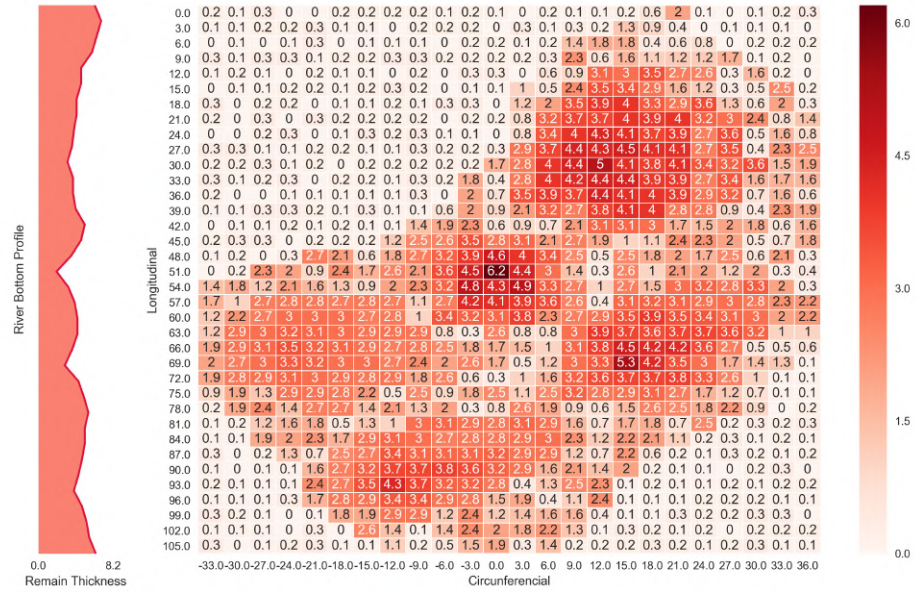
Fonte: O autor (2021).

A Figura 10 mostra um exemplo dos dados completos obtidos pela inspeção ILI (*in-line inspection*) e sua representação do perfil *river bottom*. Para obter esse perfil, podemos colocar o mapa inteiro da espessura remanescente da parede do duto em uma representação matricial, representada por $\mathbf{T}$, Equação (3.1). Portanto, a operação que pode gerar a matriz rbp $\mathbf{t}$ pode ser matematicamente representada pela Equação (3.2).

$$\mathbf{T} = \begin{bmatrix} t_{11} & t_{12} & \dots & t_{1n} \\ t_{21} & t_{22} & \dots & t_{2n} \\ \vdots & & \ddots & \vdots \\ t_{m1} & t_{m2} & \dots & t_{mn} \end{bmatrix} \tag{3.1}$$

$$\mathbf{t} = \begin{bmatrix} min(t_{11}, t_{12}, ..., t_{1n}) \\ min(t_{21}, t_{22}, ..., t_{2n}) \\ \vdots \\ min(t_{m1}, t_{m2}, ..., t_{mn}) \end{bmatrix} \tag{3.2}$$

Figura 10 – Dados completos e representação do perfil river bottom.



Fonte: O autor (2021).

Onde m e n representam respectivamente a posição do ponto nas direções longitudinal e circunferencial. Neste trabalho, para calcular a pressão de falha de um duto corroído, representado por seu rbp, é utilizada a formulação axissimétrica de elementos finitos.

Como já citado anteriormente, rotinas em python foram criadas para gerar os modelos de elementos finitos do duto corroído com o perfil *river bottom*. A entrada das rotinas é o mapa de corrosão 2D completo ou o perfil de espessura de corrosão. Este mapa inclui a profundidade mínima na direção circunferencial ao longo da direção longitudinal do duto, um exemplo é mostrado na Tabela 1. Também é necessário fornecer o diâmetro

externo (D_e), a espessura nominal da tubulação (t) e o comprimento total da seção do duto para a construção completa do modelo. O código também aceita o diâmetro interno no lugar da espessura nominal nos dados de entrada.

Tabela 1 – Dados na forma de perfil *river bottom*.

Posição Longitudinal [mm]	Espessura [mm]
0.00	13.987
2.20	14.018
3.80	14.025
4.50	14.050
5.70	14.015
6.20	14.005
7.20	13.080
8.30	13.065
9.60	13.085

Fonte: O autor (2021).

Os *scripts* criam um perfil suave usando curvas do tipo *spline* cúbicas paramétricas, a cada dois pontos de espessura, para interpolar a localização dos nós da malha com base nos dados rbp. As curvas do tipo *splines* cúbicas, criadas para cada trecho, podem ser definidos pela equação (3.3) (MSC.PATRAN, 2015).

$$Z(\xi_1) = S_1\xi_1^3 + S_2\xi_1^2 + S_3\xi_1 + S_4\xi_1 \tag{3.3}$$

em que $Z(\xi_1)$ representa a coordenada geral das coordenadas globais X, Y e Z; S_1, S_2, S_3, e S_4 são constantes arbitrárias e ξ_1 é um parâmetro no intervalo de $0 \leq \xi_1 \leq 1$. Essas curvas cúbicas naturais garantem a continuidade da primeira e da segunda derivada em seus limites.

3.1.1 Geração da malha

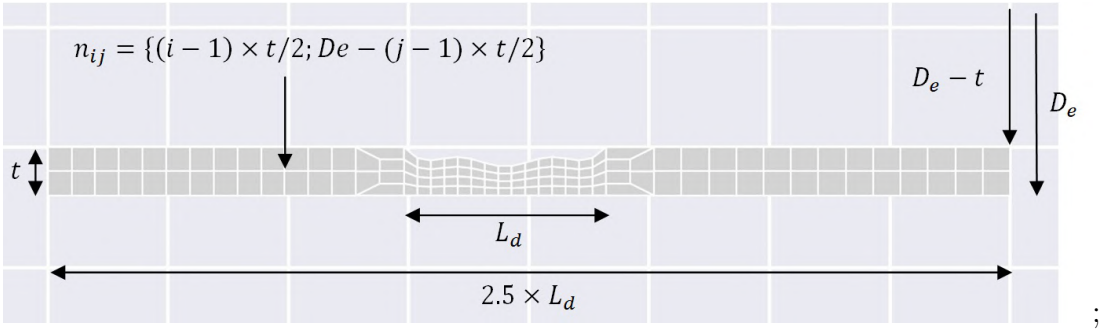
A geração automática da malha é realizada através do cálculo da posição dos nós e posterior definição dos elementos. O primeiro passo é realizado através da interpolação cúbica dos pontos na espessura da região corroída. Na sequência, pontos são criados equidistantes em $a_r \times t/4$, onde t representa a espessura nominal do duto e a_r é uma sigla para *aspect ratio* ou razão de aspecto em português. Isso permite diminuir a densidade de elementos na direção longitudinal da região corroída. Neste trabalho todos os modelos axissimétricos foram gerados com a razão de aspecto definida como 1.

Os outros nós da região corroída, posicionados ao longo da espessura, são calculados através de interpolação linear entre o ponto de corrosão e a parede oposta não corroída. No total mais quatro nós são gerados para que o modelo final possua 4 elementos ao longo da espessura.

Os demais nós, fora da zona corroída, podem ser calculados utilizando relações algébricas com os dados do duto e do defeito como o diâmetro externo (D_e), a espessura nominal do duto (t), e o comprimento do defeito (L_d). Este último é obtido diretamente do mapa de inspeções que contém os pontos de corrosão. O comprimento total do modelo é definido como $2.5 \times L_d$.

Um exemplo do cálculo das coordenadas de um nó qualquer n_{ij} a esquerda da região do defeito, pode ser observado na Figura 11.

Figura 11 – Cálculo das posições dos nós.



;

Fonte: O autor (2021).

Os nós criados após a região do defeito podem ser calculados de maneira semelhante ao da Figura 11, somando-se a coordenada horizontal um valor equivalente a $1.5 \times L_d$. Após todos os nós criados os elementos são definidos nas sub-regiões limitadas por 4 nós. Nos modelos prontos para serem analisados no code_aster, elementos unidimensionais precisam ser definidos para aplicação dos carregamentos.

Para reduzir o custo computacional das análises via MEF, uma malha fina de EF é criada na área corroída e uma malha mais grossa é criada longe da corrosão. Os scripts geram automaticamente essa transição suave entre essas duas regiões.

3.1.2 Geração dos modelos sintéticos

Na utilização de redes neurais, a qualidade dos dados utilizados para treinamento é de fato mais importante que a sofisticação do código utilizado, desta forma é de extrema importância ter exemplos de alta qualidade e que representem bem o problema real. Como não há disponibilidade de casos reais suficientes para alimentar a rede neural, neste trabalho casos sintéticos são utilizados.

Para prever a pressão de ruptura em dutos com um perfil de corrosão complexo, é necessária uma grande quantidade de dados, responsáveis por fornecer todas as informações que a rede neural precisa para aprender e gerar uma capacidade preditiva adequada. Neste trabalho, modelos sintéticos são gerados e utilizados como conjuntos de dados de treinamento e validação da rede neural. No total, 9612 casos sintéticos foram gerados e usados como parâmetros de entrada para a rede neural profunda e aproximadamente oitocentos (801) para cada um dos 12 perfis *river bottom* (Figura 12) obtidos por inspeção ultrassônica. Os modelos foram gerados aleatoriamente usando os parâmetros estatísticos obtidos ajustando cada perfil de corrosão.

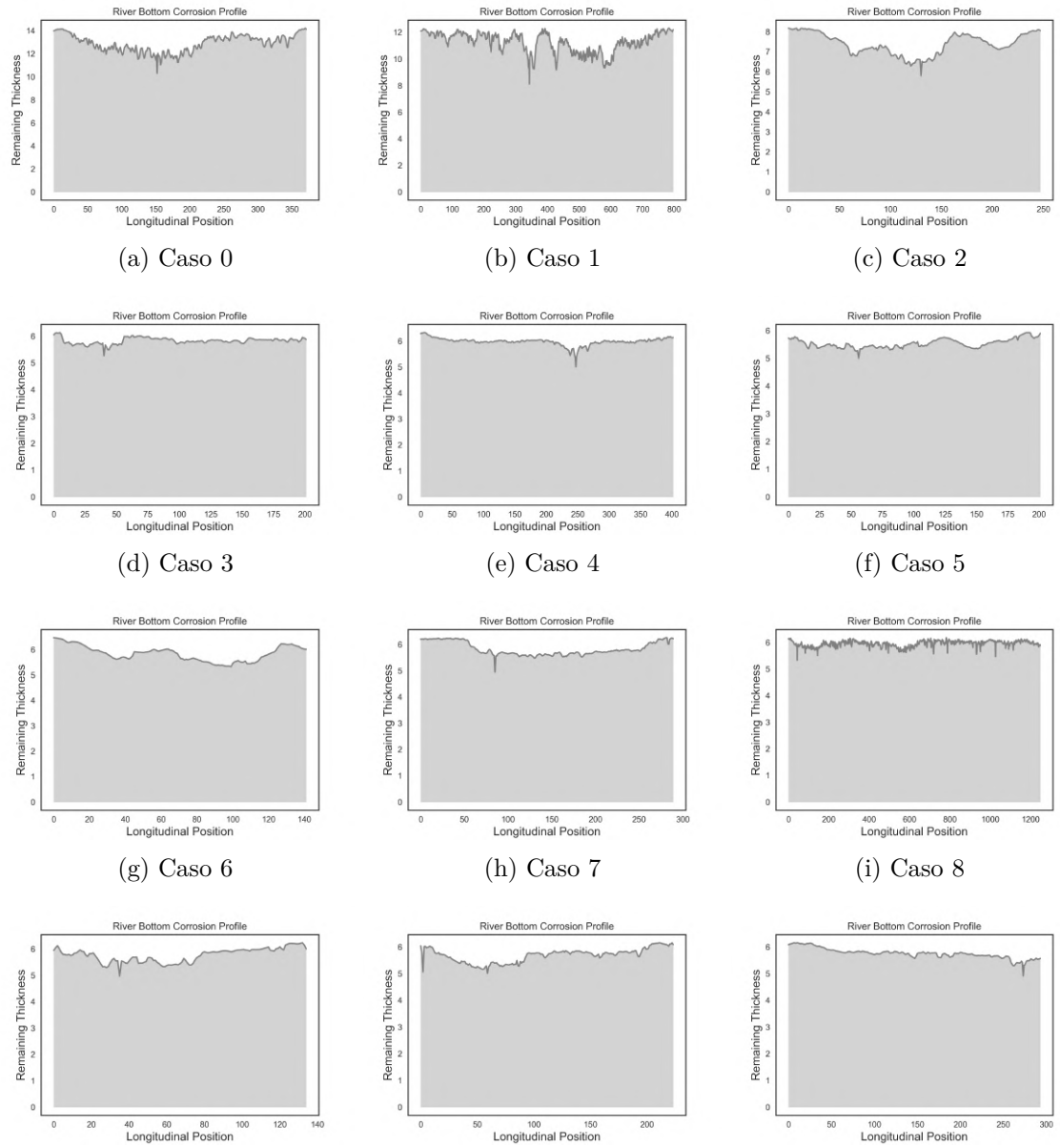
Todo defeito de corrosão real tem um perfil diferente. Para gerar defeitos sintéticos que são de alguma forma representativos de defeitos reais, são criados perfis aleatórios, cujas distribuições de profundidade têm propriedades estatísticas semelhantes às de defeitos reais de corrosão. As profundidades aleatórias são geradas de acordo com distribuições estatísticas ajustadas a defeitos reais de corrosão. Para escolher a distribuição que melhor se ajustou a cada perfil, a soma dos erros ao quadrado (SSE) entre o histograma das espessuras remanescentes no perfil *river bottom* $\mathbf{t}$ e a função densidade de probabilidade (pdf) ajustada $\hat{\mathbf{t}}$ é calculada, Equação (3.4). Para estimar os parâmetros de cada distribuição de probabilidade e criar uma função de densidade de probabilidade, o *Maximum LikeLihood Estimation* (ENDERLEIN, 1964) foi utilizado.

$$\mathrm{SSE} = \sum_{i=1}^{n} \left(t_i - \hat{t}_i\right)^2 \tag{3.4}$$

A Figura 13 mostra um exemplo de resultado da melhor distribuição ajustada para o caso 2. Para este caso, a distribuição Geral de Valor Extremo resultou um menor SSE entre todas as testadas.

O procedimento é realizado para todos os doze perfis *river bottom*. A Tabela 2 mostra a melhor distribuição ajustada para cada caso e estimativas para localização, escala e outros parâmetros presentes na distribuição.

Figura 12 – Perfis *river-bottom* utilizados no trabalho.



(a) Caso 0 (b) Caso 1 (c) Caso 2

(d) Caso 3 (e) Caso 4 (f) Caso 5

(g) Caso 6 (h) Caso 7 (i) Caso 8

(j) Caso 9 (k) Caso 10 (l) Caso 11

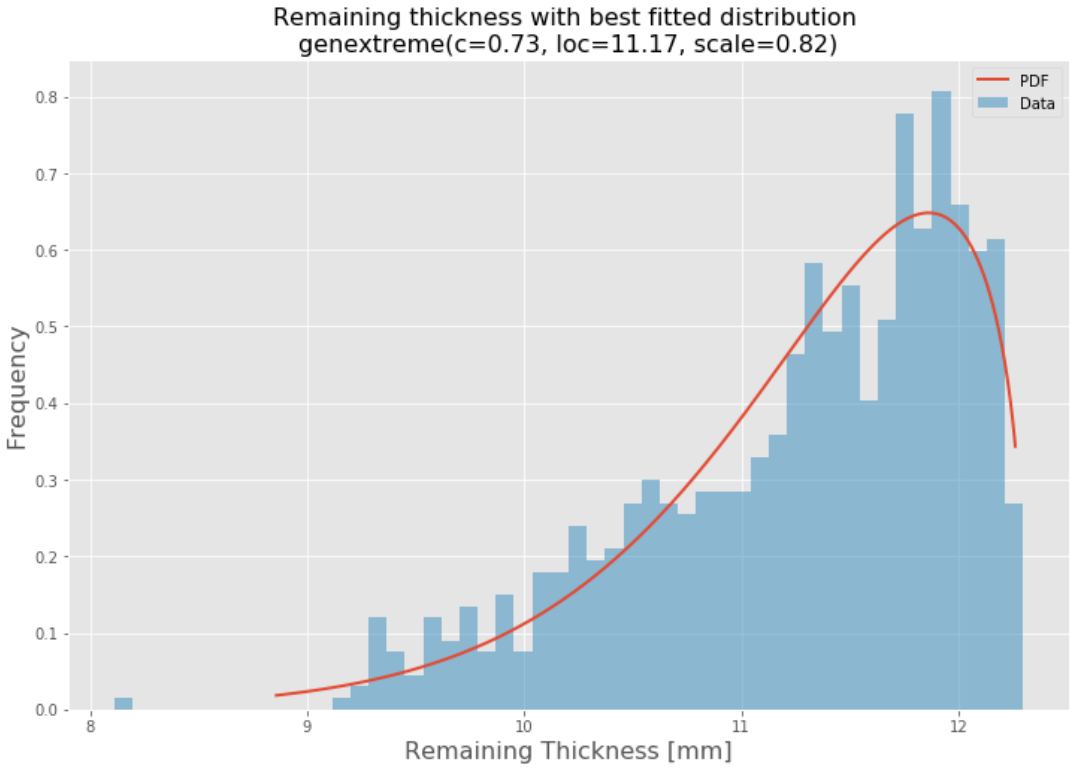
Fonte: O autor (2021).

Com a distribuição e os parâmetros, todos os 9612 perfis sintéticos de *river bottom* foram gerados aleatoriamente e usando o gerador de malha escrito em Python, descrito neste capítulo, foram criados os modelos de elementos finitos com condições de contorno prescritas que foram posteriormente submetidos à análise não linear no software code_aster.

3.2 MODELOS COM PERFIL DE CORROSÃO TRIDIMENSIONAL COMPLEXA

Nesta seção será apresentado o processo de geração de modelos com defeitos tridimensionais complexos de corrosão. Estes modelos tridimensionais tendem a gerar resultados mais próximos aos encontrados atraveś de experimentos. Nas Figuras 14a e são

Figura 13 – Melhor distribuição ajustada para o caso 2.



Fonte: O autor (2021).

Tabela 2 – Distribuição e parâmetros para cada caso.

caso	distribuição	param#01	param#02	loc	scale
0	expweib	0.4	8.1	10.1	3.6
1	genextreme	-	0.7	11.2	0.8
2	expweib	1e-2	195.1	5.7	2.5
3	loggamma	-	19.4	4.3	0.5
4	expweib	2.5	2.3e7	-4.4e6	4.4e6
5	invweib	-	1.9e8	-2.9e7	2.9e7
6	expweib	0.1	7.3	5.33	1.0
7	expnorm	-	2.6	5.6	0.1
8	expweib	0.4	16.1	5.1	1.0
9	expweib	0.3	8.2	4.9	1.1
10	loggamma	-	4.5	4.9	0.5
11	expweib	10.1	40.6	12.7	18.0

Fonte: O autor (2021).

apresentados exemplos de defeitos de corrosão reais encontrados em campo.

Figura 14 – Exemplo de Defeitos Reais

(a) Exemplo de defeito #01

(b) Exemplo de defeito #02

Fonte: https://www.steeldata.info/macro/demo/data/2751.html.

3.2.1 Geração da malha

Da mesma forma que para defeitos de corrosão *river_bottom*, rotinas em Python foram criadas para a geração automática dos modelos tridimensionais. Cálculos geométricos foram utilizados para encontrar as posições dos nós e os elementos foram definidos consequentemente após a definição dos nós. O processo automático de geração consiste inicialmente na leitura do arquivo com os dados de inspeção, da mesma forma que é feita com os defeitos tipo *river-bottom*. Com as coordenadas dos pontos de espessura remanescente são gerados os nós da região do defeito através de interpolação nas coordenadas desejadas. Os pontos interpolados são gerados através do ajuste de *splines* bicúbicas utlizando Mínimos Quadrados (DIERCKX, 1981; ABBASSL et al., 1991). A representação da *b_spline* é dada através da equação 3.5.

$$s(x,y) = \sum_{i=1}^{g+4} \sum_{j=1}^{h+4} C_{ij} M_i(x) N_j(y) \tag{3.5}$$

onde $M_i(x), N_j(y)$ são normalizadas *b_splines* cúbicas e o coeficiente C_{ij} é determinado como a solução do seguinte problema de otimização:

Minimize η sujeito a restrição

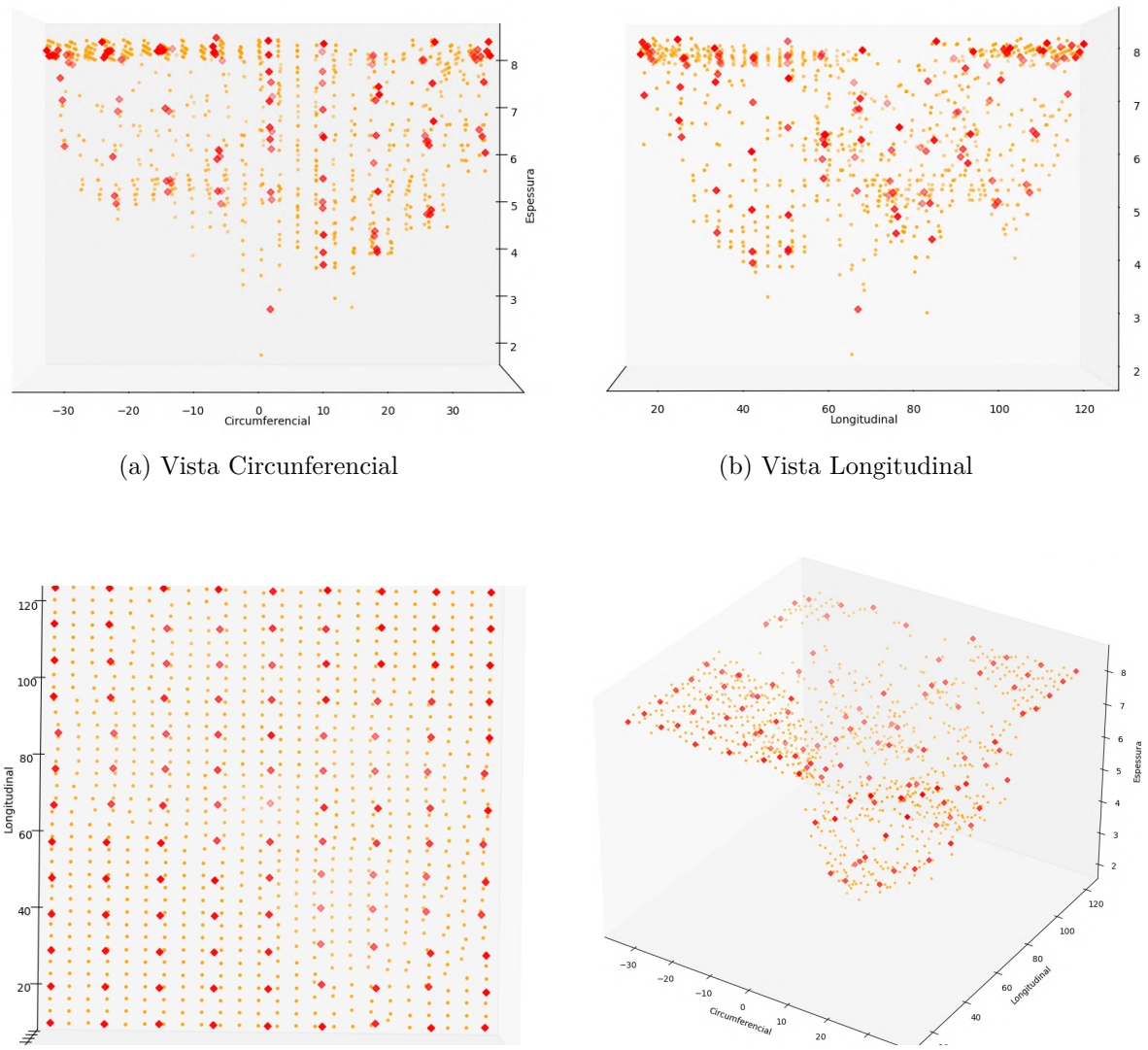
$$\epsilon_r^2 < S \tag{3.6}$$

onde η é a medida da falta de suavização de $s(x,y)$, ϵ_r representa o resíduo ponderado $\omega_r(f_i - s(x,y))$ e S é um número não negativo que controla a suavização da *spline*, chamado de fator de suavização.

O resultado da interpolação dos pontos para um caso selecionado é apresentada na Figura 15, onde os pontos originais são representados como círculos laranjas e os pontos interpolados são representados como losangos vermelhos. A distância entre pontos utilizada

para a interpolação levou em conta a discretização da malha que possui quatro elementos ao longo da espessura do duto.

Figura 15 – Interpolação dos nós em relação a espessura medida.



(a) Vista Circunferencial

(b) Vista Longitudinal

(c) Vista Superior

(d) Vista em Perspectiva

Fonte: O autor (2021).

Na Figura 15a tem-se a projeção na direção circunferencial do defeito, na Figura 15b tem-se a vista da interpolação no sentido longitudinal, na Figura 15c observa-se a vista superior onde fica claro a distribuição equidistantes dos pontos. Na Figura 15d temos a perspectiva que nos dá uma visão tridimensional do resultado da interpolação cúbica.

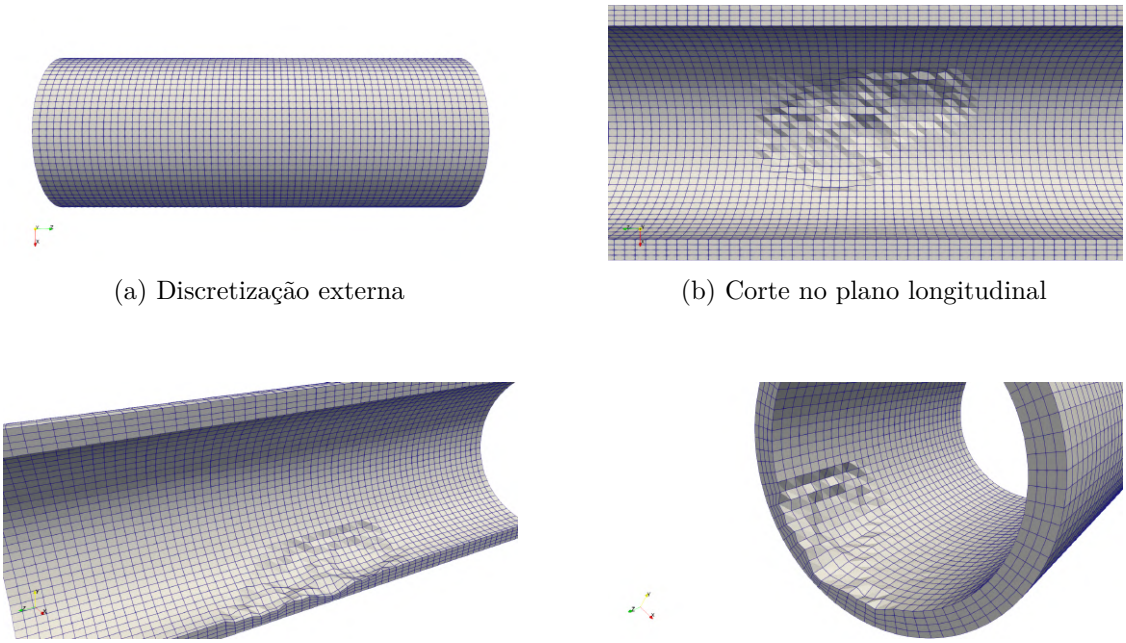
Os demais nós da região corroída, posicionados ao longo da espessura, são calculados da mesma forma que no modelo *river bottom* interpolando linearmente o ponto de corrosão e a parede oposta não corroída. No total quatro nós são gerados para que o modelo final possua 4 elementos ao longo da espessura.

Utilizando equações algébricas e utilizando interpolação linear as coordenadas dos

nós fora da região do defeito são calculados.

Após a definição dos nós, os elementos hexaédricos são definidos e armazenados na forma de um dicionário. Para aplicar o carregamento de pressão, elementos planos de quatro nós precisam ser criados, isto é realizado utilizando elementos lógicos e o banco de dados criado anteriormente. Ao final uma malha de elementos devidamente suavizada na região do defeito é gerada. Essa suavização evita problemas numéricos no momento da análise não linear. Um exemplo de uma malha criada é apresentado na Figura 16.

Figura 16 – Malha de Elementos Finitos.



(a) Discretização externa

(b) Corte no plano longitudinal

(c) Corte no plano vertical

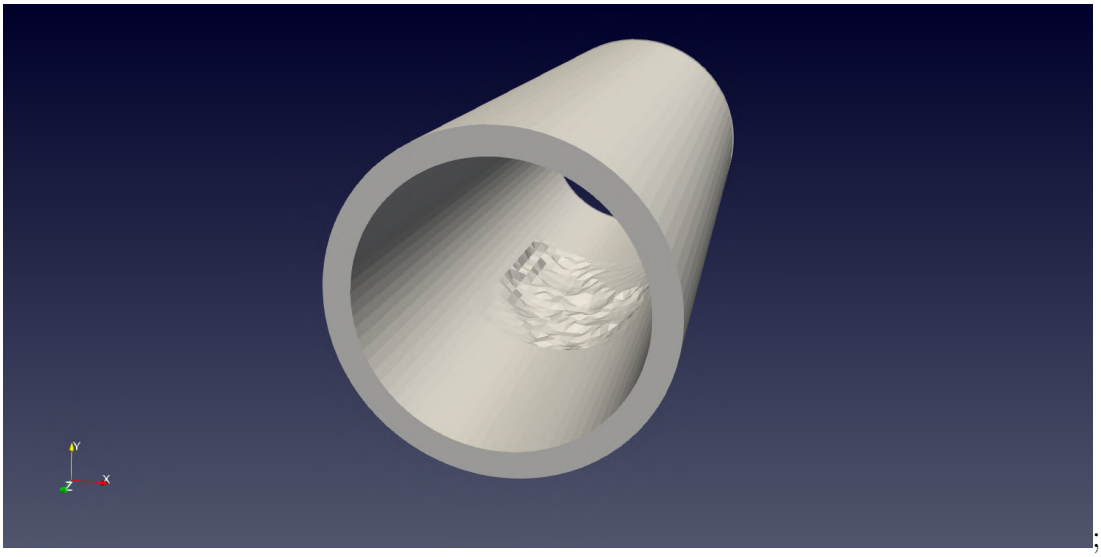
(d) Corte na seção do duto no centro do defeito

Fonte: O autor (2021).

Na Figura 16a temos a visualização da discretização na região oposta a posição do defeito. Figuras 16b, 16c e 16d representam cortes nos planos horizonal vertical na seção do duto no centro do defeito respectivamente.

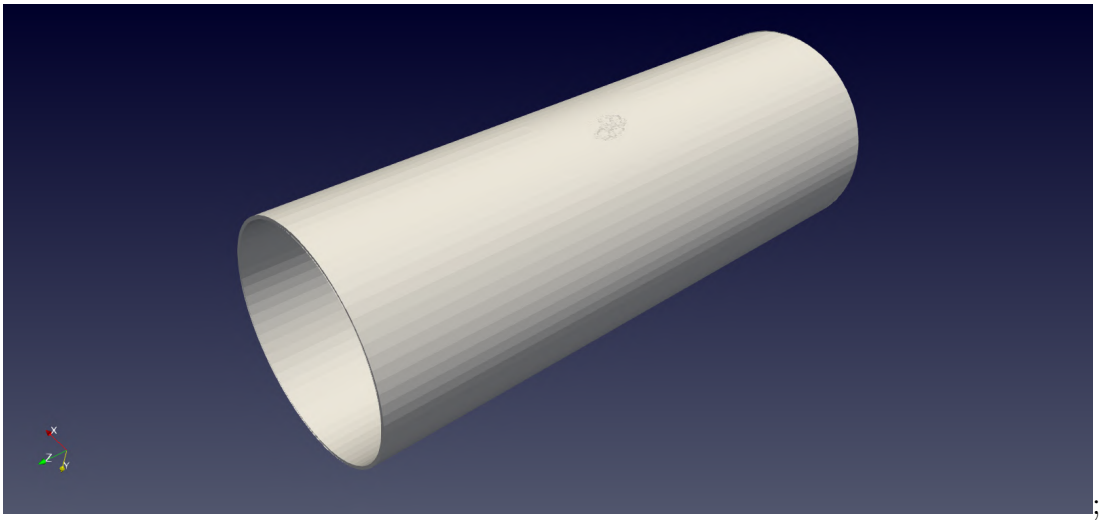
A ferramenta desenvolvida permite a geração de defeitos localizados tanto na superfície interna quanto na superfície externa do duto. Exemplos destes defeitos sobre a geometria de um duto podem ser observados nas Figuras 17 e 18.

Figura 17 – Defeito de corrosão interna.



Fonte: O autor (2021).

Figura 18 – Defeito de corrosão externa.



Fonte: O autor (2021).

3.2.2 Geração dos modelos sintéticos

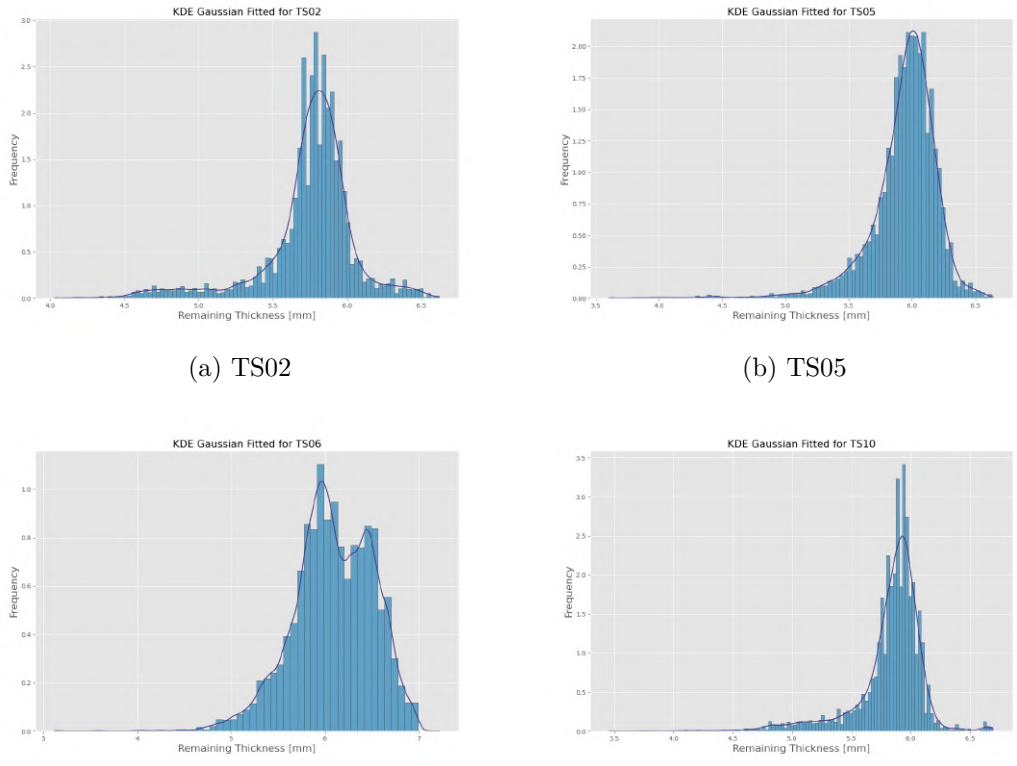
Tal como anteriormente mencionado, devido a escassez de dados reais para alimentar uma rede neural de forma que a mesma seja capaz de prever com precisão a pressão de falha de dutos corroídos, surge a necessidade de gerar defeitos sintéticos que sirvam como substitutos para estes dados. Neste trabalho campos estocásticos de perfis de espessura são gerados e utilizados para a geração de modelos de elementos finitos sintéticos de onde a pressão de falha será obtida via análise não linear via MEF. Diversos testes foram realizados até a definição do procedimento descrito a seguir.

A geração de modelos sintéticos tridimensionais se baseou no trabalho de (FERREIRA, 2011). O ajuste estatístico para os casos reais deste trabalho foi realizado utilizando *Kernel Density Estimation (KDE)*(PARZEN, 1962), pois esta estimativa foi a única que se demonstrou eficiente em ajustar mais fielmente os perfis reais complexos. KDE,é um estimador PDF baseado em observações locais. Este considera que cada observação é um parâmetro aleatório com uma dada função de distribuição (f_K). A PDF local tradicionalmente utilizada é a função densidade normal padrão. Para uma amostra de valores de função $g_i = g(X_i)$ a estimativa de PDF do kernel para a função g pode ser expressa como (MOTTA et al., 2021):

$$f_G^{kde} = \frac{1}{N\lambda} \sum f_K \left(\frac{g - g_i}{\lambda} \right) \tag{3.7}$$

em que N o tamanho da amostra e f_K é o PDF local do kernel. Aqui, a função de densidade normal é considerada. A definição do desvio padrão (λ) das observações também é um problema a ser resolvido. Está relacionado com a largura de banda e suavidade do PDF estimado (f_G^{kde}). A largura de banda pode ser definida de forma a minimizar o erro. Neste trabalho foi utilizado o pacote KDE disponível na biblioteca ScyPy (VIRTANEN et al., 2020). Na Figura 19 temos a representação do ajuste aplicado aos defeitos de corrosão utilizados neste trabalho para gerar os casos sintéticos que alimentam a rede neural.

Figura 19 – Ajuste via KDE para os defeitos reais.



(a) TS02 (b) TS05

(c) TS06 (d) TS10

Fonte: O autor (2021).

O campo bi-dimensional estocástico foi criado utilizando além do gerador de números aleatórios uma função de covariância exponencial analítica que correlaciona os dados de espessura remanescente da região corroída. Funções exponenciais já foram utilizadas na geração de defeitos reais nos trabalhos (BAO; ZHOU, 2021) e (FERREIRA et al., 2021). Esta função depende, além da variância (σ) das coordenadas espaciais de cada elemento no campo (GHANEM, 1998). A função de correlação é dada pela equação (3.8):

$$cov(x, y) = \sigma \exp^{\frac{-|x_1 - y_1|}{l_1}} \exp^{\frac{-|x_2 - y_2|}{l_2}} \tag{3.8}$$

onde x e y são as coordenadas dos pontos mapeados de espessura remanescente da região do defeito, zona corroída, σ é a variância e l_1 e l_2 são comprimentos de correlação que neste caso foi de 1.5 vezes a distância entre os pontos. Este valor foi escolhido através de testes visando um melhor ajuste dos campos criados em relação aos modelos reais de referência.

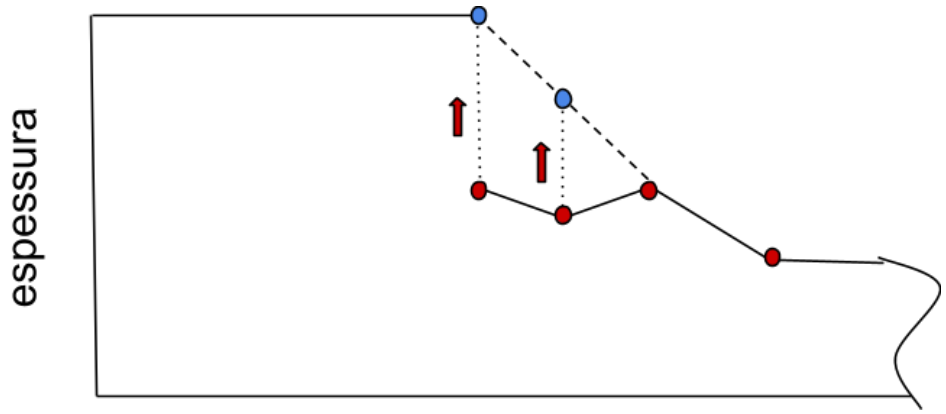
Esse processo de gerar defeitos aleatórios e aplicar uma correlação não é suficiente, pois a região do defeito não se ajusta ao restante do duto devido aos valores das espessuras remanescentes dos pontos que delimitam esta região não serem iguais ao valor da espessura nominal do duto. Isto resulta em uma variação brusca no valor da espessura da região corroída e da região não corroída e essa descontinuidade forma uma região com elevada

concentração de tensões nas zonas que circundam o defeito. Além deste problema a falta de uma região de concordância que suavize a transição entre a zona corroída e a zona não corroída leva a construção de elementos muito distorcidos que podem ocasionar erros numéricos.

Para solucionar o problema foi utilizado um procedimento que impõe uma transição linear que forma uma região de concordância entre a zona do defeito e o duto, onde o perfil de profundidades tem uma variação gradual entre um ponto que está dentro da zona do defeito e um ponto localizado sobre uma das superfícies intactas, não corroída, do duto.

Este processo é ilustrado na Figura 20 onde temos um perfil aleatório inicial, representado pelas circunferências vermelhas e o ajuste realizado com a função linear $f(x) = x$, representado pela linha tracejada com circunferências azuis.

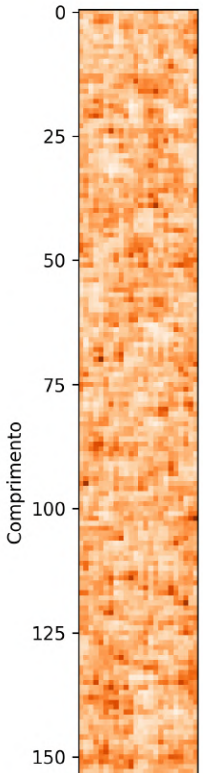
Figura 20 – Ajuste entre a zona corroída e a não corroída.



Fonte: O autor (2021).

Para este trabalho, 1000 modelos sintéticos foram gerados para representar 4 dos 5 defeitos reais. Um dos casos não foi incluído propositalmente para testar se a rede consegue prever um caso totalmente novo que não tenha casos sintéticos correspondentes. Na Figura 21 é apresentado um defeito sintético gerado pelo procedimento desenvolvido neste trabalho.

Figura 21 – Exemplo de defeito sintético utilizado neste trabalho.



Fonte: O autor (2021).

4 AVALIAÇÃO DE INTEGRIDADE

A pressão de falha de tubulações corroídas sujeitas a pressão interna pode ser estimada por simulações não lineares usando o método dos elementos finitos e um critério apropriado de falha previamente validado.

4.1 ANÁLISE NÃO LINEAR

Neste trabalho, são consideradas não linearidades geométricas e físicas. A não linearidade física é caracterizada por uma lei constitutiva elastoplástica com endurecimento isotrópico em grandes deformações. As propriedades do material são representadas aqui por relações tensão-deformação multilineares. A não linearidade geométrica é necessária porque antes da falha da tubulação a região corroída pode ser sujeita a grandes deslocamentos e grandes deformações. As análises não lineares foram realizadas utilizando o software livre e de código aberto code_aster (ASTER, 2019).

4.1.1 Modelagem axissimétrica via MEF

A modelagem axissimétrica permite a análise de um modelo 3D gerado pela rotação de uma figura plana em torno de um eixo de simetria. As relações deformação-deslocamento e tensão-deformação para elementos finitos axissimétricos são mostradas nas Equações (4.1) e (4.2) (LOGAN, 2016).

$$\begin{Bmatrix} \varepsilon_r \\ \varepsilon_z \\ \varepsilon_\theta \\ \gamma_{xy} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial u}{\partial r} \\ \frac{\partial w}{\partial z} \\ \frac{u}{r} \\ \frac{\partial u}{\partial z} + \frac{\partial w}{\partial r} \end{Bmatrix} \tag{4.1}$$

$$\begin{Bmatrix} \sigma_r \\ \sigma_z \\ \sigma_\theta \\ \tau_{xy} \end{Bmatrix} = \Big[D\Big] \begin{Bmatrix} \varepsilon_r \\ \varepsilon_z \\ \varepsilon_\theta \\ \gamma_{xy} \end{Bmatrix} \tag{4.2}$$

onde D é mostrado na Equação (4.3). u, w, z são os deslocamentos e r, z são o eixo.

$$D = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 & 0 \\ \nu & 1-\nu & 0 & 0 \\ 0 & 0 & 1-\nu & 0 \\ 0 & 0 & 0 & \frac{1}{2}(1-2\nu) \end{bmatrix} \tag{4.3}$$

4.1.2 Biblioteca code_aster

Code_aster é uma biblioteca de código aberto desenvolvido principalmente pelo departamento de pesquisa e desenvolvimento da companhia de Eletricidade da França (EDF) conhecido pela sigla EDF R&D. Este departamento tem 9 laboratórios de pesquisa e desenvolvimento, sendo os 3 maiores na França e outros localizados na Inglaterra, Alemanha, Itália, China e Singapura.

A sigla ASTER é um acrônimo para *Analyses des Structures et Thermomécanique pour des Études et des Recherches* que em tradução livre significa Análise Estrutural e Termomecânica para Estudos e Pesquisa. O code_aster é um código para solução de problemas utilizando o método dos elementos finitos. A parte principal do sistema foi desenvolvida utilizando a linguagem de programação Fortran (GATES, 2020) e alguns módulos adicionais foram escritos em python. Esta última pode ser utilizada nos arquivos de configuração da análise, o que dá ao usuário a capacidade de utilizar bibliotecas desta linguagem, aumentando o número de possibilidades e dando a capacidade de configurar a análise de maneira personalizada.

O código do code_aster é distribuído gratuitamente, este pode ser baixado junto com a plataforma chamada Salome-Meca (EDF, 2020). Salome é uma plataforma de código aberto que pode integrar *solvers* e sistemas de CAE&CAD. A Salome possui pacotes para gerar geometria, malha, gerar os arquivos para análise via code_aster e integração com o programa Paraview (INC.; LABORATORY., 2020). Tudo isso por meio de uma interface carregada de opções e com um fluxo intuitivo para a realização de um processo de análise utilizando o método dos elementos finitos.

Os comandos do code_aster são na linguagem nativa dos desenvolvedores, que é o francês. O *debug* e todas as mensagens de alertas ou erros que aparecem também estão na língua francesa. Os manuais possuem versões em francês e inglês, porém o documento na língua inglesa foi construído através de tradução automática e apresentam alguns erros de tradução, inclusive alguns comandos escritos de forma errada.

Como apresentado na seção 3, os nós da região do defeito são calculados através de interpolação cúbica dos dados. Nos modelos axissimétricos, os elementos são definidos como as sub-regiões limitadas pelos 4 nós mais próximos. Para a execução do code_aster elementos unidimensionais são gerados para a aplicação das cargas de pressão interna e, quando necessário, em dutos que possuem tampa para aplicação da pressão longitudinal equivalente a esta condição. Nos modelos tridimensionais, os elementos são hexaédricos e elementos bidimensionais são necessários para aplicação dos carregamentos de pressão interna e longitudinal.

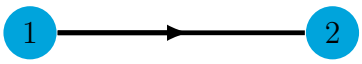
Para completar o processo de geração adequadamente é necessário saber a orientação dos elementos em relação aos nós e suas normais. Na análise de dutos, como o

carregamento principal em muitos casos é pressão, essa normal é extremamente importante pois carregamentos de pressão geralmente tem sua direção positiva na direção contrária a normal do elemento. O código que faz a conversão do modelo do ANSYS para o modelo do code_aster necessitou levar em consideração a equivalência entre as orientações dos elementos dos dois programas.

Outra diferença extremamente importante entre os modelos do ANSYS e do code_aster é na definição das condições de contorno e carregamento. O code_aster aplica carregamentos e condições de contorno apenas em grupos de nós ou elementos. Desta forma para aplicação de pressão, por exemplo, em elementos tridimensionais, um elemento bidimensional necessitou ser gerado na face que recebe o carregamento, ao contrário do ANSYS que recebe esses carregamentos de pressão como uma face do elemento tridimensional. Para os casos bidimensionais, também é necessário criar elementos unidimensionais para a aplicação correta do carregamento de pressão, sempre respeitando as normais para a aplicação na direção desejada.

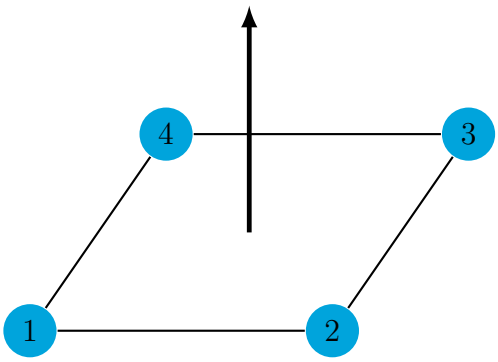
A orientação dos nós e as respectivas normais dos elementos com 2 nós, 4 nós e 8 nós, que foram os utilizados neste trabalho podem ser observados nas Figuras 22, 23 e 24 respectivamente.

Figura 22 – Orientação do elemento linear de 2 nós.



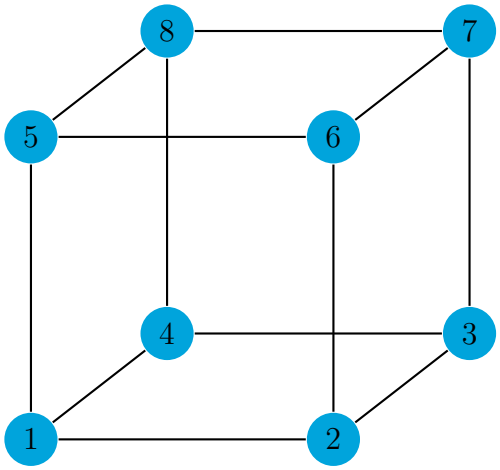
Fonte: O autor (2021).

Figura 23 – Orientação do elemento quadrilateral de 4 nós.



Fonte: O autor (2021).

Figura 24 – Orientação do elemento hexaédrico de 8 nós.



Fonte: O autor (2021).

Para possibilitar a análise de milhares de casos, necessária para treinar uma rede neural em tempo hábil, neste trabalho foi utilizado o *cluster* de computadores do grupo PADMEC. Conforme (PADMEC,), o sistema SGI ICE X instalado no PADMEC possui 17 nós com o total de 544 núcleos computacionais e 34 GPUs. Os nós são interconectados através de uma rede Infiniband de alto desempenho. Cada nó possui a seguinte configuração:

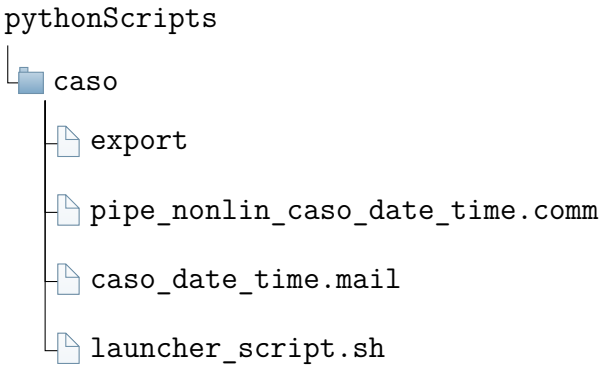
1. **CPU**
    - Modelo: AMD Opteron(TM) Processor 6272
    - Threads por core: 2
    - Cores por socket: 8
    - Quantidade de sockets: 2
    - CPU MHz: 2.100
    - Arquitetura: x86_64
    - Quantidade total de núcleos por nó: 32
2. **GPU**
    - Modelo: NVIDIA Tesla M2075
    - Quantidade: 2
    - Memória: 6 GB
    - Cores: 448
3. **Memória**
    - Quantidade: 132GB

4. **Ambiente Operacional**

   - Sistema Operacional: CentOS release 6.3
   - Versão do Kernel: 2.6.32-279.el6.x86_64
   - Escalonador: SLURM

As rotinas em Python neste trabalho geram automaticamente a malha de elementos finitos e os arquivos necessários para a análise automática no code_aster. Os arquivos gerados pelo código podem ser observados na Figura 25.

Figura 25 – Organização dos arquivos gerados.



Fonte: O autor (2021).

Para possibilitar a análise diretamente no *cluster* alguns arquivos precisam ter configurações específicas que não são facilmente encontrados nos manuais e em fóruns de discussão. O arquivo principal de configuração para possibilitar a análise no code_aster utilizando o *cluster* é o nomeado *export*. A configuração desse arquivo com os comandos necessários para a execução automática são apresentados na Figura 26.

Detalhes do que cada comando apresentado na Figura 26 faz, podem ser vistos no manual em (EŔIC, 2013b). Para execução em um dos nós do cluster, o arquivo nomeado *launcher_script.sh* também é criado automaticamente, as linhas de programação deste podem ser observadas na Figura 27.

Também apresentados na Figura 25, o arquivo com extensão “mail” é o arquivo de malha e o arquivo com extensão “comm” é o arquivo de configuração da análise não linear. Um exemplo de configurações destes arquivos pode ser observado no Anexo A.

4.1.3 Formulação matemática

No code_aster, a relação entre o comportamento mecânico e o endurecimento isotrópico em grandes deformações usa a formulação proposta por Simo e Miehe, para obter detalhes, consulte (SIMO; MIEHE, 1992).

Figura 26 – Formato do arquivo *export*.

```
P actions make_etude
P aster_root /home/adriano/salome_meca/V2016/tools/Code_aster_frontend
   -20160
P memjob 10000.0
P memory_limit 1000.0
P mode interactif
P mpi_nbcpu 1
P nomjob RunCase_1_Stage_1
P origine AsterStudy 2017.0.2
P profastk adriano@zumbi.padmec.org:/storage/adriano/CodeAster/TS61/a.
   astk
P rep_trav /tmp/root-adriano-VirtualBox-interactif.7060
P serveur zumbi.padmec.org
P studyid 10289-0020-adriano-VirtualBox
P uclient adriano
P username adriano
P version stable
A memjeveux 4000.0
A tpmax 35000000
F comm /storage/adriano/CodeAster/Template_Path/
   pipe_nonlin_template_model.comm D  1
F mail /storage/adriano/CodeAster/Template_Path/template_model.mail D
   20
F mess /storage/adriano/CodeAster/Template_Path/template_model.mess R  6
F resu /storage/adriano/CodeAster/Template_Path/template_model.resu R  8
F rmed /storage/adriano/CodeAster/Template_Path/template_model.rmed R
   81
```

Fonte: O autor (2021).

Figura 27 – Formato do arquivo *shell*.

```
#!/bin/bash
#SBATCH --mem=15000
#SBATCH --ntasks=1
#SBATCH --threads-per-core=1
#SBATCH --output=out.out

#Title: launcher
#Author: Ferreira, A.D.M.
#Date: 2019
#Code version: 1.3
#Submits a code_aster job to the cluster.

/home/adriano/salome_meca/appli_V2016/salome shell -p 2810 -u adriano -d
    /home/adriano/salome_meca/appli_V2016/ -- as_run export > runout.out
```

Fonte: O autor (2021).

As características específicas desse modelo são (ERIC, 2013a):

- supõe-se que exista uma configuração intermediária local sem tensão, que permita a

decomposição multiplicativa do gradiente de deformação em uma parte termoelástica e uma parte plástica;

- as deformações elásticas são medidas na configuração atual (deformadas) enquanto as deformações plásticas são medidas na configuração inicial;
- como em pequenas deformações, as restrições dependem apenas das deformações termoelásticas;
- as deformações plásticas são realizadas com volume constante.
- este modelo é incrementalmente objetivo, o que significa que ele é invariante à qualquer mudança na janela espacial de referência.

A formulação variacional é apresentada na Equação (4.4).

$$\int_{\Omega} \boldsymbol{\sigma} \nabla_x \delta \mathbf{v} d\Omega = \int_{\Omega} \rho \mathbf{f} \delta \mathbf{v} d\Omega + \int_{\delta\Omega^f} \mathbf{s^d} \delta \mathbf{v} dS \tag{4.4}$$

onde $\boldsymbol{\sigma}$ é o tensor de tensão Euleriano de Cauchy, $\mathbf{v}$ é o campo cinemático admissível de deslocamentos virtuais, ρ é a densidade na configuração atual, $\mathbf{f}$ é a força volumétrica aplicada ao campo Ω e $\delta\Omega^f$ a parte do contorno do campo Ω onde as forças de superfície $\mathbf{s^d}$ são aplicadas.

4.1.4 Procedimento de solução

O método de Newton-Raphson (KELLEY, 2003) é usado para a solução do problema incremental não linear. Neste método, o conjunto de equações é localmente linearizado e resolvido. Se a solução não atender às equações não lineares originais com uma tolerância prescrita, as últimas serão novamente linearizadas na nova solução. Este procedimento é repetido até convergência (DHONDT, 2011).

A matriz tangente é usada na fase de previsão para calcular uma estimativa do campo de deslocamentos a fim de fazer o método de Newton convergir mais rapidamente. A matriz tangente é mostrada em (4.5).

$$\mathbf{K}_{i-1} = \left. \frac{\partial \mathbf{L}^{int}}{\partial \mathbf{u}} \right|_{(\mathbf{u}_{i-1})} \tag{4.5}$$

Em que $\mathbf{L}_{int}$ representa as forças internas do problema da mecânica quasi-estática não linear e é expresso por:

$$\mathbf{L}_i^{int} = \int_{\Omega} \varepsilon(\mathbf{w_i}) \colon \sigma(\mathbf{u_i}) . d\Omega \tag{4.6}$$

onde $\mathbf{u}$ é o campo de deslocamentos retirado de uma configuração de referência e $\mathbf{w_i}$ indica o campo de deslocamentos virtuais.

Neste trabalho, para todas as análises, o incremento da pressão interna começa com $\Delta P = 1MPa$ e se a convergência não for alcançada, o incremento será dividido automaticamente por dez, $\Delta P = \Delta P/10$. O procedimento é repetido até que a tensão máxima de von Mises atinja a resistência à tração final do material ou a convergência não seja alcançada.

4.1.5 Critério de escoamento

O critério de escoamento de von Mises é usado para detectar o escoamento plástico de tubulações de aço. O critério estabelece que o escoamento plástico começa quando a energia de distorção elástica atinge um valor crítico. Matematicamente, podemos afirmar que o escoamento plástico começa quando o invariante do desvio de tensão (J_2) atinge um valor crítico igual a J_2 no estado de tensão uniaxial (Equação (4.10)). O tensor desviador de tensão ($\boldsymbol{s}$) é expresso pela Equação (4.7).

$$\boldsymbol{s} = \boldsymbol{\sigma} - \boldsymbol{\sigma_{hyd}} \tag{4.7}$$

Onde $\boldsymbol{\sigma}$ são as tensões de Cauchy e $\boldsymbol{\sigma_{hyd}}$ pode ser calculado conforme Equação (4.8), apresentada a seguir:

$$\boldsymbol{\sigma_{hyd}} = -\frac{1}{3}\boldsymbol{\sigma} \colon I \tag{4.8}$$

Em um estado de tensões uniaxial nós temos a Equação (4.9):

$$\boldsymbol{s} = \begin{bmatrix} \frac{2}{3}\sigma & 0 & 0 \\ 0 & -\frac{1}{3}\sigma & 0 \\ 0 & 0 & -\frac{1}{3}\sigma \end{bmatrix} \tag{4.9}$$

onde σ é a tensão uniaxial e o invariante da tensão desviadora J_2 é apresentado na Equação (4.10).

$$J_2 = \frac{1}{2}\boldsymbol{s} \colon \boldsymbol{s} = \frac{1}{3}\sigma^2 \tag{4.10}$$

Consequentemente a tensão equivalente de von Mises é representada pela Equação (4.11).

$$\sigma_v = \sqrt{3J_2} \tag{4.11}$$

4.2 CRITÉRIO DE FALHA

Para a aplicação em dutos corroídos, vários autores como os das referências (CHOI et al., 2003; MOTTA et al., 2017; CHOUCHAOUI; PICK; YOST, 1992; BENJAMIN et al.,

2006; CHOI et al., 2015; ALOWAISI; BECKER; SUN, 2016), definiram alguns critérios de falha para prever a ruptura de dutos. Alguns adotaram que a falha ocorre quando a tensão equivalente de von Mises em todos os elementos ao longo de uma linha radial da parede do duto atinge a tensão última verdadeira, outros afirmaram que a falha ocorre quando a tensão de von Mises na área corroída atinge um valor no intervalo entre a tensão última verdadeira e a tensão última de engenharia ou quando é atingida uma tensão de referência na espessura remanescente.

Nesta tese, a pressão de falha para o duto é definida como a pressão quando a tensão equivalente de von Mises em qualquer ponto da área de corrosão atinge a tensão última verdadeira do material.

4.3 PROPRIEDADES DO MATERIAL

A relação de Ramberg-Osgood (RAMBERG; OSGOOD, 1943), que pode ser expressa por Equação (4.12), é adotada como lei constitutiva de todos os materiais utilizados neste trabalho.

$$\varepsilon = \frac{\sigma}{E} + \alpha \frac{\sigma}{E} \left(\frac{\sigma}{\sigma_y} \right)^n \tag{4.12}$$

Onde n e α são coeficientes de endurecimento que dependem do material que está sendo considerado e sua regra de endurecimento. A variável σ_y é a tensão de escoamento.

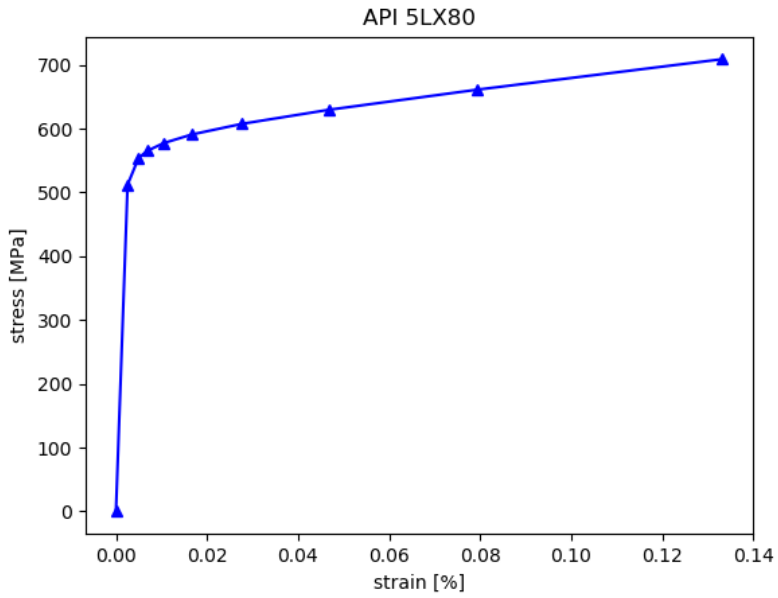
Todos os materiais utilizados neste trabalho tem módulo de Young (E) e coeficiente de Poisson (ν) iguais a 200 GPa e 0.3, respectivamente.

Para a avaliação dos dutos com defeitos axissimétricos foram utilizados todos os materiais de alta resistência da família API 5LX. Os dados de tensão de escoamento (σ_y), tensão última (σ_u), deformação de escoamento (ε_y) e deformação última (ε_u) utilizados foram os padrões da indústria. Para os dutos com defeitos tridimensionais, foram utilizados os dados de tensão e deformação obtidos por ensaio de tração apresentados no trabalho de (SOUZA et al., 2005) e utilizados para análise via MEF no trabalho de (PIMENTEL et al., 2020).

Após conseguir as tensões e deformações de engenharia é necessário obter a tensão e deformação última verdadeiras, conforme apresentados no trabalho de (ANDRADE et al., 2008). Com a tensão última verdadeira, a deformação última verdadeira, a tensão de escoamento e a deformação de escoamento, os coeficientes de endurecimento são calculados e os outros pontos da curva são encontrados.

Figura 28 mostra a curva da tensão verdadeira como função da deformação verdadeira para o aço API 5LX80, construída utilizando a equação de Ramberg-Osgood.

Figura 28 – Tensão verdadeira em função da deformação verdadeira para o aço API 5LX80.



Fonte: O autor (2021).

4.4 CONDIÇÕES DE CONTORNO E CARREGAMENTO

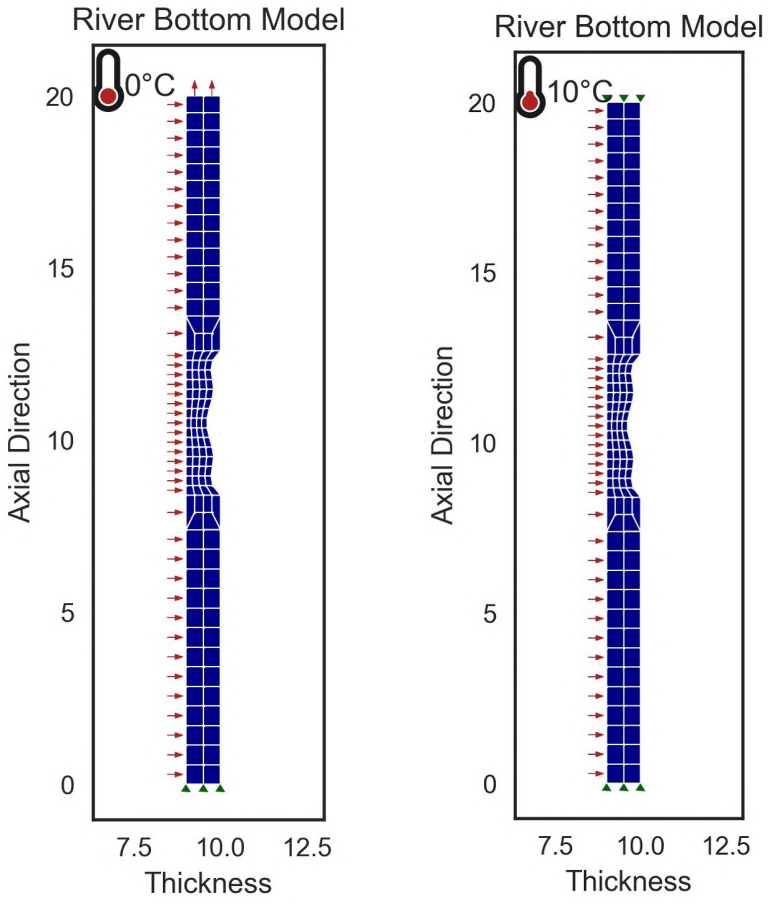
Os dutos com defeitos de corrosão podem estar sujeitos a vários tipos de carregamentos e condições de contorno.

As condições de contorno (CC) e as cargas que podem ser aplicadas, usando as rotinas criadas são: a restrição no deslocamento longitudinal, pressão interna, pressão longitudinal devido ao efeito de tampa e temperatura. Esta última é uma temperatura uniforme atribuída a todos os nós do modelo, que resulta em deformações térmicas aplicadas ao modelo. A malha e as condições de contorno e carregamento geradas são impressas em arquivos de entrada para o software code_aster (ASTER, 2019).

Para exemplificar as possibilidades de CC e carregamento, a Figura 29 mostra dois modelos com condição de contorno e carregamentos. Caso 1 (a), com pressão interna, pressão longitudinal (simulando a presença de tampa) e sem diferença de temperatura uniforme; e Caso 2 (b), com pressão interna, extremidades totalmente restritas e com diferença de temperatura uniforme.

Os modelos analisados e usados para alimentar a Rede Neural Profunda (Seção 7.3.3) dos modelos axissimétricos são dutos enterrados sujeitos a pressão interna. As restrições de deslocamento foram impostas na direção axial, em todos os nós localizados nas extremidades da tubulação para simular a continuidade e a restrição de deformação axial (ancoragem) dos dutos enterrados. Uma representação esquemática das cargas aplicadas e das condições de contorno utilizadas nesta seção é mostrada na Figura 30.

Figura 29 – Malha, condições de contorno e carregamento.



(a) Caso 1 (b) Caso 2

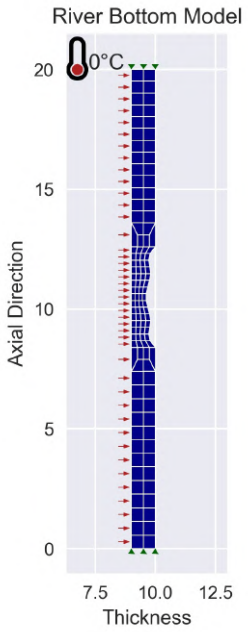
Fonte: O autor (2021).

Os dois modelos analisados na validação dos modelos axissimétricos (Seção 7.1) são dutos submetidos a uma pressão interna com tampas nas extremidades. Essas tampas finais geram uma pressão longitudinal que é considerada na simulação. As restrições de deslocamento são impostas na direção axial em uma das extremidades do modelo. A representação das CC e cargas para essa condição é mostrada na Figura 31.

A implementação realizada para defeitos tridimensionais também permite a aplicação das condições de contorno restrita e com tampa, além dos carregamentos de pressão interna, pressão longitudinal e temperatura.

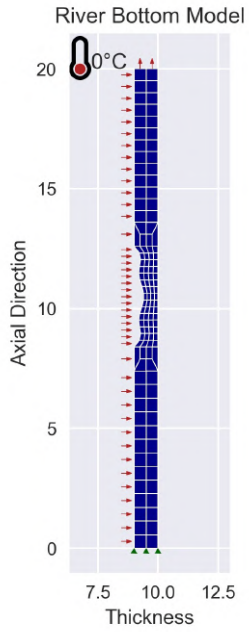
A restrição de deslocamento na direção longitudinal é sempre aplicada em todos os nós de uma das extremidades do duto. Na simulação de dutos que possuem restrição longitudinal de deslocamento, a restrição longitudinal é aplicada em todos os nós da outra extremidade do duto. Para evitar movimento de corpo rígido, é aplicado restrição de deslocamento nas direções transversal e vertical em um nó.

Figura 30 – Condições de contorno e carregamento dos casos utilizados para alimentar a rede neural.



Fonte: O autor (2021).

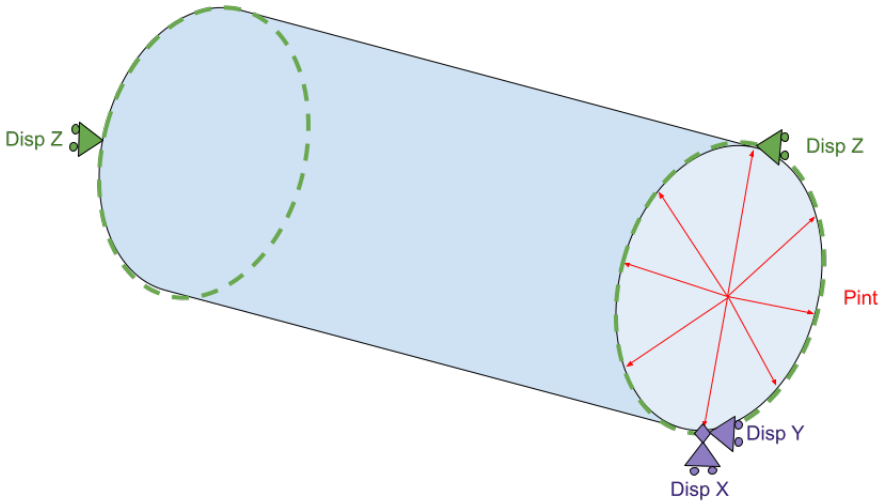
Figura 31 – Condições de contorno e carregamento utilizadas na validação.



Fonte: O autor (2021).

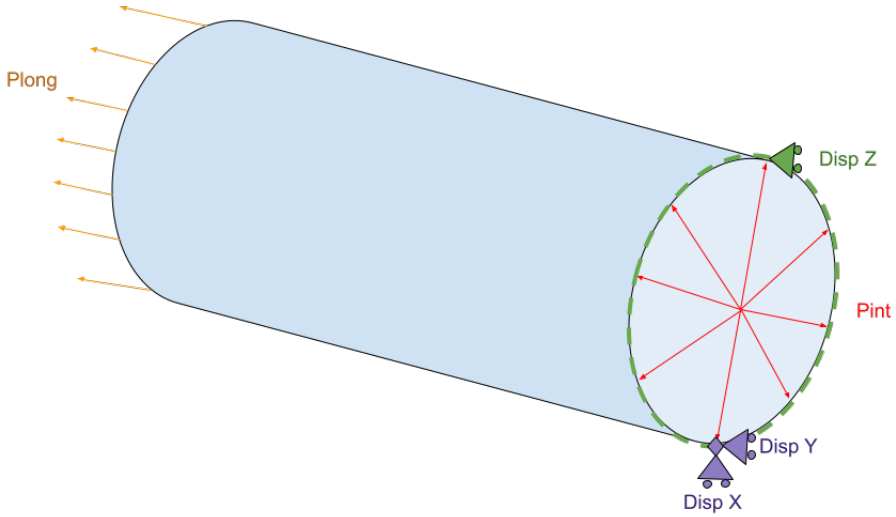
Nas Figuras 32 e 33 são apresentadas visualizações da aplicação de contorno e carregamento dos modelos trimensionais.

Figura 32 – Condições de contorno e carregamento para o modelo tridimensional resrito.



Fonte: O autor (2021).

Figura 33 – Condições de contorno e carregamento para o modelo tridimensional com tampa.



Fonte: O autor (2021).

Onde *Disp Z*, *Disp X* e *Disp Y* são respectivamente as restrições de deslocamento longitudinal, transversal e vertical, *Pin* representa a pressão interna e *Plong* a pressão longitudinal. Nesse trabalho tanto para a validação quanto para a geração de modelos sintéticos que alimentaram a rede neural, foi utilizada a condição de contorno com tampa (Figura 33).

4.5 PROCEDIMENTO DA ANÁLISE NÃO LINEAR

Neste trabalho, um procedimento para automatização da análise não linear foi desenvolvido. O objetivo era que todas as análises fossem realizadas sem a necessidade de intervenção manual e sem problemas numéricos como falta de convergência. No primeiro passo da análise um carregamento unitário é aplicado e a tensão máxima de von Mises é obtida. Com essa tensão máxima estima-se a pressão de escoamento através utilizando uma aproximação linear. No segundo passo 80% desta pressão estimada de escoamento é aplicada de modo que o duto não plastifique neste passo da análise. A partir daí os incrementos são definidos como unitários. Caso a análise pare por falte convergência ou a tensão máxima seja alcançada, a análise volta para o passo anterior e um novo incremento equivalente a $0.1\Delta P$ é utilizado. O processo continua até que pare novamente por falta de convergência ou a tensão última seja alcançada. O algoritmo 1 representa o processo desenvolvido.

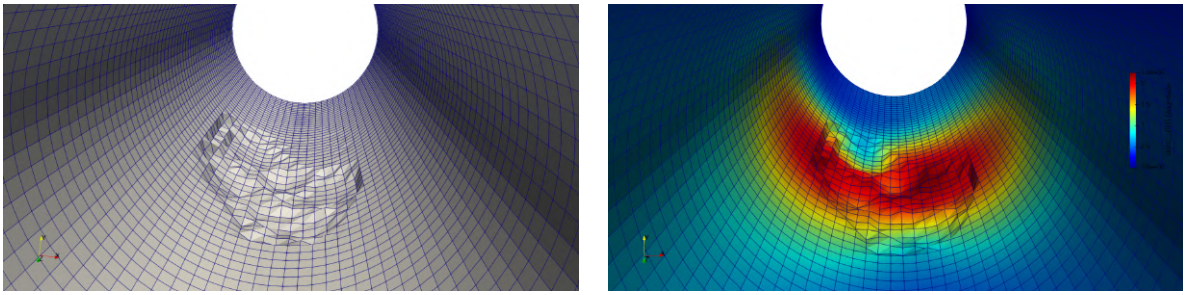
Algoritmo 1: Análise não-linear

Result: σ_v
$P_{int} \leftarrow 1$;
$P_{long} \leftarrow P_{int} \times (D_i^2/(D_i^2 - D_e^2))$;
$\boldsymbol{\sigma_v} \leftarrow run\_codeaster()$;
$P_0 \leftarrow \sigma_e/max(\sigma_v)$;
Function `loop_aster()`:
  while $max(\boldsymbol{\sigma_v}) \leq \sigma_{ult}$ **do**
    try:
      $\boldsymbol{\sigma_v} \leftarrow run\_codeaster()$;
      $i \leftarrow i + 1$;
      $P_{int}^i \leftarrow P_{int}^{i-1} + \Delta_P$;
      $P_{long}^i \leftarrow P_{int}^i \times (D_i^2/(D_i^2 - D_e^2))$;
    except:
      "raise error";
    end
  end
End Function;
$i \leftarrow 1$;
$P_{int}^i \leftarrow 0.8 \times P_0$;
$P_{long}^i \leftarrow P_{int} \times (D_i^2/(D_i^2 - D_e^2))$;
$\Delta_P \leftarrow 1$;
loop_aster();
$\Delta_P \leftarrow 0.1$;
loop_aster();

Em todas as análises foram utilizados elementos lineares. Considerou-se que os materiais são elastoplásticos com endurecimento isotrópico que é característico dos aços. O método de Newton-Raphson foi utilizado na análise não linear. O critério de parada implementado foi um ponto alcançar a tensão última ou não haver convergência utilizando o incremento de pressão de $0.1MPa$. Em todos os casos analisados nesse trabalho a análise foi finalizada pelo critério de tensão última.

Nas Figuras 34a e 34b são apresentados respectivamente a malha de elementos finitos e o mapa de tensões de von Mises após a geração e análise utilizando as ferramentas e o procedimentos desenvolvidos.

Figura 34 – Exemplo de Defeitos Reais



(a) Malha de EF (b) Tensões de von Mises

Fonte: O autor (2021).

5 ANÁLISE MULTIRESOLUÇÃO

A análise multirresolução ou aproximação multiescala, se baseia no uso de uma sequência de aproximações de uma determinada função ou sinal, $f(t)$, em diferentes níveis de resolução (CHUI, 1992).

Uma das metodologias mais conhecidas na área de processamento de sinais é a transformada de Fourier (COOLEY; TUKEY, 1965). Esta é utilizada para analisar os componentes de frequência presentes no sinal. O método de Fourier transforma um sinal contido em um domínio do tempo em um sinal no domínio de frequências. Como na transformação são utilizadas funções senoidais, que são infinitas por definição, o sinal perde sua característica temporal. Para a solução deste problema foi criado um método conhecido como transformada de Fourier de curto tempo, STFT do inglês *short-time Fourier transform* onde o método é aplicado em janelas de tempo, porém utilizando o princípio da incerteza (COHEN, 2001) observa-se que o tamanho desta janela limita a resolução da frequência.

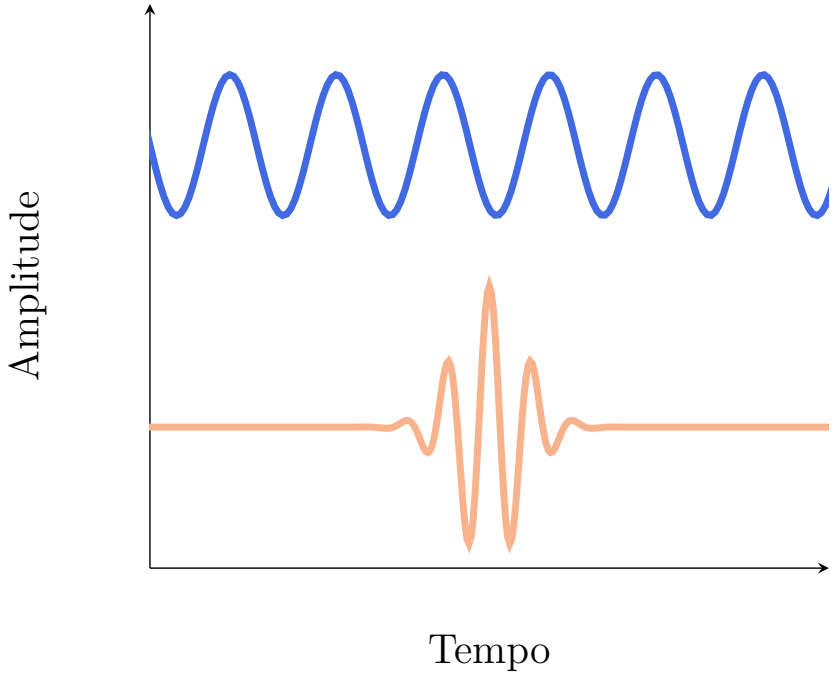
Uma possível solução para o problema descrito é utilizar funções finitas no tempo, *wavelets*, que são transladadas ao longo do sinal para avaliação em várias regiões do mesmo. Estas *wavelets* também são escalonadas para avaliação do sinal em diferentes escalas nos dando uma caracterização maior do sinal estudado. Uma representação de uma função senoidal e de uma *wavelet* pode ser observada na Figura 35.

A Figura 36 apresenta a diferença na discretização de uma série temporal no espaço de frequências entre a transformada de Fourier, a STFT e a transformada *wavelet*.

Na Figura 36 o eixo das abscissas representa o tempo e o eixo das ordenadas representa a frequência.

Neste trabalho, transformada *wavelet* discreta (TWD) é utilizada para parametrizar a espessura remanescente de dutos corroídos. Os coeficientes resultantes da transformação, no espaço transformado, são usados como dados de entrada para um código de aprendizado de máquina (SAMUEL, 1959) e para prever a pressão de ruptura em alguns segundos. Uma das principais vantagens de parametrizar a geometria do defeito é reduzir a quantidade de dados de geometria para cada defeito, que podem ser na ordem de dezenas de milhares de pontos, e manter todos os dados de entrada aproximadamente na mesma magnitude.

Figura 35 – Representação da função senoidal e de uma wavelet



Fonte: O autor (2021).

5.1 TRANSFORMADA WAVELET

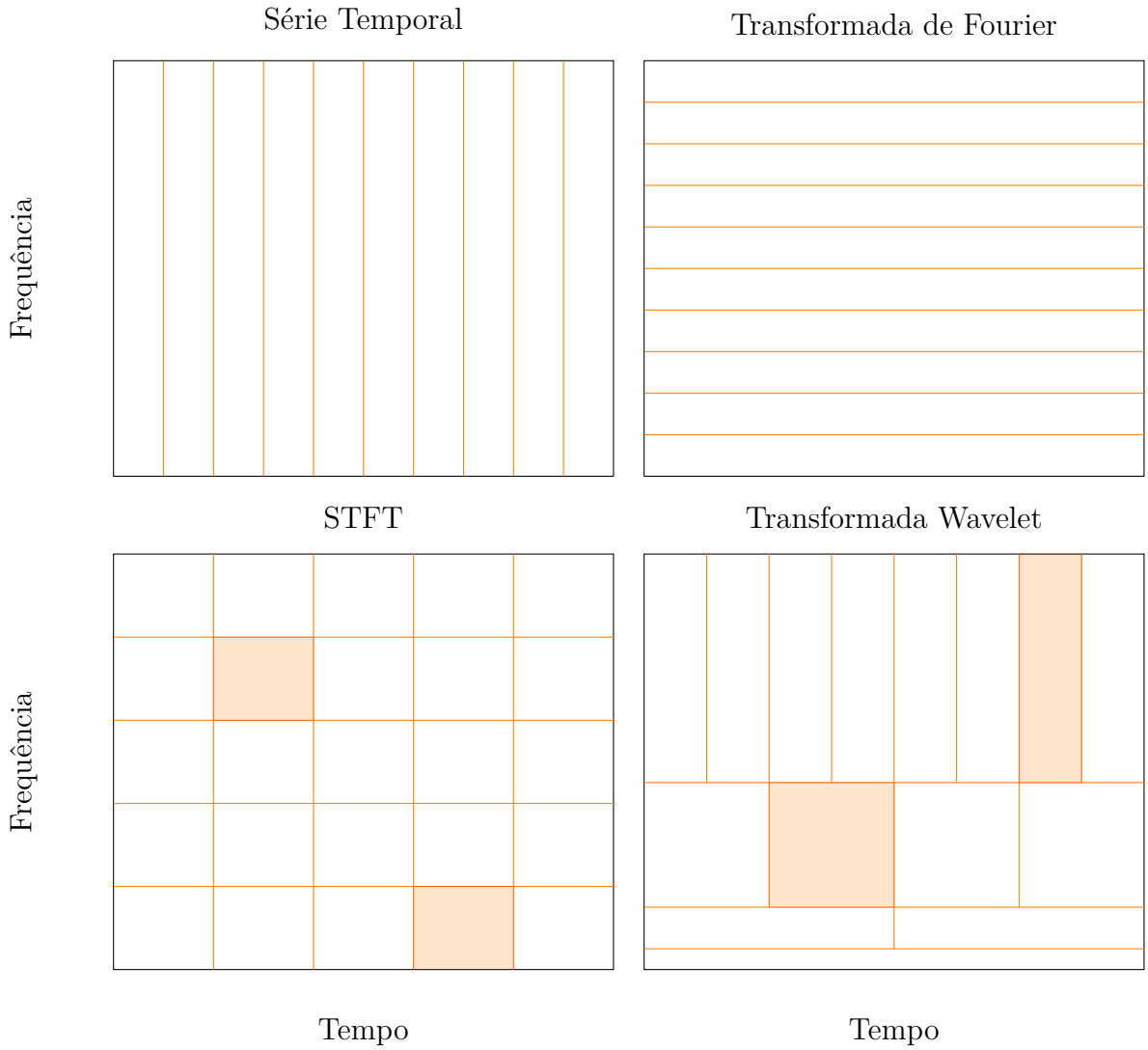
A ideia de transformada *wavelet* apareceu pela primeira vez na literatura na tese do matemático húngaro Alfréd Haar no ano 1909 (HAAR, 1910), porém na época não existia a transformada no conceito atual. Este conceito foi proposto em 1981 pelo geofísico Jean Morlet (MORLET et al., 1982b; MORLET et al., 1982a) que em 1984 junto ao físico chamado Alex Grossman criaram o termo *wavelet*. Morlet utilizou o conceito de *wavelets* para processar digitalmente dados de atividades sísmicas.

A base da teoria das transformadas *wavelet* descende da álgebra linear, onde uma função $f(t)$ qualquer pode ser decomposta como combinações lineares de uma base, se a função estiver contida no subespaço gerado pela base, Equação (5.1) (LIU, 2010; GONZALEZ, 2002).

$$f(t) = \sum_{k} a_k \phi_k(t) \tag{5.1}$$

Onde k é um índice inteiro da soma finita ou infinita, a_k são coeficientes de expansão e $\phi_k(t)$ são funções de expansão. Se a função de expansão é única, ou seja, existe apenas um conjunto de a_k para qualquer função $f(t)$ então $\phi_k(t)$ são chamadas funções de base e o conjunto expandido $\{\phi_k(t)\}$ é chamado base da classe de funções que pode ser expressa. A Equação (5.2) expressa o *closed span* do conjunto expandido.

Figura 36 – Representação gráfica das transformadas.



Fonte: O autor (2021).

$$V = \overline{\text{Span}\{\phi_k(t)\}} \tag{5.2}$$

Desta forma se $f(t) \in V$ então $f(t)$ está contido no subespaço fechado gerado por $\{\phi_k(t)\}$ e a Equação (5.1) é válida.

Escolhendo a base adequadamente, existe outra base $\tilde{\phi}_k(t)$ ortogonal a $\phi_k(t)$ com o produto interno dado pela Equação (5.3).

$$< \phi_i(t), \tilde{\phi}_j(t) >= \int \phi_i(t)\tilde{\phi}_j^*(t)dt = \delta_{ij} \tag{5.3}$$

Onde $\tilde{\phi}_k(t)$ é chamada de função dual de $\phi_k(t)$ e * representa o complexo conjugado. Com esta propriedade ortonormal, nós podemos encontrar os coeficientes através da

Equação (5.4):

$$\begin{aligned} < f(t), \tilde{\phi}_k(t) > &= \int f(t)\tilde{\phi}_k^*(t)dt; \\ &= \int \left(\sum_{k'} a_{k'}\phi_{k'}(t) \right) \tilde{\phi}_k^*(t)dt; \\ &= \sum_{k'} a_{k'} \left(\int \phi_{k'}(t)\tilde{\phi}_k^*(t)dt \right); \\ &= \sum_{k'} a_{k'}\delta_{k'k}; \\ &= a_k. \end{aligned} \tag{5.4}$$

Rescrita como a Equação (5.5) a seguir.

$$a_k = < f(t), \tilde{\phi}_k(t) > = \int f(t)\tilde{\phi}_k^*(t)dt \tag{5.5}$$

Na transformada de Fourier a função $\phi(t)$ é da forma apresentada na Equação (5.6).

$$\phi(t) = \exp^{\left(j\frac{2\pi kt}{T}\right)} \tag{5.6}$$

Obedecendo a propriedade ortogonal desta função exponencial complexa podemos encontrar os coeficientes a_k através da Equação (5.7).

$$a_k = \frac{1}{T}\int_{-T/2}^{T/2} f(t)\exp^{\left(-j\frac{2\pi kt}{T}\right)} dt \tag{5.7}$$

A transformada *wavelet* usa duas funções, $\phi(t)$ que é chamada função de escala (também conhecida como *wavelet* pai) e $\Psi(t)$, a função *wavelet* (também conhecida como *wavelet* mãe), para decompor alguma função em $L^2(\mathbb{R})$ [1]. Na análise multiresolução a função é aproximada através da translação e escala de bases. Com estas características as funções de escala e *wavelet* são construídas utilizando estes dois parâmetros. A definição de ambas está nas Equações (5.8) e (5.9) (GONZALEZ, 2002).

$$\phi_{j,k}(t) = 2^{j/2}\phi(2^j t - k) \tag{5.8}$$

$$\Psi_{j,k}(t) = 2^{j/2}\Psi(2^j t - k) \tag{5.9}$$

Onde j é o parâmetro que indica a dilatação e k define a posição. Podemos ver os dados completos em qualquer resolução j definindo os subespaços nas Equações 5.10 e 5.11.

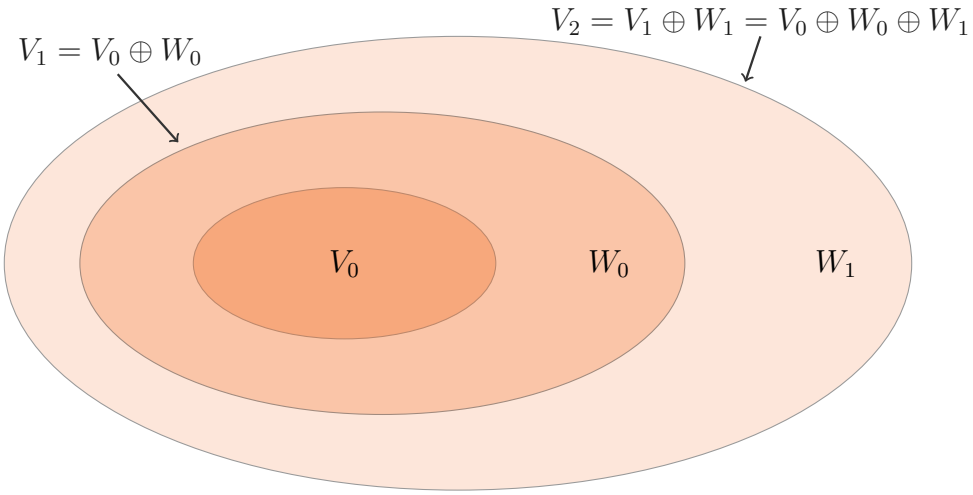
[1] $L^2(\mathbb{R}) = \left\{ f(x) \in \mathbb{R} | \int |f(x)|^2 dx < \infty \right\}$

$$V_j = \text{Span}\{\phi_{j,k}(t)\} \tag{5.10}$$

$$W_j = \text{Span}\{\Psi_{j,k}(t)\} \tag{5.11}$$

Os subespaços gerados pelas funções de escala em baixas escalas estão aninhados dentro dos subespaços de maiores escalas, ou seja $V_j \subset V_{j+1}$. A relação entre os espaços das funções de escala e *wavelet* podem ser observados na Figura 37.

Figura 37 – Relação entre os subespaços V e W.



Fonte: O autor (2021).

Através da figura também podemos concluir que $\phi_{-1}(t) = \phi_0(t) + \phi_0(t-1)$. Assim as funções expandidas do subespaço V_j podem ser expressas como um somatório ponderado das funções expandidas do subespaço V_{j+1} como apresentado na Equação (5.12).

$$\phi_{j,k}(t) = \sum_n \alpha_n \phi_{j+1,n}(t) \tag{5.12}$$

Utilizando a Equação (5.8) para substituir $\phi_{j+1,n}(t)$ e trocando a variável α_n por $h_\phi(n)$, podemos reescrever a Equação (5.12) como apresentado na Equação (5.13).

$$\phi_{j,k}(t) = \sum_n h_\phi(n) 2^{(j+1)/2} \phi(2^{j+1}t - n) \tag{5.13}$$

Para j e k iguais a 0 temos equação fundamental da análise mulitiresolução chamada de equação de refinamento ou equação de dilatação conforme apresentada na Equação (5.14).

$$\phi(t) = \sum_n h_\phi(n) \sqrt{2} \phi(2t - n) \tag{5.14}$$

De forma análoga podemos encontrar uma equação similar para $\Psi(t)$, esta é apresentada na Equação (5.15).

$$\Psi(t) = \sum_n h_\Psi(n)\sqrt{2}\phi(2t - n) \tag{5.15}$$

A transformada *wavelet* se divide em dois grupos: discreto e contínuo, onde, como o termo sugere, um grupo utiliza funções contínuas para encontrar os coeficientes e o outro utiliza funções discretas. Como diferença básica a translação e o parâmetro de escala na transformada contínua podem ser quaisquer valores maiores que zero, ao contrário da função discreta onde estes valores são inteiros. Este trabalho utilizou apenas a transformada *wavelet* discreta, devido a sua capacidade de trabalhar como banco de filtros, reduzindo a quantidade dos dados. A base teórica detalhada e as deduções matemáticas desta transformada são apresentadas no Apêndice A.

Nesta pesquisa foram estudadas a aplicação de transformada *wavelets* tanto para corrosão no formato de perfil *river-bottom* quanto para corrosão real tridimensional.

5.2 APLICAÇÃO

Como explicado no Apêndice A, na transformada *wavelet* discreta a redução da amostragem (*downsampling*) pode ser feita várias vezes em sequência. Para reduzir ao máximo os dados, a decomposição neste trabalho é realizada até o nível máximo. Esse é alcançado quando o *downsampling* torna a dimensão dos dados da amostra menores que a dimensão do filtro.

O algoritmo 2 simplificado do processo de transformação e reconstrução das espessuras corroídas no formato *river-bottom* pode ser observado a seguir.

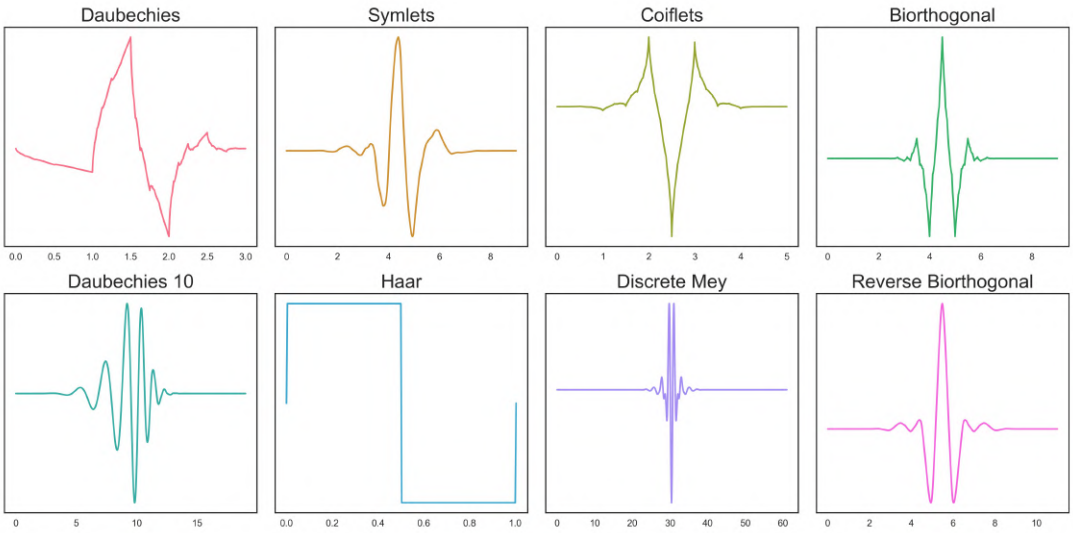
Algoritmo 2: Transformada Wavelet Discreta

Result: $\tilde{t}$
$W_\phi \leftarrow \mathbf{t}$;
while $length(W_\phi) \geq length(h_\phi)$ **do**
  $W_\phi[n] \leftarrow \sum_k h_\phi[k] W_\phi[n-k]$;
  $W_\Psi[n] \leftarrow \sum_k h_\Psi[k] W_\phi[n-k]$;
  $W_\phi \downarrow 2$;
  $W_\Psi \downarrow 2$;
  $W_\Psi.append\,(W_\Psi)$;
end
$W_\Psi \leftarrow 0$;
$size_{w_\phi} \leftarrow length(W_\phi)$;
$\tilde{t} \leftarrow concatenate(W_\phi, W_\Psi)$;
while $size_{w_\phi} < length(\tilde{t})$ **do**
  $W_\phi^{-1} \leftarrow \tilde{t}[: size_{W_\phi}] \uparrow 2$;
  $W_\Psi^{-1} \leftarrow \tilde{t}[size_{W_\phi} : 2 \times size_{W_\phi}] \uparrow 2$;
  $W_\phi^{-1}[n] \leftarrow \sum_k h_\phi^{-1}[k] W_\phi[n-k]$;
  $W_\Psi^{-1}[n] \leftarrow \sum_k h_\Psi^{-1}[k] W_\phi[n-k]$;
  $size_{w_\phi} \leftarrow size_{w_\phi} \times 2$;
  $\tilde{t} \leftarrow W_\phi + W_\Psi + \tilde{t}[size_{w_\phi} :]$;
end

Onde $\mathbf{t}$ representa o vetor de espessuras originais obtidas por inspeção do duto, $\tilde{t}$ é o vetor de espessuras após o processo de transformação e reconstrução, h_ϕ^{-1} e h_Ψ^{-1} representam os filtros inversos de reconstrução. As variáveis W_ϕ^{-1} e W_Ψ^{-1} foram utilizadas para representar os coeficientes na fase de reconstrução. O operador $\downarrow 2$ realiza o *downsampling* de ordem 2 e o operador $\uparrow 2$ faz a operação inversa (*upsampling*) de ordem 2. O operador *length* retorna o número de elementos da variável, o *append* adiciona uma variável a outra e o *concatenate* junta duas variáveis vetoriais em um único vetor. As rotinas necessárias para realização das convoluções foram omitidas para simplificação e estão representadas pelo somatório $\sum_k$. Como explicado anteriormente, a convolução pode ser realizada através do cálculo da transformada de Fourier, diminuindo o custo computacional nesta fase do processo. Neste trabalho foi utilizado uma biblioteca em python chamada PyWavelets.

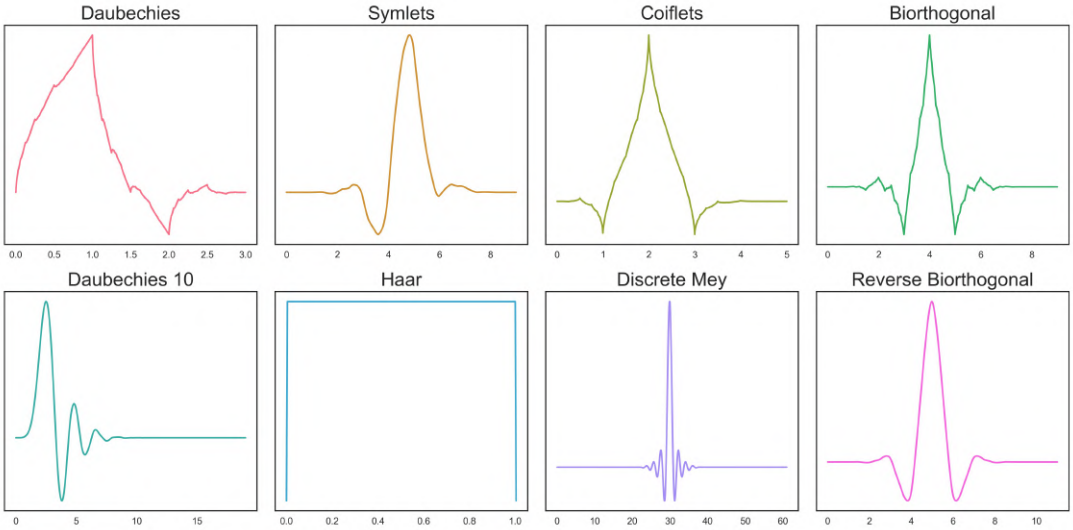
Para encontrar a melhor decomposição a ser aplicada à análise de dutos corroídos, foram testadas várias famílias de *wavelets* (Biorthogonal, Coiflet, Daubechies, Meyer, Haar e Reverse Biorthogonal), cada uma variando o número de momentos de fuga e o tamanho do filtro (Apêndice A). O formato das funções *wavelet* ϕ e as funções de escala Ψ testadas para decompor os defeitos de corrosão são mostradas nas Figura 38 e Figura 39, respectivamente.

Figura 38 – Funções *Wavelet* (ϕ).



Fonte: O autor (2021).

Figura 39 – Funções de Escala (Ψ).



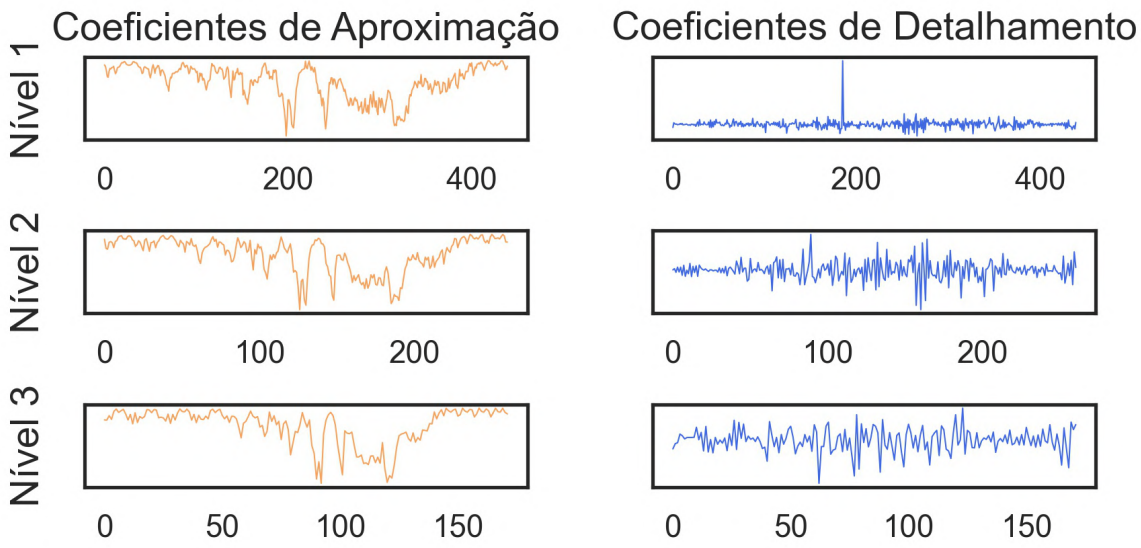
Fonte: O autor (2021).

5.2.1 Defeitos de corrosão *river bottom*

Nesta subseção, doze perfis *river bottom* reais obtidos através de inspeção ILI são usados como base para testar a decomposição e reconstrução usando transformada *wavelet* discreta e para gerar os modelos sintéticos que alimentarão a rede neural. A Figura 12 mostra a representação da espessura remanescente em função do comprimento longitudinal de todos os perfis utilizados.

Para exemplificar o processo, na Figura 40 é mostrada a decomposição aplicada ao caso 1, usando a família de wavelets coiflet, com 14 momentos de fuga e em todos os níveis possíveis. Na coluna da esquerda, os coeficientes de aproximação e na coluna da direita, os coeficientes de detalhamento são mostrados para o caso 1.

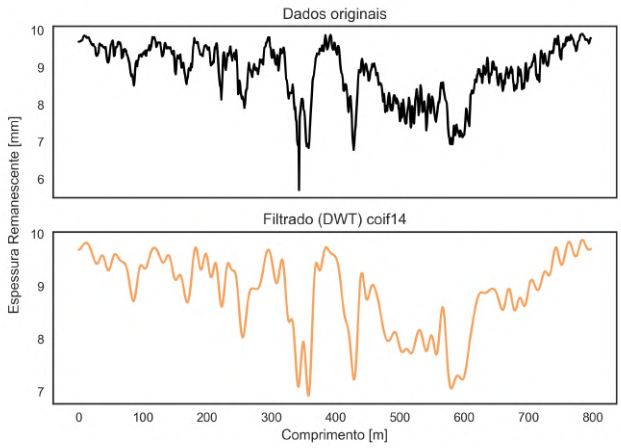
Figura 40 – Decomposição por Coiflet com 14 momentos de fuga, modelo river bottom - caso 1.



Fonte: O autor (2021).

Os coeficientes podem ser usados como um filtro passa-baixa e um filtro passa-alta. Para reduzir os dados, os coeficientes de detalhes (filtro passa-alta) são definidos como zero. A Figura 41 mostra os dados originais e filtrados após a reconstrução do sinal usando a decomposição de coiflet até o Nível 3.

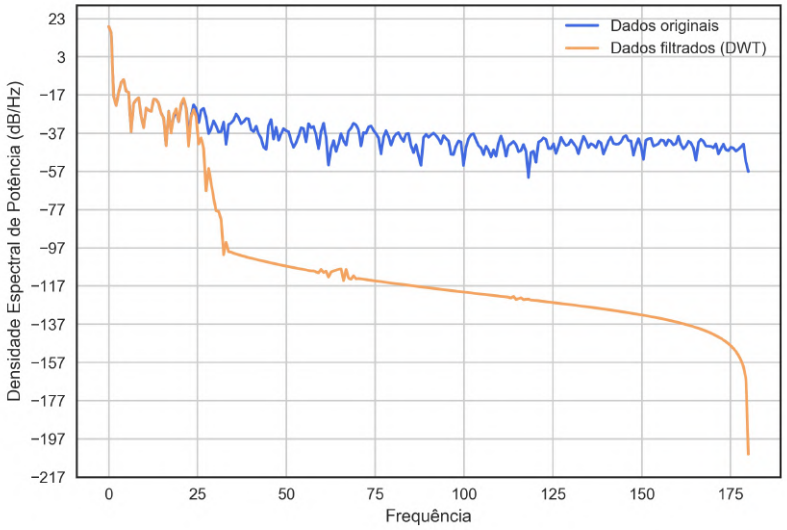
Figura 41 – Caso original e caso filtrado usando coiflet.



Fonte: O autor (2021).

Correspondendo ao caso apresentado na Figura 41, a Figura 42 apresenta a densidade espectral de potência dos dados originais e filtrados, onde pode-se observar claramente a remoção dos componentes de alta frequência.

Figura 42 – Densidade espectral de potência.



Fonte: O autor (2021).

5.2.2 Defeitos de corrosão reais

Nesta subseção, seis perfis reais tridimensionais são usados como base para testar a decomposição e reconstrução usando transformada *wavelet* discreta. Estes mesmos defeitos foram utilizados na Seção 7.1.2 para validação do modelo de EF e são utilizados parar gerar os modelos sintéticos que alimentarão uma rede neural.

O processo de realização da transformada *wavelet* discreta em duas dimensões é bastante similar ao processo unidimensional. A diferença se dá na realização da integral de convolução nas duas direções utilizando cada filtro e a combinação dos dois filtros. Ao realizar esse processo ao invés de apenas uma matriz de coeficientes de detalhamento e uma de coeficientes de aproximação, teremos três de detalhamento ($\Psi(x,y)^H, \Psi(x,y)^V, \Psi(x,y)^D$) e uma de aproximação ($\phi(x,y)$). Para encontrar cada uma matriz de coeficientes é possível utilizar as equações 5.16, 5.17, 5.18 e 5.19.

$$\phi(x,y) = \phi(x)\phi(y) \tag{5.16}$$

$$\Psi(x,y)^H = \Psi(x)\phi(y) \tag{5.17}$$

$$\Psi(x,y)^V = \phi(x)\Psi(y) \tag{5.18}$$

$$\Psi(x,y)^D = \Psi(x)\Psi(y) \tag{5.19}$$

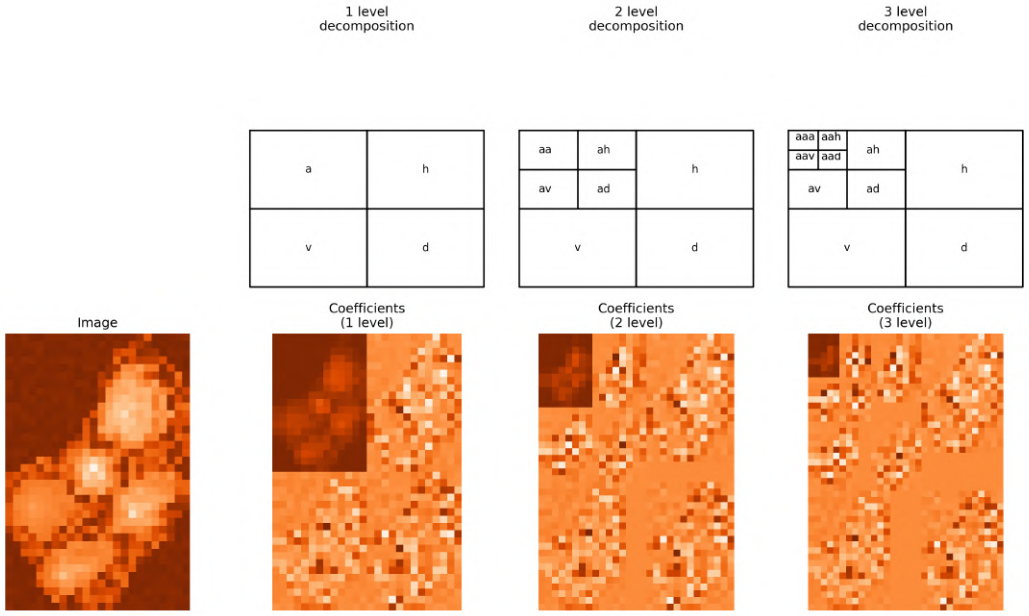
Onde $\phi(x)$ seriam os coeficientes de aproximação obtidos integrando na direção x, $\phi(y)$ seriam os coeficientes de aproximação obtidos integrando na direção y, $\Psi(x)$ seriam os coeficientes de detalhamento obtidos operando na direção x e $\Psi(y)$ seriam os coeficientes de detalhamento obtidos através da operação na direção y.

A Figura 43 apresenta um exemplo de como a decomposição é realizada. Na figura podemos observar os coeficientes de aproximação e detalhamento obtidos para um defeito real. A decomposição é apresentada em três níveis diferentes utilizando a *wavelet* da família Daubechies com um momento de fuga.

Realizando a decomposição até o último nível obtém-se reduções bastante significativas nos dados. Na Tabela 3 é apresentado o resultado da redução nos dados para os casos reais tridimensionais utilizados neste trabalho utilizando a *wavelet* Daubechies com 1 momento de fuga.

É possível concluir, através da Tabela 3, que a redução dos dados foi de aproximadamente 99.5% para todos os casos analisados. O que mostra uma grande capacidade de compressão dos dados ao utilizar transformada *wavelet* como filtro de dados.

Figura 43 – Coeficientes da Decomposição Bidimensional.



Fonte: O autor (2021).

Tabela 3 – Redução dos dados usando Daubechies (1 momento)

	Quantidade de pontos	Nº Coeficientes	Redução dos Dados (%)
TS02	4800	24	99.50
TS04	4800	24	99.50
TS05	4800	24	99.50
TS06	4200	22	99.47
TS10	4700	24	99.48

6 INTELIGÊNCIA ARTIFICIAL

O termo *Machine Learning* foi utilizado pela primeira vez por Arthur Samuel (SAMUEL, 1959) em 1959, como um campo de estudos que dá aos computadores a capacidade de aprender sem ter sido explicitamente programado.

Tom Mitchell em seu livro intitulado *Machine Learning* (MITCHELL, 1997), utiliza a seguinte definição para indicar quando um programa está aprendendo.

Definicão 1. *Um programa de computador aprende com a experiência E com respeito a uma classe de tarefas T e algum desempenho medido P se seu desempenho sobre T como medido por P aumenta com a experiência E.*

6.1 APRENDIZADO DE MÁQUINA

No campo do aprendizado de máquina, existem dois tipos principais de tarefas: a supervisionada e não supervisionada. No aprendizado supervisionado conhece-se a priori as respostas/rótulos de saída para nossas amostras, sendo o objetivo desta aprender uma função que aproxima melhor a relação entre uma dada amostra e os resultados. O processo é chamado supervisionado, pois pode ser pensado como a tarefa de um professor que tem as repostas e corrige a predição a cada iteração de aprendizado do algoritmo. Esse tipo de tarefa é utilizado normalmente no contexto de classificação, quando se quer mapear entradas para os rótulos de saída definidos e no contexto de regressão, quando se quer prever uma saída através dos dados de entrada fornecidos.

No aprendizado não supervisionado não há saídas rotuladas a priori, portanto, seu objetivo é inferir a estrutura inerente em um conjunto de dados. Este tipo de tarefa é utilizado mais usualmente em problemas de *clustering*, quando se quer encontrar estruturas ou padrões em dados não categorizados e para criar regras de associação, onde o algoritmo descobre relações de excitação entre variáveis em um banco de dados, como por exemplo, para descobrir que clientes que compram um produto X são mais propícios a comprar um produto Y. Como alguns outros exemplos de possíveis aplicações são a análise de redes sociais, análise de dados astronômicos e organização de cluster de computadores.

6.2 REDES NEURAIS

Redes neurais artificiais são inspiradas no processo biológico de aprendizado humano. O estudo se deu através da observação de que o cérebro humano utiliza um sistema complexo com uma densa rede interconectada por neurônios, Figura 44, onde o processamento é realizado através de sinapses, que são responsáveis por transmitir o impulso nervoso de

um neurônio para o outro ou para uma célula muscular ou glandular. No ano de 2009, a cientista brasileira Suzana Herculano e outros pesquisadores em (AZEVEDO et al., 2009), estimaram que o cérebro humano tivesse cerca de 86 bilhões de neurônios e 1 quadrilhão de sinapses.

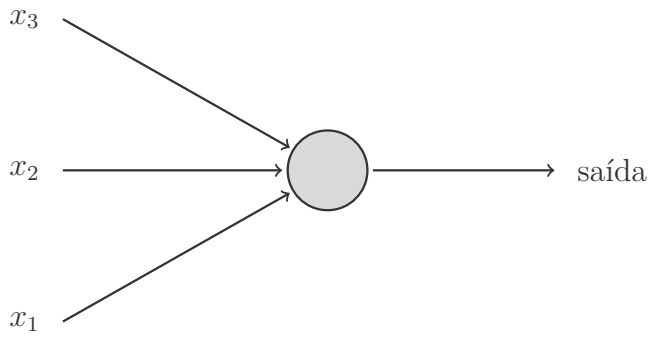
Figura 44 – Representação de rede neural humana.

Fonte: https://www.verywellmind.com/how-many-neurons-are-in-the-brain-2794889.

As redes neurais artificiais foram pensadas em uma forma análoga porém bastante simplificada ao encontrado no sistema humano. Elas utilizam uma rede densa de unidades simples, onde cada unidade recebe uma quantidade de valores reais na entrada e produz uma única saída. Estas entradas podem vir de várias unidades e a saída pode ser transmitida a muitas outras unidades. A ideia da rede artificial é capturar o processamento altamente distribuído contido no sistema humano e utilizar algoritmos otimizados para a rápida computação de problemas complexos. Seu funcionamento se dá através da utilização de exemplos para inferir automaticamente regras de reconhecimento, aumentando o número de exemplos de treinamento a rede pode aprender mais.

O primeiro neurônio artificial, conhecido como *Perceptron*, foi desenvolvido em 1958, pelo cientista Frank Rosenblatt (ROSENBLATT, 1958), inspirado em trabalhos anteriores de Warren McCulloch e Walter Pitts (MCCULLOCH; PITTS, 1943), publicados em 1943. Atualmente existem vários outros modelos de neurônios artificiais que serão apresentados posteriormente neste trabalho. Porém para o maior entendimento do funcionamento de redes neurais, primeiro será apresentado como funcionam os *perceptrons*.

Figura 45 – Representação de um perceptron.



Fonte: O autor (2021).

Um perceptron recebe várias entradas binárias, como por exemplo, x_1, x_2 e x_3 e produz uma única saída binária. Podemos visualizar a representação de um perceptron na Figura 45. No exemplo da figura, o perceptron possui três entradas, x_1, x_2 e x_3.

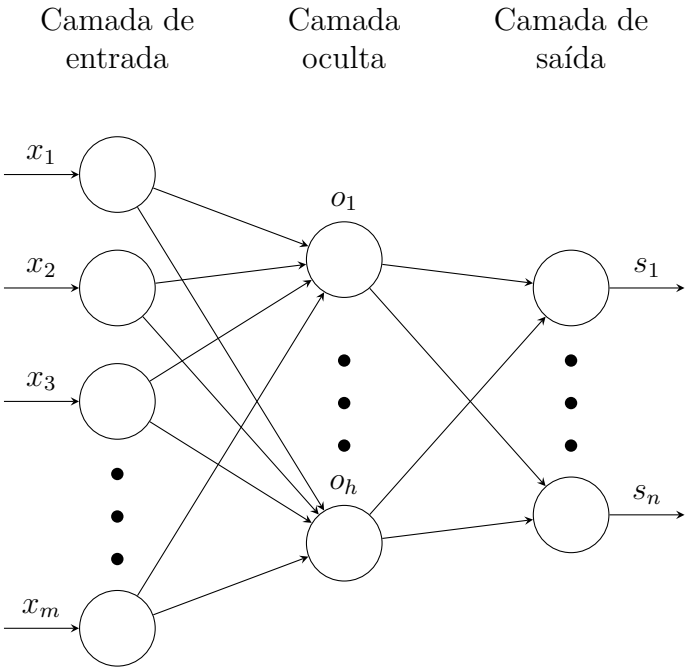
O cálculo da saída de um perceptron proposto por Rosenblatt (ROSENBLATT, 1958) consiste em adicionar um peso, $\{w_1, w_2, ... w_n \in \mathbb{R}\}$, para ponderar as variáveis de entrada. Este pesos servem para expressar o quão importante é uma variável em relação a saída. A saída do neurônio, 0 ou 1, é determinada se a soma ponderada por esses pesos $\sum_j w_j x_j$ é menor ou maior que algum valor limite ou *bias* em inglês, do neurônio, onde $\{b \in \mathbb{R}\}$. Este valor de *bias* pode ser entendido como o valor mínimo necessário para ativar o neurônio. Colocando em termos algébricos temos a Equação (6.1).

$$saída = \begin{cases} 0 & \text{se } \sum_j w_j x_j \leq b \\ 1 & \text{se } \sum_j w_j x_j > b \end{cases} \tag{6.1}$$

Para realização de inferências de problemas complexos, vários perceptrons podem ser agrupados em série e paralelo, criando o que chamamos de rede neural. Desta forma se variamos os valores dos pesos w_j e dos valores b conseguiremos ter diferentes modelos de decisão. Uma representação generalizada de uma rede neural pode ser visto na Figura 46.

A primeira coluna de neurônios é chamada de Camada de entrada (*Input Layer*), a segunda coluna Camada oculta (*Hidden Layer*) e a última Camada de saída (*Output Layer*). A quantidade de camadas ocultas pode ser aumentada para que o modelo de decisão possua mais parâmetros e consiga obter resultados mais precisos para problemas complexos. Redes neurais com diversas camadas ocultas são chamadas de Redes Neurais Profundas, tradução de *Deep Neural Network*. O termo "camada oculta" implica que os neurônios destas camadas não recebem valores de entrada do problema e não são responsáveis pela saída da rede.

Figura 46 – Representação de uma rede neural.



Fonte: O autor (2021).

Neste trabalho, redes neurais profundas foram utilizadas para a predição da pressão de falha de dutos corroídos. A demonstração detalhada do funcionamento de uma rede neural para predição, incluindo a base teórica e equações matemáticas são apresentadas no Apêndice B. Neste apêndice também é apresentado em detalhes o algoritmo de retropropagação, este é considerado um dos mais importantes na área de redes neurais e o algoritmo que permitiu a aplicação massiva de redes neurais em diversas áreas.

6.2.1 Algoritmo redes neurais profundas

Sistemas de redes neurais profundas, *Deep Neural Network* (DNN) do inglês, são sistemas poderosos para prever implicitamente o comportamento de problemas altamente não lineares e aprender informações baseadas em amostras. As redes neurais profundas (RNP) podem prever soluções com alta precisão em diversos campos de pesquisa. Devido à velocidade de obtenção de resultados, o aprendizado de máquina pode ser usado para aplicações de sistemas do tipo *digital twin* na avaliação de segurança operacional de dutos.

Como apresentado no Apêndice B, o funcionamento de redes neurais se baseia em encontrar os pesos w e desvios b de todos os neurônios artificiais da rede minimizando a função objetivo J, responsável por estimar o erro entre a solução passada e a prevista pela rede. Para minimização da função de erro é normalmente utilizado o método *Mini-Batch Gradient Descent*, no qual a expressão de atualização foi apresentada na Equação B.12. O cálculo do gradiente necessário para o método é realizado pelo algoritmo *back-propagation*

apresentado na seção B.1 do respectivo apêndice. O algoritmo 3 completo de aprendizado de uma rede neural profunda utilizando mini-lotes de tamanho m está apresentado a seguir.

Algoritmo 3: Redes Neurais Profundas

Result: w^l, b^l

for $x \leftarrow 1$ **to** m **do**

  $a^{x,1} \leftarrow f(a^{x,1})$;

  for $l \leftarrow 2$ **to** L **do**

    $z^{x,l} \leftarrow w^l a^{x,l-1} + b^l$;

    $a^{x,l} \leftarrow \sigma(z^{x,l})$;

  end

  $\delta^{x,L} \leftarrow \nabla_a J \odot \sigma'(z^{x,L})$;

  for $l \leftarrow L-1$ **to** 2 **do**

    $\delta^{x,l} \leftarrow ((w^{l+1})^T \delta^{x,l+1}) \odot \sigma'(z^{x,l})$;

    $w^l \leftarrow w^l - \frac{\eta}{m}(a^{x,l-1})^T \delta^{x,l}$;

    $b^l \leftarrow b^l - \frac{\eta}{m}\delta^{x,l}$;

  end

end

O algoritmo 3 é repetido tanto para cada mini-lote quanto para o número de épocas definido. Neste trabalho uma biblioteca gratuita e de código aberto, TensorFlow (TEAM, 2019), é utilizada. Aqui, um modelo de regressão de redes neurais profundas é utilizado para predição da pressão de falha de dutos corroídos.

6.3 REDE NEURAL PARA OS CASOS AXISSIMÉTRICOS

Para o desenvolvimento de uma rede neural capaz de avaliar defeitos com corrosão axissimétrica tipo *river bottom*, várias definições precisam ser feitas, dentre eles estão os parâmetros de entrada, a função de ativação, o algoritmo de otimização e a arquitetura da rede. No primeiro momento, a rede foi criada para avaliar dutos axissimétricos construídos de material da classe $API5LX80$. Os parâmetros de entrada utilizados neste caso são os coeficientes de aproximação (*ncoeffs*), obtidos após a decomposição por *wavelet*, e o parâmetro adimensional (d_p) mostrados na Equação (6.2).

$$d_p = \frac{L}{\sqrt{D_e t}} \tag{6.2}$$

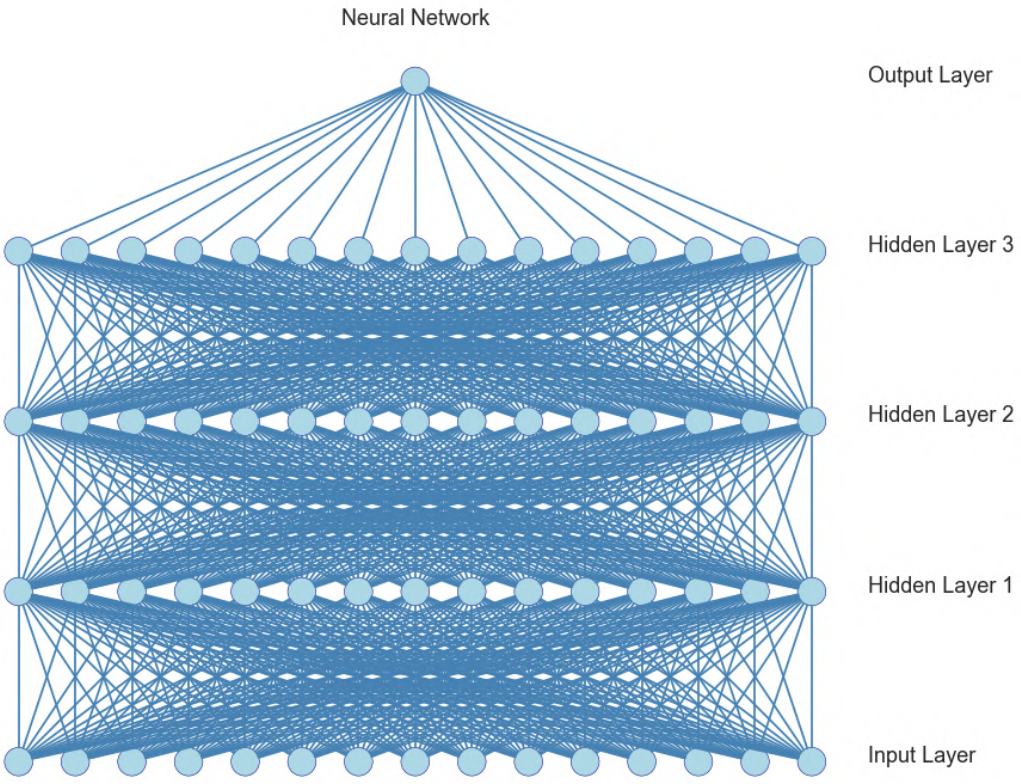
Onde, D_e é o diâmetro externo da tubulação, t é a espessura nominal e L é o comprimento do defeito de corrosão. Os parâmetros de saída utilizados nas fases de treinamento e validação da rede neural são as pressões de falha obtidas das análises não

lineares via MEF. Na fase de pré-processamento, todos os dados são normalizados no intervalo entre 0 e 1 através da Eq (6.3).

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{6.3}$$

Na camada de entrada da rede neural, o número de neurônios é definido como $max(ncoeffs) + 1$. Após a realização de vários testes buscando um melhor resultado, o número de camadas ocultas foi definido como três e o número de neurônios nas camadas ocultas foi definido igual ao número de neurônios de entrada. A camada de saída possui um neurônio. Portanto, a rede neural é composta por uma camada de entrada, três camadas ocultas e uma camada de saída. A arquitetura de uma RNP com 15 neurônios usada neste estudo é ilustrada na Figura 47.

Figura 47 – Arquitetura da Rede Neural adotada.



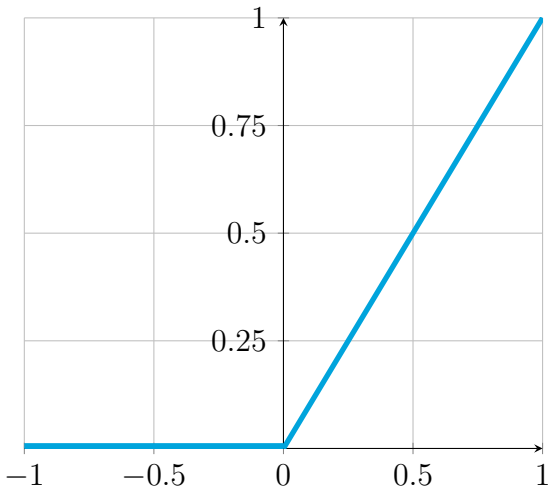
Fonte: O autor (2021).

A função de ativação é a *Rectified Linear Unit* (ReLU) dada na Equação (6.4).

$$\text{ReLU}(x) = max(0, x) \tag{6.4}$$

Onde x é a soma da entrada ponderada e do viés para cada neurônio. Um gráfico representativo desta função pode ser observado na Figura 48.

Figura 48 – Representação gráfica da função ReLU.



Fonte: O autor (2021).

A principal vantagem de usar a função de ativação ReLU é que ela não ativa todos os neurônios ao mesmo tempo. Se a entrada (x) for negativa, ela será convertida em zero e o neurônio não será ativado. Isso significa que, a qualquer momento, apenas alguns neurônios são ativados, tornando a rede esparsa e mais eficiente para o cálculo (GUPTA, 2017).

A principal desvantagem de utilizar esta função é que entradas negativas desativam os neurônios, pois sua saída é definida como 0. Este fato pode fazer o processo de ajuste e aprendizagem da rede neural diminuir sua eficácia.

O algoritmo de otimização utilizado é o Método do Subgradiente Adaptativo (Adagrad) (DUCHI; HAZAN; SINGER, 2011). Um dos principais benefícios do Adagrad é que ele elimina a necessidade de ajustar manualmente a taxa de aprendizado porque atualiza a taxa de aprendizado de cada parâmetro em vez de usar uma única taxa de aprendizado universal.

O Adagrad adapta a taxa de aprendizado aos parâmetros, baixas taxas de aprendizado são realizadas para parâmetros associados a características que ocorrem com frequência e altas taxas de aprendizado para parâmetros associados a características pouco frequentes (RUDER, 2016).

Utilizando $J(\theta)$ para representar uma função objetivo parametrizada pelos parâmetros do modelo $\theta \in \mathbb{R}^d$, $g_{\mathfrak{t}}$ para denotar o gradiente no passo de tempo $\mathfrak{t}$, a derivada parcial da função objetivo em relação ao parâmetro θ_i no passo de tempo $\mathfrak{t}$, $g_{\mathfrak{t},i}$, pode ser expressa pela Equação (6.5).

$$g_{\mathfrak{t},i} = \nabla_\theta J(\theta_{\mathfrak{t},i}) \tag{6.5}$$

Baseado nos gradientes passados que foram calculados para θ_i, o algoritmo modifica a taxa geral de aprendizado (η) em cada passo de tempo $\mathfrak{t}$ para cada parâmetro θ_i. A regra de atualização é apresentada na Equação (6.6).

$$\theta_{\mathfrak{t}+1,i} = \theta_{\mathfrak{t},i} - \frac{\eta}{\sqrt{G_{\mathfrak{t},ii} + \epsilon}} \cdot g_{\mathfrak{t},i} \tag{6.6}$$

Onde $G_{\mathfrak{t}} \in \mathbb{R}^{dxd}$ é a matriz diagonal onde cada elemento da diagonal i, i é a soma dos quadrados dos gradientes em relação à θ_i até o passo de tempo $\mathfrak{t}$, enquanto ϵ é um termo de suavização que evita divisão por zero, normalmente ele é na ordem de $1e-8$ (RUDER, 2016).

No segundo momento, a rede neural foi utilizada para prever defeitos axissimétricos considerando todos os materiais de alta resistência de grau $API5LX$. No total oito materiais diferentes foram considerados na realização da análise não linear: X42, X46, X52, X56, X65, X70 e X80. As propriedades mecânicas dos materiais são apresentadas na Tabela 4.

Tabela 4 – Propriedades dos materiais.

	X42	X46	X52	X56	X60	X65	X70	X80
σ_e	289.6	317.2	358.5	386.1	413.7	488.2	482.6	551.6
σ_u	413.7	434.4	455.1	489.5	517.1	530.9	565.4	620.5
ϵ_u	0.23	0.22	0.21	0.19	0.19	0.18	0.17	0.16

Fonte: O autor (2022).

O módulo de elasticidade para todos eles é igual a $200GPa$. Novamente foram utilizados apenas os dados sintéticos para treinar e validar a rede. A proporção entre estes é fixada em 70% e 30%, respectivamente. O tamanho dos lotes é definido como 100 e o número de épocas para iterar sobre os dados é definido como 3000. A função de ativação utilizada foi a *sigmoid*, pois resultados melhores foram encontrados com esta. O otimizador utilizado foi o AdaGrad.

7 RESULTADOS

Neste capítulo serão apresentados os resultados da pesquisa. Primeiro serão apresentados os resultados da validação do gerador de modelos e do procedimento de análise. Em sequência serão apresentados os resultados da validação da utilização da transformada *wavelets* como filtro para os dados de corrosão aplicados a defeitos do tipo *river bottom* e tridimensionais.

7.1 VALIDAÇÃO DO MODELO E DA ANÁLISE NÃO LINEAR

Nesta seção são apresentados os resultados da validação das ferramentas criadas para gerar os modelos de EF e o procedimento automático de análise não linear utilizando o code_aster. Os resultados encontrados via MEF são confrontados com resultados obtidos através de normas e de experimentos.

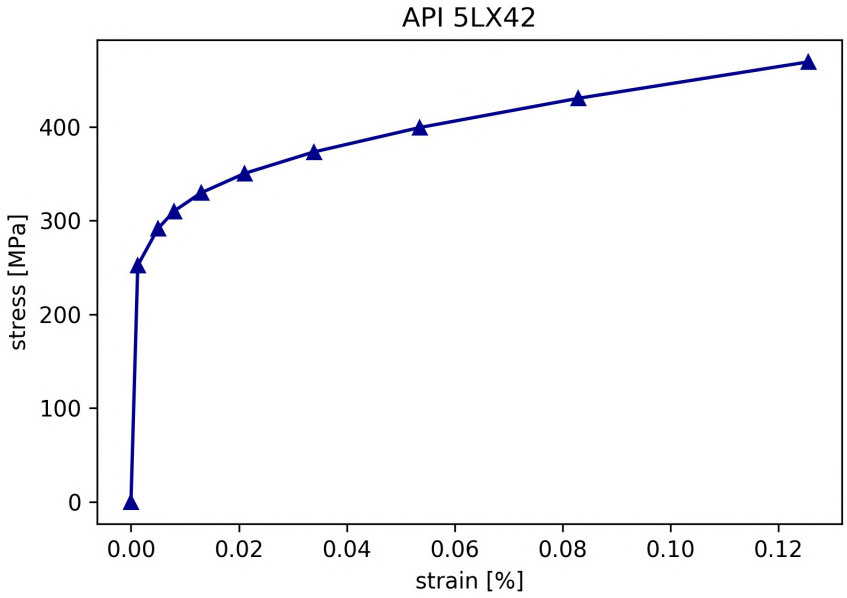
7.1.1 Validação da análise por EF axissimétrica

Para validar o procedimento automático para construção de modelos de FE, procedimento de análise não linear e critérios de falha, a pressão de falha obtida com simulações de elementos finitos é comparada com dois resultados de testes experimentais obtidos por testes de ruptura no trabalho de (SOUZA et al., 2007) e com resultados usando métodos semi-empíricos de avaliação. De acordo com o Manual de Avaliação de Defeitos de Tubulação (COSHAM; HOPKINS, 2002), neste trabalho são utilizados métodos de avaliação de Nível 2 e Nível 3, que são procedimentos baseados em padrões técnicos para defeitos isolados (Nível 2) e procedimentos que consideram o perfil do defeito ou a área efetiva de colônias de defeitos (Nível 3).

As duas amostras tubulares, denominadas $TS6.1$ e $TS6.2$, contêm uma corrosão com geometria irregular e as espessuras remanescentes foram obtidas utilizando equipamento ultrassônico manual. A distância longitudinal entre a espessura medida foi de 10 mm e a distância circunferencial foi de 40 mm. Os dutos são feitos de aço $API5LX42$ e suas propriedades foram medidas através de teste de tração uniaxial. A curva de tensão verdadeira x tensão real para esse material é mostrada na Figura 49.

Nos testes de laboratório apresentado nas referências (SOUZA, 2003; SOUZA et al., 2004; SOUZA et al., 2005), as amostras tubulares foram carregadas apenas com pressão interna e nas extremidades do duto corroído duas seções de 1 m de comprimento de duto livre de corrosão e foram fechadas por soldagem de uma chapa lisa e espessa conhecida como *endcaps*.

Figura 49 – Tensão verdadeira em função da deformação verdadeira dos espécimes testados.



Fonte: O autor (2022).

Nos modelos de elementos finitos, a carga de pressão longitudinal é considerada devido à presença de tampas nas configurações experimentais. A pressão longitudinal é calculada usando a equação (7.1).

$$P_{long} = P_i \frac{D_i^2}{D_e^2 - D_i^2} \tag{7.1}$$

Onde P_i é a pressão na iteração real da análise e D_e, D_i são o diâmetro externo e o diâmetro interno, respectivamente.

Para métodos de avaliação de nível 2 (B31G, DNV RP-F101 defeito isolado e considerando a norma B31G RSTRENG 0,85dL), os dados de entrada para a geometria do defeito são a profundidade máxima d_{max} e o comprimento geral do defeito L. Para os métodos de nível 3 (RSTRENG Effective Area e DNV RP-F101, defeitos de geometria complexa), é necessário criar os defeitos idealizados que são caracterizados por combinações de defeitos longitudinais segmentados e contínuos, com seus respectivos comprimentos e perfis de profundidade.

O método da área efetiva divide a corrosão complexa em várias cavidades com diferentes comprimentos efetivos. Para cada um, é encontrada a área efetiva da região do defeito, seguida pelo cálculo das respectivas pressões de falha. A pressão geral de falha é o mínimo entre todas as pressões calculadas.

Para os defeitos de forma complexa na DNV, o defeito de corrosão é dividido em

vários incrementos com base na profundidade. A cada incremento de profundidade, o defeito de corrosão é modelado por um conjunto contendo um número de defeitos idealizados, em que a pressão de falha é o mínimo entre todas as pressões de cada defeito isolado e das possíveis combinações de interação entre eles. Todos os métodos descritos acima foram executados usando scripts implementados em Python. A Tabela 5 mostra as equações de pressão de falha de acordo com diferentes normas. Algumas normas usam a tensão de escoamento (σ_y), enquanto outros usam a tensão última (σ_u).

Tabela 5 – Pressão de falha para diferentes normas.

Method	P_f	Length limit	M
B31G	$P_f = 1.1\sigma_y \left(\frac{2t}{D_e}\right) \left[\frac{\left(1-\frac{2d}{3t}\right)}{\left(1-\frac{2d}{3tM}\right)}\right]$	$L_L \leq \sqrt{20D_et}$	$M = \sqrt{1 + 0.8\left(\frac{L_L^2}{D_et}\right)}$
	$P_f = 1.1\sigma_y \left(\frac{2t}{D_e}\right)\left(1-\frac{d}{t}\right)$	$L_L > \sqrt{20D_et}$	$M = \infty$
DNV (Single)	$P = \frac{2t\sigma_u}{(D_e-t)}\frac{\left(1-\frac{d}{t}\right)}{\left(1-\frac{d}{tM}\right)}$	–	$M = \sqrt{1 + 0.31\left(\frac{L_L}{\sqrt{D_et}}\right)^2}$
RSTRENG 0.85dL	$P_f = (\sigma_y + 68.95MPa)\left(\frac{2t}{D_e}\right)\left[\frac{\left(1-0.85\frac{d}{t}\right)}{\left(1-0.85\frac{d}{tM}\right)}\right]$	$L \leq \sqrt{50D_et}$	$M = \sqrt{1 + 0.6275\left(\frac{L_L^2}{D_et}\right) - 0.003375\left(\frac{L_L^4}{D_e^2t^2}\right)}$
		$L_L > \sqrt{50D_et}$	$M = 0.032\left(\frac{L_L^2}{D_et}\right) + 3.3$
Effective Area	$P_f = (\sigma_y + 68.95MPa)\left(\frac{2t}{D_e}\right)\left[\frac{\left(1-\frac{A_{ef}}{A_0}\right)}{\left(1-\frac{A_{ef}}{A_0M}\right)}\right]$	$L_{ef} \leq \sqrt{50D_et}$	$M = \sqrt{1 + 0.6275\left(\frac{L_{ef}^2}{D_et}\right) - 0.003375\left(\frac{L_{ef}^4}{D_e^2t^2}\right)}$
		$L_{ef} > \sqrt{50D_et}$	$M = 0.032\left(\frac{L_{ef}^2}{D_et}\right) + 3.3$
DNV (Complex)	$P_f = \frac{2t^*\sigma_u}{(D_e-t^*)}\frac{\left(1-\frac{d^*}{t^*}\right)}{\left(1-\frac{d^*}{t^*M^*}\right)}$	–	$M^* = \sqrt{1 + 0.31\left(\frac{l^*}{\sqrt{D_et^*}}\right)^2}$

Fonte: O autor (2022).

As principais dimensões das amostras de dutos e parâmetros de defeitos são mostradas na Tabela 6 e são: diâmetro externo D_e, espessura nominal t, profundidade máxima d_{max}, comprimento do defeito L_L. Alguns outros parâmetros que caracterizam o defeito são a profundidade média d_{ave} e os valores das proporções $L_L/\sqrt{D_et}$, d/t e d_{ave}/t que dão uma idéia sobre a perda de material e sua forma.

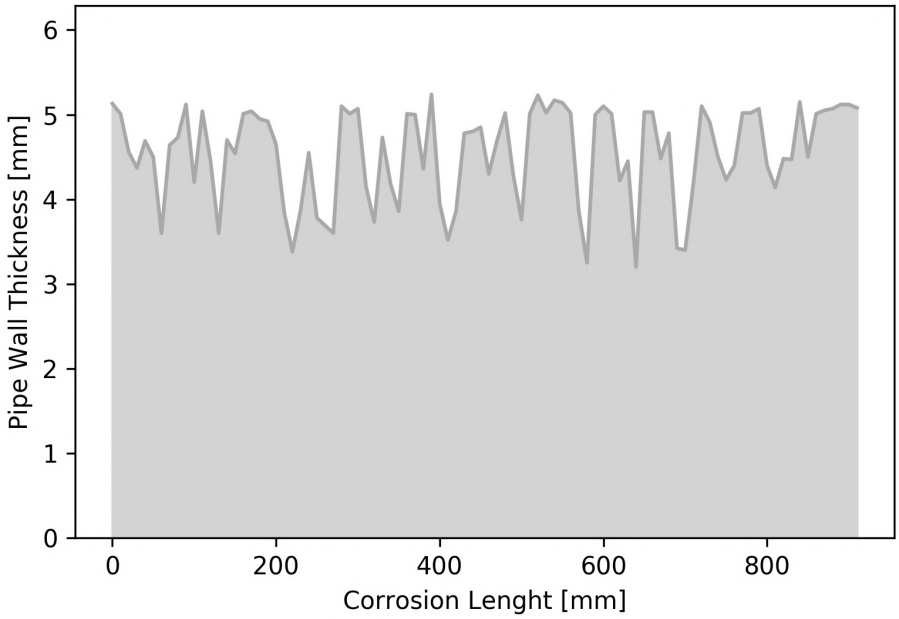
Tabela 6 – Dimensões dos espécimes e os parâmetros dos defeitos.

Espécime	D_e	t	d_{max}	L_L	$L_L/\sqrt{Dt}$	d_{ave}	d/t	d_{ave}/t
TS6.1	273.7	5.24	2.04	910.0	24.03	0.7	0.39	0.13
TS6.2	273.8	5.30	2.52	910.0	23.88	0.8	0.48	0.15

Fonte: O autor (2022).

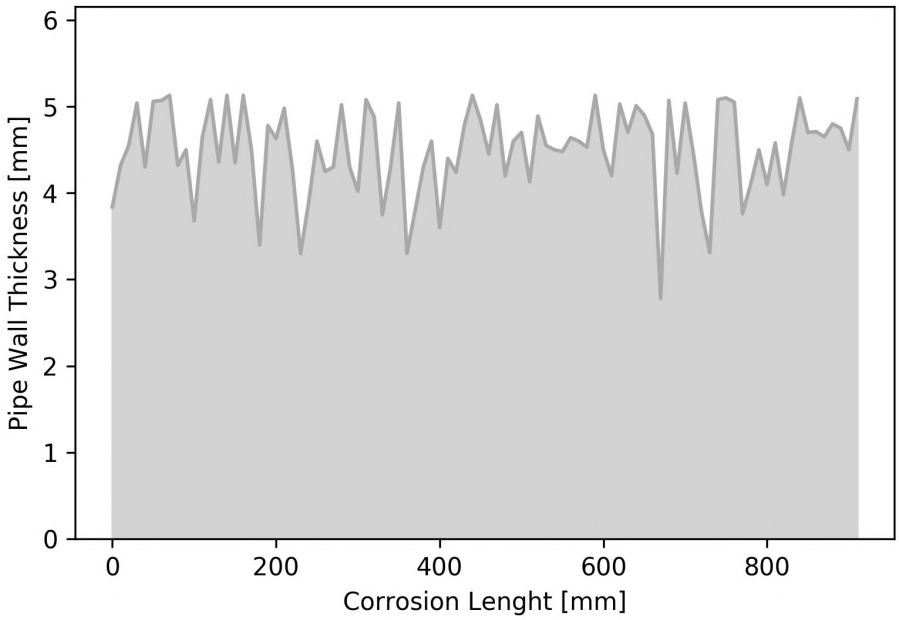
Figuras 50 and 51 mostram o perfil *river bottom* das espécies utilizadas na validação.

Figura 50 – River Bottom Profile do espécime TS6.1.



Fonte: O autor (2022).

Figura 51 – River Bottom Profile do espécime TS6.2.



Fonte: O autor (2022).

A Tabela 7 apresenta os resultados do teste experimental e previstos pelos métodos

B31G, DNV RF-F101 Single, B31G RSTRENG $0,85dL$, *Effective Area* do RSTRENG, DNV RF-F101 para defeitos complexos e elementos finitos (FEM) obtidos com as ferramentas desenvolvidas nesta tese.

Tabela 7 – Pressões de falha experimental e predita (MPa).

Espécime	Teste	B31G	DNV(sing)	0.85dL	Effec Area	DNV(comp)	FEM
TS6.1	15.34	9.77	11.52	10.29	15.37	15.54	15.60
TS6.2	15.53	8.49	10.08	8.93	15.41	15.57	15.50

Fonte: O autor (2022).

De acordo com os resultados apresentados na Tabela 7, os métodos de nível 2 apresentam resultados mais conservativos, porque a pressão de falha é muito menor que a dos resultados experimentais. Os métodos de nível 3 e as análises não lineares via elementos finitos (FEM) apresentam resultados próximos a experimentos em um contexto de engenharia indicando sua eficácia na predição da falha. Nosso grupo de pesquisa tem usado o método dos elementos finitos para avaliar a pressão de falha de tubulações corroídas, porque em aplicações reais diferentes condições de contorno podem ser necessárias, como deslocamento totalmente restrito em tubulações enterradas, momento fletor e temperatura que não são considerados nos métodos de avaliação semi-empírica.

7.1.2 Validação da análise por EF tridimensional

Nesta seção são apresentados resultados da simulação numérica de dutos com defeito de geometria complexa. Estes resultados são comparados e contrastados com resultados experimentais e semi-empíricos de cinco espécimes tubulares, cujos resultados foram encontrados nos trabalhos de (SOUZA et al., 2004) e (SOUZA et al., 2005). Os espécimes aqui utilizados são segmentos que foram removidos de linhas que transportavam petróleo.

Os principais dados relacionados à geometria do duto e dos defeitos que foram utilizados nos procedimentos experimentais e no cálculo utilizando os métodos semi-empíricos estão apresentados na Tabela 8. Esses dados são o diâmetro externo (D_e), espessura nominal do duto (t), profundidade máxima da zona corroída (d), o comprimento da zona corroída (L_L) e dois parâmetros adimensionais que indicam o quão longo e largo o defeito é em relação ao duto, onde L_C representa a largura circunferencial do defeito.

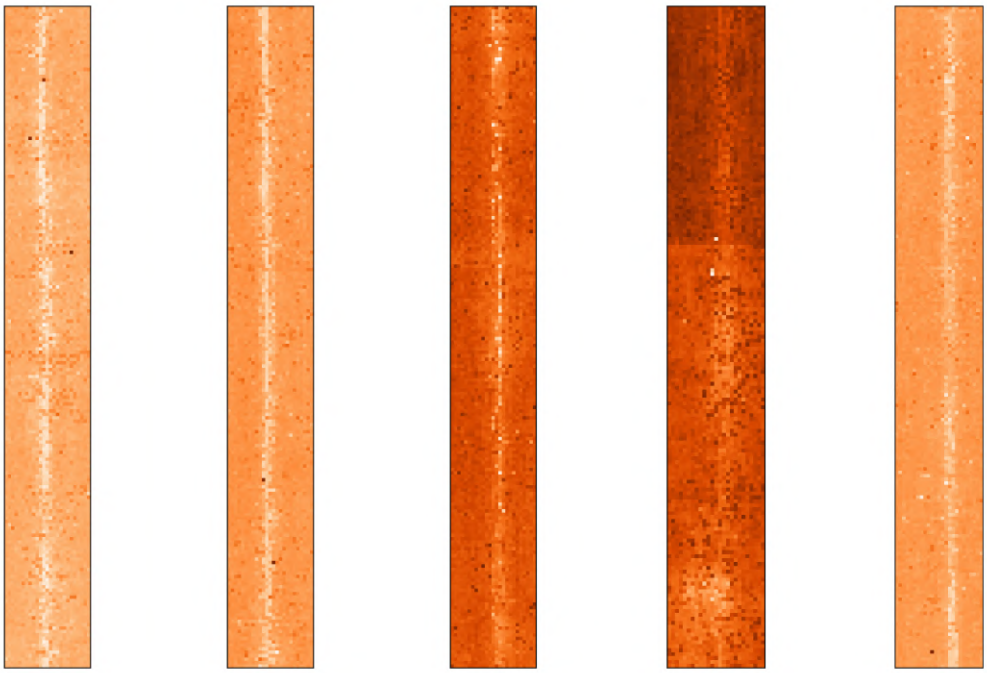
Na região corroída do duto foi criada uma grade uniformemente espaçada sobre a parede externa do duto e sobre esta grade as espessuras remanescentes do duto foram medidas utilizando o aparelho de ultrassom. Na Figura 52 podemos ver a o perfil de corrosão destes dutos. As linhas longitudinais, mais claras, correspondem as profundidades mais elevadas.

Tabela 8 – Número de Elementos

	D_e	t	d	w	L	$L_L/\sqrt{D_e \times t}$	$L_C/\sqrt{D_e \times t}$
TS02	457.7	6.02	2.79	480	3820	0.0728	9.1443
TS04	457.7	6.23	3.27	480	3820	0.0715	8.9889
TS05	457.7	6.04	2.71	480	3820	0.0727	9.1292
TS06	456.5	6.56	3.32	480	3820	0.0698	8.7714
TS10	457.1	6.09	2.80	480	3820	0.0710	8.9236

Fonte: O autor (2022).

Figura 52 – Defeitos de Corrosão Considerados

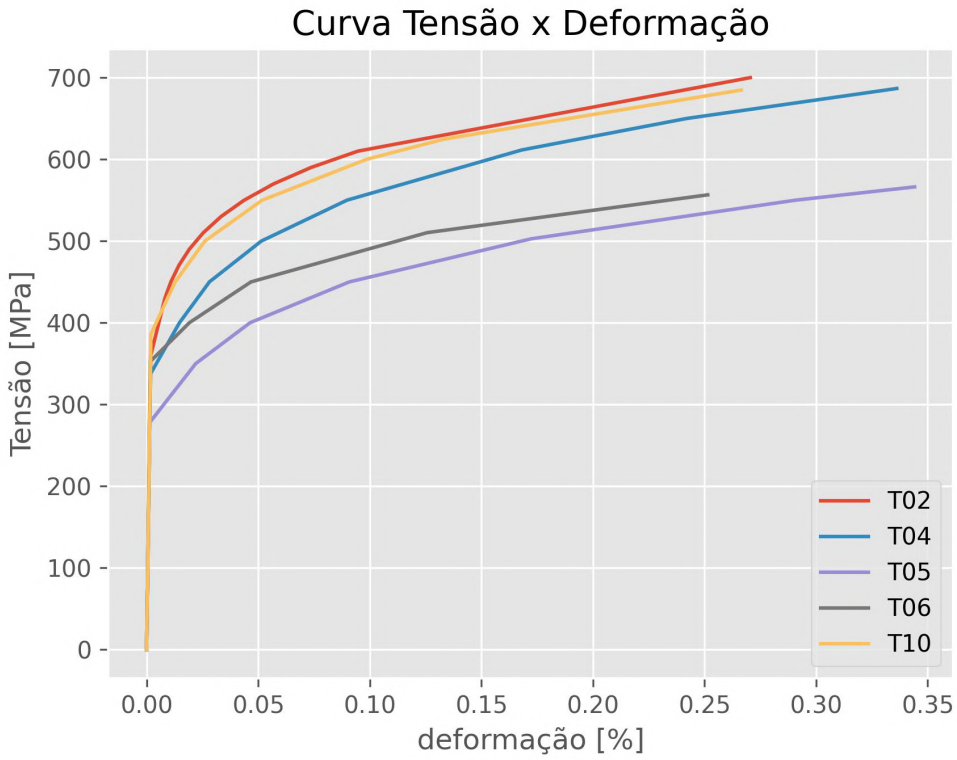


Fonte: O autor (2022).

Para quantificação das propriedades mecânicas dos espécimes experimentais, amostras de material foram removidas e submetidas a ensaios de tração. Cada material teve suas propriedades tensão de escoamento, tensão última e deformação última estimadas. Na Figura 53 são apresentadas as curvas tensão x deformação de cada um dos espécimes analisados. O procedimento utilizado para gerar as mesmas foi descrito na seção 4.3.

Nas extremidades dos dutos ensaiados experimentalmente foram soldadas chapas espessas em suas extremidades. O monitoramento das análises foi feito através de manômetros do tipo Bourdon e por dois instrumentos transdutores de pressão. No final do experimento a pressão de ruptura foi obtida.

Figura 53 – Curvas tensão x deformação



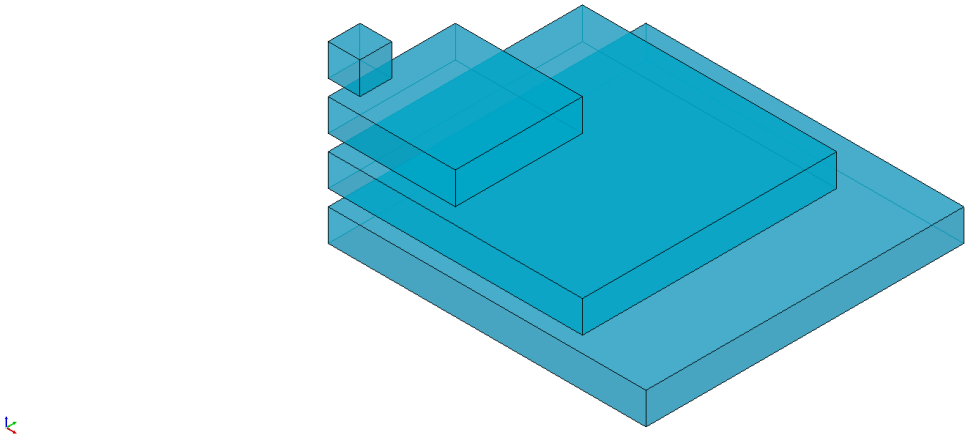
Fonte: O autor (2022).

Para realização da análise via MEF, o procedimento de geração automática desenvolvido neste trabalho foi utilizado. Para verificar até que ponto a malha precisa ser refinada sem afetar a acurácia dos resultados foi realizado um estudo de convergência de malha. Isto traz benefícios em relação ao custo computacional das análises tridimensionais não-lineares.

O estudo foi realizado alterando-se a razão de aspecto dos elementos nos sentidos longitudinal e transversal. O número de elementos ao longo da espessura foi mantido em quatro. As razões de aspecto testadas equivaleram a 4, 8 e 10, pois (FERREIRA, 2011) já havia mostrado que para análise de defeitos reais a razão de aspecto 4 era suficiente. Na Figura 54 podemos observar uma comparação da dimensão dos elementos com as razões de aspecto estudadas em relação a um elemento com razão de aspecto unitária.

Na Tabela 9 podemos observar a quantidade de elementos hexaédricos dos modelos em função das razões de aspecto 4, 8, 10.

Figura 54 – Razão de Aspecto - 1, 4, 8 e 10



Fonte: O autor (2022).

Tabela 9 – Número de Elementos x Razão de Aspecto

	4	8	10
TS02	196112	49440	31680
TS04	183080	46168	29256
TS05	195636	49200	31488
TS06	145188	36520	23408
TS10	188328	46964	30080

Fonte: O autor (2022).

Todos os modelos gerados foram então submetidos a análise não linear utilizando o procedimento automático desenvolvido neste trabalho. Os resultados encontrados da pressão de falha foram então comparados com os resultados experimentais e com os apresentados no trabalho de (PIMENTEL et al., 2020). A Tabela 10 apresenta os valores das pressões de falha encontradas e a Tabela 11 tem a média dos tempos para concluir o processo de análise não linear.

Para o desenvolvimento dos modelos sintéticos que alimentarão a rede neural desenvolvida neste trabalho optou-se pela razão de aspecto 8, pois esta apresentou um resultado acurado e com um custo computacional viável. Na Tabela 12 é apresentado uma comparação entre os resultados encontrados com o modelo EF e os métodos semi-empíricos.

Observando a Tabela 12 concluiu-se que o modelo de EF é menos conservador em todos os casos, sendo a pressão de falha mais que o dobro das encontradas utilizando métodos semi-empíricos mais simples.

Tabela 10 – Pressões de falha (MPa): Experimental x (PIMENTEL et al., 2020) x Razão de Aspecto

	Exp	Cordut	4	8	10
TS02	13.04	12.74	12.625	11.801	11,457
TS04	12.06	10.69	11.726	11.524	11,452
TS05	10.13	8.57	9.942	9.337	9,178
TS06	10.34	10.25	8.427	8.190	8,319
TS10	12.63	11.97	12.600	11.888	11.548

Fonte: O autor (2022).

Tabela 11 – Tempo médio de processamento

	4	8	10
Tempo Médio (h)	13.75	1.74	0.74

Fonte: O autor (2022).

Tabela 12 – Pressões de falha via MEF (MPa) x Métodos Semi-empíricos (MPa)

MEF	B31G	DNV (Single)	Effective Area	DNV (Complex)
11.801	5.38	8.51	10.82	10.64
11.524	5.06	7.43	10.6	10.22
9.337	4.56	6.59	8.7	8.51
8.190	5.59	6.93	7.83	7.63
11.888	5.74	8.49	11.36	11.22

Fonte: O autor (2022).

Tabela 13 – Diferença da pressão de ruptura (%).

	0	1	2	3	4	5	6	7	8	9	10	11
Biorthogonal	0,70	0,36	0,34	0,07	0,36	0,29	-0,21	0,00	-0,15	0,07	-0,07	0,07
Coiflet	0,63	0,08	0,52	0,07	0,36	0,29	0,00	0,00	-0,07	0,49	0,00	0,00
Daubechies	0,63	-0,08	0,52	0,07	0,43	0,29	-0,21	0,00	-0,15	0,00	0,00	0,00
Meyer Discreta	0,40	0,00	0,00	0,00	0,14	0,29	-0,21	0,00	-0,15	0,07	0,00	-0,07
Haar	1,99	5,89	2,01	0,00	0,93	0,73	0,14	0,58	-0,15	0,42	0,65	0,36
Reverse Biortoghonal	0,50	0,00	-2,35	0,00	0,29	0,29	-0,55	0,00	-0,07	-0,21	-0,51	0,00
Symlet	0,56	1,35	0,57	0,14	0,29	0,44	-0,21	-0,22	-0,15	0,07	0,00	0,15

Fonte: O autor (2022).

7.2 VALIDAÇÃO DA TRANSFORMADA *WAVELET* DISCRETA

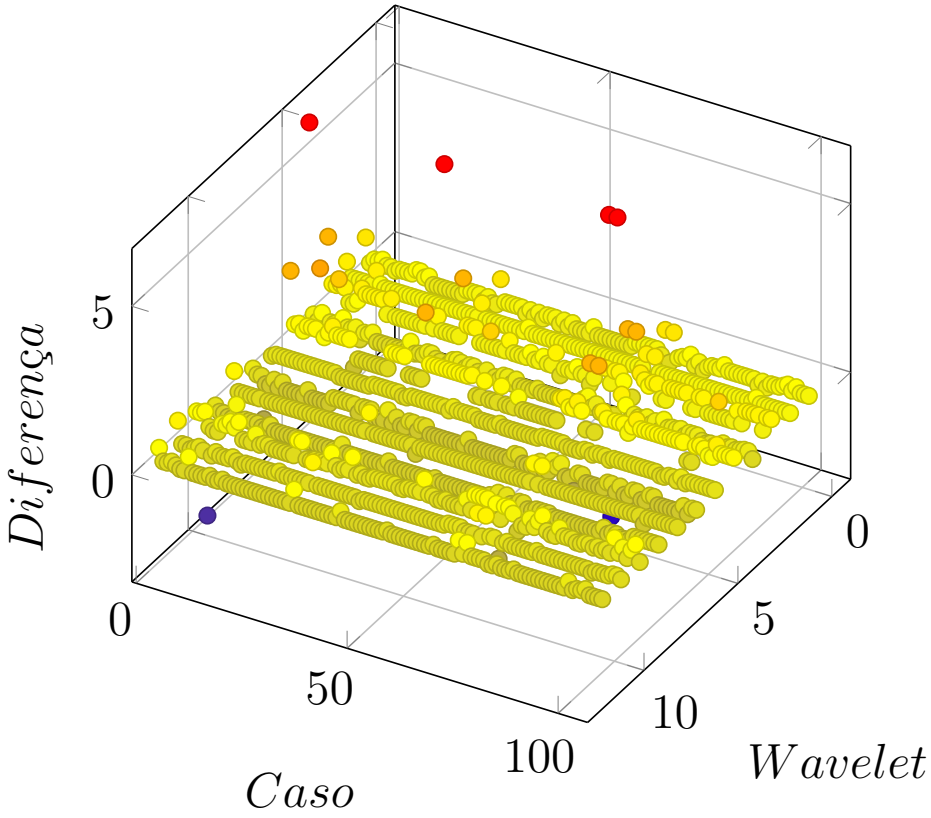
Nesta seção é avaliada a capacidade de se utilizar transformada *wavelet* como filtro de dados em defeitos de corrosão *river bottom* e reais tridimensionais. A decomposição é realizada até o nível máximo e os defeitos são reconstruídos filtrando o máximo de dados possível. Os resultados da análise via MEF dos modelos filtrados são comparados com os resultados dos modelos originais.

7.2.1 Validação para os defeitos *river bottom*

Cento e seis (106) decomposições e reconstruções utilizando famílias diferentes de *wavelets* foram realizadas para cada um dos doze (12) casos, um total de 1272 modelos. Modelos de elementos finitos axissimétricos foram gerados e submetidos à análise não linear. Para esta validação foi considerado apenas o material API 5LX80. A pressão de falha calculada com cada família de *wavelets* é comparada com a pressão calculada com os dados originais para todos os doze modelos. A Tabela 13 mostra a diferença da pressão de ruptura para algumas famílias de *wavelets*.

Através da Tabela 13 podemos observar que as diferenças encontradas são muito pequenas, em sua maioria menores do que 1%. Figura 55 apresenta um gráfico de dispersão com as diferenças das pressões de falha para todos os 12 casos e as 106 *wavelets*.

Figura 55 – Desvio da pressão para os 12 casos e as 106 *wavelets* testadas.

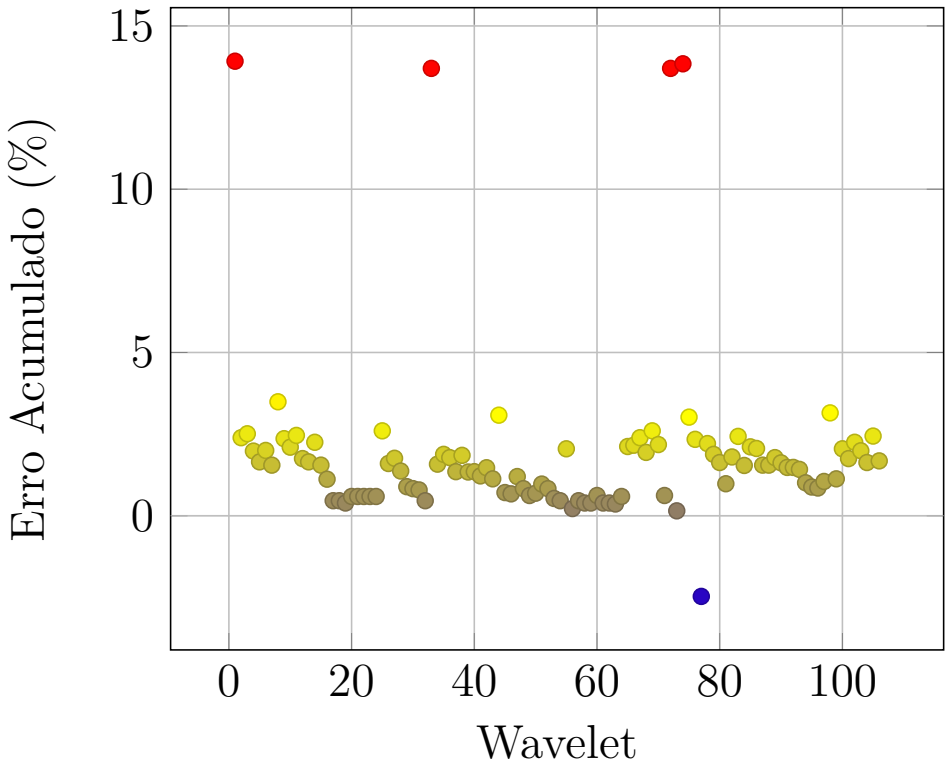


Fonte: O autor (2022).

O erro acumulado para cada tipo de *wavelet* testada, foi calculado somando-se o desvio individual para cada um dos 12 perfis *river-bottom* de corrosão. A Figura 56 apresenta um gráfico de dispersão com todos os resultados. Na Tabela 17 contida no Anexo C, são apresentados os resultados e uma coluna enumerando as *wavelets* em relação ao eixo horizontal da Figura 56. A primeira coluna da tabela contém o nome abreviado junto ao número de momentos, para identificação da *wavelet* utilizada.

Observando a Figura 56, conclui-se que a maioria das famílias de *wavelets* se comportou de maneira satisfatória, resultando em um erro acumulado menor que 5%. Apenas quatro casos se comportaram acima dos 10% e 1 dos casos se comportou de forma conservadora, resultando em um erro acumulado negativo. Considerando apenas o material API 5LX80, a família de *wavelet reverse biorothogonal* com 2 momentos de fuga é escolhida, porque os desvios entre as pressões de falha foram, no geral, menores que 5% e o erro acumulado foi conservador. Os coeficientes necessários, para decomposição e reconstrução utilizando esta família, podem ser visualizados na Figura 57.

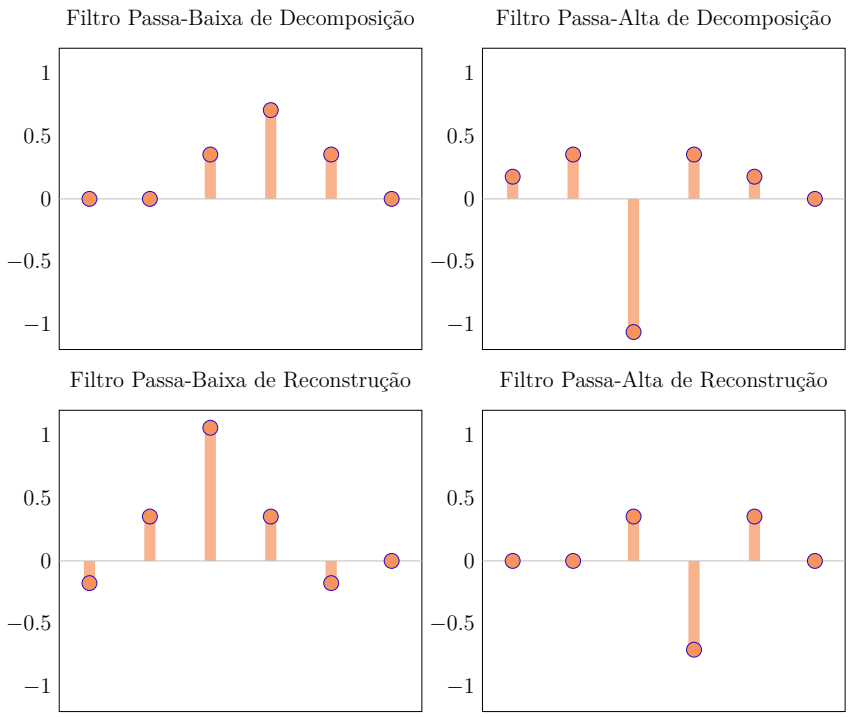
Figura 56 – Erro acumulado para as 106 *wavelets* testadas.



Fonte: O autor (2022).

Onde o eixo vertical representa o valor do coeficiente e o eixo horizontal é o número deste coeficiente.

Figura 57 – Coeficientes de decomposição e reconstrução para Reverse Biorthogonal com 2 momentos.

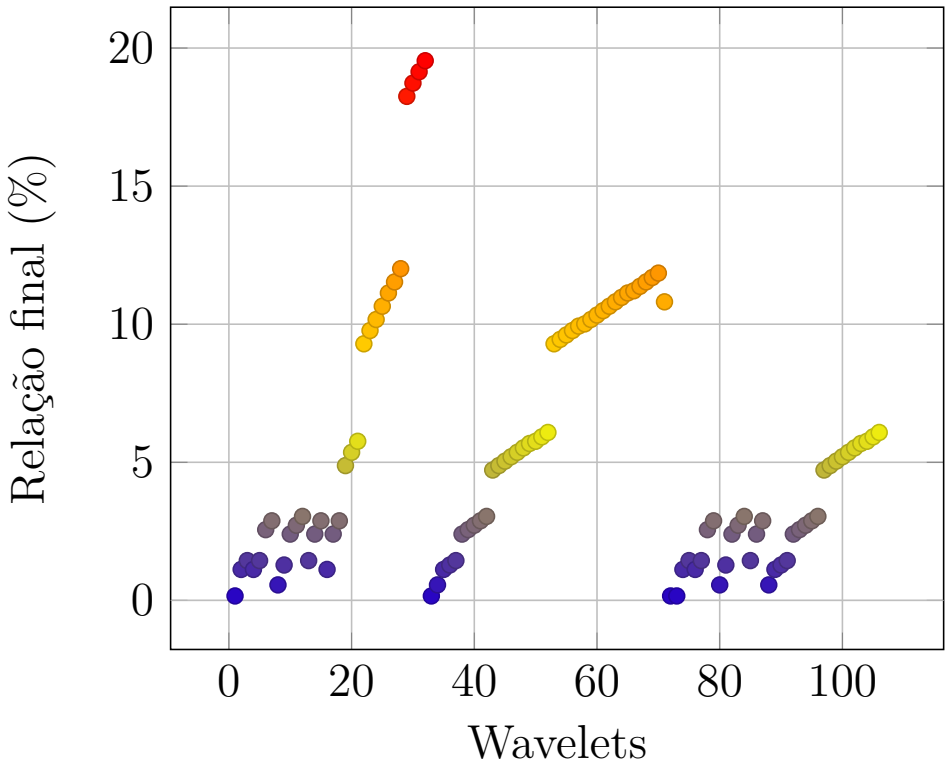


Fonte: O autor (2022).

Neste trabalho, o principal objetivo do uso da decomposição usando transformada *wavelet* é a redução do tamanho dos dados de entrada devido à parametrização das espessuras remanescentes, usando coeficientes de aproximação para representar a região do defeito. A Figura 58 apresenta a relação percentual do tamanho final dos dados após a decomposição de cada uma das 106 *wavelets* para o caso 8 do modelo *river bottom*. Este caso foi escolhido, pois possui o maior número de espessuras remanescentes, um total de 1249. A relação percentual final é calculada pela divisão entre o número de coeficientes da decomposição e o número de pontos de espessura, sendo o resultado multiplicado por 100. A tabela completa com o nome da *wavelet* junto ao número de momentos, a numeração das mesmas em relação ao eixo horizontal da Figura 58, o número de coeficientes e os valores da relação para cada *wavelet*, encontra-se no Anexo D.

Observa-se na Figura 58, que a redução percentual dos dados foi muito alta na maior parte das transformações. Várias famílias de *wavelet*, geraram reduções maiores que 95%, os valores mais baixos são próximos dos 80%.

Figura 58 – Relação percentual final dos dados para o caso 8 utilizando cada uma das 106 *wavelets*.



Fonte: O autor (2022).

A Tabela 14 mostra o número de pontos de espessura remanescentes em cada caso *river bottom*, o número de coeficientes de aproximação no último nível de decomposição utilizando a *wavelet reverse biortoghonal* com dois momentos de fuga. A razão percentual entre o número de coeficientes e o número de pontos também é apresentado na última coluna da tabela.

Na Tabela 14, o número máximo de coeficientes é 14 e o mínimo é 10, uma relação de 1.4, contra uma relação de 8.8 entre 1,249 e 142, o número máximo e mínimo de pontos respectivamente. A proporção entre o número de coeficientes e o número de pontos também é muito pequena, variando de 1.12% a 9.63%, o que significa que a redução de dados foi alcançada com êxito.

Tabela 14 – Razão percentual dos dados (%).

caso	nº pontos	nº coef.	%
0	372	10	2.69
1	798	11	1.38
2	248	12	4.84
3	202	11	5.45
4	403	11	2.73
5	202	11	5.45
6	142	13	9.15
7	290	13	4.48
8	1,249	14	1.12
9	135	13	9.63
10	224	11	4.91
11	294	14	4.76

Fonte: O autor (2022).

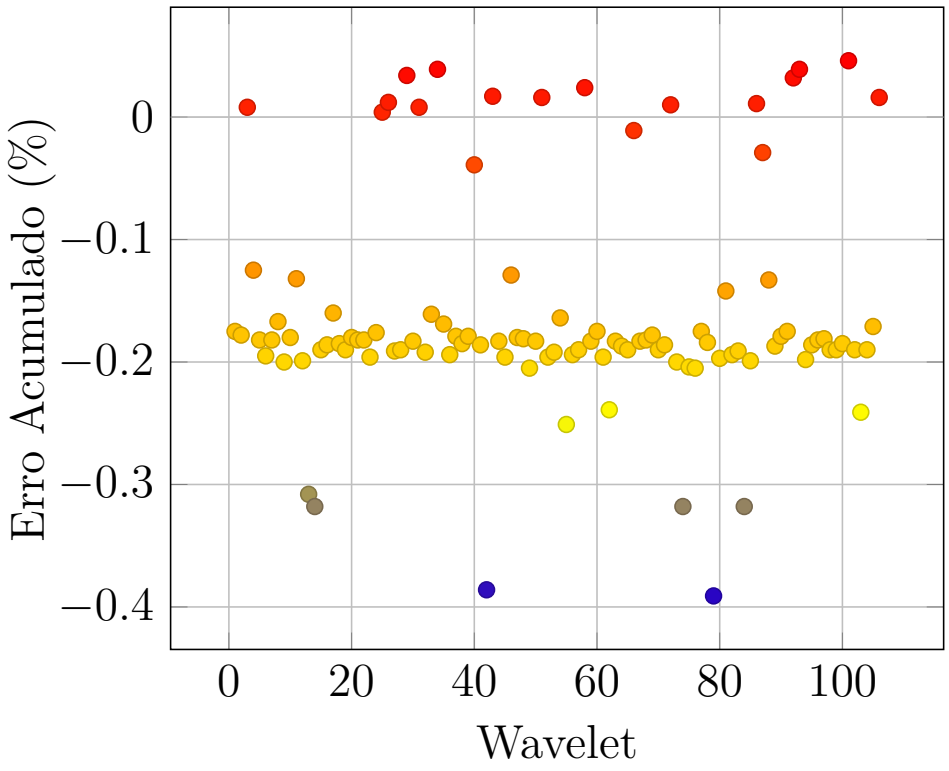
7.2.2 Validação para os casos tridimensionais de geometria complexa

Para validar a utilização das transformada *wavelet* bidimensional em defeitos reais, cento e seis (106) decomposições e reconstruções utilizando famílias diferentes foram realizadas para cada um dos cinco (5) casos. Isso resultou em um total de 530 modelos de elementos finitos tridimensionais que foram gerados e submetidos à análise não linear. A pressão de falha calculada com cada família de *wavelets* é comparada com a pressão calculada com os dados originais para todos os cinco perfis reais.

O erro acumulado para cada tipo de *wavelet* testada, foi calculado somando-se o erro individual para cada um dos 5 perfis reais de corrosão. A Figura 59 apresenta o gráfico de dispersão para todos os resultados.

Observando a Figura 56, conclui-se que a maioria das famílias de *wavelets* se comportou de maneira ótima, resultando em um erro acumulado menor que 1% para todos os casos. A maioria dos resultados se mostrou conservador, o que é bom em termos de engenharia. Isso mostra que *wavelets* bidimensionais apresentam um grande potencial para serem utilizadas como filtro de dados para defeitos reais.

Figura 59 – Erro acumulado para as 106 *wavelets* testadas aplicado aos modelos tridimensionais.



Fonte: O autor (2022).

7.3 APLICAÇÃO DA RNP

Nesta seção são apresentados os resultados da rede neural profunda aplicada a defeitos de corrosão axissimétricos e tridimensionais.

7.3.1 RNP aplicada a defeitos axissimétricos

Para treinar a rede neural, vários parâmetros precisam ser definidos. A função de ativação (ReLU) e o método de otimização (AdaGrad) utilizados nesses estudos foram descritos anteriormente. Todos os dados sintéticos são usados para treinar e validar a rede, a proporção é definida em 70% e 30%, respectivamente. O tamanho dos lotes a serem retornados é definido como 10 e o número de épocas para iterar sobre os dados é definido como 2000.

Na etapa de decomposição, o número mínimo de coeficientes é 10 e o número máximo é 14; portanto, o número de neurônios para a camada de entrada é definido como 15 (número máximo de coeficientes mais número de parâmetros adimensionais). Nos casos em que o número de coeficientes é menor que 14, é utilizada a técnica de preenchimento com zeros (*zero padding*).

Finalmente, a arquitetura da RNP foi construída com quinze neurônios nas camadas

de entrada e ocultas e um único neurônio na camada de saída. A pressão de falha prevista e a pressão de falha do conjunto de teste são comparadas e o erro médio percentual absoluto (MAPE) é calculado como na Equação (7.2).

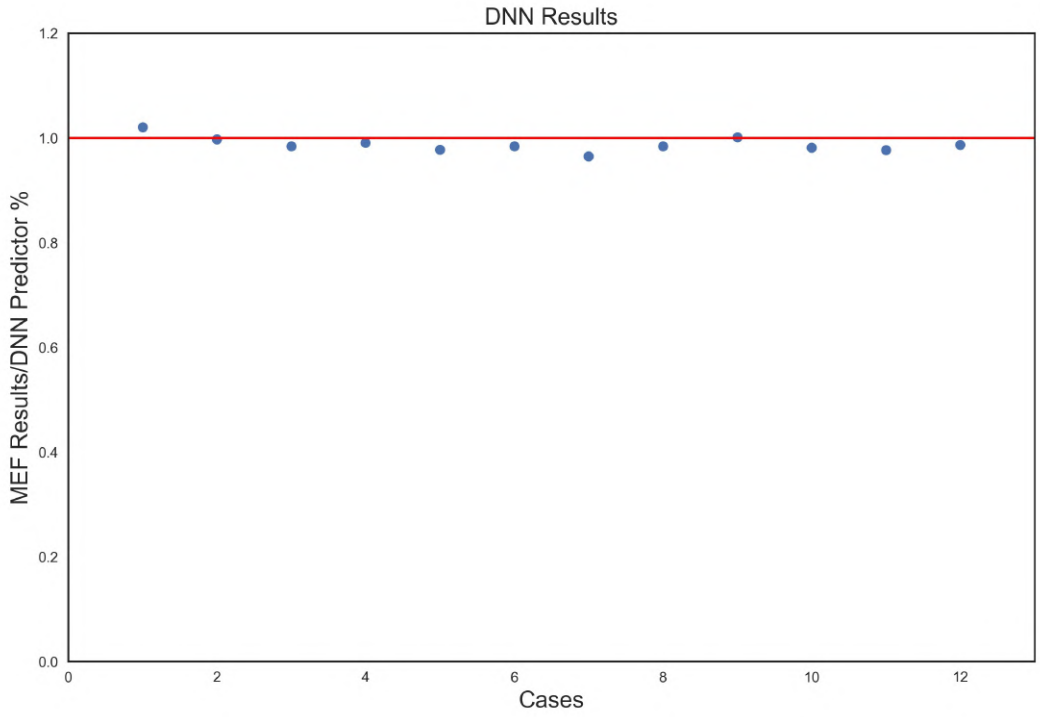
$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{Y_i - \widehat{Y_i}}{Y_i} \right| \tag{7.2}$$

Onde Y_i é a pressão encontrada via MEF, $\widehat{Y_i}$ é a pressão predita e n o número de casos.

O MAPE calculado entre os dados de treinamento e teste é 0.45%, que é um valor muito preciso. A DNN treinada é usada para prever a pressão de falha de defeitos reais com perfil *river bottom*. Com os resultados da pressão de ruptura (simulação via MEF) e os resultados previstos obtidos da rede neural, o MAPE é calculado e o valor de 1.64% de erro é obtido. Isso garante que a rede neural profunda prediz resultados muito precisos da pressão de falha de dutos corroídos, utilizando o perfil *river bottom* e a transformada *wavelets* discreta.

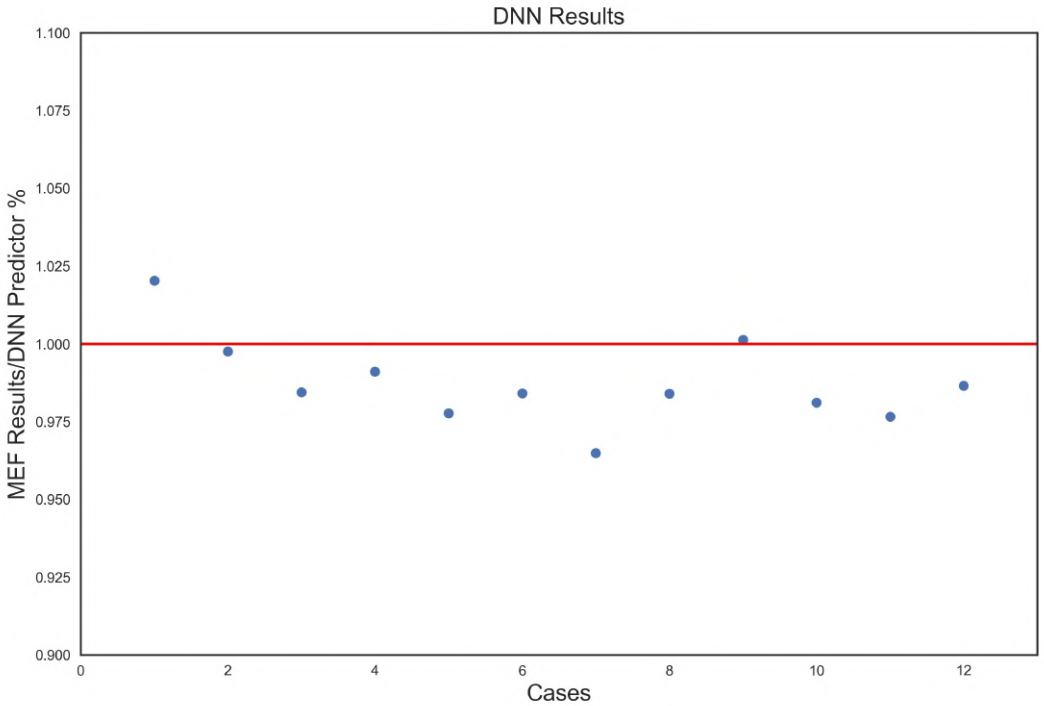
Figura 60 mostra, para todos os 12 casos analisados, os resultados da razão entre a pressão de falha prevista calculada usando análise de elementos finitos e a calculada usando redes neurais profundas. Na Figura 61, podemos ver os detalhes na comparação dos dois métodos usados para estimar a pressão de falha. A diferença máxima entre a predição via MEF e redes neurais profundas foi menor que 5%.

Figura 60 – Comparação entre a pressão de falha calculada utilizando MEF e a predição da DNN.



Fonte: O autor (2022).

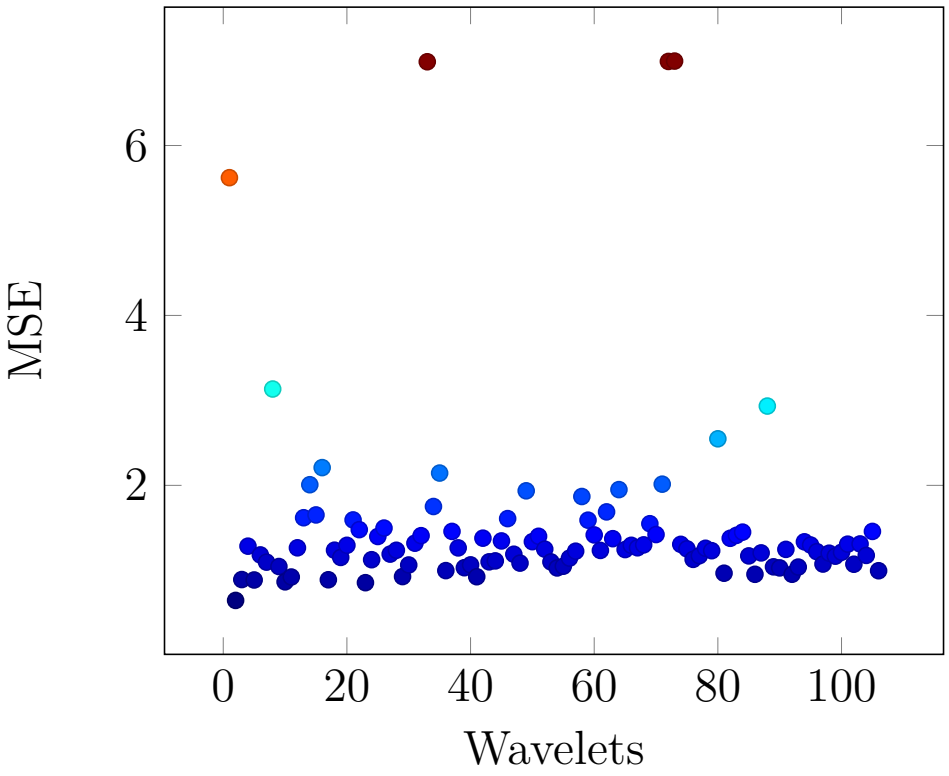
Figura 61 – Detalhe na comparação entre a pressão de falha calculada usando MEF e a predição da DNN.



Fonte: O autor (2022).

A título de comparação, foram treinadas redes neurais utilizando todas as famílias de *wavelets*. A arquitetura da rede neural foi ajustada em função do número de parâmetros obtido a partir da transformada. A quantidade de camadas ocultas foi mantida, sendo alterado apenas o número de neurônios artificiais. A quantidade de épocas de treinamento foi de 30000 e os demais parâmetros foram mantidos. O erro quadrático médio (MSE), conforme apresentado na Equação (B.5) (Apêndice B), foi calculado para cada rede treinada utilizando as 106 *wavelets*. Uma gráfico com os resultados do MSE, pode ser observado na Figura 62. No anexo E é apresentada uma tabela com os nomes das *wavelets*, a numeração das mesmas no eixo horizontal da Figura 62 e os valores do MSE.

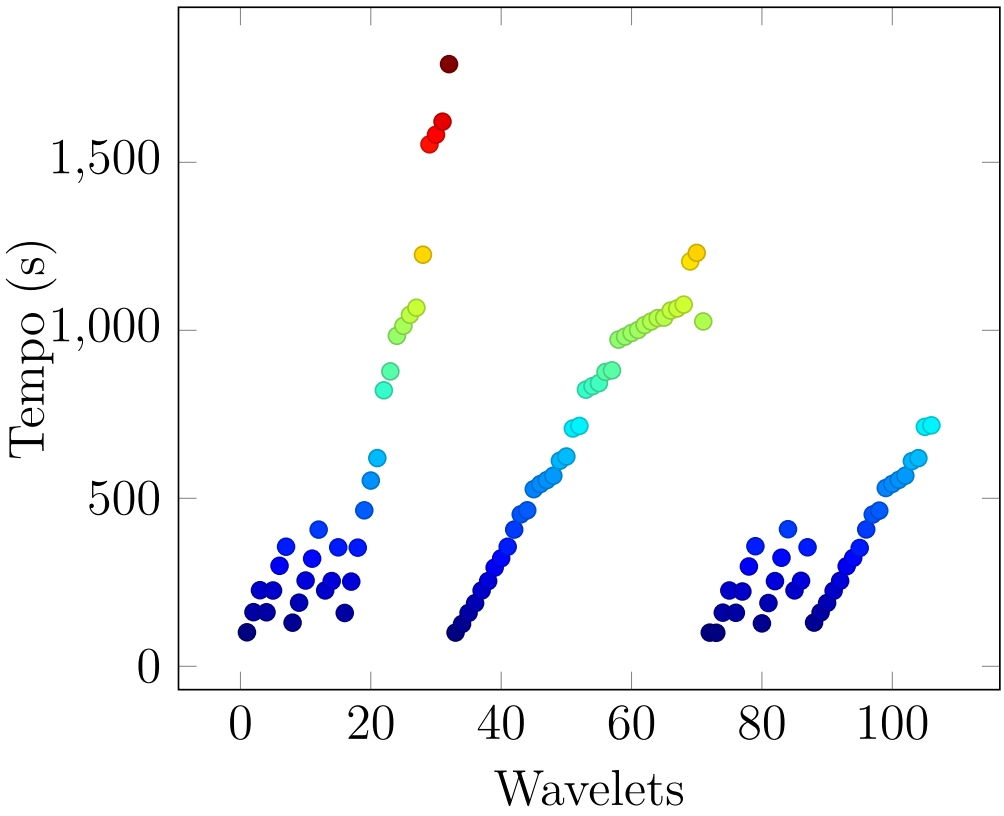
Figura 62 – Erro quadrático médio para todas as 106 *wavelets*.



Fonte: O autor (2022).

Assim, é possível concluir, através da Figura 62, que a grande maioria das *wavelets* gera resultados satisfatórios, pois o MSE calculado é muito baixo. O tempo de treinamento da rede neural para cada tipo de *wavelet* também foi obtido e os resultados encontrados são apresentados na Figura 63. O anexo F contém uma tabela com os nomes das wavelets, a numeração das mesmas no eixo horizontal da Figura 62 e os valores do tempo em segundos.

Figura 63 – Tempo de treinamento da rede para cada *wavelet*.



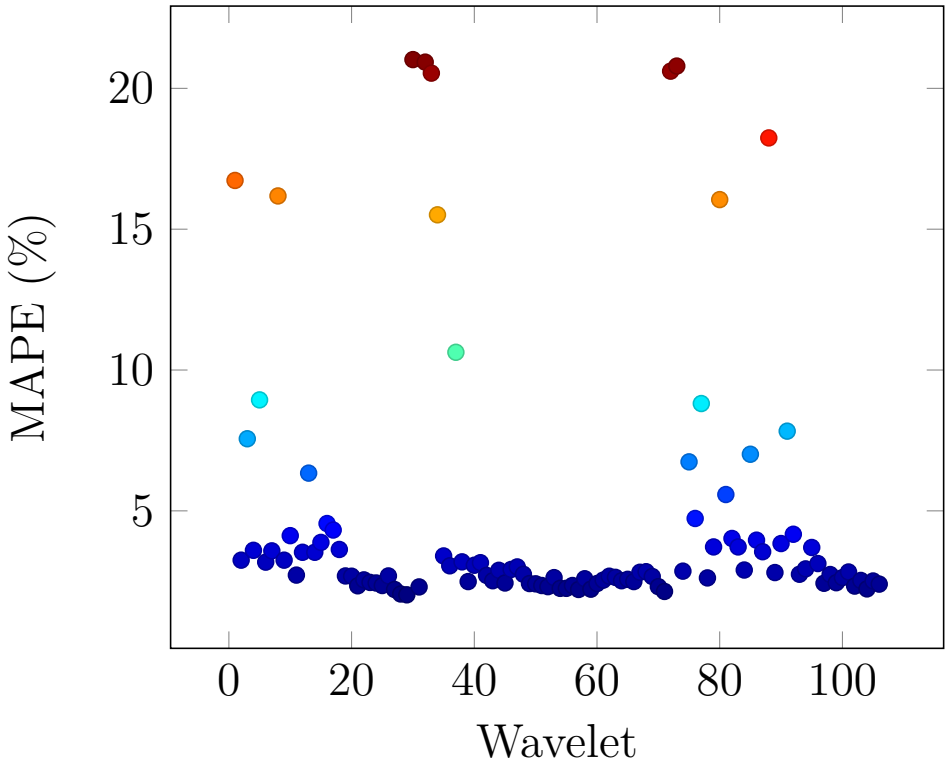
Fonte: O autor (2022).

Observando a Figura 63, temos uma variação muito grande no tempo de treinamento da rede. Esta variação é atribuída ao número de coeficientes obtidos após a transformada, pois isto implica diretamente no número de neurônios na rede, gerando mais variáveis para serem otimizadas e consequentemente aumentando o tempo de processamento. A rede que treinou mais rápido, demorou cerca de 99 segundos e a rede com o tempo de treinamento mais alto demorou cerca de 1800 segundos. A *wavelet* da família *reverse biorthogonal* com dois momentos levou aproximadamente 159 segundos para o treinamento completo.

7.3.2 RNP aplicada a defeitos axissimétricos considerando vários materiais

No total, são gerados cerca de 9612 casos sintéticos associados a um determinado material específico. Como são considerados oito materiais (Tabela 4), isso representa 76896 casos a serem executados. As pressões de falha desses casos são usados como parâmetros de entrada para a rede neural profunda. Os modelos também são gerados aleatoriamente por ajuste estatístico de cada perfil de corrosão. Na Figura 64 temos o erro médio acumulado considerando os oito materiais para cada *wavelet*.

Figura 64 – Erro médio acumulado para as 106 *wavelets* testadas considerando todos os materiais.

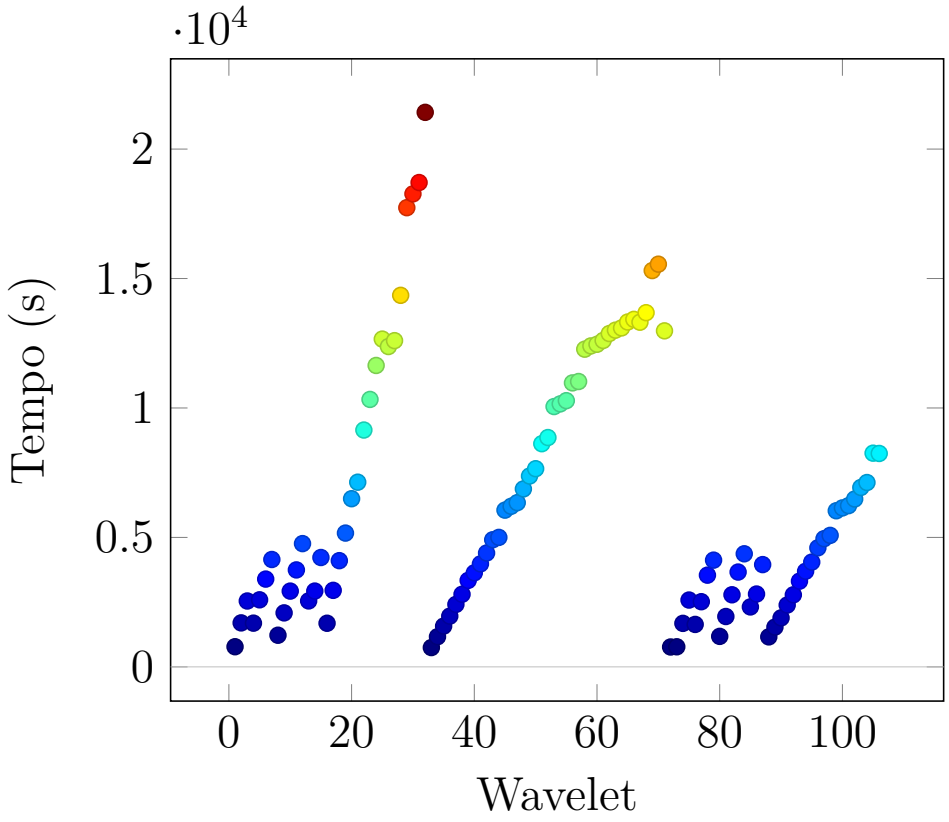


Fonte: O autor (2022).

Observa-se na Figura 64 que quando consideramos todos os materiais temos algumas decomposições que implicam em erros maiores que 20%, embora a grande maioria dos casos se encontra numa região com erro menor que 5%. Conclui-se então que a transformada *wavelet* mesmo considerando vários materiais se mostra eficiente em filtrar informação para os casos de dutos com defeitos do tipo *river bottom*.

Na Figura 65 é apresentado o tempo de duração para o treinamento das redes com cada uma das 106 *wavelets*.

Figura 65 – Tempo de treinamento para as 106 *wavelets* testadas considerando todos os materiais.

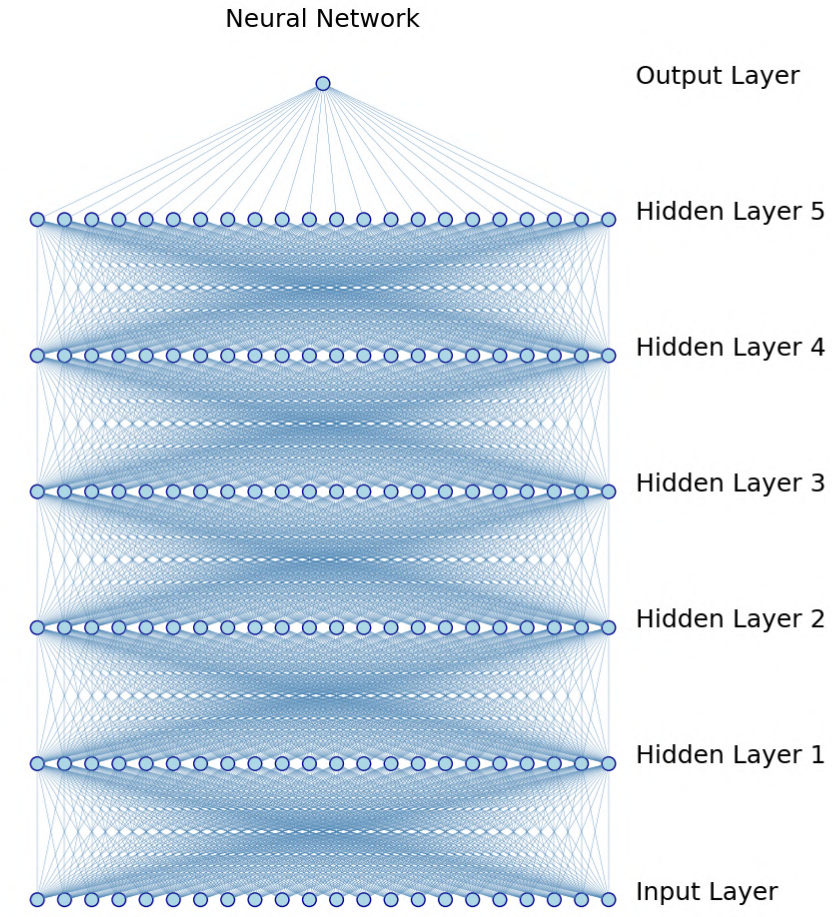


Fonte: O autor (2022).

A duração do treinamento para cada *wavelet* variou entre 778 e 21418 segundos. A partir dos resultados obtidos e aliando o menor erro da rede e o tamanho desta a *wavelet* biortoganal reversa com 3 momentos foi selecionada.

Com a escolha da *wavelet* reversa biortogonal com 3 momentos de fuga, o número de parâmetros máximo obtidos depois da decomposição foi de 18. Somando este ao parâmetro adimensional e as 3 propriedades do material $(\sigma_u, \sigma_e, \epsilon_u)$ chegamos a um total de 22 neurônios na camada de entrada. Um estudo paramétrico foi realizado buscando otimizar os resultados encontrados pela rede e a arquitetura final da rede ficou com 5 camadas ocultas possuindo cada uma delas 22 neurônios (Figura 66).

Figura 66 – Rede neural utilizada para o modelo *river bottom* com diversos materiais.

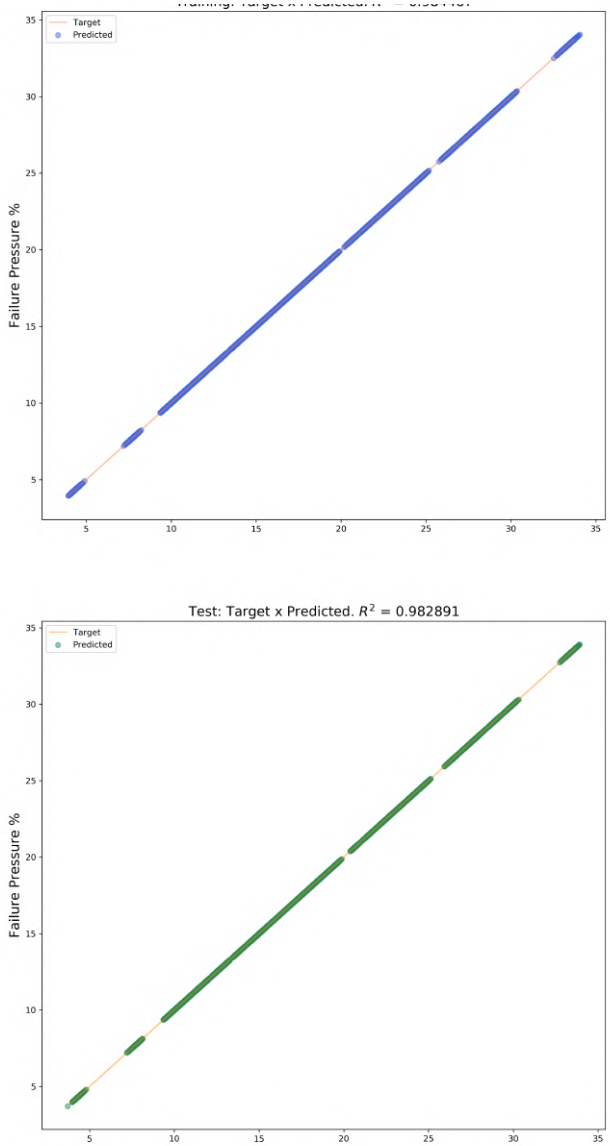


Fonte: O autor (2022).

As pressões de falha previstas no conjunto de teste são comparadas com a pressão de falha fornecida para o conjunto de teste. O MAPE calculado entre elas foi de 2.72%, que é um valor muito preciso.

A Figura 67 mostra a regressão dos valores alvo e os valores previstos para os conjuntos de treinamento e teste, respectivamente. O coeficiente de determinação R^2 (WRIGHT, 1921) também é calculado e apresentado.

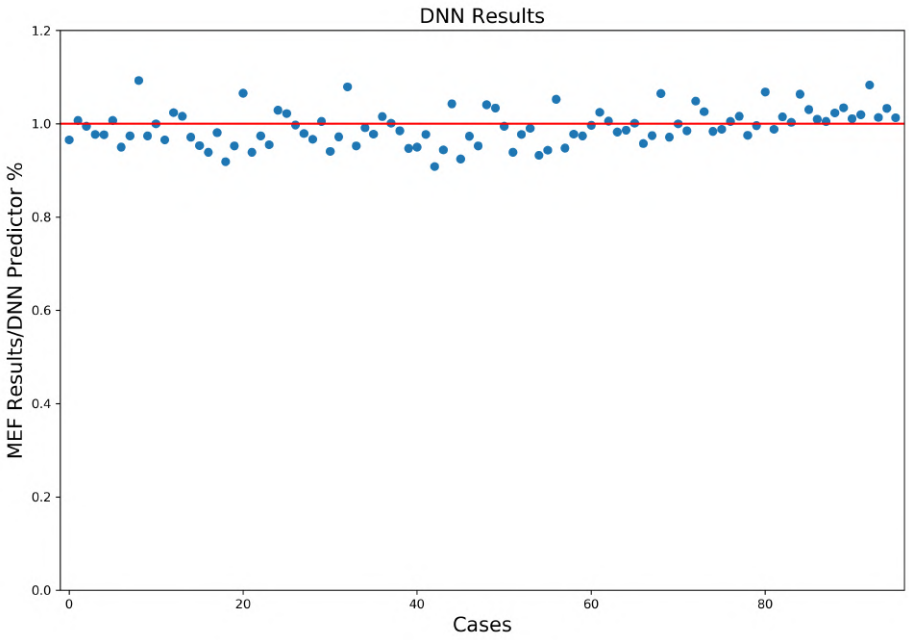
Figura 67 – Regressão para os conjuntos de teste e de treino.



Fonte: O autor (2022).

Como podemos observar o ajuste ficou bom em todo o campo de pressões de falha. Em seguida, o DNN treinado é usado para prever a pressão de falha de defeitos reais do perfil *river bottom*. Com os resultados de pressão de ruptura por MEF e os resultados previstos obtidos da rede neural, o MAPE calculado é igual a 2.85%. O coeficiente R^2 foi de 0.99. Esses cálculos consideraram todos os oito materiais investigados. A Figura 68 mostra o desempenho da RN obtida para os oito diferentes materiais investigados e os doze perfis *river bottom*.

Figura 68 – Resultado obtido via MEF em releção ao predito pela rede.



Fonte: O autor (2022).

A solução FE é a linha vermelha reta. Os cálculos da RN previstos em 96 casos (12 perfis x 8 materiais) são indicados em pontos. Como pode ser visto uma comparação muito boa é obtida. Vale ressaltar que uma única estimativa pelo NN construído levou cerca de $1e-6$ segundos. O que significa 99.9% de economia computacional em comparação com uma solução baseada em FE (modelos RBP e 3D FE). O maior erro em termos absolutos foi menor que 10%.

7.3.3 RNP aplicada a defeitos tridimensionais complexos

O objetivo do trabalho desenvolvido nesta subseção foi utilizar elementos finitos, transformada *wavelet* bidimensional e redes neurais para predição da pressão de falhas em dutos com defeitos reais considerando materiais distintos.

Utilizando todas as 106 *wavelets* foram treinadas redes neurais. Para os modelos tridimensionais, a *wavelet* Daubechies com 1 momento gerou resultados muito bons aliando-se a uma ótima redução no tamanho dos dados. O número máximo de coeficientes com essa *wavelet* é 24, conforme apresentado na Tabela 3.

Os parâmetros de entrada da rede neural para modelos tridimensionais são os coeficientes de aproximação obtidos após a decomposição por *wavelet*, as propriedades do material ($\sigma_u, \sigma_e, \epsilon_u$), e os parâmetros adimensionais d_{ll} e d_{lc} apresentados nas Equações (7.3) e (7.4). O módulo de elasticidade E não foi considerado pois é o mesmo para os materiais dos dutos avaliados.

$$d_{ll} = \frac{L_L}{\sqrt{D_e t}} \tag{7.3}$$

$$d_{lc} = \frac{L_C}{\sqrt{D_e t}} \tag{7.4}$$

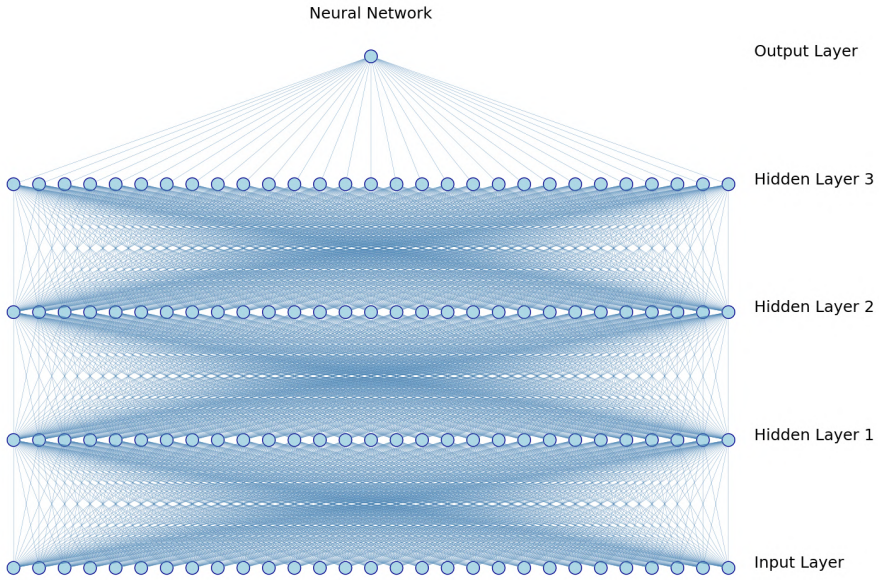
Onde, D_e é o diâmetro externo da tubulação, t é a espessura nominal, L_L é o comprimento longitudinal do defeito de corrosão e L_C é o comprimento circunferencial do defeito de corrosão. Os parâmetros de saída utilizados nas fases de treinamento e validação da rede neural são as pressões de falha obtidas das análises não lineares via MEF utilizando os modelos sintéticos. Na camada de entrada da rede neural, o número de neurônios é definido como $max(n\check{z}coeffs) + 5 = 29$, onde o 5 equivale a soma dos dois parâmetros adimensionais mais as três propriedades do material. Após a realização dos testes, o número de camadas ocultas foi definido como três e o número de neurônios nas camadas ocultas foi definido igual ao número de neurônios de entrada. A camada de saída possui um neurônio. Portanto, a rede neural é composta por uma camada de entrada, três camadas ocultas e uma camada de saída. A arquitetura da rede neural profunda utilizada neste trabalho para esta situação é ilustrada na Figura 69.

A rede neural profunda para casos tridimensionais também teve a proporção entre o conjunto de teste e de validação é fixada em 70% e 30%. O tamanho dos lotes é definido como 100 e o número de épocas para iterar sobre os dados é definido como 10000. A função de ativação utilizada foi a *sigmoid*, também chamada de logística (Figura B.4), e o otimizador foi o AdaGrad. As pressões de falha são previstas no conjunto de teste. As comparações são feitas com a pressão de falha fornecida no conjunto de teste. O MAPE calculado entre eles é de 1.6%. O tempo total de treinamento da rede foi de 114 segundos.

Na Figura 70 temos os resultados obtidos do MEF em relação aos previstos pela rede neural. Os pontos destacados na cor azul representam os casos em que modelos sintéticos foram gerados e utilizados para treinar a rede. O último resultado a direita (em laranja) representa a simulação de um novo caso que não teve modelo sintético alimentando a rede.

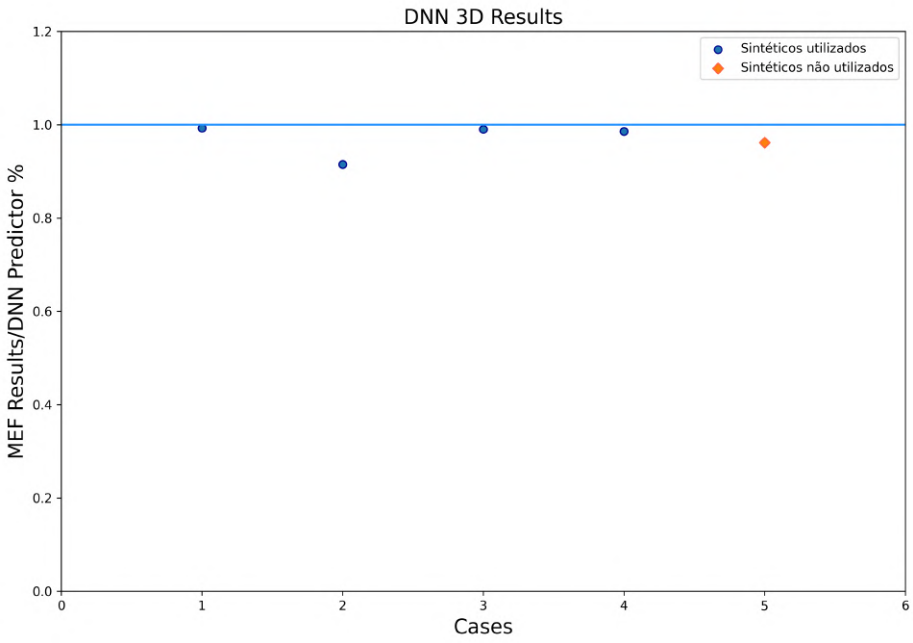
O MAPE encontrado para a rede é de de aproximadamente 3%. Como pode ser observado na Figura 70, todos os resultados que tiveram casos sintéticos utilizados no treinamento tiveram bons resultados, estando 3 destes com 99% de previsão. O caso 5,

Figura 69 – Rede Neural para os casos tridimensionais.



Fonte: O autor (2022).

Figura 70 – Resultados previstos pela rede em relação aos resultados via MEF para os modelos 3D.



Fonte: O autor (2022).

que não teve sintéticos relacionados, gerou um resultado com 96% de precisão em relação ao MEF.

8 CONCLUSÕES E TRABALHOS FUTUROS

8.1 CONCLUSÕES

Neste trabalho, um sistema híbrido foi desenvolvido utilizando método dos elementos finitos, análise multiresolução e redes neurais para avaliação, em tempo hábil, de integridade de dutos corroídos. O MEF foi utilizado para analisar defeitos sintéticos que alimentaram a rede neural e a análise multiresolução foi utilizada para parametrizar perfis reais de corrosão permitindo que seja possível alimentar uma rede neural.

Várias famílias de *wavelets* foram testadas para encontrar a melhor decomposição que pode ser aplicada a defeitos reais. Rotinas para a modelagem automática foram criados com sucesso, permitindo a geração de modelos de elementos finitos de dutos axissimétricos com corrosão de perfil *river bottom* e modelos tridimensionais complexos de forma eficiente.

Os resultados da comparação do estudo de análise não linear via MEF com os resultados experimentais e semi-empíricos indicaram que a modelagem, as condições de contorno e carregamento, os critérios de falha e o procedimento de análise adotados levam a uma estimativa de pressão de ruptura muito precisa.

Perfis de corrosão sintéticos preservando as propriedades estatísticas de defeitos reais e análise via método dos elementos finitos foram utilizados para estimar a pressão de falha em cada um deles.

Após cada rede ser treinada, elas tiveram suas predições confrontadas com resultados obtidos via MEF. As seguintes conclusões foram obtidas neste trabalho:

- A transformada *wavelet* discreta apresentou-se bastante promissora na sua utilização como filtro de perfis reais de corrosão em dutos. Os resultados das análises não lineares via MEF apontam que todas as famílias de *wavelet* testadas não implicaram em grandes desvios na pressão de falha, em relação às análises feitas com os dados originais tanto para os modelos axissimétricos quanto para os modelos 3D.

- A análise multiresolução permitiu a parametrização de configurações de defeitos reais obtidas a partir de dados de inspeção ultrassônica. Múltiplos casos de defeitos *river bottom* com comprimentos diferentes puderam ser representados por um máximo de quatorze coeficientes através da decomposição usando *wavelet* biortogonal reversa. As transformadas *wavelet* e a aplicação de um filtro passa-baixa permitiram que o tamanho dos dados fosse reduzido para uma faixa de 10% a 1,1% do tamanho dos dados originais e a razão máxima entre eles mudasse de 8.8 usando os dados originais para 1.4 com os dados compactados.

- A aplicação da transformada *wavelet* bidimensional aplicada em defeitos reais foi capaz de reduzir em 99.5% o tamanho dos dados sem desconfigurar o defeito original, o que pode ser considerado ótimo dado a quantidade de dados que caracterizam uma corrosão real.

- Os modelos sintéticos podem reproduzir perfis de corrosão complexos reais, desde que a distribuição das espessuras remanescentes seja ajustada para cada conjunto de dados. Nos defeitos axissimétricos o ajuste individual se deu avaliando a soma dos erros quadráticos entre as pdf's de cada distribuição. Para o defeito real tridimensional a estimativa por KDE foi utilizada para se conseguir um ajuste adequado.

- Redes neurais profundas foram aplicadas para prever com precisão as pressões de falha de dutos com defeitos *river bottom* e tridimensionais. Os resultados para os casos axissimétricos considerando apenas o material $API5LX80$, indicaram que a rede neural profunda pode prever com erro menor que 2%, quando comparado a solução via MEF. Quando considerados todos os 8 materiais do tipo $API5LX$ o desvio ficou em 2.85%. A rede neural apresentou um MAPE de aproximadamente 3% na predição de modelos tridimensionais, mesmo incluindo um defeito que não teve casos sintéticos utilizados no treinamento. O coeficiente R^2 computado em todas tanto nas redes neurais com modelos *river bottom* quanto nos modelos tridimensionais ficou em torno de 0.99. Esses resultados mostram que redes neurais podem servir como modelos substitutos para avaliação da integridade de dutos sendo uma alternativa eficiente à análise de elementos finitos ou métodos semi-empíricos.

- O tempo para avaliação da integridade de dutos utilizando este sistema é quase instantâneo e quando comparado ao MEF, considerando o tempo de modelagem e análise, o ganho de tempo é muito alto, viabilizando a utilização na indústria e em sistemas *Digital Twin*.

O uso de sistemas híbridos, incluindo diferentes métodos e técnicas, pode ser uma ferramenta poderosa para resolver problemas complexos. Nesse caso em particular, a transformada *wavelet*, o MEF e o aprendizado de máquina foram usados juntos para ajudar os engenheiros da indústria de petróleo e gás na tomada de decisões quando são encontrados defeitos de corrosão. A análise multiresolução é de fundamental importância para manipular os dados e manter o comprimento dos dados na mesma ordem de grandeza. Os sistemas propostos podem prever em tempo real a pressão de falha de dutos com defeitos de corrosão do tipo *river bottom* axissimétrica e reais tridimensionais, permitindo aplicação em sistemas de gêmeos digitais e programas de planejamento de manutenção, onde a previsão correta do tempo é de fundamental importância para evitar acidentes.

8.2 TRABALHOS FUTUROS

O trabalho aqui desenvolvido possui espaço para diversos estudos futuros. Dentre as propostas de continuidade do trabalho destacam-se:

- A rede neural poderá ser treinada considerando modelos de crescimento e taxas de corrosão para predição de falha futura.
- Os parâmetros que definem a arquitetura da rede neural, como número de camadas, neurônios por camada, função de ativação, número de épocas e etc, poderiam ser obtidos via algoritmos de otimização, podendo encontrar dessa forma a rede que fornece os melhores resultados de uma maneira mais adequada.
- O modelo substituto também pode ser usado para realizar análises futuras que podem ajudar no planejamento da manutenção.
- Distribuições estatísticas de cada parâmetro de entrada podem ser usadas para gerar modelos aleatórios. Esses modelos podem ser decompostos através da transformada *wavelet* e usados para alimentar uma rede neural profunda que poderia realizar análises de confiabilidade em apenas alguns segundos.
- O modelo substituto poderia ser utilizado na otimização de intervalos de manutenção, visto que este último pode trazer ganhos financeiros para as empresas e tendo um modelo substituto confiável viabilizaria a realização deste tipo de avaliação que exige muito poder computacional.

REFERÊNCIAS

ABBASSL, T. E. et al. A comparison of surfacefitting algorithmsfor geophysical data. *Terra Research*, 1991. Citado na página 38.

Abdalla Filho, J. et al. On the failure pressure of pipelines containing wall reduction and isolated pit corrosion defects. *Computers Structures*, v. 132, p. 22–33, 2014. ISSN 0045-7949. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0045794913002964>. Citado na página 22.

ALOWAISI, S.; BECKER, A.; SUN, W. Analysis of shape and location effects of closely spaced metal loss defects in pressurised pipes. *Engineering Failure Analysis*, v. 68, 04 2016. Citado 2 vezes nas páginas 53 e 54.

ANDRADE, E. Q. et al. Finite element modeling of the behavior of pipelines containing interacting corrosion defects. In: *4th International Conference on Offshore Mechanics and Arctic Engineering 25th International Conference on Offshore Mechanics and Arctic Engineering*. [S.l.: s.n.], 2006. Citado na página 27.

Finite Element Modeling of the Failure Behavior of Pipelines Containing Interacting Corrosion Defects, Volume 4: Terry Jones Pipeline Technology; Ocean Space Utilization; CFD and VIV Symposium de *International Conference on Offshore Mechanics and Arctic Engineering*, (International Conference on Offshore Mechanics and Arctic Engineering, Volume 4: Terry Jones Pipeline Technology; Ocean Space Utilization; CFD and VIV Symposium). 315-325 p. Disponível em: <https://doi.org/10.1115/OMAE2006-92600>. Citado 3 vezes nas páginas 24, 27 e 54.

ASME. *Manual for Determining the Remaining Strength of Corroded Pipelines*. [S.l.]: American Society of Mechanical Engineers, 2009. Citado na página 22.

ASME boiler and pressure vessel code: an international code. I - Rules for construction of power boiler. New York, NY: American Society of Mechanical Engineers, 2010. Citado na página 26.

ASTER, C. *Documentation 15*. https://www.code-aster.org/, 2019. Citado 4 vezes nas páginas 17, 29, 46 e 55.

AZEVEDO, F. A. et al. Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology*, v. 513, n. 5, p. 532–541, 2009. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cne.21974>. Citado na página 75.

BAO, J.; ZHOU, W. A random field model of external metal-loss corrosion on buried pipelines. *Structural Safety*, v. 91, p. 102095, 2021. ISSN 0167-4730. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167473021000205>. Citado 2 vezes nas páginas 25 e 43.

BENJAMIN, A. C. et al. Burst tests on pipeline containing interacting corrosion defects. In: *4th International Conference on Offshore Mechanics and Arctic Engineering*. [S.l.: s.n.], 2005. Citado na página 27.

Burst Tests on Pipeline Containing Closely Spaced Corrosion Defects, Volume 4: Terry Jones Pipeline Technology; Ocean Space Utilization; CFD and VIV Symposium de *International Conference on Offshore Mechanics and Arctic Engineering*, (International Conference on Offshore Mechanics and Arctic Engineering, Volume 4: Terry Jones Pipeline Technology; Ocean Space Utilization; CFD and VIV Symposium). 103-116 p. Citado 2 vezes nas páginas 53 e 54.

BHOWMIK, S. *Digital Twin of Subsea Pipelines: Conceptual Design Integrating IoT, Machine Learning and Data Analytics*. Houston, Texas: Offshore Technology Conference, 2019. 9 p. OTC. Disponível em: <https://doi.org/10.4043/29455-MS>. Citado 2 vezes nas páginas 21 e 22.

BRUÈRE, V. et al. Failure pressure prediction of corroded pipes under combined internal pressure and axial compressive force. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, v. 41, p. 172, 03 2019. Citado na página 24.

CABRAL, H. D. L. *Desenvolvimento de Ferramentas Computacionais para Modelagem e Análise Automática de Defeitos de Corrosão em Dutos*. Dissertação (Master Dissertation) — Universidade Federal de Pernambuco - UFPE, Recife, PE, 2007. In Portuguese. Citado na página 24.

CABRAL, H. L. D. et al. The development of a computational tool for generation of high quality fe models of pipelines with corrosion defects. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, v. 39, 06 2017. Citado na página 16.

CABRAL, R. M. et al. Assessment by finite element modeling of pipelines with corrosion defects based on river-bottom profile model. *Engineering Structures*, v. 261, p. 114246, 2022. ISSN 0141-0296. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0141029622003741>. Citado 2 vezes nas páginas 16 e 22.

CHOI, J. B. et al. Development of limit load solutions for corroded gas pipelines. *International Journal of Pressure Vessels and Piping*, 2003. ISSN 03080161. Citado 3 vezes nas páginas 24, 53 e 54.

CHOI, K.-H. et al. Comparison of computational and analytical methods for evaluation of failure pressure of subsea pipelines containing internal and external corrosion. *Journal of Marine Science and Technology*, v. 21, 12 2015. Citado 2 vezes nas páginas 53 e 54.

CHOUCHAOUI, B. A.; PICK, R. J.; YOST, D. B. Burst pressure predictions of line pipe containing single corrosion pits using the finite element method. In: *11th International Conference on Offshore Mechanics and Arctic Engineering*. [S.l.: s.n.], 1992. Citado 2 vezes nas páginas 53 e 54.

CHUI, C. K. *An Introduction to Wavelets*. 1st edition. ed. [S.l.]: Academic Press, 1992. (Wavelet Analysis and Its Applications 1). ISBN 0121745848,9780121745844,0121745651,9780121745653,9780585470900. Citado na página 62.

COHEN, L. The uncertainty principle for the short-time fourier transform and wavelet transform. 01 2001. Citado na página 62.

COOLEY, J. W.; TUKEY, J. W. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, American Mathematical Society, v. 19, n. 90, p. 297–301, 1965. ISSN 00255718, 10886842. Disponível em: <http://www.jstor.org/stable/2003354>. Citado 2 vezes nas páginas 62 e 122.

COSHAM, A.; HOPKINS, P. The pipeline defect assessment manual. In: AMERICAN SOCIETY OF MECHANICAL ENGINEERS DIGITAL COLLECTION. *2002 4th International Pipeline Conference*. [S.l.], 2002. p. 1565–1581. Citado 2 vezes nas páginas 20 e 82.

DHONDT, G. *CalculiX CrunchiX USER'S MANUAL version 2.1*. 2011. Citado 2 vezes nas páginas 29 e 52.

DIERCKX, P. An algorithm for surface fitting with spline functions. *IMA Journal of Numerical Analysis*, v. 1, 1981. Citado na página 38.

DIN, M. M. et al. An artificial neural network modeling for pipeline corrosion growth prediction. *ARPN Journal of Engineering and Applied Sciences*, v. 10, p. 512–519, 02 2015. Citado na página 26.

DNV. *DNVGL-RP-F101 Corroded pipelines*. [S.l.]: DET NORSKE VERITAS, 2017. Citado 2 vezes nas páginas 22 e 26.

DUCHI, J.; HAZAN, E.; SINGER, Y. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, JMLR.org, v. 12, p. 2121–2159, jul. 2011. ISSN 1532-4435. Citado na página 80.

EDF. *Documentation*. https://www.code-aster.org/V2/docsmeca/default/en/?lang=en, 2020. Citado na página 47.

ENDERLEIN, G. Wilks, s. s.: Mathematical statistics. j. wiley and sons, new york-london 1962; 644 s., 98 s. *Biometrische Zeitschrift*, v. 6, n. 3, p. 214–215, 1964. Citado na página 35.

EŔIC, L. *Modeling elásto (visco) plastic with isotropic work hardening in great deformations*. https://www.code-aster.org/, 2013. Citado na página 51.

EŔIC, L. *Various syntaxes: files .export*. https://www.code-aster.org/V2/doc/v14/en/man_d/d1/d1.02.05.pdf, 2013. Citado na página 50.

FERREIRA, A. D. M. *Ferramentas Computacionais para Análise Estrutural de Dutos com Defeitos Reais*. Dissertação (Master Dissertation) — Universidade Federal de Pernambuco - UFPE, Recife, PE, 2011. In Portuguese. Citado 3 vezes nas páginas 24, 42 e 88.

FERREIRA, A. D. M. et al. Stochastic assessment of burst pressure for corroded pipelines. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, v. 43, n. 4, p. 193, mar. 2021. Citado 2 vezes nas páginas 24 e 43.

GATES, M. *Fortran 90/95 reference*. http://www.icl.utk.edu/ mgates3/docs/fortran.html, 2020. Citado na página 47.

GHANEM, R. Probabilistic characterization of transport in heterogeneous media. *Comput. Methods Appl. Mech. Engrg*, v. 158, p. 199–220, 1998. Citado na página 43.

GONZALEZ, R. E. W. R. C. *Digital Image Processing*. 2nd ed. ed. [S.l.]: Prentice Hall, 2002. ISBN 9780201180756,0201180758,0130946508,9780130946508. Citado 2 vezes nas páginas 63 e 65.

GUPTA, D. *Fundamentals of Deep Learning - Activation Functions and When to Use Them?* 2017. Disponível em: <https://www.analyticsvidhya.com/>. Citado na página 80.

HAAR, A. Zur theorie der orthogonalen funktionensysteme. *Mathematische Annalen*, v. 69, n. 3, p. 331–371, Sep 1910. ISSN 1432-1807. Disponível em: <https://doi.org/10.1007/BF01456326>. Citado na página 63.

HAN, L.; HAN, L.; LIU, C. Neural network applied to prediction of the failure stress for a pressurized cylinder containing defects. *International Journal of Pressure Vessels and Piping*, v. 76, n. 4, p. 215 – 219, 1999. ISSN 0308-0161. Citado na página 26.

HAN, Y.-L.; SHEN, S.-M.; DAI, S.-H. Artificial neural network technology as a method to evaluate the failure bending moment of a pipe with a circumferential crack. *International Journal of Pressure Vessels and Piping*, v. 68, n. 1, p. 1 – 6, 1996. ISSN 0308-0161. Citado na página 26.

INC., K.; LABORATORY., L. A. N. *Paraview project*. https://www.paraview.org/, 2020. Citado na página 47.

IYER, A. *Digital Twin: Possibilities of the new Digital twin technology*. 1. ed. [s.n.], 2018. Disponível em: <https://www.amazon.com.br/Digital-Twin-Possibilities-technology-English-ebook/>. Citado na página 20.

KELLEY, C. T. *Solving Nonlinear Equations with Newton's Method*. [S.l.]: Society for Industrial and Applied Mathematics, 2003. Citado na página 52.

KIEFNER, J. F. et al. Failure stress levels of flaws in pressurized cylinders. In: . [S.l.: s.n.], 1973. Citado na página 26.

KIEFNER, J. F.; VIETH, P. H. A modified criterion for evaluating the remaining strength of corroded pipe. 12 1989. Citado na página 22.

KISHAWY, H. A.; GABBAR, H. A. *Review of pipeline integrity management practices*. 2010. Citado na página 20.

KUMAR, S.; KARUPPANAN, S.; OVINIS, M. Artificial neural network-based failure pressure prediction of api 5l x80 pipeline with circumferentially aligned interacting corrosion defects subjected to combined loadings. *Materials*, v. 15, p. 2259, 03 2022. Citado na página 28.

LIU, C.-L. A tutorial of the wavelet transform. 01 2010. Citado na página 63.

LIU, S. et al. Quantitative identification of pipeline crack based on bp neural network. *Key Engineering Materials*, v. 737, p. 477–480, 06 2017. Citado 2 vezes nas páginas 20 e 23.

LOGAN, D. L. *A First Course in the Finite Element Method*. 6th. ed. [S.l.]: CL Engineering; Cengage Learning, 2016. ISBN 1305635116,9781305635111. Citado na página 46.

LOWE, A.; EREN, H. Corrosion signal processing using wavelet analysis and nyquist diagrams. In: . [S.l.: s.n.], 2002. v. 1, p. 855 – 860 vol.1. ISBN 0-7803-7218-2. Citado na página 20.

MALLAT, S. G. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 11, n. 7, p. 674–693, July 1989. ISSN 1939-3539. Citado na página 120.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, v. 5, n. 4, p. 115–133, Dec 1943. ISSN 1522-9602. Disponível em: <https://doi.org/10.1007/BF02478259>. Citado na página 75.

MICHAEL A., N. *Neural Networks and Deep Learning*. Determination Press, 2015. Disponível em: <http://neuralnetworksanddeeplearning.com>. Citado na página 133.

MITCHELL, T. M. *Machine Learning*. New York: McGraw-Hill, 1997. ISBN 978-0-07-042807-2. Citado na página 74.

MORLET, J. et al. Wave propagation and sampling theory–part ii: Sampling theory and complex waves. *Geophysics*, v. 47, p. 222–236, 02 1982. Citado na página 63.

MORLET, J. et al. Wave propagation and sampling theory?part i: Complex signal and scattering in multilayered media. *Geophysics*, v. 47, n. 2, p. 203–221, 1982. Disponível em: <https://doi.org/10.1190/1.1441328>. Citado na página 63.

MOTTA, R. et al. Comparative studies for failure pressure prediction of corroded pipelines. *Engineering Failure Analysis*, v. 81, 07 2017. Citado na página 27.

MOTTA, R. de S. et al. Reliability analysis of ovalized deep-water pipelines with corrosion defects. *Marine Structures*, v. 77, p. 102969, 2021. ISSN 0951-8339. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0951833921000319>. Citado na página 42.

MOTTA, R. S. et al. Comparative studies for failure pressure prediction of corroded pipelines. *Engineering Failure Analysis*, v. 81, p. 178–192, nov 2017. ISSN 13506307. Citado 3 vezes nas páginas 24, 53 e 54.

MSC.PATRAN. *User's Guide Reference Manual*. http://www.mscsoftware.com, 2015. Citado na página 33.

NYQUIST, H. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, v. 47, n. 2, p. 617–644, 1928. Citado na página 123.

PADMEC. *Infraestrutura*. Disponível em: <https://zumbi.padmec.org/sitepadmec/pt/sobre/infraestrutura/>. Citado na página 49.

PARZEN, E. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 33, n. 3, p. 1065 – 1076, 1962. Disponível em: <https://doi.org/10.1214/aoms/1177704472>. Citado na página 42.

PATIL, S. Estimation of burst pressure of corroded pipeline using finite element analysis (fea). In: *Advanced Materials Conference*. [S.l.: s.n.], 2012. Citado na página 24.

PIMENTEL, J. T. et al. New procedure of automatic modeling of pipelines with realistic shaped corrosion defects. *Engineering Structures*, v. 221, p. 111030, 2020. ISSN 0141-0296. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0141029619353805>. Citado 5 vezes nas páginas 12, 24, 54, 89 e 90.

PIPELINE Exploration. <https://geo-matching.com/content/pipeline-exploration>. Accessed: 2022-07-18. Citado na página 21.

PYTHON. *Tutorial and Library Reference Manual*. http://www.python.org/doc/, 2011. Citado na página 17.

RAMBERG, W.; OSGOOD, W. R. Description of stress-strain curves by three parameters. In: . [S.l.: s.n.], 1943. Citado na página 54.

ROSENBLATT, F. F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, v. 65 6, p. 386–408, 1958. Citado 2 vezes nas páginas 75 e 76.

RUDER, S. *An overview of gradient descent optimization algorithms*. https://ruder.io/: [s.n.], 2016. Citado 2 vezes nas páginas 80 e 81.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, v. 323, n. 6088, p. 533–536, Oct 1986. ISSN 1476-4687. Disponível em: <https://doi.org/10.1038/323533a0>. Citado na página 133.

SALOMON, D. *Data Compression: The Complete Reference*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 1846286026. Citado na página 120.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, v. 3, n. 3, p. 210–229, July 1959. ISSN 0018-8646. Citado 2 vezes nas páginas 62 e 74.

SEGHIER, M. et al. Structural Reliability Analysis of Corroded Pipeline made in X60 Steel Based on M5 Model Tree Algorithm and Monte Carlo Simulation. In: *Procedia Structural Integrity*. [S.l.: s.n.], 2018. ISSN 24523216. Citado na página 25.

SILVA, M.; GOUYON, R.; LEPOUTRE, F. Hidden corrosion detection in aircraft aluminum structures using laser ultrasonics and wavelet transform signal analysis. *Ultrasonics*, v. 41, n. 4, p. 301 – 305, 2003. ISSN 0041-624X. Citado na página 23.

SILVA, R.; GUERREIRO, J.; LOULA, A. A study of pipe interacting corrosion defects using the fem and neural networks. *Advances in Engineering Software*, v. 38, n. 11, p. 868 – 875, 2007. ISSN 0965-9978. Engineering Computational Technology. Citado 2 vezes nas páginas 20 e 26.

SIMO, J.; MIEHE, C. Associative coupled thermoplasticity at finite strains: Formulation, numerical analysis and implementation. *Computer Methods in Applied Mechanics and Engineering*, v. 98, n. 1, p. 41 – 104, 1992. ISSN 0045-7825. Citado na página 50.

SONG, Q. et al. Pipe defect detection with remote magnetic inspection and wavelet analysis. *Wireless Personal Communications*, v. 95, 03 2017. Citado 2 vezes nas páginas 23 e 24.

SOUZA, A. H. T. *Ferramentas Computacionais para Análise de Dutos com Corrosão.* Dissertação (Master Dissertation) — Universidade Federal de Pernambuco - UFPE, Recife, PE, 2008. Citado na página 24.

SOUZA, R. et al. Part 4: Rupture tests of pipeline segments containing long real corrosion defects. *Experimental Techniques*, v. 31, p. 46 – 51, 01 2007. Citado na página 82.

SOUZA, R. D. *Avaliação Estrutural de Dutos com Defeitos de Corrosão Reais.* Dissertação (Master Dissertation) — PUC-Rio, Rio de Janeiro, RJ, 2003. Citado 2 vezes nas páginas 24 e 82.

SOUZA, R. D. et al. Burst tests on pipeline containing long real corrosion defects. In: *International Pipeline Conference*. [S.l.: s.n.], 2004. Citado 3 vezes nas páginas 24, 82 e 86.

SOUZA, R. D. et al. PART 4: RUPTURE TESTS OF PIPELINE SEGMENTS CONTAINING LONG REAL CORROSION DEFECTS FEATURES A series on Applications of Experimental Techniques in the Field of Pipeline Integrity. Citado na página 24.

SOUZA, R. D. et al. Burst tests of corroded pipe segments removed from service. In: *Rio Pipeline Conference and Exposition*. [S.l.: s.n.], 2005. Citado 4 vezes nas páginas 24, 54, 82 e 86.

SUPRIYATMAN, D. et al. Artificial neural networks for corrosion rate prediction in gas pipelines. v. 2, 01 2012. Citado na página 26.

TEAM, G. B. *TensorFlow tutorials*. https://www.tensorflow.org/tutorials, 2019. Citado na página 78.

VIRTANEN, P. et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, v. 17, p. 261–272, 2020. Citado na página 42.

WANG, X. et al. Determination of corrosion type by wavelet-based fractal dimension from electrochemical noise. *International Journal of Electrochemical Science*, v. 8, p. 7211–7222, 05 2013. Citado na página 20.

WRIGHT, S. Correlation and causation. *Journal of Agricultural Research*, p. 557–558, 1921. Citado na página 104.

XU, W. et al. Corroded pipeline failure analysis using artificial neural network scheme. *Advances in Engineering Software*, v. 112, p. 255–266, 05 2017. Citado 2 vezes nas páginas 20 e 27.

APÊNDICE A – TRANSFORMADA WAVELET DISCRETA

Supondo que a função de escala e a função *wavelet* são dadas como bases conhecidas, podemos aproximar um sinal discreto $f[t]$ em $l^2(\mathbb{R})$[1] como mostrado em Equação (A.1) (SALOMON, 2006).

$$f[t] = \sum_{k=-\infty}^{\infty} W_\phi[k]\phi[t-k] + \sum_{k=-\infty}^{\infty}\sum_{j=0}^{\infty} W_\Psi[j,k]\Psi[2^j t - k] \tag{A.1}$$

onde $\phi[t]$ e $\Psi[t]$ são funções *wavelet* discretas e são definidas pela equação da análise multiresolução como mostrado em Equações (5.14) e (5.15). Os parâmetros $j \in \mathbb{N}$ e $k \in \mathbb{N}$ referem-se a funções de escala e translação e W_ϕ e W_Ψ são os coeficientes que precisam ser calculados. Como estas funções formam uma base ortornormal podemos encontrar estes coeficientes através do produto interno, conforme apresentado nas Equações (A.2) e (A.3)

$$W_\phi[k] = < f[n], \phi_k[n] > = \sum_n f[n]\phi_k[n] \tag{A.2}$$

$$W_\Psi[j,k] = < f[n], \Psi_{j,k}[n]) > = \sum_n f[n]\Psi_{j,k}[n] \tag{A.3}$$

No ano de 1989 Mallat em (MALLAT, 1989) propôs um algoritmo para a computação mais eficiente da transformada *wavelet* discreta. A ideia é encontrar uma maneira de achar os coeficientes sem conhecer a versão multiplicada por um fator de escala e dilatada das funções de escala e *wavelet*, levando a um tempo computacional menor para a realização do processo. A partir da Equação (5.8) nós temos a Equação (A.4) a seguir.

$$\phi_{j,k}[n] = \ 2^{j/2}\phi[2^j n - k] \ = \sum_n h_\phi(n')\sqrt{2}\phi[2(2^j n - k) - n'] \tag{A.4}$$

fazendo $n' = m - 2k$ nós encontramos a Equação (A.5).

$$\phi_{j,k}[n] = = \sum_m h_\phi(m-2k)\sqrt{2}\phi[2^{j+1}n - m] \tag{A.5}$$

[1] $l^2(\mathbb{R}) = \left\{ f[x] \in \mathbb{R} | \sum_{n=-\infty}^{\infty} |f[x]|^2 < \infty \right\}$

Combinando a Equação (A.2) com a Equação (A.5), teremos:

$$\begin{aligned} W_\phi[j,k] &= \sum_n f[n]\phi_{j,k}[n] \\ &= \sum_n f[n]2^{j/2}\phi[2^j n - k] \\ &= \sum_n f[n]2^{j/2}\sum_m h_\phi[m-2k]\sqrt{2}\phi[2^{j+1}n - m] \\ &= \sum_m h_\phi[m-2k]\left(\sum_n f[n]2^{(j+1)/2}\phi[2^{j+1}n-m]\right) \\ &= \sum_m h_\phi[m-2k]W_\phi[j+1,m] \\ &= h_\phi[-n] \circledast W_\phi[j+1,n]\Big|_{n=2k,k\leq 0} \end{aligned} \tag{A.6}$$

De maneira similar podemos encontrar os coeficientes de detalhamento como mostrado na Equação (A.7).

$$W_\Psi[j,k] = h_\Psi[-n] \circledast W_\Psi[j+1,n]\Big|_{n=2k,k\leq 0} \tag{A.7}$$

O processo de transformada wavelet discreta (TWD) é inicialmente realizado através da convolução do sinal com a resposta ao impulso do banco de filtros, onde $h_\phi[k]$ é o filtro passa-baixa e $h_\Psi[k]$ é o filtro passa-alta. Equações (A.8) e (A.9) mostram o processo matematicamente.

$$W_\phi[n] = (h_\phi \circledast f)[n] = \sum_k h_\phi[k]f[n-k] \tag{A.8}$$

$$W_\Psi[n] = (h_\Psi \circledast f)[n] = \sum_k h_\Psi[k]f[n-k] \tag{A.9}$$

O $\left\{W_\phi[n]\right\}_{n\in\mathbb{Z}}$ são também conhecidos como coeficientes de aproximação e $\left\{W_\Psi[n]\right\}_{n\in\mathbb{Z}}$ são chamados de coeficientes de detalhamento.

O processo de reconstrução do sinal na transformada *wavelet* é realizado de maneira similar, calculando a integral convolução dos coeficientes pelos filtros inversos.

A solução da integral de convolução pode ser computacionalmente caro com o aumento do número de pontos discretos, desta forma, para aumentar ainda mais a eficiência do processo pode-se utilizar o teorema da convolução.

Definição 2. *A transformada de Fourier de duas funções convoluídas no domínio do espaço é igual ao produto das transformadas das duas funções no domínio do tempo.*

Esta definição equivale matematicamente ao apresentado na equação (A.10) a seguir:

$$\mathcal{F}(f \circledast g) = \mathcal{F}(f)\cdot\mathcal{F}(g) \tag{A.10}$$

se aplicarmos a transformada inversa temos a Equação (A.11).

$$f \circledast g = \mathcal{F}^{-1}(\mathcal{F}(f) \cdot \mathcal{F}(g)) \tag{A.11}$$

onde nas Equações (A.10) e (A.11), $\mathcal{F}$ é o operador da transformada de Fourier e consequentemente $\mathcal{F}(f)$ e $\mathcal{F}(g)$ são as transformadas das funções f e g. O operador $\cdot$ representa a multiplicação elemento por elemento. As equações para o cálculo da transformada de Fourier discreta e da transformada inversa são apresentas nas Equações (A.12) e (A.13).

$$\mathcal{F}(f[n]) = F[k] = \sum_{n=0}^{N-1} e^{-2\pi \mathrm{i} \frac{kn}{N}} f[n] \tag{A.12}$$

$$\mathcal{F}^{-}1(F[k]) = f[n] = \frac{1}{N} \sum_{n=0}^{N-1} e^{2\pi \mathrm{i} \frac{kn}{N}} F[k] \tag{A.13}$$

onde i representa a unidade imaginária que possui a propriedade $\mathrm{i}^2 = -1$. O processo direto de aplicar a transformada tem uma complexidade computacional igual a $\mathcal{O}(N^2)$ onde N é o número de pontos discretos. No ano de 1965 James Cooley e John Tukey, no artigo (COOLEY; TUKEY, 1965), propuseram um algoritmo conhecido como transformada rápida de Fourier. Este algoritmo se aproveita de algumas simetrias contidas na transformada para diminuir a quantidade de cálculos realizados, isto reduz o custo da transformada para $\mathcal{O}(N \log N)$.

Existem muitas famílias de wavelets, com diferentes funções de *wavelets* e de escala e subcategorias diferentes devido ao número de momentos de fuga e ao número de pontos (ou comprimento da wavelet).

O momento de fuga é um critério sobre como uma função decai em direção ao infinito. A função *wavelet* $\phi(t)$ tem p momentos de fuga se a expressão na Equação (A.14) for verdadeira.

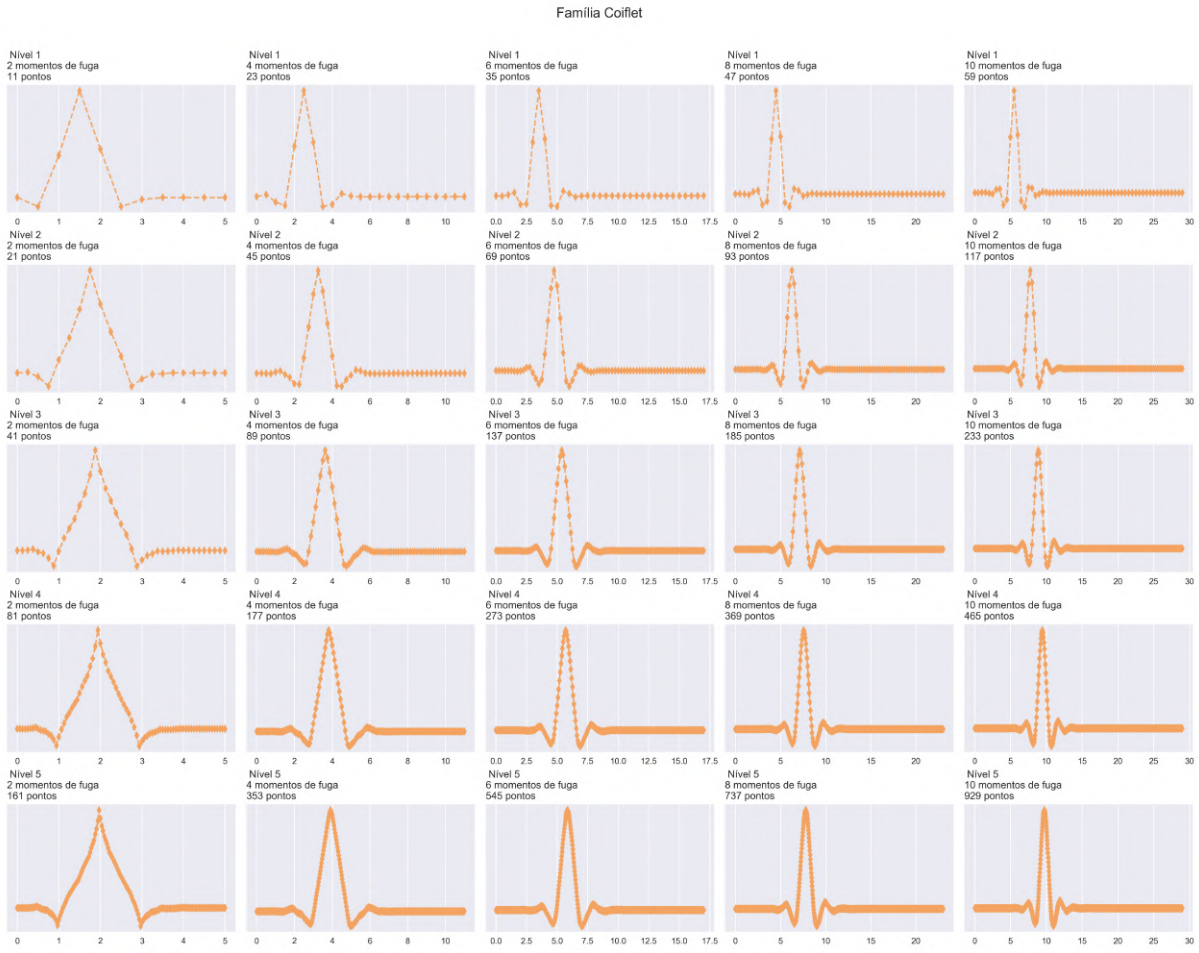
$$\int_{-\infty}^{\infty} t^k \psi(t) dt = 0 \quad for \quad 0 \leq k < p \tag{A.14}$$

Os momentos de fuga indicam a ordem e a suavidade de uma *wavelet*; se uma wavelet tiver p momentos de fuga, esta poderá aproximar polinômios de grau $p-1$. A Figura 71 apresenta as funções de escala da família Coiflet através de 5 níveis de decomposição e variando o número de momentos de fuga que consequentemente altera o número de pontos discretos que compõe a *wavelet*.

A Figura 72 apresenta as funções de decomposição da família *reverse biorthogonal* também em 5 níveis de decomposição e variando o número de pontos discretos da *wavelet*.

Nas Figuras 73 e 74 são apresentados os coeficientes dos filtros passa-baixa e passa-alta de decomposição e reconstrução, para a *wavelet* da família Daubechies com 2 momentos de e da família Coiflet com 1 momento de fuga.

Figura 71 – Família Coiflet.



Fonte: O autor (2021).

Apoiando-se no teorema 3 da amostragem de Nyquist-Shannon, datado de 1928 no artigo (NYQUIST, 1928), em cada estágio subsequente (nível) no processo de decomposição por TDW, o número de amostras é reduzido por um fator de 2, este processo é chamado de *downsampling* e é representado simbolicamente por $\downarrow$. De forma prática o teorema afirma que um sinal pode ser completamente definido se o número de amostras discretas daquele sinal, possuir no mínimo 2 vezes o valor da maior frequência f_m contida naquele sinal. Por exemplo para representar discretamente uma onda senoidal com frequência de $60Hz$ precisamos de no mínimo 120 pontos discretos a cada segundo. Esse limite de $2f_m$ é conhecido como Nyquist rate.

Definição 3. *Seja um sinal, limitado em banda, e seu intervalo de tempo dividido em partes iguais, de forma que se obtenham intervalos tais que, cada subdivisão compreenda um intervalo com período T segundos, onde T é menor do que $f_m/2$, e se uma amostra instantânea é tomada arbitrariamente de cada subintervalo, então o conhecimento da amplitude instantânea de cada amostra somado ao conhecimento dos instantes em que é*

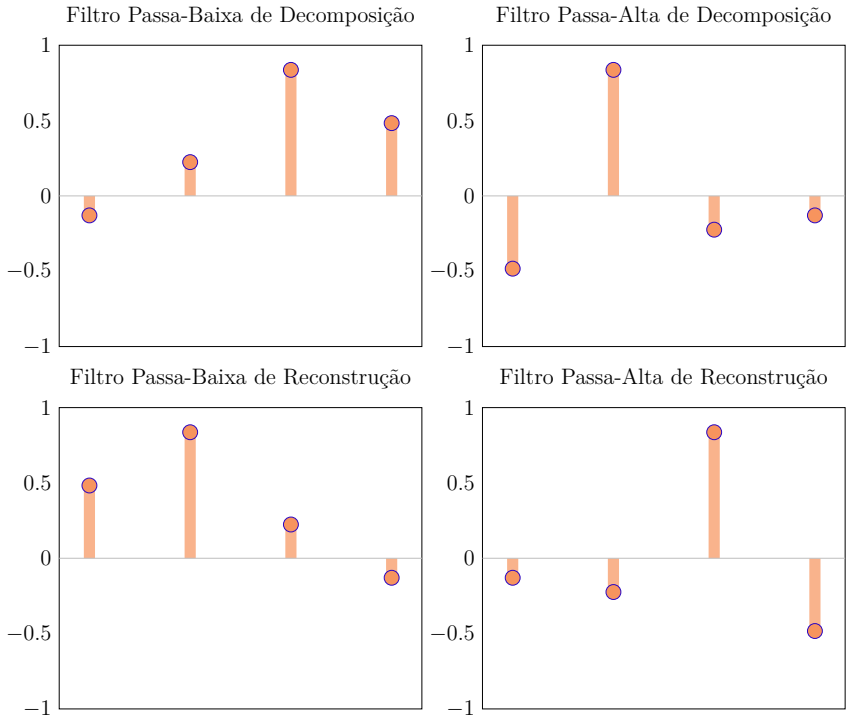
Figura 72 – Família Reverse Biorthogonal.

Fonte: O autor (2021).

tomada a amostra de cada subintervalo contém toda a informação do sinal original.

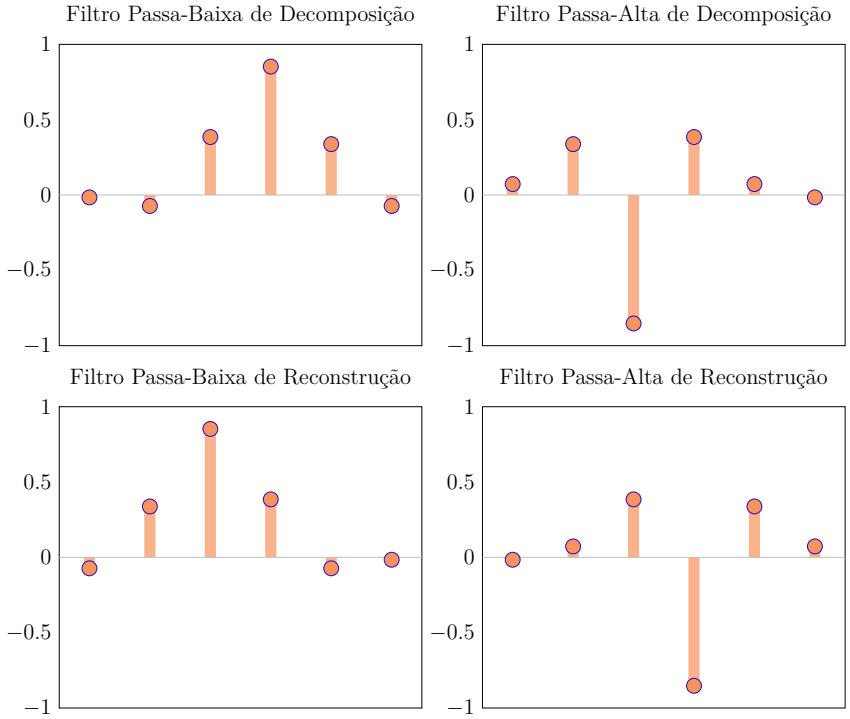
Um diagrama esquemático do procedimento de aplicação da decomposição via TWD pode ser observado na Figura 75. A redução nos dados pode ser realizada diversas vezes até o nível máximo que depende do tamanho dos dados e do filtro.

Figura 73 – Coeficientes de decomposição e reconstrução para Daubechies com 2 momentos.



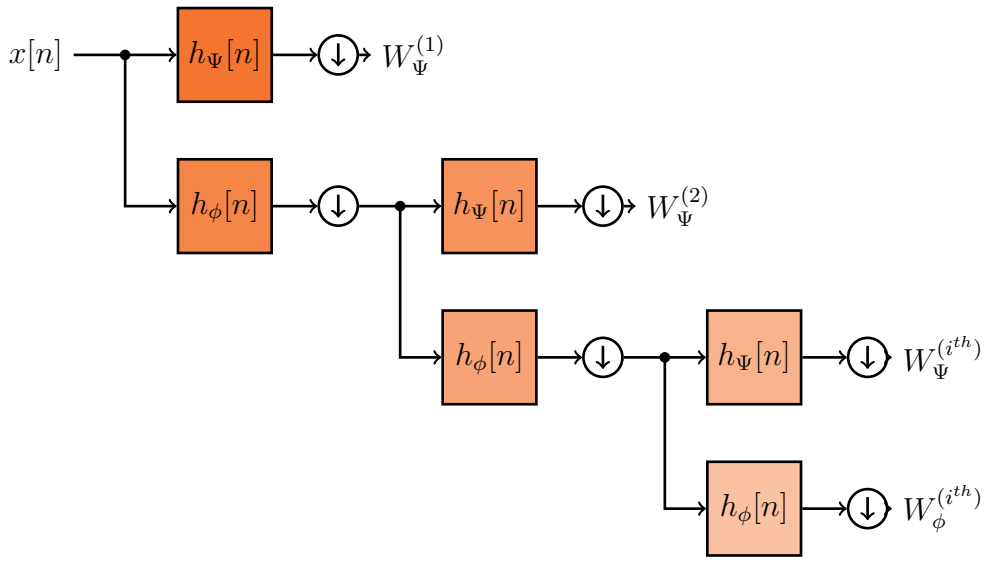
Fonte: O autor (2021).

Figura 74 – Coeficientes de decomposição e reconstrução para Coiflet com 1 momento.



Fonte: O autor (2021).

Figura 75 – Processo de decomposição via TWD.



Fonte: O autor (2021).

APÊNDICE B – REDES NEURAIS ARTIFICIAIS

Como inicialmente explicitado na seção 6.2, redes neurais funcionam através do agrupamento de neurônios artificiais em série e paralelo. Dentro desta rede, é realizado o cálculo da ativação de cada neurônio, baseando-se nas entradas recebidas por este. Na mesma seção 6.2 foi apresentada a Equação 6.1 que apresenta as funções básicas para a ativação do neurônio artificial conhecido como *perceptron*.

Para simplificar a Equação (6.1) podemos reescrever $\sum_j w_j x_j$ como um produto escalar, pois $w \cdot x \equiv \sum_j w_j x_j$. Implementando a notação indicada e passando o valor de b para o outro lado da inequação, temos a Equação (B.1).

$$saída = \begin{cases} 0 & \text{se } w \cdot x + b \leq 0 \\ 1 & \text{se } w \cdot x + b > 0 \end{cases} \tag{B.1}$$

A solução das redes neurais consiste em procurar os valores de w e b que a partir dos dados de entrada aproximem as respostas desejadas. Desta forma as redes neurais podem se adaptar e aprender soluções para vários problemas diferentes.

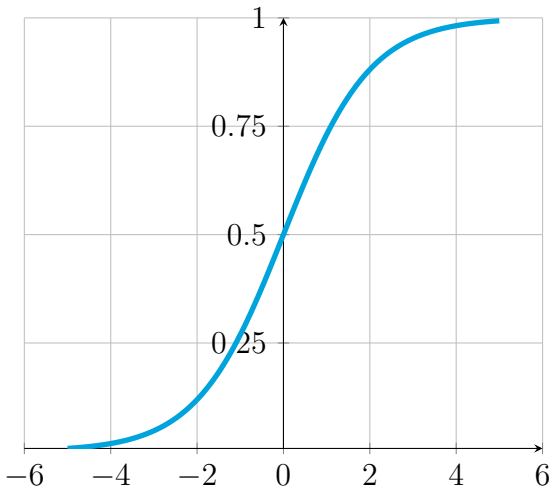
Da forma apresentada até o momento, a rede neural pode inverter a solução abruptamente durante a busca pelos pesos e *bias* de alguma solução, pois esta possui apenas uma saída binária, 0 ou 1. Este fato pode fazer a rede neural alterar completamente seus resultados enquanto tenta encontrar a solução para um caso específico. Para solucionar este problema, funções de ativação são adicionadas aos neurônios além da possibilidade das entradas receberem não só valores binários, mas valores contidos no conjunto de entradas chamado de **E** (B.2). Desta maneira, dependendo da função de ativação, a saída pode pertencer a um conjunto maior de valores que podem estar entre 0 e 1 ou entre -1 e 1, além de que, pequenas alterações nos pesos ou *bias* da rede afetam menos os resultados já encontrados e facilitam sua convergência.

$$E = \{x \in \mathbb{R} | 0 \leq x \leq 1\} \tag{B.2}$$

Para entender o novo equacionamento do sistema vamos exemplificar utilizando a função de ativação logística apresentada na Equação (B.4). A sua forma pode ser observada na Figura 76

$$f(z) = \frac{1}{1 + \exp^{-z}} \tag{B.3}$$

Figura 76 – Representação gráfica da função logística.



Fonte: O autor (2021).

O parâmetro de entrada z para função da Equação (B.4) será $w \cdot x + b$, sendo o resultado de saída para o neurônio agora resultante de $f(w \cdot x + b)$. Apresentando de forma mais explicita temos a nova saída do neurônio representado pela Equação

$$saída = \frac{1}{1 + \exp^{(-w \cdot x - b)}} \tag{B.4}$$

A tabela 15 apresenta equações para as funções de ativação mais comuns na aplicação de redes neurais artificiais.

No funcionamento apresentado das redes neurais, a saída de uma camada de neurônios é utilizada para alimentar a próxima camada. Este tipo de rede neural é chamada de *feedforward*, do inglês. Isto significa que não há *loops* na alimentação dos dados evitando uma solução explícita onde a entrada dependeria da saída. Existe outro tipo de rede neural chamada de *recurrent neural network* onde o funcionamento desta se baseia na ativação temporária dos neurônios, que podem ativar outros neurônios em cadeia. Neste tipo de rede *loops* não causam problemas.

Podemos dizer que a rede neural aprendeu a solução de determinado problema com a definição dos valores de pesos e *biases* de cada neurônio. Assim é necessário definir um algoritmo que para determinadas entradas, x, encontre os parâmetros w e b e aproxime a solução dada.

Adotando x como o vetor de variáveis de entrada, $y(x)$ o vetor contendo as saídas desejadas e h_x como o vetor de saída da rede neural dada às entradas x. Definimos a função de custo ou objetivo, que é conhecida como *Mean Squared Error (MSE)* ou Erro Quadrático Médio em tradução livre. No MSE cada erro parcial equivale à área do quadrado criado

Tabela 15 – Funções de Ativação

Nome	Equação	Derivada
Linear	$f(z) = z$	$f'(z) = 1$
Step Binary	$f(z) = \begin{cases} 0 & \text{se } z < 0 \\ 1 & \text{se } z \geq 0 \end{cases}$	$f'(z) = \begin{cases} 0 & \text{se } z \neq 0 \\ \nexists & \text{se } z = 0 \end{cases}$
Logistic	$f(z) = \frac{1}{1+\exp^{-z}}$	$f'(z) = f(z)(1 - f(z))$
Tanh	$f(z) = \tanh(z) = \frac{2}{1+\exp^{-2z}} - 1$	$f'(z) = 1 - f(z)^2$
ArcTan	$f(z) = \tan^- 1(z)$	$f'(z) = \frac{1}{z^2+1}$
Rectified Linear Unit (ReLU)	$f(z) = \begin{cases} 0 & \text{se } z < 0 \\ z & \text{se } z \geq 0 \end{cases}$	$f'(z) = \begin{cases} 0 & \text{se } z < 0 \\ 1 & \text{se } z \geq 0 \end{cases}$
Parametric ReLU (PReLU)	$f(z) = \begin{cases} \alpha z & \text{se } z < 0 \\ z & \text{se } z \geq 0 \end{cases}$	$f'(z) = \begin{cases} f(z) + \alpha & \text{se } z < 0 \\ 1 & \text{se } z \geq 0 \end{cases}$
Exponential Linear Unit (ELU)	$f(z) = \begin{cases} \alpha(\exp^z - 1) & \text{se } z < 0 \\ z & \text{se } z \geq 0 \end{cases}$	$f'(z) = \begin{cases} f(z) + \alpha & \text{se } z < 0 \\ 1 & \text{se } z \geq 0 \end{cases}$
Softplus	$f(z) = \ln(1 + \exp^z)$	$f'(z) = \frac{1}{1+\exp^{-z}}$

a partir da distância geométrica entre os pontos medidos. Podemos visualizar a medida destes erros na Figura 77 para um exemplo de uma dada função. O valor do MSE é computado através do cálculo da média do somatório das áreas preenchidas na cor laranja e definidas como ε_1, ε_2 e ε_3.

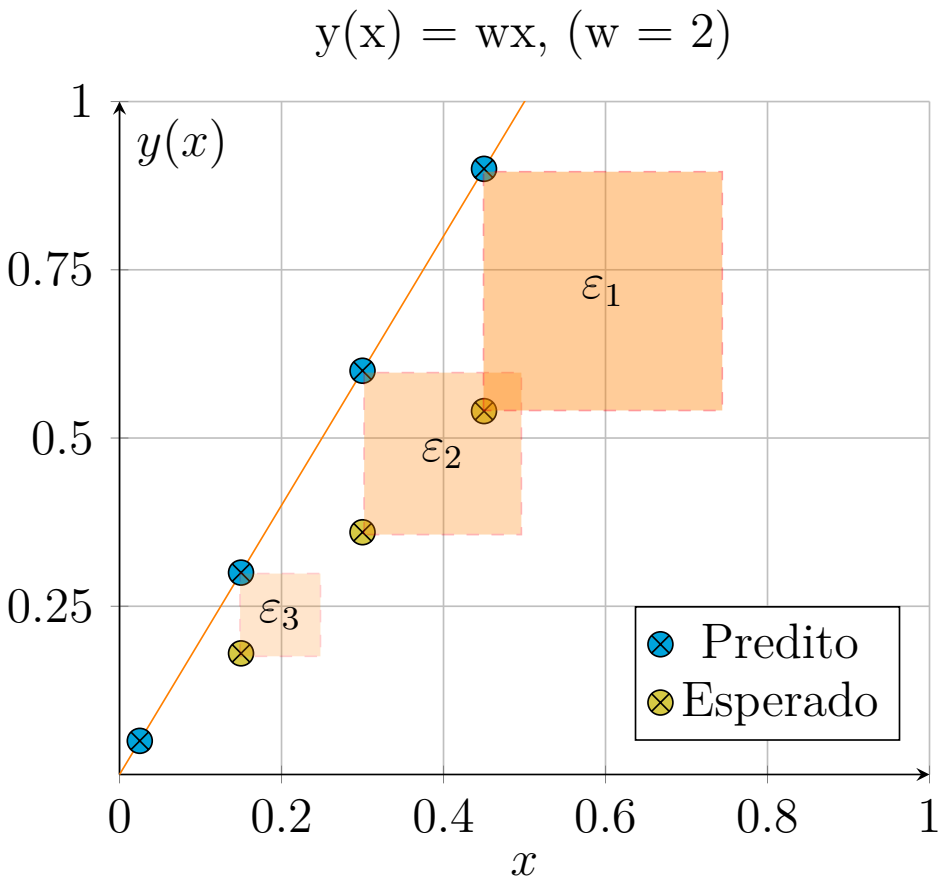
A fórmula que representa o Erro Quadrático Médio pode ser visualizada na Equação (B.5).

$$J(w, b) = \frac{1}{2n} \sum_x \|y(x) - h_x\|^2 \tag{B.5}$$

Onde w representa todos os pesos na rede, b todos os desvios e n é o número total de entradas para treinamento. h_x, como já explicitado, é o vetor de saídas da rede e que depende tanto das entradas x quanto de w e b. A forma desta função objetivo pode ser observada na Figura 78.

A partir desta nova função definida, a solução se torna um problema de otimização, onde é necessário minimizar o erro médio quadrado entre a saída predita pela rede neural e a saída desejada, já informada na entrada do sistema. Para solucionar este problema de otimização, o método utilizado é o Gradiente Descendente. Este método utiliza a derivada

Figura 77 – Representação gráfica do Erro Quadrático Médio.



Fonte: O autor (2021).

da função objetivo, para atualizar os dados no sentido contrário ao gradiente da função. A atualização dos pesos w e vieses b a cada passo da otimização é dado como descrito nas Equações(B.6) e (B.7).

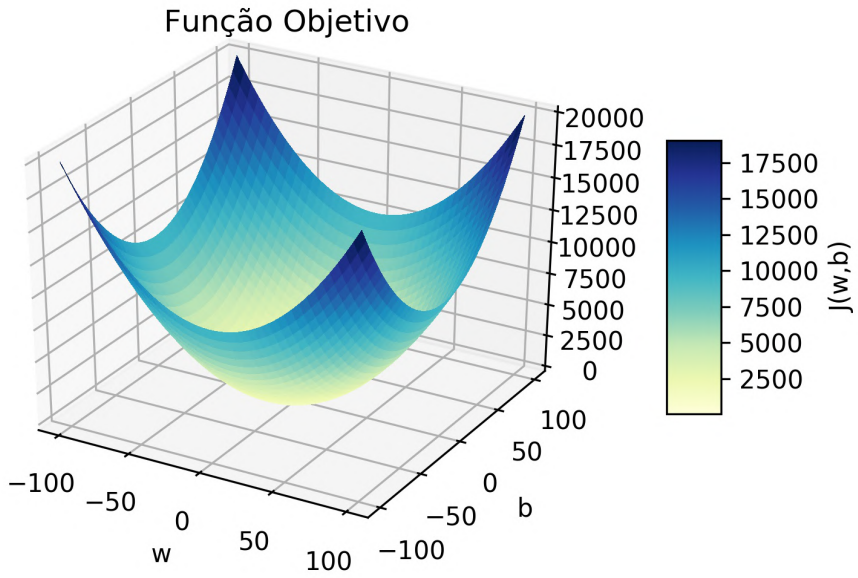
$$w = w - \eta \nabla J(w) \tag{B.6}$$

$$b = b - \eta \nabla J(b) \tag{B.7}$$

Onde η é conhecido como taxa de aprendizado e $\nabla J(w)$ e $\nabla J(b)$ são os gradientes da função de custo em relação à w e b respectivamente.

Os algoritmos de otimização Gradiente Descendente podem ser divididos em três variantes que no inglês são chamados *Batch Gradient Descent*, *Stochastic Gradient Descent (SGD)* e *Mini-Batch Gradient Descent*. Uma explicação sobre os três métodos é apresentada a seguir. Por conveniência vamos substituir as variáveis a serem encontradas, que são os pesos w e vieses b, por $\theta \in \mathbb{R}^d$, que é a variável comumente utilizada na área e assumir que esta engloba as duas anteriores.

Figura 78 – Forma da função objetivo.



Fonte: O autor (2021).

Na variante *Batch Gradient Descent*, todos os dados de treinamento são considerados para cada passo. A média dos gradientes de todos os exemplos é calculada e este valor é utilizado para fazer a atualização dos parâmetros. Este método é muito bom quando aplicado em funções convexas ou suaves. A representação matemática deste método é apresentada na Equação B.8.

$$\theta = \theta - \eta \nabla_\theta J(\theta) = \theta - \frac{\eta}{n} \sum_{i=1}^{n} \nabla J_i(\theta) \tag{B.8}$$

O *Stochastic Gradient Descent (SGD)* tenta corrigir o problema do método anterior. Pois se tivemos uma rede neural que tenha muitos dados o custo de calcular todos os gradientes é muito alto e o método se torna ineficiente. O procedimento para realizar este método é o seguinte.

1. Seleciona-se um exemplo
2. Alimenta-o na rede neural
3. Calcula-se o gradiente
4. Utiliza-se o gradiente calculado no passo anterior para atualizar os pesos
5. Repete-se os passos de 1-4 para todos os exemplos utilizados para treinamento.

Quando o procedimento é realizado para todos os exemplos diz-se que se completou uma época (*epoch*) de treinamento. Antes da rotina de repetição os dados são sorteados aleatoriamente. Como eles utiliza apenas um exemplo por vez à convergência é bastante ruidosa. Na Equação B.9 temos a representação matemática do cálculo de atualização.

$$\theta = \theta - \eta \nabla_\theta J(\theta; \dot{x}^{(i)}; y^{(i)}) \tag{B.9}$$

Comparando os dois métodos temos que o SGD pode ser usado para conjuntos de dados maiores. Ele converge mais rapidamente quando o conjunto de dados é grande, pois causa atualizações nos parâmetros com mais frequência. O *Batch Gradient Descent* pode ser usado para curvas mais suaves. O SGD pode ser usado quando o conjunto de dados é grande. O *Batch Gradient Descent* converge diretamente para os mínimos enquanto o SGD converge mais rapidamente para conjuntos de dados maiores. Porém, como no SGD usa-se apenas um exemplo de cada vez, não é possível implementar uma forma vetorizada dele. Isso pode tornar os cálculos mais lentos. Para resolver esse problema, é utilizada uma mistura dos dois métodos que é o *Mini-Batch Gradient Descent*.

Mini-Batch utiliza m exemplos ao invés de um como no SGD. Neste método são criados lotes (*batches*) com um número fixo de exemplos de treinamento selecionados aleatoriamente, onde este número é menor que o valor total de exemplos. Estes exemplos são chamados de *mini-batch* ou mini-lotes em português brasileiro e serão chamados de $X_1, X_2 ... X_m$. Este método assume que a aproximação apresentada na Equação (B.10) é válida, i.e., a média do gradiente do mini-lote com tamanho m é aproximadamente igual à média do gradiente de toda a amostra com tamanho n.

$$\frac{\sum_{j=1}^{m} \nabla J_{X_j}}{m} \approx \frac{\sum_x \nabla J_x}{n} = \nabla J \tag{B.10}$$

Rearrumando a Equação (B.10) e passando o ∇J para a esquerda da equação temos a Equação (B.11):

$$\nabla J \approx \frac{\sum_{j=1}^{m} \nabla J_{X_j}}{m} \tag{B.11}$$

Os passos para aplicação deste método para cada época de treinamento são:

1. Seleciona-se um mini-lote
2. Alimenta-se o lote na rede neural
3. Calcula-se o gradiente médio do mini-lote
4. Utiliza-se a média do gradiente calculado no passo anterior para atualizar os pesos

5. Repete-se os passos de 1-4 para todos os mini-lotes criados .

A equação de atualização do *Mini-Batch Gradient Descent* está expressa na Equação B.12.

$$\theta = \theta - \eta \nabla_\theta J(\theta; x^{(i:\dot{i+n})}; y^{(i:i+n)}) \tag{B.12}$$

B.1 ALGORITMO RETRO-PROPAGAÇÃO

Embora o número de avaliações do gradiente tenha diminuído como o emprego do mini-lote Gradiente Descendente, o cálculo ainda é computacionalmente caro no contexto de redes neurais. No ano de 1986, Rumelhart et al (RUMELHART; HINTON; WILLIAMS, 1986) criaram o algoritmo conhecido como *back-propagation* ou retro-propagação em português brasileiro. Este é o responsável por ajustar os pesos (w) e os vieses (b) da rede minimizando a função objetivo de maneira mais eficiente. A grande visibilidade e popularidade do método vem, deste fornecer um jeito rápido de calcular o gradiente da função de custo, fazendo com que a rede neural aprenda mais rápido do que com outros métodos. Este algoritmo é considerado um dos mais importantes na história de redes neurais, pois ele viabilizou o treinamento das redes de aprendizagem profundas utilizadas nos dias atuais que podem ter milhões ou até bilhões de pontos.

A seguir será apresentado o algoritmo baseado no descrito em (MICHAEL A., 2015). Para entender o funcionamento do mesmo, será adotada uma notação para identificar a posição dos pesos w dentro da rede. A identificação será da forma w^l_{jk} onde este representa o peso do k^{th} neurônio na $(l-1)^{th}$ camada para o j^{th} neurônio na l^{th} camada. Figura 79 apresenta dois exemplos de identificação destes pesos em uma rede.

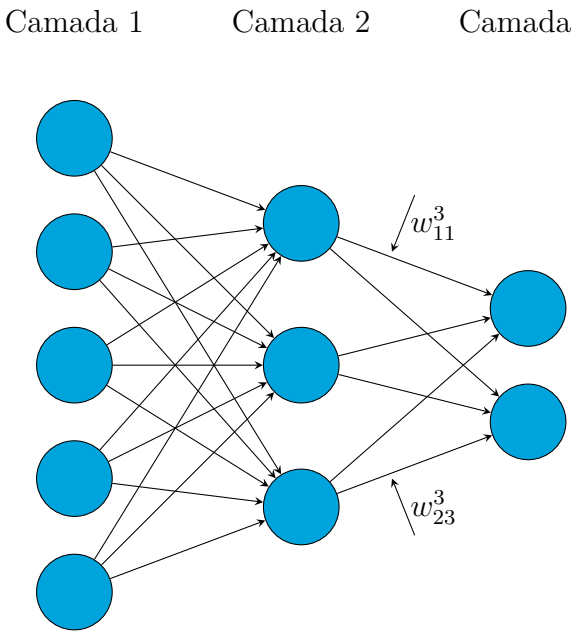
É utilizado uma notação similar para identificação dos vieses b e ativações a em uma rede neural. A notação é da forma b^l_j para os vieses e a^l_j para a ativação do j^{th} neurônio na l^{th} camada. A Figura 80 apresenta alguns exemplos da identificação destes parâmetros em uma rede neural.

Com estas notações definidas, podemos definir a ativação a^l_j do j^{th} neurônio da l^{th} camada, utilizando uma função de ativação $f(z)$, como as apresentadas anteriormente na Tabela 15, a partir das equações B.13 e B.14.

$$z^l_j = \sum_k w^l_{jk} a^{l-1}_k + b^l_j \tag{B.13}$$

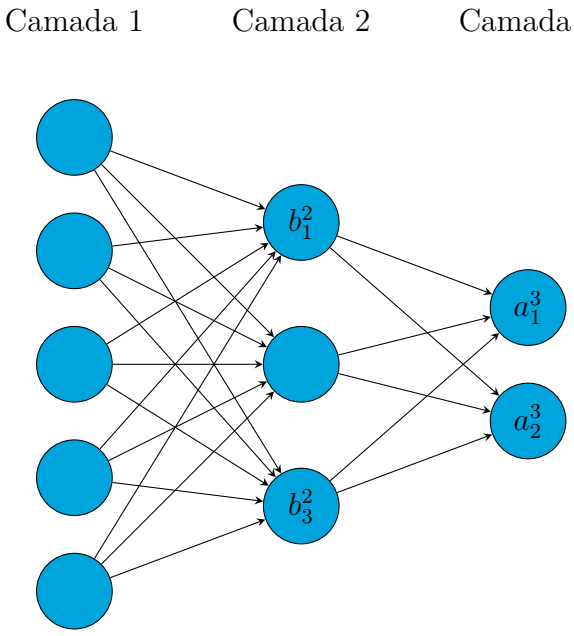
$$a^l_j = f(\sum_k w^l_{jk} a^{l-1}_k + b^l_j) \tag{B.14}$$

Figura 79 – Modelo de identificação dos pesos w em uma rede neural.



Fonte: O autor (2021).

Figura 80 – Modelo de identificação dos vieses b e ativações a em uma rede neural.



Fonte: O autor (2021).

Para simplificar a escrita da equação será considerada a vetorização da função $f(z)$, pois esta é aplicada em cada elemento individualmente. Desta forma considera-se a aplicação da função f em um vetor v, e utiliza-se a notação $f(v)$ como a aplicação da função em cada elemento do vetor v. Isso significa que $f(v)_j = f(v_j)$. Desta forma podemos simplificar a Equação B.14 para a forma apresentada na Equação B.15.

$$a^l = f(w^l a^{l-1} + b^l) \tag{B.15}$$

Esta forma simplifica o entendimento do valor da ativação dos neurônios em uma camada em relação ao valor da ativação da camada anterior. Podemos utilizar mais uma simplificação e adotar $z^l \equiv f(w^l a^{l-1} + b^l)$, então podemos escrever o cálculo da ativação em uma camada como $a^l = f(z^l)$. O cálculo da função objetivo adotando a^L como a saída da rede na última camada L temos a Equação B.16.

$$J = \frac{1}{2n} \sum_x \|y(x) - a^L(x)\|^2 \tag{B.16}$$

A partir deste ponto, para melhor entedimento do *back-propagation*, será adotado que a função de custo J é uma média sobre as funções de custo individuais J_x, conforme apresentado na Equação B.17.

$$J = \frac{1}{n} \sum_x J_x \tag{B.17}$$

Desta forma para um simples exemplos, teremos uma função de custo como na Equação B.18. Como estamos tratando apenas de um exemplo o x do J_x também será omitido.

$$J_x = J = \frac{1}{2}\|y - a^L)\|^2 \tag{B.18}$$

O algoritmo de retro-propragação se propõe a descobrir como os pesos e vieses alteram a função objetivo e minimizá-la. Uma das definições necessárias é a do cálculo do erro δ_j^l do j^{th} neurônio da l^{th} camada.

$$\delta_j^l = \frac{\partial J}{\partial z_j^l} \tag{B.19}$$

A partir desta definição o algoritmo dá um caminho para calcular δ^l e associar isso com as variáveis de interesse que são as derivadas $\frac{\partial J}{\partial w_{jk}^l}$ e $\frac{\partial J}{\partial b_j^l}$ que compõe o gradiente da função objetivo.

O algoritmo *back-propagation* se resume nas quatro equações que serão apresentadas a seguir. A primeira calcula o erro da camada de saída L conforme Equação B.20.

$$\delta_j^L = \frac{\partial J}{\partial a_j^L} f'(z_j^L) \tag{B.20}$$

O primeiro termo $\frac{\delta J}{\delta a_j^L}$ mede o quanto a função de custo varia em relação a função de ativação da saída e o segundo termo $f'(z_j^L)$ mede a variação da função de ativação (f) está mudando em z_j^L e que surge naturalmente a partir da regra da cadeia na derivada. O primeiro termo pode ser calculado facilmente para a função de custo MSE pela expressão dada na Equação B.21. As derivadas das funções de ativação já foram apresentadas na Tabela 15.

$$\frac{\partial J}{\partial a_j^L} = (a_j^L - y_j) \tag{B.21}$$

Adicionando o símbolo do produto de Hadamard $\odot$ que representa uma multiplicação elemento por elemento entre vetores ou matrizes, equivalente a $z_i = x_i \times y_i$, será definido o erro δ^L da última camada de forma vetorial. Reescrevendo a Equação B.20 temos a Equação B.22

$$\delta^L = \nabla_a J \odot \sigma'(z^L) \tag{B.22}$$

A segunda equação do algoritmo, e que define o termo retro-propagação, vem de encontrar o erro σ em uma camada l a partir do erro da camada $l+1$. Podemos encontrar a equação para este erro utilizando regra da cadeia como apresentado na Equação B.23.

$$\delta_j^l = \frac{\partial J}{\partial z_k^l} = \sum_k \frac{\partial J}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_k^l} = \frac{\partial z_k^{l+1}}{\partial z_k^l} \delta_k^{l+1} \tag{B.23}$$

O termo z_k^{l+1} pode ser escrito como Equação B.24

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} a_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1} f(z_j^l) + b_k^{l+1} \tag{B.24}$$

Diferenciando em relação à ∂z_k^l temos a Equação B.25

$$\frac{\partial z_k^{l+1}}{\partial z_k^l} = w_{kj}^{l+1} f'(z_j^l) \tag{B.25}$$

Inserindo a Equação B.25 na Equação B.23 e colocando na forma vetorial temos a segunda equação do algoritmo dado na Equação B.26.

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot f'(z^L) \tag{B.26}$$

A terceira equação B.27 mede a taxa de alteração da função de custo em relação aos desvios b da rede em determinada camada l.

$$\frac{\partial J}{\partial b_j^l} = \delta_j^l \tag{B.27}$$

Podemos encontrar essa equação de maneira similar a anterior (Equação B.23) utilizando regra da cadeia, conforme apresentado na Equação B.28.

$$\frac{\partial J}{\partial b_j^l} = \frac{\partial J}{\partial z_j^l}\frac{\partial z_j^l}{\partial b_j^l} = \delta_j^l \times 1 \tag{B.28}$$

E a quarta e última equação B.29 do *back-propagation* mede a taxa de alteração da função de custo novamente agora em relação aos pesos w da rede em determinada camada l.

$$\frac{\partial J}{\partial w_{jk}^l} = a_k^{l-1}\delta_j \tag{B.29}$$

De uma forma muito similar as já apresentadas como na Equação B.28, podemos demonstrar a Equação B.29 através da Equação B.30.

$$\frac{\partial J}{\partial w_{jk}^l} = \frac{\partial J}{\partial z_j^l}\frac{\partial z_j^l}{\partial w_{jk}^l} = \delta_j a_k^{l-1} \tag{B.30}$$

Assim, podemos concluir que o algoritmo *back-propagation* nos dá uma maneira rápida de calcular o gradiente da função de custo. O algoritmo 4 completo de retropropagação pode ser visualizado a seguir.

Algoritmo 4: Retro-propagação

Result: $\frac{\partial J}{\partial b_j^l}, \frac{\partial J}{\partial w_{jk}^l}$

$a^1 \leftarrow f(a^1)$;

for $l \leftarrow 2$ **to** L **do**

  $z^l \leftarrow w^l a^{l-1} + b^l$;

  $a^l \leftarrow \sigma(z^l)$;

end

$\delta^L \leftarrow \nabla_a J \odot \sigma'(z^L)$;

for $l \leftarrow L-1$ **to** 2 **do**

  $\delta^l \leftarrow ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^L)$;

end

$\frac{\partial J}{\partial b_j^l} \leftarrow \delta_j^l$;

$\frac{\partial J}{\partial w_{jk}^l} \leftarrow a_k^{l-1}\delta_j$;

Um resumo descritivo com os passos para a aplicação deste algoritmo é apresentado a seguir.

- **Entrada** θ: Define a ativação correspondente a^1 para a camada de entrada.
- **Feedformard**: Para cada $l = 2, 3, ...L$ calcula-se o $z^l = w^l a^{l-1} + b^l$ e $a^l = \sigma(z^l)$.
- **Erro na saída** δ^L: Calcular o vetor $\delta^L = \nabla_a J \odot \sigma'(z^L)$
- **Propagar de volta o erro**: Para cada $l = L-1, L-2, ..., 2$ calcular $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^L)$
- **Saída**: O gradiente da função de custo é dado por $\frac{\partial J}{\partial b^l_j} = \delta^l_j$ e $\frac{\partial J}{\partial w^l_{jk}} = a^{l-1}_k \delta_j$.

Analisando o algoritmo conclui-se que computamos os erros δ^l exatamente no sentido contrário da alimentação da rede neural, por isso é chamado de retro-propagação.

ANEXO A – EXEMPLO DE ARQUIVO DE MALHA PARA UM MODELO AXISSIMÉTRICO

```
TITRE
    caso_0
FINSF
COOR_3D
    N1 290.2400 551.6020 0.000000
    N2 293.8800 551.6020 0.000000
    N3 297.5200 551.6020 0.000000
    N4 301.1600 551.6020 0.000000
    N5 304.8000 551.6020 0.000000
    N6 297.5200 556.8476 0.000000
    N7 293.8800 556.9548 0.000000
    N8 301.1600 556.9838 0.000000
    N9 290.2400 557.2933 0.000000
    N10 304.8000 557.2933 0.000000
                .
                .
                .
     N1481 290.2400 -6.867333 0.000000
    N1482 293.7662 -6.867811 0.000000
    N1483 297.2841 -6.868298 0.000000
    N1484 300.8020 -6.868786 0.000000
    N1485 304.3283 -6.869264 0.000000
    N1486 290.2400 -3.433666 0.000000
    N1487 293.7423 -3.434037 0.000000
    N1488 297.2362 -3.434472 0.000000
    N1489 300.7301 -3.434907 0.000000
    N1490 304.2325 -3.435278 0.000000
FINSF
QUAD4
    M1 N1 N2 N7 N9
    M2 N2 N3 N6 N7
    M3 N3 N4 N8 N6
    M4 N4 N5 N10 N8
    M5 N9 N7 N12 N14
    M6 N7 N6 N11 N12
    M7 N6 N8 N13 N11
    M8 N8 N10 N15 N13
    M9 N14 N12 N17 N19
    M10 N12 N11 N16 N17
                .
                .
                .
```

```
    M783 N1478 N1479 N1484 N1483
    M784 N1479 N1480 N1485 N1484
    M785 N1481 N1482 N1487 N1486
    M786 N1482 N1483 N1488 N1487
    M787 N1483 N1484 N1489 N1488
    M788 N1484 N1485 N1490 N1489
    M789 N1486 N1487 N162 N161
    M790 N1487 N1488 N163 N162
    M791 N1488 N1489 N164 N163
    M792 N1489 N1490 N165 N164
FINSF
SEG2
    PI1 N9 N1
    PI2 N14 N9
    PI3 N19 N14
    PI4 N24 N19
    PI5 N30 N27
    PI6 N33 N30
    PI7 N36 N33
    PI8 N39 N36
    PI9 N42 N39
    PI10 N45 N42

           .
           .
           .
    PI205 N1441 N1436
    PI206 N1446 N1441
    PI207 N1451 N1446
    PI208 N1421 N1451
    PI209 N1466 N102
    PI210 N1471 N1466
    PI211 N1476 N1471
    PI212 N1481 N1476
    PI213 N1486 N1481
    PI214 N161 N1486
FINSF
GROUP_MA NOM=ALL
     M1 M2 M3 M4 M5 M6 M7 M8
     M9 M10 M11 M12 M13 M14 M15 M16
     M17 M18 M19 M20 M21 M22 M23 M24
     M25 M26 M27 M28 M29 M30 M31 M32
     M33 M34 M35 M36 M37 M38 M39 M40
                  .
                  .
                  .
     M761 M762 M763 M764 M765 M766 M767 M768
```

```
90       M769 M770 M771 M772 M773 M774 M775 M776
91       M777 M778 M779 M780 M781 M782 M783 M784
92       M785 M786 M787 M788 M789 M790 M791 M792
93  FINSF
94  GROUP_NO NOM=DispY
95       N107 N108 N109 N27 N28 N29
96  FINSF
97  GROUP_MA NOM=PINT
98       PI1 PI2 PI3 PI4 PI5 PI6 PI7 PI8
99       PI9 PI10 PI11 PI12 PI13 PI14 PI15 PI16
100      PI17 PI18 PI19 PI20 PI21 PI22 PI23 PI24
101                    .
102                    .
103                    .
104      PI193 PI194 PI195 PI196 PI197 PI198 PI199 PI200
105      PI201 PI202 PI203 PI204 PI205 PI206 PI207 PI208
106      PI209 PI210 PI211 PI212 PI213 PI214
107 FINSF
108 GROUP_MA NOM=PLONG
109
110 FINSF
111 FIN
```

ANEXO B – ARQUIVO DE CONFIGURAÇÃO DA ANÁLISE NO CODE_ASTER

```
#Title: Default .comm file to axisymmetric pipe analysis
#@Author: Ferreira, A.D.M.
#Date: 2019
#Code version: 1.0
#Corroded pipe MEF Analysis
#PIPE SUBJECTED TO PRESURE

import aster

DEBUT(PAR_LOT='NON',);

MAIL=LIRE_MAILLAGE(FORMAT='ASTER',
                   VERI_MAIL=_F(VERIF='OUI',),);


MODMECA=AFFE_MODELE(MAILLAGE=MAIL,
                    AFFE=_F(GROUP_MA=('ALL','PINT','PLONG'),
                            PHENOMENE='MECANIQUE',
                            MODELISATION='AXIS',),);

MAIL=MODI_MAILLAGE(reuse=MAIL, MAILLAGE=MAIL,ORIE_PEAU_2D=_F(GROUP_MA=('
    PINT')));
#MAIL=MODI_MAILLAGE(reuse=MAIL, MAILLAGE=MAIL,ORIE_PEAU_2D=_F(GROUP_MA
    =('PLONG')));

SI=DEFI_FONCTION(NOM_PARA='EPSI',
                 VALE=(
                       0.0000,000.00000,
                        ),
                        );

print ('Yield Strenght' )
sigmaE=SI(0,1)
MATNLIN=DEFI_MATERIAU(ELAS=_F(E=2.05E5,
                              NU=0.3,),
                       TRACTION=_F(SIGM=SI,),);

CHMAT=AFFE_MATERIAU(MAILLAGE=MAIL,
                    AFFE=_F(GROUP_MA='ALL',
                            MATER=MATNLIN,),);

CINE0=AFFE_CHAR_CINE(MODELE=MODMECA,
```

```
                      MECA_IMPO=(_F(GROUP_NO=('DispY'),
                                    DY=0.0,),),);
de=000.0
di=000.0
s_ult=000.0
p_int=1
#220120-Invertendo o elemento e colocando plong positivo
p_long=-(p_int*(di/1000)**2)/((de/1000)**2-(di/1000)**2)
carga1=FORMULE(NOM_PARA='p_int',VALE='p_int')

CHARGE1=AFFE_CHAR_MECA(MODELE=MODMECA,
                       PRES_REP=_F(GROUP_MA='PINT',
                                   PRES=p_int,),);

CHARGE2=AFFE_CHAR_MECA(MODELE=MODMECA,
                       PRES_REP=_F(GROUP_MA='PLONG',
                                   PRES=p_long,),);

L_INST=DEFI_LIST_REEL( DEBUT=0.0,
                        INTERVALLE=(_F(JUSQU_A=p_int,NOMBRE=1,),),);

#Verificar se pequenas ('PETIT') deformacoes ou grandes ('SIMO_MIEHE')

try:
  stat=STAT_NON_LINE(MODELE=MODMECA,
           CHAM_MATER=CHMAT,
           EXCIT=(_F(CHARGE=CHARGE1),
           _F(CHARGE=CHARGE2),
           _F(CHARGE=CINE0,),),
           COMPORTEMENT=_F(RELATION='VMIS_ISOT_TRAC',
             DEFORMATION='SIMO_MIEHE',),
           INCREMENT=_F(LIST_INST=L_INST,),
         NEWTON=_F(MATRICE='TANGENTE',PREDICTION='TANGENTE'),
         #RECH_LINEAIRE=_F( ),
           CONVERGENCE=_F(ITER_GLOB_MAXI=200),
           INFO=1,);

except aster.NonConvergenceError, message:
  print 'Convergence not reached. Exiting...'
except aster.MatriceSinguliereError, message:
  print 'Singular matrix. Exiting...'
except aster.error, message:
  print "Another error occurred: %s" % str (message)
except:
  print 'error'

stat=CALC_CHAMP(
```

```
88      reuse=stat,
89      RESULTAT=stat,
90      CONTRAINTE=(
91          'SIEF_ELNO',
92          'SIGM_ELNO',
93      ),
94      CRITERES=('SIEQ_ELNO','SIEQ_NOEU'),
95  );
96
97
98  tab1=CREA_TABLE(RESU=_F(RESULTAT=stat,
99                          NOM_CHAM='SIEQ_ELNO',
100                         NOM_CMP='VMIS',
101                         TOUT='OUI',),);
102 tab2 = tab1.EXTR_TABLE ()
103
104 import numpy as np
105 print tab2.para
106 print max(tab2 ['VMIS'])
107 print ('Iniciando iteracoes')
108 maxVMises=max(tab2 ['VMIS'])
109 p_int=sigmaE/maxVMises['VMIS']
110 p_int_limit=np.ceil(2*s_ult*(de-di)/di)
111 #220120-Invertendo o elemento na malha e colocando plong positivo
112 p_long_limit=-(p_int_limit*(di/1000)**2)/((de/1000)**2-(di/1000)**2)
113 incr=p_int/2
114 print 'Pressao de Escoamento Estimada'
115 print p_int
116 print 'Pressao de Falha Superestimada'
117 print p_int_limit
118 print p_long_limit
119
120 RAMPE = DEFI_FONCTION(NOM_PARA='INST',VALE=(0,0,p_int_limit,p_int_limit)
        );
121 RAMPE2 = DEFI_FONCTION(NOM_PARA='INST',VALE=(0,0,p_int_limit,
        p_long_limit));
122
123
124
125 LOAD_PI = AFFE_CHAR_MECA_F(MODELE=MODMECA,
126                         PRES_REP=_F(PRES=RAMPE,
127                         GROUP_MA='PINT'))
128
129 LOAD_PI2 = AFFE_CHAR_MECA_F(MODELE=MODMECA,
130                         PRES_REP=_F(PRES=RAMPE2,
131                         GROUP_MA='PLONG'))
132
```

```
133
134 L_INST2=DEFI_LIST_REEL( DEBUT=0.0,
135                        INTERVALLE=(_F(JUSQU_A=p_int_limit,NOMBRE=
    p_int_limit,),
136                      ),)
137
138
139
140
141 DEFLIST=DEFI_LIST_INST(DEFI_LIST =_F(METHODE   = 'AUTO',
142           LIST_INST =L_INST2,
143           PAS_MINI  = 0.1,
144           PAS_MAXI  = 4),
145           ECHEC=_F(EVENEMENT='DELTA_GRANDEUR',
146           VALE_REF=2.5,
147           NOM_CHAM='VARI_ELGA',
148           NOM_CMP='V1'),
149           #ECHEC=_F(SUBD_NIVEAU =2., SUBD_PAS  = 4.),
150           );
151
152
153 fTrace = FORMULE(NOM_PARA= ('SIXX', 'SIYY', 'SIZZ'),VALE= """SIXX+SIYY+
    SIZZ""");
154
155 fVonMis   =   FORMULE(NOM_PARA=   ('SIXX',   'SIYY',   'SIZZ',   'SIXY',
      'SIXZ','SIYZ'),
156         VALE= """sqrt (3./2.*(SIXX**2 + SIYY**2 + SIZZ**2+ 2*SIXY**2+2*
    SIXZ**2+2*SIYZ**2)-1./2.*fTrace(SIXX, SIYY, SIZZ)**2)""");
157 #LMBO = CALC_CHAMP (reuse=RES,
158 #                  RESULTAT=RES,
159 #                  CHAM_UTIL=_F (NOM_CHAM=' SIEF_ELGA',
160 #                               FORMULE= (fTrace, fVonMis),
161 #                               NUME_CHAM_RESU=2,),)
162 #Verificar se pequenas ('PETIT') deformacoes ou grandes ('SIMO_MIEHE')
163 try:
164   stat2=STAT_NON_LINE(MODELE=MODMECA,
165           CHAM_MATER=CHMAT,
166           EXCIT=(_F(CHARGE=LOAD_PI),
167           _F(CHARGE=LOAD_PI2),
168           _F(CHARGE=CINE0,),),
169           COMPORTEMENT=_F(RELATION='VMIS_ISOT_TRAC',
170             DEFORMATION='SIMO_MIEHE',),
171           NEWTON=_F(MATRICE='TANGENTE',PREDICTION='TANGENTE'),
172           INCREMENT=_F(LIST_INST=DEFLIST,),
173           CONVERGENCE=_F(
174                #RESI_GLOB_MAXI=1e-4,
175                               #RESI_GLOB_RELA=1e-4,
```

```
                                ITER_GLOB_MAXI=200,
                                ),
          OBSERVATION = _F(NOM_CHAM='SIEF_ELGA',
          EVAL_CMP='FORMULE',
          FORMULE=fVonMis,
          NOM_CMP=('SIXX','SIXY','SIXZ','SIYY','SIYZ','SIZZ'),
          EVAL_ELGA='MAX',
          GROUP_MA='ALL'),
          INFO=1,);

except aster.NonConvergenceError, message:
  print 'Convergence not reached. Exiting...'
except aster.MatriceSinguliereError, message:
  print 'Singular matrix. Exiting...'
except aster.error, message:
  print "Another error occurred: %s" % str (message)
  print 'curva nao extrapolada'
  try:
    L_INST3=DEFI_LIST_REEL( DEBUT=0.0,
        INTERVALLE=(_F(JUSQU_A=p_int_limit,NOMBRE=p_int_limit*10,),
        ),)
    stat2=STAT_NON_LINE(reuse=stat2,
    ETAT_INIT=_F(EVOL_NOLI=stat2,),
    MODELE=MODMECA,
    CHAM_MATER=CHMAT,
    EXCIT=(_F(CHARGE=LOAD_PI),
    _F(CHARGE=LOAD_PI2),
    _F(CHARGE=CINE0,),),
    COMPORTEMENT=_F(RELATION='VMIS_ISOT_TRAC',
      DEFORMATION='SIMO_MIEHE',),
  NEWTON=_F(MATRICE='TANGENTE',PREDICTION='TANGENTE'),
  #RECH_LINEAIRE=_F ( ),
    INCREMENT=_F(LIST_INST=L_INST3,),
    CONVERGENCE=_F(
       ITER_GLOB_MAXI=200,
       #RESI_GLOB_MAXI=1e-4,
                    #RESI_GLOB_RELA=1e-4,
                    ),
    OBSERVATION = _F(NOM_CHAM='SIEF_ELGA',
    EVAL_CMP='FORMULE',
    FORMULE=fVonMis,
    NOM_CMP=('SIXX','SIXY','SIXZ','SIYY','SIYZ','SIZZ'),
    EVAL_ELGA='MAX',
    GROUP_MA='ALL'),
    INFO=1,);
  except:
    print '0.1 finalizado'
```

```
223
224 stat2=CALC_CHAMP(
225     reuse=stat2,
226     RESULTAT=stat2,
227     CONTRAINTE=(
228   'SIEF_ELNO',
229   'SIGM_ELNO',
230     ),
231     CRITERES=('SIEQ_ELNO','SIEQ_NOEU'),
232 );
233
234 tab3=CREA_TABLE(RESU=_F(RESULTAT=stat2,
235       NOM_CHAM='SIEQ_ELNO',
236       NOM_CMP='VMIS',
237       TOUT='OUI',),);
238
239 tab4 = tab3.EXTR_TABLE ()
240 print tab4.para
241 print max(tab4 ['VMIS'])
242 maxVMises=max(tab4 ['VMIS'])
243
244 #OBSV = RECU_TABLE(CO=stat2, NOM_TABLE='OBSERVATION',)
245 #IMPR_TABLE(TABLE=OBSV)
246
247 #IMPR_TABLE (TABLE=tab3)
248
249 IMPR_RESU(FORMAT='MED',
250           UNITE=80,
251           RESU=(
252                 _F(MAILLAGE=MAIL,
253                   RESULTAT=stat2,
254         ),
255         ),
256          )
257
258 FIN(FORMAT_HDF='OUI',);
```

ANEXO C – ERRO ACUMULADO PARAS AS 106 WAVELETS

Erro calculado utilizando cada *wavelet* conforme apresentado na seção 7.2.

wavelet	número	Erro (%)
bior1.1	1	13.92
bior1.3	2	2.39
bior1.5	3	2.51
bior2.2	4	1.98
bior2.4	5	1.65
bior2.6	6	2.00
bior2.8	7	1.55
bior3.1	8	3.49
bior3.3	9	2.36
bior3.5	10	2.10
bior3.7	11	2.46
bior3.9	12	1.75
bior4.4	13	1.65
bior5.5	14	2.25
bior6.8	15	1.55
coif1	16	1.12
coif10	17	0.46
coif11	18	0.46
coif12	19	0.39
coif13	20	0.59
coif14	21	0.59
coif15	22	0.59
coif16	23	0.59
coif17	24	0.59
coif2	25	2.60
coif3	26	1.60
coif4	27	1.76
coif5	28	1.37
coif6	29	0.89
coif7	30	0.83
coif8	31	0.79
coif9	32	0.46
db1	33	13.70

db10	34	1.58
db11	35	1.88
db12	36	1.78
db13	37	1.35
db14	38	1.85
db15	39	1.34
db16	40	1.35
db17	41	1.22
db18	42	1.47
db19	43	1.13
db2	44	3.08
db20	45	0.71
db21	46	0.67
db22	47	1.20
db23	48	0.83
db24	49	0.62
db25	50	0.69
db26	51	0.97
db27	52	0.83
db28	53	0.53
db29	54	0.46
db3	55	2.05
db30	56	0.22
db31	57	0.46
db32	58	0.39
db33	59	0.39
db34	60	0.62
db35	61	0.39
db36	62	0.39
db37	63	0.36
db38	64	0.59
db4	65	2.12
db5	66	2.16
db6	67	2.39
db7	68	1.94
db8	69	2.60
db9	70	2.18
dmey	71	0.62
haar	72	13.70

none	73	0.15
rbio1.1	74	13.84
rbio1.3	75	3.02
rbio1.5	76	2.34
rbio2.2	77	-2.47
rbio2.4	78	2.21
rbio2.6	79	1.88
rbio2.8	80	1.63
rbio3.3	81	0.98
rbio3.5	82	1.80
rbio3.7	83	2.43
rbio3.9	84	1.54
rbio4.4	85	2.11
rbio5.5	86	2.06
rbio6.8	87	1.55
sym10	88	1.55
sym11	89	1.78
sym12	90	1.62
sym13	91	1.48
sym14	92	1.48
sym15	93	1.42
sym16	94	1.01
sym17	95	0.88
sym18	96	0.85
sym19	97	1.05
sym2	98	3.15
sym20	99	1.13
sym3	100	2.05
sym4	101	1.75
sym5	102	2.25
sym6	103	1.99
sym7	104	1.63
sym8	105	2.44
sym9	106	1.68

Tabela 16 – Erro acumulado para as 106 wavelets.

ANEXO D – REDUÇÃO PERCENTUAL PARA AS 106 WAVELETS EM RELAÇÃO AO CASO 8.

Redução dos dados após utilizando da transformada *wavelet* até o máximo nível conforme apresentado na seção 7.2.

wavelet	número	nº coef.	Redução (%)
bior1.1	1	2	0.16
bior1.3	2	14	1.12
bior1.5	3	18	1.44
bior2.2	4	14	1.12
bior2.4	5	18	1.44
bior2.6	6	32	2.56
bior2.8	7	36	2.88
bior3.1	8	7	0.56
bior3.3	9	16	1.28
bior3.5	10	30	2.40
bior3.7	11	34	2.72
bior3.9	12	38	3.04
bior4.4	13	18	1.44
bior5.5	14	30	2.40
bior6.8	15	36	2.88
coif1	16	14	1.12
coif2	17	30	2.40
coif3	18	36	2.88
coif4	19	61	4.88
coif5	20	67	5.36
coif6	21	72	5.76
coif7	22	116	9.29
coif8	23	122	9.77
coif9	24	127	10.17
coif10	25	133	10.65
coif11	26	139	11.13
coif12	27	144	11.53
coif13	28	150	12.01
coif14	29	228	18.25
coif15	30	234	18.73
coif16	31	239	19.14

coif17	32	244	19.54
db1	33	2	0.16
db2	34	7	0.56
db3	35	14	1.12
db4	36	16	1.28
db5	37	18	1.44
db6	38	30	2.40
db7	39	32	2.56
db8	40	34	2.72
db9	41	36	2.88
db10	42	38	3.04
db11	43	59	4.72
db12	44	61	4.88
db13	45	63	5.04
db14	46	65	5.20
db15	47	67	5.36
db16	48	69	5.52
db17	49	71	5.68
db18	50	72	5.76
db19	51	74	5.92
db20	52	76	6.08
db21	53	116	9.29
db22	54	118	9.45
db23	55	120	9.61
db24	56	122	9.77
db25	57	124	9.93
db26	58	125	10.01
db27	59	127	10.17
db28	60	129	10.33
db29	61	131	10.49
db30	62	133	10.65
db31	63	135	10.81
db32	64	137	10.97
db33	65	139	11.13
db34	66	140	11.21
db35	67	142	11.37
db36	68	144	11.53
db37	69	146	11.69
db38	70	148	11.85

dmey	71	135	10.81
haar	72	2	0.16
rbio1.1	73	2	0.16
rbio1.3	74	14	1.12
rbio1.5	75	18	1.44
rbio2.2	76	14	1.12
rbio2.4	77	18	1.44
rbio2.6	78	32	2.56
rbio2.8	79	36	2.88
rbio3.1	80	7	0.56
rbio3.3	81	16	1.28
rbio3.5	82	30	2.40
rbio3.7	83	34	2.72
rbio3.9	84	38	3.04
rbio4.4	85	18	1.44
rbio5.5	86	30	2.40
rbio6.8	87	36	2.88
sym2	88	7	0.56
sym3	89	14	1.12
sym4	90	16	1.28
sym5	91	18	1.44
sym6	92	30	2.40
sym7	93	32	2.56
sym8	94	34	2.72
sym9	95	36	2.88
sym10	96	38	3.04
sym11	97	59	4.72
sym12	98	61	4.88
sym13	99	63	5.04
sym14	100	65	5.20
sym15	101	67	5.36
sym16	102	69	5.52
sym17	103	71	5.68
sym18	104	72	5.76
sym19	105	74	5.92
sym20	106	76	6.08

Tabela 17 – Redução percentual para as 106 wavelets em relação ao caso 8.

ANEXO E – MSE PARA AS 106 WAVELETS.

Erro médio quadrático para cada *wavelet* conforme apresentado na subseção 7.3.1.

wavelet	número	MSE
bior1.1	1	5.62
bior1.3	2	0.65
bior1.5	3	0.89
bior2.2	4	1.28
bior2.4	5	0.89
bior2.6	6	1.18
bior2.8	7	1.10
bior3.1	8	3.13
bior3.3	9	1.04
bior3.5	10	0.86
bior3.7	11	0.92
bior3.9	12	1.26
bior4.4	13	1.62
bior5.5	14	2.01
bior6.8	15	1.65
coif1	16	2.21
coif2	17	0.89
coif3	18	1.24
coif4	19	1.15
coif5	20	1.29
coif6	21	1.59
coif7	22	1.48
coif8	23	0.85
coif9	24	1.12
coif10	25	1.40
coif11	26	1.50
coif12	27	1.19
coif13	28	1.24
coif14	29	0.93
coif15	30	1.06
coif16	31	1.32
coif17	32	1.41
db1	33	6.99

db2	34	1.75
db3	35	2.14
db4	36	1.00
db5	37	1.46
db6	38	1.26
db7	39	1.03
db8	40	1.06
db9	41	0.92
db10	42	1.38
db11	43	1.10
db12	44	1.11
db13	45	1.35
db14	46	1.61
db15	47	1.19
db16	48	1.08
db17	49	1.94
db18	50	1.34
db19	51	1.40
db20	52	1.25
db21	53	1.10
db22	54	1.03
db23	55	1.05
db24	56	1.14
db25	57	1.23
db26	58	1.87
db27	59	1.59
db28	60	1.42
db29	61	1.23
db30	62	1.69
db31	63	1.37
db32	64	1.95
db33	65	1.24
db34	66	1.29
db35	67	1.27
db36	68	1.30
db37	69	1.55
db38	70	1.42
dmey	71	2.01
haar	72	6.99

rbio1.1	73	6.99
rbio1.3	74	1.31
rbio1.5	75	1.25
rbio2.2	76	1.13
rbio2.4	77	1.17
rbio2.6	78	1.26
rbio2.8	79	1.23
rbio3.1	80	2.55
rbio3.3	81	0.97
rbio3.5	82	1.38
rbio3.7	83	1.41
rbio3.9	84	1.45
rbio4.4	85	1.17
rbio5.5	86	0.95
rbio6.8	87	1.21
sym2	88	2.93
sym3	89	1.04
sym4	90	1.03
sym5	91	1.25
sym6	92	0.95
sym7	93	1.04
sym8	94	1.34
sym9	95	1.30
sym10	96	1.22
sym11	97	1.07
sym12	98	1.20
sym13	99	1.17
sym14	100	1.21
sym15	101	1.31
sym16	102	1.07
sym17	103	1.31
sym18	104	1.18
sym19	105	1.46
sym20	106	1.00

Tabela 18 – MSE calculado para cada uma das 106 *wavelets*.

ANEXO F – TEMPO DE TREINAMENTO PARA AS 106 WAVELETS.

Tempo de treinamento utilizando cada *wavelet* para os modelos axissimétricos conforme mostrado na subseção 7.3.1.

wavelet	número	tempo (s)
bior1.1	1	101.11
bior1.3	2	161.02
bior1.5	3	226.39
bior2.2	4	160.72
bior2.4	5	225.59
bior2.6	6	299.06
bior2.8	7	355.86
bior3.1	8	129.76
bior3.3	9	189.33
bior3.5	10	255.21
bior3.7	11	320.32
bior3.9	12	406.74
bior4.4	13	225.75
bior5.5	14	253.64
bior6.8	15	354.00
coif1	16	158.64
coif2	17	252.05
coif3	18	353.07
coif4	19	464.16
coif5	20	552.80
coif6	21	619.67
coif7	22	821.40
coif8	23	877.69
coif9	24	983.57
coif10	25	1 013.13
coif11	26	1 046.06
coif12	27	1 067.18
coif13	28	1 224.96
coif14	29	1 553.50
coif15	30	1 582.21
coif16	31	1 620.96
coif17	32	1 792.19

db1	33	100.31
db2	34	125.97
db3	35	158.86
db4	36	187.76
db5	37	225.43
db6	38	253.49
db7	39	293.62
db8	40	321.43
db9	41	355.74
db10	42	406.90
db11	43	452.10
db12	44	464.24
db13	45	527.09
db14	46	542.22
db15	47	553.99
db16	48	567.13
db17	49	611.77
db18	50	624.29
db19	51	707.76
db20	52	715.39
db21	53	823.37
db22	54	833.66
db23	55	842.71
db24	56	875.90
db25	57	880.77
db26	58	972.31
db27	59	981.30
db28	60	991.89
db29	61	1 000.96
db30	62	1 015.52
db31	63	1 026.14
db32	64	1 036.18
db33	65	1 037.75
db34	66	1 058.81
db35	67	1 065.01
db36	68	1 076.45
db37	69	1 204.84
db38	70	1 230.42
dmey	71	1 026.36

haar	72	100.17
rbio1.1	73	99.66
rbio1.3	74	159.61
rbio1.5	75	225.56
rbio2.2	76	159.17
rbio2.4	77	222.56
rbio2.6	78	297.02
rbio2.8	79	357.31
rbio3.1	80	127.70
rbio3.3	81	188.22
rbio3.5	82	253.44
rbio3.7	83	323.10
rbio3.9	84	407.96
rbio4.4	85	225.88
rbio5.5	86	254.30
rbio6.8	87	354.14
sym2	88	130.03
sym3	89	159.99
sym4	90	188.47
sym5	91	224.60
sym6	92	254.22
sym7	93	297.85
sym8	94	322.38
sym9	95	352.10
sym10	96	407.33
sym11	97	451.65
sym12	98	463.32
sym13	99	530.57
sym14	100	542.86
sym15	101	554.72
sym16	102	567.27
sym17	103	610.53
sym18	104	619.44
sym19	105	712.65
sym20	106	717.15

Tabela 19 – Tempo de treinamento, em segundos, da rede neural para cada uma das 106 *wavelets*.