

**UNIVERSIDADE FEDERAL DE PERNAMBUCO  
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS  
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA**

**ESTUDO DO ACOPLAMENTO DE GRUPOS  
MOTOR-GERADOR COM UNINTERRUPTIBLE  
POWER SUPPLY APLICANDO WAVELETS E  
REDES NEURAIS ARTIFICIAIS**

por

**Gustavo Maciel dos Santos**

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da  
Universidade Federal de Pernambuco como parte dos requisitos para a obtenção do grau de  
Mestre em Engenharia Elétrica.

**ORIENTADOR: Prof. RONALDO R. B. DE AQUINO, D.Sc., UFPE**

Recife, julho de 2011

© Gustavo Maciel dos Santos, 2011

Catálogo na fonte  
Bibliotecária Rosineide Mesquita Gonçalves Luz / CRB4-1361 (BCTG)

S237e Santos, Gustavo Maciel dos.  
Estudo do acoplamento de grupos Motor-Gerador com Uninterruptible Power Supply aplicando Wavelets e Redes Neurais Artificiais. / Gustavo Maciel dos Santos. - Recife: O Autor, 2011.  
xii, 71f., il., figs., gráfs., tabs.

Orientador : Prof. Dr. Ronaldo R. B. de Aquino.  
Dissertação (Mestrado) – Universidade Federal de Pernambuco.  
CTG. Programa de Pós-Graduação em Engenharia Elétrica, 2011.  
Inclui Referências Bibliográficas e Anexos.

1. Engenharia Elétrica 2. Rede Neural Artificial. 3. RNA 4. *Wavelets*.  
5. Motor-Gerador. 6. GMG. 6. UPS. 7. *No-break*. I. Aquino, Ronaldo R.B.de. (Orientador). II. Título.

621.3 CDD (22.ed) UFPE/BCTG-182/2011



**Universidade Federal de Pernambuco**  
***Pós-Graduação em Engenharia Elétrica***

PARECER DA COMISSÃO EXAMINADORA DE DEFESA DE  
DISSERTAÇÃO DO MESTRADO ACADÊMICO DE

**GUSTAVO MACIEL DOS SANTOS**

TÍTULO

**“ESTUDO DO ACOPLAMENTO DE GRUPOS MOTOR-GERADOR COM  
UNINTERRUPTIBLE POWER SUPPLY APLICANDO WAVELETS  
E REDES NEURAIS ARTIFICIAIS”**

A comissão examinadora composta pelos professores: RONALDO RIBEIRO BARBOSA DE AQUINO, DEE/UFPE, MANOEL AFONSO DE CARVALHO JÚNIOR, DEE/UFPE, e MILDE MARIA DA SILVA LIRA, DEE/UFPE sob a presidência do primeiro, consideram o candidato **GUSTAVO MACIEL DOS SANTOS** **APROVADO.**

Recife, 08 de julho de 2011

---

**RAFAEL DUEIRE LINS**  
Coordenador do PPGE

---

**RONALDO RIBEIRO BARBOSA DE AQUINO**  
Orientador e Membro Titular Interno

---

**MILDE MARIA DA SILVA LIRA**  
Membro Titular Externo

---

**MANOEL AFONSO DE CARVALHO JÚNIOR**  
Membro Titular Externo

*A meu pai Gildo Torres dos Santos (in  
memorian), por me tornar o engenheiro que  
ele sempre sonhou*

## **AGRADECIMENTOS**

Existem situações na vida em que, mesmo com todos os nossos esforços, não seguimos adiante sem o auxílio de algumas pessoas. Participar do programa de pós-graduação era um deles. Para este passo ser possível, agradeço ao professor Dr. Ronaldo Aquino, orientador deste trabalho, pela oportunidade, seu apoio e sua atenção.

Agradeço ao Prof. Manoel Afonso, por nossas conversas de fim de aula, pois dentre uma dessas surgiu a inspiração para o tema desse trabalho.

Agradeço a Prof<sup>a</sup> Milde Maria pela disponibilidade e empenho em me ajudar no decorrer desse trabalho.

Agradeço aos amigos Carlos Frederico e Wladimir Moura, companheiros de jornada. Formamos uma estrutura de colaboração com três pilares nos quais uns sustentavam os outros em todas as dificuldades. Sem isso esse trabalho não seria possível.

Agradeço a minha mãe Conceição pela motivação constante em minha vida.

Agradeço a minha esposa Paula e filhas Maria Beatriz e Isabela, pela paciência e colaboração durante o desenvolvimento desse trabalho.

Agradeço a todos que de alguma forma auxiliaram neste projeto.

*“O sucesso é construído à noite!*

*Durante o dia você faz o que todos fazem. Mas, para obter resultado diferente da maioria, você tem que ser especial. Se fizer igual a todo mundo, obterá os mesmos resultados. Não compare à maioria, pois infelizmente ela não é modelo de sucesso.”*

*(Roberto Shinyashiki)*

Resumo da Dissertação apresentada à UFPE como parte dos requisitos necessários para a obtenção do grau de Mestre em Engenharia Elétrica.

# **Estudo do Acoplamento de Grupos Motor-Gerador com Uninterruptible Power Supply Aplicando Wavelets e Redes Neurais Artificiais**

**Gustavo Maciel dos Santos**

Julho/2011

Orientador: Prof. RONALDO R. B. DE AQUINO, D.Sc., UFPE

Área de Concentração: Processamento de energia.

Palavras-chave: Rede Neural Artificial, *Wavelets*, Motor-Gerador, UPS, *No-break*.

Número de Páginas: 82

RESUMO: Com a sociedade globalizada e a evolução da informática, o conceito de horário comercial não é mais simples. Transações bancárias, negociações na bolsa de valores e compras pelo computador ocorrem em qualquer horário e as empresas precisam garantir a disponibilidade de seus serviços sempre. Para garantir o fornecimento ininterrupto de energia para estes sistemas de informações são utilizados grupos motor-gerador (GMG) e *uninterruptible power supply* (UPS). Quando há dificuldade de acoplamento entre os dois equipamentos, na prática, os projetistas superdimensionam o GMG em relação ao UPS sem embasamento teórico para esta ação. Este estudo propõe apresentar a correlação entre os dados de placa desses equipamentos e o distúrbio de tensão entre eles para proporcionar uma maior segurança no dimensionamento de sistemas com UPS e GMG. A metodologia desse estudo propõe o uso de Redes Neurais Artificiais (RNA's) para descrever esta correlação e transformada *wavelet* para destacar os distúrbios medidos e auxiliar na convergência da RNA em seu treinamento. Os resultados obtidos com as redes desenvolvidas neste trabalho sugerem que o menor distúrbio no sinal, obtido devido ao acoplamento de UPS e GMG, ocorrerá se as potências dos equipamentos (dados de placa) forem próximas, independente do nível de carregamento do sistema. À medida que esta relação se distancia do valor unitário, a perturbação aumenta e um UPS com potência menor que o GMG contribui mais com a distorção do sinal do que ao contrário.

Abstract of Dissertation presented to UFPE as part of the requirements to the degree of  
Master of Electrical Engineering

# **Study of the Coupling of Groups Motor-Generators with Uninterruptible Power Supply Using Wavelets and Artificial Neural Network**

**Gustavo Maciel dos Santos**

July/2011

Supervisor: Prof. RONALDO R. B. DE AQUINO, D.Sc., UFPE

Area of Concentration: Processing Energy.

Keywords: Artificial Neural Network, Wavelet Transform, UPS, Motor-Generator

Number of Pages: 82

**ABSTRACT:** Through out the globalized society and the evolution of the computer science, the concept of commercial time-table is no more simple. Bank transactions, negotiations in the Stock Exchange and internet purchases take place in anyone time-table and the enterprises need always to guarantee the availability of their services. To guarantee the continuous supply of energy for these information systems, groups motor-generators (GMG) and uninterruptible power supply (UPS) are used. When there is difficulty of coupling between two equipments, in practice, the designers supercalculate the size of the GMG regarding the UPS without theoretical foundation for this action. This study proposes to present the correlation between the data of plate of these equipments and the voltage disturbance between them to provide a bigger security in the projects of systems with UPS and GMG. The methodology of this study proposes the use of Artificial Neural Networks (ANN's) to describe this correlation and Wavelet Transform to highlight the measured disturbances in order to help in the convergence of the ANN during the training process. The results obtained with the networks developed in this study suggest that the minor disturbance in the signal due to the coupling of UPS and GMG, will take place if the powers of the equipments (data plate) are approximately equal, independently of the level of load of the system. When this relation is not close to the unit value, the disturbance increases and an UPS with power less than the GMG contributes more with the distortion of the signal then on the contrary.



## SUMÁRIO

LISTA DE FIGURAS .....	X
LISTA DE TABELAS .....	XII
CAPÍTULO 1 .....	1
1 INTRODUÇÃO.....	1
1.1 Descrição do Problema.....	2
1.2 Justificativa.....	3
1.3 Objetivos.....	3
1.3.1 Objetivos Gerais .....	3
1.3.2 Objetivos Específicos .....	3
1.4 Metodologia.....	4
1.5 Estrutura do Trabalho .....	4
CAPÍTULO 2 .....	5
2 TEORIA <i>WAVELET</i> .....	5
2.1 Wavelets .....	6
2.2 Detectando Descontinuidade .....	13
CAPÍTULO 3 .....	15
3 REDES NEURAS ARTIFICIAIS.....	15
3.1 Introdução.....	15
3.2 Aprendizagem.....	22
3.2.1 Aprendizagem Por Correção De Erro.....	24
3.2.2 Aprendizagem baseada em memória.....	24
3.2.3 Aprendizagem Hebbiana .....	25
3.2.4 Aprendizagem competitiva.....	25
3.2.5 Aprendizagem de Boltzmann .....	26
3.3 Rede Neural Backpropagation.....	27
3.3.1 Introdução.....	27
3.3.2 Ajuste dos Pesos .....	29
3.3.3 Observações.....	32
3.4 Projeto de uma Rede Neural.....	33
3.5 Vantagens .....	34
3.6 Aplicações .....	35
CAPÍTULO 4 .....	36
4. APLICAÇÃO DA METODOLOGIA PROPOSTA .....	36
4.1 Introdução.....	36
4.2 Ferramentas e Equipamentos Utilizados .....	36
4.3 Banco de Dados .....	36
4.4 Variáveis da Rede Neural.....	38
4.5 Preparação dos Dados.....	39
4.6 Rede Neural .....	42
4.6.1 Introdução.....	42
4.6.2 Embaralhar os Dados.....	42

4.6.3 Normalização dos Dados .....	43
4.6.4 Representação dos Padrões de Saída .....	45
4.6.5 Arquitetura da Rede.....	47
4.6.6 Melhor Inicialização de Pesos .....	48
4.6.7 Projeto Ótimo de Rede .....	49
4.7 Utilizando a Rede Neural .....	51
4.7.1 Variáveis de Entrada.....	51
4.7.2 Tratamento das Variáveis de Saída .....	53
4.8 Resultados.....	54
5. CONCLUSÕES E SUGESTÕES PARA TRABALHOS FUTUROS .....	56
ANEXO 1 – GRÁFICOS DAS REDES TREINADAS .....	58
REFERÊNCIAS BIBLIOGRÁFICAS .....	69

## LISTA DE FIGURAS

<b>Figura - 1.1:</b> UPS Paralelo Redundante.....	2
<b>Figura - 2.1:</b> Jean Morlet e Ingrid Daubechies .....	6
<b>Figura - 2.2:</b> (a) Função escala de Haar, (b) Função <i>wavelet</i> de Haar .....	8
<b>Figura - 2.4:</b> (a) Função escala de Meyer; (b) Função <i>wavelet</i> de Meyer .....	10
<b>Figura - 2.5:</b> (a) Função escala de Daubechies; (b) Função <i>wavelet</i> de Daubechies.....	12
<b>Figura - 2.6:</b> Sinal com descontinuidade .....	13
<b>Figura - 3.1:</b> Modelo do neurônio artificial .....	16
<b>Figura - 3.3:</b> Rede alimentada adiante ou acíclica com uma única camada de neurônios .	19
<b>Figura - 3.4:</b> Rede alimentada adiante ou acíclica totalmente conectada com uma camada oculta e uma camada de saída.....	20
<b>Figura - 3.5:</b> Rede recorrente sem laços de auto-realimentação e sem neurônios ocultos .	21
<b>Figura - 3.6:</b> Rede recorrente com neurônios ocultos.....	21
<b>Figura - 3.7:</b> Manifestação do supertreinamento.....	23
<b>Figura - 3.8:</b> Grafo arquitetural de um perceptron de múltiplas camadas com duas camadas ocultas.....	27
<b>Figura - 3.9:</b> Correção dos pesos por <i>backpropagation</i> .....	29
<b>Figura - 3.10:</b> Erro em função do peso para uma única conexão .....	31
<b>Figura - 4.1:</b> Software ANL6000 apresentando um dos 26 registros de transitórios identificados em tela.....	37
<b>Figura - 4.2:</b> Variáveis propostas para a rede neural (arquitetura 1).....	38
<b>Figura - 4.3:</b> Variáveis propostas para a rede neural com <i>Wavelet</i> (arquitetura 2) .....	39
<b>Tabela - 4.1:</b> Cabeçalho da tabela de dados.....	39
<b>Figura - 4.4:</b> Exemplo da aplicação de $\Delta$ para ajustar o sinal fundamental .....	40
<b>Figura - 4.5:</b> Trecho da tabela de dados no Matlab. ....	41
<b>Figura - 4.6:</b> Amostra gráfica das variáveis utilizadas na formação da rede neural.....	41
<b>Figura - 4.7:</b> Amostra gráfica das variáveis utilizadas na formação da rede neural após serem embaralhadas.....	43
<b>Figura - 4.8:</b> Gráfico de desempenho – Arquitetura Final 1.....	49
<b>Figura - 4.9:</b> Gráfico de desempenho – Arquitetura Final 2.....	49
<b>Figura - 4.10:</b> Gráfico de regressão – Arquitetura 1.....	50
<b>Figura - 4.11:</b> Gráfico de regressão – Arquitetura 2.....	51
<b>Figura - 4.12:</b> Amostra gráfica das variáveis de entrada utilizadas na rede neural .....	52

<b>Tabela -4.6:</b> Intervalos para cada ciclo em um conjunto de 2.134 registros .....	53
<b>Figura - 4.13:</b> Resposta da rede: Arquitetura 1.....	54
<b>Figura - 4.14:</b> Resposta da rede: Arquitetura 2.....	54
<b>Figura - 4.15:</b> Resposta da rede: Arquitetura 1 (resultado amortizado) .....	55
<b>Figura - A.1:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 10% .....	59
<b>Figura - A.2:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 10%(escala logarítmica).....	59
<b>Figura - A.3:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 20% .....	60
<b>Figura - A.4:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 20%(escala logarítmica).....	60
<b>Figura - A.5:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 30% .....	61
<b>Figura - A.6:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 30%(escala logarítmica).....	61
<b>Figura - A.7:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 40% .....	62
<b>Figura - A.8:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 40%(escala logarítmica).....	62
<b>Figura - A.9:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 50% .....	63
<b>Figura - A.10:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 50%(escala logarítmica).....	63
<b>Figura - A.11:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 60% .....	64
<b>Figura - A.12:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 60%(escala logarítmica).....	64
<b>Figura - A.13:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 70% .....	65
<b>Figura - A.14:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 70%(escala logarítmica).....	65
<b>Figura - A.15:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 80% .....	66
<b>Figura - A.16:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 80%(escala logarítmica).....	66
<b>Figura - A.17:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 90% .....	67
<b>Figura - A.18:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 90%(escala logarítmica).....	67
<b>Figura - A.19:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 100% .....	68
<b>Figura - A.20:</b> Resposta da rede: Arquitetura 1 e 2 com carregamento de 100%(escala logarítmica).....	68

## LISTA DE TABELAS

<b>Tabela - 4.1:</b> Cabeçalho da tabela de dados.....	39
<b>Tabela - 4.2:</b> Valores máximos e mínimos utilizados na normalização dos dados .....	44
<b>Tabela - 4.3:</b> Erros médios quadráticos dos testes.....	46
<b>Tabela - 4.4:</b> Média dos erros médios quadráticos variando os neurônios na camada oculta .....	47
<b>Tabela - 4.5:</b> Erros médios quadráticos verificados para cada inicialização aleatória das redes após definido o nº de neurônios na camada oculta .....	48
<b>Tabela -4.6:</b> Intervalos para cada ciclo em um conjunto de 2.134 registros .....	53

## CAPÍTULO 1

### 1 INTRODUÇÃO

Com a evolução da informática, é cada vez maior o número de componentes eletrônicos utilizados em residências, empresas e indústrias. Com um mercado mais competitivo, as empresas têm utilizado das facilidades proporcionadas pela informática, como a centralização de banco de dados, utilização de softwares para controle de produção, comunicação em redes de computadores, redes sem fio, entre outras.

Com equipamentos mais precisos, a qualidade de energia é fundamental para o perfeito funcionamento dos *hardware*. Energia estabilizada se torna um item essencial para o perfeito funcionamento desses componentes.

Neste século 21, as necessidades dos clientes vão além dos horários comerciais. Há pessoas utilizando caixas-rápidos às 2h00, atuando na bolsa de valores do Japão às 23h00, fazendo compras em lojas virtuais às 3h00. No Natal há inclusive shoppings funcionando 24 horas ininterruptas.

As indústrias trabalham 24 horas. Transações bancárias são realizadas nas madrugadas. Sistemas informatizados atualizam banco de dados nos horários em que não há expediente funcional.

Com isso, além da energia estabilizada ser necessária para o perfeito funcionamento desses componentes, ela agora deve ser também ininterrupta. Para atender a essa necessidade, torna-se comum o uso de grupos motor-gerador (GMG) e *Uninterruptible Power Supply* (UPS)

Os UPS's possuem sua autonomia limitada pelas baterias de seu sistema, pré-definido no seu processo construtivo, e podem funcionar *on-line* com as cargas (as cargas não percebem a falta) fornecendo energia estabilizada necessária para o perfeito funcionamento de componentes eletrônicos delicados.

Os GMG's funcionam enquanto houver combustível no sistema, o qual pode ser repostado indefinidamente mesmo com o GMG em funcionamento, porém não garante energia estabilizada na qualidade necessária.

Neste cenário, um sistema com GMG em série com UPS se torna fundamental para quem busca energia de qualidade e ininterrupta com segurança.

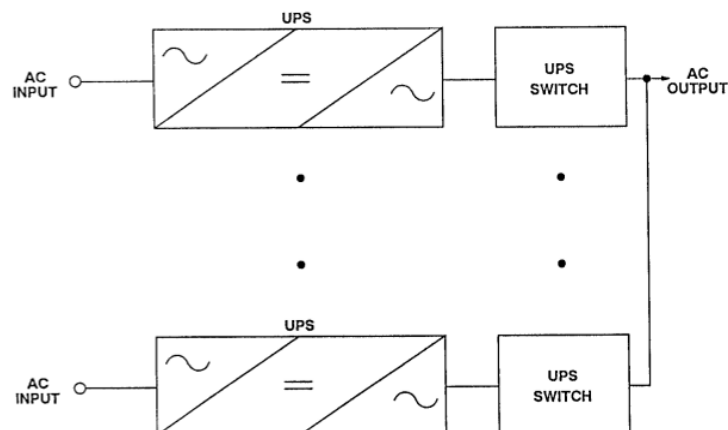
## 1.1 Descrição do Problema

Ao se ligar o GMG, inicialmente se tem a partida do motor, e apenas com o motor na velocidade nominal é que a carga é conectada ao sistema. O GMG não sabe qual a intensidade e nem o tipo da carga que será conectado, e neste instante (transitório) o GMG pode não apresentar as características de saída ideais para alimentação da carga.

Os UPS's que trabalham para sistemas de precisão normalmente têm as seguintes características:

- São on-line com Dupla Conversão: estão sempre alimentando a carga com energia condicionada após o inversor
- Possuem by-pass automático: desligam o UPS e ligam direto a carga com a rede da concessionária se for identificada alguma falha no sistema.
- Utilizam retificador controlado por transistor bipolar de porta isolada (IGBT)
- Monitoram a rede de entrada e desligam o sistema se for identificado algum risco a integridade dos componentes internos.

O conjunto GMG - UPS se torna um exercício delicado para seu dimensionamento. Normalmente os UPS's não alimentam cargas lineares e sim cargas que deformam a forma de onda. Arranjos de UPS's como em paralelo-redundante (Figura 1.1), situação em que há dois ou mais UPS's em paralelo, dimensionados de tal forma que na falta de um deles, os demais suportariam a carga atendida, há mais distorção na forma de onda da rede elétrica. Assim, na necessidade do acoplamento entre o GMG e UPS há situações em que o sistema UPS é auto-desligado, "rejeitando" o GMG.



**Figura - 1.1:** UPS Paralelo Redundante

Fonte: (INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, 1996)

Com estes fenômenos, o projetista superdimensiona o GMG em relação à potência dos UPS's, entre 20% e 30% em média, com o objetivo de não ser surpreendido com esta rejeição.

## **1.2 Justificativa**

Não há estudos que comprovem que exista correlação entre a potência do GMG com a potência dos UPS's a serem alimentados.

Superdimensionamento de GMG onera os projetos, jogam mais energia reativa no sistema elétrico, além de impactar negativamente com a política ambiental em racionar recursos.

Um estudo neste tema pode proporcionar alternativas tecnológicas para que exista um perfeito acoplamento entre GMG e UPS.

## **1.3 Objetivos**

### **1.3.1 Objetivos Gerais**

- Analisar as grandezas elétricas durante o transitório do chaveamento do GMG com o UPS.
- Levantar a curva da relação da potência do GMG pela potência do UPS em função da distorção harmônica total (THD) do sinal de tensão entre estes componentes.

### **1.3.2 Objetivos Específicos**

- Coletar as grandezas elétricas durante o transitório do chaveamento do GMG com o UPS.
- Utilizar Transformadas *Wavelets* para destacar as distorções das formas de onda de energia elétrica.
- Utilizar Redes Neurais para correlacionar a relação da potência do GMG pela potência do UPS em função da THD do sinal de tensão entre eles.



## 1.4 Metodologia

Serão coletadas as grandezas elétricas do transitório de chaveamento de sete GMG's em operação, os quais possuem UPS em seu circuito. Juntamente com esta medição, será levantado o nível de carregamento do sistema elétrico de cada GMG.

De posse dos transitórios, utilizaremos transformadas *Wavelets* para destacar as distorções da forma de onda originadas do chaveamento do GMG com UPS em seu circuito, o qual alimenta cargas não lineares.

Utilizaremos Redes Neurais para identificar a correlação da relação da potência do GMG pela potência do UPS em função da THD do sinal de tensão. A Rede Neural terá como dados de entrada uma fundamental, a potência de cada equipamento (GMG e UPS), e o nível de carregamento do sistema. O dado de saída é o sinal medido, o qual está distorcido devido ao chaveamento do GMG com o UPS em série. A saída será representada também por sinais resultantes da decomposição do sinal medido no 1º nível através da *Wavelet* com a hipótese de que assim obteremos a melhor arquitetura para formação da Rede Neural.

Com a Rede Neural treinada serão feitas simulações com variados tipos de carregamento e potência dos equipamentos e assim será traçada a curva de THD em função da relação entre a potência do GMG e a potência do UPS para vários níveis de carregamento do sistema.

## 1.5 Estrutura do Trabalho

Este trabalho está organizado em cinco capítulos, que serão descritos a seguir. Neste primeiro capítulo descrevem-se as linhas que serão seguidas para o desenvolvimento deste trabalho, com a justificativa, objetivos e importância.

No capítulo 2 apresenta-se uma revisão de literatura em transformadas *wavelets*, dando-se um enfoque na escolha da *wavelet* para representar o momento e intensidade do distúrbio do acoplamento GMG e UPS

No capítulo 3 tem-se uma revisão de literatura em Redes Neurais, destacando-se o uso desta técnica para correlacionar banco de dados.

No capítulo 4, apresenta-se o desenvolvimento da metodologia proposta.

Finalizando, no capítulo 5 apresenta-se a conclusão com as devidas considerações.

## CAPÍTULO 2

### 2 TEORIA WAVELET

Uma função *wavelet* é a interpretação de uma onda de curta duração com crescimento e decrescimento rápidos. Sua teoria baseia-se na representação de funções em diferentes escalas e diferentes resoluções (tempo-escala), considerando assim uma das suas principais características (DAUBECHIES, 1992). A teoria *wavelet*, que se constitui um método para decomposição e reconstrução de sinais, emprega funções de base que são locais nos domínios do tempo e da frequência (KIM, 2000), características estas que a torna superior à transformada de Fourier na análise de sinais transitórios (GU, 2000, POISSON, 2000).

A transformada *wavelet* representa um sinal como uma soma de “ondinhas” em diferentes escalas e localizações, cujos coeficientes maximizam as contribuições destas “ondinhas” em cada uma destas escalas e localizações. (LIRA, 2004, p. 2).

O primeiro registro do termo "*wavelet*" data de 1909, em uma tese de Alfred Haar (OLIVEIRA, 2007), apresentando uma função que, décadas depois, viria a ser conhecida como a primeira função *wavelet*.

O conceito de *wavelet*, em sua forma teórica atual, foi proposto em meados dos anos oitenta por Jean Morlet (geofísico), Yves Meyer (matemático) e a equipe do Centro de Física Teórica de Marseille, trabalhando sob orientação de Alex Grossman (físico teórico) na França. Os métodos de análise *wavelet* foram desenvolvidos principalmente por Yves Meyer e seus colegas, que asseguraram a sua disseminação.

A atenção da comunidade de processamento de sinais foi atraída quando Ingrid Daubechies e Stephane Mallat, além de suas contribuições para a teoria de *wavelets*, estabeleceram a conexão entre os dois assuntos e obtiveram resultados via processamento de sinal discreto. A Figura 2.1 apresenta Jean Morlet e Ingrid Daubechies.

O algoritmo de Mallat é considerado o elo definitivo entre *wavelets* e processamento de sinais. Desde então, pesquisas em *wavelets* tornaram-se difundidas internacionalmente (RIOUL; VETTERLI, 1991).



**Figura - 2.1:** Jean Morlet e Ingrid Daubechies  
Fonte: (OLIVEIRA, 2007)

## 2.1 Wavelets

As funções *wavelets*, geralmente denotadas por  $\psi(t)$  são definidas como um conjunto de funções originadas através das operações matemáticas de translação e escalonamento da função escala, com propriedades particulares que as tornam adequadas para servirem de base para a decomposição de outras funções (FARIA, 1997).

A função escala (Figura 2.2 (a) ) é uma função básica definida num espaço  $V_j$ , usualmente denotada por  $\phi$  tendo como funções básicas associadas:

$$\phi_i^j(t) := \phi(2^j t - i), \quad i = 0, \dots, 2^j - 1 \quad (2.1)$$

sendo:

$\phi$  : função escala;

$i$  : deslocamento;

$j$  : escala;

$t$  : tempo.

Para que uma função seja considerada uma *wavelet* é preciso satisfazer as seguintes condições básicas e necessárias:

1. que  $\psi(t) \in L^2(\mathfrak{R})$ , ou seja, a função pertença ao espaço das funções de quadrado integrável ou, ainda, o espaço das funções de energia finita, no sentido que:

$$\int_{-\infty}^{\infty} |\psi(t)|^2 dt < \infty \quad (2.2)$$

2. que sua Transformada de Fourier  $\hat{\psi}(\omega)$  satisfaça a condição de admissibilidade (DAUBECHIES, 1992):

$$C_{\psi} = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(\omega)|^2}{|\omega|} d\omega < \infty \quad (2.3)$$

Segue da condição de admissibilidade que:

$$\lim_{\omega \rightarrow 0} \hat{\psi}(\omega) = 0 \quad (2.4)$$

Assim, se  $\hat{\psi}(\omega)$  é contínua então  $\hat{\psi}(0) = 0$ , ou seja,

$$\int_{-\infty}^{\infty} \psi(\omega) d\omega = 0 \quad (2.5)$$

Geometricamente, a condição (2.3) estabelece que o gráfico de  $\psi(t)$  deve oscilar de modo a cancelar as áreas negativas a fim de anular a integral (2.5). Portanto, o gráfico de  $\psi(t)$  tem a forma de onda, conforme ilustra a Figura 2.2 (b), que é um exemplo de *wavelet*.

Desde que  $\psi(t)$  esteja bem localizada no tempo, este decaimento será muito rápido, formando uma onda de curta duração.

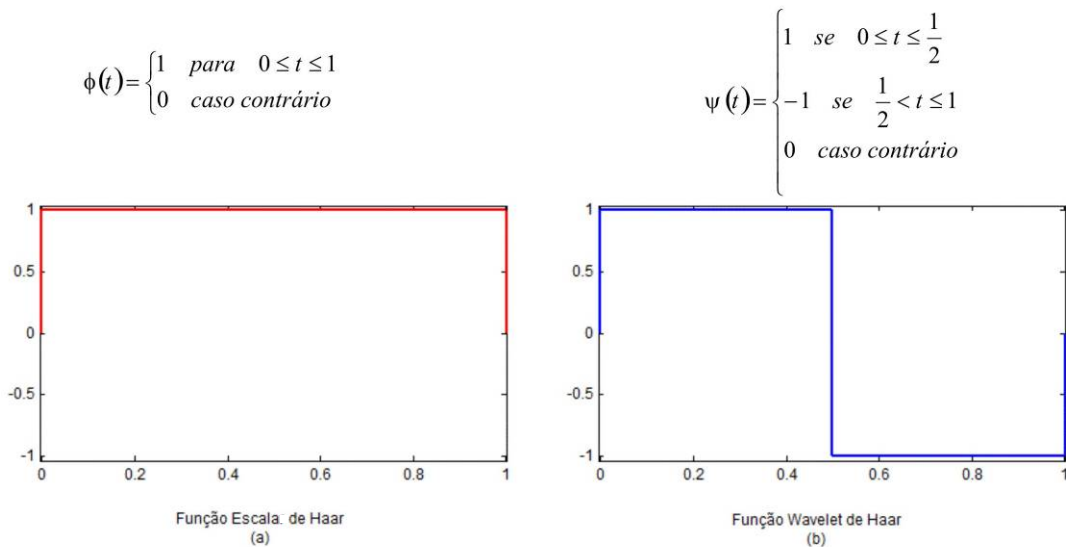
A Transformada *Wavelet* Contínua essencialmente mapeia um sinal unidimensional (no tempo) em uma representação bidimensional (tempo, escala) que é altamente redundante. As Transformadas *Wavelets* Discretas são mais atraentes do ponto de vista de implementação computacional. Elas não são transladadas nem escalonadas continuamente, mas sim em intervalos discretos (BULTTHEEL, 1995, MALLAT, 1989).

Atualmente, existem inúmeras funções *wavelets* que geralmente recebem o nome de seus criadores, dentre as quais serão apresentadas, a seguir, as mais conhecidas.

Começando pelo exemplo mais simples, proposto em 1909 pelo matemático húngaro Alfred Haar. A *wavelet* de Haar demonstra as grandezas que envolvem os valores de forma não contínua, tornando-se deste modo um caso particular da transformada *wavelet* discreta definida por:

$$\psi_i^j(t) = \phi(2t) - \phi(2t - 1) \quad (2.6)$$

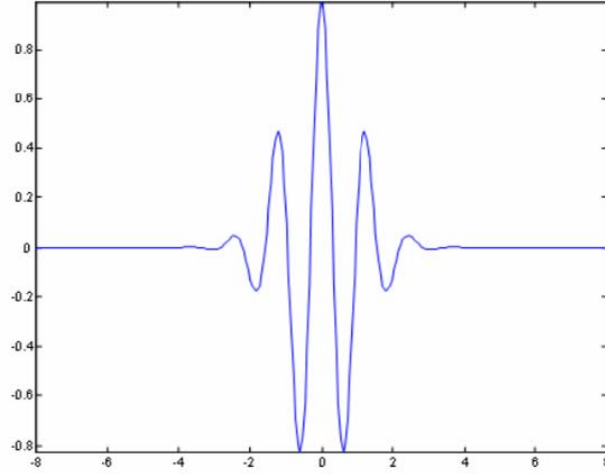
É através da equação (2.6) que podemos obter  $\psi(t)$ , apresentada na Figura 2.2 (b).



**Figura - 2.2:** (a) Função escala de Haar, (b) Função *wavelet* de Haar

Outra função é a *wavelet* de Morlet, introduzida por Jean Morlet pertence à família das *wavelets* não-ortogonais. A *wavelet* de Morlet não possui função escala e é explícita por uma Gaussiana modulada (*shifted*), levemente ajustada. De forma que  $\psi(0) = 0$  conforme a equação (2.7), cujo gráfico é apresentado na Figura 2.3:

$$\psi(t) = Ce^{-\frac{t^2}{2}} \cos(5t) \quad (\text{MISITI } et \text{ al, 2002}). \quad (2.7)$$



**Figura - 2.3:** Função *wavelet* de Morlet

Yves Meyer em 1980 construiu a primeira *wavelet* trivial diferente da *wavelet* de Haar, que é continuamente diferenciável, o que limita suas aplicações (SILVA; EYNG, 2000). Desta forma, uma base *wavelet* suave ortonormal foi criada. Primeiro, definiu-se a Transformada de Fourier  $\hat{\phi}(t)$  de uma função escala  $\phi(t)$  como:

$$\hat{\phi}(t) = \begin{cases} (2\pi)^{-1/2}, & \text{se } |t| \leq \frac{2\pi}{3} \\ (2\pi)^{-1/2} \cos\left[\frac{\pi}{2} v\left(\frac{3}{4\phi\pi}|t| - 1\right)\right], & \text{se } \frac{2\pi}{3} \leq |t| \leq \frac{4\pi}{3} \\ 0, & \text{se } |t| > \frac{4\pi}{3} \end{cases}, \quad (2.8)$$

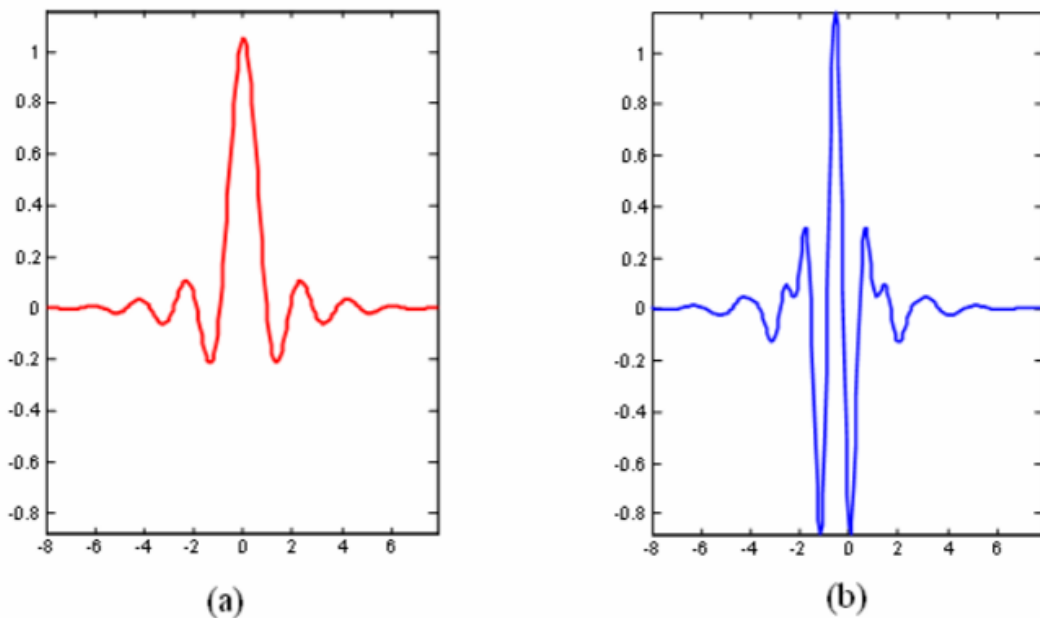
onde

$$v(a) = a^4(35 - 84a + 70a^2 - 20a^3), \quad a \in [0,1] \quad (2.9)$$

Deste modo, a função *wavelet*  $\psi(t)$  pode ser encontrada facilmente através de  $\phi(t)$ .

$$\hat{\psi}(t) = \begin{cases} (2\pi)^{-1/2} e^{t/2} \sin\left[\frac{\pi}{2} v\left(\frac{3}{2\pi}|t|-1\right)\right], & \text{se } \frac{2\pi}{3} \leq |t| \leq \frac{4\pi}{3} \\ (2\pi)^{-1/2} e^{t/2} \cos\left[\frac{\pi}{2} v\left(\frac{3}{4\pi}|t|-1\right)\right], & \text{se } \frac{4\pi}{3} \leq |t| \leq \frac{8\pi}{3} \\ 0, & \text{caso contrário} \end{cases} \quad (2.10)$$

A Figura 2.4 ilustra as equações (2.8) e (2.10) respectivamente.



**Figura - 2.4:** (a) Função escala de Meyer; (b) Função *wavelet* de Meyer

Daubechies propôs um procedimento de partida para a construção das bases ortonormais ao invés de construir a *wavelet* e a função de escala através de um subespaço  $V_j$ .

O procedimento parte de coeficientes apropriados e então investiga se eles correspondem a uma base de *wavelet* ortonormal. Esses coeficientes representam um conjunto particular de números gerados por filtros. Em 1987 as bases ortonormais de *wavelets* foram consideradas como sendo funções de suporte compacto contidas no intervalo  $[0, 2r+1]$ .

Quanto maior o número de coeficientes, mais suave será a *wavelet*. A construção de Daubechies resulta em uma coleção de coeficientes  $N^{h_n}$ , sendo:

$$N = 2, 3, 4, \dots \text{ e } 0 \leq n \leq 2N - 1$$

A seguir é apresentado um exemplo da *wavelet* de Daubechies, a DAUB4, gerada a partir de apenas quatro coeficientes.

$$(h_0, h_1, h_2, h_3) = \left( \frac{1 + \sqrt{3}}{4\sqrt{2}}, \frac{3 + \sqrt{3}}{4\sqrt{2}}, \frac{3 - \sqrt{3}}{4\sqrt{2}}, \frac{1 - \sqrt{3}}{4\sqrt{2}} \right) \quad (2.11)$$

A partir desses coeficientes pode-se construir a função escala:

$$\phi(t) = \sqrt{2} \sum_{k=0}^{2N-1} h_k \phi(2t - k) \quad (2.12)$$

e calcular  $g_n$ :

$$(g_0, g_1, g_2, g_3) = \left( \frac{1 - \sqrt{3}}{4\sqrt{2}}, \frac{-3 + \sqrt{3}}{4\sqrt{2}}, \frac{3 + \sqrt{3}}{4\sqrt{2}}, \frac{-1 - \sqrt{3}}{4\sqrt{2}} \right) \quad (2.13)$$

Assim, a *wavelet* de Daubechies é dada por:

$$\psi(t) = \sqrt{2} \sum_{k=0}^{2N-1} g_k \phi(2t - k) \quad (2.14)$$

Portanto, a matriz A representa os coeficientes da *wavelet* de Daubechies (DAUB4) na decomposição de um sinal.

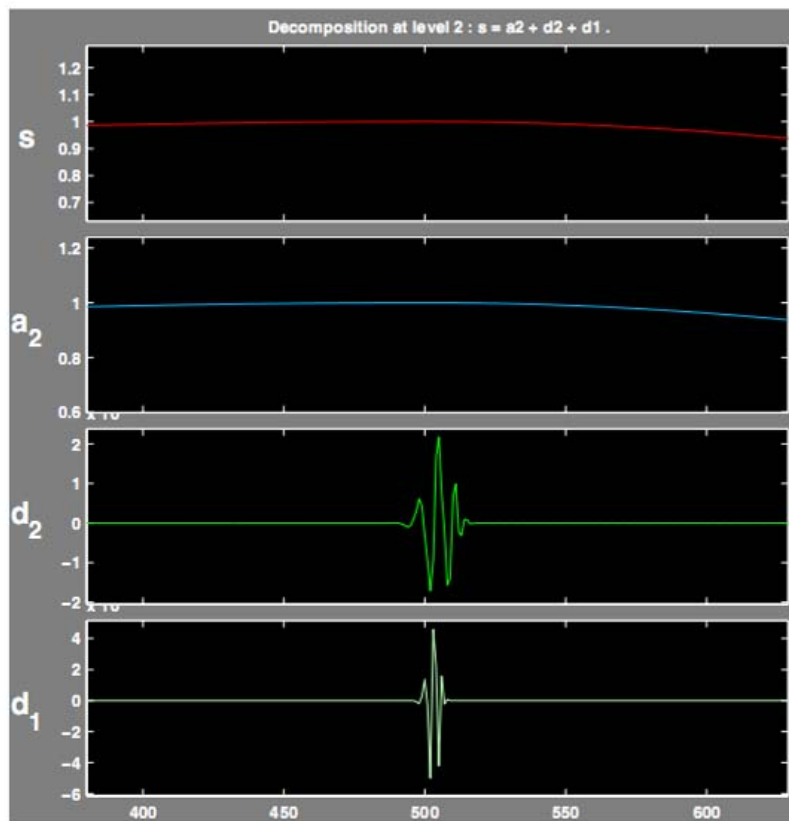




## 2.2 Detectando Descontinuidade

A análise por *wavelets* fornece acesso a informações imediatas que não seriam possíveis se utilizássemos outro método como a análise de *Fourier* (LIRA *et al*, 2007)

A Figura 2.6 mostra um exemplo de como a análise por *wavelets* pode descobrir descontinuidade de um sinal. O sinal, enquanto parece uma curva amortecida e única, é de fato composto de dois sinais exponenciais que são unidos no tempo = 500. A descontinuidade ocorre só no tempo = 500. Foi ampliado na figura o trecho da metade do sinal para mostrar mais claramente o que acontece em volta do tempo = 500. Os detalhes são altos só no meio do sinal e insignificantes em outro lugar. Isto sugere a presença da informação de alta frequência — uma modificação súbita ou descontinuidade — em volta do tempo = 500. A db4 é suficientemente regular para esta análise. Utilizando a *wavelet* de haar, a descontinuidade não teria sido descoberta (MISITI *et al*, 2002).



**Figura - 2.6:** Sinal com descontinuidade  
Fonte: (MISITI *et al*, 2002)

De acordo com SANTOSO (2000) e BORRÁS (2001), muitos métodos modernos de classificação utilizam a Transformada Wavelet usando seu primeiro nível de decomposição para esta finalidade.

A Transformada Wavelet tem sido utilizada com sucesso em diferentes aplicações nas áreas de processamento de sinais e tem sido proposta como ferramenta para análise de transitórios de sistemas elétricos (RIBEIRO, 1994)

A *Wavelet* de Daubechies de nível 4 (DAUB4) será utilizada neste trabalho para identificar os momentos de descontinuidades dos sinais medidos no chaveamento do GMG com o UPS. As decomposições *wavelets* serão utilizadas na arquitetura da rede neural que será desenvolvida com o objetivo de identificar a correlação das potências dos equipamentos estudados com os distúrbios provocados à rede.

A técnica de redes neurais será apresentada no capítulo seguinte.

# CAPÍTULO 3

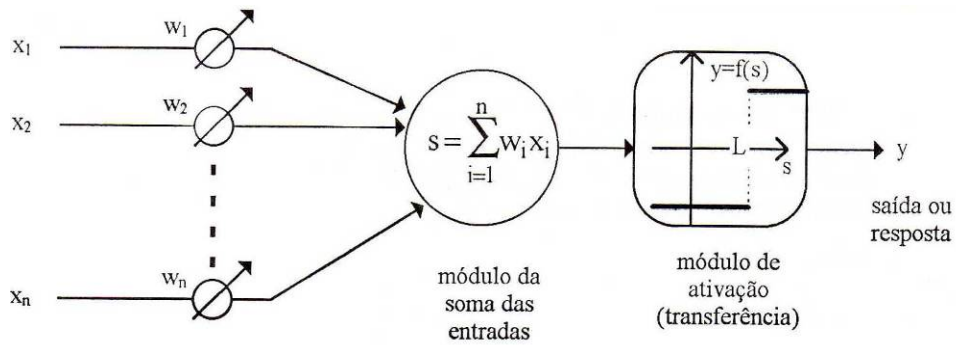
## 3 REDES NEURAIIS ARTIFICIAIS

### 3.1 Introdução

A rede neural artificial é uma técnica de Inteligência Artificial que tenta simular em máquinas (computadores) o funcionamento do cérebro humano, de uma maneira simplificada. Ela é capaz de reconhecer padrões, extrair regularidades e detectar relações subjacentes em um conjunto de dados aparentemente desconexos. Além disso, ela apresenta habilidade de lidar com dados ruidosos, incompletos ou imprecisos, e de prever sistemas não lineares. Ela pode ser definida como a área das ciências da computação que visa o projeto de sistemas inteligentes, ou seja, sistemas que tentam emular algum tipo de inteligência, semelhante à de um ser humano, em termos de processos computacionais. Os sistemas inteligentes são assim denominados por exibirem características que associamos ao comportamento inteligente de um ser humano, como por exemplo: percepção, aprendizagem, raciocínio, comunicação e atuação em ambientes complexos (NILSSON, 1998).

Uma rede neural, segundo HAYKIN (2001), pode ser definida como, um processador maciçamente paralelamente distribuído, constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para uso. Ela assemelha-se ao cérebro em dois aspectos: (1) o conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem; (2) forças de conexão entre neurônios (os pesos sinápticos) são utilizadas para armazenar o conhecimento adquirido.

As redes neurais são formadas por neurônios e conexões entre eles. O neurônio (Figura 3.1) representa uma região onde informações são processadas. Seus três elementos básicos são: os pesos sinápticos, a função de soma e a função de transferência.



**Figura - 3.1:** Modelo do neurônio artificial  
 Fonte: (LOESCH; SARI, 1996)

As conexões entre os neurônios, denominadas pesos sinápticos, são responsáveis pelo armazenamento das informações. Além disso, elas definem o efeito que a saída de um neurônio exerce sobre a entrada do neurônio seguinte. Os pesos sinápticos são de grande importância para uma rede neural, pois determinam toda a manipulação de valores da rede.

A função de soma processa os estímulos ponderados pelos respectivos pesos, ou seja:

$$x_j = \sum_i w_{ij} y_i \quad (3.1)$$

onde  $y_i$  é a saída gerada por cada neurônio da camada anterior.

Já a função de transferência, também chamada de função de ativação, limita a amplitude do intervalo do sinal de saída do neurônio para algum valor finito, geralmente no intervalo normalizado  $[0, 1]$  ou  $[-1, 1]$ .

$$y_j = f(x_j) \quad (3.2)$$

Aqui nós identificamos três tipos básicos de funções de ativação (HAYKIN, 2001):

a) Função de Limiar. Para este tipo de função de ativação, descrito na figura 3.2 (a), temos:

$$f(x) = \begin{cases} 1 & \text{se } x \geq 0 \\ 0 & \text{se } x < 0 \end{cases} \quad (3.3)$$

b) Função Linear por Partes. Para a função linear por partes descrita na Figura 3.2 (b) temos:

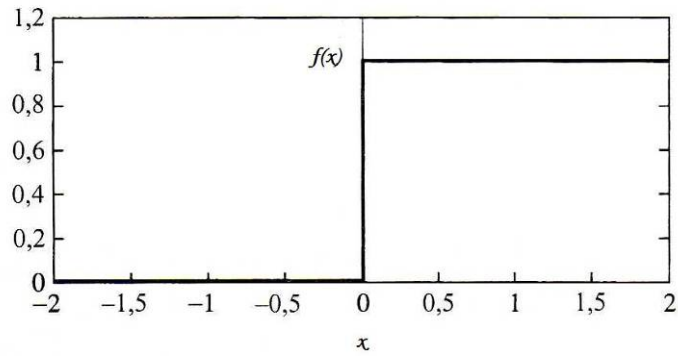
$$f(x) = \begin{cases} 1, & x \geq \frac{1}{2} \\ x, & -\frac{1}{2} < x < \frac{1}{2} \\ 0, & x \leq -\frac{1}{2} \end{cases} \quad (3.4)$$

onde  $(-\frac{1}{2}$  e  $\frac{1}{2})$  é o intervalo que define a saída linear e 0 e 1 são os limites mínimo e máximo da função.

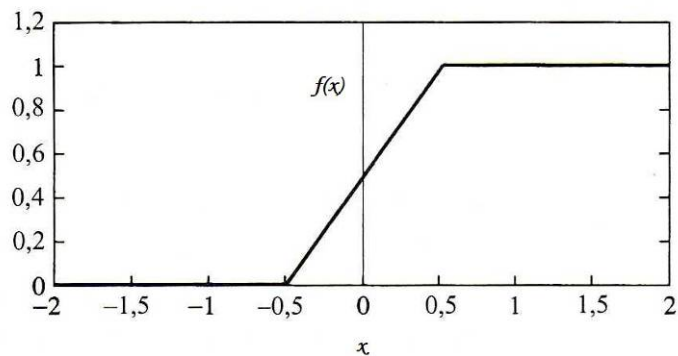
c) Função Sigmóide. A função sigmóide, cujo gráfico tem a forma de s, é de longe a forma mais comum de ativação utilizada na construção de redes neurais artificiais. Ela é definida como uma função estritamente crescente que exhibe um balanceamento adequado entre comportamento linear e não-linear. Um exemplo de função sigmóide é a função logística, definida por:

$$f(x) = \frac{1}{1 + \exp(-\alpha x)} \quad (3.5)$$

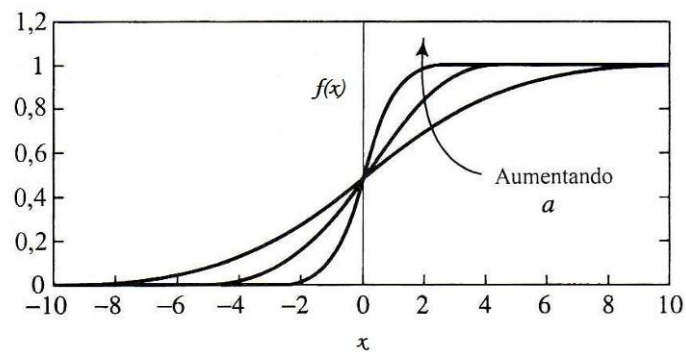
Onde  $\alpha$  é o parâmetro de inclinação da função sigmóide. Variando-se o parâmetro  $\alpha$ , obtemos funções sigmóides com diferentes inclinações, como ilustrado na Figura 3.2 (c).



(a)



(b)



(c)

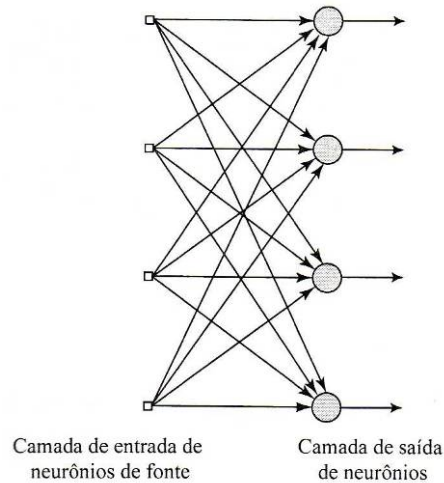
**Figura - 3.2:** (a) Função de limiar. (b) Função linear por partes. (c) Função sigmóide para parâmetro de inclinação  $\alpha$  variável.

Fonte: (HAYKIN, 2001) adaptado

Além dos três elementos básicos já citados, o neurônio pode ainda apresentar um bias que tem o efeito de aumentar ou diminuir a entrada líquida da função de ativação (HAYKIN, 2001). O termo bias age como um peso extra nas conexões das unidades cuja entrada é sempre um (FAUSETT, 1994).

Em geral, podemos identificar três classes de arquiteturas de rede fundamentalmente diferentes (HAYKIN, 2001):

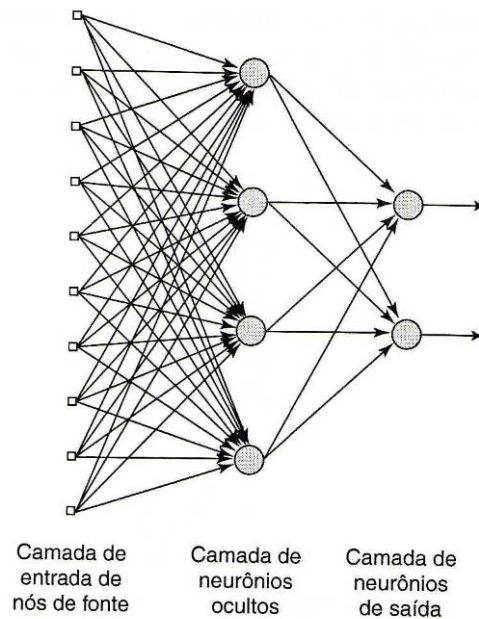
a) Redes Alimentadas Adiante com Camada Única: Em uma rede neural em camadas, os neurônios estão organizados na forma de camadas. Na forma mais simples de uma rede em camadas, temos uma camada de entrada de nós de fonte que se projeta sobre uma camada de saída de neurônios, mas não vice-versa. Ela é ilustrada na figura 3.3



**Figura - 3.3:** Rede alimentada adiante ou acíclica com uma única camada de neurônios  
Fonte: (HAYKIN, 2001)

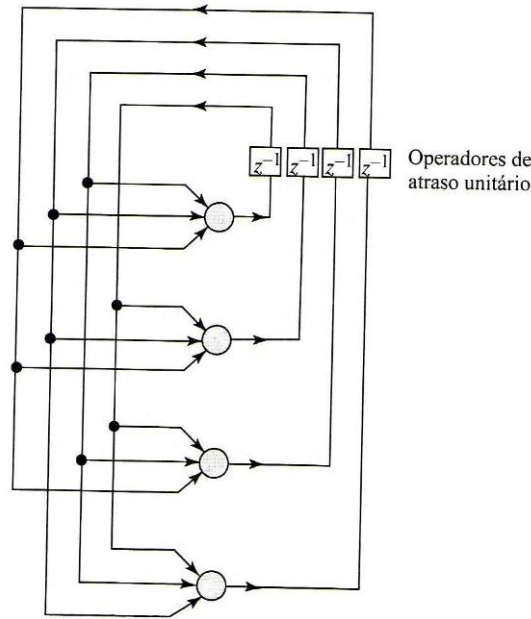
b) Redes Alimentadas Diretamente com Múltiplas Camadas: A segunda classe de uma rede neural alimentada adiante se distingue pela presença de uma ou mais camadas ocultas, cujos nós computacionais são chamados correspondentes de neurônios ocultos ou unidades ocultas. A função dos neurônios ocultos é intervir entre a entrada externa e a saída da rede de maneira útil. Adicionando-se uma ou mais camadas ocultas, tornamos a rede capaz de extrair estatísticas de ordem elevada. A figura 3.4 ilustra uma rede neural de múltiplas camadas alimentada adiante para o caso de uma única camada oculta.



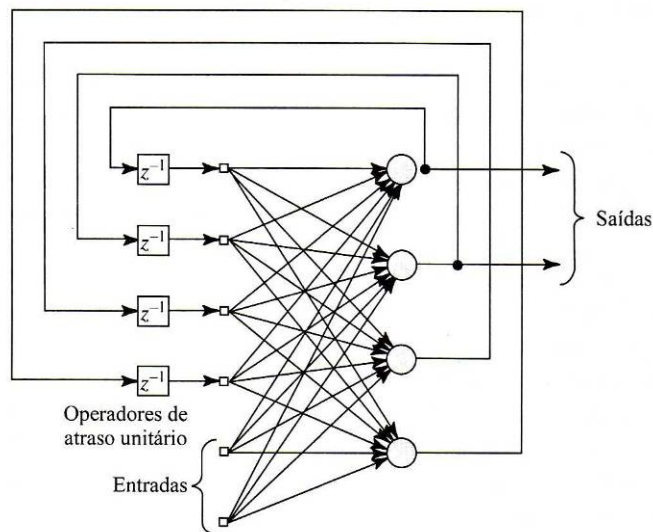


**Figura - 3.4:** Rede alimentada adiante ou acíclica totalmente conectada com uma camada oculta e uma camada de saída  
 Fonte: (HAYKIN, 2001)

c) Redes recorrentes: Uma rede neural recorrente se distingue de uma rede neural alimentada adiante por ter pelo menos um laço de realimentação. Uma rede recorrente pode consistir, por exemplo, de uma única camada de neurônios com cada neurônio alimentando seu sinal de saída de volta para as entradas de todos os outros neurônios como ilustrado nas figuras 3.5 e 3.6. Os laços de realimentação envolvem o uso de ramos particulares compostos de elementos de atraso unitário (representados por  $z^{-1}$ ), o que resulta em um comportamento dinâmico não-linear, admitindo-se que a rede neural contenha unidades não-lineares.



**Figura - 3.5:** Rede recorrente sem laços de auto-realimentação e sem neurônios ocultos  
 Fonte: (HAYKIN, 2001)



**Figura - 3.6:** Rede recorrente com neurônios ocultos  
 Fonte: (HAYKIN, 2001)

A rede neural deve ter a capacidade de generalização, ou seja, ela deve ser capaz não apenas de classificar as entradas para as quais ela recebe treinamento, mas também de generalizar e classificar entradas que não tenham sido apresentadas. Isto é possível graças a um processo de aprendizagem ao qual a rede é submetida. Esta propriedade permite que a rede encontre respostas corretas mesmo quando os dados disponíveis para as entradas estão incompletos ou danificados.

O desenvolvimento de uma rede neural ainda é um processo de tentativa e erro. A seleção da rede envolve a escolha da topologia da rede (ou arquitetura), da função de transferência e do algoritmo de aprendizagem.

### **3.2 Aprendizagem**

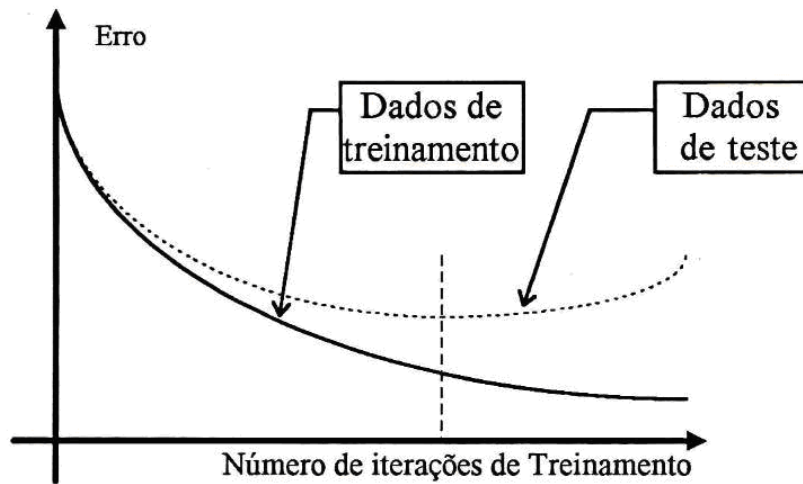
Segundo LOESCH (1996), a Psicologia possui uma dicotomia: de um lado as aproximações cognitivas e de outro a tecnologia behaviorista (estudo do comportamento). O campo de redes neurais recebe contribuições de ambas as visões. O sistema computacional aprende a generalizar e com o mesmo progresso dos animais através do treinamento behaviorista.

A propriedade mais importante das redes neurais é a habilidade de aprender de seu ambiente e com isso melhorar seu desempenho. Isto é feito através de um processo iterativo de ajustes aplicado a seus pesos, o treinamento. O aprendizado ocorre quando a rede neural atinge uma solução generalizada para uma classe de problemas.

O processo de aprendizagem nas redes neurais acontece internamente por meio do ajuste dos pesos sinápticos das conexões durante a exposição dos exemplos, em resposta à quantidade de erros gerados pela rede.

As redes neurais são treinadas para aprender a partir dos dados de entrada. Assim como o cérebro humano, elas aprendem a partir de experiências e não através de programação. Por este motivo, deve-se tomar bastante cuidado com a formação do conjunto de treinamento. Este conjunto deve ser gerado a partir de dados históricos, ou seja, a partir de experiências e fatos ocorridos no passado.

Existe uma concepção errônea perigosa relativa ao treinamento iterativo. Ela se apóia no supertreinamento da rede, baseado na idéia de que se deve tentar treinar a rede para reduzir mais e mais o erro do conjunto de treinamento para aumentar seu desempenho. Embora parcialmente verdadeira, ela pode conduzir a um superajustamento ao conjunto de treinamento, perdendo, além de certo ponto, desempenho com relação ao conjunto de testes. A Figura 3.7 plota o erro de uma rede neural para o conjunto de treinamento e para o conjunto de teste. Após passar de N iterações o erro do conjunto de testes cresce. Deve-se parar o treinamento quando o ponto N foi atingido, para obter o máximo desempenho operacional da rede (LOESCH; SARI, 1996).



**Figura - 3.7:** Manifestação do supertreinamento  
 Fonte: (LOESCH; SARI, 1996)

Dois paradigmas de aprendizagem são apresentados a seguir:

a) Aprendizagem supervisionada (ou aprendizagem com professor), quando é utilizado um agente externo que indica à rede a resposta desejada para o padrão de entrada. O ajuste dos pesos ocorre quando o sistema compara a saída da rede com a resposta desejada previamente conhecida.

b) Aprendizagem não-supervisionada (ou aprendizagem sem professor), quando não existe um agente externo indicando a resposta desejada para os padrões de entrada. A rede neural utiliza os neurônios como classificadores, e os dados de entrada como elementos de classificação. Esse tipo de rede trabalha essas entradas e se organiza de modo a classificá-las mediante algum critério de semelhança.

Em um processo de aprendizagem, os pesos dos neurônios são ajustados através de um algoritmo de aprendizagem. O algoritmo de aprendizagem é um conjunto pré-estabelecido de regras bem-definidas para resolução de um problema de aprendizagem (HAYKIN, 2001). Ele tem como objetivo encontrar pesos para a rede que permitam que esta gere saídas compatíveis com as desejadas. Os algoritmos existentes diferem entre si pela forma como ocorre o ajuste dos pesos sinápticos dos neurônios, ou seja, pela regra de aprendizagem adotada.

De acordo com HAYKIN (2001), existem cinco regras básicas de aprendizagem através das quais os pesos sinápticos de uma rede podem ser ajustados: aprendizagem por

correção de erro, baseada em memória, hebbiana, competitiva e aprendizagem de Boltzmann.

### 3.2.1 Aprendizagem Por Correção De Erro

A aprendizagem por correção de erro é baseada no paradigma de aprendizagem supervisionada no qual a saída desejada para cada padrão de entrada é fornecida para a rede. O sinal de saída gerado pela rede, representado por  $y_k(n)$ , é comparado com a resposta desejada, representada por  $d_k(n)$ , produzindo um sinal de erro  $e_k(n)$ . Este erro é dado por:

$$e_k(n) = d_k(n) - y_k(n) \quad (3.6)$$

O sinal de erro é então utilizado para ajustar os pesos das conexões com o objetivo de aproximar o sinal de saída  $y_k(n)$  da resposta desejada  $d_k(n)$ , reduzindo o erro  $e_k(n)$ .

### 3.2.2 Aprendizagem baseada em memória

Nesta regra de aprendizagem, todas as experiências passadas são armazenadas em uma grande memória de exemplos de entrada-saída classificados corretamente:  $(x_i, d_i)$ , onde  $i = 1 \dots n$ ,  $x_i$  representa um padrão de entrada e  $d_i$  representa a resposta desejada correspondente (HAYKIN, 2001). Quando se deseja classificar um vetor de teste  $X$  (não visto antes), o algoritmo responde buscando e analisando os dados de treinamento em uma "vizinhança local" de  $X$ .

Esta regra envolve dois ingredientes essenciais, e a forma como eles são definidos é que vai diferenciar os algoritmos de aprendizagem baseada em memória entre si. Os ingredientes são: o critério utilizado para definir a vizinhança local do vetor de teste  $X$  e a regra de aprendizagem aplicada aos exemplos de treinamento em uma "vizinhança local" de  $X$  (HAYKIN, 2001).

### 3.2.3 Aprendizagem Hebbiana

O postulado de aprendizado de Hebb é a mais antiga regra de aprendizagem existente [HEBB, 1949 *apud* HAYKIN, 2001]. O princípio básico desta regra é: se dois neurônios em ambos os lados de uma conexão são ativados simultaneamente, então a força desta conexão é seletivamente aumentada. A forma mais simples de aprendizagem hebbiana é descrita por HAYKIN (2001):

$$w_{jk}(n+1) = w_{jk}(n) + \eta y_k(n) x_j(n) \quad (3.7)$$

onde  $w_{jk}$  é o peso sináptico do neurônio  $k$ ,  $x_j$  e  $y_k$  são os sinais pré-sináptico e pós-sináptico, respectivamente, deste peso, e  $\eta$  é uma constante positiva que define a taxa de aprendizagem.

### 3.2.4 Aprendizagem competitiva

Na regra de aprendizagem competitiva, os neurônios da camada de saída competem entre si para se tornarem ativos, considerando que somente um neurônio pode estar ativo em um determinado instante. Este fenômeno é conhecido como '*winner-take-all*', isto é, 'o vencedor leva tudo'.

Para um neurônio ser o vencedor, isto é, estar ativo, seu campo local induzido  $v_k$  para um padrão de entrada  $x$  deve ser o maior dentre todos os neurônios da rede (HAYKIN, 2001). Quando isto acontece, o sinal de saída  $y_k$  deste neurônio é igual a um. Caso contrário o sinal de saída é colocado em zero, como demonstrado a seguir:

$$y_k = \begin{cases} 1 & \text{se } v_k > v_j \text{ para todos } j, j \neq k \\ 0 & \text{caso contrário} \end{cases} \quad (3.8)$$

Nesta regra, a variação  $\Delta w_{jk}$  que é aplicada ao peso  $w_{jk}$  é definida por:

$$\Delta w_{jk} = \begin{cases} \eta(x_j - w_{jk}) & \text{se o neurônio } k \text{ vencer a competição} \\ 0 & \text{se o neurônio } k \text{ perder a competição} \end{cases} \quad (3.9)$$

onde  $\eta$  é a taxa de aprendizagem. Como efeito desta regra de aprendizagem, o vetor de peso  $w_k$  é movido na direção do padrão de entrada  $x$ , a partir do neurônio vencedor  $k$ .

Este tipo de aprendizagem é adequado para descobrir características nos dados de entrada que podem ser utilizadas para agrupar padrões similares.

### 3.2.5 Aprendizagem de Boltzmann

A regra de aprendizagem de Boltzmann é um algoritmo de aprendizagem estocástico que realiza o ajuste dos pesos baseando-se na probabilidade e na mecânica estatística. A rede neural que utiliza esta regra é denominada máquina de Boltzmann (HAYKIN, 2001).

Os neurônios nesta máquina formam uma estrutura recorrente e podem assumir dois estados: ligado (+1) ou desligado (-1). Os estados de cada neurônio na máquina determinam o valor de uma função de energia que caracteriza esta rede:

$$E = -\frac{1}{2} \sum_j \sum_k w_{jk} x_j x_k \quad j \neq k \quad (3.10)$$

onde  $x_j$  é o estado do neurônio  $j$  e  $w_{jk}$  é o peso sináptico entre os neurônios  $j$  e  $k$ . A restrição  $j \neq k$  é para garantir que nenhum dos neurônios tenha auto-realimentação. O equilíbrio é alcançado quando esta função de energia alcança um mínimo.

A máquina escolhe um neurônio ao acaso e, em um determinado passo do processo de aprendizagem, troca seu estado de  $+x_k$  para  $-x_k$ , a uma pseudotemperatura  $T$ , com probabilidade:

$$P(x_k \rightarrow -x_k) = \frac{1}{1 + \exp(-\Delta E_k/T)} \quad (3.11)$$

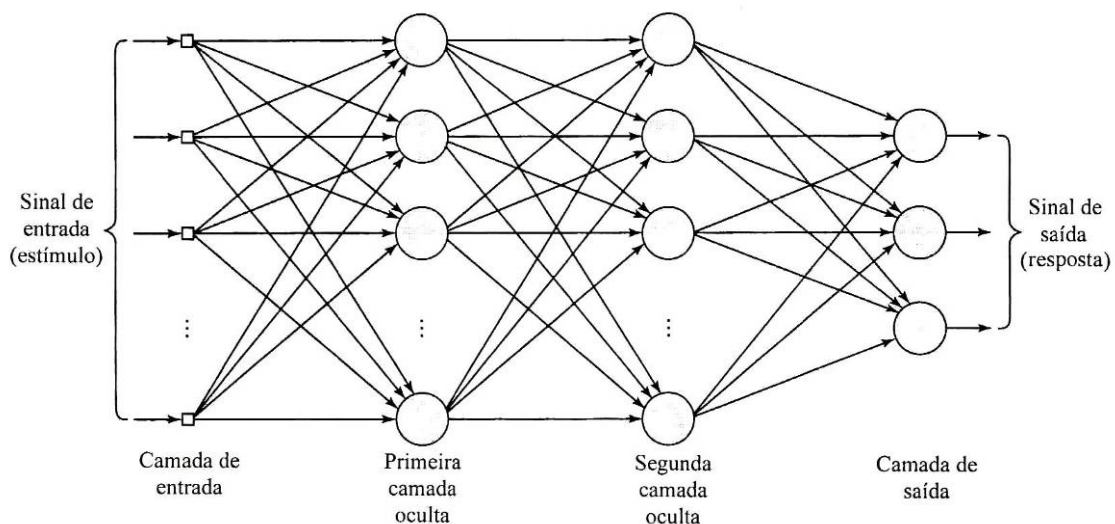
onde  $\Delta E_k$  é a variação de energia resultante da troca.

Os neurônios desta rede podem ser visíveis ou ocultos e o sistema é operado em dois modos: condição presa, no qual os neurônios visíveis estão presos em estados específicos determinados pelo meio; e condição de operação livre, onde os neurônios visíveis e ocultos podem operar livremente (HAYKIN, 2001).

### 3.3 Rede Neural *Backpropagation*

#### 3.3.1 Introdução

A rede neural comumente denominada *backpropagation* é na verdade uma Rede Alimentada Diretamente com Múltipla Camada (Figura 3.8), treinada pelo algoritmo *backpropagation* (algoritmo de retropropagação do erro). O desenvolvimento do algoritmo de retropropagação representa um marco nas redes neurais (HAYKIN, 2001). Este trabalho foi o primeiro que possibilitou o ajuste dos pesos em redes multicamadas *feedforward*, abrindo caminho para a elaboração de redes neurais mais genéricas. Devido à grande popularidade do método *backpropagation*, seu nome é utilizado para denominar as redes que o utilizam no seu treinamento.



**Figura - 3.8:** Grafo arquitetural de um perceptron de múltiplas camadas com duas camadas ocultas  
Fonte: (HAYKIN, 2001)

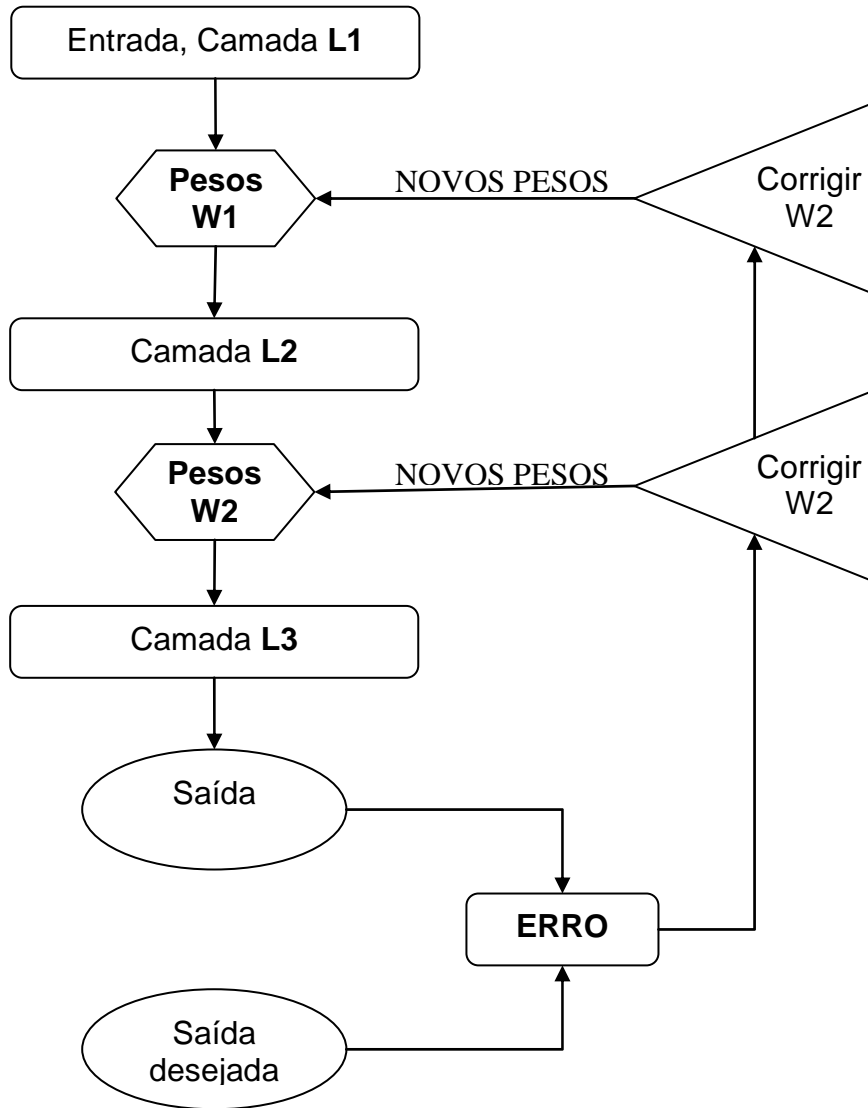
A atratividade do método *backpropagation* vem do conjunto de equações bem definidas e explícitas para correção dos pesos da rede. Este algoritmo consiste em realizar a retropropagação do erro gerado na comparação entre a saída da rede e a saída desejada com o objetivo de minimizar o erro total da saída gerada pela rede. O treinamento de uma



rede através deste algoritmo envolve três etapas: a propagação dos dados da camada de entrada para a camada de saída da rede, o cálculo e a retropropagação do erro gerado pela rede, e o ajuste dos pesos (FAUSETT, 1994).

Na primeira etapa, estímulos de entrada são apresentados à rede e as ativações fluem até chegarem à camada de saída, gerando um resultado. Já na segunda e terceira etapas, o resultado obtido pela rede é comparado com a saída desejada e o erro gerado é computado para as unidades de saída. Os pesos conectados às unidades de saída são então ajustados para reduzir este erro. Em seguida, o erro da camada de saída é utilizado para derivar estimativas de erro para as unidades da(s) camada(s) oculta(s), para que o erro seja então propagado para trás até a conexão da camada de entrada. O fluxo de informação deste processo é ilustrado de uma forma resumida na Figura 3.9.

O método *backpropagation* atualiza os pesos incrementalmente, depois de analisar cada par entrada-saída. Depois da apresentação de todos os pares entrada-saída diz-se que uma época foi concluída. Este treinamento, em geral, requer muitas épocas.



**Figura - 3.9:** Correção dos pesos por *backpropagation*

### 3.3.2 Ajuste dos Pesos

Uma regra de aprendizagem por correção de erro, a regra delta, é onde está baseado o ajuste dos pesos realizado pelo algoritmo *backpropagation*. Este ajuste é feito baseado na retropropagação do erro através da qual o erro gerado pelos neurônios na camada de saída é distribuído para os demais neurônios da rede.

Mesmo conhecendo o erro global da rede, não é possível determinar os pesos exatos para poder corrigi-lo. Entretanto, com base nesta informação, pode-se estabelecer a direção na qual os pesos devem ser ajustados para minimizar o erro quadrado total da saída

da rede. Conhecida esta direção, é possível ajustar os pesos até que o menor erro global seja atingido. O ajuste de um peso  $w_{ij}$  que define seu valor para a próxima iteração é definido por:

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n) \quad (3.12)$$

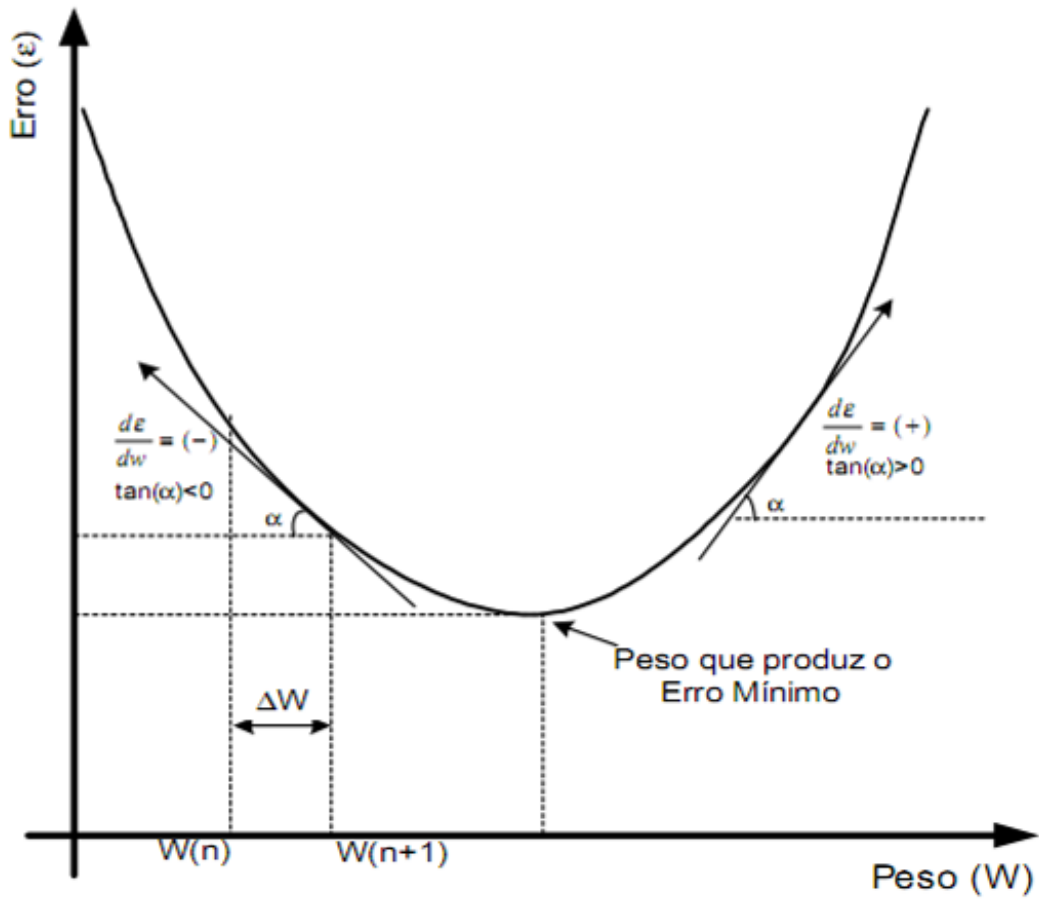
A variação  $\Delta w_{ij}$  que é aplicada ao peso  $w_{ij}$  deve ser proporcional ao sinal de entrada  $x_j$ , que é definido pelos sinais de saída da camada anterior ponderados pelos pesos, e ao erro gerado na saída. Ela é dada por:

$$\Delta w_{ij}(n) = \eta \delta_j(n) x_j(n) \quad (3.13)$$

onde  $\eta$  é a taxa de aprendizagem e  $\delta_j$  é o gradiente local do erro para o neurônio  $j$ .

A taxa de aprendizagem é um valor positivo, geralmente menor do que 1, que regula a intensidade com que as atualizações dos parâmetros (pesos) serão efetuadas. Taxas muito baixas, próximas de zero, tendem a fazer com que o aprendizado seja bastante lento, porém taxas muito altas, próximas de 1, podem fazer com que a rede oscile, como se estivesse aprendendo e desaprendendo, e às vezes nem consiga chegar a um patamar aceitável de aprendizado. O valor da taxa de aprendizado não precisa permanecer fixo durante todo o treinamento. Em algumas implementações ela pode ser adaptativa e controlada pela própria rede.

O gradiente local do erro é determinado através do método gradiente descendente. Ele é o termo responsável pela distribuição do erro da camada de saída para as camadas anteriores. O ajuste dos pesos ( $\Delta w$ ) deve ser realizado na direção contrária ao gradiente, conforme mostra a Figura 3.10. Se o peso  $w^{(n)}$  (valor do peso na iteração  $n$ ) está à esquerda do erro mínimo, o ajuste  $\Delta w$  deve ser positivo para que  $w^{(n+1)}$  (valor do peso da próxima iteração) esteja mais próximo do valor de  $w$  que minimiza o erro. Por outro lado, se o peso  $w^{(n)}$  está à direita do erro mínimo, o ajuste  $\Delta w$  deve ser negativo.



**Figura - 3.10:** Erro em função do peso para uma única conexão

O ajuste dos pesos para os neurônios da camada de saída em um algoritmo *backpropagation* é diferenciado do ajuste dos pesos para os neurônios da camada oculta. A seguir será demonstrado o processo de ajuste dos pesos para cada caso separadamente.

- Ajuste dos pesos na camada de saída

Devido ao fato da aprendizagem ser do tipo supervisionada, o resultado desejado para a camada de saída  $l$  é fornecido para a rede. Com isto, pode-se fazer uma comparação deste com o resultado obtido pela rede nesta camada, gerando um sinal de erro que é utilizado para realizar o ajuste dos pesos dos neurônios desta camada:

$$\epsilon_{l,j}(n) = d_{l,j}(n) - y_{l,j}(n) \quad (3.14)$$

Tendo o valor do erro, seu gradiente local é definido como:

$$\delta_{l,j}(n) = \varepsilon_{l,j}(n) \frac{\partial y_{l,j}(n)}{\partial x_{l,j}(n)} \quad (3.15)$$

A expressão para o ajuste dos pesos entre a camada de saída  $l$  e a camada  $l-1$  é então determinada como:

$$\Delta w_{l-1,i,j}(n) = \eta \delta_{l,j}(n) y_{l-1,i}(n) \quad (3.16)$$

- Ajuste dos pesos na camada oculta

Da mesma forma que os neurônios na camada de saída devem apresentar saídas próximas aos alvos ( $d$ ), os neurônios na(s) camada(s) oculta(s) também devem exibir saídas determinadas, contudo desconhecidas. A saída desejada para este tipo de camada não é informada para a rede. Neste ponto o algoritmo *backpropagation* justifica seu nome, retropropagando o erro gerado pelos neurônios na camada de saída para as camadas internas, distribuindo o erro para cada um dos neurônios nas camadas ocultas.

O gradiente local do erro para a(s) camada(s) oculta(s) é então definido como:

$$\delta_{l,j}(n) = \sum_k [\delta_{l+1,k}(n) w_{l,j,k}(n)] \frac{\partial y_{l,j}(n)}{\partial x_{l,j}(n)} \quad (3.17)$$

E o ajuste dos pesos entre os neurônios da camada  $l$  e da camada  $l-1$  é determinado pela Equação (3.16) a qual também determina este ajuste para a camada de saída.

### 3.3.3 Observações

a) A escolha da função de transferência em uma rede neural *backpropagation* deve obedecer aos requisitos de continuidade, diferenciabilidade e monotonicidade. Estes requisitos são exigidos pelo algoritmo *backpropagation* para permitir que uma expressão analítica, para o ajuste dos pesos da rede, seja obtida.

b) Um termo extra que pode ser adicionado ao ajuste dos pesos na tentativa de melhorar performance da rede é a taxa de momentum. Esta taxa é um parâmetro de uso opcional, de valor positivo menor do que 1, cuja utilização visa imprimir uma dinâmica no treinamento tal que, eventualmente, possibilite que o algoritmo livre-se de mínimos locais durante o processo de busca pelo mínimo global. Para utilizá-la, os pesos de um ou mais padrões de treinamento anteriores devem ser salvos. Sua utilização é recomendada quando alguns dados de entrada são muito diferentes da maioria (FAUSETT, 1994). Normalmente resulta em uma aprendizagem mais rápida, mas pode causar instabilidade em alguns casos se for muito grande. Quando a taxa de momentum  $\mu$  é utilizada, a expressão para o ajuste dos pesos entre a camada de saída  $l$  e da camada  $l-1$  passa a ser definida como:

$$\Delta w_{l-1,i,j}(n) = \eta \delta_{l,j}(n) y_{l-1,i}(n) + \mu [w_{l-1,i,j}(n) - w_{l-1,i,j}(n-1)] \quad (3.18)$$

### 3.4 Projeto de uma Rede Neural

Diante de um projeto de uma rede neural, não pensamos mais em procedimentos, regras ou fórmulas algorítmicas de processamento de dados, mas sim em tipos de dados de entrada, dados de saída e tratamento de dados. "O projeto de uma rede neural é baseado diretamente nos dados do mundo real, permitindo-se que o conjunto de dados fale por si mesmo" (HAYKIN, 2001).

O projeto de um sistema neural consiste de diversas etapas que devem ser executadas em seqüência, de forma interativa e até mesmo com diversos ciclos de repetição. A construção do sistema começa pela identificação e coleta dos dados históricos relevantes para o problema. O passo seguinte é a preparação e adequação dos dados ao formato requerido pela rede neural, ou seja, a formatação dos dados. Neste processo cria-se uma escala, estabelecendo um novo intervalo válido dentro do qual todos os dados são colocados. Os intervalos mais utilizados são  $[-1,1]$  ou  $[0,1]$ . Este processo também é conhecido como normalização dos dados.

Após a escolha de uma representação para os dados do problema, deve-se separar os dados em dois conjuntos: o conjunto de treinamento o qual é gerado a partir de dados históricos, ou seja, a partir de experiências e fatos ocorridos no passado; e o conjunto de

teste com o qual o funcionamento da rede é testado. Cada dado do conjunto de teste é apresentado uma única vez ao sistema.

O próximo passo é realizar a escolha do modelo neural a ser adotado e definir a topologia da rede. Em seguida realiza-se o desenvolvimento, treinamento e otimização do modelo, seguidos pela validação do mesmo. Na etapa de validação, faz-se uma comparação do resultado obtido pela rede com o resultado desejado.

Por fim, após a validação do modelo, realiza-se a aplicação do mesmo.

### **3.5 Vantagens**

Uma das vantagens da rede neural artificial que mais se destaca é a sua capacidade de modelar e prever sistemas não-lineares. Este é o grande diferencial das redes neurais quando comparadas com outros métodos como, por exemplo, modelos estatísticos. Além desta, existem inúmeras outras vantagens atribuídas às redes neurais, dentre as quais se destacam:

- capacidade de encontrar soluções eficientes para problemas do mundo real;
- habilidade de lidar com dados ruidosos, incompletos ou imprecisos;
- capacidade de análise e reconhecimento de padrões;
- capacidade de resolver problemas práticos sem a necessidade da definição de regras ou de modelos precisos;
- capacidade de buscar a solução através de um método próprio de treinamento e auto-aprendizado;
- alta capacidade de adaptação e generalização;

A principal limitação das redes neurais é não ter capacidade de explicação. Os dados entram na rede e uma previsão sai, mas o tipo de relacionamento entre as variáveis não é revelado, assim como detalhes de como as redes raciocinam com os dados para chegar às conclusões também não são fornecidos. Por este motivo, as redes neurais são recomendadas para serem aplicadas em áreas de conhecimento cujas teorias ainda não conseguem explicar adequadamente o comportamento dos fenômenos observados ou em áreas de grande complexidade que não necessitam de modelos precisos da realidade física do problema.

### **3.6 Aplicações**

Mesmo com algumas restrições, a área de redes neurais tem demonstrado sua potencialidade em diversas aplicações, superando expectativas e gerando resultados até então não alcançados com qualquer outra técnica, seja computacional ou convencional.

As redes neurais artificiais podem ser treinadas para encontrar soluções, interpretar e classificar dados, reconhecer padrões, aproximar funções e prever eventos futuros. Atualmente, são inúmeras as áreas nas quais as redes neurais têm sido aplicadas, e os bons desempenhos alcançados têm incentivado pesquisadores a fazer das redes neurais uma alternativa de solução para problemas nas mais diversas áreas de atuação.



## CAPÍTULO 4

### 4. APLICAÇÃO DA METODOLOGIA PROPOSTA

#### 4.1 Introdução

Neste capítulo apresenta-se o desenvolvimento de uma rede neural *feedforward* com um algoritmo de retropropagação para interpretar as distorções das formas de ondas de tensão no acoplamento de GMG com UPS.

Na tentativa de auxiliar o desenvolvimento da rede neural, o sinal de saída será formado por decomposição *wavelet*.

#### 4.2 Ferramentas e Equipamentos Utilizados

Para o desenvolvimento da rede neural e decomposição *wavelet*, foi utilizado o programa computacional MATLAB, versão 7.3.0.267 (R2006b).

Todas as simulações foram realizadas em um computador Pentium *Dual-Core* T2080; 1,73GHz; 2,50 GB de memória RAM e sistema operacional Windows Vista. O equipamento teve sua utilização restringida a esse fim enquanto se processava a execução das redes.

As medições das grandezas elétricas foram realizadas pelo Analisador de Energia RE6000/N com módulo H e T, fabricado pela EMBRASUL, capaz de medir harmônicas de tensão e corrente até a 50ª ordem numa janela de 128ms e transientes de tensão e corrente na frequência de 8kHz. As leituras foram armazenadas no equipamento e analisadas pelo software ANL6000 do mesmo fabricante.

#### 4.3 Banco de Dados

Os dados para desenvolver este trabalho são medições coletadas em sete unidades comerciais que possuem o conjunto GMG-UPS como alternativa de fornecimento ininterrupto de energia elétrica. Estas sete unidades possuem níveis de carregamentos, UPS e GMG distintos.

As medições foram coletadas durante a simulação de falta de energia e no momento em que o GMG assumia as cargas alimentando os UPS's. Ao identificar transitórios, o analisador de energia armazenava 2.134 registros de tensão e corrente em um intervalo de 266,63ms que contivesse este transitório identificado. Em apenas uma simulação, vários transitórios já podiam ser identificados. A Figura 4.1 mostra uma das medições realizadas em que foram armazenados 26 intervalos de 2.134 registros (266,63ms) referentes a transitórios identificados.

As sete unidades possuem as seguintes configurações:

Unidade 1: um GMG de 180kVA e dois UPS's de 160kVA em paralelo-redundante (totalizando 320kVA em equipamentos UPS) ;

Unidade 2: um GMG de 180kVA e um UPS de 30kVA;

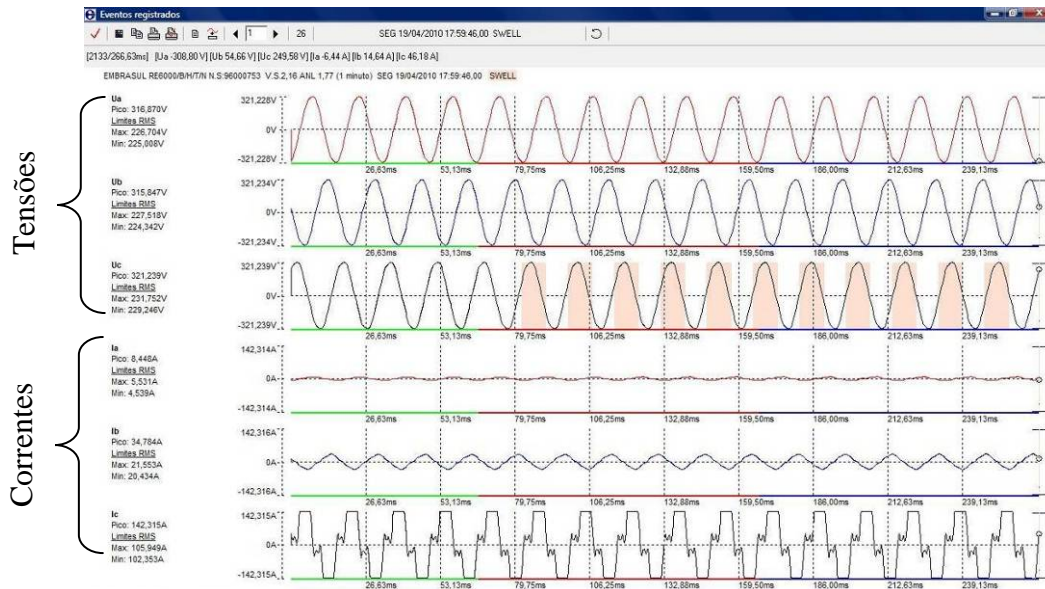
Unidade 3: um GMG de 180kVA e um UPS de 30kVA;

Unidade 4: um GMG de 13,5kVA e dois UPS's de 5kVA (totalizando 10kVA em equipamentos UPS);

Unidade 5: um GMG de 13,5kVA e dois UPS's de 5kVA (totalizando 10kVA em equipamentos UPS);

Unidade 6: um GMG de 88kVA e três UPS's de 5kVA (totalizando 15kVA em equipamentos UPS);

Unidade 7: um GMG de 145kVA e um UPS de 20kVA;



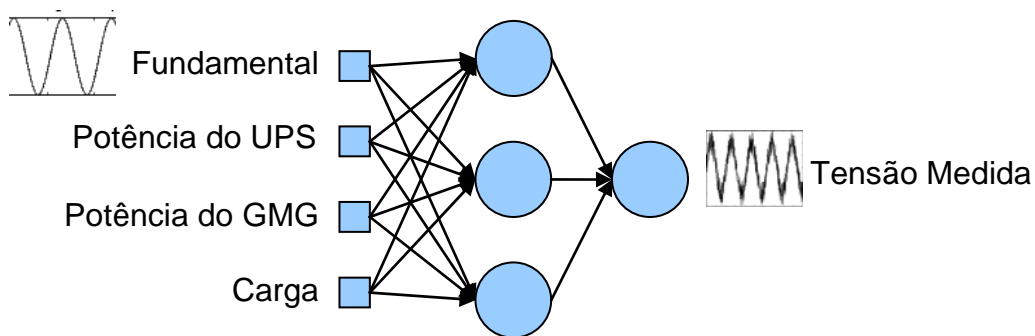
**Figura - 4.1:** Software ANL6000 apresentando um dos 26 registros de transitórios identificados em tela

#### 4.4 Variáveis da Rede Neural

Uma das decisões mais importantes na construção de uma rede neural é a escolha das variáveis nas quais a rede irá se basear para poder aprender. No nosso caso, desejamos entender a relação entre as potências dos equipamentos do sistema GMG-UPS e a deformação da forma de onda neste acoplamento.

É evidente que a potência do GMG, a potência do UPS e o sinal medido no acoplamento entre eles seriam variáveis para a rede neural, mas entendemos que não seriam suficientes para a construção da rede neural, pois as potências dos equipamentos (GMG e UPS) seriam dados de placa enquanto que o sinal seria uma grandeza elétrica variável no tempo e dependeria das cargas alimentadas no sistema. A solução adotada foi acrescentar como variável um sinal unitário de 60Hz com THD = 0%, ou seja, uma forma de onda fundamental e acrescentar a informação do carregamento do sistema.

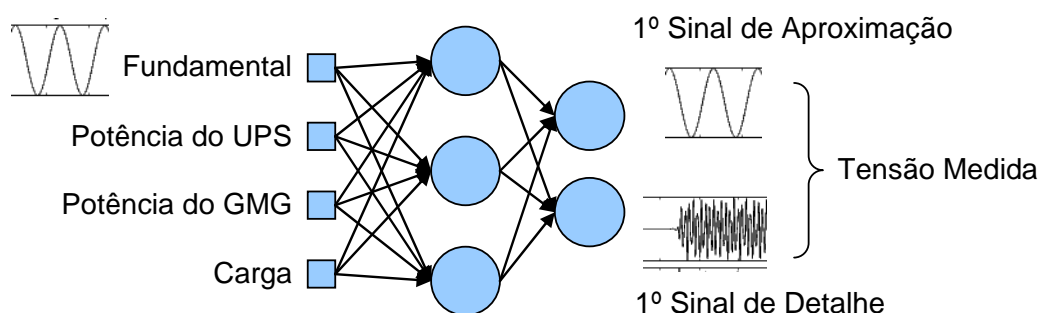
Assim, definimos que a rede neural teria como variáveis de entrada uma forma de onda fundamental, a potência de placa do UPS, a potência de placa do GMG e a potência aparente da carga alimentada. Como sinal de saída seria utilizado o sinal de tensão, no caso da medição ser em equipamentos trifásicos utilizaríamos o sinal de tensão de uma fase. Contextualizando, a rede neural seria um componente que ao informarmos um sinal perfeito, a potência nos equipamentos e o nível de carregamento do sistema, sua resposta seria o sinal distorcido fruto das medições realizadas. A Figura 4.2 mostra a organização das variáveis propostas em uma rede neural para a arquitetura 1.



**Figura - 4.2:** Variáveis propostas para a rede neural (arquitetura 1)

Após esta primeira proposta, houve ainda a hipótese que a rede neural poderia ter dificuldades em convergir, pois a distorção do sinal medido poderia não ser suficiente para excitar a rede. Para este problema, a solução proposta foi de decompor o sinal medido em dois sinais por meio da decomposição *wavelet* utilizando a *Wavelet* de Daubechies Nível 4

(DAUB4) conforme explicado no Capítulo 2. Assim, surge uma segunda proposta de rede neural (arquitetura 2) apresentada na Figura 4.3.



**Figura - 4.3:** Variáveis propostas para a rede neural com *Wavelet* (arquitetura 2)

#### 4.5 Preparação dos Dados

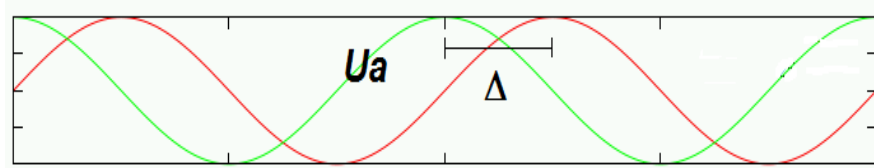
Tendo o entendimento de como seriam organizadas as variáveis para a preparação do banco de treinamento da rede neural, partimos para a adequação dos dados de forma a poder utilizar o Matlab na construção do sistema.

Decidimos criar uma tabela única contendo todos os dados necessários com o seguinte cabeçalho:

**Tabela - 4.1:** Cabeçalho da tabela de dados

Cabeçalho	Fonte de Dados
Tensão da Fase A ( $U_a$ )	Medição do Analisador de Energia
Tensão da Fase B ( $U_b$ )	Medição do Analisador de Energia
Tensão da Fase C ( $U_c$ )	Medição do Analisador de Energia
Corrente da Fase A ( $I_a$ )	Medição do Analisador de Energia
Corrente da Fase B ( $I_b$ )	Medição do Analisador de Energia
Corrente da Fase C ( $I_c$ )	Medição do Analisador de Energia
Potencia Aparente (Carga)	$Carg a =  U_a  \cdot  I_a  +  U_b  \cdot  I_b  +  U_c  \cdot  I_c $
Potência do UPS	Dados de placa do equipamento
Potência do GMG	Dados de placa do equipamento
Sinal Fundamental	Senóide em fase com $U_a$
Aproximação <i>Wavelet</i> Nível 1 (A1)	Decomposição de $U_a$
Detalhe <i>Wavelet</i> Nível 1 (D1)	Decomposição de $U_a$

Para criarmos o sinal fundamental utilizamos a função seno com deslocamento  $\Delta$  para que ficasse em fase com  $U_a$  medido (Figura 4.4).



**Figura - 4.4:** Exemplo da aplicação de  $\Delta$  para ajustar o sinal fundamental

Cada intervalo de leitura do Analisador de Energia contém 16 ciclos de 60Hz com 2.134 amostras, então cada registro do sinal fundamental seria calculado da seguinte forma:

$$Fundamental_i = sen[2 \cdot \pi \cdot 60 \cdot (t_i + \Delta)] \quad (4.1)$$

Onde:

$$i = 1, 2, \dots, 2.134;$$

$$t_1 = 0;$$

$$t_i = ((1/60 \times 16)/2.133) + t_{i-1}, \text{ para } 1 < t \leq 2.134$$

A decomposição *Wavelet* é realizada utilizando as funções *dwt* e *idwt* da toolbox *wavelet* do Matlab da seguinte forma:

- Primeiro consideramos que, para cada intervalo de leitura (2.134 amostras) de  $U_a$ , seria calculada a decomposição *wavelet* utilizando o comando

$$[cA1, cD1] = dwt(s, 'db4') \quad (4.2)$$

onde  $s$  é o sinal formado pelo intervalo de cada leitura de  $U_a$  com 2.134 registros e *'db4'* significa que a decomposição será calculada utilizando a *Wavelet* Daubechies de nível 4 para obtermos os coeficientes da aproximação de nível 1 ( $cA1$ ) e detalhes ( $cD1$ ) (MISITI *et al*, 2002).

- A partir destes coeficientes, construímos os sinais  $A1$  e  $D1$  utilizando os comandos

$$A1 = idwt[cA1, [], 'db4', L] \quad (4.3)$$

$$D1 = idwt[[], cD1, 'db4', L] \quad (4.4)$$

onde  $L$  representa o tamanho do sinal (*length*) que no nosso caso é 2.134.

A Figura 4.5 ilustra um trecho da tabela de dados no Matlab, cujos conteúdos da coluna se encontram na tabela 5.1, e a Figura 4.6 mostra as variáveis utilizadas na formação da rede neural em uma amostra gráfica.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3201	169.708	138.3262	-304.8175	12.1043	1.9428	-13.0051	6.2871	180	30	0.4187	169.3003	0.4077				
3202	156.0112	150.3648	-304.515	15.5152	1.9595	-16.2894	7.6755	180	30	0.3754	156.1661	-0.1549				
3203	142.0567	164.6352	-304.2588	18.5767	1.854	-19.1453	8.7693	180	30	0.3313	141.9011	0.1555				
3204	125.8265	178.9056	-303.7589	21.1253	1.7763	-21.6998	9.5674	180	30	0.2865	126.7959	-0.9694				
3205	110.7128	190.7511	-300.6913	23.3082	1.6875	-23.6937	10.0269	180	30	0.2411	110.1006	0.6122				
3206	93.0657	204.7639	-298.6174	24.8745	1.6897	-25.2656	10.2055	180	30	0.1951	91.4783	1.5874				
3207	73.2718	223.0043	-297.278	25.9277	1.5321	-26.2323	10.0397	180	30	0.1486	74.8964	-1.6246				
3208	59.3817	236.4936	-294.3616	26.3043	1.5043	-26.666	9.7657	180	30	0.1019	60.3739	-0.9921				
3209	46.5006	243.3047	-291.1428	26.2115	1.31	-26.6713	9.3027	180	30	0.0549	45.0407	1.4599				
3210	30.2705	253.176	-286.8222	25.7422	1.1546	-26.1741	8.5789	180	30	0.0078	29.7108	0.5597				
3211	14.4697	264.2704	-281.227	24.6835	1.0436	-25.074	7.6844	180	30	-0.0393	15.3567	-0.887				
3212	1.1378	272.9828	-275.1566	23.1063	1.0991	-23.5033	6.7934	180	30	-0.0864	1.7379	-0.6001				
3213	-10.3907	280.1502	-270.3608	21.0489	1.149	-21.5369	5.9256	180	30	-0.1332	-11.2621	0.8714				
3214	-23.6797	287.2318	-263.9663	18.3857	1.0436	-18.9073	4.8553	180	30	-0.1798	-23.482	-0.1977				
3215	-35.1653	292.897	-258.1335	15.5043	1.1546	-16.1995	3.9746	180	30	-0.2259	-34.8187	-0.3466				
3216	-43.9674	295.9227	-252.7328	11.9461	1.199	-12.5978	3.0135	180	30	-0.2715	-45.2133	1.2459				
3217	-58.4586	299.3133	-242.6386	8.3606	1.2656	-9.0088	2.0773	180	30	-0.3166	-57.4447	-1.0138				
3218	-72.4989	303.9914	-233.4846	3.7546	1.249	-4.2839	1.1077	180	30	-0.3609	-71.7292	-0.7697				
3219	-83.0185	305.9227	-223.3744	-0.3493	1.4377	-0.2644	0.5279	180	30	-0.4045	-84.0562	1.0367				
3220	-94.3538	306.698	-212.4649	-0.6276	1.3711	-0.0688	0.4942	180	30	-0.4471	-94.9105	0.5567				
3221	-108.4371	306.6094	-198.9415	-0.7477	2.015	-0.3967	0.7778	180	30	-0.4887	-107.6881	-0.739				
3222	-122.7136	306.2661	-183.4954	-0.8513	5.2401	-3.5382	2.3586	180	30	-0.5293	-121.9304	-0.7833				
3223	-134.0275	305.9442	-171.4193	-0.6221	9.0203	-7.8432	4.1876	180	30	-0.5687	-135.0839	1.0664				
3224	-147.2735	305.1502	-157.3608	-0.9387	12.373	-10.805	5.6141	180	30	-0.6068	-147.5987	0.3251				
3225	-160.9704	305.2146	-141.9961	-0.8186	15.4371	-13.9676	6.8268	180	30	-0.6436	-160.441	-0.5294				
3226	-174.4096	304.6996	-126.6796	-1.0096	18.0017	-16.3899	7.7374	180	30	-0.6789	-173.2303	-1.1793				
3227	-186.1529	302.3176	-112.1841	-1.0642	20.3275	-18.6376	8.4343	180	30	-0.7128	-187.1092	0.9564				
3228	-200.2576	299.8498	-94.7073	-1.0533	22.0594	-20.0762	8.7268	180	30	-0.745	-202.2697	2.0121				
3229	-218.3555	298.2189	-74.1413	-1.0969	23.1696	-21.1392	8.7164	180	30	-0.7756	-216.2341	-2.1215				
3230	-230.2705	296.2661	-60.229	-1.0696	23.8857	-22.1176	8.6549	180	30	-0.8045	-229.16	-1.1105				
3231	-238.9438	291.6524	-48.9955	-0.9659	24.1743	-22.1758	8.3678	180	30	-0.8316	-240.7163	1.7726				
3232	-250	285.6223	-33.4413	-0.9168	23.8857	-21.7157	7.7777	180	30	-0.8588	-250.2722	0.2722				
3233	-260.7996	281.7167	-16.6559	-0.9496	22.942	-20.8166	7.0575	180	30	-0.8802	-259.9468	-0.8518				
3234	-269.4933	277.5968	-0.6366	-0.8658	21.7319	-19.5737	6.3152	180	30	-0.9016	-269.5127	0.0191				

Figura - 4.5: Trecho da tabela de dados no Matlab.

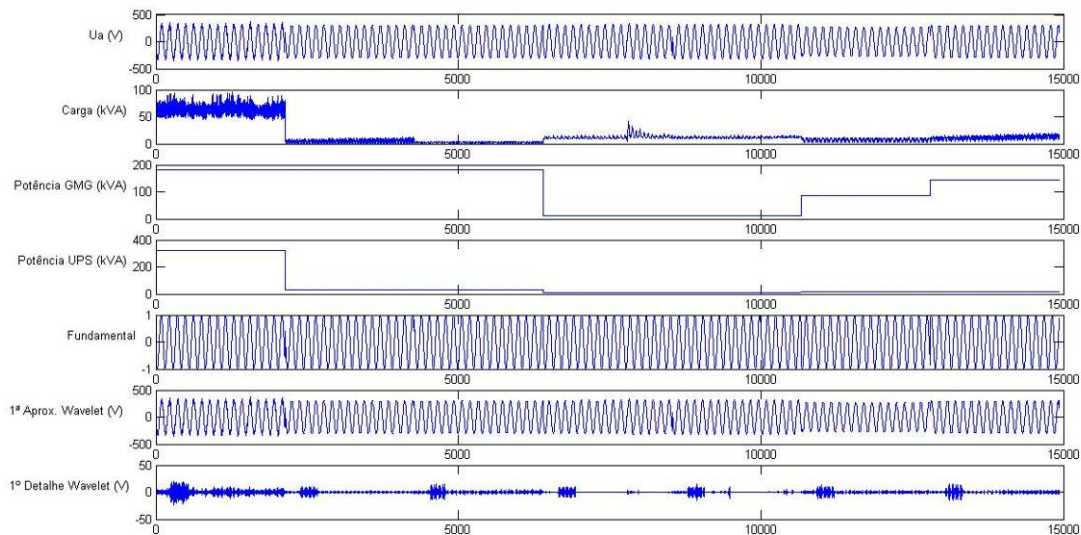


Figura - 4.6: Amostra gráfica das variáveis utilizadas na formação da rede neural

## 4.6 Rede Neural

### 4.6.1 Introdução

A utilização de uma rede neural artificial exige que uma série de escolhas, não triviais, sejam feitas na busca de um modelo que apresente um resultado considerado satisfatório. Dentre as principais escolhas estão: a topologia da rede, o algoritmo de aprendizagem, a função de ativação, a taxa de aprendizagem, a taxa de momentum e o número ideal de épocas.

Optou-se pelo modelo de rede neural *feedforward*, com um algoritmo de retropropagação, no qual os pesos são ajustados baseados na regra de aprendizagem por correção de erro, pois, de acordo com Loesch (1996), as redes *feedforward*, ou Perceptrons multicamadas constituem, provavelmente, a mais difundida e largamente usada entre todas as arquiteturas e dinâmicas de redes neurais correntes conhecidas. A rede possui capacidade de generalização, ou seja, é capaz de classificar corretamente um padrão complexo mesmo quando este não pertencer ao conjunto de treinamento da rede.

Apresentaremos a seguir os passos utilizados para o desenvolvimento da rede neural.

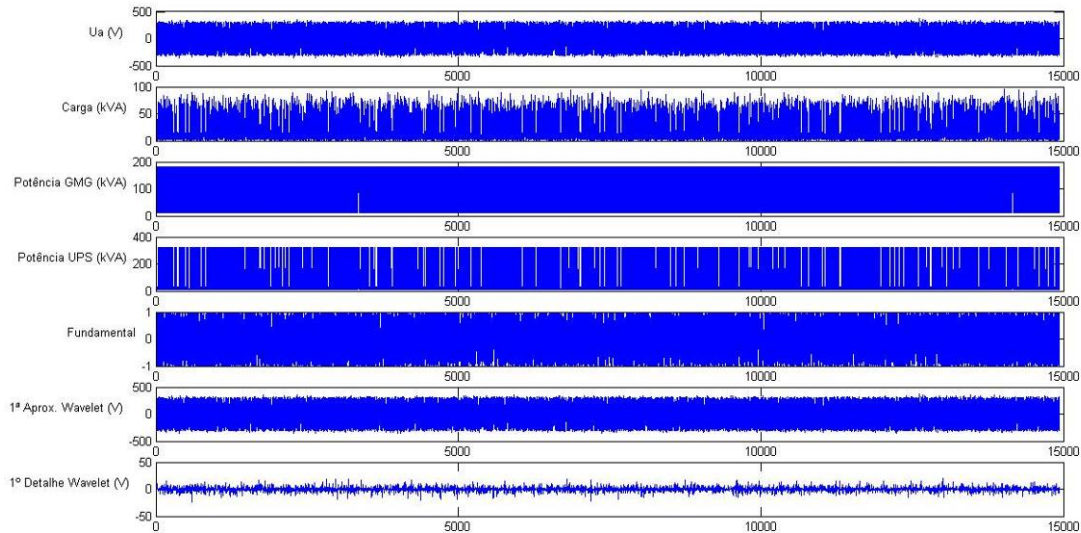
Frequentemente, é dito que o projeto de uma rede neural utilizando o algoritmo de retropropagação é mais uma arte do que uma ciência, significando que muitos dos numerosos fatores envolvidos no projeto são o resultado da experiência particular de cada um. (HAYKIN, 2001, p. 205).

### 4.6.2 Embaralhar os Dados

De acordo com HAYKIN (2001), nas tarefas de classificação de padrões usando a aprendizagem por retropropagação sequencial, uma técnica simples bastante utilizada é tornar aleatória a ordem em que os exemplos são apresentados ao perceptron de múltiplas camadas de uma época para a seguinte. Idealmente, a aleatoriedade garante que os exemplos sucessivos apresentados à rede em uma época raramente pertençam à mesma classe. Assim, melhoramos o desempenho do algoritmo de retropropagação.



Assim, a tabela de dados teve as sua seqüência de linhas embaralhadas. A Figura 4.7 ilustra um trecho das variáveis utilizadas na formação da rede neural em uma amostra gráfica após serem embaralhadas.



**Figura - 4.7:** Amostra gráfica das variáveis utilizadas na formação da rede neural após serem embaralhadas

#### 4.6.3 Normalização dos Dados

Cada variável deve ser pré-processada de modo que o seu valor médio, calculado sobre todo o conjunto de treinamento, ou seja próximo de zero, ou seja pequeno comparado com o desvio padrão (LeCun, 1993 *apud* HAYKIN, 2001). A normalização é realizada para assegurar que todas as variáveis utilizadas tenham a mesma importância durante o treinamento, por isso as funções de ativação dos neurônios artificiais são limitadas. Então, a normalização deve colocar os dados entre os limites inferior e superior da função de ativação (AQUINO *et al*, 2009).

Todos os dados foram normalizados para o intervalo  $[-1, 1]$  utilizando a equação

$$X_{norm} = \frac{(2 \cdot X - X_{max} - X_{min})}{X_{max} - X_{min}} \quad (4.5)$$

onde  $X$  é o valor da variável a ser normalizada e  $X_{norm}$  é o valor normalizado.  $X_{max}$  e  $X_{min}$  são o valor máximo e mínimo assumidos pela variável em toda a série de dados.



A atribuição dos valores de  $X_{\max}$  e  $X_{\min}$  especificamente para as variáveis “Carga”, “Potência do UPS” e “Potência do GMG” foi tratada de maneira diferente. Caso limitássemos estas variáveis pelos valores máximos e mínimos colhidos em campo, ficaríamos impossibilitados de simularmos, na rede neural desenvolvida, valores fora dessa faixa. Então atribuímos como valores máximos para estas variáveis a potência de 2.200kVA escolhida por ser o maior valor de potência encontrado em pesquisa para o fornecimento de equipamentos UPS e GMG comerciais. Para valores mínimos, atribuímos a potência de 1kVA para as variáveis “Potência do UPS” e “Potência do GMG”, por ser o menor valor de potência encontrado em pesquisa para o fornecimento de equipamentos UPS e GMG comerciais e o valor zero para a variável “Carga” para ser possível a simulação de carregamento zero nos equipamentos, conforme Tabela 4.2.

**Tabela - 4.2:** Valores máximos e mínimos utilizados na normalização dos dados

Variáveis	Normalização	
	$X_{\max}$	$X_{\min}$
Tensão da Fase A - $U_a$ (V)	Valor Máximo da Série	Valor Mínimo da Série
Potencia Aparente - Carga (kVA)	2.200	0
Potência GMG (kVA)	2.200	1
Potência UPS (kVA)	2.200	1
Fundamental	Valor Máximo da Série	Valor Mínimo da Série
1ª Aproximação <i>Wavelet</i>	Valor Máximo da Série	Valor Mínimo da Série
1º Detalhe <i>Wavelet</i> Nível 1	Valor Máximo da Série	Valor Mínimo da Série

#### 4.6.4 Representação dos Padrões de Saída

Foi dito que havia duas propostas de redes neurais para a resolução do problema, uma com apenas um sinal na saída e outra com dois sinais na saída (o mesmo sinal decomposto pela transformada *Wavelet*).

Assim, resolvemos testar os dois modelos de rede e comparamos os resultados.

Para o teste da melhor representação dos padrões de saída, utilizamos as arquiteturas propostas na Figura 4.2 e 4.3, batizando-as de “Arquitetura 1” e “Arquitetura 2” respectivamente.

Decidimos, para este teste, utilizar três neurônios na camada oculta e apenas uma camada oculta, pois, de acordo com LOESCH e SARI (1996), alguns autores sugerem que o número ideal de elementos na camada oculta seja algo em torno da média entre as camadas de entrada e saída e normalmente redes de 3 camadas são suficientes para os propósitos mais gerais.

A função de ativação sigmóide na forma de uma tangente hiperbólica foi escolhida para todos os neurônios (camada oculta e saída). Segundo HAYKIN (2001), um perceptron de múltiplas camadas treinado com o algoritmo de retropropagação pode, em geral, aprender mais rápido quando a função de ativação sigmóide incorporada no modelo do neurônio da rede for anti-simétrica do que quando for não-simétrica. Um exemplo popular de uma função de ativação anti-simétrica é uma não-linearidade de sigmóide na forma de uma tangente hiperbólica.

Para o treinamento, utilizamos o algoritmo Levenberg-Marquardt que é uma variação do algoritmo Back-Propagation (BRAGA *et al*, 2007). Em geral, em problemas de aproximação de função, para as redes que contêm até algumas centenas de pesos, o algoritmo Levenberg-Marquardt terá a convergência mais rápida. Esta vantagem é especialmente perceptível se o treinamento muito exato é necessitado. Em muitos casos, este algoritmo é capaz de obter erros médios quadráticos mais baixos do que outros algoritmos (DEMUTH *et al*; 2007).

No Matlab a função utilizada foi a “newff”, o treinamento foi limitado em 10.000 épocas e o desempenho da rede foi medido pelo erro médio quadrático.

Os dados foram divididos na seguinte proporção: 50% para treinamento, 25% para validação e 25% para teste.

Segundo HAYKIN (2001), uma boa escolha para os valores iniciais dos pesos sinápticos e limiares da rede pode ser de tremenda ajuda para um projeto de rede ser bem-

sucedido. Então, de forma aleatória, inicializamos 20 vezes as redes com 3 neurônios na camada oculta e comparamos seus resultados que estão apresentados na Tabela 4.3.

**Tabela - 4.3:** Erros médios quadráticos dos testes

<b>Inicializações</b>	<b>Arquitetura 1</b>	<b>Arquitetura 2</b>
<b>1<sup>a</sup></b>	0,1050	0,0608
<b>2<sup>a</sup></b>	0,1999	0,0613
<b>3<sup>a</sup></b>	0,1094	0,0620
<b>4<sup>a</sup></b>	0,1054	0,0603
<b>5<sup>a</sup></b>	0,1044	0,0605
<b>6<sup>a</sup></b>	0,1090	0,0607
<b>7<sup>a</sup></b>	0,1048	0,0586
<b>8<sup>a</sup></b>	0,1059	0,0626
<b>9<sup>a</sup></b>	0,6291	0,0639
<b>10<sup>a</sup></b>	0,1084	0,0609
<b>11<sup>a</sup></b>	0,1054	0,0628
<b>12<sup>a</sup></b>	0,1094	0,6658
<b>13<sup>a</sup></b>	0,1074	0,0590
<b>14<sup>a</sup></b>	0,1087	0,0630
<b>15<sup>a</sup></b>	0,1089	0,0599
<b>16<sup>a</sup></b>	0,1151	0,9626
<b>17<sup>a</sup></b>	0,1074	0,0651
<b>18<sup>a</sup></b>	0,1999	0,6703
<b>19<sup>a</sup></b>	0,1045	0,0594
<b>20<sup>a</sup></b>	0,1048	0,7001
<b>Média</b>	<b>0,14264</b>	<b>0,19898</b>

Os resultados dos testes mostram que obteríamos um melhor resultado se utilizássemos uma rede neural com apenas um neurônio na saída (Arquitetura 1) do que utilizando dois (Arquitetura 2).

Apesar desta informação, ao invés de optarmos por continuar o desenvolvimento da rede utilizado apenas um neurônio na saída, decidimos por prosseguir o trabalho utilizando ambos os experimentos e continuar comparando os resultados entre eles.

#### 4.6.5 Arquitetura da Rede

Serão agora realizados os treinamentos para selecionar a arquitetura que melhor representa o problema em questão, ou seja, serão avaliadas diversas topologias de rede a partir do número de neurônios ocultos nas redes neurais.

Segundo AQUINO *et al* (2006 e 2007), a arquitetura deve ser definida modificando o número de neurônios da camada oculta e selecionada aquela que apresenta a melhor performance no processo de treinamento.

Testamos as redes neurais com o número de neurônios da camada oculta variando de 3 até 20 neurônios.

Seguindo as mesmas premissas do item anterior, inicializamos cada rede 20 vezes para cada conjunto de neurônios na camada oculta. A Tabela 5.4 apresenta a média dos erros médios quadráticos de 20 inicializações aleatórias para cada conjunto de neurônios na camada oculta.

**Tabela - 4.4:** Média dos erros médios quadráticos variando os neurônios na camada oculta

Neurônios na camada oculta	Arquitetura 1	Arquitetura 2
3	0,1158	0,1408
4	0,1162	0,0771
5	0,1107	0,2318
6	0,1035	0,0913
7	0,1073	0,0861
8	0,1060	0,0942
9	0,1005	0,1219
10	0,1134	<b>0,0645</b>
11	0,1180	0,0994
12	0,1280	0,1299
13	0,0984	0,1573
14	0,1106	0,1947
15	0,0984	0,0703
16	<b>0,0982</b>	0,1563
17	0,1431	0,1450
18	0,1155	0,0860
19	0,1059	0,0948
20	0,1078	0,1341

Com este teste, verificamos que o número de neurônios na camada oculta que apresentaram a menor média dos erros médios quadráticos para cada arquitetura foram:

- Arquitetura Final 1: 16 neurônios (4-16-1)
- Arquitetura Final 2: 10 neurônios (4-10-2)

#### 4.6.6 Melhor Inicialização de Pesos

Após determinar a melhor arquitetura para o problema em questão, procuramos uma melhor inicialização de pesos para esta arquitetura a fim de realizar as simulações em uma rede que tenha boa escolha de pesos iniciais.

Foram testadas 20 redes com inicializações distintas para cada arquitetura e identificamos as que apresentaram o menor erro médio quadrático, conforme Tabela 4.5.

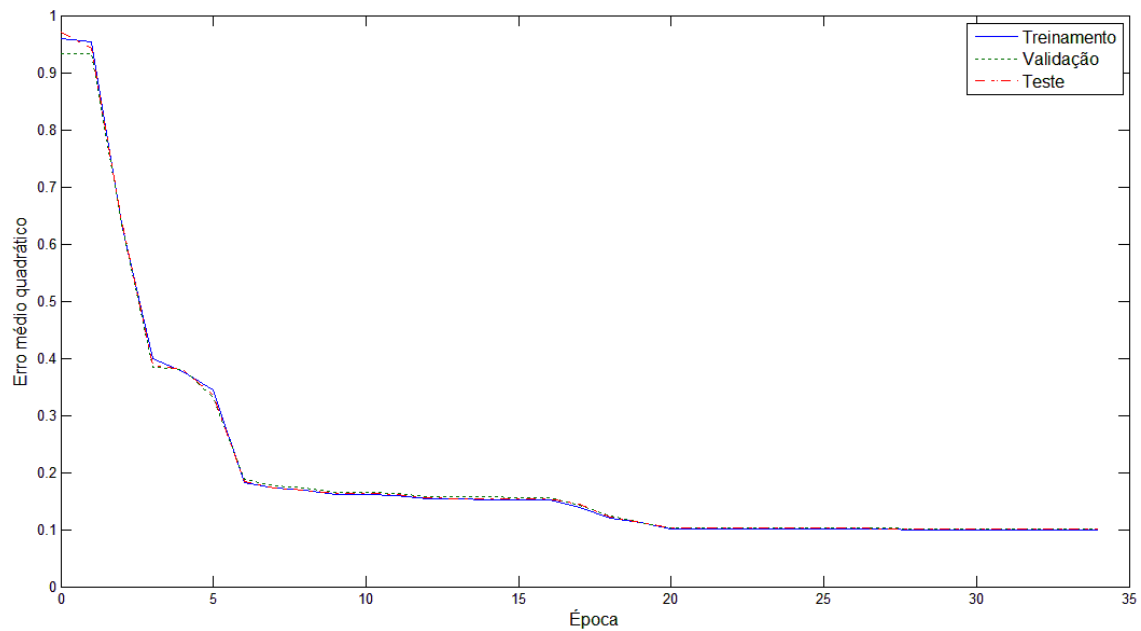
**Tabela - 4.5:** Erros médios quadráticos verificados para cada inicialização aleatória das redes após definido o n° de neurônios na camada oculta

Inicializações	Arquitetura	Arquitetura
	Final 1	Final 2
1 <sup>a</sup>	0,0961	0,0555
2 <sup>a</sup>	0,0975	0,0562
3 <sup>a</sup>	0,0961	0,0553
4 <sup>a</sup>	0,0967	0,0556
5 <sup>a</sup>	0,0958	0,0563
6 <sup>a</sup>	0,0967	0,1522
7 <sup>a</sup>	0,0995	0,0550
8 <sup>a</sup>	0,0977	0,9853
9 <sup>a</sup>	0,0960	0,6639
10 <sup>a</sup>	<b>0,0940</b>	0,2434
11 <sup>a</sup>	0,0986	0,1172
12 <sup>a</sup>	0,0980	0,0559
13 <sup>a</sup>	0,0978	<b>0,0548</b>
14 <sup>a</sup>	0,0967	0,0553
15 <sup>a</sup>	0,0957	0,0563
16 <sup>a</sup>	0,0996	0,9460
17 <sup>a</sup>	0,0983	0,3789
18 <sup>a</sup>	0,0990	0,0556
19 <sup>a</sup>	0,0956	0,0575
20 <sup>a</sup>	0,0988	0,4949

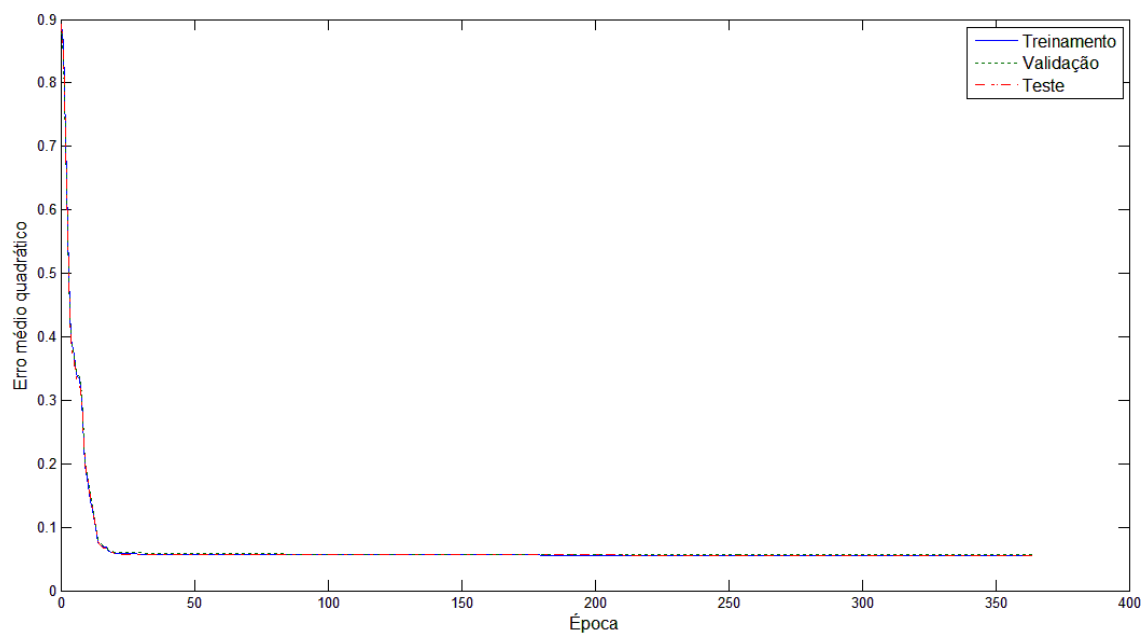
#### 4.6.7 Projeto Ótimo de Rede

Tendo-se então definido o número ótimo de neurônios na camada oculta, melhor escolha para valores iniciais de pesos sinápticos, obtemos nosso projeto ótimo de rede.

A seguir serão apresentadas as curvas de aprendizagem de cada arquitetura (gráficos de desempenho da rede):



**Figura - 4.8:** Gráfico de desempenho – Arquitetura Final 1



**Figura - 4.9:** Gráfico de desempenho – Arquitetura Final 2

Percebe-se que os gráficos apresentados não mostram oscilações do erro médio quadrático no final de suas curvas, sugerindo uma superfície de erro “bem comportada” para o problema (HAYKIN, 2001).

O desempenho de uma rede treinada pode ser medida até certo ponto por erros no treinamento, validação e dados de teste, mas, normalmente, investiga-se a resposta da rede de forma mais detalhada. Uma opção é executar uma análise de regressão entre as respostas de rede e as respostas desejadas (valores-alvo) (DEMUTH *et al*; 2007).

A seguir serão apresentadas as análises de regressão de cada experimento. Os gráficos apresentados são as respostas do programa Matlab utilizando o comando “postreg”, onde:

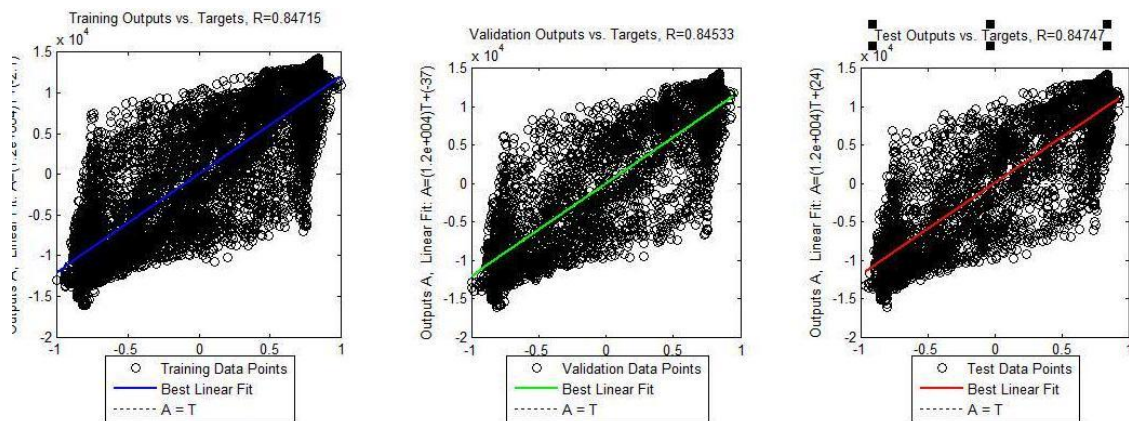
*Training Outputs* = Resposta da Rede com variáveis de treinamento;

*Validation Outputs* = Resposta da Rede com variáveis de validação;

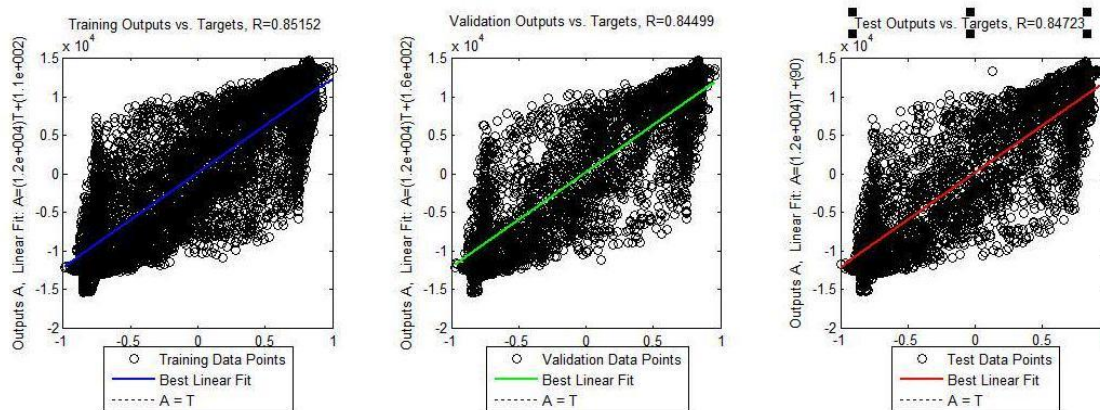
*Test Outputs* = Resposta da Rede com variáveis de teste;

*Targets* = Respostas desejadas (Valores-Alvo);

R = Coeficiente de correlação.



**Figura - 4.10:** Gráfico de regressão – Arquitetura 1



**Figura - 4.11:** Gráfico de regressão – Arquitetura 2

Percebe-se que as respostas da rede seguem as respostas desejadas razoavelmente bem com valores de R próximos de 0,9, apesar de se tratar de um problema de razoável complexidade. Neste caso a rede neural responde satisfatoriamente e pode ser utilizada com novos padrões de entrada (DEMUTH *et al*; 2007).

## 4.7 Utilizando a Rede Neural

### 4.7.1 Variáveis de Entrada

Tendo a rede neural pronta para receber novas informações de entrada e apresentar os resultados do problema, definimos que a rede seria alimentada com as variáveis em conjuntos de 2.134 dados.

A variável denominada “Sinal Fundamental” seria a mesma definida no item 4.5 e equação 4.1.

As variáveis “Potência do UPS” e “Potência do GMG” seriam definidas em duas simulações distintas (Simulação 1 e Simulação 2) ficando uma variável no seu valor máximo e outra variando entre os valores mínimos e máximos conforme descritos na Tabela 4.2.

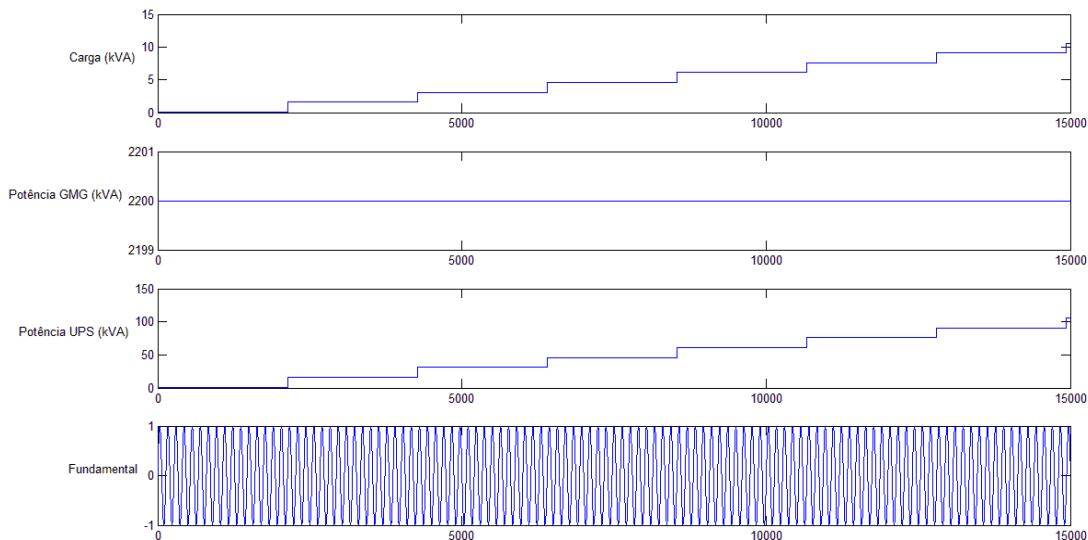
No que chamamos de “Simulação 1”, fixamos “Potência do GMG” em seu valor máximo e variamos “Potência do UPS” entre o valor mínimo e máximo. Os passos dessa variação ficaram definidos em 15kVA e cada passo formaria um conjunto de 2.134 dados. Com esta definição de valor dos passos teríamos uma entrada com 313.698 registros, em



cada variável. Com passos menores a quantidade de variáveis não seriam suportadas pelo programa Matlab.

No que chamamos de “Simulação 2”, fixamos “Potência do UPS” em seu valor máximo e variamos “Potência do GMG” entre o valor mínimo e máximo. Atribuímos os mesmos passos apresentados para “Simulação 1”. Com esses passos teríamos uma entrada com 313.698 registros, em cada variável.

A variável “Carga” teria o mesmo valor em todo o conjunto de cada simulação, porém seriam simuladas varias situações de carregamento. Definiu-se que a variável “Carga” seria um percentual do valor máximo descrito na Tabela 4.2 da seguinte forma: seriam simuladas 10 condições de carregamento, sendo elas variando entre 10% e 100% do valor máximo descrito na Tabela 4.2 a passos de 10%. A Figura 4.12 apresenta uma representação gráfica de um trecho dos dados de entrada para a “Simulação 1”.



**Figura - 4.12:** Amostra gráfica das variáveis de entrada utilizadas na rede neural

Com essas duas simulações, buscamos entender o comportamento do sinal de tensão entre o GMG e UPS para vários valores de potência entre eles, mas, evidentemente, obedecendo aos limites reais de carregamento, ou seja, um GMG de 80kVA só pode alimentar o UPS de 100kVA se esse estiver apenas com 80% de carregamento, por exemplo.

#### 4.7.2 Tratamento das Variáveis de Saída

Como resultado da rede após a aplicação dos dados de entrada, esperamos um sinal de saída que represente a tensão nos terminais do UPS. Porém, o que interessa ao trabalho é a distorção do sinal, fruto do resultado da rede neural.

A rede da “Arquitetura 2” apresenta em sua saída dois sinais que representam a decomposição *Wavelet* de 1º nível, mas o nosso interesse é apenas o sinal completo, portando para a “Arquitetura 2” trataremos o resultado como a soma das duas saídas da rede.

As respostas de saída da rede são compostas por séries de sinais com 2.134 registros, então resolvemos calcular o THD desta série e atribuí-lo ao conjunto de entrada (Potência do UPS, Potência do GMG e Carga) que o gerou.

Cada sinal de 2.134 registros contém 16 ciclos e definimos que o THD da série seria o maior THD calculado em cada um dos 16 ciclos. Como esse é um sinal discreto, definimos que cada ciclo conteria 134 registros. A Tabela 4.7 apresenta os intervalos definidos de cada ciclo.

**Tabela -4.6:** Intervalos para cada ciclo em um conjunto de 2.134 registros

Ciclo	Registro de Início	Registro Fim	Intervalo de registros
1	1	134	134
2	134	267	134
3	267	400	134
4	401	534	134
5	534	667	134
6	668	801	134
7	801	934	134
8	934	1067	134
9	1068	1201	134
10	1201	1334	134
11	1334	1467	134
12	1468	1601	134
13	1601	1734	134
14	1735	1868	134
15	1868	2001	134
16	2001	2134	134

## 4.8 Resultados

As duas simulações propostas no item 4.7.1 apresentam situações em que o UPS está sendo alimentado por GMG com capacidade de potência superior ao UPS (“Simulação 1”) e situações em que o UPS está sendo alimentado por GMG com capacidade de potência inferior ao UPS (“Simulação 2”).

Utilizando a relação entre “Potência do UPS” e “Potência do GMG” das duas simulações e o THD calculado para suas respectivas saídas, conforme item 4.7.2, pôde-se compor um gráfico único para cada “Arquitetura”, unindo os resultados das duas simulações, conforme Figuras 4.13 e 4.14.

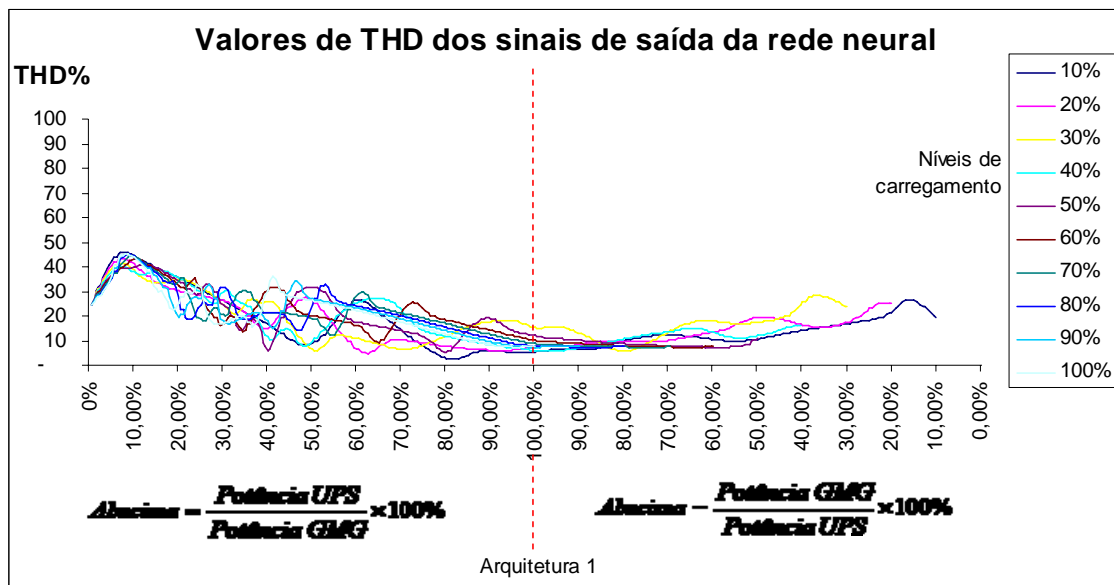


Figura - 4.13: Resposta da rede: Arquitetura 1

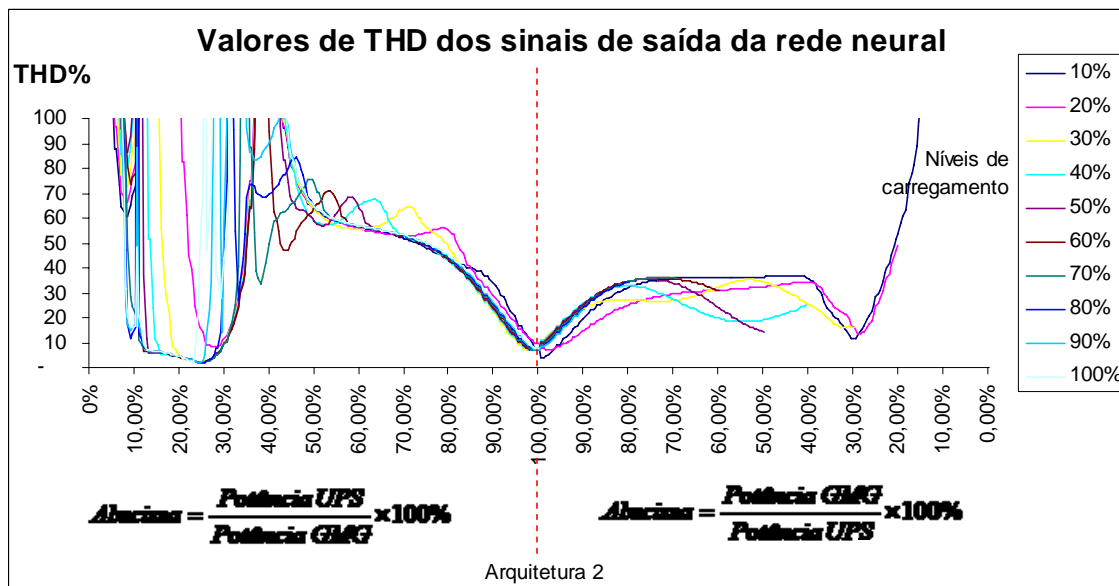
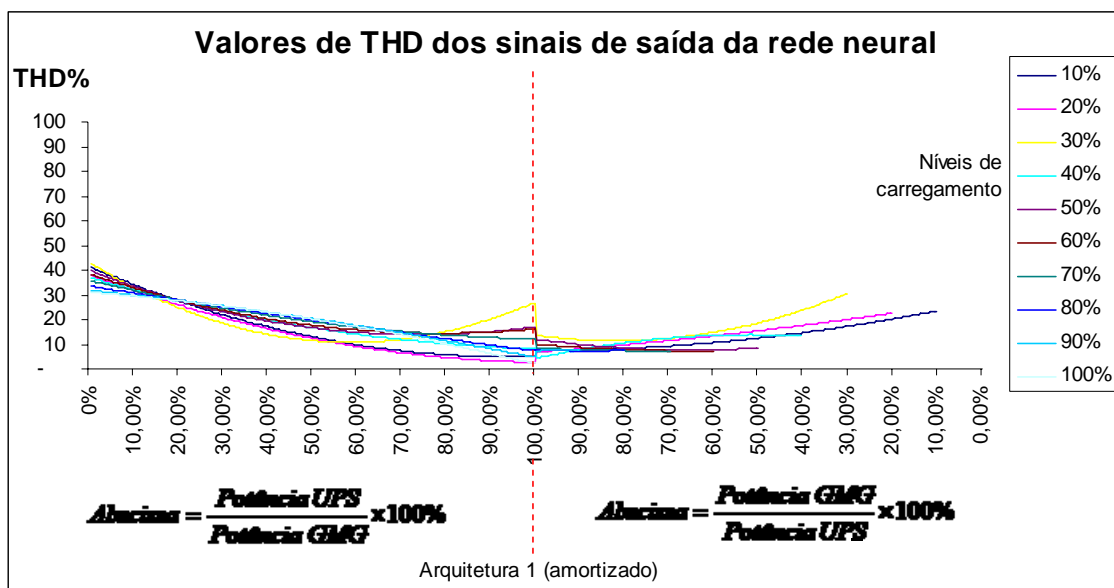


Figura - 4.14: Resposta da rede: Arquitetura 2

Observando os gráficos das redes treinadas, nas Figuras 4.13 e 4.14, percebemos que a “Arquitetura 1” (Figura 4.13) apresentou resultados “comportados” em todo o eixo das abscissas, ao passo que a “Arquitetura 2” (Figura 4.14) apresentou resultados “comportados” apenas na região central. Em anexo, Figuras A.1 a A.20, estão apresentadas as figuras comparando o THD das duas arquiteturas, 1 e 2, para cada percentual de carga, além de apresentar o eixo das abscissas em escala logarítmica.

O objetivo deste trabalho não propõe resultados exatos, mas a possibilidade de indicar uma tendência no comportamento do acoplamento de UPS e GMG. Então, utilizamos os resultados apresentados na Figura 4.13 e traçamos curvas de tendência. A resposta ajustada é apresentada na Figura 4.15.



**Figura - 4.15:** Resposta da rede: Arquitetura 1 (resultado amortizado)

## 5. CONCLUSÕES E SUGESTÕES PARA TRABALHOS FUTUROS

Problemas no acoplamento de UPS e GMG são comuns e como não há fabricantes que produzam os dois tipos de equipamentos, quando são questionados, ficam jogando o problema de um para o outro.

Estas máquinas são testadas com cargas resistivas e não há informações de seu comportamento quando alimentam cargas não-lineares.

O presente trabalho foi motivado pela possibilidade do desenvolvimento de um estudo que proporcionasse uma maior segurança no dimensionamento de sistemas com UPS e GMG. A escolha das redes neurais artificiais justifica-se, principalmente, pela sua capacidade de captar não-linearidades no sistema. O uso de transformada *wavelet* serve como auxílio para a rede neural artificial em seu aprendizado.

Com aplicação de duas topologias para a rede neural, sendo uma considerando o uso de *wavelet* e outra não, pudemos avaliar se o uso da transformada *wavelet* foi positivo ou não. Nos resultados apresentados a topologia com o uso de *wavelets* não promoveu uma resposta adequada para todas as relações UPS-GMG, porém, se considerarmos esta relação a partir de 50%, visualmente já é possível extrair conclusões de correlação entre as relações UPS-GMG e a deformação do sinal de tensão, enquanto que na topologia sem considerar a transformada *wavelet*, tivemos que traçar curvas de tendência para ter essa percepção.

Os resultados obtidos com as redes desenvolvidas neste trabalho sugerem que o menor distúrbio no sinal, obtido devido ao acoplamento de UPS e GMG, ocorrerá se as potências dos equipamentos (dados de placa) forem próximas, independente do nível de carregamento do sistema. À medida que esta relação se distancia do valor unitário, a perturbação aumenta e um UPS com potência menor que o GMG contribui mais com a distorção do sinal do que ao contrário.

A grande contribuição deste trabalho foi mostrar em seus resultados que a solução utilizada na prática, dimensionando a potência do GMG com valores superiores à potência do UPS, não contribui com o acoplamento UPS-GMG. Se essa solução prática resolve os problemas em campo, isso ocorre devido a outro fator sem ser simplesmente a relação entre a potência dos equipamentos. Na década passada o dimensionamento de condutores poderia ser feito sem considerar o tipo de carga a ser alimentada, nos tempos atuais se não

considerarmos o tipo de carga, por exemplo, o condutor NEUTRO poderá ser dimensionado de forma errada se não forem considerados os harmônicos do sistema. Assim, da mesma forma, devemos tratar projetos que envolvam este tipo de sistema, UPS-GMG, de forma mais criteriosa considerando a tecnologia da atualidade.

A utilização de redes neurais artificiais para solucionar problemas complexos, demonstrou-se eficiente e nos dá margens para inúmeros trabalhos futuros envolvendo este tema, no qual destacamos:

a) A implementação de outros tipos de redes neurais. A realização de testes com diferentes topologias de rede pode vir a melhorar o desempenho do modelo, aproximando ainda mais a saída obtida do resultado desejado.

b) A substituição da variável CARGA por uma função e apresentar valores mais adequados, visto que nesse trabalho são consideradas constantes no tempo.

c) A utilização de outros tipos de *Wavelet* mãe.

d) A utilização de outros algoritmos de treinamento.

## **ANEXO 1 – GRÁFICOS DAS REDES TREINADAS**

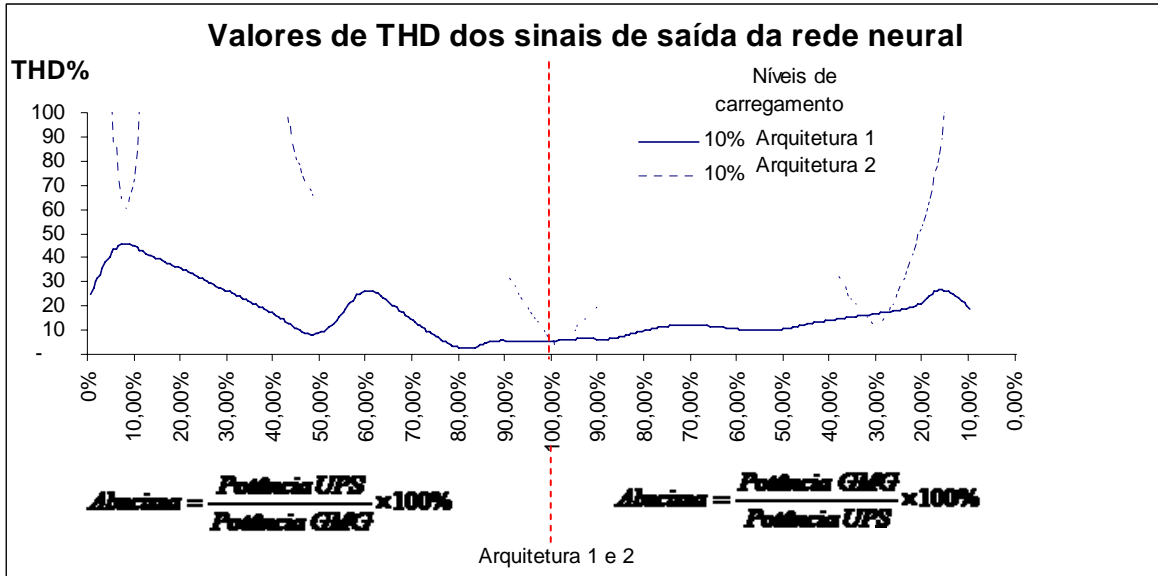


Figura - A.1: Resposta da rede: Arquitetura 1 e 2 com carregamento de 10%

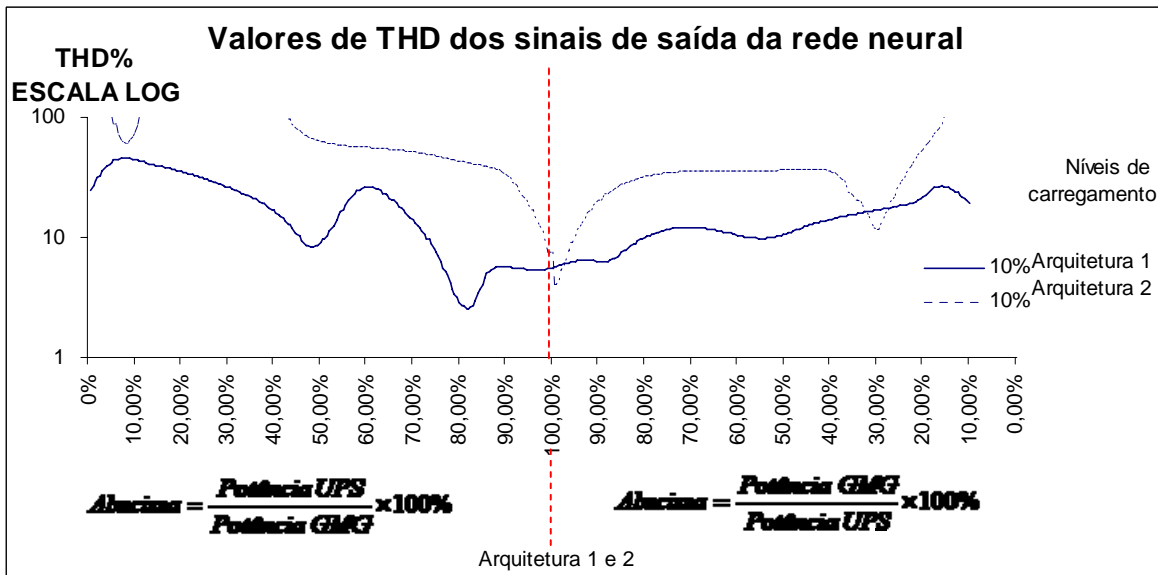


Figura - A.2: Resposta da rede: Arquitetura 1 e 2 com carregamento de 10%(escala logarítmica)



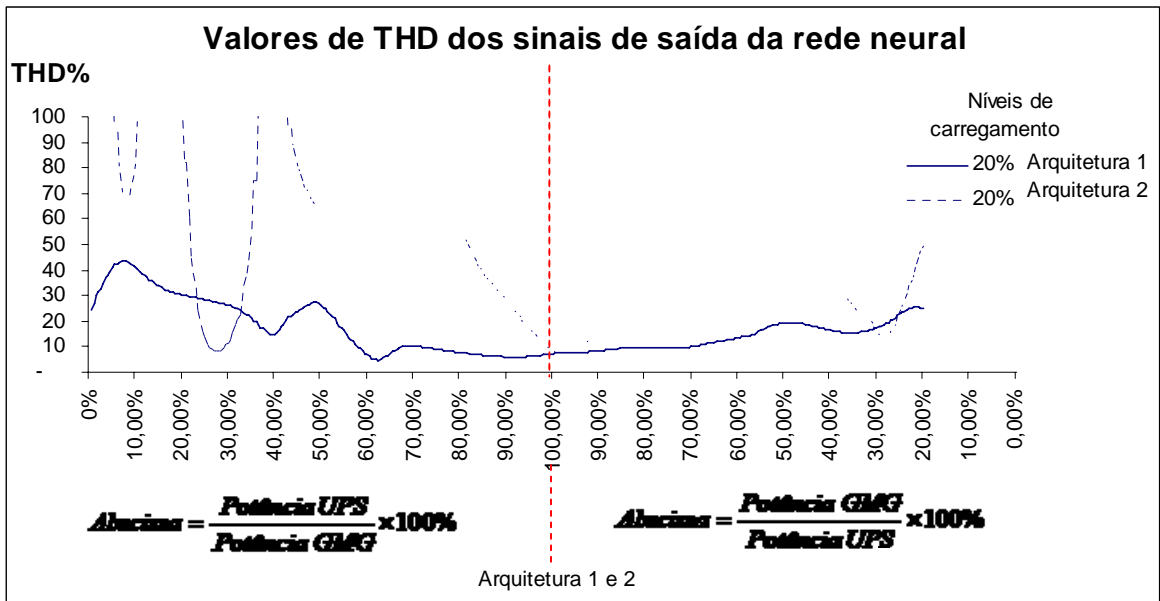


Figura - A.3: Resposta da rede: Arquitetura 1 e 2 com carregamento de 20%

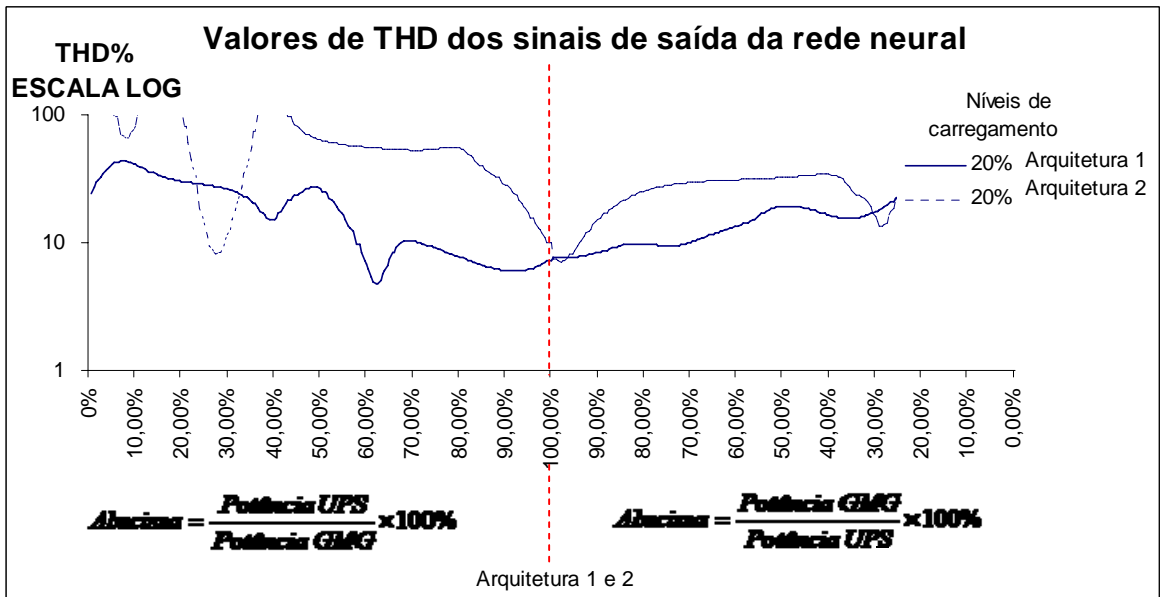


Figura - A.4: Resposta da rede: Arquitetura 1 e 2 com carregamento de 20%(escala logarítmica)

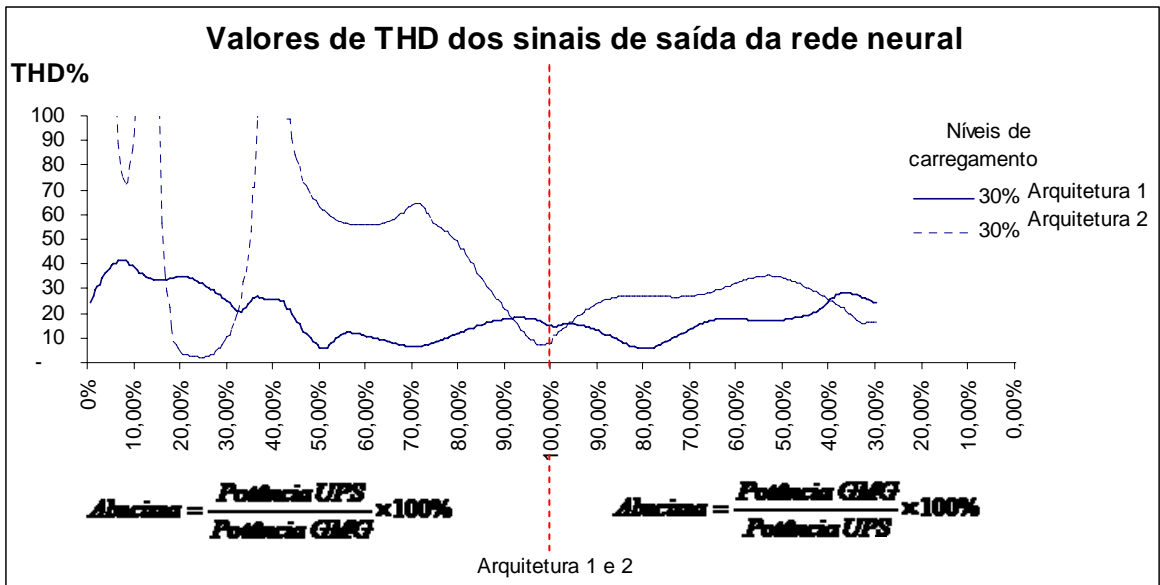


Figura - A.5: Resposta da rede: Arquitetura 1 e 2 com carregamento de 30%

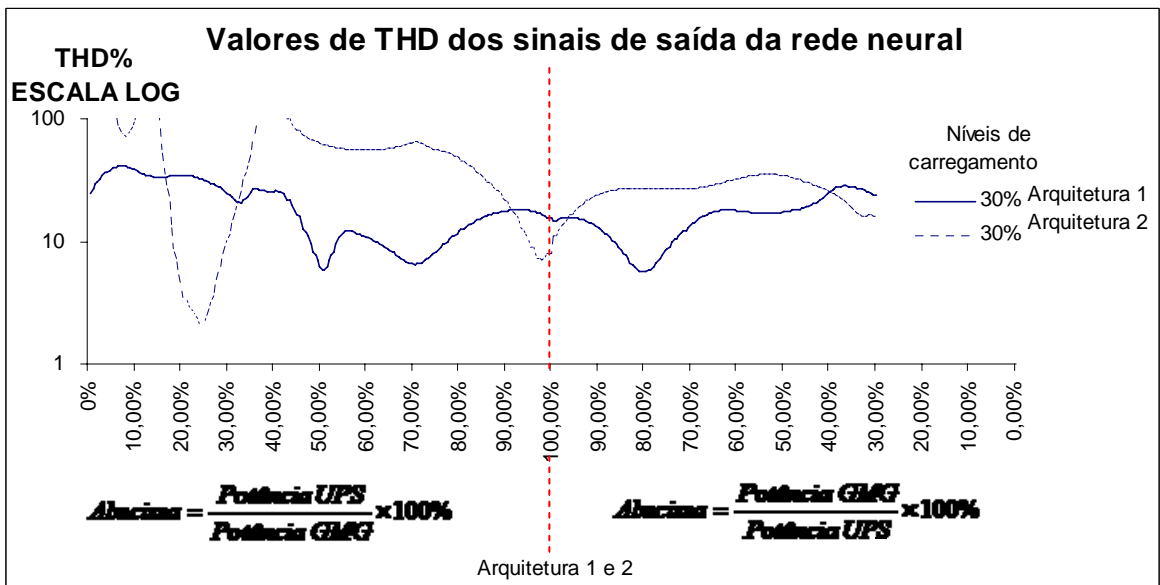


Figura - A.6: Resposta da rede: Arquitetura 1 e 2 com carregamento de 30%(escala logarítmica)

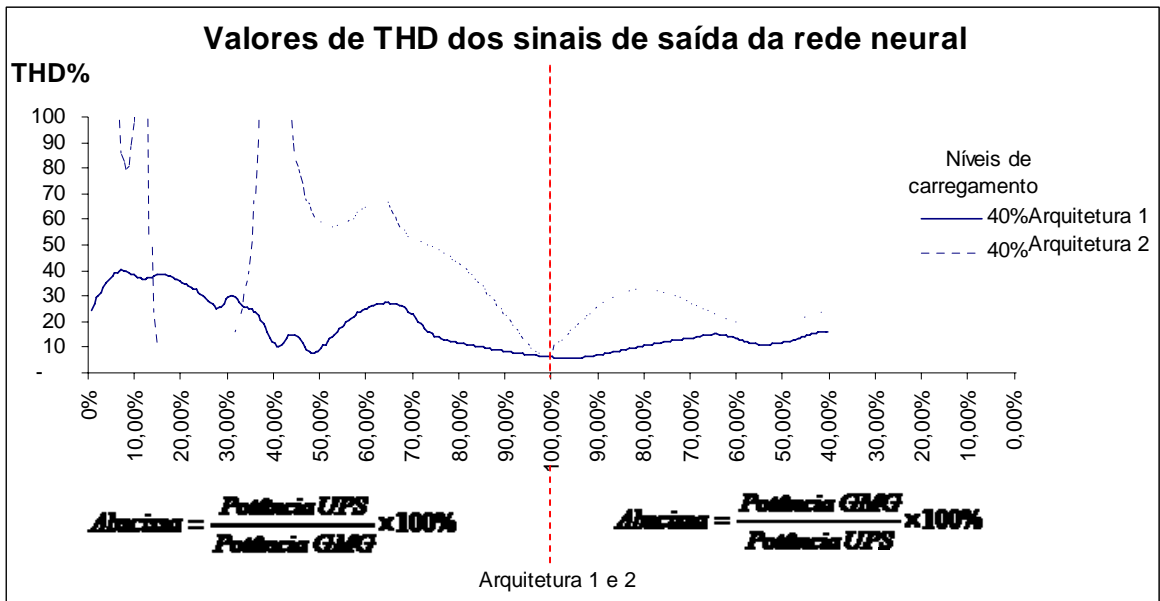


Figura - A.7: Resposta da rede: Arquitetura 1 e 2 com carregamento de 40%

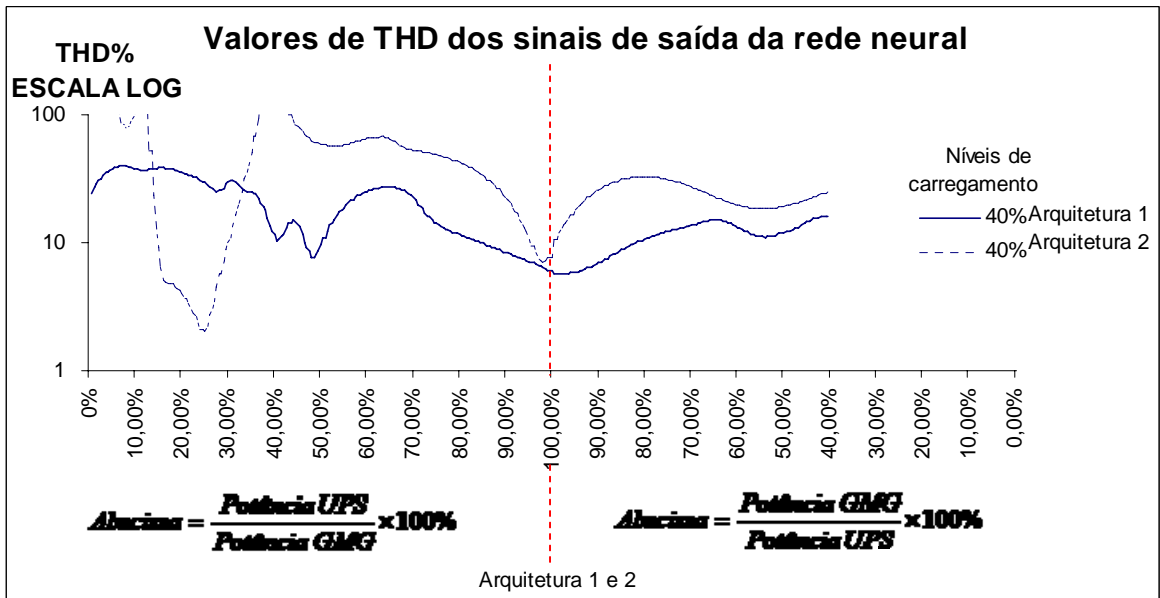


Figura - A.8: Resposta da rede: Arquitetura 1 e 2 com carregamento de 40%(escala logarítmica)

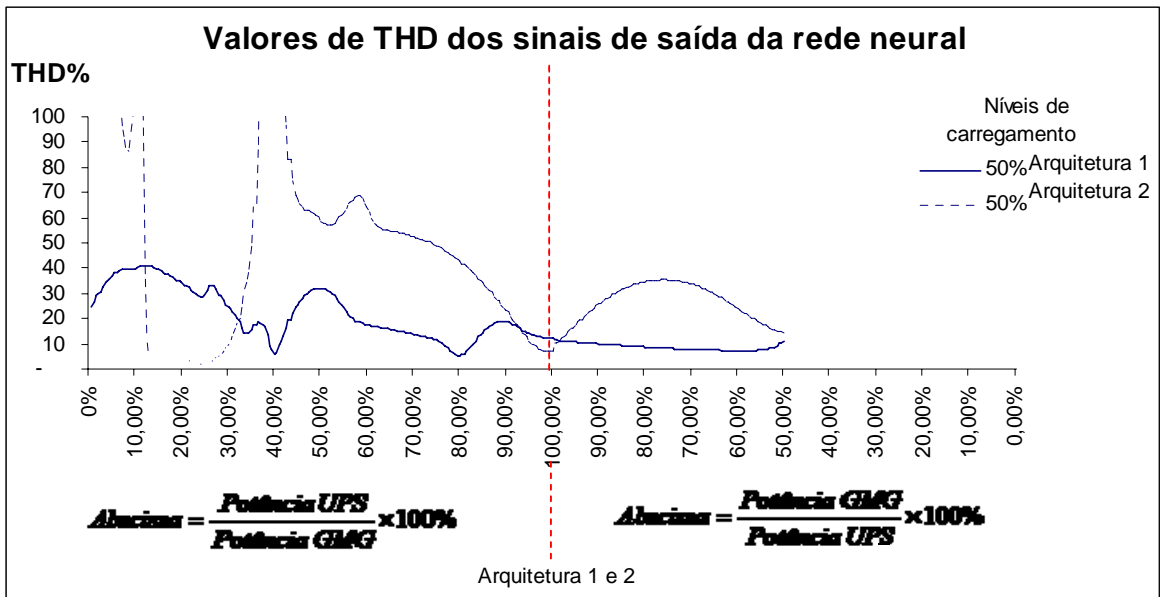


Figura - A.9: Resposta da rede: Arquitetura 1 e 2 com carregamento de 50%

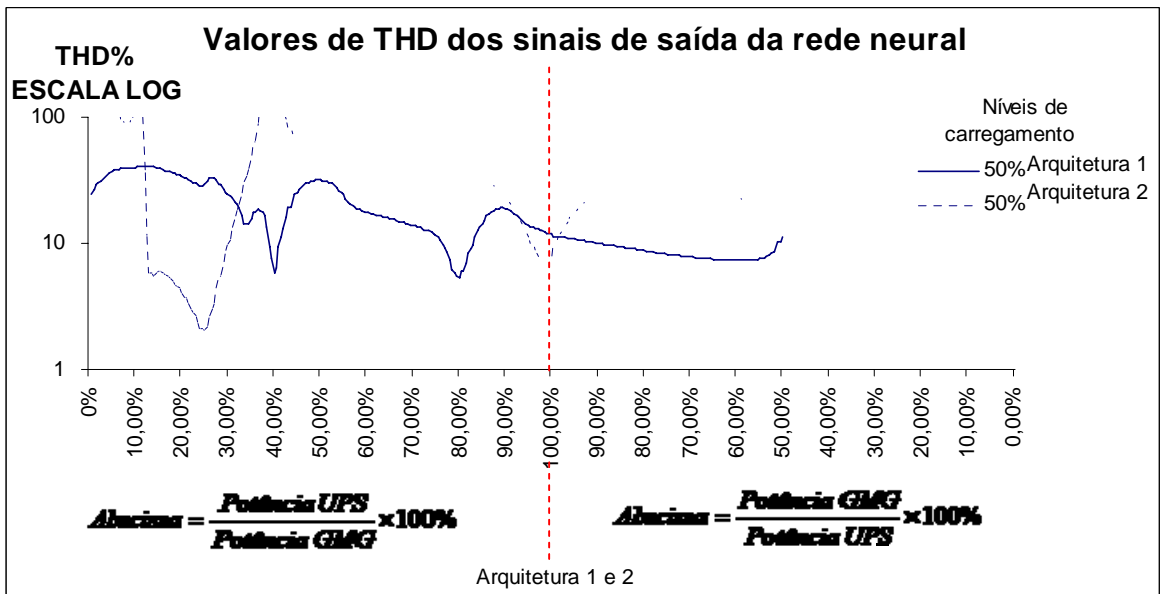


Figura - A.10: Resposta da rede: Arquitetura 1 e 2 com carregamento de 50% (escala logarítmica)

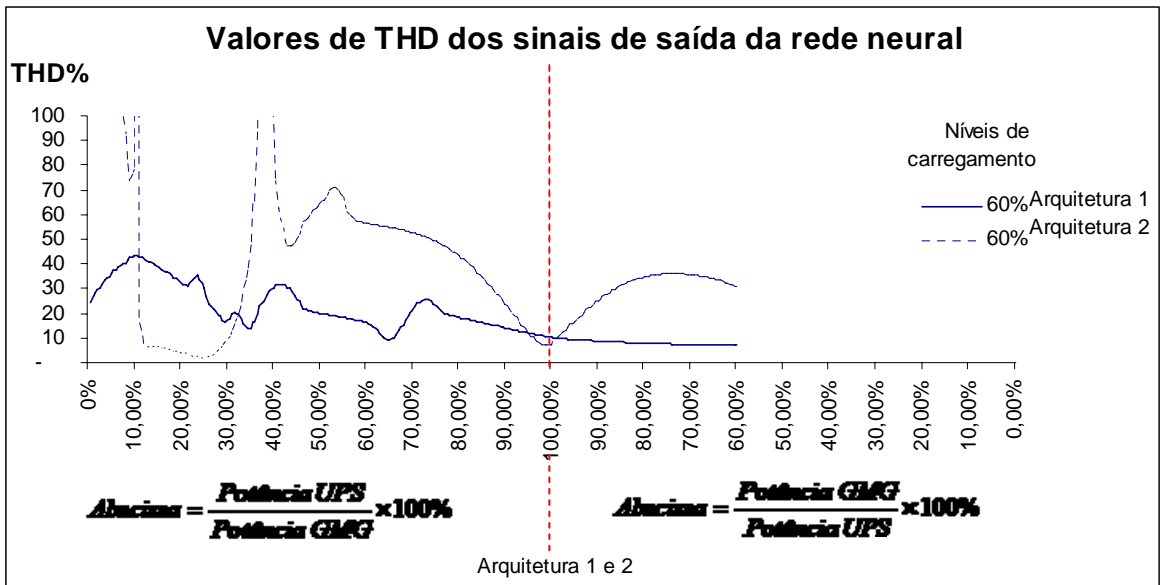


Figura - A.11: Resposta da rede: Arquitetura 1 e 2 com carregamento de 60%

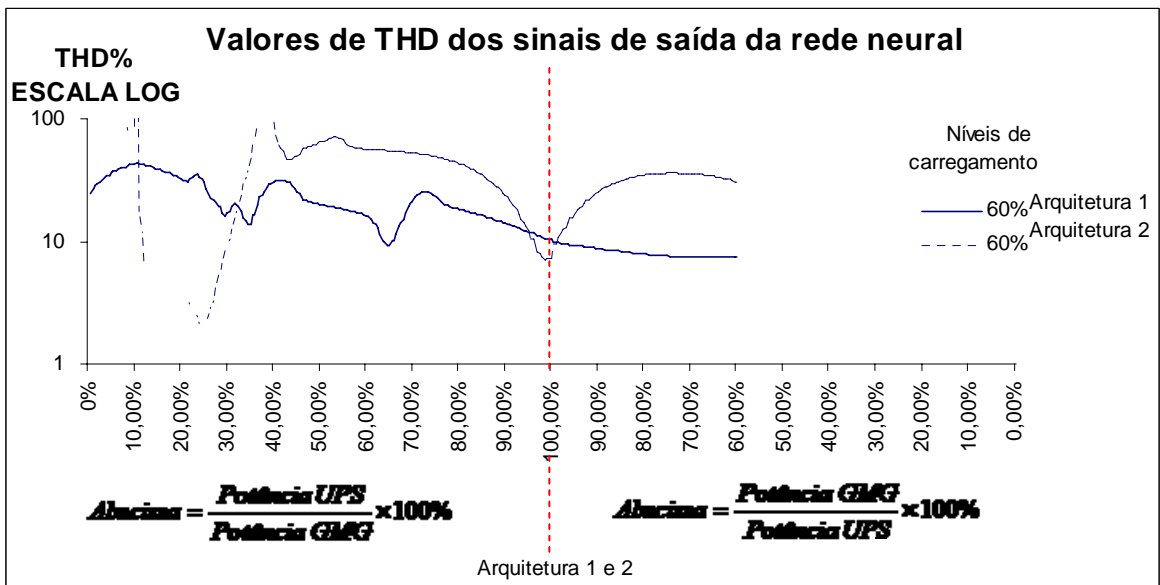


Figura - A.12: Resposta da rede: Arquitetura 1 e 2 com carregamento de 60%(escala logarítmica)

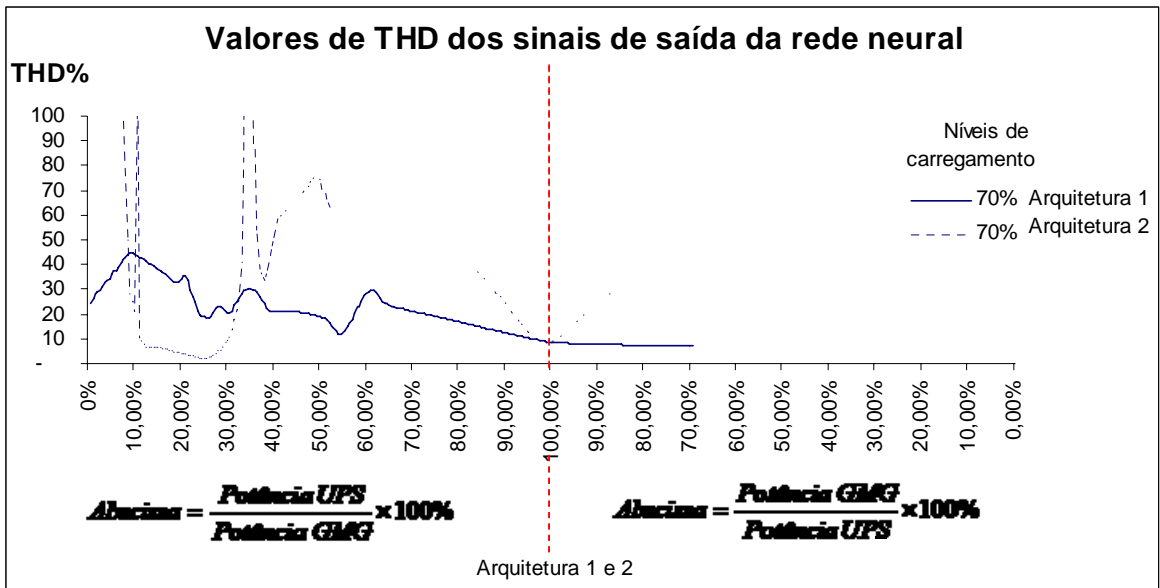


Figura - A.13: Resposta da rede: Arquitetura 1 e 2 com carregamento de 70%

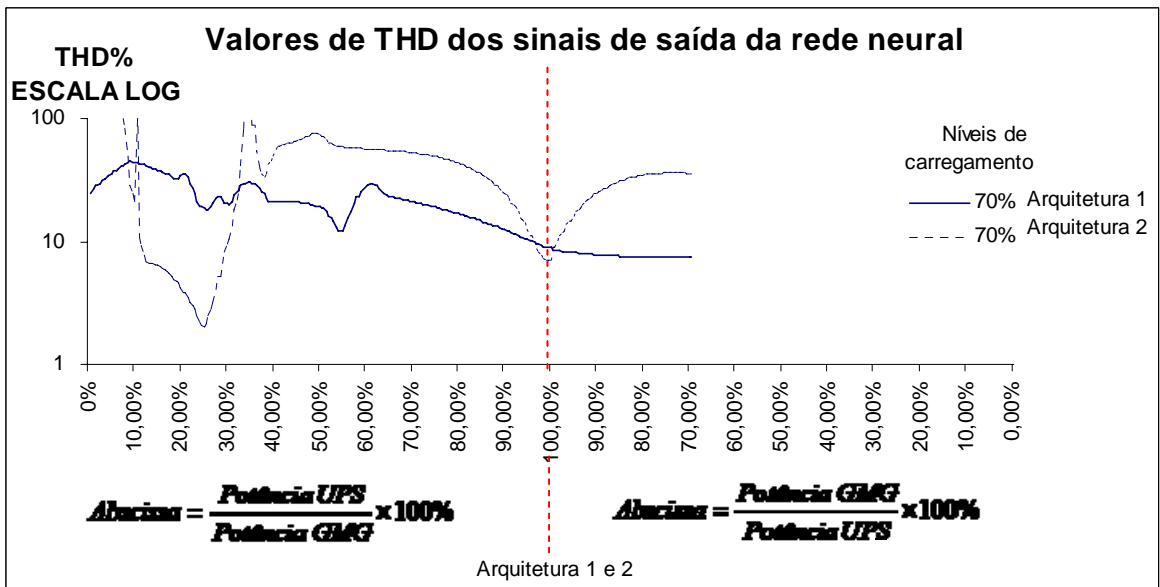


Figura - A.14: Resposta da rede: Arquitetura 1 e 2 com carregamento de 70% (escala logarítmica)

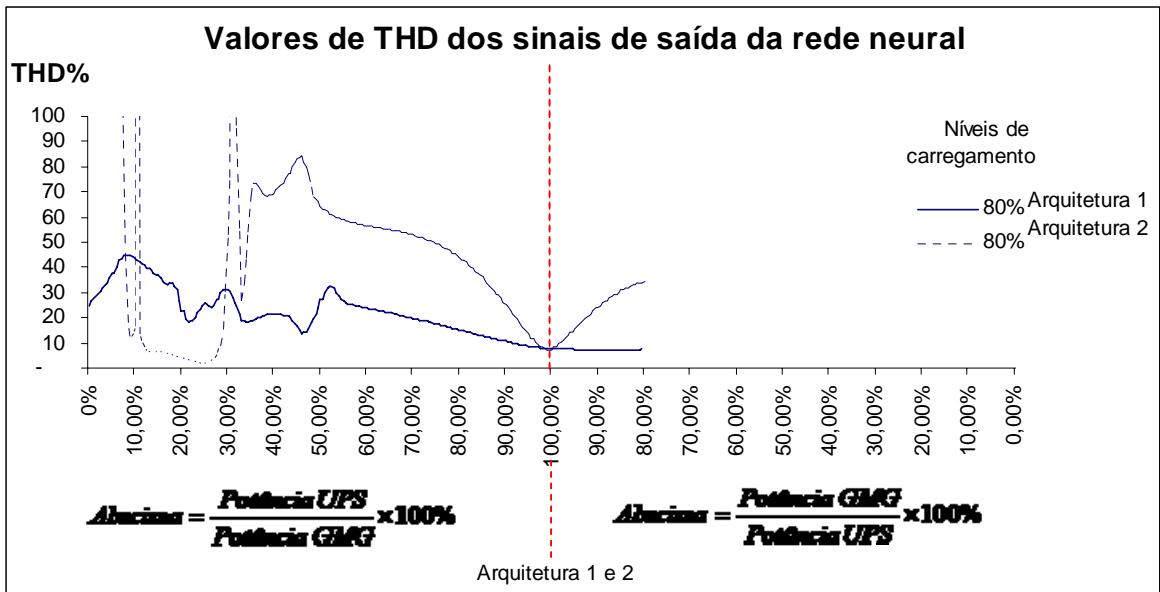


Figura - A.15: Resposta da rede: Arquitetura 1 e 2 com carregamento de 80%

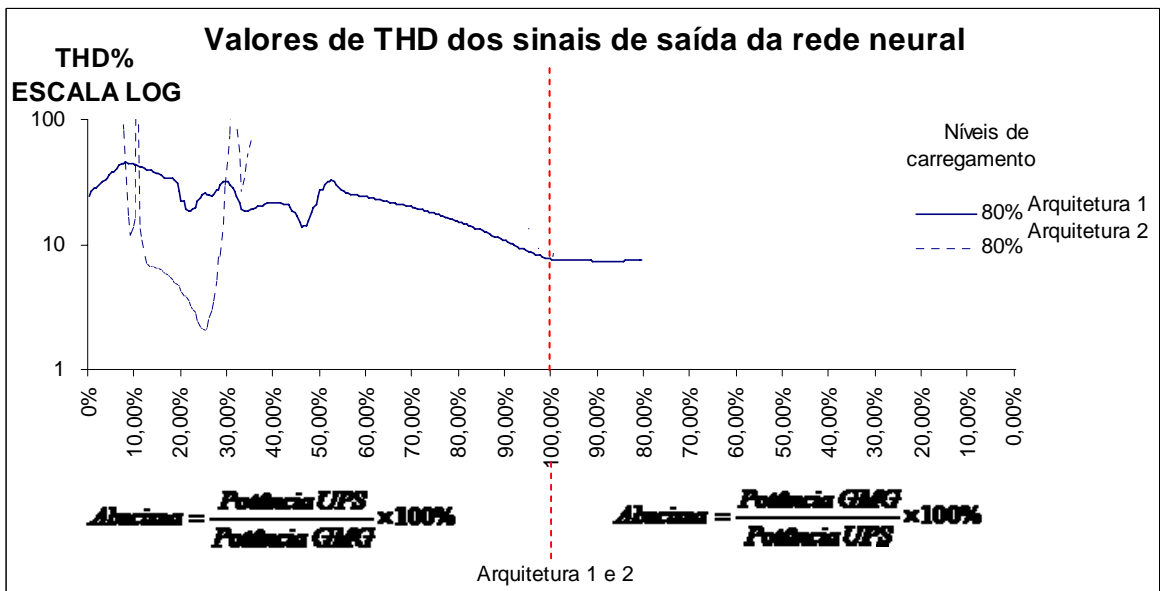


Figura - A.16: Resposta da rede: Arquitetura 1 e 2 com carregamento de 80%(escala logarítmica)

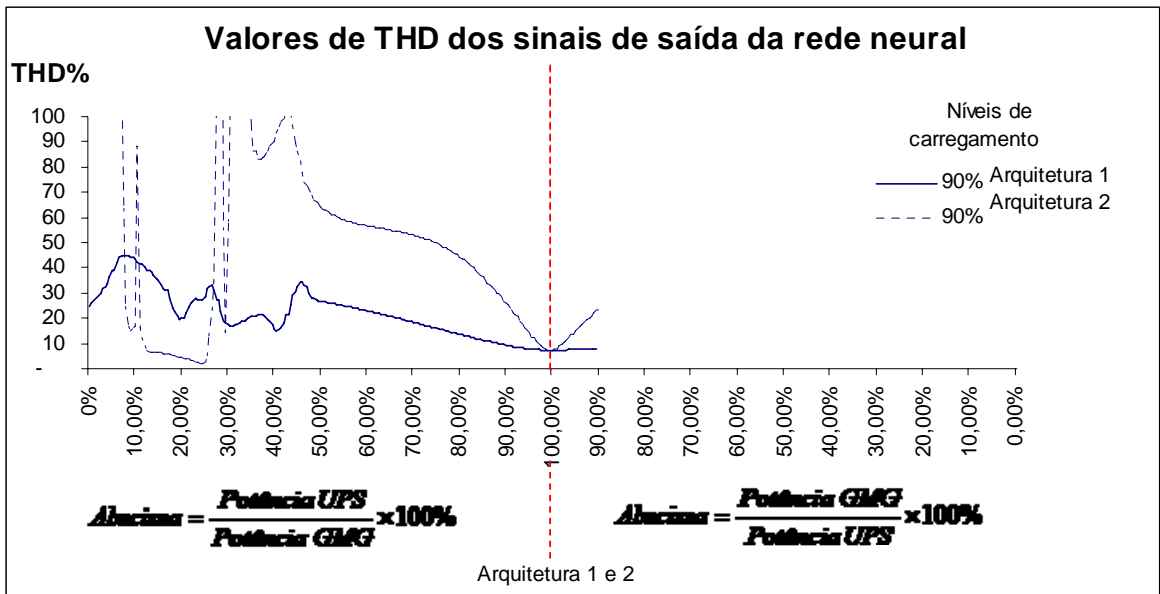


Figura - A.17: Resposta da rede: Arquitetura 1 e 2 com carregamento de 90%

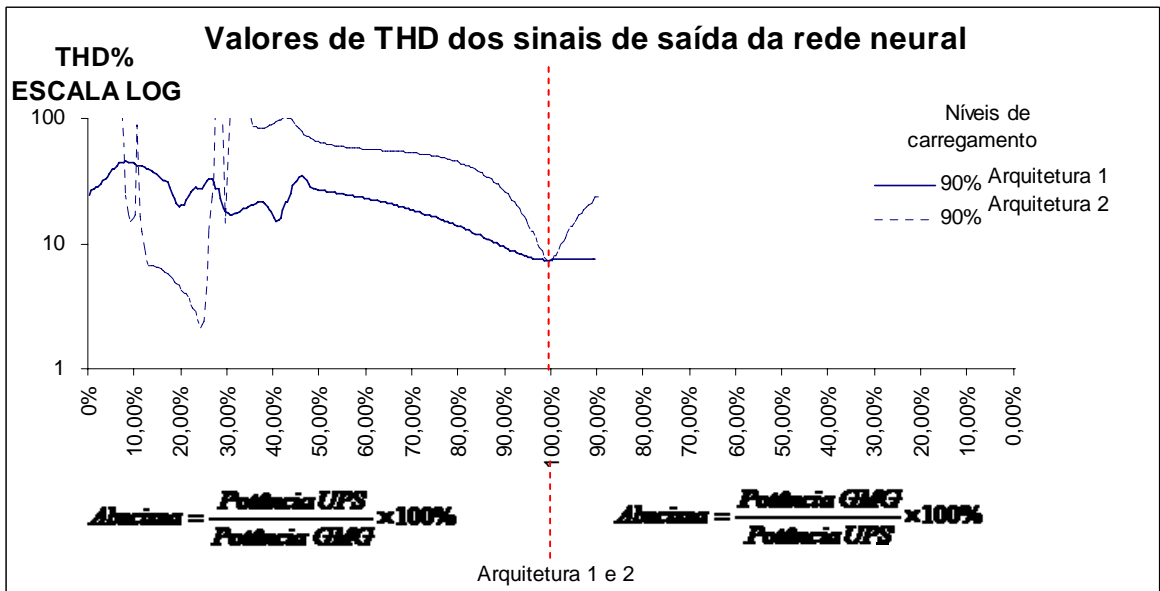


Figura - A.18: Resposta da rede: Arquitetura 1 e 2 com carregamento de 90% (escala logarítmica)



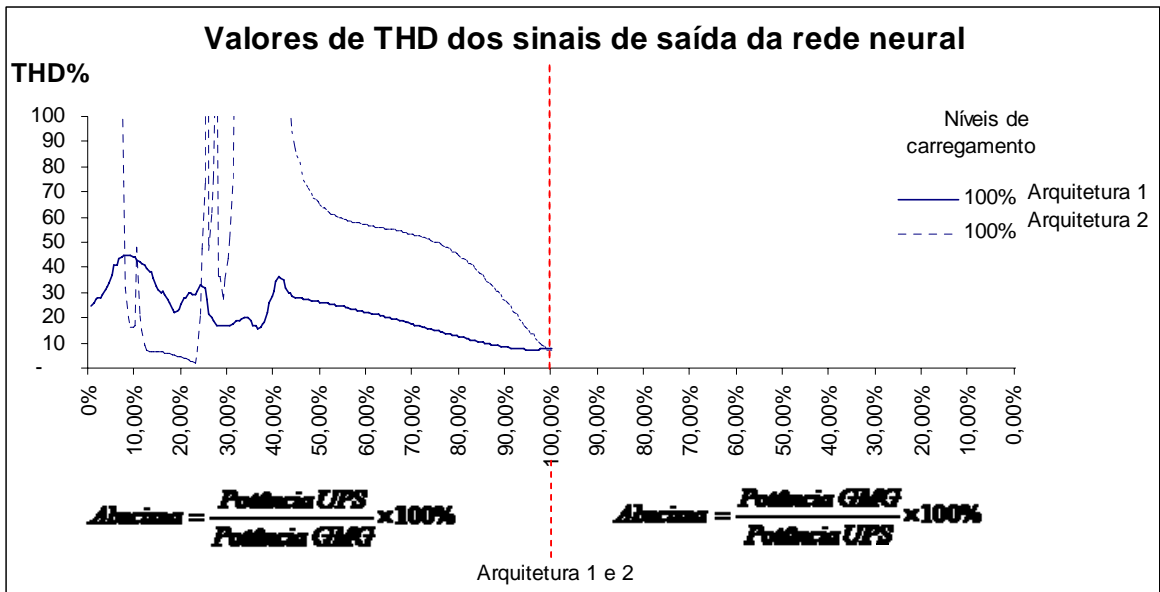


Figura - A.19: Resposta da rede: Arquitetura 1 e 2 com carregamento de 100%

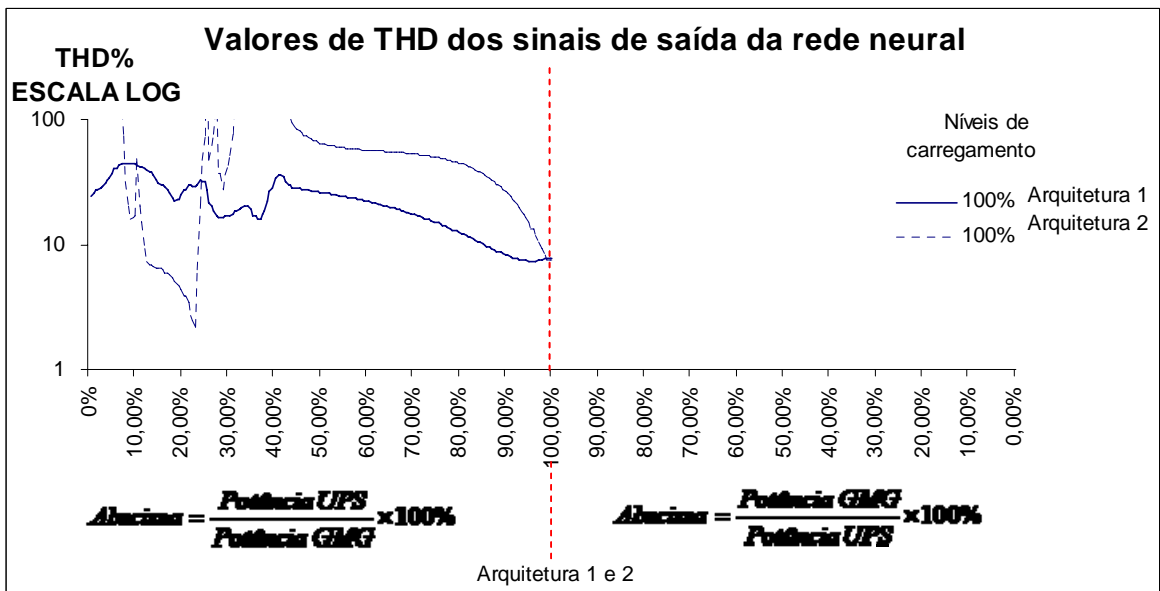


Figura - A.20: Resposta da rede: Arquitetura 1 e 2 com carregamento de 100%(escala logarítmica)

## REFERÊNCIAS BIBLIOGRÁFICAS

- AQUINO, R. R. B.; FERREIRA, A. A.; LIRA, M. M. S. *et al.*; 2006. **A hybrid intelligent system for short and mid-term forecasting for the CELPE Distribution Utility.** IEEE World Congress on Computational Intelligent, IJCNN,. IEEE Xplore, pp. 2556-2661.
- AQUINO, R. R. B.; NETO, O. N.; LIRA, M. M. S. *et al.*; 2007. **Using Genetic Algorithm to Develop a Neural-Network-Based Load Forecasting.** Lecture Notes in Computer Science, vol. 4669, pp. 738/1611-3349-747.
- AQUINO, R. R. B.; LIRA, M. M. S.; OLIVEIRA, J. B. *et al.*; 2009. **Application of Wavelet and Neural Network Models for Wind Speed and Power Generation Forecasting in a Brazilian Experimental Wind Park.** s.l.:s.n.
- BORRÁS, D.; CASTILLA, M.; MORENO, M. *et al.*; 2001. **Wavelet and Neural Structure: A new Tool for Diagnostic of Power System Disturbances.** IEEE Trans. on Industry Applications, vol. 37, no.1, p. 184-190, Jan.
- BULTTHEEL, A.; 1995. **Learning to Swim in a Sea of Wavelets.** Bull. Belg. Math. Soc., 2 v, p.1-46.
- BRAGA, A. P.; CARVALHO, A. C. P. L. F.; LUDERMIR, T. B.; 2007. **Redes neurais artificiais: teoria e aplicações.** 2.ed. Rio de Janeiro:LTC.
- DAUBECHIES, I.; 1992. **Ten lectures on wavelets.** Philadelphia: SIAM Books.
- DEMUTH, H.; BEALE, M.; HAGAN, M.; 2007. **Matlab: Neural Network Toolbox User's Guide.** Natick: Math Works.
- FAUSETT, L.; 1994. **Fundamentals of Neural Networks - Architecture, Algorithms, and Applications.** [S.l.]: Prentice Hall International, Inc.

- FARIA, R. R. A.; 1997. **Wavelets e as artes multirresolucionárias**. (s.l.:s.n.,1997).  
Disponível em: <http://www.lsi.usp.br/~regis/wlets.html>. Acessado em: 12 jul. 2010
- GU, Y.; BOLLEN, M. H. J.; 2000. **Time-Frequency and Time-Scale Domain Analysis of Voltages Disturbances**. IEEE Transactions on Power Delivery, 15 v, p. 1279-1284, Oct.
- HAYKIN, S.; 2001. **Redes Neurais, Princípios e Prática**. 2.ed. Porto Alegre: Bookman.
- INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS; 1996. **IEEE Recommended for Emergency and Stand-by Power Systems for Industrial and Comercial Applications**. USA: IEEE .
- KIM, C. H.; AGGARWAL, R.; 2000. **Wavelet Transforms in Power Systems. Part1: General Introduction to the Wavelet Transforms**. IEEE Power Engineering Journal, p.81-87, Apr.
- LIRA, M. M. S.; JUNIOR, M. A. C.; AQUINO, R. R. B. *et al.*; 2004. **Classificação de Distúrbios Elétricos Utilizando Wavelets, Componentes Principais e Redes Neurais Artificiais**. s.l.:s.n.
- LIRA, M. M. S.; AQUINO, R. R. B.; FERREIRA, A. A *et al.*; 2007. **Combining Multiple Artificial Neural Networks Using Random Committee to Decide upon Electrical Disturbance Classification**. s.l.:s.n.
- LOESCH, C.; SARI, S. T.; 1996. **Redes neurais artificiais: fundamentos e modelos**. Blumenau: Ed. Da FURB.
- MALLAT, S.; 1989. **A Theory for Multiresolution Signal Decomposition: The Wavelet Representa-tion**. IEEE Trans. Pattern Analysis and Machine Intelligence, 11 v., p.674-693, July.

- MISITI, M.; MISITI, Y.; OPPENHEIM, G. *et al.*; 1998. **Matlab: Wavelet Toolbox User's Guide**. Natick: Math Works.
- NILSSON, N. J.; 1998. **Artificial Intelligence: A New Synthesis**. San Francisco: Morgan Kaufmann Publishers, Inc..
- OLIVEIRA, H. M.; 2007. **Análise de sinais para engenheiros: uma abordagem via Wavelets**. Rio de Janeiro: Brasport.
- POISSON, O.; RIOUAL, P.; MEUNIER, M.; 2000. **Detection and Measurement of Power Quality Disturbances Using Wavelet Transform**. IEEE Transactions on Power Delivery, 15 v., p. 1039-1044, July.
- Ribeiro, P.F.; 1994. **Wavelet Transform: An Advanced Tool for Analyzing Non-Stationary Harmonic Distortions in Power Systems**. Proceedings IEEE ICHPS VI, Bologna, Italy, pp. 365-369, Sep.
- RIOUL, O.; VETTERLI, M.; 1991. **Wavelets and signal processing**. IEEE Signal Processing Magazine, New York, v.8, n.4, p. 14-38.
- SANTOSO, S.; POWERS, E. J.; GRADY, W. M. *et al.*; 2000. **Power Quality Disturbance Waveform Recognition Using Wavelet-Based Neural Classifier – Part 1: Theoretical Foundation**. IEEE Trans. on Power Delivery, vol. 15, no. 1, pp. 222-228, Jan.
- SILVA, A.V.; EYNG, J. **Wavelets e wavelet de packets**. In: SEMINÁRIO VISÃO COMPUTACIONAL, 2, 2000, Florianópolis. Visão computacional. Florianópolis: PPGCC-INE-UFSC, 2000. Disponível em: <http://www.inf.ufsc.br/~visao/2000/Wavelets/index.html>. Acessado em: 14 jul. 2010.