Universidade Federal Fluminense

JORGE LUIZ PEREIRA

Identificação de Danos Estruturais baseada em um Modelo de Dano Contínuo via Redes Neurais Artificiais

Volta Redonda

2018

JORGE LUIZ PEREIRA

Identificação de Danos Estruturais baseada em um Modelo de Dano Contínuo via Redes Neurais Artificiais

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

Orientadora:

Rosilene Abreu Portella Corrêa

Coorientador:

Cleber de Almeida Corrêa Junior

Universidade Federal Fluminense

Volta Redonda

2018

Ficha catalográfica automática - SDC/BEM

P436i Pereira, Jorge Luiz
Identificação de Danos Estruturais baseada em um Modelo de Dano Contínuo via Redes Neurais Artificiais / Jorge Luiz Pereira ; Rosilene Abreu Portella Corrêa, orientadora ; Cleber de Almeida Corrêa Junior, coorientador. Volta Redonda, 2018.
98 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Volta Redonda, 2018.

DOI: http://dx.doi.org/10.22409/PPG-MCCT.2018.m.00905867785

1. Análise Modal. 2. Redes Neurais Artifciais. 3. Localização de Danos. 4. Backpropagation. 5. Produção intelectual. I. Título II. Corrêa,Rosilene Abreu Portella, orientadora. III. Junior, Cleber de Almeida Corrêa, coorientador. IV. Universidade Federal Fluminense. Escola de Engenharia Industrial e Metalúrgica de Volta Redonda.

CDD -

Bibliotecária responsável: Andreia de Oliveira Paim - CRB7/5183

Identificação de Danos Estruturais baseada em um Modelo de Dano Contínuo via Redes Neurais Artificiais

Jorge Luiz Pereira

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

Aprovada por:

Profa. Dra. Rosilene Abreu Portella Corrêa / MCCT-UFF

Prof. Dr. Cleber de Almeida Corrêa Junior / MCCT-UFF

Prof. Dr. Gustavo Benitez Alvarez / MCCT-UFF

Prof. Dr. Yoisell Rodríguez Núñez / MCCT-UFF

Prof. Dr. Luerbio Faria / IME-UERJ

Prof. Dr. Bruno da Fonseca Monteiro / DEG-UFRJ

Volta Redonda, 13 de Julho de 2018.

Para meus pais, minha filha Débora, minha esposa Cristina e minha irmã Denise.

Agradecimentos

A Deus, pela vida e saúde para chegar até aqui.

Ao meu pai Floracy (*in memorian*) pelo apoio.

A minha mãe Marilza pelo incentivo.

A minha amada filha Débora pela paciência nas minhas ausências.

A minha esposa Cristina por toda palavra de ânimo.

A minha irmã Denise pela ajuda.

Aos meus orientadores Rosilene e Cleber pela dedicação que tiveram em me auxiliar.

A todos os demais que contribuíram para a realização deste trabalho de forma direta ou indiretamente.

Resumo

Danos estruturais, além de comprometer o desempenho da estrutura, ainda podem colocar vidas humanas em risco e causar grandes prejuízos financeiros. O monitoramento contínuo da mesma visa que danos em estágio inicial sejam identificados, aumentando a confiabilidade e a segurança daqueles que a utilizam. Neste trabalho, utiliza-se um modelo de identificação de danos baseado na matriz de flexibilidade estrutural onde o campo de danos para a viga de Euler-Bernoulli simplesmente apoiada é descrito via Método dos Elementos Finitos (Stutz, 2005 [41]). Apresenta-se uma revisão do desempenho dos modelos de localização de danos analisados por Corrêa, 2013 [10] e sugere-se a utilização de Redes Neurais Artificiais (RNA) para lidar com os problemas de identificar danos quando dados incompletos são utilizados e quando há presença de sinais corrompidos por ruído. Apresenta-se também uma breve revisão sobre a utilização de RNA na identificação de danos, assim como um estudo sobre o seu funcionamento. Resultados numéricos da identificação de falhas estruturais em diferentes posições da viga são apresentados utilizando-se a RNA, assim como um estudo sobre a capacidade desta em identificar defeitos estruturais na presença de número limitado de graus de liberdade e dados ruidosos.

Abstract

Structural damage, as well as compromising the performance of the structure, you can still put lives at risk and cause major financial losses. The continuous monitoring of the same aims that damage in the initial stage are identified, increasing the reliability and security of those who use it. In this study, it uses a model of identification of damage based on the matrix of structural flexibility where the field of damage to Euler-Bernoulli beam simply supported is described via the Finite Element Method (Stutz, 2005 [41]). It presents a review of the performance of the models of location of damage analyzed by Corrêa, 2013 [10] and suggests the use of artificial neural networks (ANN) to deal with the problems of identifying damage when incomplete data are used and when there is presence of signals corrupted by noise. It also presents a brief review on the use of RNA in the identification of damage, as well as a study on its operation. Numerical results from the identification of damage in different positions of the beam are presented using the RNA, as well as the ability to identify structural damage in the presence of a limited number of degrees of freedom and noisy data.

Palavras-chave

1. Análise Modal
2. Método dos Elementos Finitos
3. Redes Neurais Artificiais
4. Parâmetro de Coesão
5. Matriz de Rigidez
6. Matriz de Flexibilidade
7. Localização de Danos
8. *Backpropagation*
9. Redes *Perceptron* de Múltiplas Camadas
10. Modelo de Dano Contínuo
11. Funções de Interpolação

Glossário

FEMAP	:	*Finit Element Modeling and Postprocesing*
MATLAB	:	*Matrix Laboratory*
MEF	:	Método dos Elementos Finitos
NASTRAN	:	*Nasa Structural Analysis*
RNA	:	Redes Neurais Artificiais
CALFEM	:	*Compute Aided Learning of the Finite Element Method*
PMC	:	*Perceptron* de Múltiplas Camadas
FRF	:	Funções de Resposta em Frequência
GDL	:	Graus de Liberdade
NNTool	:	*Neural Network Toolbox*
PSO	:	*Particle Swarm Optimization*
MDCM	:	Multiplicação da Diferença na Curvatura dos Modos
CAD	:	*Computer Aided Design*
MAC	:	*Modal Assurance Criterion*
Adaline	:	*ADAptive LINear Element*
LMS	:	*Least Mean Square*

Lista de símbolos

$\boldsymbol{u}$	:	vetor de deslocamentos com dimensão $n \times 1$.
$\boldsymbol{K}$	:	matriz de rigidez com dimensão $n \times n$.
$\boldsymbol{K^{-1}}$	:	matriz inversa da matriz de rigidez com dimensão $n \times n$.
$\boldsymbol{G}$	:	matriz de flexibilidade com dimensão $n \times n$.
$\boldsymbol{f}$	:	vetor de forças com dimensão $n \times 1$.
u_{ij}	:	deslocamento em i provocado por uma força externa unitária aplicada em j.
u_{ji}	:	deslocamento em j provocado por uma força externa unitária aplicada em i.
G_{ij}	:	elemento da matriz de flexibilidade na posição ij.
G_{ji}	:	elemento da matriz de flexibilidade na posição ji.
$\boldsymbol{M}$	:	matriz de massa com dimensão $n \times n$.
$\boldsymbol{C}$	:	matriz de amortecimento com dimensão $n \times n$.
$\boldsymbol{X}$	:	vetor de deslocamentos nodais com dimensão $n \times 1$.
$\dot{\boldsymbol{X}}$	:	vetor de velocidades nodais com dimensão $n \times 1$.
$\ddot{\boldsymbol{X}}$	:	vetor de acelerações nodais com dimensão $n \times 1$.
$\boldsymbol{\phi_i}$	:	*i-ésima* forma modal da estrutura ou é o *i-ésimo* autovetor.
T	:	representa a transposição de uma matriz.
t	:	tempo ou quantidade de elementos finitos.
ω_i	:	*i-ésima* frequência natural da estrutura ou o *i-ésimo* autovalor.
ω_i^2	:	*i-ésima* frequência natural da estrutura ao quadrado.
φ	:	ângulo de fase.
$\boldsymbol{I}$	:	matriz identidade.
$\boldsymbol{\Phi}$	:	matriz modal da estrutura com dimensão $n \times n$.
$\boldsymbol{\Phi^T}$	:	transposta da matriz modal.
$\boldsymbol{\Phi^{-T}}$	:	inversa da transposta da matriz modal.
$\boldsymbol{\Phi^{-1}}$	:	inversa da matriz modal.
$\boldsymbol{\Lambda}$	:	matriz diagonal $n \times n$ formada pelos valores quadráticos das frequências naturais.
$\boldsymbol{\Lambda^{-1}}$	:	inversa da matriz $\boldsymbol{\Lambda}$.
m	:	número de modos experimentais medidos no teste de vibração.
$\boldsymbol{G_{exp}}$	:	matriz de flexibilidade experimental.

Lista de símbolos

$\omega_{i,exp}$	:	*i-ésima* frequência natural obtida experimentalmente.
$\boldsymbol{\phi_{i,exp}}$	:	*i-ésima* forma modal obtida experimentalmente.
$\boldsymbol{K_{mm}}$	:	submatriz de rigidez onde m refere-se aos graus de liberdade medidos.
$\boldsymbol{K_{mo}}$	:	submatriz de rigidez onde o refere-se aos graus de liberdade omitidos.
$\boldsymbol{K_{mo}^T}$	:	transposta da submatriz de rigidez $\boldsymbol{K_{mo}}$.
$\boldsymbol{K_{oo}}$	:	submatriz de rigidez.
$\boldsymbol{K_{oo}^{-1}}$	:	inversa da submatriz de rigidez $\boldsymbol{K_{oo}}$.
$\boldsymbol{G_{mm}}$	:	submatriz de flexibilidade: m refere-se aos graus de liberdade medidos.
$\boldsymbol{G_{mo}}$	:	submatriz de flexibilidade: o refere-se aos graus de liberdade omitidos.
$\boldsymbol{G_{mo}^T}$	:	transposta da submatriz de flexibilidade $\boldsymbol{G_{mo}}$.
$\boldsymbol{G_{oo}}$	:	submatriz de flexibilidade.
A_1^e, A_2^e, A_3^e, A_4^e	:	são as funções de interpolação do campo de deslocamento do elemento.
L	:	é o comprimento de cada elemento finito.
$\boldsymbol{K^e}$	:	matriz de rigidez do elemento.
$E(\overline{x})$	:	módulo de elasticidade ou módulo de Young.
$I(\overline{x})$	:	momento de inércia da área da seção transversal da estrutura.
E_0	:	valor inicial do módulo de elasticidade.
I_0	:	momento de inércia inicial da área da seção transversal da estrutura.
$\boldsymbol{A^e}$	:	vetor do elemento finito cujos elementos são A_1^e, A_2^e, A_3^e e A_4^e.
$\overline{x}$	:	coordenada espacial local com origem no centro do elemento.
$\overline{y}$	:	coordenada espacial local com origem no centro do elemento.
$u(\overline{x})$	:	polinômio que aproxima o campo de deslocamentos.
$\simeq$	:	aproximadamente.
$\overline{y}^e(\bar{x})$	:	função desconhecida no elemento.
m	:	unidade de comprimento: *metro*.
kg	:	unidade de massa: *quilograma*.
m^3	:	unidade de volume: *metro cúbico*.
Pa	:	unidade de pressão: *Pascal*.

Lista de símbolos

m^4	:	unidade do momento de inércia (para medidas em *metro*).
Hz	:	unidade de medida para frequência: *Hertz*.
N_1^e, N_2^e	:	funções de interporlação do campo de coesão.
β_1^e	:	valor nodal de coesão no nó 1 do elemento finito.
β_2^e	:	valor nodal de coesão no nó 2 do elemento finito.
h_0	:	indica a espessura original da estrutura.
$h(x)$	:	indica a espessura na posição danificada.
x	:	coordenada espacial global cartesiana do vértice do dano triangular.
θ	:	*Bias* ou limiar de ativação ou *threshold* de um neurônio artificial.
$x_1, x_2, \cdots, x_n$	:	sinais de entrada de uma rede neural artificial.
$w_1, w_2, \cdots, w_n$	:	são os pesos sinápticos de uma rede neural artificial.
Σ	:	combinador linear de uma rede neural artificial.
u	:	potencial de ativação de uma rede neural artificial.
g	:	função de ativação de uma rede neural artificial.
$y_1, y_2, \cdots, y_n$	:	sinais de saída de uma rede neural artificial.
w_i	:	*i-ésimo* peso sináptico.
x_i	:	*i-ésimo* sinal de entrada.
c	:	representa a largura de uma estrutura.
d	:	valor do comprimento da base do triângulo que forma o perfil do dano.
$\mathbf{\Phi}_r$	:	corresponde aos modos de vibração contaminados por ruído.
rand	:	função geradora de números aleatórios.

Sumário

Lista de Figuras **xiv**

Lista de Tabelas **xviii**

1 Introdução **19**

1.1 Motivação 19

1.2 Revisão Bibliográfica 21

1.3 Objetivos 25

1.4 Organização do Trabalho 25

2 Fundamentos Teóricos **27**

2.1 Análise Matricial de Estruturas 27

2.2 Análise Modal 31

2.3 Redes Neurais Artificiais 36

2.3.1 Resumo Histórico 37

2.3.2 Neurônios Biológicos 38

2.3.3 Neurônios Artificiais 40

2.3.4 Funções de Ativação 42

2.3.5 Arquiteturas de Redes Neurais Artificiais 45

2.3.5.1 Arquitetura *feedforward* de Camadas Simples 45

2.3.5.2 Arquitetura *feedforward* de Camadas Múltiplas 46

2.3.5.3 Arquitetura Recorrente ou Realimentada 47

2.3.5.4 Arquitetura em Estrutura Reticulada 47

2.3.6 Topologias de Redes Neurais Artificiais 48

2.3.7 Tipos de Treinamento em uma Rede Neural Artificial 48

2.3.8 Redes *Perceptron* de Múltiplas Camadas 49

2.3.9 Princípio de Funcionamento de Redes *Perceptron* de Múltiplas Camadas 50

2.3.10 Processo de Treinamento de Redes *Perceptron* de Múltiplas Camadas 51

2.3.10.1 O Algoritmo *Backpropagation* 52

2.3.10.2 O Algoritmo *Backpropagation* com Taxa de *Momentum* . 62

2.3.10.3 Algoritmo *Backpropagation* - Pseudocódigo 63

3 Modelagem do Problema 65

3.1 Definição de Parâmetros - Análise Modal e Método dos Elementos Finitos . 65

3.1.1 Modelo de Dano Constante por Partes 70

3.1.2 Modelo de Dano Contínuo 70

4 Estudo de Caso 75

4.1 Parâmetros da Rede Neural Artificial 75

4.1.1 Definição dos Padrões de Entrada e Saída 77

4.1.2 Definição dos Parâmetros de Treinamento da RNA 78

4.2 Resultados Computacionais 80

4.2.1 Teste da Rede Neural Artificial 80

4.2.2 Dano na Posição 4 81

4.2.2.1 Com Ruído de 0% 81

4.2.2.2 Com Ruído de 3% 82

4.2.3 Dano na Posição 10 83

4.2.3.1 Com Ruído de 0% 83

4.2.3.2 Com Ruído de 3% 84

4.2.4 Dano na Posição 13 85

4.2.4.1 Com Ruído de 0% 86

4.2.4.2 Com Ruído de 3% 87

4.2.5 Dano na Posição 16 88

4.2.5.1 Com Ruído de 0% 88

4.2.5.2 Com Ruído de 3% 89

5 Conclusões e Trabalhos Futuros **91**

5.1 Conclusões 91

5.2 Trabalhos Futuros 92

Referências **94**

Lista de Figuras

1.1 Dano em uma viga de concreto armado. Fonte: http://www.carmona.eng.br (Acessado em 04/2018). 19

1.2 Ensaio não destrutivo. Fonte: http://www.modulusengenharia.com.br (Acessado em 04/2018). 20

1.3 Placa simulada, com os furos que deverão ser localizados. Fonte: Martinez, 2013 [27] 23

1.4 Discretização da viga metálica. Fonte: Maia, 2017 [22] 24

1.5 Seção transversal do elemento de viga. Fonte: Maia, 2017 [22] 24

1.6 Resultados obtidos para uma das Redes. Fonte: Maia, 2017 [22]. 25

2.1 Ilustração do Teorema de Maxwell. Fonte: sinop.unemat.br (Acessado em 01/2018) 28

2.2 Ação de duas forças externas em uma viga. Fonte: sinop.unemat.br (Acessado em 07/2018) 28

2.3 Ação de P_1. Fonte: sinop.unemat.br (Acessado em 07/2018) 29

2.4 Ação de P_2. Fonte: sinop.unemat.br (Acessado em 07/2018) 29

2.5 Viga deformada sob ação das forças P_1 e P_2. Fonte: sinop.unemat.br (Acessado em 07/2018) 30

2.6 Esquema de um neurônio biológico. Fonte: Silva, 2010 [36] 39

2.7 Conexões sinápticas entre neurônios. Fonte: Silva, 2010 [36]. 40

2.8 Neurônio artificial. Fonte: Adaptado de Silva, 2010 [36]. 41

2.9 Gráfico da função degrau. Fonte: Silva, 2010 [36] 42

2.10 Gráfico da função degrau bipolar. Fonte: Silva, 2010 [36] 43

2.11 Gráfico da função rampa simétrica. Fonte: Silva, 2010 [36] 43

2.12 Gráfico da função logística. Fonte: Silva, 2010 [36] 44

2.13 Gráfico da função tangente hiperbólica. Fonte: Silva, 2010 [36] 44

2.14 Gráfico da função gaussiana. Fonte: Silva, 2010 [36] 44

2.15 Gráfico da função linear. Fonte: Silva, 2010 [36] 45

2.16 Arquitetura *feedforward* de camada simples. Fonte: Silva, 2010 [36]. 46

2.17 Arquitetura *feedforward* de camadas múltiplas. Fonte: Silva, 2010 [36]. . . 46

2.18 Arquitetura recorrente ou realimentada. Fonte: Silva, 2010 [36]. 47

2.19 Arquitetura em estrutura reticulada. Fonte: Silva, 2010 [36]. 47

2.20 Exemplo de rede *Perceptron* multicamadas. Fonte: Silva, 2010 [36]. 49

2.21 As duas fases de treinamento da rede PMC. Fonte: Silva, 2010 [36]. 51

2.22 Notação para derivação do algoritmo *backpropagation*. Fonte: Silva, 2010 [36]. 53

2.23 Configuração de neurônio utilizado na derivação do algoritmo *backpropagation*. Fonte: Silva, 2010 [36]. 54

2.24 Ilustração do processo de treinamento utilizando o método de inserção do termo de *momentum*. Fonte: Silva, 2010 [36]. 63

3.1 (a) Viga contínua e (b) viga discretizada em 20 elementos. 66

3.2 Viga discretizada em 20 elementos finitos e com as 21 nós ou posições. . . . 67

3.3 Elemento finito padrão e GDL por elemento. 67

3.4 Modelo de dano constante por partes. 70

3.5 Perfil do dano simulado. 71

3.6 Dois elementos afetados pelo dano. 71

3.7 Função de interpolação unidimensional linear N_1. 72

3.8 Função de interpolação unidimensional linear N_2. 72

3.9 Elemento padrão e parâmetros de coesão. 73

3.10 Parâmetro de coesão β como função da relação entre a espessura da viga com dano $h(x)$ e sem dano h_0 (Equação (3.19)). 74

4.1 Função de ativação Tangente Hiperbólica (*tansig*). Fonte: Demuth, 1994 [13]. 75

4.2 Função de ativação Logística (*logsig*). Fonte: Demuth, 1994 [13]. 76

4.3 Topologia de rede PMC adotada nesta dissertação. 77

4.4 Ilustração do dano na posição 4 da viga discretizada. 81

4.5 Dano na posição 4 com ruído de 0% e $\frac{h(x)}{h_0} = 0,80$. 81

4.6 Dano na posição 4 com ruído de 0% e $\frac{h(x)}{h_0} = 0,70$. 82

4.7 Dano na posição 4 com ruído de 3% e $\frac{h(x)}{h_0} = 0,80$. 82

4.8 Dano na posição 4 com ruído de 3% e $\frac{h(x)}{h_0} = 0,70$. 83

4.9 Ilustração do dano na posição 10 da viga discretizada. 83

4.10 Dano na posição 10 com ruído de 0% e $\frac{h(x)}{h_0} = 0,80$. 84

4.11 Dano na posição 10 com ruído de 0% e $\frac{h(x)}{h_0} = 0,70$. 84

4.12 Dano na posição 10 com ruído de 3% e $\frac{h(x)}{h_0} = 0,80$. 85

4.13 Dano na posição 10 com ruído de 3% e $\frac{h(x)}{h_0} = 0,70$. 85

4.14 Ilustração do dano na posição 13 da viga discretizada. 86

4.15 Dano na posição 13 com ruído de 0% e $\frac{h(x)}{h_0} = 0,80$. 86

4.16 Dano na posição 13 com ruído de 0% e $\frac{h(x)}{h_0} = 0,70$. 86

4.17 Dano na posição 13 com ruído de 3% e $\frac{h(x)}{h_0} = 0,80$. 87

4.18 Dano na posição 13 com ruído de 3% e $\frac{h(x)}{h_0} = 0,70$. 87

4.19 Ilustração do dano na posição 16 da viga discretizada. 88

4.20 Dano na posição 16 com ruído de 0% e $\frac{h(x)}{h_0} = 0,84$. 88

4.21 Dano na posição 16 com ruído de 0% e $\frac{h(x)}{h_0} = 0,70$. 89

4.22 Dano na posição 16 com ruído de 3% e $\frac{h(x)}{h_0} = 0,84$. 89

4.23 Dano na posição 16 com ruído de 3% e $\frac{h(x)}{h_0} = 0,70$. 90

Lista de Tabelas

3.1 Frequências naturais da viga biapoiada. 66

3.2 Propriedades nominais da viga. 67

4.1 Intensidade dos danos ou os valores do parâmetro β para a fase de treinamento. 78

4.2 Intensidade dos danos ou os valores do parâmetro β para a fase de teste. . 78

4.3 Posições em que foram colocados os danos para o treinamento da rede PMC. 79

4.4 Posições escolhidas para testar a RNA após o treinamento. 80

Capítulo 1

Introdução

1.1 Motivação

Estruturas de pequeno ou de grande porte - tais como residências, edifícios, pontes, barragens - desfrutam hoje em dia de avanços tecnológicos ligados a processos construtivos e uso de materiais mais elaborados que aumentam a segurança e a resistência da estrutura. Porém, apesar da sofisticação, as estruturas podem apresentar falhas mesmo em condições normais de uso. Danos estruturais podem prejudicar a performance e a integridade de uma estrutura, podendo colocar em risco vidas humanas (Figura 1.1). Inserida nesse contexto, a identificação de danos em estruturas é um tópico relevante tanto para o meio acadêmico quanto para a indústria (Cunha, 2014 [11]).

Figura 1.1: Dano em uma viga de concreto armado. Fonte: http://www.carmona.eng.br (Acessado em 04/2018).

Por isso, as pesquisas sobre identificação de danos, em uma determinada estrutura,

bem no seu início e, o seu contínuo monitoramento, têm chamado a atenção de estudiosos de diversas áreas como: Engenharia Civil, Engenharia Mecânica, Engenharia Naval, Engenharia Aeronáutica, e assim por diante. A identificação do dano na fase inicial permite a redução de custos de manutenção e reparação, possibilitando um aumento da vida útil da estrutura analisada, além de fornecer segurança e conforto ao ser humano (Carrillo, 2007 [9]).

Várias técnicas de identificação de danos, mesmo se tratando de ensaios não destrutivos de inspeção, ou são baseados em observações visuais ou são métodos experimentais localizados - Figura 1.2. Estes métodos porém, apresentam algumas desvantagens quando aplicados a estruturas complexas, como: conhecimento da área próxima ao dano *a priori* e facilidade de acesso à parte da estrutura a ser inspecionada (Miguel, 2007 [25]).

Figura 1.2: Ensaio não destrutivo. Fonte: http://www.modulusengenharia.com.br (Acessado em 04/2018).

Portanto, a necessidade de métodos adicionais não destrutivos para identificação de danos, que possam ser aplicados a estruturas mais complexas, levou ao desenvolvimento de técnicas baseadas nas características de vibração da estrutura como apresentado por Stutz (1999 [41]).

Rytter (1993) [34] classifica os métodos de inspeção de danos em quatro níveis:

Nível 1: detecção da presença de dano na estrutura;

Nível 2: localização da região afetada pelo dano estrutural;

Nível 3: indicação da severidade do dano;

Nível 4: previsão da vida útil restante.

A minimização de um funcional, criado com base na diferença entre a matriz de flexibilidade numérica e a do experimento sintético do problema de identificação de danos em vigas via métodos de otimização estocásticos, apresentou resultados satisfatórios conforme

pode ser visto no trabalho de Corrêa, 2013 [10]. No entanto, estruturas mais complexas, onde um grande número de variáveis precisam ser atualizados, podem gerar um grande custo computacional ao se utilizar tais métodos, o que dificulta a sua utilização para um monitoramento contínuo estrutural. No trabalho supracitado fez-se um estudo sobre diversos métodos de localização de danos, visando utilizá-los a fim de diminuir o número de parâmetros a serem atualizados pelos métodos de otimização. O objetivo era fazer o processo de identificação de danos em duas etapas, onde na primeira etapa se utilizaria um método de localização para determinar a região afetada pelo dano estrutural e, em uma segunda etapa, um método de otimização atualizaria os parâmetros apenas dessa região, diminuindo assim a dimensão do problema. Infelizmente, nenhum dos métodos utilizados apresentou bons resultados quando um número reduzido de graus de liberdade foram medidos, e ainda, foi impossível distinguir a posição do dano na presença de sinais ruidosos. Constatada esta deficiência dos métodos de localização estudados, optou-se no presente trabalho, pela utilização de redes neurais artificiais (RNA) para substituir tais métodos nessa primeira etapa do processo de identificação de danos. A pesquisa e os resultados sobre a utilização de RNA no processo de identificação de danos baseado em um modelo de dano contínuo será apresentada nesta dissertação. Redes neurais artificiais são modelos computacionais inspirados no sistema nervoso de seres vivos. Possuem a capacidade de aquisição e manutenção do conhecimento (baseado em informações). São um conjunto de unidades de processamento que são interligados por um grande número de interconexões (sinapses artificiais) (Silva, 2010 [36]).

1.2 Revisão Bibliográfica

Corrêa (2013) [10] estuda a identificação de danos em estruturas via matriz de flexibilidade e utiliza o modelo de identificação de danos baseado em um modelo de dano contínuo apresentado por Stutz *et al* (2005) [42]. Apresentou-se os resultados da identificação de danos em uma viga simplesmente apoiada, em uma viga engastada em uma de suas extremidades e em uma placa. Na primeira parte do seu trabalho, em uma viga de alumínio simplesmente apoiada, utilizou-se e comparou-se sete métodos para localização de danos:

1. Variação na matriz de flexibilidade;
2. Método da curvatura dos modos;
3. Índice de danos baseados na flexibilidade e na curvatura dos modos;

4. Variação na curvatura da flexibilidade;
5. Multiplicação da diferença na curvatura dos modos (MDCM);
6. Critério de confiança modal – MAC;
7. Vetor de localização de dano baseado na matriz de flexibilidade.

A influência de danos e ruídos nos modos de vibração é levada em consideração na referência supracitada, que apresenta uma análise numérica do comportamento de uma viga de Euler-Bernoulli simplesmente apoiada. Foi constatado que existem regiões na viga onde a identificação dos danos apresenta maior dificuldade, principalmente na presença de dados ruidosos, que influenciam bastante os modos de vibração. No trabalho citado acima são apresentados os resultados obtidos com os sete métodos de localização de danos citados acima. Mostrou-se que na ausência de ruído nos sinais, e quando um número elevado de graus de liberdade são considerados, mesmo em um cenário de dano duplo e de pouca intensidade, os danos foram localizados com boa exatidão. Em outra simulação, ainda utilizando sete métodos de localização de danos, mostrou-se que com um nível de 1% de ruído incorporado aos modos de vibração considerados medidos, com apenas um dano e com maior intensidade do que a simulação anterior, constatou-se que os métodos de localização de danos apresentaram dificuldade em fornecer a região afetada pela falha estrutural, com exceção, neste caso, do método chamado Multiplicação da Diferença na Curvatura dos Modos (MDCM). Em uma terceira simulação, quando 3% de nível de ruído é incorporado aos modos de vibração, nenhum dos sete métodos foi capaz de determinar com clareza a região afetada pelo dano estrutural. Neste caso, devido às limitações encontradas pelos métodos de localização analisados, para localizar os danos impostos, os resultados obtidos foram omitidos. Verificou-se, também, que os sete métodos utilizados necessitam de um número de pontos de medições elevado, não sendo adequado, mesmo para pequenos níveis de ruído. Corrêa (2013) [10] conclui que nenhum dos sete métodos foi eficiente em determinar a região afetada por um dano estrutural na presença de dados ruidosos ou com dados incompletos de medição, como número reduzido de graus de liberdade (GDL) e modos de vibração medidos. E ainda diz que diversos pesquisadores têm afirmado que o uso de Redes Neurais Artificiais (RNA) é uma alternativa para lidar com problemas de identificação de danos, devido à sua robutez, adaptabilidade e capacidade de lidar com dados incompletos.

Martinez (2013) [27] trabalha com a modelagem do problema inverso de detecção de danos via Método dos Elementos Finitos (MEF) e RNA. Implementa o MEF em um

software chamado FEMAP/NASTRAN que é utilizado para a modelagem de uma peça (placa com furos) mediante um programa de CAD - ver Figura 1.3. Após a análise realizada no software, dados das diferentes tensões que se localizam em posições distintas da peça são coletados para uma posterior utilização dentro de um programa implementado com o software MATLAB (Bang, 2000 [5]). Neste software está desenvolvido um programa (toolbox) para redes neurais artificiais (Martinez, 2013 [27]). Com este programa pronto e implementado, verificou-se que o programa não foi capaz de determinar com precisão a posição dos furos na placa. Na maioria das vezes, localizava em uma posição próxima ao do furo. Martinez (2013) [27] faz um trabalho específico para localização de furos. Neste trabalho não são utilizados dados vibracionais (modos, frequências, etc.).

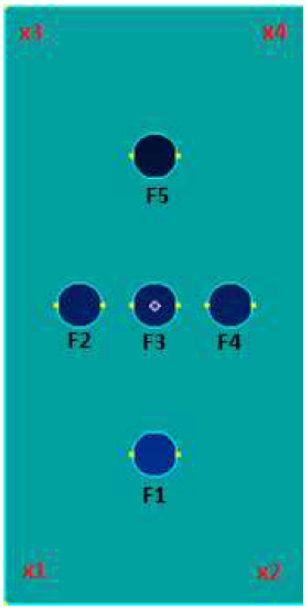

Figura 1.3: Placa simulada, com os furos que deverão ser localizados. Fonte: Martinez, 2013 [27]

Maia (2017) [22] utiliza RNA e as características dinâmicas da estrutura para analisar a integridade de elementos estruturais. Ele testou várias RNA utilizando o algoritmo *backpropagation* utilizando as características dinâmicas obtidas numericamente e experimentalmente de uma viga metálica e três pórticos planos. Para fase de treinamento da RNA foram criados modelos numéricos das estruturas estudadas e as suas propriedades dinâmicas foram utilizadas como entradas das redes. Já para a fase de teste, foram utilizados dados provenientes tanto de modelos numéricos quanto de estruturas ensaiadas experimentalmente. A análise numérica da viga metálica foi dividida em duas etapas: modelagem numérica e treinamento das RNA. Na etapa de modelagem numérica foi utilizado o Método dos Elementos Finitos (MEF). Usou-se uma rotina computacional criada

em ambiente MATLAB que faz uso das equações da Análise modal (que serão descritas no próximo capítulo) para obtenção das características dinâmicas da estrutura (ver Figuras 1.4, onde a medida está em metro, e 1.5, onde as medidas estão em milímetro).

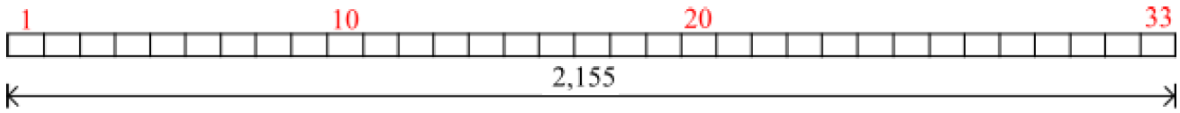

Figura 1.4: Discretização da viga metálica. Fonte: Maia, 2017 [22]

Para identificar os danos, Maia (2017) [22] utiliza os gráficos do *Neural Network Toolbox* (NNTool) do MATLAB para RNA. É mostrado um gráfico para o treinamento, outro para validação e mais um para o teste da rede neural. É apresentado ainda um quarto gráfico intitulado *All*, onde são plotados todos os pontos referentes às três fases anteriores, para que possa ser feita uma avaliação geral do desempenho da rede (ver Figura 1.6).

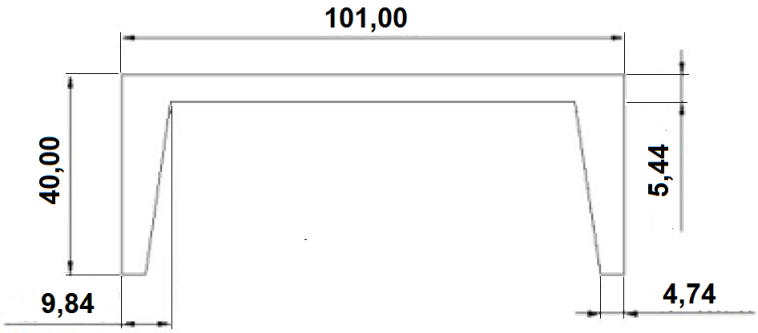

Figura 1.5: Seção transversal do elemento de viga. Fonte: Maia, 2017 [22]

A técnica mostrada por Maia (2017) [22] só mostra se há ou não dano. O critério é o seguinte: o valor de R, apresentado no topo de cada gráfico da Figura 1.6, representa a relação entre a resposta da rede neural artificial e a resposta esperada. Quanto mais próximo de 1 este valor estiver, melhor é o desempenho da rede nas fases de treinamento, validação e teste. Maia (2017) [22] considera aceitáveis, na identificação de danos, valores de R acima de $0,92$ para as fases de validação (*Validation*) e generalização (*Test*), representando um erro aceitável de 8% (ver Figura 1.6).

Todo o estudo de Maia (2017) [22] é feito com base nestes gráficos e nos critérios mencionados acima, inclusive os estudos envolvendo pórticos planos. Na grande maioria das vezes o R da validação e da generalização ficam com valores acima de $0,92$.

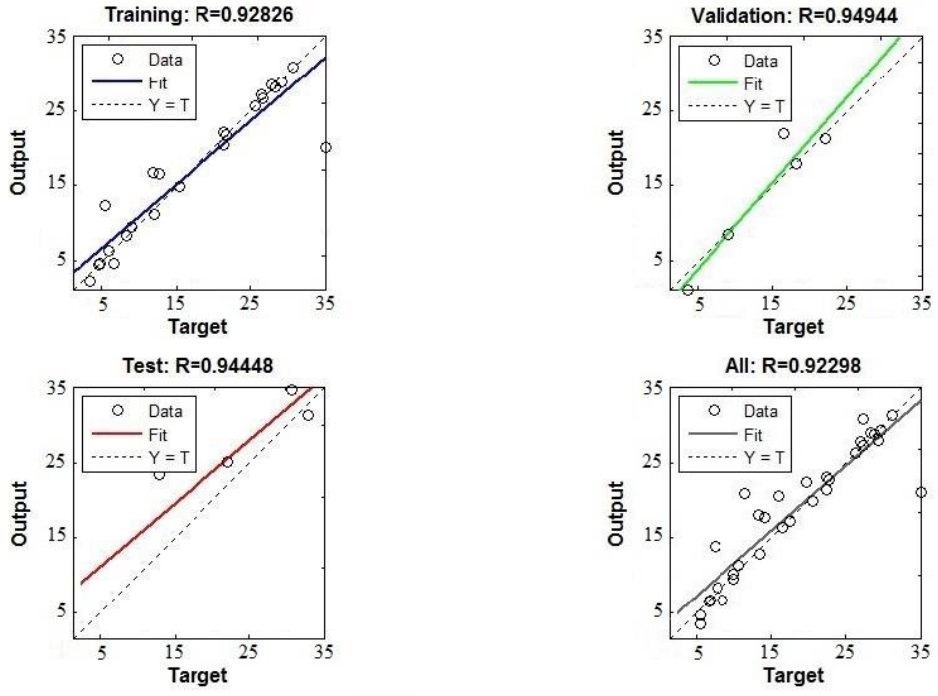

Figura 1.6: Resultados obtidos para uma das Redes. Fonte: Maia, 2017 [22].

1.3 Objetivos

Os objetivos deste trabalho são:

- Sabendo-se das limitações encontradas pelos métodos de localização de danos analisados por Corrêa (2013) [10], o objetivo deste trabalho é utilizar as características de generalização de uma Rede Neural Artificial para conseguir localizar e quantificar danos em uma viga de Euler-Bernoulli simplesmente apoiada tanto na presença de dados incompletos de medição quanto na presença de sinais ruidosos;

- Desenvolver um modelo de identificação de danos estruturais baseado em redes neurais artificiais que possa ser expandido para estruturas mais complexas, de forma a substituir os métodos de otimização utilizados por Corrêa, 2013 ou, pelo menos, diminuir o número de parâmetros a serem atualizados por tais métodos, funcionando assim como um método de localização de danos.

1.4 Organização do Trabalho

A dissertação está organizada em cinco capítulos. Neste capítulo introdutório, a motivação, a revisão bibliográfica e os objetivos deste trabalho foram apresentados.

No Capítulo 2 apresenta-se o modelo matemático para o problema direto de identificação de danos via matriz de flexibilidade estrutural. Logo após, define-se neurônio artificial e rede neural artificial. Aborda-se também o seguinte: função de ativação de um neurônio artificial, arquitetura, topologia e modos de treinar uma rede neural artificial. Comenta-se também sobre a arquitetura *perceptron* de múltiplas camadas e sobre o algorítmo de treinamento *bacpropagation*.

No Capítulo 3 é descrita a metodologia utilizada neste trabalho. Fala-se da discretização de uma viga de Euler-Bernoulli, contínua e com 1 metro de comprimento, em 20 elementos finitos e do campo de deslocamentos de um elemento finito padrão. Apresenta-se a matriz de rigidez $\mathbf{K}^e$ para cada elemento padrão da viga discretizada. Comenta-se brevemente sobre a diferença entre o modelo de dano constante por partes e o modelo de dano contínuo. Adota-se neste trabalho o segundo modelo citado. Apresenta-se o campo de danos em um elemento padrão. Define-se o parâmetro de coesão β. Fala-se um pouco mais sobre as redes neurais artificiais: funções de ativação utilizadas nas simulações com RNA, o tipo de arquitetura adotada, como também a topologia. Mostra-se as propriedades nominais da viga em estudo. Explica-se como os padrões de entrada da rede neural foram criados.

No Capítulo 4 são exibidos os resultados computacionais para as simulações realizadas em uma viga de Euler-Bernoulli simplesmente apoiada. Foram considerados diversos cenários de dano ao longo da viga, tanto na presença quanto na ausência de sinais ruidosos.

E no Capítulo 5 encontram-se as conclusões sobre o trabalho, discorrendo-se sobre desempenho das redes neurais artificiais na localização e quantificação de dano estrutural. Também fala-se sobre pesquisas que podem ser feitas no futuro para aprofundar o que foi discutido neste trabalho ou, até mesmo, para descobrir novos caminhos na utilização das redes neurais.

Capítulo 2

Fundamentos Teóricos

O presente capítulo tem como objetivo introduzir o conhecimento teórico necessário para o desenvolvimento do trabalho, esclarecendo os conceitos que serão utilizados nos capítulos seguintes.

2.1 Análise Matricial de Estruturas

Da Análise Matricial de Estruturas[1], sabe-se que a matriz de flexiblidade $\boldsymbol{G}$ de uma estrutura com n Graus de Liberdade (GDL) é a matriz $n \times n$ definida como a matriz inversa de sua matriz de rigidez $\boldsymbol{K}$. Daí, em um problema estático, é a matriz que relaciona a força $\boldsymbol{f}$ aplicada na estrutura com o deslocamento resultante $\boldsymbol{u}$ (Przemieniecki, 1985 [30]; Soriano, 2005 [38]; Domício, 1977 [14]; Bittencourt, 2010 [8]; Bellman, 1997 [7]).

$$\boldsymbol{u} = \boldsymbol{K}^{-1}\boldsymbol{f} \;\Rightarrow\; \boldsymbol{u} = \boldsymbol{G}\boldsymbol{f} \;\Rightarrow\; \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_n \end{Bmatrix} = \begin{bmatrix} G_{11} & G_{12} & G_{13} & \cdots & G_{1n} \\ G_{21} & G_{22} & G_{23} & \cdots & G_{2n} \\ G_{31} & G_{32} & G_{33} & \cdots & G_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ G_{n1} & G_{n2} & G_{n3} & \cdots & G_{nn} \end{bmatrix} \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{Bmatrix} \tag{2.1}$$

Com base na Equação (2.1) a *i-ésima* coluna de $\boldsymbol{G}$ representa o deslocamento da estrutura quando uma força unitária é aplicada em seu *i-ésimo* GDL. Para verificar esta condição, basta considerar-se a *i-ésima* entrada do vetor $\boldsymbol{f}$ igual a 1 e as demais iguais a 0.

[1] Muitos conceitos da Análise Matricial de Estruturas formulados no Método dos Deslocamentos surgem também na formulação do Método dos Elementos Finitos (Soriano, 2005 [38]).

O que foi afirmado acima tem como base o Teorema 2.1.1 a seguir (Mascia, 2017 [23]):

Teorema 2.1.1 (da Reciprocidade ou de Maxwell[2]). *O deslocamento do ponto i , provocado por uma força externa unitária aplicada no ponto j, é igual ao deslocamento no ponto j, provocado por uma força externa unitária aplicada no ponto i (ver Figura 2.1).*

$$u_{ij} = u_{ji} \qquad i, j = 1, 2, \ldots, n \tag{2.2}$$

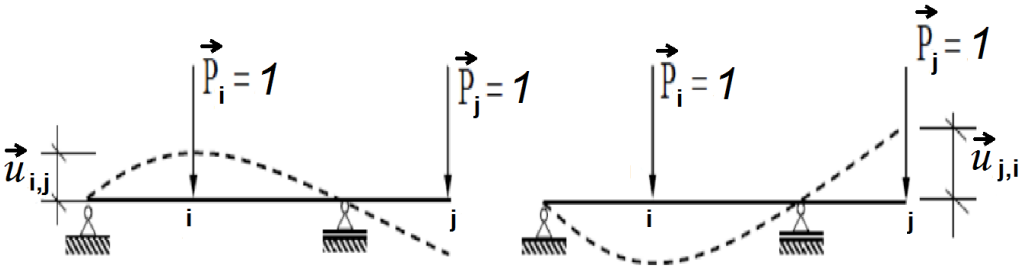

Figura 2.1: Ilustração do Teorema de Maxwell. Fonte: sinop.unemat.br (Acessado em 01/2018)

Demonstração. Seja uma viga submetida a duas forças externas, conforme Figura 2.2.

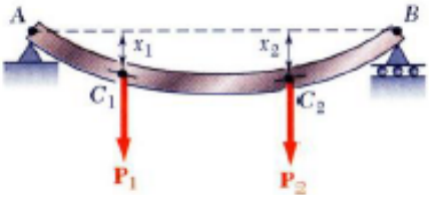

Figura 2.2: Ação de duas forças externas em uma viga. Fonte: sinop.unemat.br (Acessado em 07/2018)

Inicialmente, considera-se somente a atuação de uma força externa sendo aplicada no ponto C_1, cujo valor varia gradualmente de zero a P_1 (Figura 2.3).

A aplicação desta força gera deformação na viga e provoca os deslocamentos u_{11} e u_{21}. Tais deslocamentos são proporcionais entre si, podendo ser escritos do seguinte modo:

$$\begin{cases} u_{11} &= a_{11}P_1 \\ u_{21} &= a_{21}P_1 \end{cases} \tag{2.3}$$

[2] James Clerk Maxwell foi um físico e matemático escocês. Nasceu em 1831 em Edimburgo, na Escócia e faleceu em 1879 em Cambridge, na Inglaterra (Oberziner, 2008 [28]).

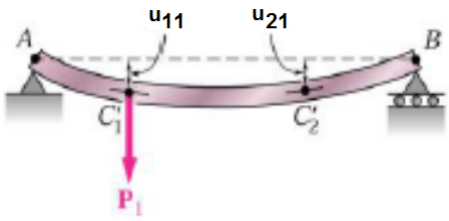

Figura 2.3: Ação de P_1. Fonte: sinop.unemat.br (Acessado em 07/2018)

Assim, conforme o Teorema de Clapeyron[3], tem-se:

$$U_{i,P_1}^{C_1} = \frac{1}{2} P_1 u_{11} \tag{2.4}$$

Substituindo a primeira equação de (2.3) na Equação (2.4):

$$U_{i,P_1}^{C_1} = \frac{1}{2} a_{11} P_1^2 \tag{2.5}$$

Mantendo-se a aplicação da carga P_1 e considerando-se o sistema da Figura 2.3 em equilíbrio na posição deformada (ver Figura 2.5 - linha tracejada), aplica-se a força externa P_2, gradualmente, no ponto C_2 (ver Figura 2.4).

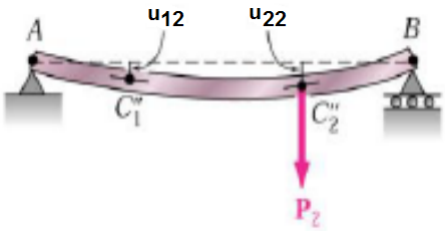

Figura 2.4: Ação de P_2. Fonte: sinop.unemat.br (Acessado em 07/2018)

A aplicação da força P_2 deforma novamente a viga (ver Figura 2.5 - linha contínua), provocando os deslocamentos u_{12} e u_{22}.

Tais deslocamentos são proporcionais entre si, podendo ser escritos da seguinte forma:

$$\begin{cases} u_{12} &= a_{12} P_2 \\ u_{22} &= a_{22} P_2 \end{cases} \tag{2.6}$$

Conforme o Teorema de Clapeyron, tem-se:

[3] A energia de deformação de uma estrutura, solicitada por diversos esforços externos P_i, é igual à metade da soma dos produtos dos valores finais de cada esforço pelo deslocamento de seu ponto de aplicação, medido na direção e sentido do esforço considerado

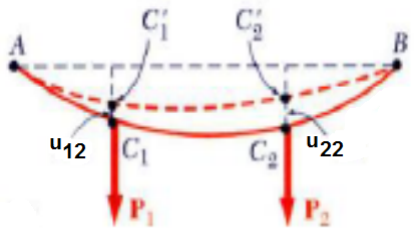

Figura 2.5: Viga deformada sob ação das forças P_1 e P_2. Fonte: sinop.unemat.br (Acessado em 07/2018)

$$U_{i,P_2}^{C_2} = \frac{1}{2} P_2 u_{22} \tag{2.7}$$

Substituindo a segunda equação de (2.6) na Equação (2.7):

$$U_{i,P_2}^{C_2} = \frac{1}{2} a_{22} P_2^2 \tag{2.8}$$

No ponto C_1, devido o deslocamento u_{12} causado pela aplicação da força P_2, gera o seguinte valor de energia de deformação (Teorema de Clapeyron):

$$U_{i,P_2/P_2}^{C_1} = \frac{1}{2} P_1 u_{12} \tag{2.9}$$

Substituindo a primeira equação de (2.6) na Equação (2.9):

$$U_{i,P_2/P_2}^{C_1} = \frac{1}{2} a_{12} P_1 P_2 \tag{2.10}$$

Logo, a partir da superposição de efeitos, tem-se a energia de deformação total no elemento, a partir da soma das expressões apresentadas nas equações (2.5), (2.8) e (2.10):

$$U_{i,total} = \frac{1}{2} a_{11} P_1^2 + \frac{1}{2} a_{22} P_2^2 + \frac{1}{2} a_{12} P_1 P_2 \tag{2.11}$$

Procedendo de forma inversa, ou seja, aplicando primeiramente a força P_2 e, em seguida, a Força P_1, obtém-se:

$$U_{i,total} = \frac{1}{2} a_{11} P_1^2 + \frac{1}{2} a_{22} P_2^2 + \frac{1}{2} a_{21} P_1 P_2 \tag{2.12}$$

Como a energia de deformação apresentada nas equações (2.11) e (2.12) são iguais, chega-se a seguinte conclusão:

$$a_{12} = a_{21} \tag{2.13}$$

□

Foi demonstrado, através do Teorema 2.1.1, que a matriz de flexibilidade é simétrica, ou seja

$$G_{ij} = G_{ji} \qquad i, j = 1, 2, \ldots, n \tag{2.14}$$

Esse significado físico dos componentes da matriz de flexibilidade (Equação (2.14)) representa uma forte motivação para sua utilização na identificação de danos estuturais, pois o comportamento estático da estrutura é modificado com a alteração das propriedades físicas por causa da presença de danos.

2.2 Análise Modal

Vibração ou oscilação é qualquer movimento que se repita após um intervalo de tempo. A teoria de vibração trata do estudo de movimentos oscilatórios de corpos e as forças associadas a eles. Se nenhuma energia for perdida ou dissipada por atrito ou outra resistência durante a oscilação, a vibração é conhecida como vibração não amortecida. Se um sistema, após uma perturbação inicial, continuar a vibrar por conta própria, a vibração resultante é conhecida como vibração livre.

Análise modal é o estudo das propriedades dinâmicas de uma estrutura sob excitação por vibrações. Na análise modal procura-se determinar os parâmetros modais de uma estrutura, ou seja, as frequências naturais e formas modais da estrutura. As formas modais são formas assumidas pela estrututra em cada uma das frequências naturais, essas também são chamadas de modos normais de vibração ou modos naturais de vibração. Os modos de vibrar são propriedades inerentes de uma estrutura e são determinados pelas propriedades do material (massa, rigidez e amortecimento) e pelas condições de contorno da estrutura. Cada modo de vibrar é definido por uma frequência natural, amortecimento modal e formas modais, ou seja, pelos parâmetros modais, citados acima. Se as propriedades do material ou as condições de contorno de uma estrutura mudam, esses modos irão mudar.

Com base na teoria de vibração e na análise modal, a matriz de flexibilidade $\mathbf{G}$ poderá ser obtida a partir das frequências naturais e formas modais não amortecidas de

uma estrutura vibrando livremente (Fitzpatrick, 2013 [15]; Rao, 2009 [31]; Sobrinho, 2006 [37]).

As equações de equilíbrio dinâmico (ou equações do movimento) para um sistema com n GDL podem ser escritas como:

$$\boldsymbol{M\ddot{u} + C\dot{u} + Ku = f(t)} \tag{2.15}$$

onde

- $\boldsymbol{M}$ é a matriz de massa da estrutura relativa aos n GDL considerados e de dimensão $n \times n$;
- $\boldsymbol{C}$ é a matriz de amortecimento da estrutura de dimensão $n \times n$;
- $\boldsymbol{K}$ é a matriz de rigidez da estrutura de dimensão $n \times n$;
- $\boldsymbol{\ddot{u}}$ é o vetor de acelerações nodais de dimensão $n \times 1$;
- $\boldsymbol{\dot{u}}$ é o vetor de velocidades nodais de dimensão $n \times 1$;
- $\boldsymbol{u}$ é o vetor de deslocamentos nodais de dimensão $n \times 1$;
- $\boldsymbol{f(t)}$ é o vetor de forças nodais definido em função do tempo de dimensão $n \times 1$, ou seja, são as forças de excitação da estrutura.

Desprezou-se o efeito do amortecimento na estrutura, assim a Equação (2.15) pode ser escrita em uma forma mais simplificada:

$$\boldsymbol{M\ddot{u} + Ku = 0} \tag{2.16}$$

Admite-se que o movimento da estrutura quando vibra com uma dada frequência ω é do tipo harmônico dado por uma equação do tipo:

$$\boldsymbol{u} = \boldsymbol{\phi_i} \cos(\omega_i t - \varphi) \tag{2.17}$$

onde

- $\boldsymbol{\phi_i}$ é a *i-ésima* forma modal da estrutura;

- ω_i é a *i-ésima* frequência natural de vibração da estrutura;
- φ é um ângulo de fase arbitrário;

Derivando duas vezes a Equação (2.17) em relação ao tempo t obtém-se as equações das acelerações ao longo do tempo:

$$\dot{\boldsymbol{u}} = -\omega_i \boldsymbol{\phi_i} \text{sen}(\omega_i t - \varphi) \tag{2.18}$$

$$\ddot{\boldsymbol{u}} = -\omega_i^2 \boldsymbol{\phi_i} \cos(\omega_i t - \varphi) \tag{2.19}$$

Substituindo as Equações (2.17) e (2.19) na Equação do movimento (2.16) tem-se:

$$\boldsymbol{M}(-\omega_i^2 \boldsymbol{\phi_i} \cos(\omega_i t - \varphi)) + \boldsymbol{K}(\boldsymbol{\phi_i} \cos(\omega_i t - \varphi)) = \boldsymbol{0} \tag{2.20}$$

$$\boldsymbol{M}(-\omega_i^2 \boldsymbol{\phi_i}) + \boldsymbol{K}(\boldsymbol{\phi_i}) = \boldsymbol{0} \tag{2.21}$$

A partir da Equação (2.21) chega-se à Equação (2.22), que representa o modelo numérico de elementos finitos para vibrações livres não amortecidas.

$$\left(\boldsymbol{K} - \omega_i^2 \boldsymbol{M}\right) \boldsymbol{\phi_i} = \boldsymbol{0} \tag{2.22}$$

Para que o sistema de Equações (2.22) tenha uma solução não trivial (e esta seria $\boldsymbol{\phi_i}$=$\boldsymbol{0}$), é necessário que se anule o determinante da matriz $[\boldsymbol{K} - \omega_i^2 \boldsymbol{M}]$. Logo, a determinação de frequências e modos de vibração resulta num problema de determinação de autovalores e autovetores, em que os autovalores representam as frequências naturais e os autovetores os modos de vibração. Assim, a cada frequência ω_i corresponde um modo de vibração $\boldsymbol{\phi_i}$.

O objetivo, agora, é conseguir uma forma geral para a Equação (2.22). Reescrevendo esta equação tem-se:

$$\boldsymbol{K}\boldsymbol{\phi_i} - \omega_i^2 \boldsymbol{M}\boldsymbol{\phi_i} = \boldsymbol{0} \tag{2.23}$$

$$\boldsymbol{K}\boldsymbol{\phi_i} = \omega_i^2 \boldsymbol{M}\boldsymbol{\phi_i} \tag{2.24}$$

É habitual representar os modos de vibração (ou formas modais) através de uma determinada norma que facilite a interpretação e a comparação entre eles. A forma de normalização dos modos de vibração mais usada, é a normalização em relação à matriz de massa (Guerreiro, 1999 [16]; Rao, 2009 [31]; Kelly,2012 [21]; Pain, 2005 [29]; Fitzpatrick, 2013 [15]; Sobrinho, 2006 [37]; Soriano, 2009 [39]). Logo, normalizando as formas modais da estrutura em relação à matriz de massa tem-se:

$$\boldsymbol{\Phi^T M \Phi} = \mathbf{I} \tag{2.25}$$

Adapta-se a Equação (2.24), logo após, multiplicá-se esta adaptação à esquerda por $\boldsymbol{\Phi^T}$ e utilizando a Equação (2.25) chega-se a:

$$\boldsymbol{\Phi^T K \Phi} = \boldsymbol{\Lambda} \tag{2.26}$$

Onde $\boldsymbol{\Phi}$ é a matriz modal da estrutura, de dimensão $n \times n$ (Equação (2.27)), cuja *i-ésima* coluna corresponde a um modo de vibração, ou seja, à forma modal ϕ_i.

$$\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_i, \cdots, \boldsymbol{\phi}_n]^T \quad \Rightarrow \quad \boldsymbol{\Phi} = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1i} & \cdots & \phi_{1n} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2i} & \cdots & \phi_{2n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{ni} & \cdots & \phi_{nn} \end{bmatrix}^T \tag{2.27}$$

Tem-se também que $\boldsymbol{\Lambda}$ é uma matriz diagonal, de dimensão $n \times n$ (Equação (2.28)), formada pelos valores quadráticos das frequências naturais, ou seja, $\lambda_{ii} = \omega_i^2$.

$$\boldsymbol{\Lambda} = \begin{bmatrix} \omega_1^2 & 0 & \cdots & 0 \\ 0 & \omega_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \omega_n^2 \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{\Lambda} = \begin{bmatrix} \lambda_{11} & 0 & \cdots & 0 \\ 0 & \lambda_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{nn} \end{bmatrix} \tag{2.28}$$

$\mathbf{I}$ é a matriz identidade e T representa a transposição de uma matriz.

Com base na Equação (2.26), a matriz de rigidez pode ser expressa como:

$$\boldsymbol{\Phi}^{-\boldsymbol{T}}\boldsymbol{\Phi}^{\boldsymbol{T}}\mathbf{K}\boldsymbol{\Phi}=\boldsymbol{\Phi}^{-\boldsymbol{T}}\boldsymbol{\Lambda} \tag{2.29}$$

$$\mathbf{K}\boldsymbol{\Phi} = \boldsymbol{\Phi}^{-\boldsymbol{T}}\boldsymbol{\Lambda} \tag{2.30}$$

$$\boldsymbol{K}\boldsymbol{\Phi}\boldsymbol{\Phi}^{-\mathbf{1}} = \boldsymbol{\Phi}^{-\boldsymbol{T}}\boldsymbol{\Lambda}\boldsymbol{\Phi}^{-\mathbf{1}} \tag{2.31}$$

$$\mathbf{K} = \boldsymbol{\Phi}^{-\boldsymbol{T}}\boldsymbol{\Lambda}\boldsymbol{\Phi}^{-\mathbf{1}} \tag{2.32}$$

Partindo da Equação (2.32) e da definição de matriz de flexibilidade (Seção 2.1), tem-se:

$$\mathbf{G} = \boldsymbol{K}^{-\mathbf{1}} = \boldsymbol{\Phi}\boldsymbol{\Lambda}^{-\mathbf{1}}\boldsymbol{\Phi}^{\boldsymbol{T}} = \sum_{i=1}^{n} \frac{1}{\omega_i^2}\phi_i\phi_i^T \tag{2.33}$$

Na prática, por causa de limitações experimentais, nem todos os n modos de uma estrutura com n GDL serão medidos. Logo, tem-se a seguinte aproximação para a matriz de flexibilidade $\mathbf{G}$, denominada matriz de flexibilidade experimental da estrutura e denotada como $\mathbf{G}_{exp}$ (Stutz, 1999, 2005 [41] [42]; Corrêa, 2013 [10]):

$$\boldsymbol{G}_{exp} = \sum_{i=1}^{m} \frac{1}{\omega_{i,exp}^2}\phi_{i,exp}\phi_{i,exp}^T \tag{2.34}$$

onde:

- $m < n$ representa o número de modos experimentais obtidos ou medidos no teste de vibração, geralmente muito menor do que o número total de modos da estrutura;
- $\omega_{i,exp}$ é a *i-ésima* frequência natural não amortecida obtida experimentalmente;
- $\phi_{i,exp}$ é a *i-ésima* forma modal obtida experimentalmente.

A matriz de flexibilidade é preferível à matriz de rigidez, visto que, a matriz de flexibilidade experimental pode ser estimada facilmente e de forma precisa a partir de alguns modos de mais baixa frequência, que podem ser obtidos durante um teste de

vibração. Isso se deve à relação inversa da matriz de flexibilidade com o quadrado das frequências naturais, como é mostrado na Equação (2.33) e na Equação (2.34).

Da Equação (2.34) nota-se que a dimensão da matriz $\mathbf{G}_{exp}$ depende apenas do número m de GDL medidos no ensaio de vibrações. Logo, a matriz de flexibilidade experimental possui dimensão $m \times m$. Neste caso, a matriz de flexibilidade não possui uma relação inversa com nenhuma submatriz $m \times m$ da matriz de rigidez analítica. Para a obtenção de uma relação inversa entre a matriz de flexibilidade $m \times m$ e uma matriz $m \times m$ que contenha informações das propriedades de rigidez da estrutura, é necessário que a matriz de rigidez original seja reorganizada de forma diferente, com a reorganização sendo ditada pelos GDL que foram medidos no ensaio de vibrações (Stutz, 1999 [41]). Para que isto ocorra, a matriz de rigidez original deve ser particionada na forma:

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{mm} & \mathbf{K}_{mo} \\ \mathbf{K}_{mo}^T & \mathbf{K}_{oo} \end{bmatrix} \tag{2.35}$$

onde:

- m refere-se aos GDL medidos;
- o refere-se aos GDL omitidos.

Do mesmo modo, a matriz de flexibilidade particionada deve ser dada por:

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_{mm} & \mathbf{G}_{mo} \\ \mathbf{G}_{mo}^T & \mathbf{G}_{oo} \end{bmatrix} \tag{2.36}$$

Pode-se mostrar (Alvin *et tal.*, 1995 [2]) que a matriz de flexibilidade analítica $\mathbf{G}_{mm}$ relativa aos GDL medidos é igual à matriz inversa da matriz de rigidez reduzida, obtida pela redução estática de Guyan (1965) [17] para a dimensão dos GDL medidos, ou seja:

$$\mathbf{G}_{mm} = \left[\mathbf{K}_{mm} - \mathbf{K}_{mo}\mathbf{K}_{oo}^{-1}\mathbf{K}_{mo}^T \right]^{-1} \tag{2.37}$$

2.3 Redes Neurais Artificiais

O cérebro humano é um sistema de processamento de informação bastante complexo, não-linear e paralelo. Como nosso cérebro processa informações de uma maneira bastante

diferente de um computador digital convencional, e isto por causa dos bilhões de neurônios que lá estão, as pesquisas utilizando Redes Neurais Artificiais (RNA) tem crescido cada vez mais, na tentativa de replicar o funcionamento do cérebro artificialmente. O ser humano tenta imitar ou simular o funcionamento das redes neurais naturais. Em consequência, as RNAs possuem a capacidade de aquisição e manutenção do conhecimento (baseado em informações) e podem ser definidas como um conjunto de unidades de processamento, caracterizadas por neurônios artificiais, que são interligados por um grande número de interconexões (sinapses artificiais), sendo as mesmas representadas por vetores/matrizes de pesos sinápticos. As características mais relevantes envolvidas com aplicação de RNAs são: adaptação por experiência, capacidade de aprendizado e habilidade de generalização. Por isso, estão inseridas dentro da área conhecida como sistemas inteligentes (conexionistas) ou inteligência computacional (Haykin, 2007 [19]; Silva, 2010 [36]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]).

2.3.1 Resumo Histórico

A primeira publicação relacionada com a neurocomputação é de 1943, através de um artigo elaborado por McCulloch & Pitts denominado: *A logical calculus of the ideas immanent in nervous activity* [24]. Neste trabalho, os autores realizaram a primeira modelagem matemática inspirada no neurônio biológico, surgindo, assim, a primeira concepção de neurônio artificial (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]).

Em 1949, foi proposto o primeiro método de treinamento para redes neurais artificiais, que denominou-se de regra de aprendizado de Hebb. Esta foi baseada em hipóteses e observações de caráter neurofisiológico (Hebb, 1949 [20]; Silva, 2010 [36]).

Muitos outros pesquisadores continuaram o trabalho de desenvolver modelos matemáticos fundamentados no neurônio biológico. Entre as linhas de pesquisa que surgiram destaca-se o trabalho de Frank Rosenblatt que, entre 1957 e 1958, desenvolveu o primeiro neurocomputador, denominado *Mark I - Perceptron*, idealizando o modelo básico do *Perceptron* (Rosenblatt, 1958 [32]; Silva, 2010 [36]; Braga, 2007 [12])

O modelo do *Perceptron* despertou interesse devido à sua capacidade em reconhecer padrões simples. Widrow & Hoff (1960) desenvolveram um tipo de rede denominada *Adaline*, que é a abreviatura de *ADAptive LINear Element*. Posteriormente, propôs-se o *Madaline*, a *Adaline* múltipla, que é uma rede cujo aprendizado é fundamentado na denominada regra Delta, também conhecida como algoritmo de aprendizado *LMS*(*least*

mean square) (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]).

Após esses trabalhos pioneiros, muitos pesquisadores da época ficaram incentivados a realizar pesquisas relacionadas às redes neurais artificiais. Porém, em 1969, a neurocomputação sofreu um revés com a publicação do livro *Perceptrons: an introduction to computational geometry* por Minsky & Papert (2017 [26]). Estes autores demonstraram de forma enfática a limitação das redes neurais artificiais, constituídas de apenas uma única camada, como o *Perceptron* e a *Adaline*, em aprender o relacionamento entre as entradas e saídas de funções lógicas bem simples como o X_{OR} (ou-exclusivo). Ou seja, nesta publicação houve a demonstração da impossibilidade de as redes realizarem a correta classificação de padrões para classes não linearmente separáveis (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Yegnanarayana, 2009 [43]).

A partir de reflexos desta publicação, ocorreu um período em que pouquíssimas pesquisas eram desenvolvidas em redes neurais artificiais. Em 1982, Hopfield propôs redes recorrentes baseadas em funções de energia. Isto fez com que a área de redes neurais artificiais retomasse o destaque que possuía antes de 1969. Mas, somente no final dos anos 1980 é que os pesquisadores voltaram a ter significativo interesse nesta área. A retomada definitiva das pesquisas se deve a diversos fatores, tais como: o desenvolvimento de computadores com maior capacidade de processamento, a criação de algoritmos de otimização mais eficientes e robustos e as novas descobertas sobre o sistema nervoso biológico.Um dos destaques daquele período foi a publicação do livro de Rumelhart, Hinton e Willians, intitulado *Parallel distributed processing* (Rumelhart *et alii*, 1987 [33]), em que os autores desenvolveram um algoritmo que permitia ajustar os pesos em uma rede neural artificial com mais de uma camada, solucionando-se inclusive o antigo problema de aprendizado dos padrões da função lógica X_{OR} (ou-exclusivo). A proposição de tal algoritmo, denominado *backpropagation*, reascendeu e motivou definitivamente as pesquisas em redes neurais artificiais (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Yegnanarayana, 2009 [43]; Abraham, 2005 [1]).

2.3.2 Neurônios Biológicos

O processamento de informações no cérebro humano é regido por elementos processadores biológicos que operam em paralelo. A célula elementar do sistema nervoso cerebral é o neurônio (ver Figura 2.6). Sua função é basicamente conduzir impulsos (estímulos elé-

tricos advindos de reações físico-químicas) sob determinadas condições (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Abraham, 2005 [1]).

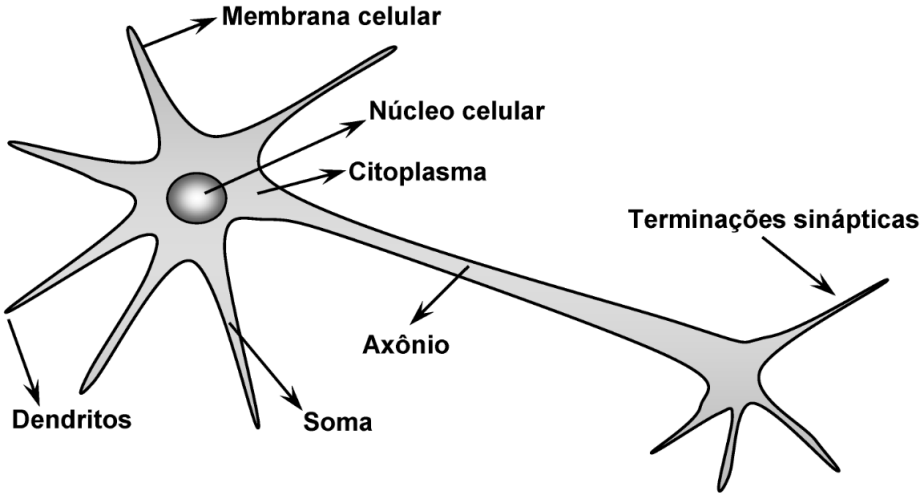

Figura 2.6: Esquema de um neurônio biológico. Fonte: Silva, 2010 [36]

O neurônio biológico pode ser dividido em três partes principais:

1. **Dendritos** → São constitídos por vários finos prolongamentos que formam a árvore dendrital (Haykin, 2007 [19]). Sua principal função é captar, de forma contínua, os estímulos provenientes de outros neurônios (Silva, 2010 [36]; Haykin, 2007 [19]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Abraham, 2005 [1]).

2. **Corpo Celular ou Soma** → Tem a função de processar todas as informações advindas dos dentritos a fim de produzir um potencial de ativação que indicará se o neurônio disparará um impulso elétrico ao longo do axônio (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Abraham, 2005 [1]).

3. **Axônio** → É constituído por um longo filamento. Trabalha conduzindo os impulsos elétricos para outros neurônios. O seu final também é constituído de ramificações denominadas terminações sinápticas (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]).

As conexões sinápticas viabilizam a transferência de impulsos elétricos do axônio de um neurônio para os dendritos de outros, como pode ser visto na Figura 2.7. Inexiste contato físico entre os neurônios na junção (fenda) sináptica, sendo que elementos neurotransmissores liberados são os responsáveis por ponderar a transmissão de impulsos

elétricos de um neurônio para o outro (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Abraham, 2005 [1]).

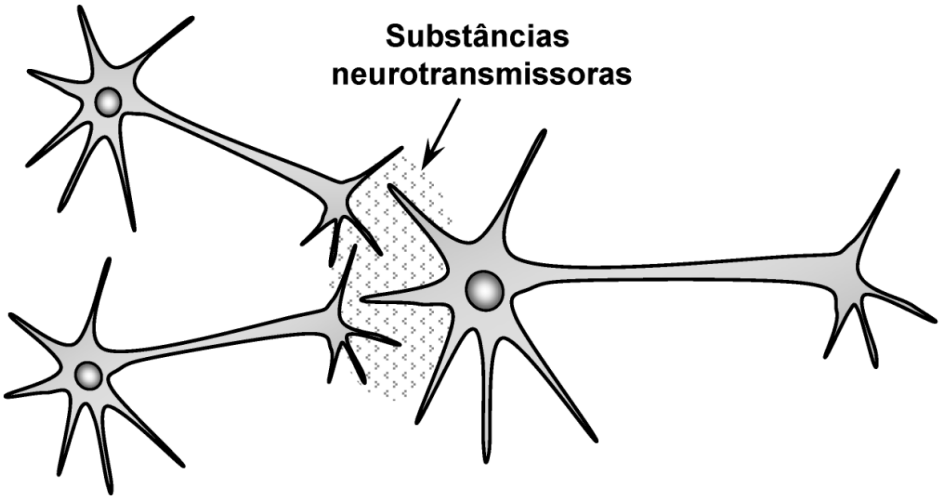

Figura 2.7: Conexões sinápticas entre neurônios. Fonte: Silva, 2010 [36].

2.3.3 Neurônios Artificiais

Os neurônios artificiais, são modelos bem simplificados dos neurônios biológicos. Tais modelos foram inspirados a partir da análise da geração e propagação de impulsos elétricos pela membrana celular dos neurônios (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]).

O modelo de neurônio mais simples e que engloba as principais características de uma rede neural biológica, isto é, paralelismo e alta conectividade, foi proposto por McCulloch & Pitts em 1943, sendo ainda o modelo mais utilizado nas diferentes arquiteturas de redes neurais artificiais. Nessa representação, cada neurônio da rede pode ser implementado conforme mostra a Figura 2.8. Os diversos sinais de entrada advindos do meio externo são representados pelo conjunto $\{x_1, x_2, \cdots, x_n\}$, que são análogos aos impulsos elétricos externos captados pelos dendritos no neurônio biológico. As ponderações exercidas pelas junções sinápticas do modelo biológico são representadas no neurônio artificial pelo conjunto de pesos sinápticos $\{w_1, w_2, \cdots, w_n\}$. De forma análoga, a relevância de cada uma das entradas $\{x_i\}$ do neurônio artificial é então executada por meio de suas multiplicações pelos respectivos pesos sinápticos $\{w_i\}$, ponderando-se, portanto, todas as informações externas que chegam ao neurônio. Assim, a saída do corpo celular artificial, denotado por u, é a soma ponderada de suas entradas (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Yegnanarayana, 2009 [43]; Abraham, 2005 [1]).

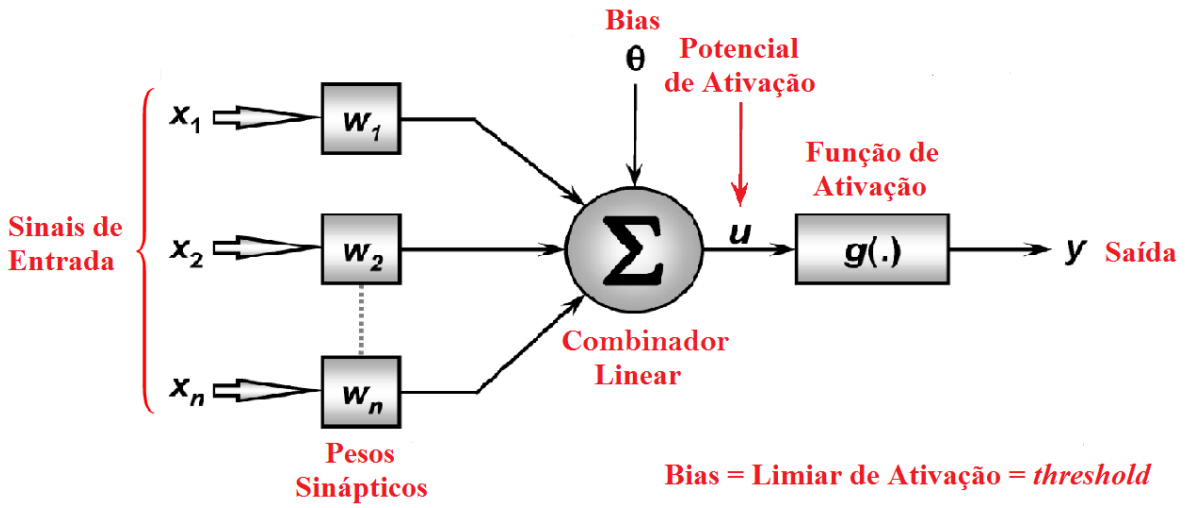

Figura 2.8: Neurônio artificial. Fonte: Adaptado de Silva, 2010 [36].

Considerando-se a Figura 2.8, observa-se que o neurônio artificial é constituído de sete elementos básicos, que são:

- **Sinais de entrada $\{x_1, x_2, \cdots, x_n\}$** $\rightarrow$ São os sinais ou medidas provenientes do meio externo e que representam os valores assumidos pelas variáveis de uma aplicação específica. Os sinais de entrada são usualmente normalizados visando incrementar a eficiência computacional dos algoritmos de aprendizagem;

- **Pesos sinápticos $\{w_1, w_2, \cdots, w_n\}$** $\rightarrow$ São os valores que servirão para ponderar cada uma das variáveis de entrada da rede, permitindo-se quantificar as suas relevâncias em relação à funcionalidade do respectivo neurônio;

- **Combinador linear Σ** $\rightarrow$ Sua função é agregar todos os sinais de entrada que foram ponderados pelos respectivos pesos sinápticos a fim de produzir um valor de potencial de ativação;

- **Limiar de ativação (ou Bias) θ** $\rightarrow$ É uma variável que especifica qual será o patamar apropriado para que o resultado produzido pelo combinador linear possa gerar um valor de disparo em direção à saída do neurônio;

- **Potencial de ativação u** $\rightarrow$ É o resultado produzido pela diferença do valor produzido entre o combinador linear e o limiar de ativação (Bias);

- **Função de ativação g** $\rightarrow$ Seu objetivo é limitar a saída do neurônio dentro de um intervalo de valores razoáveis a serem assumidos pela sua própria imagem funcional;

- **Sinal de saída y** $\rightarrow$ Consiste do valor final produzido pelo neurônio em relação a um determinado conjunto de sinais de entrada, podendo ser também utilizado por outros neurônios que estão sequencialmente interligados.

O neurônio artificial da Figura 2.8 em notação matemática fica da seguinte forma:

$$y = g(u) \quad \Rightarrow \quad y = g\left[\left(\sum_{i=1}^{n} w_i.x_i\right) - \theta\right] \tag{2.38}$$

onde

$$u = \left(\sum_{i=1}^{n} w_i.x_i\right) - \theta \tag{2.39}$$

2.3.4 Funções de Ativação

As funções de ativação podem ser divididas em dois grupos principais:

1. Funções parcialmente diferenciáveis $\rightarrow$ São aquelas que possuem pontos cujas derivadas de primeira ordem são inexistentes. As três principais funções de ativação desse grupo são (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]):

    - função degrau: O resultado fornecido pela aplicação desta função assumirá valores unitários positivos quando o potencial de ativação do neurônio artificial for maior ou igual a zero, caso contrário, o resultado assumirá valores nulos.

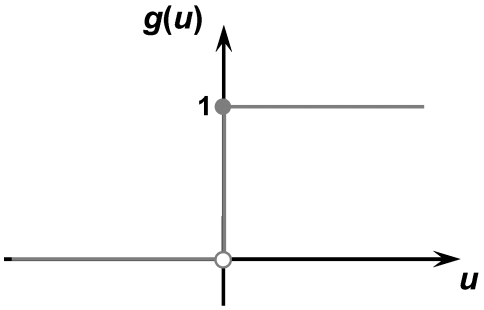

Figura 2.9: Gráfico da função degrau. Fonte: Silva, 2010 [36]

    - função degrau bipolar: O resultado fornecido pela aplicação desta função assumirá valores unitários positivos quando o potencial de ativação do neurônio

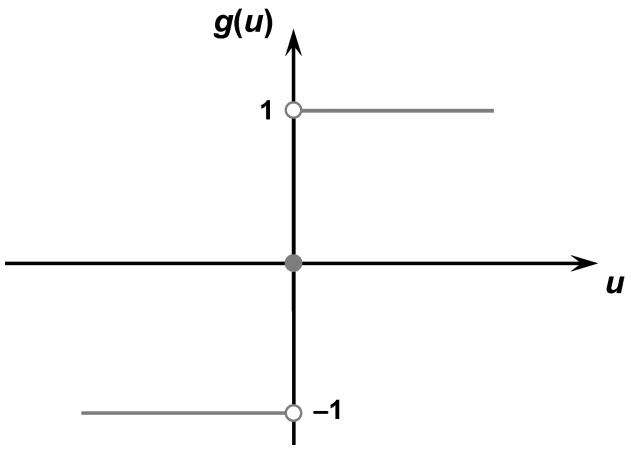

Figura 2.10: Gráfico da função degrau bipolar. Fonte: Silva, 2010 [36]

artificial for maior que zero; valor nulo quando o potencial de ativação também for nulo e valores unitários negativos quando o potencial for menor que zero.

- função rampa simétrica: Os resultados fornecidos pela aplicação desta função são iguais aos próprios valores dos potenciais de ativação quando estes estão definidos no intervalo $[-a, a]$, restringindo-se aos valores limites em caso contrário.

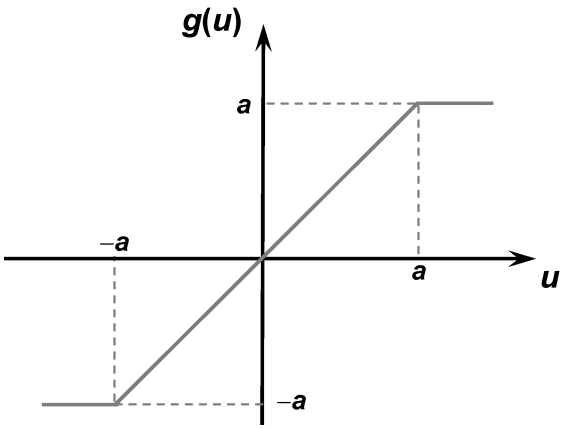

Figura 2.11: Gráfico da função rampa simétrica. Fonte: Silva, 2010 [36]

2. Funções totalmente diferenciáveis → São aquelas cujas derivadas de primeira ordem existem e são conhecidas em todos os pontos de seu domínio de definição. As quatro principais funções desse grupo, e que podem ser empregadas em redes neurais artificiais são (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]):

   - função logística: O resultado fornecido pela aplicação desta função assumirá sempre valores reais entre zero e um. θ é uma constante real associada ao nível de inclinação da função em relação ao seu ponto de inflexão.

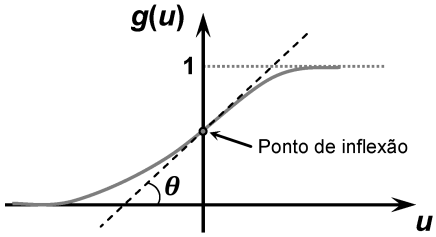

Figura 2.12: Gráfico da função logística. Fonte: Silva, 2010 [36]

- função tangente hiperbólica: O resultado fornecido pela aplicação desta função assumirá sempre valores reais entre -1 e 1. θ é uma constante real que também está associada ao nível de inclinação da função em relação ao seu ponto de inflexão.

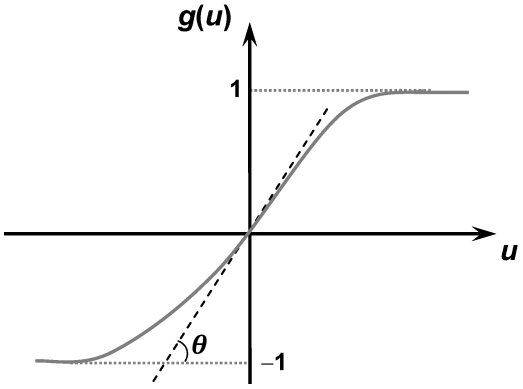

Figura 2.13: Gráfico da função tangente hiperbólica. Fonte: Silva, 2010 [36]

- função gaussiana: O resultado fornecido pela aplicação desta função assumirá valores iguais para aqueles valores de potencial de ativação que estejam posicionados a uma mesma distância de seu centro (média), sendo que a curva é simétrica em relação a este. c é um parâmetro que define o centro desta função e σ denota o desvio padrão relacionado à mesma.

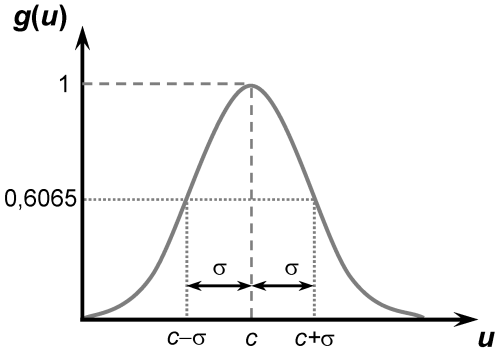

Figura 2.14: Gráfico da função gaussiana. Fonte: Silva, 2010 [36]

- função linear: O resultado fornecido pela aplicação desta função assumirá sempre valores de saída idênticos aos valores do potencial de ativação.

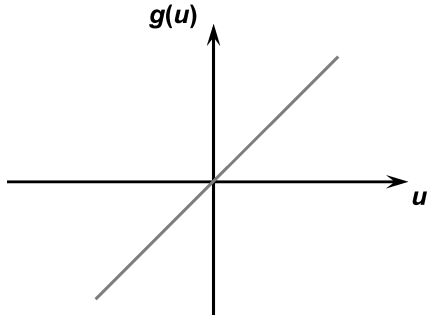

Figura 2.15: Gráfico da função linear. Fonte: Silva, 2010 [36]

2.3.5 Arquiteturas de Redes Neurais Artificiais

A arquitetura de uma RNA define a forma como os seus diversos neurônios estão arranjados uns em relação aos outros. Fazem parte da definição de arquitetura os seguintes parâmetros:

- números de camadas da rede;
- número de neurônios artificiais em cada camada;
- tipo de conexão entre os neurônios.

Existem quatro arquiteturas principais:

2.3.5.1 Arquitetura *feedforward* de Camadas Simples

Este tipo de arquitetura de RNAs possui apenas uma camada de entrada e uma única camada de neurônios, que é a própria camada de saída. A Figura 2.16 mostra uma rede *feedforward* de camada simples formada por n entradas e m saídas. Por meio da Figura 2.16, percebe-se que a quantidade de saídas nas redes pertencentes a esta arquitetura sempre coincidirá com o número de neurônios. O fluxo de informações segue sempre uma única direção (unidirecional), ou seja, da camada de entrada em direção à camada de saída. Entre os principais tipos de redes desta arquitetura estão o *Percepton* e o *Adaline* (Silva, 2010 [36]; Hagan, 1996 [18]; Schalkoff, 1997 [35]).

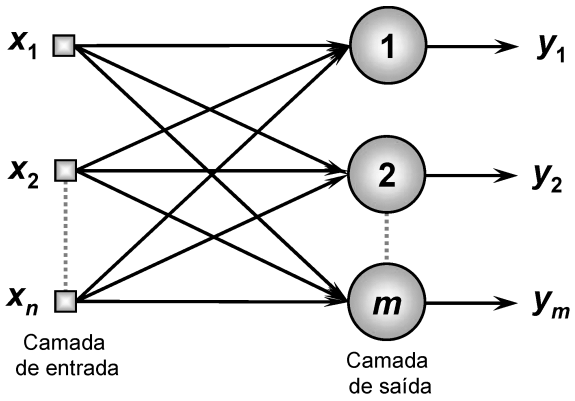

Figura 2.16: Arquitetura *feedforward* de camada simples. Fonte: Silva, 2010 [36].

2.3.5.2 Arquitetura *feedforward* de Camadas Múltiplas

As redes *feedforward* de camadas múltiplas (ou multicamadas) são constituídas por uma ou mais camadas escondidas de neurônios artificiais. A Figura 2.17 mostra uma rede *feedforward* de camadas múltiplas formada por uma camada de entrada com n sinais, duas camadas neurais escondidas constituídas respectivamente de n_1 e n_2 neurônios e uma camada neural de saída composta de m neurônios. Entre os principais tipos de redes desta arquitetura se encontram o *Percepton* multicamadas e as redes de base radial. A quantidade de camadas escondidas e seus respectivos neurônios constituintes dependem do tipo e da complexidade do problema a ser mapeado pela rede (Silva, 2010 [36]; Haykin, 2007 [19]; Yegnanarayana, 2009 [43]).

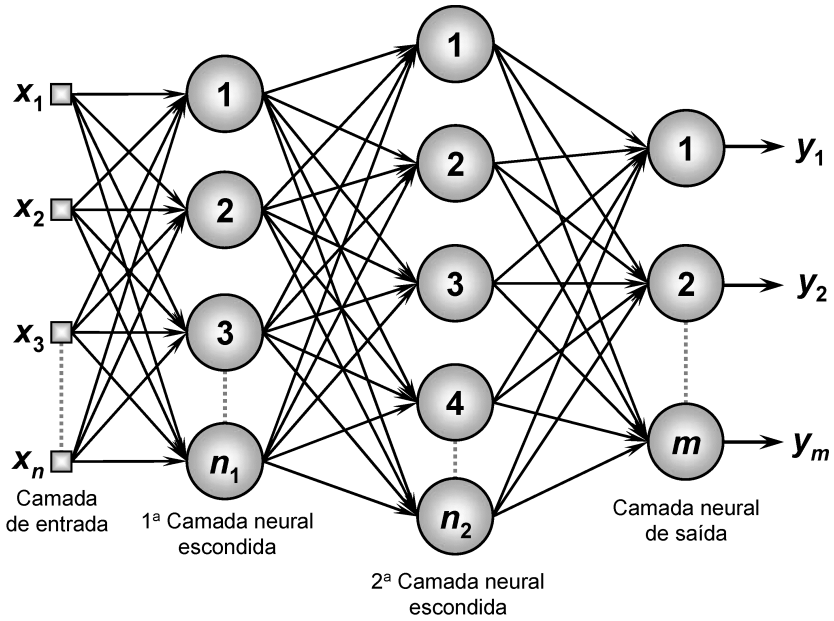

Figura 2.17: Arquitetura *feedforward* de camadas múltiplas. Fonte: Silva, 2010 [36].

2.3.5.3 Arquitetura Recorrente ou Realimentada

São redes em que as saídas dos neurônios são reutilizadas como sinais de entrada para outros neurônios. Entre os principais tipos de redes desta arquitetura estão a rede de Hopfield e a rede *Perceptron* multicamadas com realimentação entre neurônios situados em camadas distintas (Silva, 2010 [36]; Braga, 2007 [12]; Hagan, 1996 [18]).

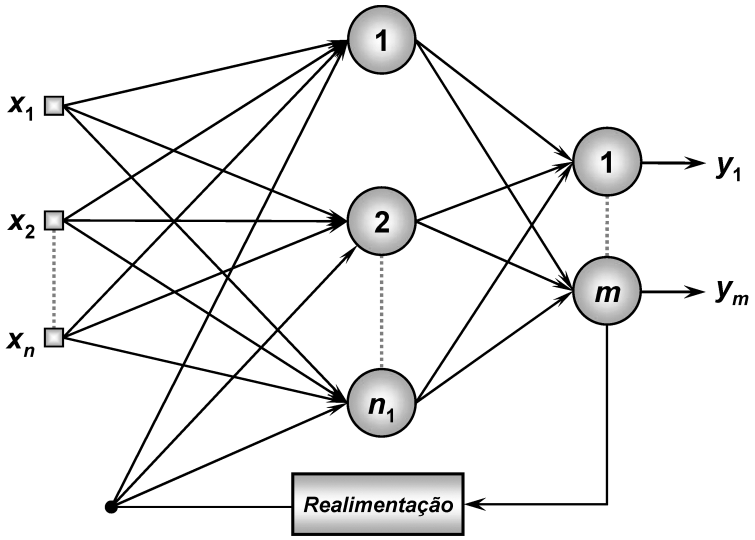

Figura 2.18: Arquitetura recorrente ou realimentada. Fonte: Silva, 2010 [36].

2.3.5.4 Arquitetura em Estrutura Reticulada

As principais características de redes com estruturas reticuladas estão na consideração da disposição espacial dos neurônios artificiais visando propósitos de extração de características, ou seja, a localização espacial dos neurônios está diretamente relacionada com o processo de ajuste de seus pesos e limiares (Figura 2.19) (Silva, 2010 [36]; Haykin, 2007 [19]).

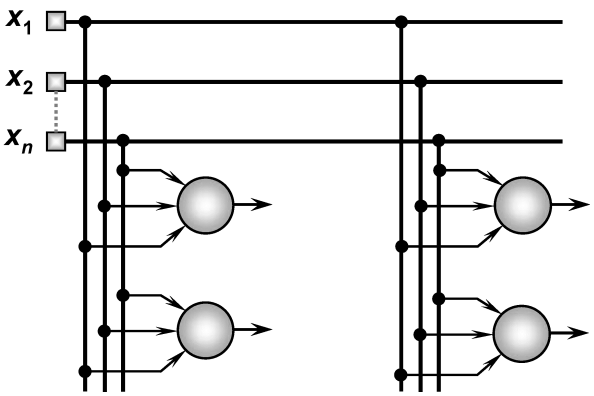

Figura 2.19: Arquitetura em estrutura reticulada. Fonte: Silva, 2010 [36].

2.3.6 Topologias de Redes Neurais Artificiais

A topologia de uma RNA pode ser definida como sendo as diferentes formas de composições estruturais que esta poderá assumir dentro de uma determinada arquitetura. Dentro de uma mesma arquitetura, pode-se variar a quantidade de neurônios e/ou as funções de ativação (Silva, 2010 [36]).

2.3.7 Tipos de Treinamento em uma Rede Neural Artificial

Um dos destaques das redes neurais artificiais está na capacidade de aprender a partir da apresentação de amostras (padrões) que expressem o comportamento do sistema, sendo que, em seguida, após a rede ter aprendido o relacionamento entre entradas e saídas, esta é capaz de generalizar soluções. A rede será então capaz de produzir uma saída próxima daquela esperada (desejada) a partir de quaisquer sinais inseridos em suas entradas. Através de um treinamento a rede neural artificial é ensinada. Utiliza-se para isso um algoritmo denominado algoritmo de aprendizagem. Geralmente, o conjunto total das amostras disponíveis sobre o comportamento do sistema é dividido em dois subconjuntos, os quais são denominados de subconjuntos de treinamento e subconjunto de teste. O subconjunto de treinamento, composto com cerca de 60% a 90% das amostras do conjunto total, será usado no processo de aprendizado da rede. Já o subconjunto de teste, cuja composição está entre 10% e 40% do conjunto total de amostras, será utilizado para verificar se os aspectos referentes à generalização de soluções por parte da rede já estão em patamares aceitáveis, possibilitando assim a validação da topologia escolhida. Durante o processo de treinamento de RNA, cada apresentação completa das amostras pertencentes ao subconjunto de treinamento, será denominada de época de treinamento (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Yegnanarayana, 2009 [43]). Tem-se dois processos principais para treinamento de uma RNA:

- **Treinamento Supervisionado** → A estratégia deste tipo de treinamento consiste em se ter disponível , considerando cada amostra dos sinais de entrada, as respectivas saídas desejadas, ou seja, cada amostra de treinamento é composta pelos sinais de entradas e suas correspondentes saídas. Neste caso há a necessidade de se disponibilizar uma tabela de dados (entradas/saídas) representativa do processo. Neste treinamento os parâmetros livres da rede (pesos sinápticos e limiares) são ajustados em função de se conhecer *a priori* quais são as saídas desejadas (Silva, 2010 [36]; Braga, 2007 [12]).

- **Treinamento Não-Supervisionado** → Durante a aplicação de um algoritmo de aprendizado basado neste treinamento inexistem as respectivas saídas desejadas. Daí, a própria rede deve se auto-organizar em relação às particularidades existentes entre os elementos componentes do conjunto total de amostras, identificando subconjuntos (*clusters*) que contenham similaridades. A quantidade máxima desses *clusters* pode ser especificada *a priori* pelo próprio projetista da rede (Silva, 2010 [36]; Braga, 2007 [12]).

2.3.8 Redes *Perceptron* de Múltiplas Camadas

As redes *Perceptron* de múltiplas camadas (PMC) se caracterizam pela presença de pelo menos uma camada escondida ou intermediária de neurônios, situada entre a camada de entrada e a camada neural de saída. Consequentemente, as redes *Perceptron* de múltiplas camadas possuem no mínimo duas camadas de neurônios: uma camada escondida e uma camada de saída. As redes PMC pertencem à arquitetura do tipo *feedforward* de camadas múltiplas. O fluxo de informações na rede se inicia na camada de entrada, percorre em seguida as camadas escondidas, sendo então finalizado na camada neural de saída. Mostra-se um exemplo de uma rede PMC na Figura 2.20 (Silva, 2010 [36]; Haykin, 2007 [19]; Yegnanarayana, 2009 [43]).

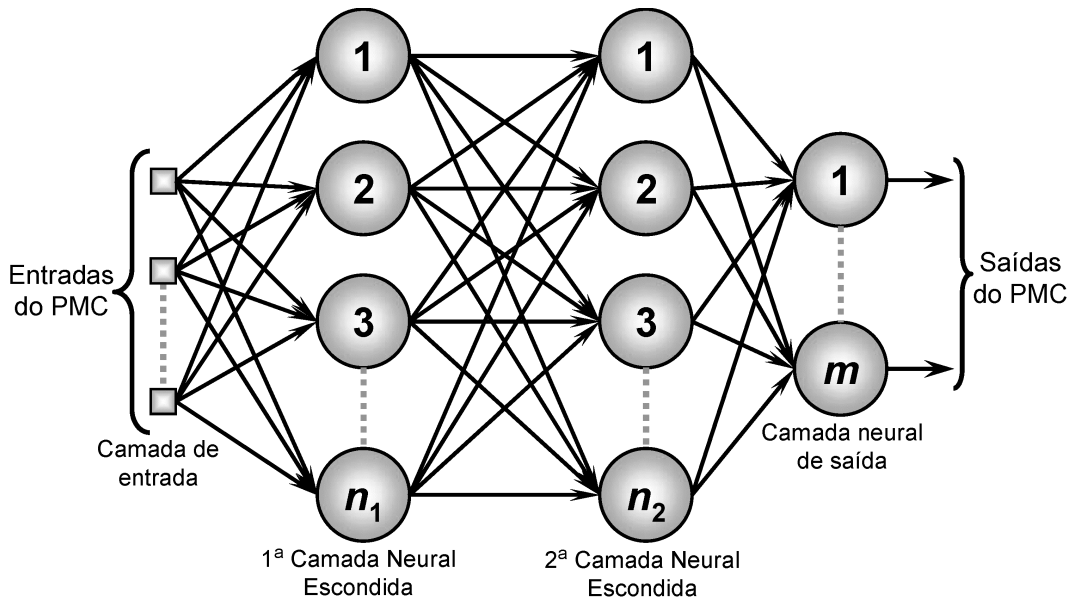

Figura 2.20: Exemplo de rede *Perceptron* multicamadas. Fonte: Silva, 2010 [36].

As redes PMC são ainda caracterizadas pelas elevadas possibilidades de aplicações em diversos tipos de problemas relacionados com as mais diferentes áreas do conhecimento. É considerada uma das arquiteturas mais versáteis quanto à aplicabilidade. Entre essas

áreas, destacam-se (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]):

- Aproximação universal de funções;
- Reconhecimento de padrões;
- Identificação e controle de processos;
- Previsão de séries temporais;
- Otimização de sistemas.

Em uma rede PMC o treinamento é efetivado de forma supervisionada. O início da grande popularidade e das extensas aplicabilidades das redes PMC se deram a partir do fim dos anos 1980, sendo atribuídos tais efeitos principalmente à publicação do livro *Parallel Distributed Processing* (Rumelhart, 1987), no qual foi consistentemente explicitado o algoritmo de aprendizagem denominado *backpropagation*, permitindo a sua implementação no processo de treinamento dessas redes (Silva, 2010 [36]; Haykin, 2007 [19]).

2.3.9 Princípio de Funcionamento de Redes *Perceptron* de Múltiplas Camadas

Por meio da Figura 2.20, observa-se que cada uma das entradas da rede, representando os sinais advindos de determinada aplicação, será propagada uma a uma em direção à camada neural de saída do PMC. Neste caso, as saídas dos neurônios da primeira camada neural escondida serão as entradas daqueles neurônios pertencentes à segunda camada neural escondida. As saídas dos neurônios da segunda camada neural escondida serão as entradas dos neurônios pertencentes à sua camada neural de saída. Assim, a propagação dos sinais de entrada da rede PMC, independentemente da quantidade de camadas escondidas, é sempre realizada num único sentido, ou seja, da camada de entrada em direção à camada neural de saída. O conhecimento relacionado ao comportamento entrada/saída do sistema será distribuído por todos os neurônios constituintes do PMC. Os estímulos ou sinais são apresentados à rede em sua camada de entrada. As camadas intermediárias extraem a maioria das informações referentes ao seu comportamento e as codificam por meio dos pesos sinápticos e limiares de seus neurônios, formando assim uma representação própria do ambiente em que está inserido o referido sistema a ser tratado. Finalmente, os neurônios da camada de saída recebem os estímulos advindos dos

neurônios da última camada intermediária, produzindo um padrão de resposta que será a saída disponibilizada pela rede (Silva, 2010 [36]; Haykin, 2007 [19]).

A especificação da configuração topológica de uma rede PMC, tais como a quantidade de camadas intermediárias e seus respectivos números de neurônios, depende de diversos fatores tais como a classe de problema a ser tratada pelo PMC e a disposição espacial das amostras de treinamento (Silva, 2010 [36]; Braga, 2007 [12]).

O algoritmo de aprendizado utilizado no decorrer do processo de treinamento de redes PMC é denominado *backpropagation* ou algoritmo de retropropagação do erro. O algoritmo é implementado mediante a aplicação sucessiva de duas fases bem definidas (Silva, 2010 [36]).

2.3.10 Processo de Treinamento de Redes *Perceptron* de Múltiplas Camadas

O processo de treinamento de redes PMC utilizando o algoritmo *backpropagation* é comumente realizado, conforme mencionado no final da Subseção 2.3.9, mediante as aplicações sucessivas de duas fases bem específicas. A ilustração de tais fases é mostrada na Figura 2.21, onde é considerada uma topologia de PMC constituída de duas camadas escondidas, contendo n sinais em sua camada de entrada, tendo ainda n_1 neurônios na primeira camada neural escondida, n_2 neurônios na segunda camada neural escondida e n_3 sinais associados à camada neural de saída (terceira camada neural) (Silva, 2010 [36]; Hagan, 1996 [18]; Schalkoff, 1997 [35]).

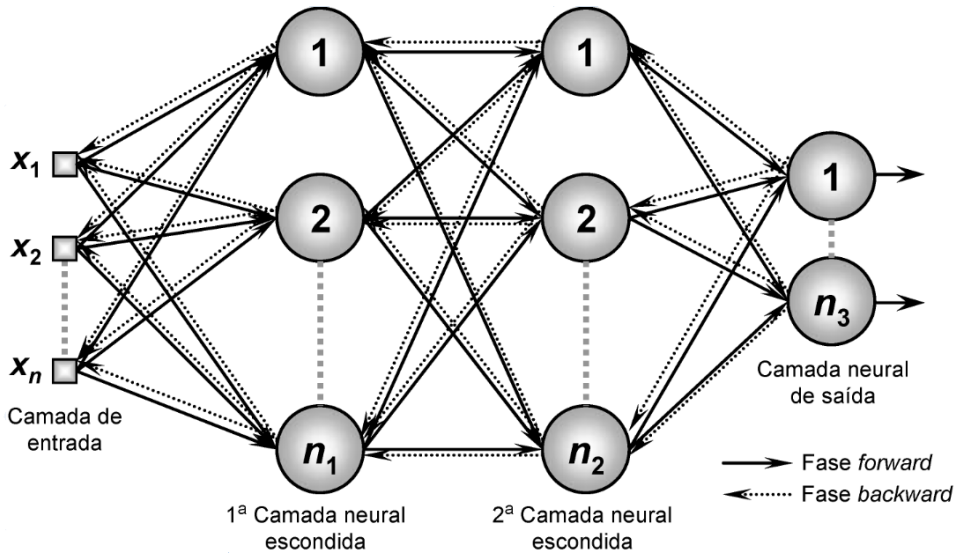

Figura 2.21: As duas fases de treinamento da rede PMC. Fonte: Silva, 2010 [36].

A primeira fase a ser aplicada é denominada de "propagação adiante"(*forward*), onde os sinais $\{x_1, x_2, \cdots, x_n\}$ de uma amostra do conjunto de treinamento são inseridos nas entradas da rede e são propagados camada a camada até a produção das respectivas saídas. Logo, a aplicação desta fase visa tão somente obter as respostas da rede, levando-se em consideração apenas valores atuais de pesos sinápticos e limiares de seus neurônios, os quais permanecerão inalterados durante cada execução desta fase. Em seguida, as respostas produzidas pelas saídas da rede são comparadas com as respectivas respostas desejadas, pois trata-se de um processo de aprendizado supervisionado. Considerando-se uma rede PMC, como a da Figura 2.21, constituída de n_3 neurônios em sua camada de saída, os respectivos n_3 desvios (erros) entre as respostas desejadas e aquelas produzidas pelos neurônios da camada de saída são então calculados, os quais serão posteriormente utilizados para ajustar os pesos e limiares de todos os seus neurônios. Desse modo, em função desses valores de erros, aplica-se, em seguida, a segunda fase do algoritmo *backpropagation*, denominada de "propagação reversa"(*backward*). Ao contrário da fase anterior, as alterações (ajustes) dos pesos sinápticos e limiares de todos os neurônios da rede são excutados no decorrer desta fase. Ou seja, as aplicações sucessivas das fases *forward* e *backward* fazem com que os pesos sinápticos e limiares dos neurônios se ajustem automaticamente em cada iteração, implicando-se na gradativa diminuição da soma dos erros produzidos pelas respostas da rede frente àquelas desejadas (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Yegnanarayana, 2009 [43]).

2.3.10.1 O Algoritmo *Backpropagation*

Para um melhor entendimento do princípio de funcionamento envolvido no algoritmo *backpropagation*, há a necessidade de se definir *a priori* diversas variáveis e parâmetros auxiliares que serão utilizados para tal propósito. Baseando-se na topologia de PMC da Figura 2.21, apresenta-se na Figura 2.22 um conjunto de variáveis que norteiam a derivação do algoritmo (Silva, 2010 [36]; Braga, 2007 [12]).

Cada um dos neurônios j pertencentes a uma das camadas L da topologia ilustrada na Figura 2.22 pode ser configurado conforme a terminologia abordada na Figura 2.23, onde $g(.)$ representa uma função de ativação que deve ser contínua e diferenciável em todo o seu domínio, tais como aquelas representadas pela função de ativação logística ou tangente hiperbólica (Silva, 2010 [36]).

A partir das Figuras 2.22 e 2.23, adota-se a seguinte terminologia para os seus parâmetros constituintes:

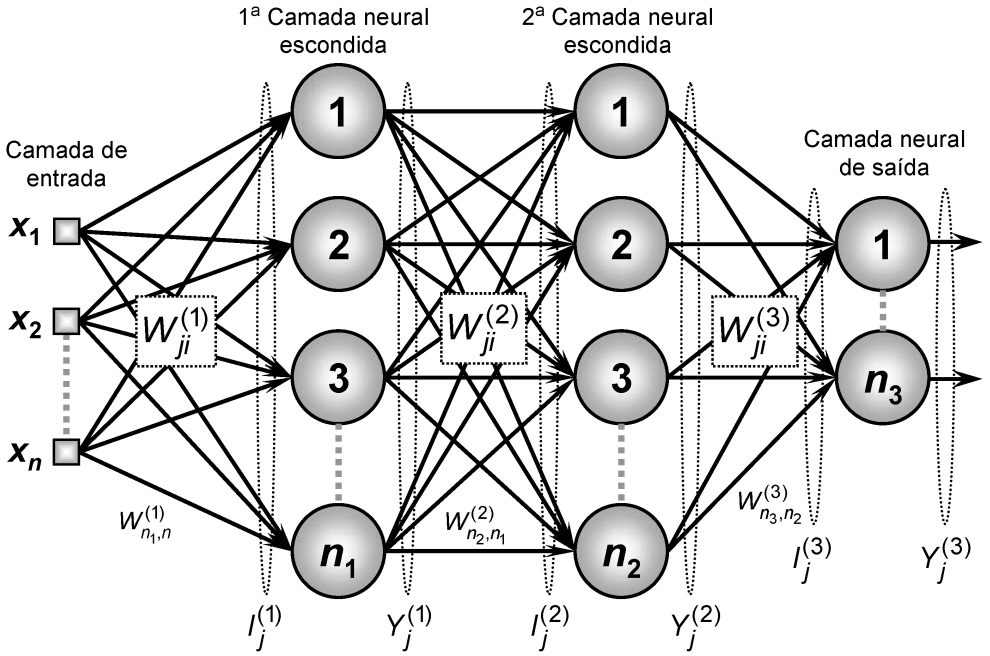

Figura 2.22: Notação para derivação do algoritmo *backpropagation*. Fonte: Silva, 2010 [36].

- $\boldsymbol{W}_{ji}^{(L)} \rightarrow$ São matrizes de pesos cujos elementos denotam o valor do peso sináptico conectando o j-ésimo neurônio da camada (L) ao i-ésimo neurônio da camada $L-1$. Para a topologia da Figura 2.22, tem-se:

  - $\boldsymbol{W}_{ji}^{(3)} \rightarrow$ É o peso sináptico conectando o j-ésimo neurônio da camada de saída ao i-ésimo neurônio da camada 2;

  - $\boldsymbol{W}_{ji}^{(2)} \rightarrow$ É o peso sináptico conectando o j-ésimo neurônio da camada escondida 2 ao i-ésimo neurônio da camada 1;

  - $\boldsymbol{W}_{ji}^{(1)} \rightarrow$ É o peso sináptico conectando o j-ésimo neurônio da camada 1 ao i-ésimo sinal da camada de entrada.

- $\boldsymbol{I}_j^{(L)} \rightarrow$ São vetores cujos elementos denotam a entrada ponderada em relação ao j-ésimo neurônio da camada L, os quais são definidos por

$$\boldsymbol{I}_j^{(1)} = \sum_{i=0}^{n} \boldsymbol{W}_{ji}^{(1)}.x_i \Leftrightarrow \boldsymbol{I}_j^{(1)} = \boldsymbol{W}_{j,0}^{(1)}.x_0 + \boldsymbol{W}_{j,1}^{(1)}.x_1 + \ldots + \boldsymbol{W}_{j,n}^{(1)}.x_n \tag{2.40}$$

$$\boldsymbol{I}_j^{(2)} = \sum_{i=0}^{n_1} \boldsymbol{W}_{ji}^{(2)}.\boldsymbol{Y}_i^{(1)} \Leftrightarrow \boldsymbol{I}_j^{(2)} = \boldsymbol{W}_{j,0}^{(2)}.\boldsymbol{Y}_0^{(1)} + \boldsymbol{W}_{j,1}^{(2)}\boldsymbol{Y}_1^{(1)} + \ldots + \boldsymbol{W}_{j,n_1}^{(2)}.\boldsymbol{Y}_{n_1}^{(1)} \tag{2.41}$$

$$\boldsymbol{I}_j^{(3)} = \sum_{i=0}^{n_2} \boldsymbol{W}_{ji}^{(3)}.\boldsymbol{Y}_i^{(2)} \Leftrightarrow \boldsymbol{I}_j^{(3)} = \boldsymbol{W}_{j,0}^{(3)}.\boldsymbol{Y}_0^{(2)} + \boldsymbol{W}_{j,1}^{(3)}\boldsymbol{Y}_1^{(2)} + \ldots + \boldsymbol{W}_{j,n_2}^{(3)}.\boldsymbol{Y}_{n_2}^{(2)} \tag{2.42}$$

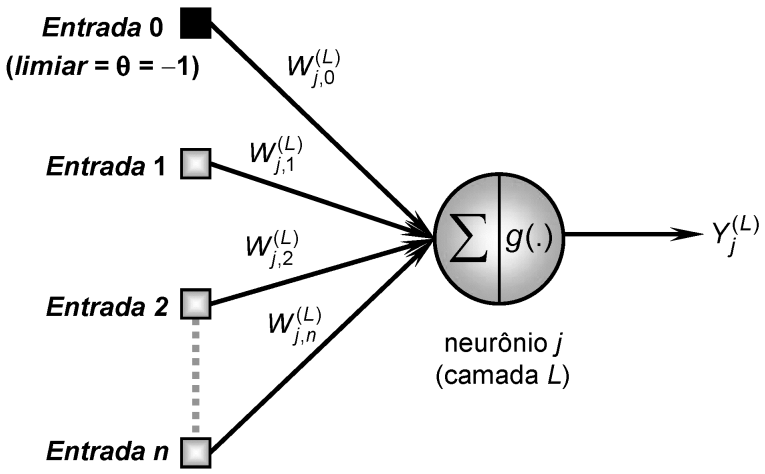

Figura 2.23: Configuração de neurônio utilizado na derivação do algoritmo *backpropagation*. Fonte: Silva, 2010 [36].

- $\boldsymbol{Y}_j^{(L)} \rightarrow$ São vetores cujos elementos denotam a saída do j-ésimo neurônio em relação à camada L, os quais são definidos por:

$$\boldsymbol{Y}_j^{(1)} = g\left(\boldsymbol{I}_j^{(1)}\right) \tag{2.43}$$

$$\boldsymbol{Y}_j^{(2)} = g\left(\boldsymbol{I}_j^{(2)}\right) \tag{2.44}$$

$$\boldsymbol{Y}_j^{(3)} = g\left(\boldsymbol{I}_j^{(3)}\right) \tag{2.45}$$

O próximo passo para o início da derivação do algoritmo *backpropagation* consiste em definir a função representativa do erro de aproximação, cuja incumbência será medir o desvio entre as respostas produzidas pelos neurônios de saída da rede em relação aos respectivos valores desejados. Assim, considerando a k-ésima amostra de treinamento para a topologia da Figura 2.22, assume-se a função erro quadrático como aquela a ser utilizada para medir o desempenho local associado aos resultados produzidos pelos neurônios de saída frente à referida amostra, ou seja:

$$\boldsymbol{E}(k) = \frac{1}{2}\sum_{j=1}^{n_3}\left(\boldsymbol{d}_j(k) - \boldsymbol{Y}_j^{(3)}(k)\right)^2 \tag{2.46}$$

onde $\boldsymbol{Y}_j^{(3)}(k)$ é o valor produzido pelo j-ésimo neurônio de saída da rede considerando-se a k-ésima amostra de treinamento, enquanto que $\boldsymbol{d}_j(k)$ é o seu respectivo valor desejado (Silva, 2010 [36]; Haykin, 2007 [19]).

Consequentemente, assumindo um conjunto de treinamento composto por p amostras, a medição da evolução do desempenho global do algoritmo *backpropagation* pode ser

efetuada por meio da avaliação do "erro quadrático médio", definido por:

$$\boldsymbol{E}_M = \frac{1}{p}\sum_{k=1}^{p}\boldsymbol{E}(k) \tag{2.47}$$

onde $\boldsymbol{E}(k)$ é o erro quadrático da Equação 2.46.

Para um melhor entendimento dos passos necessários para a compreensão do algoritmo *backpropagation*, divide-se a sua descrição em duas etapas. Far-se-á uso de método baseado no gradiente da função erro quadrático dada na Equação (2.46). Esta descrição do algoritmo *backpropagation* terá como base a Figura 2.22.

ETAPA I - AJUSTE DOS PESOS SINÁPTICOS DE CAMADA DE SAÍDA

O objetivo é ajustar a matriz de pesos $\boldsymbol{W}_{ji}^{(3)}$ a fim de minimizar o erro entre a saída produzida pela rede em relação à respectiva saída desejada. Neste caso, considerando-se o erro dado na Equação (2.46) em relação à k-ésima amostra de treinamento referente ao j-ésimo neurônio da camada de saída e empregando a definição de gradiente, como também a regra de diferenciação em cadeia, tem-se:

$$\nabla\boldsymbol{E}^{(3)} = \frac{\partial\boldsymbol{E}}{\partial\boldsymbol{W}_{ji}^{(3)}} = \frac{\partial\boldsymbol{E}}{\partial\boldsymbol{Y}_j^{(3)}}\cdot\frac{\partial\boldsymbol{Y}_j^{(3)}}{\partial\boldsymbol{I}_j^{(3)}}\cdot\frac{\partial\boldsymbol{I}_j^{(3)}}{\partial\boldsymbol{W}_{ji}^{(3)}} \tag{2.48}$$

Por intermédio das definições anteriores, tem-se:

$$\frac{\partial\boldsymbol{I}_j^{(3)}}{\partial\boldsymbol{W}_{ji}^{(3)}} = \boldsymbol{Y}_i^{(2)} \tag{2.49}$$

Equação (2.49) obtida a partir da Equação (2.42).

$$\frac{\partial\boldsymbol{Y}_j^{(3)}}{\partial\boldsymbol{I}_j^{(3)}} = g'\left(\boldsymbol{I}_j^{(3)}\right) \tag{2.50}$$

Equação (2.50) obtida a partir da Equação (2.45).

$$\frac{\partial\boldsymbol{E}}{\partial\boldsymbol{Y}_j^{(3)}} = -\left(\mathbf{d}_j - \mathbf{Y}_j^{(3)}\right) \tag{2.51}$$

Equação (2.51) obtida a partir da Equação (2.46).

onde $g'(.)$ denota a derivada de primeira ordem da função de ativação considerada.

Substituindo as equações (2.49), (2.50) e (2.51) na Equação (2.48), obtém-se:

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(3)}} = -\left(\mathbf{d}_j - \mathbf{Y}_j^{(3)}\right) \cdot g'\left(\boldsymbol{I}_j^{(3)}\right) \cdot \boldsymbol{Y}_i^{(2)} \tag{2.52}$$

Logo, o ajuste da matriz de pesos $\boldsymbol{W}_{ji}^{(3)}$ deve ser efetuado em direção oposta ao gradiente a fim de minimizar o erro, ou seja:

$$\Delta \boldsymbol{W}_{ji}^{(3)} = -\eta \cdot \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(3)}} \quad \Leftrightarrow \quad \Delta \boldsymbol{W}_{ji}^{(3)} = \eta \cdot \boldsymbol{\delta}_j^{(3)} \cdot \boldsymbol{Y}_i^{(2)} \tag{2.53}$$

onde $\boldsymbol{\delta}_j^{(3)}$ é definido como o gradiente local em relação ao j-ésimo neurônio da camada de saída, sendo o mesmo dado por:

$$\boldsymbol{\delta}_j^{(3)} = \left(\mathbf{d}_j - \mathbf{Y}_j^{(3)}\right) \cdot g'\left(\boldsymbol{I}_j^{(3)}\right) \tag{2.54}$$

A expressão (2.53) pode ser convertida no seguinte procedimento iterativo:

$$\boldsymbol{W}_{ji}^{(3)}(t+1) = \boldsymbol{W}_{ji}^{(3)}(t) + \eta \cdot \boldsymbol{\delta}_j^{(3)} \cdot \boldsymbol{Y}_i^{(2)} \tag{2.55}$$

onde η é a taxa de aprendizagem do algoritmo *backpropagation*. Em notação algorítmica, a expressão é equivalente à seguinte:

$$\boldsymbol{W}_{ji}^{(3)} \leftarrow \boldsymbol{W}_{ji}^{(3)} + \eta \cdot \boldsymbol{\delta}_j^{(3)} \cdot \boldsymbol{Y}_i^{(2)} \tag{2.56}$$

Portanto, a expressão (2.56) ajusta os pesos dos neurônios da camada de saída da rede levando-se em consideração a diferença observada entre as respostas produzidas por suas saídas em relação aos seus respectivos valores desejados.

ETAPA II - AJUSTE DOS PESOS SINÁPTICOS DE CAMADAS INTERMEDIÁRIAS

Ao contrário dos neurônios pertencentes à camada de saída do PMC, para os neurônios das camadas intermediárias não se tem acesso de forma direta aos valores desejados para as suas saídas. Nesta situação, os ajustes de seus pesos sinápticos são efetuados por intermédio de estimativas dos erros de saída produzidos por aqueles neurônios da camada imediatamente posterior, os quais já foram previamente ajustados (Silva, 2010 [36]).

Seguindo a sequência de ajustes impetrada pela fase *backward* para a topologia da Figura 2.22, somente após os ajustes dos neurônios da camada neural de saída é que se iniciam as correções dos pesos para aqueles da segunda camada intermediária. Nesta condição específica, indispõem-se de valores desejados para as saídas de tais neurônios, sendo então que seus ajustes serão baseados nos valores ponderados daqueles pesos sinápticos que já foram ajustados para os neurônios da camada de saída. É justamente neste aspecto que se encontra a essência do algoritmo de retropropagação do erro (*backpropagation*), pois, em primeira instância, tem-se ajustado os pesos sinápticos dos neurônios da camada de saída mediante valores verdadeiros dos desvios observados entre suas respostas produzidas e os respectivos valores desejados. Em segunda instância, retropropaga-se este erro para os neurônios das camadas anteriores, ponderando-se os mesmos pelos valores de pesos sinápticos que já foram previamente ajustados em todas as camadas posteriores. Logo, a resposta desejada para um neurônio de camada escondida deve ser então determinada em função dos neurônios (da camada imediatamente posterior) que estão diretamente conectados a este e que já foram previamente ajustados no passo anterior (Silva, 2010 [36]; Haykin, 2007 [19]; Schalkoff, 1997 [35]).

(A) Ajuste dos pesos sinápticos da segunda camada escondida:

O processo de treinamento para a segunda camada neural escondida consiste em ajustar a matriz de pesos $\boldsymbol{W}_{ji}^{(2)}$ a fim de minimizar o erro entre a saída produzida pela rede em relação à retropropagação do erro advindo dos ajustes dos neurônios da camada neural de saída. Assim, tem-se:

$$\nabla \boldsymbol{E}^{(2)} = \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(2)}} = \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_{j}^{(2)}} \cdot \frac{\partial \boldsymbol{Y}_{j}^{(2)}}{\partial \boldsymbol{I}_{j}^{(2)}} \cdot \frac{\partial \boldsymbol{I}_{j}^{(2)}}{\partial \boldsymbol{W}_{ji}^{(2)}} \tag{2.57}$$

Por intermédio das definições anteriores, tem-se:

$$\frac{\partial \boldsymbol{I}_{j}^{(2)}}{\partial \boldsymbol{W}_{ji}^{(2)}} = \boldsymbol{Y}_{i}^{(1)} \tag{2.58}$$

Equação (2.58) obtida a partir da Equação (2.41).

$$\frac{\partial \boldsymbol{Y}_{j}^{(2)}}{\partial \boldsymbol{I}_{j}^{(2)}} = g'\left(\boldsymbol{I}_{j}^{(2)}\right) \tag{2.59}$$

Equação (2.59) obtida a partir da Equação (2.44).

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_j^{(2)}} = \sum_{k=1}^{n_3} \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{I}_k^{(3)}} \cdot \frac{\partial \boldsymbol{I}_k^{(3)}}{\partial \boldsymbol{Y}_j^{(2)}} = \sum_{k=1}^{n_3} \underbrace{\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{I}_k^{(3)}}}_{fator(i)} \cdot \underbrace{\frac{\partial \left(\sum_{k=1}^{n_3} \boldsymbol{W}_{kj}^{(3)} \cdot \boldsymbol{Y}_j^{(2)} \right)}{\partial \boldsymbol{Y}_j^{(2)}}}_{fator(ii)} \tag{2.60}$$

onde o valor da derivada parcial do argumento do fator (ii) em relação à $\boldsymbol{Y}_j^{(2)}$ é o próprio valor de $\boldsymbol{W}_{kj}^{(3)}$, ou seja:

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_j^{(2)}} = \sum_{k=1}^{n_3} \underbrace{\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{I}_k^{(3)}}}_{fator(i)} \cdot \underbrace{\boldsymbol{W}_{kj}^{(3)}}_{fator(ii)} \tag{2.61}$$

O valor do fator (ii) da Equação (2.61) refere-se aos pesos sinápticos de todos os neurônios da camada de saída que estão interligados a um determinado neurônio j da segunda camada intermediária. Como já mencionado, todos os pesos $\boldsymbol{W}_{ji}^{(3)}$ foram ajustados na Etapa I com base em valores reais de erro, sendo que estes serão utilizados para o ajuste dos pesos da segunda camada neural intermediária (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]; Schalkoff, 1997 [35]; Yegnanarayana, 2009 [43]).

Em relação à obtenção de valores para o fator (i) da Equação (2.61), observa-se que o seu resultado pode ser fornecido por intermédio da multiplicação de (2.50) por (2.51), que resulta no valor absoluto de $\boldsymbol{\delta}_j^{(3)}$ explicitado em (2.54). Assim, levando-se em consideração tais substituições, a Equação (2.61) pode ser escrita como:

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_j^{(2)}} = -\sum_{k=1}^{n_3} \boldsymbol{\delta}_k^{(3)} \cdot \boldsymbol{W}_{kj}^{(3)} \tag{2.62}$$

Substituindo-se (2.58), (2.59) e (2.62) em (2.57), tem-se:

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(2)}} = -\left(\sum_{k=1}^{n_3} \boldsymbol{\delta}_k^{(3)} \cdot \boldsymbol{W}_{kj}^{(3)} \right) \cdot g'\left(\boldsymbol{I}_j^{(2)} \right) \cdot \boldsymbol{Y}_i^{(1)} \tag{2.63}$$

Portanto, o ajuste da matriz de pesos $\boldsymbol{W}_{ji}^{(2)}$ deve ser efetuado na direção oposta ao do gradiente a fim de minimizar o erro, ou seja:

$$\Delta \boldsymbol{W}_{ji}^{(2)} = -\eta \cdot \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(2)}} \quad \Leftrightarrow \quad \Delta \boldsymbol{W}_{ji}^{(2)} = \eta \cdot \boldsymbol{\delta}_j^{(2)} \cdot \boldsymbol{Y}_i^{(1)} \tag{2.64}$$

onde $\boldsymbol{\delta}_j^{(2)}$ é definido como o gradiente local em relação ao j-ésimo neurônio da segunda camada intermediária, sendo o mesmo dado por:

$$\boldsymbol{\delta}_j^{(2)} = \left(\sum_{k=1}^{n_3} \boldsymbol{\delta}_k^{(3)} \cdot \boldsymbol{W}_{kj}^{(3)} \right) \cdot g' \left(\boldsymbol{I}_j^{(2)} \right) \tag{2.65}$$

A expressão (2.64) pode ser convertida no seguinte procedimento iterativo:

$$\boldsymbol{W}_{ji}^{(2)}(t+1) = \boldsymbol{W}_{ji}^{(2)}(t) + \eta \cdot \boldsymbol{\delta}_j^{(2)} \cdot \boldsymbol{Y}_i^{(1)} \tag{2.66}$$

Em notação algorítmica, a expressão (2.66) é equivalente à seguinte:

$$\boldsymbol{W}_{ji}^{(2)} \leftarrow \boldsymbol{W}_{ji}^{(2)} + \eta \cdot \boldsymbol{\delta}_j^{(2)} \cdot \boldsymbol{Y}_i^{(1)} \tag{2.67}$$

Portanto, a expressão (2.67) ajusta os pesos dos neurônios da segunda camada escondida da rede, levando-se em consideração a retropropagação do erro advinda a partir dos neurônios da camada de saída.

(B) Ajuste dos pesos sinápticos da primeira camada escondida:

O processo de treinamento para a primeira camada neural escondida consiste em ajustar a matriz de pesos $\boldsymbol{W}_{ji}^{(1)}$ a fim de minimizar o erro entre a saída produzida pela rede em relação à retropropagação do erro advindo dos ajustes dos neurônios da segunda camada escondida. Assim, tem-se:

$$\nabla \boldsymbol{E}^{(1)} = \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(1)}} = \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_j^{(1)}} \cdot \frac{\partial \boldsymbol{Y}_j^{(1)}}{\partial \boldsymbol{I}_j^{(1)}} \cdot \frac{\partial \boldsymbol{I}_j^{(1)}}{\partial \boldsymbol{W}_{ji}^{(1)}} \tag{2.68}$$

Por intermédio das definições anteriores, tem-se:

$$\frac{\partial \boldsymbol{I}_j^{(1)}}{\partial \boldsymbol{W}_{ji}^{(1)}} = x_i \tag{2.69}$$

Equação (2.69) obtida a partir da Equação (2.40).

$$\frac{\partial \boldsymbol{Y}_j^{(1)}}{\partial \boldsymbol{I}_j^{(1)}} = g' \left(\boldsymbol{I}_j^{(1)} \right) \tag{2.70}$$

Equação (2.70) obtida a partir da Equação (2.43).

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_j^{(1)}} = \sum_{k=1}^{n_2} \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{I}_k^{(2)}} \cdot \frac{\partial \boldsymbol{I}_k^{(2)}}{\partial \boldsymbol{Y}_j^{(1)}} = \sum_{k=1}^{n_2} \underbrace{\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{I}_k^{(2)}}}_{fator(i)} \cdot \underbrace{\frac{\partial \left(\sum_{k=1}^{n_2} \boldsymbol{W}_{kj}^{(2)} \cdot \boldsymbol{Y}_j^{(1)} \right)}{\partial \boldsymbol{Y}_j^{(1)}}}_{fator(ii)} \tag{2.71}$$

Similarmente à Equação (2.61), o valor da derivada parcial do argumento do fator (ii) da Equação (2.71) em relação à $\boldsymbol{Y}_j^{(1)}$ é o próprio valor de $\boldsymbol{W}_{kj}^{(2)}$, ou seja:

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_j^{(1)}} = \sum_{k=1}^{n_2} \underbrace{\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{I}_k^{(2)}}}_{fator(i)} \cdot \underbrace{\boldsymbol{W}_{kj}^{(2)}}_{fator(ii)} \tag{2.72}$$

O valor do fator (ii) da Equação (2.72) refere-se aos pesos sinápticos de todos os neurônios da segunda camada intermediária que estão interligados a um determinado neurônio j da primeira camada intermediária. Como já mencionado, todos os pesos $\boldsymbol{W}_{ji}^{(2)}$ foram ajustados na Etapa II(A) com base em valores retropropagados de erro que tiveram como base o ajuste dos pesos $\boldsymbol{W}_{ji}^{(3)}$, os quais por sua vez foram ajustados baseando-se em valores reais de erro (Silva, 2010 [36]; Yegnanarayana, 2009 [43]).

Em relação à obtenção de valores para o fator (i) da Equação (2.72), observa-se que o seu resultado pode ser fornecido por intermédio da multiplicação de (2.59) por (2.60), que resulta no valor absoluto de $\boldsymbol{\delta}_j^{(2)}$ explicitado em (2.65). Assim, levando-se em consideração tais substituições, a Equação (2.71) pode ser escrita como:

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{Y}_j^{(1)}} = -\sum_{k=1}^{n_2} \boldsymbol{\delta}_k^{(2)} \cdot \boldsymbol{W}_{kj}^{(2)} \tag{2.73}$$

Substituindo-se (2.69), (2.70) e (2.73) em (2.68), tem-se:

$$\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(1)}} = -\left(\sum_{k=1}^{n_2} \boldsymbol{\delta}_k^{(2)} \cdot \boldsymbol{W}_{kj}^{(2)} \right) \cdot g'\left(\boldsymbol{I}_j^{(1)} \right) \cdot x_i \tag{2.74}$$

Portanto, o ajuste da matriz de pesos $\boldsymbol{W}_{ji}^{(2)}$ deve ser efetuado na direção oposta ao do gradiente a fim de minimizar o erro, ou seja:

$$\Delta \boldsymbol{W}_{ji}^{(1)} = -\eta \cdot \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{W}_{ji}^{(1)}} \quad \Leftrightarrow \quad \Delta \boldsymbol{W}_{ji}^{(1)} = \eta \cdot \boldsymbol{\delta}_j^{(1)} \cdot x_i \tag{2.75}$$

onde $\boldsymbol{\delta}_j^{(1)}$ é definido como o gradiente local em relação ao j-ésimo neurônio da primeira camada intermediária, sendo o mesmo dado por:

$$\boldsymbol{\delta}_j^{(1)} = \left(\sum_{k=1}^{n_2} \boldsymbol{\delta}_k^{(2)} \cdot \boldsymbol{W}_{kj}^{(2)} \right) \cdot g' \left(\boldsymbol{I}_j^{(1)} \right) \tag{2.76}$$

A Equação (2.75) pode ser convertida no seguinte procedimento iterativo:

$$\boldsymbol{W}_{ji}^{(1)}(t+1) = \boldsymbol{W}_{ji}^{(1)}(t) + \eta \cdot \boldsymbol{\delta}_j^{(1)} \cdot x_i \tag{2.77}$$

Em notação algorítmica, a Equação (2.77) é equivalente à seguinte:

$$\boldsymbol{W}_{ji}^{(1)} \leftarrow \boldsymbol{W}_{ji}^{(1)} + \eta \cdot \boldsymbol{\delta}_j^{(1)} \cdot x_i \tag{2.78}$$

Portanto, a expressão (2.78) ajusta os pesos dos neurônios da primeira camada neural escondida da rede, levando-se em consideração a retropropagação do erro advinda a partir dos neurônios da segunda camada neural escondida. Os procedimentos de ajuste de matrizes de pesos sinápticos apresentados nas equações anteriores podem ser generalizados para quaisquer topologias de redes PMC, independentemente da quantidade de camadas intermediárias (Silva, 2010 [36]; Hagan, 1996 [18]).

O critério de parada do processo fica em função do erro quadrático médio que calculado através da Equação (2.47), levando-se em consideração todas as amostras de treinamento disponíveis. O algoritmo converge quando o erro quadrático médio entre duas épocas sucessivas for suficientemente pequeno, ou seja:

$$|E_M^{atual} - E_M^{anterior}| \leq \varepsilon \tag{2.79}$$

onde ε é a precisão requerida para o processo de convergência, sendo especificado em função do tipo de aplicação a ser mapeada pela rede.

A quantidade de épocas também pode ser utilizada como critério de parada para o PMC em situações em que a precisão especificada para o problema se torna inalcançável. Para isto, basta limitar o processo de treinamento quando a quantidade de épocas já tenha alcançado um valor pré-determinado.

Os ajustes das matrizes de pesos são realizados somente na fase de treinamento da rede, em que se aplicam os passos *forward* e *backward* a fim de proceder eventuais cor-

reções sinápticas necessárias. Já na fase de teste, nenhum tipo de ajuste é efetuado nos parâmetros internos da rede, sendo que, para esta fase, somente o passo *forward* é processado com o objetivo de gerar as saídas da rede (Silva, 2010 [36]).

2.3.10.2 O Algoritmo *Backpropagation* com Taxa de *Momentum*

Nesta dissertação utiliza-se uma versão aperfeiçoada do algoritmo *backpropagation* com o objetivo de tornar o processo de convergência mais eficiente. Para conseguir-se essa versão, insere-se o denominado termo de *momentum*. Este parâmetro visa ponderar o quanto as matrizes sinápticas foram alteradas entre duas iterações anteriores e sucessivas. Formalmente, considerando-se os neurônios pertencentes à L-ésima camada, tem-se, do algoritmo *backpropagation* convencional:

$$\boldsymbol{W}_{ji}^{(L)}(t+1) = \boldsymbol{W}_{ji}^{(L)}(t) + \eta \cdot \boldsymbol{\delta}_j^{(L)} \cdot \boldsymbol{Y}_i^{(L-1)} \tag{2.80}$$

onde, inserindo-se o termo de *momentum*, obtém-se:

$$\boldsymbol{W}_{ji}^{(L)}(t+1) = \boldsymbol{W}_{ji}^{(L)}(t) + \underbrace{\alpha \cdot (\boldsymbol{W}_{ji}^{(L)}(t) - \boldsymbol{W}_{ji}^{(L)}(t-1))}_{termo\ \ de\ \ momentum} + \underbrace{\eta \cdot \boldsymbol{\delta}_j^{(L)} \cdot \boldsymbol{Y}_i^{(L-1)}}_{termo\ \ de\ \ aprendizagem} \tag{2.81}$$

onde α é definido como taxa de momentum e seu valor está compreendido entre zero e um. Conforme se pode abstrair da Equação (2.81), quando o valor da taxa de *momentum* for igual a zero, a equação se torna equivalente à Equação (2.80). Porém, para valores diferentes de zero, o termo de *momentum* passa a ser relevante, sendo que tal contribuição afetará positivamente o processo de convergência (Silva, 2010 [36]).

De forma mais específica, quando a solução atual (refletida por suas matrizes de peso) estiver longe da solução final (mínimo da função erro), a variação na direção oposta ao gradiente da função erro quadrático entre duas iterações sucessivas será também grande. Isto implica que a diferença entre as matrizes de pesos dessas duas iterações será bem considerável e, nesta situação, pode-se imprimir um passo maior de incremento para $\boldsymbol{W}^{(L)}$ em direção ao mínimo da função erro. A execução de tal tarefa ficará então a cargo do termo de *momentum*, pois o mesmo é responsável pela medição desta variação (Silva, 2010 [36]; Hagan, 1996 [18]).

Quando a solução atual estiver bem próxima da solução final, as variações nas matrizes

de peso serão bem pequenas, pois a variação do erro quadrático entre duas iterações sucessivas será baixa e, consequentemente, a contribuição do termo *momentum* para o processo de convergência é bem ínfima. A partir deste instante, todos os ajustes nas matrizes de peso acabam sendo conduzidos (quase que na totalidade) apenas pelo termo de aprendizagem, como ocorre também no algoritmo *backpropagation* convencional (Silva, 2010 [36]; Braga, 2007 [12])

A Figura 2.24 mostra a contribuição do termo de *momentum* (TM) e do termo de aprendizagem (TA) visando a convergência em direção ao mínimo $\boldsymbol{W}^{(OT)}$ da função erro quadrático.

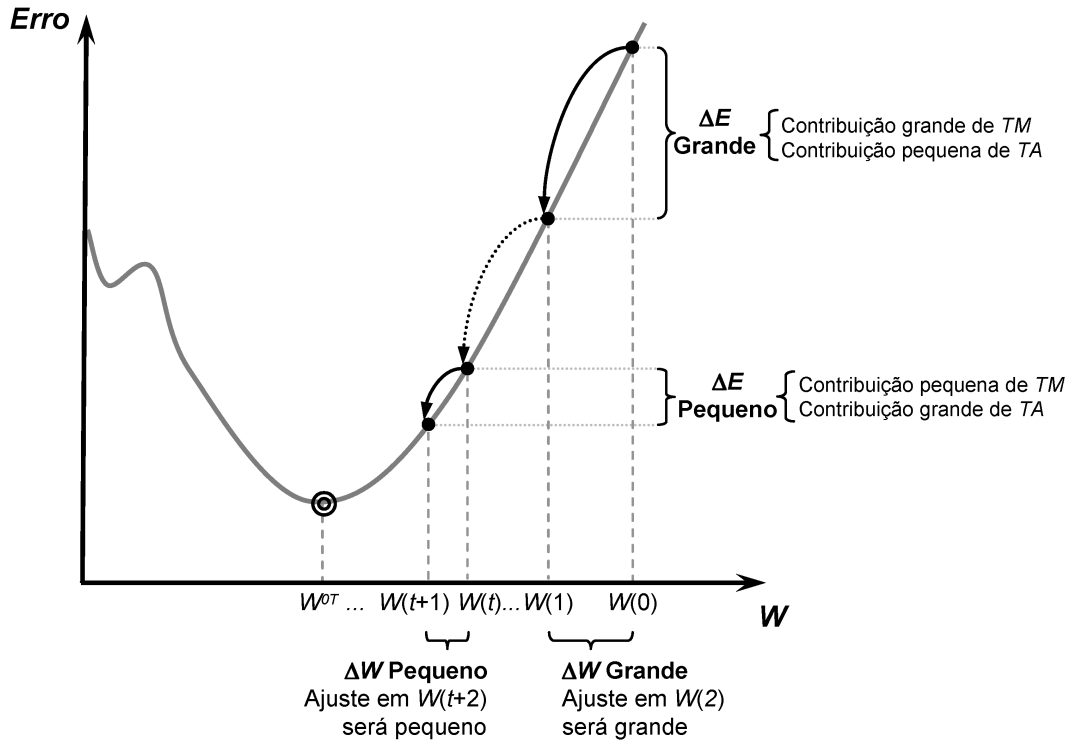

Figura 2.24: Ilustração do processo de treinamento utilizando o método de inserção do termo de *momentum*. Fonte: Silva, 2010 [36].

Logo, através da inserção do termo de *momentum*, o processo de convergência da rede se torna bem mais eficiente, pois se leva em consideração o critério de quão afastada está a solução atual da solução final (ótima). Valores compreeendidos entre $(0,05 \leq \eta \leq 0,75)$ e $(0 \leq \alpha \leq 0,9)$ são normalmente recomendados para treinamento de redes PMC (Rumelhart, 1987 [33]).

2.3.10.3 Algoritmo *Backpropagation* - Pseudocódigo

Algorithm 1: Algoritmo *Backpropagation* para PMC - Fase de Treinamento

Obter o conjunto de amostras de treinamento $\left\{\mathbf{x}^{(k)}\right\}$;
Associar o vetor de saída desejada $\left\{\mathbf{d}^{(k)}\right\}$ para cada amostra;
Iniciar $\boldsymbol{W}_{ji}^{(1)}$, $\boldsymbol{W}_{ji}^{(2)}$ e $\boldsymbol{W}_{ji}^{(3)}$ com valores aleatórios pequenos;
Especificar taxa de aprendizagem $\{\eta\}$ e precisão requerida $\{\varepsilon\}$;
Iniciar o contador de número de épocas $\{epoca \leftarrow 0\}$;
repita
  $E_M^{anterior} \leftarrow E_M$; $\{conforme\ \ (2.47)\}$
  para todas as amostras de treinamento $\left\{\mathbf{x}^{(k)}, \mathbf{d}^{(k)}\right\}$ **faça**
    Obter $\boldsymbol{I}_j^{(1)}$ e $\boldsymbol{Y}_j^{(1)}$; $\{conforme\ \ (2.40)\ \ e\ \ (2.43)\}$
    Obter $\boldsymbol{I}_j^{(2)}$ e $\boldsymbol{Y}_j^{(2)}$; $\{conforme\ \ (2.41)\ \ e\ \ (2.44)\}$
    Obter $\boldsymbol{I}_j^{(3)}$ e $\boldsymbol{Y}_j^{(3)}$; $\{conforme\ \ (2.42)\ \ e\ \ (2.45)\}$
    Determinar $\boldsymbol{\delta}_j^{(3)}$; $\{conforme\ \ (2.54)\}$
    Ajustar $\boldsymbol{W}_{ji}^{(3)}$; $\{conforme\ \ (2.56)\}$
    Determinar $\boldsymbol{\delta}_j^{(2)}$; $\{conforme\ \ (2.65)\}$
    Ajustar $\boldsymbol{W}_{ji}^{(2)}$; $\{conforme\ \ (2.67)\}$
    Determinar $\boldsymbol{\delta}_j^{(1)}$; $\{conforme\ \ (2.76)\}$
    Ajustar $\boldsymbol{W}_{ji}^{(1)}$; $\{conforme\ \ (2.78)\}$
  fim para
  Obter $\boldsymbol{Y}_j^{(3)}$ ajustado; $\{conforme\ \ itens\ \ 9, 10\ \ e\ \ 11\}$
  $E_M^{atual} \leftarrow E_M$; $\{conforme\ \ (2.47)\}$
  $epoca \leftarrow epoca + 1$;
até $|E_M^{atual} - E_M^{anterior}| \leq \varepsilon$. =0

Algorithm 2: Algoritmo *Backpropagation* para PMC - Fase de Teste

Obter uma amostra $\{\mathbf{x}\}$;
Assumir $\boldsymbol{W}_{ji}^{(1)}$, $\boldsymbol{W}_{ji}^{(2)}$ e $\boldsymbol{W}_{ji}^{(3)}$ já ajustadas no treinamento;
Execute as seguintes instruções:
  Obter $\boldsymbol{I}_j^{(1)}$ e $\boldsymbol{Y}_j^{(1)}$; $\{conforme\ \ (2.40)\ \ e\ \ (2.43)\}$
  Obter $\boldsymbol{I}_j^{(2)}$ e $\boldsymbol{Y}_j^{(2)}$; $\{conforme\ \ (2.41)\ \ e\ \ (2.44)\}$
  Obter $\boldsymbol{I}_j^{(3)}$ e $\boldsymbol{Y}_j^{(3)}$; $\{conforme\ \ (2.42)\ \ e\ \ (2.45)\}$
Disponibilizar as saídas da rede, as quais são dadas pelos elementos contidos em $\boldsymbol{Y}_j^{(3)}$. =0

Capítulo 3

Modelagem do Problema

A simulação do comportamento dinâmico da estrutura danificada é realizada impondo-se um estado de dano na estrutura através da prescrição do parâmetro de coesão β que descreve o defeito. O modelo de elementos finitos da estrutura danificada (com os valores prescritos do parâmetro de coesão), é utilizado para gerar as frequências naturais e os respectivos modos de vibração da estrutura danificada. Consideram-se estes parâmetros modais como tendo sido obtidos através de um ensaio modal, ou seja, são considerados parâmetros experimentais. Quando o modelo é ajustado com apenas um subconjunto dos modos experimentais, o restante dos modos é considerado como modos residuais não obtidos no ensaio modal. Da mesma forma, para simular um ensaio modal em que somente uma parte dos GDL serão monitorados, os modos experimentais são constituídos apenas com os GDL relativos aos GDL monitorados, o restante é considerado como GDL não medidos e não participam da formulação do modo experimental (Stutz, 1999 [41]).

3.1 Definição de Parâmetros - Análise Modal e Método dos Elementos Finitos

Como mencionado no Capítulo 2, a ideia básica para a utilização de características de vibração de uma estrutura na identificação de danos é que os parâmetros modais (frequências naturais, modos de vibração e amortecimentos modais) são funções das propriedades físicas da estrutura (massa, rigidez e amortecimento), logo, alterações nas propriedades físicas refletirão nas propriedades modais, que por sua vez, podem ser obtidas através de um ensaio experimental e serem de alguma forma utilizadas para se verificar alterações nas propriedades físicas. As técnicas modais para identificação de dano estrutural são normalmente implementadas através de um modelo de elementos finitos. Com este mo-

delo, pretende-se obter uma concordância satisfatória entre os resultados numéricos e os dados experimentais. Os parâmetros modais fornecidos pelo modelo de elementos finitos devem refletir os parâmetros modais observados no ensaio modal. Por exemplo, a Tabela 3.1 mostra as frequências naturais teóricas e aquelas obtidas a partir do MEF, para viga biapoiada do tipo Euler-Bernoulli na ausência de danos. Apenas os modos de vibração contidos na faixa de 0 a 450 Hz foram considerados medidos.

Modo de Vibração	Frequência Natural (Hz)	Frequência Natural Teórica (Hz)
1	11,73	11,73
2	46,93	46,93
3	105,60	105,60
4	187,76	187,74
5	293,41	293,34
6	422,63	422,41

Tabela 3.1: Frequências naturais da viga biapoiada.

Estes seis modos de vibração com as suas respectivas frequências naturais foram obtidos e utilizados no modelo de elementos finitos deste trabalho, para a simulação do comportamento dinâmico da estrutura danificada.

Uma viga de alumínio bidimensional de Euler-Bernoulli[1] contínua e de seção retangular é mostrada na Figura 3.1(a). Por meio do modelo de elementos finitos, a estrutura analisada foi discretizada em 20 elementos uniformes, como mostrado na Figura 3.1(b). Com isto, a viga discretizada possui 21 nós (Figura 3.2). Os resultados do ensaio modal da Tabela 3.1 foram obtidos com base neste modelo.

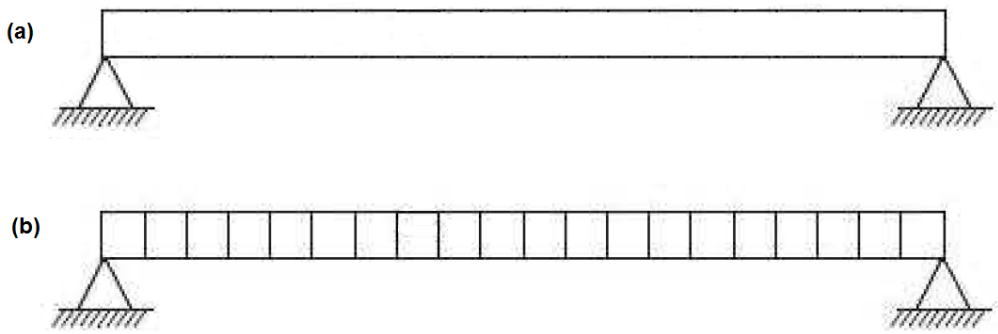

Figura 3.1: (a) Viga contínua e (b) viga discretizada em 20 elementos.

Foram adotados elementos com dois nós, onde cada ponto nodal possui dois Graus de Liberdade (GDL) de deslocamento, um transversal e outro de rotação, totalizando 40

[1]Na teoria de vigas de Euler-Bernoulli, é usual admitir que seções planas inicialmente perpendiculares ao eixo da peça permanecem planas e ainda perpendiculares a esse eixo após a deformação do elemento estrutural (Beer *et tal*, 2015 [6]).

graus de liberdade, devido à condição de contorno adotada, e um parâmetro de coesão. No entanto, foram considerados medidos ou analisados apenas 10 graus de liberdade transversais igualmente espaçados nos nós 2, 4, 6, 8, 10, 12, 14, 16, 18 e 20 (Figura 3.2).

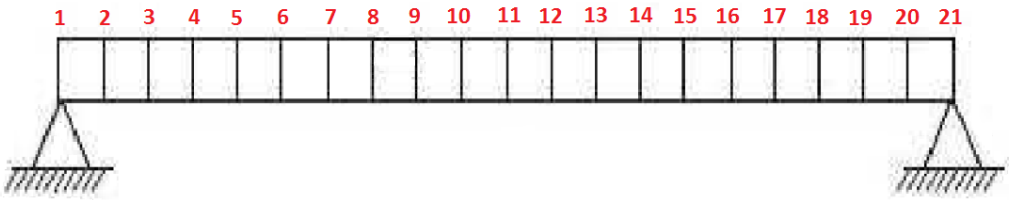

Figura 3.2: Viga discretizada em 20 elementos finitos e com as 21 nós ou posições.

As propriedades geométricas e materiais da viga de alumínio são mostradas na Tabela 3.2. Os parâmetros desta tabela serão utilizados como dados para a geração das matrizes de flexibilidade $\mathbf{G}_{exp}$ e $\mathbf{G}_{mm}$ no MEF (ver Capítulo 2 e Seção 2.2). As matrizes $\mathbf{G}_{exp}$ e $\mathbf{G}_{mm}$ possuirão, cada uma, após geradas, 10 linhas e 10 colunas. Cada coluna representará um GDL medido.

Comprimento	1 m
Espessura (h_0)	0,005 m
Largura	0,05 m
Área da seção transversal	$2,5 \times 10^{-4} m^2$
Massa específica	2700 kg/m^3
Momento de inércia de área	$5,21 \times 10^{-10} m^4$
Módulo de elasticidade nominal	$7,26 \times 10^{10} Pa$

Tabela 3.2: Propriedades nominais da viga.

Baseando-se na discretização da viga na Figura 3.1(b), a Figura 3.3 apresenta um elemento finito padrão, com dois nós extremos.

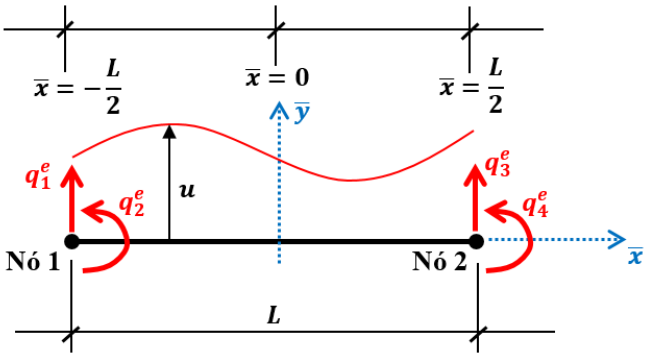

Figura 3.3: Elemento finito padrão e GDL por elemento.

O campo de deslocamento $y^e(\bar{x}, t)$, no interior do elemento padrão é aproximado por:

$$y^e(\bar{x},t) \simeq A_1^e(\bar{x})q_1^e(t) + A_2^e(\bar{x})q_2^e(t) + A_3^e(\bar{x})q_3^e(t) + A_4^e(\bar{x})q_4^e(t) \tag{3.1}$$

onde

- $\bar{x}$ é a coordenada espacial local, com origem no centro do elemento;
- q_1^e e q_2^e são, respectivamente, o deslocamento transversal e a rotação no nó 1;
- q_3^e e q_4^e são, respectivamente, o deslocamento transversal e a rotação no nó 2;
- A_1^e, A_2^e, A_3^e e A_4^e são as funções de forma.

As funções de forma (ou interpolação) são dadas pelas seguintes igualdades:

$$A_1 = \frac{1}{2} - \frac{3\bar{x}}{2L} + \frac{2\bar{x}^3}{L^3} \tag{3.2}$$

$$A_2 = \frac{L}{8} - \frac{\bar{x}}{4} - \frac{\bar{x}^2}{2L} + \frac{\bar{x}^3}{L^2} \tag{3.3}$$

$$A_3 = \frac{L}{2} + \frac{3\bar{x}}{2L} - \frac{2\bar{x}^3}{L^3} \tag{3.4}$$

$$A_4 = -\frac{L}{8} - \frac{\bar{x}}{4} + \frac{\bar{x}^2}{2L} + \frac{\bar{x}^3}{L^2} \tag{3.5}$$

onde

- L é o comprimento do elemento finito padrão em questão.

Considerando a aproximação para o campo de deslocamento, dada pela Equação (3.1), pode-se mostrar (ver Soriano, 2009 [39]), que a matriz de rigidez para cada elemento padrão é dada por:

$$\mathbf{K}^e = \int_{-\frac{L}{2}}^{\frac{L}{2}} E(\bar{x})I(\bar{x})\frac{\partial^2 \mathbf{A}^e(\bar{x})}{\partial \bar{x}^2}\frac{\partial^2 \mathbf{A}^{e^T}(\bar{x})}{\partial \bar{x}^2}d\bar{x} \tag{3.6}$$

onde

- E é o módulo de elasticidade;

- I é o momento de inércia da área da seção transversal da viga;
- $\mathbf{A}^e$ é dado por

$$\mathbf{A}^e = \begin{bmatrix} A_1^e(\bar{x}) \\ A_2^e(\bar{x}) \\ A_3^e(\bar{x}) \\ A_4^e(\bar{x}) \end{bmatrix}. \tag{3.7}$$

Definindo

$$E(\bar{x})I(\bar{x}) = \beta^e(\bar{x})E_0I_0 \tag{3.8}$$

onde

- E_0 é o valor inicial (ou nominal) do módulo de elasticidade;
- I_0 é o momento de inércia inicial (ou nominal) da área da seção transversal da viga.

Daí, a Equação (3.6) fica da seguinte forma:

$$\mathbf{K}^e = \int_{-\frac{L}{2}}^{\frac{L}{2}} \beta^e(\bar{x})E_0I_0\frac{\partial^2\mathbf{A}^e(\bar{x})}{\partial\bar{x}^2}\frac{\partial^2\mathbf{A}^{e^T}(\bar{x})}{\partial\bar{x}^2}d\bar{x}. \tag{3.9}$$

O parâmetro β, que varia no intervalo [0,1], é utilizado para descrever o estado de dano da estrutura e está relacionado com a ligação entre os pontos materiais do corpo, podendo ser interpretado como uma medida do estado de coesão local do material (Stutz, 1999 [41] Corrêa, 2013 [10]). β será denominado parâmetro de coesão.

Se β=1, a estrutura está livre de danos.

Se β=0, uma ruptura local é considerada.

A presença de um dano estrutural pode resultar em uma alteração no módulo de elasticidade do material, no momento de inércia da seção transversal ou em ambos (Corrêa, 2013 [10]). Segundo a Equação 3.8, o parâmetro de coesão reflete qualquer alteração na rigidez à flexão EI devido a presença de danos.

3.1.1 Modelo de Dano Constante por Partes

No modelo de dano comumente adotado na literatura, considera-se que a presença de dano reduz o módulo de elasticidade dos elementos que se encontram na região danificada. Dessa forma, considerando-se a Equação (3.8), tem-se, no interior de um elemento

$$E(\bar{x}) = \beta^e E_0; \quad 0 \leq \beta^e \leq 1. \tag{3.10}$$

A matriz de rigidez é, portanto, dada por:

$$\boldsymbol{K}^e = \beta^e \frac{E_0 I_0}{L} \begin{bmatrix} \frac{12}{L^2} & \frac{6}{L} & -\frac{12}{L^2} & \frac{6}{L} \\ \frac{6}{L} & 4 & -\frac{6}{L} & 2 \\ -\frac{12}{L^2} & -\frac{6}{L} & \frac{12}{L^2} & -\frac{6}{L} \\ \frac{6}{L} & 2 & -\frac{6}{L} & 4 \end{bmatrix} \tag{3.11}$$

ou equivalentemente,

$$\boldsymbol{K}^e = \beta^e K_0^e; \quad 0 \leq \beta^e \leq 1, \tag{3.12}$$

onde K_0^e refere-se a matriz de rigidez nominal. Portanto, no modelo de dano constante por partes (Figura 3.4), a matriz de rigidez de um elemento arbitrário é proporcional à sua matriz de rigidez nominal (sem dano), sendo a constante de proporcionalidade o parâmetro de coesão associado ao elemento em questão (Stutz, 1999 [41] Corrêa, 2013 [10]).

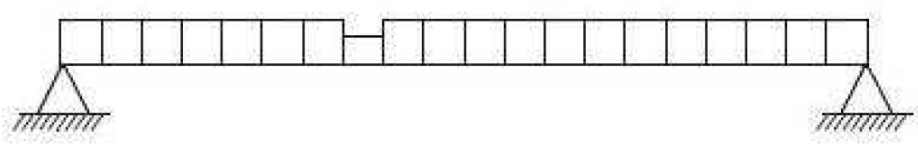

Figura 3.4: Modelo de dano constante por partes.

3.1.2 Modelo de Dano Contínuo

Neste trabalho, o dano é continuamente descrito ao longo da viga. Todos os danos apresentados , para a estrutura do tipo viga, possuem um perfil triangular de acordo com o modelo de dano contínuo ilustrado nas Figuras 3.5 e 3.6, onde:

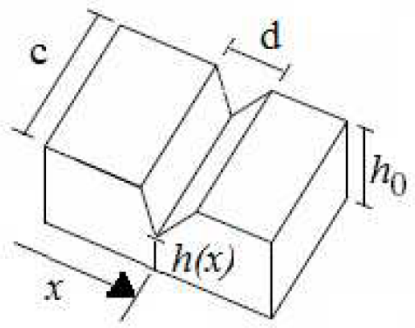

Figura 3.5: Perfil do dano simulado.

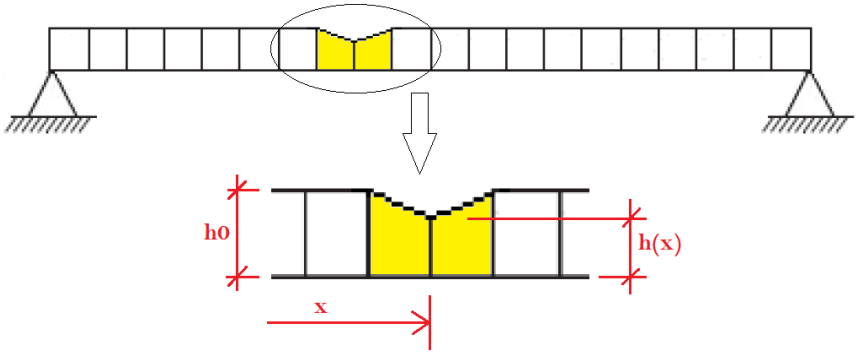

Figura 3.6: Dois elementos afetados pelo dano.

- c representa a largura da viga;
- h_0 indica a espessura original (nominal) da viga;
- $h(x)$ indica a espessura na posição danificada;
- x refere-se à posição do vértice do triângulo formado;
- d corresponde ao valor da medida do comprimento dos dois elementos afetados pelo dano.

Para este modelo de dano ocorrer, no interior de um elemento finito, o campo de coesão $\beta^e(\bar{x})$, assim como o campo de deslocamento $y^e(\bar{x}, t)$, é interpolado a partir de valores nodais.

$$\beta^e(\bar{x}) = N_1^e(\bar{x})\beta_1^e(\bar{x}) + N_2^e(\bar{x})\beta_2^e(\bar{x}) \tag{3.13}$$

onde

- β_1^e e β_2^e são, respectivamente, os valores nodais de coesão nos nós locais 1 e 2;

- N_1^e e N_2^e são as funções de interpolação do campo de coesão, dadas por:

$$N_1 = \frac{1}{2} - \frac{\bar{x}}{L}, \tag{3.14}$$

$$N_2 = \frac{1}{2} + \frac{\bar{x}}{L}. \tag{3.15}$$

Para a discretização do campo de danos utilizaram-se as funções de interpolação lineares mostradas acima, cujas representações, para um elemento com comprimento de valor unitário, são apresentadas nas Figuras 3.7 e 3.8.

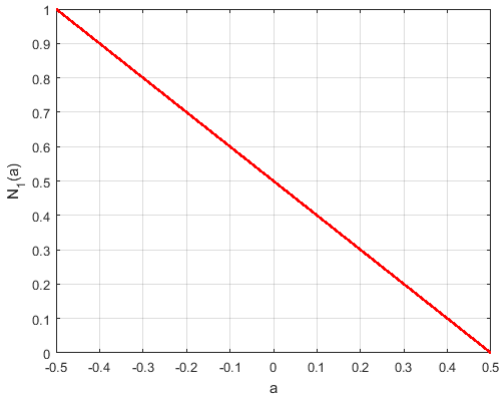

Figura 3.7: Função de interpolação unidimensional linear N_1.

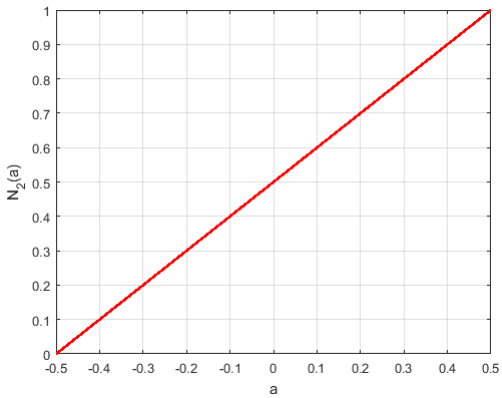

Figura 3.8: Função de interpolação unidimensional linear N_2.

A discretização do campo β não depende da discretização do campo de deslocamentos, de forma que diferentes malhas poderiam ser adotadas. A discretização do campo de coesão está relacionada à capacidade de discretização do dano, ou seja, quanto maior a discretização, maior a capacidade de se descrever o dano (Stutz, 1999 [41]; Corrêa, 2013 [10]).

A Figura 3.9 representa o elemento de viga do tipo Euller-Bernoulli utilizado neste trabalho, com dois nós extremos para a interpolação tanto do campo de deslocamento quanto do parâmetro de coesão. A mesma malha de elementos finitos foi utilizada para aproximar o campo de deslocamentos e também o campo de coesão (dano). (Stutz, 1999 [41]; Corrêa, 2013 [10]).

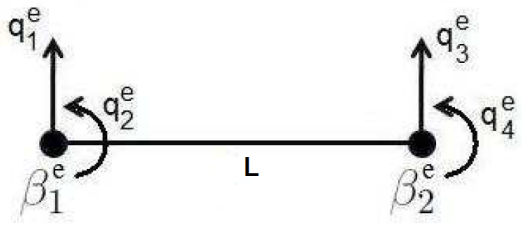

Figura 3.9: Elemento padrão e parâmetros de coesão.

Com base nas Equações (3.9) e (3.13) e nas funções de interpolação (3.14) e (3.15) utilizadas para descrever o dano, a matriz de rigidez (ver Equação (3.16)) de um elemento finito da estrutura, com modelo de dano contínuo, é dada por:

$$\mathbf{K}^e = \frac{E_0 I_0}{2l} \begin{bmatrix} \frac{12}{l^2}(\beta_1+\beta_2) & \frac{4}{l}(2\beta_1+\beta_2) & -\frac{12}{l^2}(\beta_1+\beta_2) & \frac{4}{l}(\beta_1+2\beta_2) \\ \frac{4}{l}(2\beta_1+\beta_2) & 2(3\beta_1+\beta_2) & -\frac{4}{l}(2\beta_1+\beta_2) & 2(\beta_1+\beta_2) \\ -\frac{12}{l^2}(\beta_1+\beta_2) & -\frac{4}{l}(2\beta_1+\beta_2) & \frac{12}{l^2}(\beta_1+\beta_2) & -\frac{4}{l}(\beta_1+2\beta_2) \\ \frac{4}{l}(\beta_1+2\beta_2) & 2(\beta_1+\beta_2) & \frac{4}{l}(\beta_1+2\beta_2) & 2(\beta_1+3\beta_2) \end{bmatrix} \tag{3.16}$$

Nos nós extremos o dano afeta apenas um elemento. Nos demais nós, um dano imposto a partir de um nó arbitrário, envolve dois elementos adjacentes (Figura 3.6).

Como já mencionado, o dano pode ser interpretado como uma variação no módulo de elasticidade, no momento de inércia da área transversal da estrutura ou em ambos. Daí, por simplicidade, considera-se que o módulo de elasticidade é uniforme ao longo da viga, ou seja, o dano só altera as propriedades geométricas da mesma. Portanto, o momento de inércia da área da seção transversal da estrutura danificada é dado por:

$$I(x) = \beta(x) I_0 \tag{3.17}$$

Considerando-se uma viga com a área de seção transversal retangular, o momento de inércia de área é dado por (Stewart, 2001 [40]; Anton, 2000 [3]):

$$I_0 = \int_A y^2 dA = \int_{-\frac{h_0}{2}}^{\frac{h_0}{2}} y^2 c dy = \frac{c}{3}\left[y^3\right]_{-\frac{h_0}{2}}^{\frac{h_0}{2}} \quad \Rightarrow \quad I_0 = \frac{ch_0^3}{12} \tag{3.18}$$

Logo, o parâmetro de coesão pode ser definido como uma relação cúbica que depende apenas da espessura original da viga (h_0) e da espessura na posição danificada ($h(x)$).

$$\beta(x) = \left(\frac{h(x)}{h_0}\right)^3. \tag{3.19}$$

A Figura 3.10 mostra a relação entre a razão da espessura da viga com dano e sem dano e o parâmetro de coesão. A curva é uma cúbica.

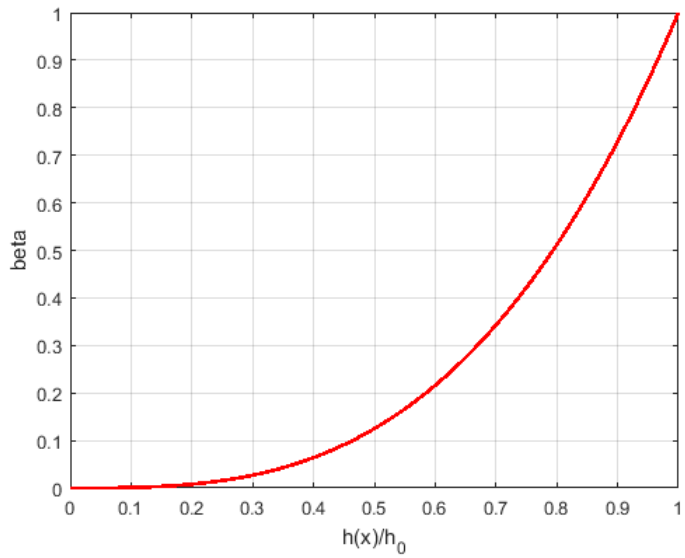

Figura 3.10: Parâmetro de coesão β como função da relação entre a espessura da viga com dano $h(x)$ e sem dano h_0 (Equação (3.19)).

Para obter-se a matriz de rigidez da viga de Euler-Bernoulli, foi utilizado o pacote (toolbox) CALFEM[2] do software MATLAB (Bang, 2000 [5]).

Porém, o arquivo referente à matriz de rigidez do elemento ($\mathbf{K}^e$) foi alterado para fornecer a matriz de rigidez de cada elemento em função dos parâmetros de coesão β_1 e β_2 conforme a Equação 3.16.

[2] CALFEM é a abreviação de *Computer Aided Learning of the Finite Element Method* (Austrell, 2004 [4]).

Capítulo 4

Estudo de Caso

Na Seção 4.1 é mostrado como a Rede Neural Artificial foi definida para este estudo. Na Seção 4.2 são apresentados e discutidos os resultados obtidos a partir de simulações numéricas com as RNA para identificação de danos em uma viga simplesmente apoida de Euler-Bernoulli, após o treinamento e teste da rede.

4.1 Parâmetros da Rede Neural Artificial

Utilizam-se duas funções de ativação que são totalmente diferenciáveis, ou seja, são funções cujas derivadas de primeira ordem existem e são conhecidas em todos os pontos de seu domínio de definição.

A primeira delas é a função Tangente Hiperbólica, também conhecida como Função de Transferência Tan-Sigmoid ou *tansig*. O seu gráfico é mostrado na Figura 4.1.

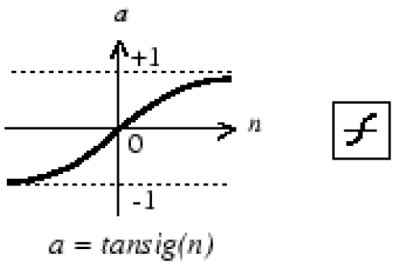

Figura 4.1: Função de ativação Tangente Hiperbólica (*tansig*). Fonte: Demuth, 1994 [13].

A função de ativação Tangente Hiperbólica (*tansig*) é dada pela Equação (4.1):

$$g(u) = \frac{1 - e^{-\theta.u}}{1 + e^{-\theta.u}}, \tag{4.1}$$

A outra função de ativação é a Função Logística, também conhecida como Função de Transferência Log-Sigmoid ou *logsig*. O seu gráfico é mostrado na Figura 4.2.

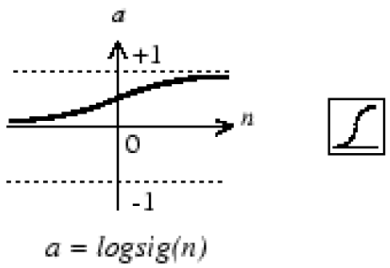

Figura 4.2: Função de ativação Logística (*logsig*). Fonte: Demuth, 1994 [13].

A função de ativação Logística (*logsig*) é dada pela Equação (4.2):

$$g(u) = \frac{1}{1 + e^{-\theta.u}}, \tag{4.2}$$

Onde θ é uma constante real associada ao nível de inclinação da função de ativação Tangente Hiperbólica ou Logística em relação ao seu ponto de inflexão (Silva, 2010 [36]; Haykin, 2007 [19]; Braga, 2007 [12]; Hagan, 1996 [18]).

A arquitetura de RNA adotada, neste trabalho, é a *feedforward* de camadas múltiplas, ou mais especificamente uma rede *Perceptron* de Múltiplas Camadas (PMC). Utiliza-se para o processo de treinamento supervisionado da rede PMC o algoritmo *backpropagation* com *momentum*, como descrito no Capítulo 2 na Seção 2.3. A topologia para o PMC desta dissertação é composta pelos seguintes itens, conforme também é ilustrado na Figura 4.3:

- 100 sinais de entrada;
- 100 neurônios na primeira camada neural escondida, cada neurônio utilizando a função de ativação *tansig*;
- 100 neurônios na segunda camada neural escondida, cada neurônio utilizando a função de ativação *tansig*;
- 100 neurônios na terceira camada neural escondida, cada neurônio utilizando a função de ativação *tansig*;
- 21 sinais associados à camada neural de saída, cada neurônio utilizando a função de ativação *logsig*.

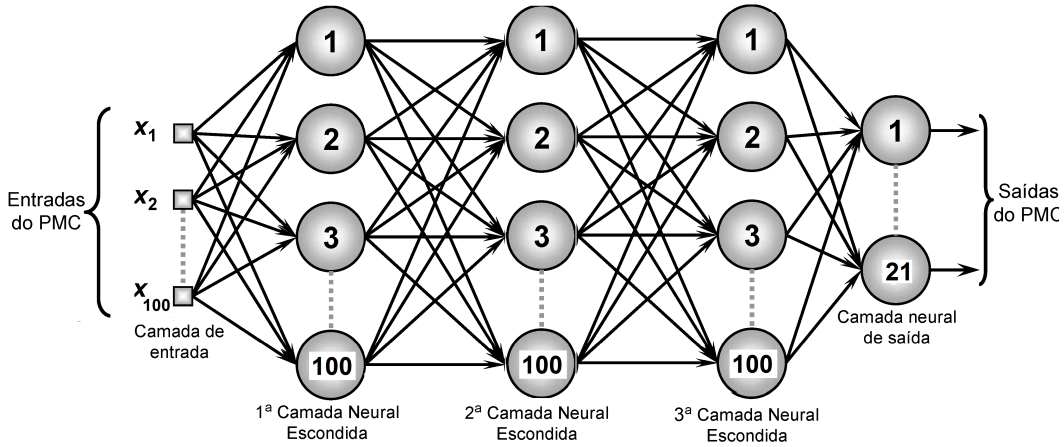

Figura 4.3: Topologia de rede PMC adotada nesta dissertação.

4.1.1 Definição dos Padrões de Entrada e Saída

Para implementar o código da Rede Neural Artificial, foram criados 67 padrões de entrada e 67 padrões de saída, pois o treinamento é supervisionado (Capítulo 2 na Seção 2.3). Sendo que 51 destes padrões de entrada e 51 dos padrões de saída são para o treinamento da rede neural. Os outros 16 padrões de entrada e de 16 padrões de saída são para teste da rede neural artificial. Deve-se deixar claro que, para cada padrão de entrada criado há um respectivo padrão de saída.

A imposição do dano é realizado escolhendo-se um nó ou posição e um valor para o parâmetro de coesão β (ver Tabela 4.3). Neste estudo, na fase de treinamento, o valor do β varia no intervalo [0,216;0,729]. Nos gráficos dos resultados computacionais, que estão no Capítulo 4, o valor do parâmetro de coesão β é sempre convertido, por conveniência, para $\dfrac{h(x)}{h_0}$ através da utilização da Equação (4.3):

$$\beta(x) = \left(\frac{h(x)}{h_0}\right)^3 \quad \Rightarrow \quad \frac{h(x)}{h_0} = \sqrt[3]{\beta(x)} \tag{4.3}$$

A Tabela 4.1 mostra as intensidades de dano, ou os valores do parâmetro β, impostos para a identificação de danos na viga simplesmente apoiada, na fase de treinamento. A porcentagem é calculada através da Equação (4.4).

$$Porcentagem = 100 - h(x)/h_0 \tag{4.4}$$

A Tabela 4.2 mostra as intensidades de dano, ou os valores do parâmetro β, impostos para a identificação de danos na viga simplesmente apoiada, na fase de teste.

β	$h(x)/h_0$	Dano em porcentagem
0,216	0,60	40%
0,422	0,75	25%
0,729	0,90	10%

Tabela 4.1: Intensidade dos danos ou os valores do parâmetro β para a fase de treinamento.

β	$h(x)/h_0$	Dano em porcentagem
0,600	0,84	16%
0,512	0,80	20%
0,343	0,70	30%

Tabela 4.2: Intensidade dos danos ou os valores do parâmetro β para a fase de teste.

Durante a realização deste trabalho, utilizou-se a fórmula, com norma euclidiana, $\|\mathbf{G}_{mm} - \mathbf{G}_{exp}\|$ para gerar os dados de entrada para a RNA. Os resultados encontrados não foram satisfatórios. A partir de então buscou-se um modo alternativo para gerar os dados de entrada. Após algumas tentativas e erros, o meio utilizado para gerar cada padrão de entrada da Rede Neural Artificial foi através da Equação (4.5), onde também é usada a norma euclidiana.

$$\frac{\|\mathbf{G}_{mm} - \mathbf{G}_{exp}\|}{\|\mathbf{G}_{mm}\|} \cdot 100 \tag{4.5}$$

A rede PMC foi treinada sem a presença de dados ruidosos, ou seja, com ruídos de 0%. Mas, na fase de teste, foram incluídos dados ruidosos de 3%, como também os de 0%.

A Tabela 4.3 mostra as posições ou nós em que foram impostos os danos para o treinamento da rede PMC, seus respectivos parâmetros de coesão β e estes convertidos em $h(x)/h_0$.

Observando a Tabela 4.3 vê-se que há 51 valores para β, ou seja, 51 valores de danos impostos. Cada nó tem três valores para β e, ao lado, encontram-se os seus respectivos valores em termos de $h(x)/h_0$.

4.1.2 Definição dos Parâmetros de Treinamento da RNA

Os parâmetros de treinamento que produziram os melhores resultados, após a implementação do código criado para a RNA, são os mostrados nos itens abaixo:

- Possui 100 sinais de entrada;

Posição do dano (nó)	β	$h(x)/h_0 = \sqrt[3]{\beta}$
3	0,216; 0,422; 0,729	0,6; 0,75; 0,9
4	0,216; 0,422; 0,729	0,6; 0,75; 0,9
5	0,216; 0,422; 0,729	0,6; 0,75; 0,9
6	0,216; 0,422; 0,729	0,6; 0,75; 0,9
7	0,216; 0,422; 0,729	0,6; 0,75; 0,9
8	0,216; 0,422; 0,729	0,6; 0,75; 0,9
9	0,216; 0,422; 0,729	0,6; 0,75; 0,9
10	0,216; 0,422; 0,729	0,6; 0,75; 0,9
11	0,216; 0,422; 0,729	0,6; 0,75; 0,9
12	0,216; 0,422; 0,729	0,6; 0,75; 0,9
13	0,216; 0,422; 0,729	0,6; 0,75; 0,9
14	0,216; 0,422; 0,729	0,6; 0,75; 0,9
15	0,216; 0,422; 0,729	0,6; 0,75; 0,9
16	0,216; 0,422; 0,729	0,6; 0,75; 0,9
17	0,216; 0,422; 0,729	0,6; 0,75; 0,9
18	0,216; 0,422; 0,729	0,6; 0,75; 0,9
19	0,216; 0,422; 0,729	0,6; 0,75; 0,9

Tabela 4.3: Posições em que foram colocados os danos para o treinamento da rede PMC.

- Três camadas neurais escondidas com 100 neurônios cada uma;
- Uma camada neural de saída com 21 neurônios;
- Funções de ativação: *tansig* nas camadas neurais escondidas e *logsig* na camada neural de saída.
- Algoritmo de treinamento *Gradient descent backpropagation* com *momentum* e taxa adaptativa (*traingdx*);
- Número de épocas ou iterações estabelecidas = 2000000;
- Número de épocas atingido pela RNA após treinamento = 676756;
- Erro final desejado ou estabelecido $= 0,0000001 = 1 \cdot 10^{-7}$;
- Taxa de aprendizagem $= 0,4$;
- Taxa ou coeficiente de *momentum* $= 0,6$;
- A rede foi treinada atingindo-se o erro estabelecido.

4.2 Resultados Computacionais

4.2.1 Teste da Rede Neural Artificial

Após o treinamento da rede neural, segue-se a etapa de teste da mesma. Nesta seção serão mostrados, através de gráficos, dezesseis resultados obtidos por intermédio dos testes com a RNA, rede PMC. A Tabela 4.4 exibe as 4 posições escolhidas para testar a rede neural após o treinamento. Na primeira coluna da Tebela 4.4 há a posição ou nó onde o dano foi imposto, na segunda coluna colocou-se o valor do parâmetro de coesão β e na terceira coluna o valor do $\dfrac{h(x)}{h_0} = \sqrt[3]{\beta(x)}$.

Posição para Teste	β	$h(x)/h_0$
4	0,512; 0,343	0,80; 0,70
10	0,512; 0,343	0,80; 0,70
13	0,512; 0,343	0,80; 0,70
16	0,600; 0,343	0,84; 0,70

Tabela 4.4: Posições escolhidas para testar a RNA após o treinamento.

Os resultados gráficos que mostram a identificação de danos são apresentados da seguinte forma: o eixo vertical possui a espessura relativa pela espessura nominal $\left(\dfrac{h(x)}{h_0}\right)$ (como mostrado na Figura 3.6 da Subseção 3.1.2 do Capítulo 3) e o eixo horizontal contém o número do nó ou posição do dano. A RNA treinada será avaliada em relação à sua capacidade de localizar e quantificar o dano imposto na viga em estudo mesmo quando os sinais estão corrompidos por ruído. Vale ressaltar que apenas 10 GDL transversais estão sendo considerados medidos em um total de 40 GDL da estrutura, em uma faixa de frequência de 0 a 450 Hz.

Com o objetivo de simular de forma mais realística os dados experimentais, a simulação da presença de ruído nas medições foi abordada, adicionando-se às formas modais de vibração uma perturbação aleatória com distribuição uniforme. Os modos de vibração, com imposição de ruído, são então fornecidos através da seguinte formulação,

$$\mathbf{\Phi}_r(ij) = \mathbf{\Phi}_{(ij)}(1 + \frac{p}{100}\mathbf{rand}(-1,1)), \tag{4.6}$$

onde $\mathbf{\Phi}_r$ corresponde aos modos de vibração contaminados por ruído e $\mathbf{\Phi}$ aos modos de vibração originais; p é o nível de ruído acrescentado; e $\mathbf{rand}(-1,1)$ é uma função geradora de números aleatórios no intervalo $(-1,1)$.

4.2.2 Dano na Posição 4

A Figura 4.4 mostra o dano na posição 4 da viga discretizada. Para este nó utilizaram-se as razões $\frac{h(x)}{h_0} = 0,80$ e $\frac{h(x)}{h_0} = 0,70$ com $\beta = 0,512$ e $\beta = 0,343$, respectivamente. Isto significa que os danos impostos neste nó são de 20% e 30% da espessura nominal da viga. Onde h_0 representa a espessura nominal da viga que mede 0,005 m.

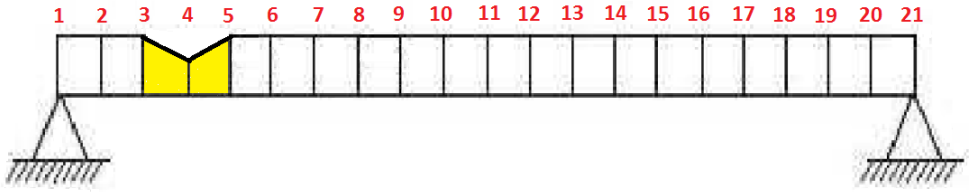

Figura 4.4: Ilustração do dano na posição 4 da viga discretizada.

4.2.2.1 Com Ruído de 0%

Os resultados obtidos considerando medições sem interferência de ruído nos sinais, representam uma situação ideal. Utilizamos esses resultados para mostrar a capacidade do método proposto em identificar danos caso fosse realizado algum tratamento nos sinais obtidos de forma a eliminar tais perturbações. Como pode ser visto nas Figuras 4.5 e 4.6, para um dano considerado na posição 4 da discretização da viga, o método proposto conseguiu localizar e quantificar o dano com bastante exatidão, quando não foram considerados ruídos de medição.

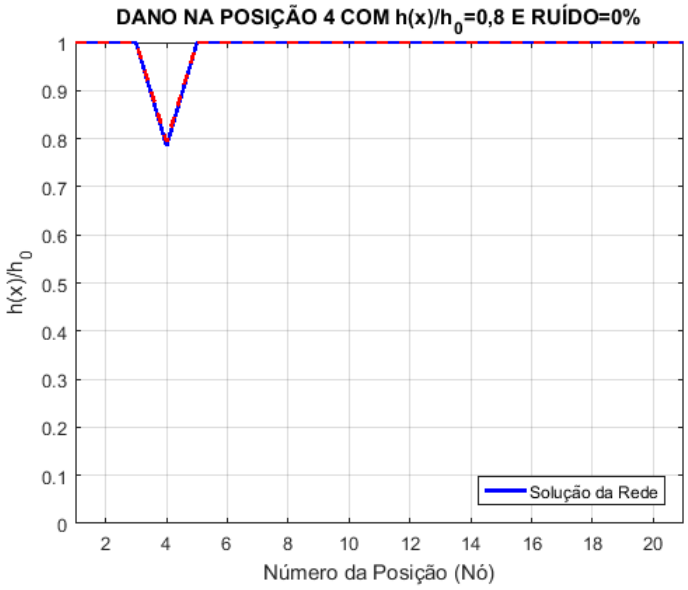

Figura 4.5: Dano na posição 4 com ruído de 0% e $\frac{h(x)}{h_0} = 0,80$.

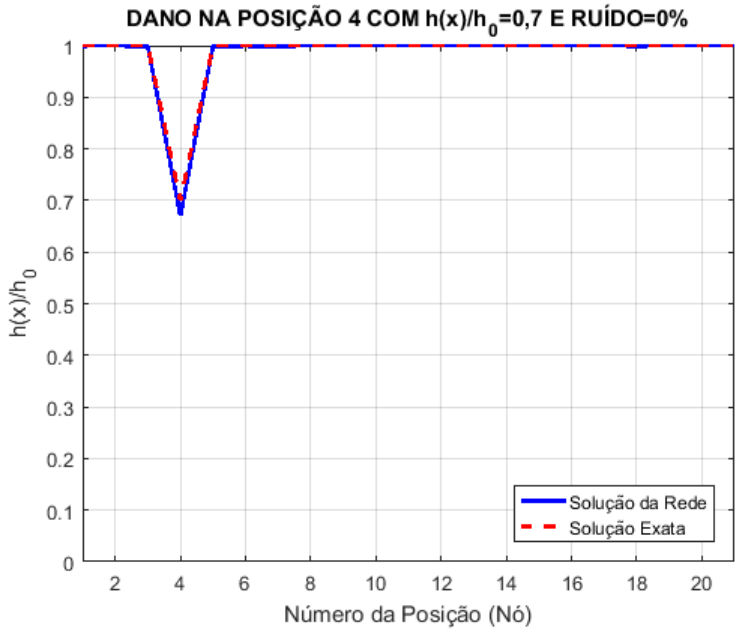

Figura 4.6: Dano na posição 4 com ruído de 0% e $\dfrac{h(x)}{h_0} = 0,70$.

4.2.2.2 Com Ruído de 3%

Com a adição de ruído igual a 3% nos modos de vibração, o modelo de identificação de danos apresentado não conseguiu localizar o dano, mas, conseguiu indicar a região do dano, como pode ser verificado nas Figuras 4.7 e 4.8.

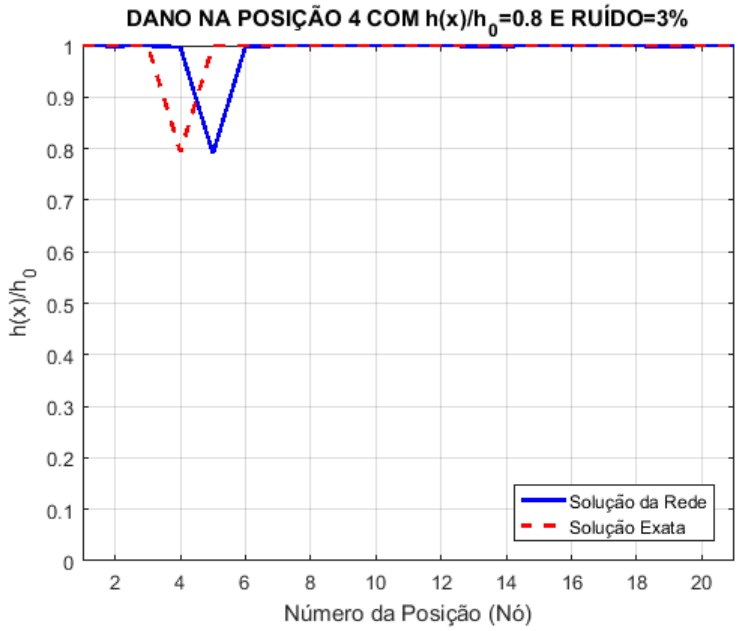

Figura 4.7: Dano na posição 4 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,80$.

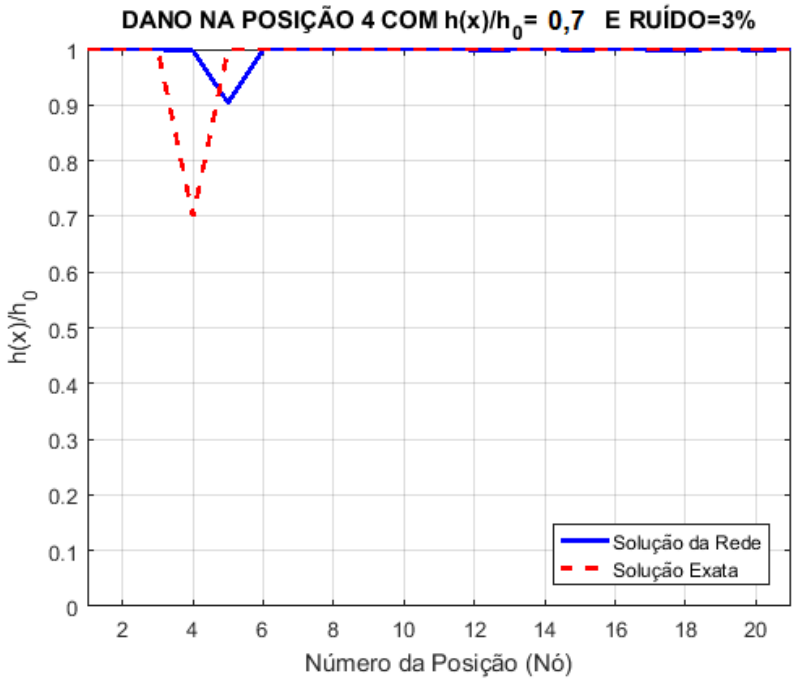

Figura 4.8: Dano na posição 4 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,70$.

4.2.3 Dano na Posição 10

A Figura 4.9 mostra o dano na posição 10 da viga discretizada. Para este nó utilizaram-se as razões $\dfrac{h(x)}{h_0} = 0,80$ e $\dfrac{h(x)}{h_0} = 0,70$ com $\beta = 0,512$ e $\beta = 0,343$, respectivamente. Isto significa que os danos impostos neste nó são de 20% e 30% da espessura nominal da viga.

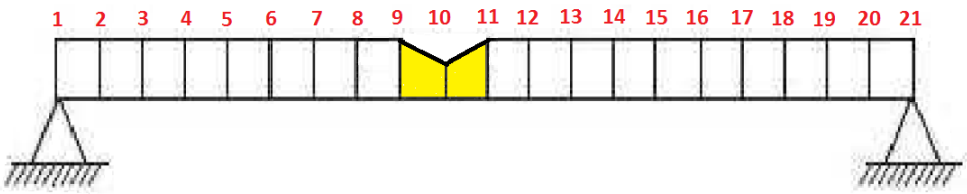

Figura 4.9: Ilustração do dano na posição 10 da viga discretizada.

4.2.3.1 Com Ruído de 0%

Novamente pode-se verificar pelas Figuras 4.10 e 4.11 que o modelo de identificação de danos proposto usando RNA conseguiu localizar e quantificar satisfatoriamente o dano, quando não foram considerados ruído de medição.

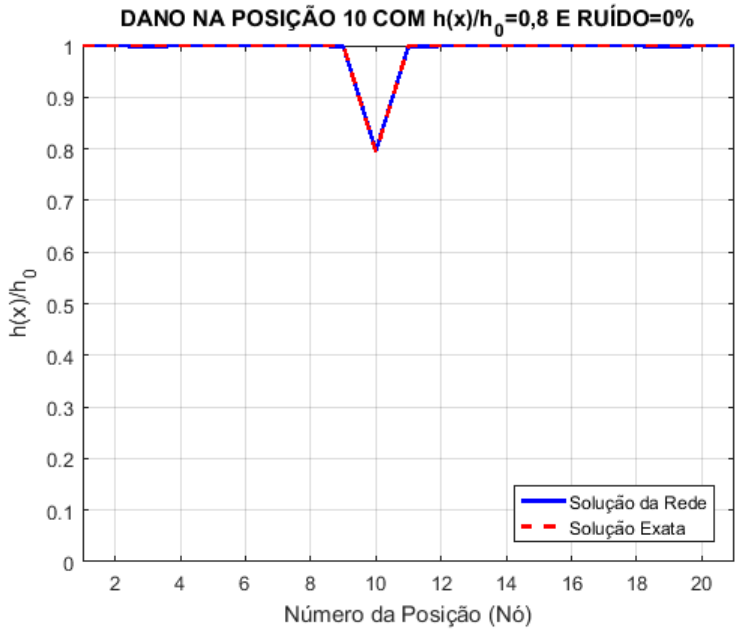

Figura 4.10: Dano na posição 10 com ruído de 0% e $\dfrac{h(x)}{h_0} = 0,80$.

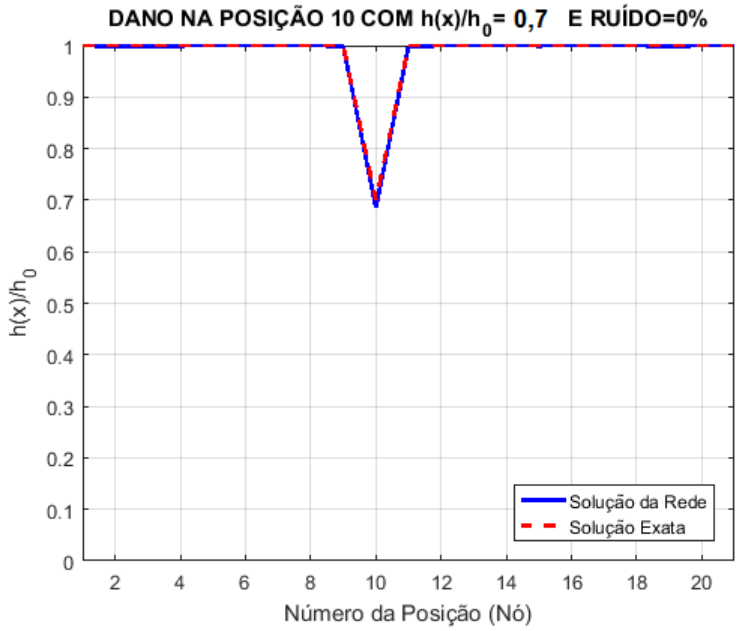

Figura 4.11: Dano na posição 10 com ruído de 0% e $\dfrac{h(x)}{h_0} = 0,70$.

4.2.3.2 Com Ruído de 3%

Considerando a posição 10 da discretização da viga e os dados contaminados com ruído de 3%, o método utilizado conseguiu localizar a posição do dano com exatidão, mas não quantificou o dano corretamente (Figuras 4.12 e 4.13).

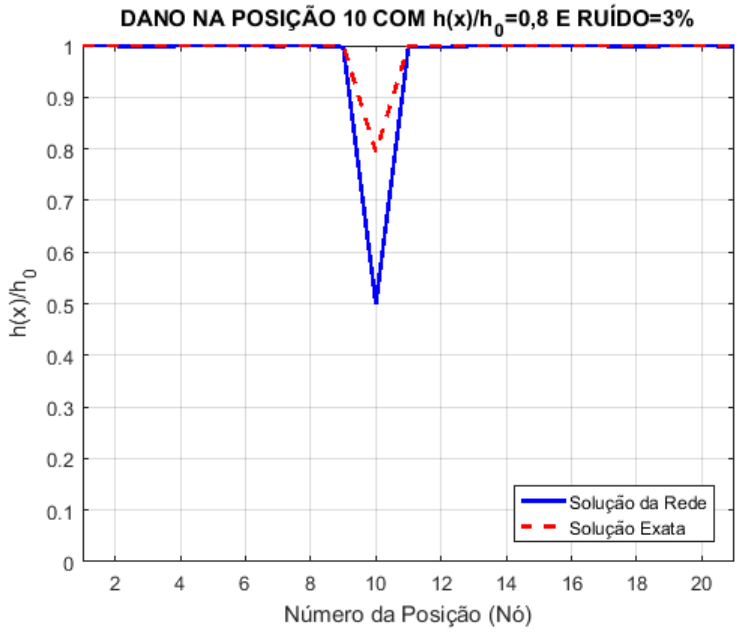

Figura 4.12: Dano na posição 10 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,80$.

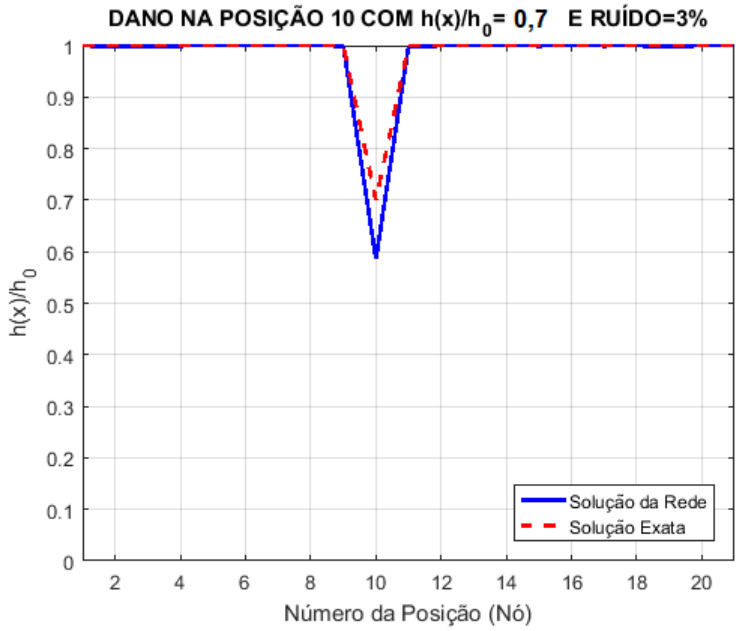

Figura 4.13: Dano na posição 10 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,70$.

4.2.4 Dano na Posição 13

A Figura 4.14 mostra o dano na posição 13 da viga discretizada. Para este nó utilizaram-se as razões $\dfrac{h(x)}{h_0} = 0,80$ e $\dfrac{h(x)}{h_0} = 0,70$ com $\beta = 0,512$ e $\beta = 0,343$, respectivamente. Isto significa que os danos impostos neste nó são de 20% e 30% da espessura nominal da viga.

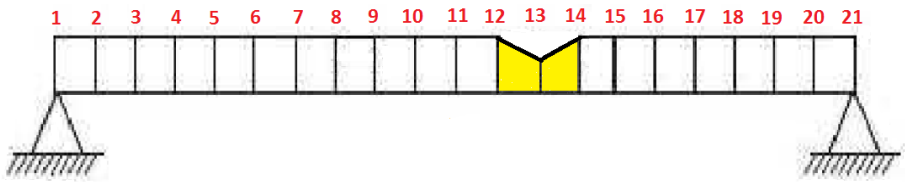

Figura 4.14: Ilustração do dano na posição 13 da viga discretizada.

4.2.4.1 Com Ruído de 0%

Considera-se, agora, um dano próximo da região central da viga, sem a adição de ruído aos modos de vibração. Pode-se verificar que o método utilizado neste trabalho conseguiu localizar e quantificar o dano com bastante exatidão (Figuras 4.15 e 4.16).

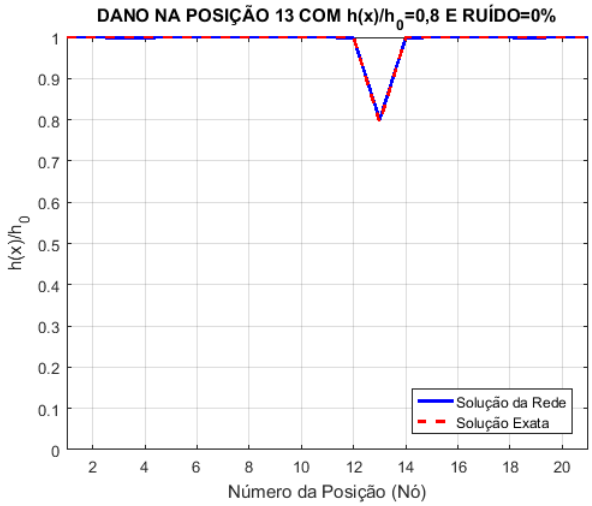

Figura 4.15: Dano na posição 13 com ruído de 0% e $\dfrac{h(x)}{h_0} = 0,80$.

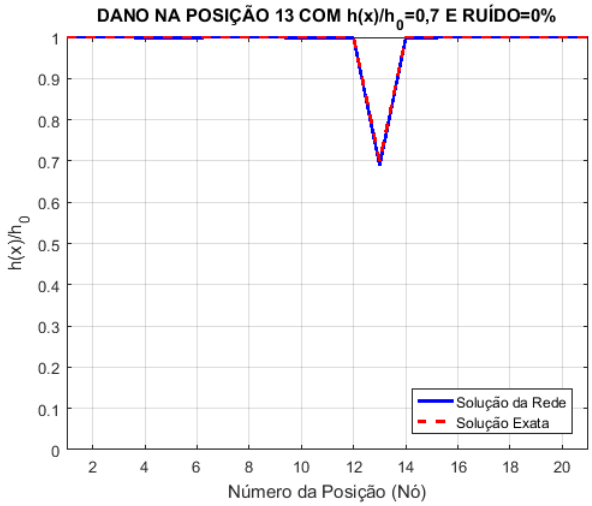

Figura 4.16: Dano na posição 13 com ruído de 0% e $\dfrac{h(x)}{h_0} = 0,70$.

4.2.4.2 Com Ruído de 3%

Considerando-se os dados contaminados com ruído de 3%, percebe-se na Figura 4.17 que a região onde a estrutura encontra-se danificada foi localizada, mas em uma posição adjacente ao nó onde foi imposto o dano. A Figura 4.18 mostra que o método conseguiu localizar e quantificar o dano satisfatoriamente mesmo considerando-se dados ruidosos.

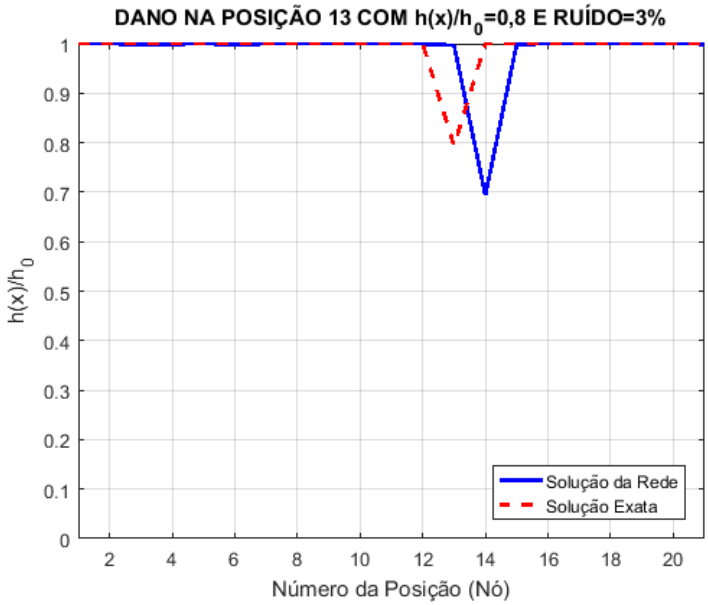

Figura 4.17: Dano na posição 13 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,80$.

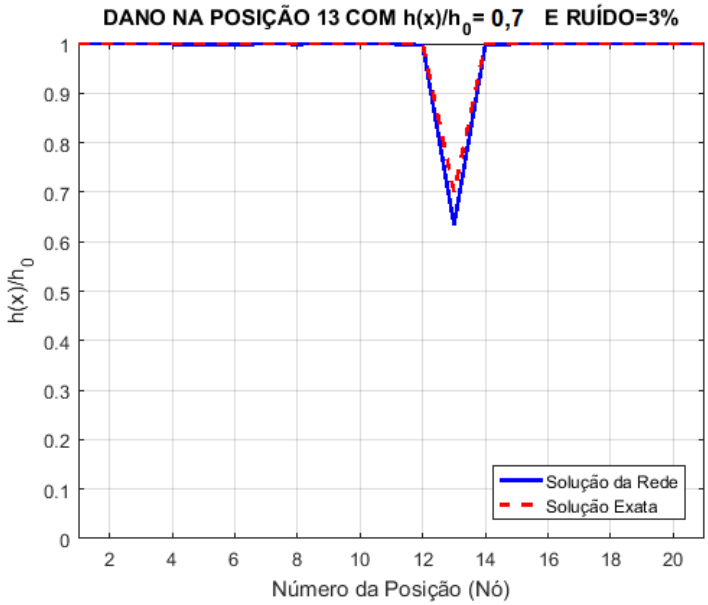

Figura 4.18: Dano na posição 13 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,70$.

4.2.5 Dano na Posição 16

A Figura 4.19 mostra o dano na posição 16 da viga discretizada. Para este nó utilizaram-se as razões $\frac{h(x)}{h_0} = 0,84$ e $\frac{h(x)}{h_0} = 0,70$ com $\beta = 0,600$ e $\beta = 0,343$, respectivamente. Isto significa que os danos impostos neste nó são de 20% e 30% da espessura nominal da viga.

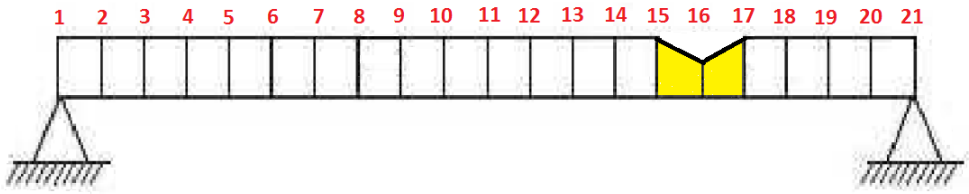

Figura 4.19: Ilustração do dano na posição 16 da viga discretizada.

4.2.5.1 Com Ruído de 0%

Para o caso considerado onde um dano é imposto na posição 16 da discretização da viga de Euler-Bernoulli, o modelo de identificação de danos apresentado conseguiu localizar e quantificar o dano com bastante exatidão (Figuras 4.20 e 4.21). Vale ressaltar que não foram considerados dados contaminados por ruído.

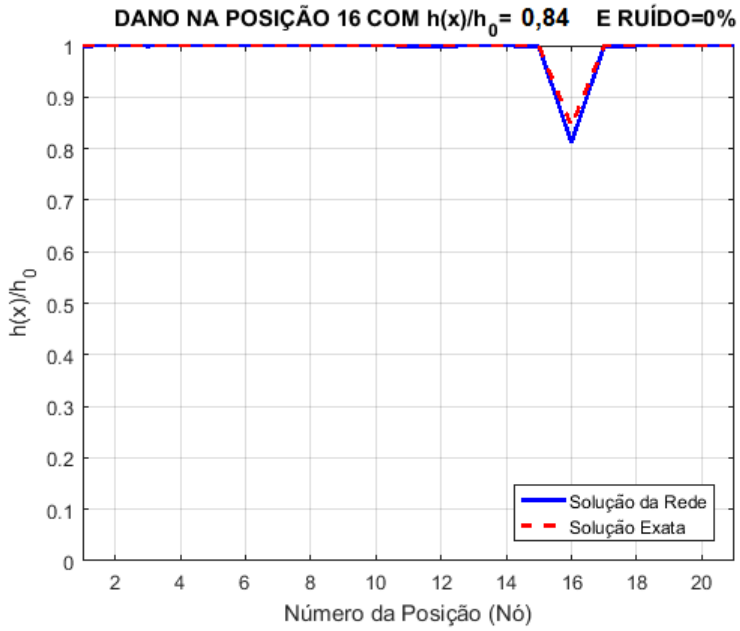

Figura 4.20: Dano na posição 16 com ruído de 0% e $\frac{h(x)}{h_0} = 0,84$.

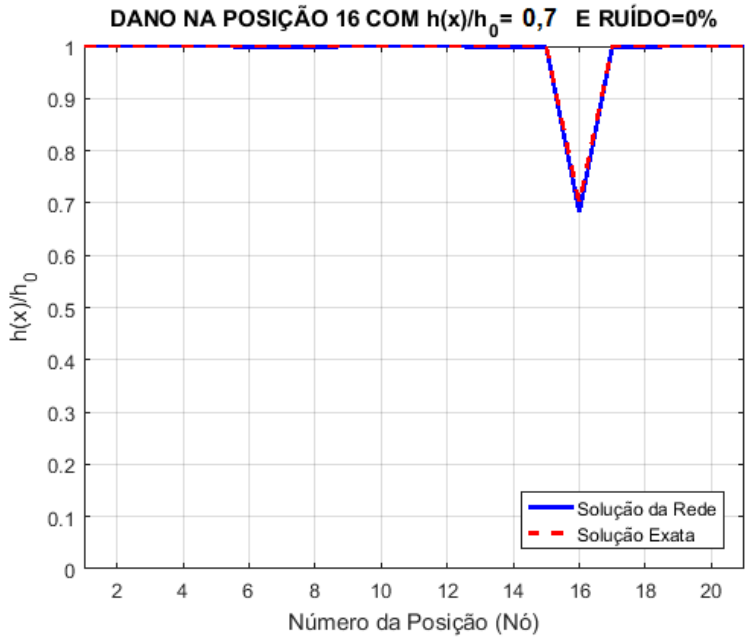

Figura 4.21: Dano na posição 16 com ruído de 0% e $\dfrac{h(x)}{h_0} = 0,70$.

4.2.5.2 Com Ruído de 3%

Considerando-se os dados contaminados com ruído de 3%, percebe-se nas Figuras 4.22 e 4.23 que a região onde a estrutura encontra-se danificada foi localizada, mas em uma posição adjacente ao nó onde foi imposto o dano.

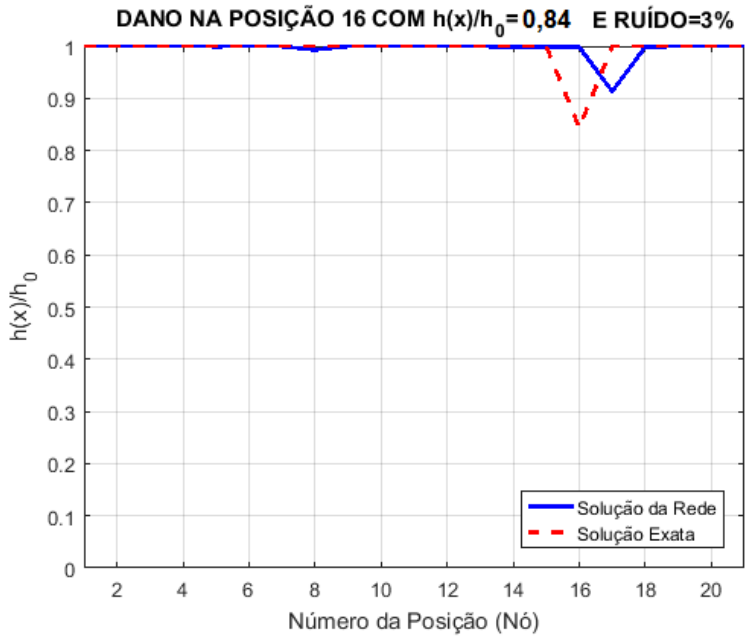

Figura 4.22: Dano na posição 16 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,84$.

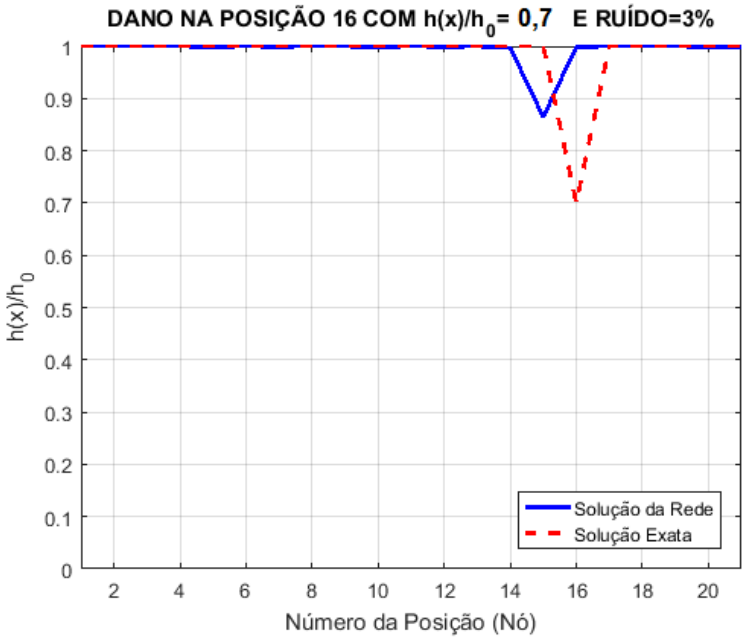

Figura 4.23: Dano na posição 16 com ruído de 3% e $\dfrac{h(x)}{h_0} = 0,70$.

Capítulo 5

Conclusões e Trabalhos Futuros

5.1 Conclusões

No presente trabalho, apresentou-se uma breve motivação para a utilização de redes neurais artificiais para resolver o problema inverso oriundo do modelo de identificação de danos. Mostrou-se também uma revisão bibliográfica onde constatou-se que a identificação de dano num estágio inicial e o monitoramento contínuo da estrutura tem despertado o interesse de diversos pesquisadores, pois possibilitam a redução de custos de manutenção ou reparo, além de aumentar a vida útil da estrutura e a segurança dos que dela dependem.

Esta dissertação teve como objetivo a identificação de danos em uma viga de Euler-Bernoulli, baseada nas características dinâmicas da estrutura usando um modelo de dano continuamente descrito ao longo da estrutura via método dos elemento finitos. A utilização da matriz de flexibilidade para a identificação de dano estrutural mostrou-se bastante adequada, uma vez que é bastante sensível à presença de dano e pode ser obtida a partir de poucos modos de baixa frequência, que são mais facilmente obtidos em um ensaio modal. A identificação de danos via matriz de flexibilidade evita a necessidade de utilização de uma técnica de expansão dos modos obtidos no experimento ou uma redução do modelo para a dimensão dos GDL considerados medidos. Essas dificuldades são frequentemente encontradas em métodos de ajustes de modelos.

A presença de dano foi descrita através de um parâmetro estrutural denominado parâmetro de coesão, utilizando-se para isso a modelagem do campo de danos através do método de elementos finitos. A utilização do parâmetro de coesão para simulação e identificação de danos difere da maioria das abordagens de identificação de danos encontradas na literatura, possibilitando a simulação de danos em estágio inicial de forma mais suave e realística. Por meio desta formulação é possível representar diferentes perfis de dano,

assim como simular uma malha de danos diferente da malha de deslocamentos utilizadas.

Foram analisados os resultados obtidos de diversas simulações de dano na ausência de ruído e na presença de sinal ruidoso. Na presença de ruído de (3%), considerado um nível de ruído elevado segundo a literatura, os resultados foram bastante satisfatórios, pois em todos os casos considerados a região do dano foi encontrada e uma estimativa para a intensidade do dano foi fornecida. Sendo assim, o método proposto é promissor na substituição dos métodos de localização apresentados por Corrêa, 2013 [10], na tentativa de diminuir o número de parâmetros a serem atualizados pelos métodos de otimização quando estruturas mais complexas forem estudadas. Cabe salientar que, diferentemente dos métodos de localização citados, o presente método apresentou resultados satisfatórios na presença de dados ruidosos e com um número reduzido de informação modal. Além disso, o método tem potencial para, sozinho, localizar e quantificar danos em estruturas mais complexas. O ruído adicionado serve para simular qualquer variação entre o modelo computacional e os dados experimentais, como por exemplo, perturbações externas, variações nas propriedades do material, interferência dos equipamentos utilizados na obtenção das frequências naturais e modos de vibração e incertezas do modelo.

Por as Redes Neurais Artificiais, após a fase de treinamento da mesma, ser uma ferramenta que fornece os resultados dos testes solicitados de forma rápida, acredita-se, diante dos casos de danos analisados, que a mesma é uma boa opção para diminuir o tempo gasto com a geração de resultados usando métodos de otimização estocásticos, facilitando assim o monitoramento contínuo da estrutura.

5.2 Trabalhos Futuros

Para trabalhos futuros, sugere-se:

- Estudo do treinamento da rede neural artificial baseado nas regiões da estrutura que são mais sensíveis à presença de danos.
- Utilização de malhas distintas para a discretização de danos e para o campo de deslocamentos.
- Análise do método proposto em identificar danos múltiplos.
- Utilização de métodos de otimização para atualizar os parâmetros da região potencialmente danificada fornecida pela RNA, visando melhorar os resultados obtidos quando considera-se sinais ruidosos.

- Utilização de algum processo para diminuir o nível de ruído nos sinais.
- Expandir o modelo proposto para estruturas mais complexas, como por exemplo uma laje.

Referências

[1] Abraham, A. *Artificial neural networks.* Wiley Online Library, 2005.

[2] Alvin, K., Peterson, L., Park, K. *Method for determining minimum-order mass and stiffness matrices from modal test data*, vol. 33. AIAA journal, 1995.

[3] Anton, H., Anton, H. *Cálculo: um novo horizonte.* Bookman, 2000.

[4] Austrell, P. E., Dahlblom, O., Lindemann, J., Olsson, A., Olsson, K.-G., Persson, K., Petersson, H., Ristinmaa, M., Sandberg, G., Wernberg. *CALFEM - a finite element toolbox.* Version 3.4, Student Litteratur AB, 2004.

[5] Bang, H., Kwon, Y. W. *The finite element method using MATLAB.* CRC press, 2000.

[6] Beer, F. P., Johnston, E. R., Mazurek, D. F., others. *Mecânica dos Materiais-7ª Edição.* AMGH Editora, 2015.

[7] Bellman, R. *Introduction to matrix analysis*, vol. 19. Siam, 1997.

[8] Bittencourt, M. L. *Análise Computacional de Estruturas.* Editora Unicamp, 2010.

[9] Carrillo, O. J. B. *Algoritmo híbrido para avaliação da integridade estrutural: uma abordagem heurística.* Tese de Doutorado, Universidade Federal de São Carlos, Brasil, 152p, 2007.

[10] Corrêa, R. *Identificaçao de danos em estruturas bi-dimensionais via matriz de flexibilidade baseada em um modelo de dano contınuo.* 129 f. Tese de Doutorado em Modelagem Computacional, Instituto Politécnico, Universidade do Estado do Rio de Janeiro, Nova Friburgo, 2013. (http://www.bdtd.uerj.br/tde_busca/arquivo.php?codArquivo=6641 - Acessado em 07/2018).

[11] Cunha, L. N. M. *Identificação de danos em placas utilizando a matriz de flexibilidade.* Tese de Doutorado, Universidade Federal do Rio de Janeiro, 2014.

[12] de Pádua Braga, A., de Leon Ferreira, A. C. P., Ludermir, T. B. *Redes neurais artificiais: teoria e aplicações.* LTC Editora, 2007.

[13] Demuth, H., Beale, M., Hagan, M. *Neural network toolbox.* Mathworks, 1994.

[14] Domício, M. F. *Análise Matricial das Estruturas.* Livros Técnicos e Científicos Editora, 1977.

[15] FITZPATRICK, R. *Oscillations and waves: an introduction.* CRC Press, 2013.

[16] GUERREIRO, L. *Revisões de análise modal e análise sísmica por espectros de resposta.* Reprografia DECivil–Instituto Superior Técnico, 1999.

[17] GUYAN, R. J. *Reduction of stiffness and mass matrices*, vol. 3. AIAA journal, 1965.

[18] HAGAN, M. T., DEMUTH, H. B., BEALE, M. H., OTHERS. *Neural network design*, vol. 20. Pws Pub. Boston, 1996.

[19] HAYKIN, S. *Redes neurais: princípios e prática.* Bookman Editora, 2007.

[20] HEBB, D. O., OTHERS. *The organization of behavior: A neuropsychological theory.* New York: Wiley, 1949.

[21] KELLY, S. G. *Mechanical vibrations: theory and applications.* Cengage learning, 2012.

[22] MAIA, R. A. *Detecção de dano estrutural em estruturas planas utilizando redes neurais.* 2017.

[23] MASCIA, N. T. *Energia de deformação e teoremas da energia.* UNICAMP, 2017.

[24] MCCULLOCH, W. S., PITTS, W. *A logical calculus of the ideas immanent in nervous activity*, vol. 5. Springer, 1943.

[25] MIGUEL, L. F. F. *Identificaçao de sistemas e avaliaçao da integridade de estruturas treliçadas.* Tese de Doutorado, 2007.

[26] MINSKY, M., PAPERT, S. A. *Perceptrons: an introduction to computational geometry.* MIT press, 2017.

[27] NIETO MARTÍNEZ, M. *Modelagem da detecção de danos via elementos finitos e redes neurais artificiais.* 2013.

[28] OBERZINER, A. P. B. As equações de maxwell e aplicações, florianópolis, sc, 2008.

[29] PAIN, H. J., PAIN, H. J. *The physics of vibrations and waves*, vol. 2. Wiley Online Library, 2005.

[30] PRZEMIENIECKI, J. S. *Theory of matrix structural analysis.* Courier Corporation, 1985.

[31] RAO, S. *Vibrações Mecânicas.* 4ª Edição. Pearson Premtice Hall. São Paulo, 2009.

[32] ROSENBLATT, F. *The perceptron: a probabilistic model for information storage and organization in the brain.*, vol. 65. American Psychological Association, 1958.

[33] RUMELHART, D. E., MCCLELLAND, J. L., GROUP, P. R., OTHERS. *Parallel distributed processing*, vol. 1. MIT press Cambridge, MA, 1987.

[34] RYTTER, A. *Vibrational based inspection of civil engineering structures.* Dept. of Building Technology and Structural Engineering, Aalborg University, 1993.

[35] SCHALKOFF, R. J. *Artificial neural networks*, vol. 1. McGraw-Hill New York, 1997.

[36] SILVA, I. N. D., SPATTI, H., FLAUZINO, R. *Redes neurais artificiais, curso prático para engenharia e ciências aplicadas*. Artliber Editora, 2010.

[37] SOBRINHO, A. D. S. C. *Introdução ao método dos elementos finitos*. Ciência Moderna, 2006.

[38] SORIANO, H. L. *Análise de estruturas: formulação matricial e implementação computacional*. Ciência Moderna, 2005.

[39] SORIANO, H. L. *Elementos finitos: formulação e aplicação na estática e dinâmica das estruturas*. Ciência Moderna, 2009.

[40] STEWART, J. Cálculo, vol. 1. *Pioneira Thomson Learning* (2001).

[41] STUTZ, L. *Identificação de defeitos estruturais via atualização da matriz de flexibilidade baseada num modelo de dano contínuo*. 74 f. Dissertação de Mestrado em Engenharia Mecânica, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 1999.

[42] STUTZ, L., CASTELLO, D., ROCHINHA, F. *A flexibility-based continuum damage identification approach*, vol. 279. Elsevier, 2005.

[43] YEGNANARAYANA, B. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.