UNIVERSIDADE FEDERAL DE OURO PRETO - ESCOLA DE MINAS
DEPARTAMENTO DE ENGENHARIA CIVIL
PROGRAMA DE PÓS – GRADUAÇÃO EM ENGENHARIA CIVIL

ANÁLISE DE CONFIABILIDADE ESTRUTURAL UTILIZANDO O MÉTODO DE MONTE CARLO E REDES NEURAIS

ANDERSON HENRIQUE BARBOSA

ORIENTADORES: Prof. Dr. Marcílio Sousa da Rocha Freitas
Prof. Dr. Francisco de Assis das Neves

Dissertação apresentada ao Programa de Pós-Graduação do Departamento de Engenharia Civil da Escola de Minas da Universidade Federal de Ouro Preto, como parte integrante dos requisitos para obtenção do título de Mestre em Engenharia Civil, área de concentração: Construção Metálica.

Ouro Preto, Dezembro de 2004.

Dedico este trabalho às pessoas mais importantes na minha vida:

Meus pais e à Priscila

Agradecimentos

Primeiramente a Deus, por ter iluminado o meu caminho ao longo deste tempo e por ter dado paz e sobriedade nos momentos de maior necessidade.

À minha esposa Priscila, pelo carinho, dedicação e compreensão que muito contribuíram para o desenvolvimento desta dissertação.

Aos meus pais, pelo carinho e apoio.

Aos professores Marcílio Sousa da Rocha Freitas e Francisco de Assis das Neves pela orientação fornecida durante a elaboração deste trabalho.

Ao professor Flávio Barboza de Lima, da Universidade Federal de Alagoas, meus sinceros agradecimentos pela amizade, pelo incentivo e pelos conselhos nas horas de maior precisão.

Aos professores e amigos da Escola de Minas que em muito contribuíram no decorrer desta caminhada.

À USIMINAS, pela concessão da bolsa de pesquisa durante o período de vigência do curso.

Resumo da Tese apresentada ao PROPEC/UFOP como parte dos requisitos necessários a obtenção do grau de Mestre em Ciências (M.Sc.)

ANÁLISE DE CONFIABILIDADE ESTRUTURAL UTILIZANDO O MÉTODO DE MONTE CARLO E REDES NEURAIS

Anderson Henrique Barbosa

Dezembro/2004

Orientadores: Prof. Dr. Marcílio Sousa da Rocha Freitas
Prof. Dr. Francisco de Assis das Neves

A análise de confiabilidade estrutural em geral apresenta algumas restrições para alcançar uma solução. Os métodos analíticos FORM e SORM apresentam alguns problemas em função da complexidade da análise, que gera dificuldades na determinação dos pontos de mínimo. O método de simulação de Monte Carlo, embora seja de fácil implementação e absolutamente geral, o grande número de simulações pode exigir um tempo de processamento elevado, o que pode tornar sua aplicação inviável. Este problema tem sido resolvido através de técnicas de redução de variância tais como Amostragem por Importância e Esperança Condicionada. Neste trabalho propõe-se a aplicação de uma rede neural treinada para a substituição de etapas necessárias ao método de Monte Carlo, assim como da substituição do processo de análise estrutural e de confiabilidade, com o objetivo de reduzir o custo computacional requerido na análise. As redes utilizadas neste trabalho são do tipo backpropagation, fazendo-se uso do algoritmo de Levenberg – Marquartdt e do algoritmo do gradiente descendente com momentum. A aplicação das redes neurais, tanto atuando em conjunto com o método de Monte Carlo quanto substituindo toda a análise, proporcionou bons resultados com baixo custo computacional, o que atesta a viabilidade de sua aplicação.

Abstract of thesis presented to PROPEC/UFOP as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

STRUCTURAL RELIABILITY ANALYSIS USING THE MONTE CARLO METHOD AND NEURAL NETWORKS

Anderson Henrique Barbosa

December/2004

Advisors: Prof. Dr. Marcílio Sousa da Rocha Freitas
Prof. Dr. Francisco de Assis das Neves

In general, the structural reliability analysis presents some restrictions to reach a solution. The analytical methods FORM and SORM present some problems in function of the complexity of the analysis, that generates difficulties in the determination of the minimum points. The Monte Carlo method, even so it is of easy implementation and absolutely general, the great number of simulation can demand a time of raised processing, what it can become its impracticable application. This problem has been decided through techniques of variance reduction such as Importance Sampling and Conditional Expectation. In this work it is considered application of a trained neural network for the substitution of necessary stages to Monte Carlo method, as well as of the substitution of the process of structural analysis and reliability, with the objective to reduce the required computational cost in the analysis. The neural networks used in this work are of the type backpropagation, becoming use of the algorithm of Levenberg – Marquardt and of the algorithm of the descendent gradient with momentum. The application of the neural nets, as much acting in set with method Carlo the mount how much substituting all the analysis, provided good results with low computational cost, what it certifies the viability of its application.

Sumário

Lista de Tabelas

CAPÍTULO 3:

CAPÍTULO 5:

Lista de Figuras

CAPÍTULO 2:

CAPÍTULO 3:

CAPÍTULO 4:

CAPÍTULO 5:

Lista de Símbolos

Letras Romanas Maiúsculas:

A	Matriz de autovalores
A	Área da seção transversal
C	Matriz de covariância
CF	Critério de falha
D	Matriz auxiliar
E	Módulo de elasticidade.
E(**X**)	Esperança matemática das variáveis aleatórias básicas
E_C	Módulo de elasticidade da coluna
E_R	Erro quadrático da saída da rede.
E_V	Módulo de elasticidade da viga
F_{xm}	Função distribuição de probabilidade acumulada da variável X_m
$F_x(X)$	Função distribuição de probabilidade acumulada
$F_{xi}(X_n/X_1\ X_2\ ...\ X_{i-1})$	Função distribuição de probabilidade acumulada da variável condicionada
G(**X**)	Função de falha
H	Carga horizontal aplicada
H	Matriz Hessiana
I	Momento de inércia
I(.)	Função indicadora
I_C	Momento de inércia da coluna
$\mathbf{I_D}$	Matriz identidade
I_V	Momento de inércia da viga
J	Matriz Jacobiana
L	Comprimento

L	Matriz triangular inferior obtida da decomposição de Cholesky da matriz dos coeficientes de correlação
P	Carga concentrada
P	Vetor de parâmetros ajustáveis da rede
P_f	Probabilidade de falha
R	Resistência
Si	Fator de escala (método de Amostragem por Importância)
V	Carga vertical aplicada
Var(**X**)	Variância das variáveis aleatórias básicas
X	Vetor das variáveis aleatórias básicas
X'	Vetor aleatório reduzido
Xi	Variável aleatória básica
Z	Vetor das variáveis no espaço reduzido
Z^*	Ponto mais provável de falha
Zi	Variável reduzida
$Z_1,...,Z_4$	Módulos plásticos resistentes
W	Matriz dos pesos

Letras Romanas Minúsculas:

b	Largura
b	Vetor de valores de limiar (bias)
d(**X**)	Deslocamento calculado em função das variáveis aleatórias do vetor **X**
e	Vetor com os erros lineares da rede
f(.)	Função de ativação
$f_{\mathbf{X}}(\mathbf{X})$	Função densidade de probabilidade conjunta das variáveis aleatórias básicas
$f_{\mathbf{Z}}(\mathbf{Z})$	Função densidade de probabilidade conjunta do vetor aleatório reduzido

$g(\mathbf{X})$	Função de desempenho
g_k	Gradiente do vetor de erro
h	Altura
$h_{\mathbf{X}}(\mathbf{X})$	Função de amostragem (método de amostragem por importância)
k_i	Curvaturas principais da superfície de falha no ponto de projeto (método SORM)
$\mathbf{m}$	Vetor com a média das variáveis
q	Carregamento distribuído
$\mathbf{r}$	Vetor de entrada da rede com média 0 e desvio padrão 1
$\mathbf{v}$	Sinal de entrada da rede
$\mathbf{x}$	Vetor de entradas da rede neural
y	Sinal de saída da rede neural
y_e	Sinal de saída exato
w	Carga por unidade de área

Letras Gregas Maiúsculas:

$\Phi(.)$	Função de distribuição acumulada da variável normal padrão
$\Phi^{-1}(.)$	Inverso da função de probabilidade acumulada normal padrão
Σ	Somatório
Γ	Matriz obtida da inversa da matriz triangular inferior da matriz dos coeficientes de correlação
$\nabla G(\mathbf{X})$	Gradiente da função de falha no espaço original

Letras Gregas Minúsculas:

α_i	Componente do vetor normal a superfície de falha

β	Índice de confiabilidade
β_C	Índice de confiabilidade de Cornell
$ß_m$	Fator de momentum
$\phi(.)$	Função densidade de probabilidade da distribuição normal padrão
ϕ_i	Autovetores da matriz Hessiana
s	Matriz diagonal contendo os desvios padrão das variáveis
s	Desvio padrão
s_P	Desvio padrão da carga aplicada
s_R	Desvio padrão da resistência
μ	Valor médio
μ_k	Constante do algoritmo de Levenberg – Marquardt
μ_P	Valor médio da carga aplicada
μ_R	Valor médio da resistência
?	Taxa de aprendizagem
?	Parâmetro de suavidade da região de transição da função sigmoidal
$?_i$	Autovalores da matriz Hessiana
d_{Max}	Deslocamento máximo da viga biapoiada
d_S	Gradiente local da função de erro
$?_{ij}$	Coeficiente de correlação entre as variáveis X_i e X_j
?	Conjunto de treinamento da rede
?	Parâmetro de ajuste da constante μ_k

Lista de Abreviaturas e Siglas:

CDF	Função cumulativa de probabilidade
COV	Coeficiente de variação
FDP	Função densidade de probabilidade
FERUM	Finite Element Reliability Using MatLab

FORM	First Order Reliability Method
FTOOL	Programa de análise de pórticos bidimensionais (Frame TOOL)
GDM	Algoritmo do Gradiente Descendente com “Momentum”
LM	Algoritmo de Levenberg – Marquardt
MatLab	Matrix Laboratory (Software)
MCC	Método de Monte Carlo Clássico
MCCE	Método de Monte Carlo com Conditional Expectation
MCIS	Método de Monte Carlo com Importance Sampling
MLP	MultiLayer Perceptron
Nci	Número de camadas intermediárias
Nnc	Número de neurônios por camada
nsim	Número de simulações
SAP	Programa de análise estrutural não linear
SORM	Second Order Reliability Method

Capítulo 1

INTRODUÇÃO

1.1 CONSIDERAÇÕES INICIAIS

A ocorrência de falhas nas estruturas pode trazer grandes prejuízos de ordem material e principalmente no que diz respeito à segurança humana. O risco de falha deve ser considerado no projeto, na seleção da qualidade dos componentes, no processo de fabricação e durante a vida útil. A falha ocorre quando a função a que se destina parte ou toda a estrutura não pode ser exercida de forma satisfatória.

A incerteza dos parâmetros envolvidos na análise estrutural é conhecida como um fator importante que influencia a segurança estrutural (PULIDO et al., 1992). A existência de incertezas nestes parâmetros contribui para que se tenha uma probabilidade não nula de que a estrutura não atenda aos objetivos para os quais ela fora concebida. Esta probabilidade é definida como probabilidade de falha e pode ser determinada pelos métodos de análise de confiabilidade estrutural.

Os métodos de confiabilidade estrutural têm se desenvolvido significativamente durante os últimos anos e têm sido documentados em um grande número de publicações. Estes constantes avanços na teoria da confiabilidade e na determinação de quantificações mais minuciosas das incertezas associadas a carregamentos e resistências têm estimulado o interesse do tratamento probabilístico das estruturas (PAPADRAKAKIS et al., 1996).

A teoria da confiabilidade é uma ferramenta que proporciona ao engenheiro, a partir do conhecimento das incertezas inerentes às variáveis de projeto, por meio de suas distribuições de probabilidade, a determinação da probabilidade da estrutura falhar e também parâmetros que fornecem a importância de cada variável nesta probabilidade. Estas informações são seguramente de fundamental importância na tomada de decisões que envolvam a segurança da estrutura.

A análise de confiabilidade estrutural em geral, por envolver um grande número de variáveis aleatórias ou exigir uma grande quantidade de simulações, se depara com a questão do custo computacional. Duas técnicas utilizadas para esta avaliação são o método de simulação de Monte Carlo e os métodos analíticos do tipo FORM / SORM.

Os métodos analíticos apresentam algumas restrições em função da complexidade da análise, que geram dificuldades na determinação dos pontos de mínimo da superfície de falha. Em relação ao método de Monte Carlo, embora seja de fácil implementação e absolutamente geral, o grande número de simulações pode exigir um tempo de processamento elevado, o que pode inviabilizar a sua aplicação. Este problema tem sido solucionado através de técnicas de redução de variância tais como Amostragem por Importância e Esperança Condicionada.

Uma outra estratégia para a análise de confiabilidade estrutural seria a aplicação das redes neurais artificiais. Apesar de recente, a utilização de redes neurais tem se difundido em muitas áreas do conhecimento, consequentemente aumentando-se também o interesse pelo assunto. Esta aplicação já se mostrou bastante eficaz na solução de diversos tipos de problemas, em especial na área de engenharia.

Uma rede neural artificial é uma estrutura computacional que procura imitar o comportamento do cérebro humano em desempenhar tarefas particulares, que dependem principalmente da capacidade de aprender. A eficiência da rede em produzir resultados satisfatórios depende de alguns parâmetros escolhidos de forma experimental numérica, que são de extrema importância para a sua performance e podem exercer influência em seu tempo de treinamento e na resposta fornecida por esta.

A abordagem de redes neurais para a solução de problemas em diversas áreas passa por um processo de treinamento e aprendizado, no qual são ajustados os seus parâmetros, através de técnicas de otimização que visam encontrar um ponto de erro mínimo, fazendo com que a resposta fornecida pela rede seja próxima da resposta almejada.

1.2 OBJETIVOS

O objetivo principal deste trabalho é a análise de problemas de confiabilidade estrutural com a utilização de redes neurais artificiais, aplicadas em conjunto com o método de simulação de Monte Carlo.

Como objetivos específicos, o trabalho tem por finalidade:

- Estudar os métodos de análise de confiabilidade estrutural para posterior comparação dos resultados obtidos;
- Avaliar os parâmetros intervenientes no treinamento de uma rede neural e analisar a sua influência na solução de problemas de confiabilidade.

Neste trabalho também será adotada uma outra estratégia, que é a utilização de uma rede neural treinada para substituir a solução do problema estrutural mais a análise de confiabilidade, com o objetivo de reduzir o custo computacional requerido nestas análises em conjunto.

Para tal aplicação, será utilizado o software Matlab 6.5 (DEMUTH e BEALE, 1998), que possui um ambiente de redes neurais já implementado, dispondo de diversas arquiteturas de redes e de algoritmos de treinamento.

Serão utilizados outros programas, tanto para a análise de confiabilidade quanto para análise estrutural como, por exemplo, o programa **FERUM**, desenvolvido por HAUKAAS (2001), da Universidade da Califórnia, em Berkeley, o programa **FTOOL**,

desenvolvido na Pontíficia Universidade Católica do Rio de Janeiro, e o programa **SAP 2000** para a geração do conjunto de dados para o treinamento da rede neural.

1.3 APRESENTAÇÃO

Neste ítem será apresentada de forma objetiva a descrição de cada capítulo e as etapas para a realização do trabalho.

O segundo capítulo trata de uma revisão sobre os métodos de análise de confiabilidade estrutural e alguns aspectos pertinentes a cada um destes na determinação da probabilidade de falha. Os métodos citados neste capítulo são os métodos analíticos FORM e SORM e o método de simulação de Monte Carlo, em conjunto com as técnicas de redução de variância Amostragem por Importância e Esperança Condicionada.

No terceiro capítulo é feita uma explanação sobre os conceitos relacionados às redes neurais artificiais, um histórico de seu surgimento, a comparação com o cérebro humano e como se dá o seu treinamento, as principais arquiteturas e tipos de aprendizado, e principais algoritmos de treinamento e uma descrição dos parâmetros que interferem em seu aprendizado, expondo algumas sugestões para a inicialização destes parâmetros.

No quarto capítulo expõe-se uma metodologia de aplicação das redes neurais artificiais em conjunto com o método de Monte Carlo para utilização em problemas de confiabilidade estrutural, mostrando em substituição a que partes do processo a rede neural pode ser aplicada. Três algoritmos são propostos para esta análise e são descritos os passos de sua aplicação e onde a rede neural está inserida.

O quinto capítulo apresenta as análises dos casos estudados de acordo com os algoritmos propostos no capítulo 4, realizando comparações com dados encontrados na literatura e avaliadas discussões sobre cada um destes casos, mostrando a eficácia das redes para a aplicação em problemas de confiabilidade estrutural, no que diz respeito a

qualidade da resposta fornecida e do tempo computacional necessário para que esta resposta seja alcançada.

No sexto capítulo são apresentadas as conclusões sobre a aplicação objetivada no trabalho, ressaltando as vantagens e desvantagens desta aplicação, bem como sugestões para continuidade de pesquisa com a utilização das redes neurais artificiais.

Capítulo 2

MÉTODOS DE ANÁLISE DE CONFIABILIDADE

2.1 INTRODUÇÃO

O que se busca em um projeto estrutural é o desempenho satisfatório da estrutura ao longo de seu tempo de vida útil, ou seja, que esta não venha a falhar. Este critério de falha pode caracterizar-se pela falha de um componente do sistema ou do sistema como um todo, tornando-se necessário analisar a estrutura local e globalmente.

As incertezas presentes na análise estrutural são inerentes às variáveis envolvidas, por exemplo, dimensões do elemento, propriedades dos materiais, ações (carregamentos) aplicadas. Daí, torna-se conveniente fazer uma análise em termos probabilísticos com o intuito de garantir o funcionamento da estrutura ou elemento estrutural através da determinação da probabilidade de falha.

Em geral, as incertezas podem ser classificadas, de acordo com MACHADO (2000), como:

- Incertezas fenomenológicas, associadas à ocorrência de eventos imprevisíveis, devidas ao desconhecimento de qualquer aspecto de um possível comportamento estrutural sob condições de serviço ou condições extremas;
- Incertezas de avaliação, associadas à definição e à quantificação do desempenho do sistema estrutural, bem como à caracterização dos estados limites;

- Incertezas do modelo, associadas às simplificações e às hipóteses adotadas no modelamento do sistema estrutural, ao emprego de novos materiais, ao uso de técnicas construtivas. Este tipo de incerteza é devida, em geral, à falta de conhecimento, mas pode ser reduzida com pesquisa ou aumento da informação disponível;
- Incertezas estatísticas, associadas à extrapolação dos parâmetros estatísticos extraídos de populações finitas;
- Incertezas devidas a fatores humanos, associadas aos erros humanos ou à intervenção humana no comportamento do sistema estrutural;
- Incertezas físicas, associadas à aleatoriedade inerente às variáveis de projeto. Podem ser reduzidas com aumento dos dados disponíveis, ou em alguns casos, com o controle de qualidade. Entre estes podem citar-se: tensão de escoamento, resistência à compressão, dimensões, cargas, etc.

A segurança estrutural é caracterizada pela probabilidade de falha, que representa a probabilidade da estrutura não atender de forma satisfatória à determinadas condições chamadas de Estados Limites, os quais podem ser classificados em estado limite último, que diz respeito ao colapso de toda ou de parte da estrutura, e de utilização, que diz respeito a situação em que a estrutura não satisfaz às condições de uso, apesar de ainda continuar funcionando ao critério de resistência.

2.2 CONCEITOS BÁSICOS DE CONFIABILIDADE

A representação do estado limite se dá através de uma função chamada de função de performance ou de falha, usualmente chamada de $G(\mathbf{X})$, que envolve as variáveis aleatórias de interesse contidas no vetor $\mathbf{X}$, onde este é representado pela equação 2.1:

$$\mathbf{X}^t = \{X_1, X_2, \ldots, X_n\} \tag{2.1}$$

onde X_i representa as variáveis básicas do problema.

A função de performance pode representar uma grande quantidade de variáveis, cujo vetor possui uma dimensão n, sendo representada pela equação 2.2:

$$G(\mathbf{X}) = G(X_1, X_2, \ldots, X_n) \tag{2.2}$$

A partir da avaliação desta função, pode-se assumir duas configurações possíveis, que representarão os domínios em que estrutura se mostra segura ou insegura, podendo ser ilustrada graficamente pela figura 2.1, denotando o caso bidimensional.

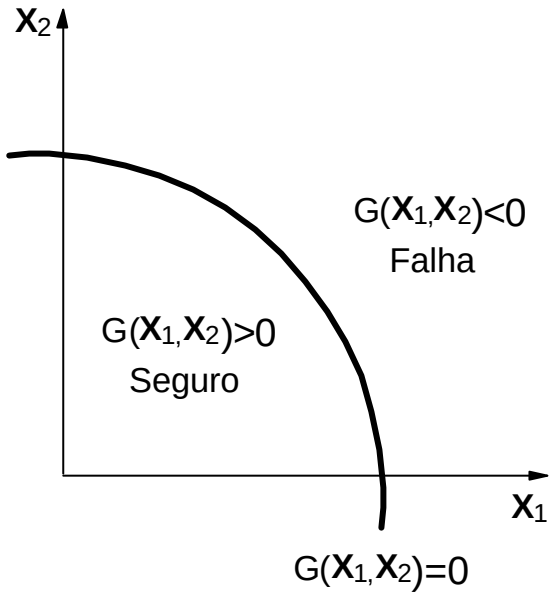

Figura 2.1: Domínios possíveis para a função de performance.

Da figura 2.1, define-se a região de falha como sendo a região onde ocorre $G(\mathbf{X}) \leq 0$, que pode ser representada pela integral da função densidade de probabilidade conjunta $f_{\mathbf{X}}(\mathbf{X})$ sobre a região mostrada acima, definida pela equação 2.3:

$$P_f = \int_{G(\mathbf{X}) \leq 0} \ldots \int f_{\mathbf{X}}(\mathbf{X}) d\mathbf{X} . \tag{2.3}$$

A avaliação da integral contida na equação 2.3 não é muito simples, uma vez que ela envolve a avaliação de uma integral n-dimensional num domínio complexo, e geralmente a função densidade de probabilidade conjunta não é conhecida. Mesmo com a utilização de técnicas modernas de integração numérica e com computadores cada vez

mais eficientes, na prática a avaliação da integral fica restrita a problemas com 5 a 6 variáveis aleatórias.

As dificuldades encontradas na resolução da integral que define a probabilidade de falha levaram a utilização do chamado índice de confiabilidade de segunda ordem ou índice de confiabilidade de Cornell. CORNELL apud SARAIVA (1997), propôs uma medida quantitativa da confiabilidade com base no significado estatístico de E(**X**) e Var(**X**), medidas de posição e dispersão de um conjunto de dados, respectivamente, mostrado na equação 2.4:

$$\beta_C = \frac{E[G(\mathbf{X})]}{\sqrt{Var[G(\mathbf{X})]}}. \tag{2.4}$$

Este índice representa a distância entre o valor médio de G(**X**) e zero em unidades de desvios padrão. O cálculo da média e da variância são aproximados, pois dependem da linearização da função, sendo este índice somente invariável quando a função é linear pelo fato dos parâmetros necessários à sua determinação não poderem ser expressos em termos do primeiro e segundo momento da variável X.

A determinação da probabilidade de falha também pode ser efetuado através das variáveis reduzidas, ou seja, com média igual a 0 e desvio padrão igual a 1.

Um outro conceito importante envolvido na determinação do índice de confiabilidade é o conceito de ponto de projeto ou ponto mais provável de falha, que é o ponto situado sobre a superfície de falha que possui a menor distância à origem. Esta distância corresponde ao índice de confiabilidade.

O índice de confiabilidade pode ser aplicado para um número qualquer de variáveis. Portanto, sendo **X** o vetor de variáveis aleatórias básicas e **Z** o vetor no espaço reduzido correspondente, pode-se expressar a função de performance pela equação 2.5:

$$G(\mathbf{X}) = a_0 + \sum_{i=1}^{n} a_i . Z_i \tag{2.5}$$

onde o índice de confiabilidade é dado pela equação 2.6:

$$\beta = \frac{a_0 + \sum_{i=1}^{n} a_i.\mu_{Zi}}{\sqrt{\sum_{i=1}^{n} a_i^2.\sigma_{Zi}^2}} \tag{2.6}$$

e $a_0, a_1, \ldots, a_n$ constantes.

A equação 2.6 representa a distância do hiperplano até a origem no espaço das variáveis reduzidas. As coordenadas do ponto mais próximo à origem, Z*, no espaço reduzido é dada pela equação 2.7:

$$Z_i^* = -\alpha_i.\beta \tag{2.7}$$

onde α_i é a componente do vetor normal à superfície de falha, no ponto de projeto, definido pela equação 2.8:

$$\alpha_i = \frac{\frac{\partial \mathbf{Z}}{\partial X_i}}{\sqrt{\sum_{i=1}^{n} \left(\frac{\partial \mathbf{Z}}{\partial X_i} \right)^2}} \tag{2.8}$$

onde $\frac{\partial \mathbf{Z}}{\partial X_i}$ denota a componente relacionada a variável X_i do vetor gradiente da função de falha.

No caso das variáveis serem todas normais reduzidas e estatisticamente independentes, o cálculo da média e da variância podem seguir as equações 2.9 e 2.10:

$$E[G(\mathbf{Z})] = a_0 + \sum_{i=1}^{n} a_i.E(Z_i) = a_0 \tag{2.9}$$

$$Var[G(\mathbf{Z})] = \sum_{i=1}^{n} a_i^2.Var(Z_i) = \sum_{i=1}^{n} a_i^2 \tag{2.10}$$

e a probabilidade de falha passa então a ser calculada como mostra a equação 2.11:

$$P_f = \Phi(-\beta) = \Phi\left(\frac{-a_0}{\sqrt{\sum_{i=1}^{n} a_i^2}}\right) \tag{2.11}$$

onde F é a função de distribuição acumulada da variável normal padrão.

2.3 MÉTODO FORM

A idéia do método FORM é obter a confiabilidade através de uma aproximação linear da função de falha, com as variáveis normais estatisticamente independentes no espaço reduzido.

Neste método, as variáveis, que podem ser relacionadas entre si, são transformadas em variáveis normais padrão estatisticamente independentes **Z**. A função de falha pode ser escrita em termos das variáveis **Z** como g(**Z**), e esta superfície é aproximada por uma superfície linear, ou hiperplano, no ponto de projeto do espaço reduzido Z*****, ou seja, no ponto com a menor distância até a origem.

A probabilidade de falha pode ser calculada como mostrado na equação 2.12:

$$P_f = \Phi(-\beta) \tag{2.12}$$

onde Φ representa a distribuição cumulativa normal padrão e β é a distância da origem até o ponto de projeto e pode ser calculado pela equação 2.13:

$$\beta = |\mathbf{Z}^*|. \tag{2.13}$$

De acordo com a equação 2.7, pode-se chegar às equações 2.14 e 2.15:

$$\mathbf{Z}^* = -\mathbf{a}.\beta \tag{2.14}$$

$$g(\mathbf{Z}) = \beta - \sum_{i=1}^{n} \alpha_i . Z_i \tag{2.15}$$

onde **a** é o vetor normal à superfície de falha.

O procedimento descrito pelo método FORM para o cálculo da probabilidade de falha pode ser visualizado na figura 2.2:

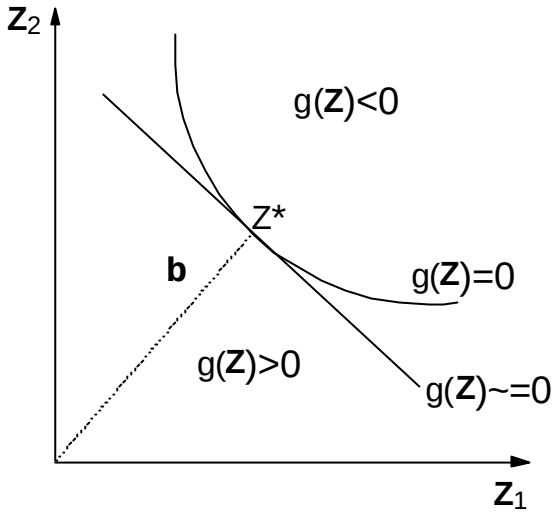

Figura 2.2: Representação gráfica do método FORM.

O método, por ser aproximado, depende diretamente da forma da função de falha, podendo este ser a favor da segurança quando a superfície for convexa nas proximidades do ponto de projeto, e contrário a segurança, quando a mesma for côncava [ANG e TANG, 1984]. Este fato pode ser melhor entendido observando-se a figura 2.3:

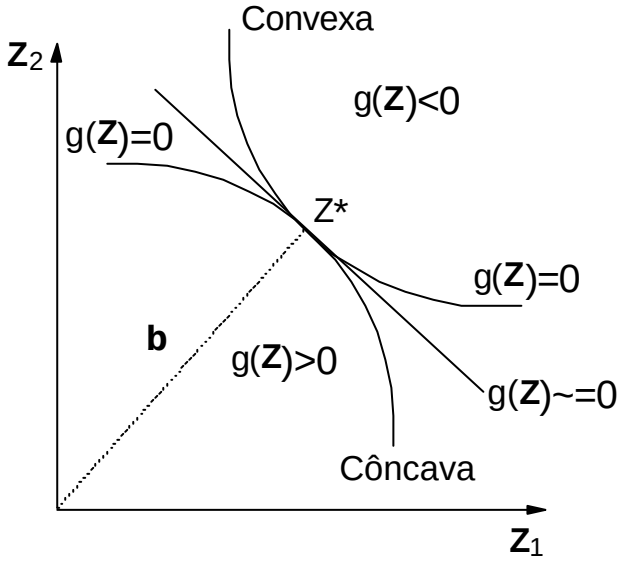

Figura 2.3: Superfícies côncavas e convexas aproximadas pelo método FORM.

Uma outra importante questão relacionada ao método FORM consiste na realização da transformação das variáveis do espaço original para o espaço reduzido, e pode ser resolvida com a utilização da transformada de Rosenblatt, que é definida pela equação 2.16 [ANG e TANG, (1984)]:

$$\begin{aligned} Z_1 &= \Phi^{-1}\left[F_{X_1}(x_1)\right] \\ Z_2 &= \Phi^{-1}\left[F_{X_2}(x_2 / x_1)\right] \\ &\vdots \\ Z_n &= \Phi^{-1}\left[F_{X_n}(x_n / x_1, x_2, \ldots, x_{n-1})\right] \end{aligned} \tag{2.16}$$

onde $F_{X_i}(x_n / x_1, x_2, \ldots, x_{i-1})$ é a função de distribuição acumulada da variável x_i condicionada a $(X_1 = x_1, X_2 = x_2, \ldots, X_{i-1} = x_{i-1})$.

A transformação de Rosenblatt requer o conhecimento da distribuição de probabilidade condicionada, que nem sempre é conhecida. Para suprir esta desvantagem, uma outra possibilidade é a utilização da transformada de Nataf, que faz uso dos parâmetros μ e σ^2 das variáveis, e leva em consideração a correlação entre estas variáveis, sendo definida pela equação 2.17:

$$\mathbf{Z} = [\mathbf{G}].[\mathbf{s}]^{-1}.(\mathbf{X} - \mathbf{m}) \tag{2.17}$$

onde **m** é o vetor com as médias das variáveis X_i, **s** é a matriz diagonal contendo os desvios padrão das variáveis X_i e $\mathbf{G} = \mathbf{L}^{-1}$, onde **L** é a matriz triangular inferior obtida da decomposição de Cholesky da matriz dos coeficientes de correlação. A matriz **L** pode ser expressa pela equação 2.18:

$$\mathbf{L} = \begin{bmatrix} L_{11} & 0 & 0 & 0 \\ L_{12} & L_{22} & 0 & 0 \\ . & . & . & . \\ L_{1n} & L_{2n} & . & L_{nn} \end{bmatrix} \tag{2.18}$$

onde:

$$
\begin{aligned}
L_{11} &= 1.0 \\
L_{i1} &= ?_{i1} \qquad i = 1, n \\
L_{ik} &= \frac{1}{L_{kk}}\left(r_{ik} - \sum_{j=1}^{k-1} L_{ij} L_{kj} \right) \qquad 1 < k < i \\
L_{ii} &= \sqrt{1 - \sum_{j=1}^{i-1} L_{ij}^2} \qquad i > 1
\end{aligned}
\tag{2.18}
$$

onde ρ_{ij} é o coeficiente de correlação entre X_i e X_j.

Na determinação do ponto de projeto é necessário a definição do Jacobiano da transformação, que pode ser definido como descrito na equação 2.19:

$$\mathbf{J} = \frac{\partial \mathbf{Z}}{\partial \mathbf{X}} \tag{2.19}$$

e a partir da equação 2.17, a equação 2.19 pode ser redefinida como a equação 2.20:

$$\mathbf{J} = \mathbf{G}.\mathbf{s}^{-1}. \tag{2.20}$$

Neste método é necessário transformar as variáveis em normais equivalentes, caso estas não sejam normais, e no caso destas variáveis serem correlacionadas, é possível usar a mesma transformação com os coeficientes de correlação entre as variáveis originais corrigidos. O coeficiente de correlação equivalente pode ser determinado pela equação 2.21:

$$\rho_{ij}^{E} = F.\rho_{ij} \tag{2.21}$$

onde F é um valor que depende somente de ρ_{ij} e dos coeficientes de variação das variáveis X_i e X_j.

Procedidas as trasformações e as respectivas correlações equivalentes, a equação 2.17 pode ser empregada para obter as variáveis normais padrão estatisticamente independentes.

Uma etapa fundamental do método FORM é o de encontrar o ponto de projeto, que pode ser formulada como um problema de otimização. Existem vários algoritmos

utilizados para resolver este problema, podendo-se citar como exemplo o desenvolvido por Hasofer e Lind, e aprimorado por Rackwitz e Fiessler, denominado por HLRF, que pode ser descrito pela equação 2.22:

$$\mathbf{Z}^{k+1} = \frac{1}{\left|\nabla g\left(\mathbf{Z}^{k}\right)\right|^{2}}\left[\nabla g\left(\mathbf{Z}^{k}\right)^{T}.\mathbf{Z}^{k} - g\left(\mathbf{Z}^{k}\right)\right].\nabla g\left(\mathbf{Z}^{k}\right)^{T}. \tag{2.22}$$

Torna-se necessário, para a utilização do algoritmo HLRF, o conhecimento da equação 2.23:

$$\begin{aligned} &g(\mathbf{Z}) = G(\mathbf{X}) \\ &\mathbf{Z} = \mathbf{G_s}^{-1}.(\mathbf{X} - \mathbf{m}) \\ &\nabla g(\mathbf{Z}) = \left(\mathbf{J}^{-1}\right)^{T}.\nabla G(\mathbf{X}) \end{aligned} \tag{2.23}$$

onde $\nabla G(\mathbf{X})$ é o gradiente da função de falha no espaço original avaliado no ponto X.

Algumas limitações podem ser encontradas no método FORM, das quais podem ser citadas:

- Se a função de falha não for explícita, o gradiente desta função será avaliado de forma aproximada;
- Superfícies de falha altamente não lineares poderão ocasionar vários pontos de mínimo, levando a dificuldades na avaliação do ponto de projeto.

2.4 MÉTODO SORM

O fundamento deste método é semelhante a do método FORM. A principal diferença está na aproximação da superfície de falha no espaço reduzido, feita no FORM por uma aproximação linear. No método SORM a aproximação é feita por uma superfície quadrática, como mostrado na figura 2.4:

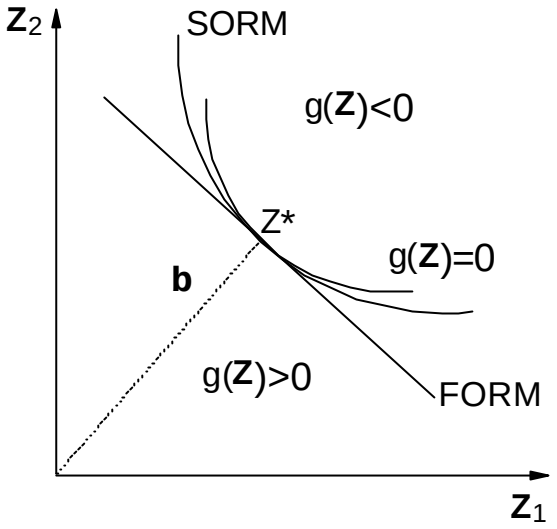

Figura 2.4: Representação gráfica do método SORM.

A probabilidade de falha pode ser determinada pela fórmula de Breitung, descrita na equação 2.24:

$$P_f = \Phi(-\beta)\prod_{i=1}^{n-1}(1+\beta ki)^{-1/2} \qquad (2.24)$$

onde ßki são as curvaturas principais da superfície de falha no ponto de projeto e n o número de variáveis aleatórias envolvidas.

Os procedimentos para a determinação de ßki envolvem derivadas de segunda ordem da função de falha no ponto de projeto, o que gera um aumento no tempo computacional.

Os passos para a determinação da confiabilidade pelo método SORM são os mesmos do método FORM, exceto no cálculo da probabilidade de falha, que utilizam as expressões 2.24 e 2.12, respectivamente.

2.5 MÉTODO DE MONTE CARLO

O método de Monte Carlo é um método de amostragem artificial utilizado na solução de experimentos aleatórios onde se tem conhecimento das distribuições de

probabilidade das variáveis envolvidas. Este tem sido utilizado para determinar a confiabilidade de sistemas estruturais (PULIDO et al., 1992).

Neste são geradas N amostras independentes do vetor das variáveis aleatórias **X** obtidas a partir da função densidade de probabilidade conjunta $f_{\mathbf{X}}(\mathbf{X})$.

A probabilidade de falha pode ser expressa, utilizando o método de Monte Carlo, partindo da integral que a define mostrada na equação 2.25:

$$P_f = \int_{G_x(\mathbf{X})\leq 0} \int \ldots \int f_{\mathbf{X}}(\mathbf{X}) d\mathbf{X} \tag{2.25}$$

ou pela equação 2.26:

$$P_f = \int_{\forall \mathbf{X}} \int \ldots \int I[G_{\mathbf{X}}(\mathbf{X}) \leq 0] f_{\mathbf{X}}(\mathbf{X}) d\mathbf{X} \tag{2.26}$$

onde a função I[.] é um indicador que corresponde aos valores apresentados na equação 2.27:

$$I[.] = \begin{cases} 0 \text{ se } G_{\mathbf{X}}(\mathbf{X}) > 0 \\ 1 \text{ se } G_{\mathbf{X}}(\mathbf{X}) \leq 0 \end{cases} \tag{2.27}$$

onde $G_X(\mathbf{X})$ representa a função de performance.

Pode-se reescrever a equação 2.26 como a equação 2.28:

$$P_f = \sum_{i=1}^{N} I\left[G_{\mathbf{X}}\left(\mathbf{X}^i\right) \leq 0\right] . \frac{1}{N} \tag{2.28}$$

onde X^i representa a ıésima amostra do vetor das variáveis **X** geradas a partir da função densidade de probabilidade $f_{\mathbf{X}}(\mathbf{X})$. A probabilidade de falha passa a ser determinada pela equação 2.29:

$$P_f = \frac{\text{Nº de Simulações em que } G_{\mathbf{X}}(\mathbf{X}) \leq 0}{N} . \tag{2.29}$$

A variância e o coeficiente de variação para valores pequenos da probabilidade de falha são expressos pelas equações 2.30 e 2.31:

$$\mathrm{Var}\,(P_f) = P_f \cdot \frac{(1 - P_f)}{N} \cong \frac{P_f}{N} \tag{2.30}$$

$$\mathrm{COV}(P_f) = \frac{\sqrt{\mathrm{Var}\,(P_f)}}{P_f}\,. \tag{2.31}$$

Devido ao valor de P_f ser pequeno para estruturas usuais, geralmente da ordem de 10^{-3} a 10^{-5}, e como a sua variância é expressa de forma inversamente proporcional ao número total de simulações, o valor de N deve ser elevado para que se possa obter aproximações aceitáveis de P_f. Por este motivo o método de Monte Carlo é frequentemente utilizado para checar outras técnicas aproximadas (PULIDO et al., 1992).

Para suprir tais dificuldades como o número de simulações que requer o método de Monte Carlo, apesar deste ser bastante abrangente, foram desenvolvidas as técnicas de redução de variância, com o objetivo de reduzir este número de simulações.

2.6 TÉCNICAS DE REDUÇÃO DE VARIÂNCIA

2.6.1 AMOSTRAGEM POR IMPORTÂNCIA (IMPORTANCE SAMPLING)

A técnica de amostragem por importância tem sido uma das aproximações mais usadas no contexto de métodos baseados em simulação para a estimação da confiabilidade estrutural (SCHUËLLER et al., 2003). É geralmente reconhecida como a técnica de redução de variância mais eficiente (PAPADRAKAKIS e LAGAROS, 2002).

Trata-se de uma forma de aumentar o número de simulações bem sucedidas, modificando o vetor das variáveis aleatórias. Para tal, utiliza-se outra função densidade de probabilidade $h_{\mathbf{X}}(\mathbf{X})$, que aumente a probabilidade de conter amostras dentro da região de falha, conforme pode ser ilustrado na figura 2.5 para o caso de duas variáveis aleatórias.

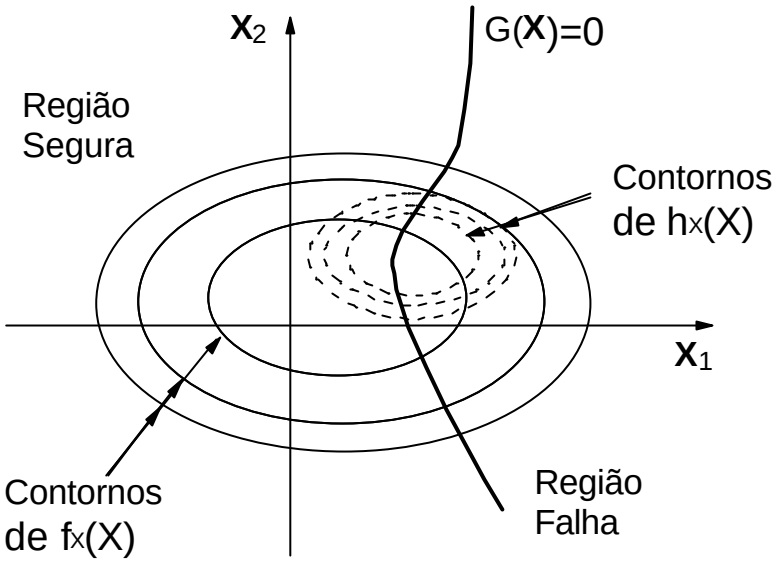

Figura 2.5: Função de amostragem por importância
para o caso de duas variáveis.

Partindo-se da equação 2.26, aplica-se a nova função densidade de probabilidade $h_{\mathbf{X}}(\mathbf{X})$, obtendo-se a equação 2.32:

$$P_f = \int_{\forall \mathbf{X}} \int \ldots \int I[G_{\mathbf{X}}(\mathbf{X}) \leq 0] \frac{f_{\mathbf{X}}(\mathbf{X})}{h_{\mathbf{X}}(\mathbf{X})} h_{\mathbf{X}}(\mathbf{X}) d\mathbf{X}. \quad (2.32)$$

A integral mostrada na equação 2.32 passa a assumir o valor esperado de I[.], descrito na equação 2.33:

$$P_f = E(I[G_{\mathbf{X}}(\mathbf{X}) \leq 0]) \frac{f_{\mathbf{X}}(\mathbf{X})}{h_{\mathbf{X}}(\mathbf{X})}. \quad (2.33)$$

Nota-se que $h_{\mathbf{X}}(\mathbf{X})$ deve ser maior que 0 se $I[G_{\mathbf{X}}(\mathbf{X}) \leq 0] f_{\mathbf{X}}(\mathbf{X}) > 0$. Analogamente, pode-se escrever a equação 2.34:

$$P_f = \sum_{i=1}^{N} I\left[G_{\mathbf{X}}\left(\mathbf{X}^i\right) \leq 0\right] \frac{f_{\mathbf{X}}\left(\mathbf{X}^i\right)}{h_{\mathbf{X}}\left(\mathbf{X}^i\right)} \cdot \frac{1}{N}. \quad (2.34)$$

Uma vez definida $h_{\mathbf{X}}(\mathbf{X})$, são geradas N amostras desta e calculadas para cada amostra X^i o valor da função de falha $G_{\mathbf{X}}(\mathbf{X})$.

A probabilidade de falha é então obtida utilizando-se a equação 2.35:

$$P_f = \sum_{i=1}^{N} \frac{S_i}{N} \tag{2.35}$$

onde S_i corresponde ao fator de escala definido pela equação 2.36:

$$S_i = \begin{cases} 0, \text{se } G_{\mathbf{X}}(\mathbf{X}) > 0 \\ \dfrac{f_{\mathbf{X}}(\mathbf{X}^i)}{h_{\mathbf{X}}(\mathbf{X}^i)}, \text{se } G_{\mathbf{X}}(\mathbf{X}) \le 0 \end{cases} \tag{2.36}$$

e a variância da probabilidade de falha é dada pela equação 2.37:

$$\text{Var}(P_f) = \frac{1}{N.(N-1)} \sum_{i=1}^{N} (S_i - P_f)^2 \tag{2.37}$$

e o coeficiente de variação é determinado pela equação 2.31.

Para a aplicação desta técnica é importante uma boa escolha da função $h_{\mathbf{X}}(\mathbf{X})$ e de sua posição no espaço amostra. No caso de uma escolha inadequada, esta pode levar a resultados errôneos da probabilidade de falha.

MARTINS apud SARAIVA (1997) sugere a escolha da função $h_{\mathbf{X}}(\mathbf{X})$ tendo como base os seguintes parâmetros:

- Tipo de variável aleatória X_i (carregamento ou resistência);
- Forma da função densidade de probabilidade $f_{Xi}(xi)$;
- Estimativa do ponto onde $f_{Xi}(xi)$ atinge um máximo.

2.6.2 ESPERANÇA CONDICIONADA (CONDICIONAL EXPECTATION)

Segundo ANG e TANG (1984), fundamenta-se esta técnica na equação 2.38, válida para duas variáveis quaisquer:

$$\text{Var}_y\left[\text{E}\left(x/y\right)\right] = \text{Var}(x) - \text{E}_y\left[\text{Var}\left(x/y\right)\right] \le \text{Var}(x) \tag{2.38}$$

O objetivo desta técnica é reduzir o espaço amostral das variáveis para a obtenção da probabilidade de falha, caracterizando o problema de forma condicional.

Considerando X a variável de maior dispersão (equação 2.38), ou seja, que contribui com maior significância para a probabilidade de falha, e que seja estatisticamente independente das outras, a ordem de integração da integral que define a probabilidade de falha pode ser reduzida em uma unidade, caso esta variável não seja estatisticamente dependente das outras. Pode-se expressar a variável X em função das demais através da igualdade $G(\mathbf{X}) = 0$, quando esta possui somente uma raiz, obtendo um vetor aleatório **X'** reduzido, com função densidade de probabilidade conjunta $f_{\mathbf{X}'}(\mathbf{X}')$.

A integral passa a ser agora descrita pela equação 2.39:

$$P_f = \int_{G(\mathbf{x}')\leq 0} \cdots \int F_{x_m}(X_m).f_{\mathbf{X}'}(\mathbf{X}')d\mathbf{X}' \tag{2.39}$$

sendo F_{x_m} a função de distribuição acumulada da variável X_m.

Com a aplicação do método de Monte Carlo, a integral acima pode ser escrita como mostrado na equação 2.40:

$$P_f = \frac{1}{N}\sum_{i=1}^{N} F_{x_m}(X_{mi}) \tag{2.40}$$

A variância para esta técnica é dada pela equação 2.41:

$$\mathrm{Var}(P_f) = \frac{1}{N(N-1)}\sum_{i=1}^{N}(P_{fi} - P_f)^2 \tag{2.41}$$

onde P_{fi} é determinado pela equação 2.42:

$$P_{fi} = F_{x_m}(X_{mi}, i = 1,2,...,n) \tag{2.42}$$

e o coeficiente de variação dado pela equação 2.31.

A utilização desta técnica tem como principais vantagens, além da redução da ordem da integral que define a falha, a redução do número de simulações e todas as simulações efetuadas geram informações sobre a probabilidade de falha.

Capítulo 3

REDES NEURAIS ARTIFICIAIS

3.1 INTRODUÇÃO

Com o processo evolutivo das máquinas, um grande objetivo do homem é a construção de uma máquina que possa executar tarefas sem depender do controle humano, que tenha a capacidade de aprender e de interagir com ambientes que lhe são desconhecidos, podendo ser denominada de inteligente. Estas teriam uma grande habilidade de aprendizado de afazeres não facilmente manipulados pelas máquinas atuais, e continuariam a se adaptar e a realizar tarefas com maior eficiência.

Não há dúvidas de que um computador possui uma capacidade muito maior que a do cérebro humano para a realização de certas tarefas como, por exemplo, cálculos algébricos. No entanto, uma pessoa possui uma facilidade muito grande em certas tarefas, como a comunicação verbal, enquanto que os computadores atuais são praticamente incapazes de realizá-las. Essas diferenças têm origem no tipo de abordagem de cada uma dessas estruturas: tarefas cuja solução pode ser obtida diretamente por meio de uma seqüência lógica de processos tendem a ser melhor desempenhadas por computadores, cuja velocidade de processamento é extremamente superior à do cérebro. Por outro lado, tarefas cujas soluções não podem ser obtidas facilmente por um algoritmo, mas sim por experiência adquirida, tendem a ser melhor desempenhadas pelo cérebro humano (DEVRAIGNES e NETO, 2003).

Redes neurais artificiais podem ser caracterizadas como modelos computacionais baseados em processamento distribuído paralelo com propriedades

particulares como habilidade para aprender, generalizar, classificar e organizar dados (GOMES e AWRUCH, 2004).

Existem vários tipos de redes e algoritmos de treinamento que podem ser aplicados em várias áreas da engenharia, e em especial, na área da confiabilidade estrutural, por se tratar de uma utilização recente e que tem fornecido bons resultados.

3.2 HISTÓRICO

O primeiro modelo artificial de um neurônio biológico data de 1943, em trabalho de Warren McCulloch e Walter Pitts, em que sugeriam a construção de uma máquina baseada no cérebro humano, entitulado de "A Logical Calculus of the Ideas Immament in Nervous Activity" [BRAGA el al,2002; OSORIO, 2000]. Este trabalho foi concentrado na descrição do modelo artificial de um neurônio e apresentação de suas capacidades e não na apresentação de técnicas de aprendizado.

Em 1949, Donald Hebb escreveu um livro entitulado "The Organization of Behavior" (A Organização do Comportamento) que influenciou vários modelos de redes em destaque na atualidade. Hebb perseguia a idéia de que o condicionamento psicológico clássico está presente em qualquer parte dos animais pelo fato de que esta é uma propriedade dos neurônios individuais. Ele propôs uma lei de aprendizado específica para as sinapses dos neurônios, conhecida como aprendizado Hebbiano.

Frank Rosenblatt, em 1958, criou o Perceptron, que deu origem posterior aos modelos MLP (Multilayer Perceptron). Modelos semelhantes ao Perceptron foram também desenvolvidos, como o ADALINE (Adaptive Linear Element), criado por Bernard Widrow em 1962, que diferente do Perceptron ainda permanece em uso. Estes modelos foram baseados no aprendizado supervisionado por correção de erros, uma classe que tem muita aplicação na atualidade.

Em 1969, Minsky e Papert chamaram a atenção para algumas tarefas que o Perceptron não executara, já que este só resolvia problemas linearmente separáveis, ou

seja, problemas cuja solução pode ser obtida dividindo-se o espaço de entrada em duas regiões através de uma reta [BRAGA et al, 2000]. Minsky e Papert provaram que estes modelos não eram capazes de aprender uma simples função lógica do tipo ou-exclusivo (XOR – Exclusive Or). A função XOR possui um padrão de valores de entrada e de saída cuja associação não podia ser aprendida pelo Perceptron. O impacto da publicação abalou as pesquisas realizadas nesta área.

Houve um período, entre 1969 e 1982, em que não se realizaram mais pesquisas, em grande parte devido ao trabalho de Minsky. Somente em 1982, quando o físico John Hopfield com suas publicações deu um novo impulso na área, no qual chamava a atenção para as propriedades associativas das redes neurais. O modelo criado por Hopfield era baseado em conexões recorrentes e com um comportamento baseado na competição entre os neurônios, onde o aprendizado não era supervisionado.

Outro modelo surgido na década de 80 foi o das redes multicamada, baseada em Perceptrons, onde o algoritmo de aprendizado utilizado neste modelo, o Backpropagation, resolveu parte dos problemas existentes no momento. Este modelo foi desenvolvido por Rumelhart e Hinton, que o tornaram famoso no trabalho “Parallel Distributed Processing”.

O algoritmo Backpropagation permitia que se realizasse o aprendizado por correção de erros em uma rede com múltiplas camadas e consequentemente resolvia o problema do XOR [OSORIO, 2000].

O modelo de Kohonen, outro surgido nesta época, permitia o aprendizado competitivo com auto-organização da rede, criando os chamados “self-organizing feature maps” (Mapas de atributos auto-organizáveis).

3.3 ESTRUTURA DE UMA REDE NEURAL

O cérebro humano é considerado um grande processador, estimando-se que este possua algo em torno de 10 bilhões de neurônios ligados através de conexões sinápticas, formando uma rede extensa e muito complexa.

Segundo SARAIVA(1997), apesar de extremamente complexo, e boa parte de sua estrutura biológica desconhecida pelos cientistas, é possível representar um componente básico do sistema nervoso, o neurônio, através do uso de um modelo matemático simples.

Os principais componentes dos neurônios, como pode ser observado na figura 3.1, são:

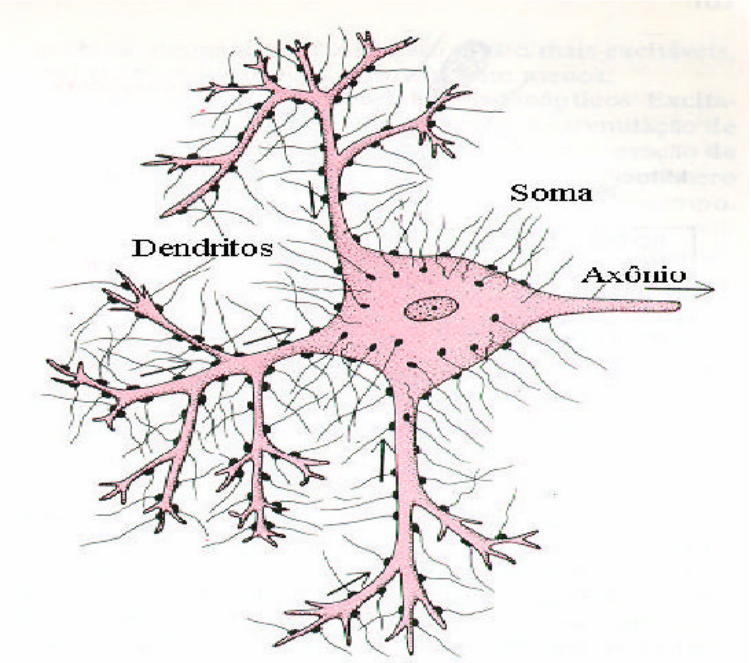

Figura 3.1: Componentes do neurônio biológico.

- Os dendritos, que são extensões filamentares que tem por função receber estímulos transmitidos pelos outros neurônios;
- O corpo do neurônio, também chamado de soma, que é responsável por coletar e combinar informações vindas de outros neurônios;

- O axônio, que é constituído de uma fibra tubular que pode atingir alguns metros e é responsável por transmitir os estímulos para outros neurônios que com ele se relacionam.

Nos neurônios a comunicação é realizada através de impulsos. Quando um impulso é recebido o neurônio o processa, e passado um limite de ação, dispara um segundo impulso que produz uma substância neurotransmissora a qual flui do corpo celular para o axônio, que pode ou não estar conectado a um dendrito de outra célula. Eles têm um papel essencial no funcionamento, comportamento e raciocínio dos seres humanos.

Dependendo do tipo de neurotransmissor, a conexão sináptica poderá ser inibitória ou excitatória. Para o caso excitatório, há uma alteração no potencial eletroquímico da membrana que envolve o corpo do neurônio, ocasionando a formação de um impulso nervoso no axônio de saída, o que não ocorre no caso inibitório.

SARAIVA(1997) ressalta que a formação deste potencial da ação da membrana, diretamente ligada a propagação ou não do sinal ao longo da fibra, depende do fato deste impulso ser de tal magnitude a ponto de conseguir despolarizar a membrana e permitir a ultrapassagem de um determinado valor conhecido como limiar (threshold) de disparo, que é um dos conceitos básicos na modelagem do neurônio artificial.

Desta forma, pode-se caracterizar uma rede neural como sendo uma estrutura processadora de informações através de unidades processadoras, os neurônios, os quais são interconectados por meio de ligações sinápticas, que podem enviar/receber sinais para/de outros neurônios.

Uma rede neural artificial se baseia neste comportamento, tendo como principal elemento o neurônio. Esta pode ser particionada em camadas de entrada, saída e intermediárias ou ocultas, como mostrado na figura 3.2. O número de neurônios das camadas de entrada e saída são dados em função do tamanho dos vetores de entrada e

saída, respectivamente, enquanto que para a camada oculta este é determinado através de experimentos númericos.

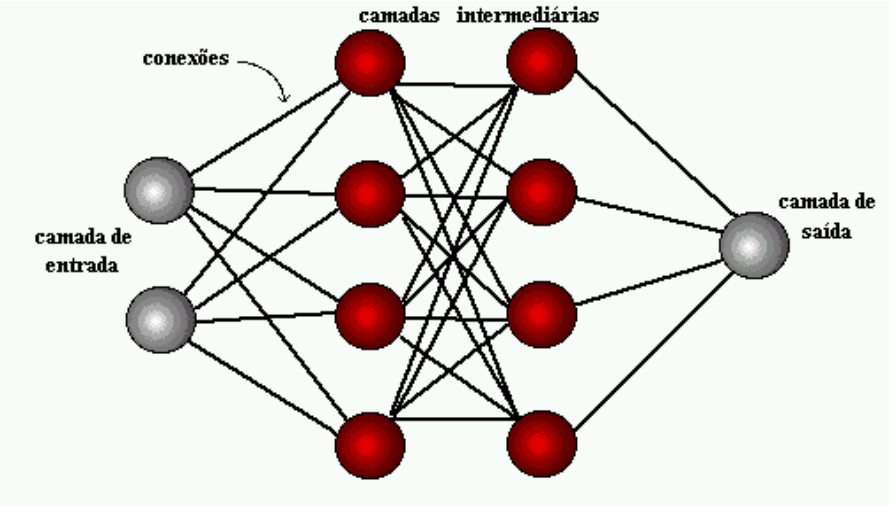

Figura 3.2: Camadas de uma rede neural artificial.

De acordo com NASCIMENTO JR. e YONEYAMA(2002), uma rede neural artificial pode ser dividida segundo as seguintes características principais:

- Um conjunto de neurônios;
- Um estado de ativação para cada unidade;
- Uma função de saída para cada unidade;
- Um padrão de conectividade entre as unidades, o qual é definido pela matriz dos pesos **W**;
- Uma regra de combinação usada para propagar os estados de ativação dos neurônios pela rede;
- Um ambiente externo que fornece informação para a rede e/ou interage com ela;
- Uma regra de aprendizado usada para modificar o padrão de conectividade da rede, usando informação fornecida pelo ambiente externo, ou seja, para modificar a matriz dos pesos **W**.

Na formação de uma rede neural, três etapas devem ser destacadas:

- A determinação da forma em que as camadas estão interligadas e como estas enviarão ou receberão sinais, o qual pode ser denominado de arquitetura de rede. Esta determinação relaciona parâmetros como: número de camadas, número de

neurônios em cada camada e regra de propagação de sinal. Geralmente, uma rede bastante simples pode ser constituídas de uma ou duas camadas intermediárias;

- A determinação da função de ativação, que define como, do neurônio, será avaliado o sinal de saída;
- A determinação da regra de aprendizado, que irá descrever como a rede aprenderá a reconhecer um determinado padrão ou sinal.

Uma importante diferença, que pode ser mencionada entre o cérebro e o computador, é mostrada na tabela 3.1. Enquanto o cérebro pode analisar grande quantidade de informações paralelamente, o computador não as pode, apesar da velocidade que este possui para processar informações.

Tabela 3.1: Comparação cérebro versus computador.

	Computador	**Cérebro**
Velocidade	Nanosegundo	Milisegundo
Tipo de Processamento	Seqüencial	Paralelo
Número de Unidades de Armazenamento	$10e^{9}$ bits	$10e^{14}$ sinapses
Número de Unidades de Processamento	± 1024	$10e^{11}$

3.4 CARACTERÍSTICAS DAS REDES NEURAIS

A capacidade de uma rede neural depende principalmente de dois fatores: sua estrutura paralela distribuída e sua habilidade de aprender e, como consequência, generalizar.

Algumas das características importantes das redes neurais são:

- Tolerância a falhas, que permite que a rede continue a apresentar resultados aceitáveis no caso de falha de algum neurônio. A informação contida na rede

está distribuida por todos os seus elementos, possibilitando que, mesmo que parte da rede seja destruída, a informação esteja contida nos elementos restantes e possa ser recuperada;

- Generalização, que possibilita à rede obter saídas adequadas como resposta a dados de entrada desconhecidos, ou seja, não pertencentes ao conjunto de treinamento;
- Capacidade de aprendizagem, processo que envolve a modificação dos pesos sinápticos de uma rede através da aplicação de um conjunto de pares de treinamento, para os quais se conhece previamente a saída que se deseja obter. O treinamento é repetido até que a rede atinja um nível em que não haja mudanças significativas nos pesos;
- Habilidade de aproximação, dada a capacidade de aprendizado a rede tem a possibilidade de encontrar qualquer mapeamento entrada/saída, desde que os dados sejam representativos do processo do que se esteja tratando e que sejam adequadamente escolhidos a arquitetura de rede e o seu algoritmo de treinamento, sendo as redes capazes de aproximar funções contínuas de ordem qualquer.

3.5 FUNÇÕES DE ATIVAÇÃO

O primeiro modelo de neurônio artificial foi desenvolvido por McCulloch e Pitts e representava uma simplificação do que era conhecido na época sobre o funcionamento do neurônio biológico.

Este neurônio pode ser observado na figura 3.3 e descrito pela seguinte formulação matemática mostrada na equação 3.1:

$$y = f\left(\sum_{i=1}^{n} W_i x_i - b \right) = f\left(\mathbf{W}^t \mathbf{x} - b\right) \tag{3.1}$$

onde: $\mathbf{x}$ = vetor com n sinais de entrada;

$\mathbf{W}^t$ = matriz com os pesos de cada neurônio acoplado;

b = valor do limiar (threshold), também conhecido como bias;

f(.) = função de ativação;

y = sinal de saída.

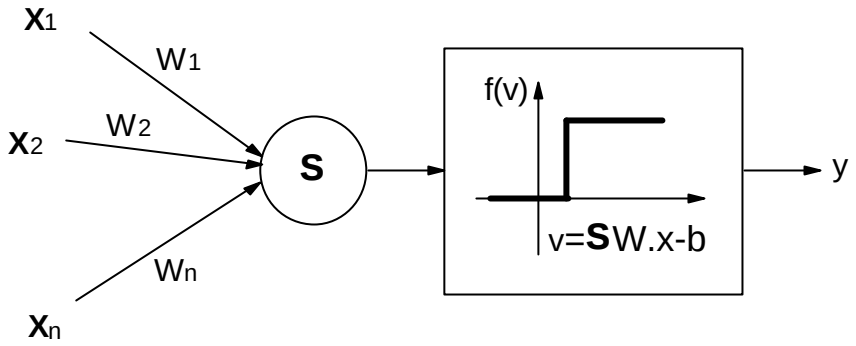

Figura 3.3: Neurônio artificial de McCuloch e Pitts.

Este modelo terá sua saída ativada quando a condição descrita na equação 3.2 acontecer:

$$\sum_{i=1}^{n} W_i x_i \geq b \tag{3.2}$$

A função de ativação é a função que define a saída do neurônio em termos do potencial de ativação. Inicialmente, McCulloch e Pitts propuseram uma função de ativação do tipo degrau unitário, no qual é válida a equação 3.3:

$$\begin{aligned} \mathbf{W}^t\mathbf{x} - b > 0 &\rightarrow y = 1 \\ \mathbf{W}^t\mathbf{x} - b \leq 0 &\rightarrow y = 0 \end{aligned} \tag{3.3}$$

A partir deste modelo, outras funções de ativação foram desenvolvidas e permitem uma saída qualquer. As funções de ativação mais empregadas são: linear, rampa, degrau, degrau unitário, tangente hiperbólica e sigmoidal.

Uma descição detalhada destas funções de ativação podem ser vistas nos itens subsequentes.

3.5.1 FUNÇÃO LINEAR

A função de ativação linear, cujo gráfico pode ser observado na figura 3.4, possui a seguinte relação matemática mostrada na equação 3.4:

$$f(v) = v \tag{3.4}$$

onde $v = \mathbf{W}^t\mathbf{x} - b$.

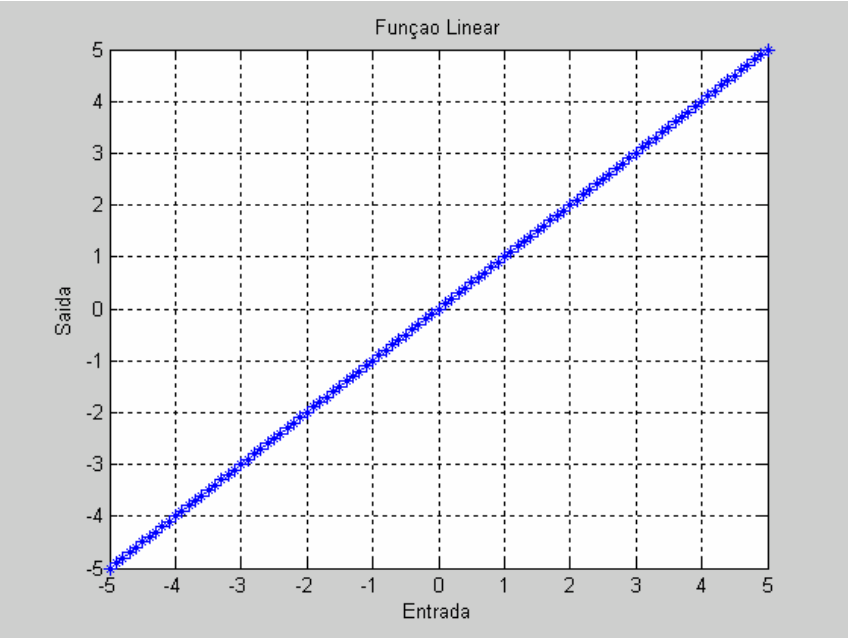

Figura 3.4: Função linear.

Este tipo de função é usada nos modelos ADALINE e na generalização multidimensional MADALINE desenvolvida por Widrow.

3.5.2 FUNÇÃO RAMPA

É um caso semelhante à função linear, com os valores de saída delimitados por um intervalo $[-1,+1]$, conforme mostrado na figura 3.5 e descrito na equação 3.5:

$$f(v) = \begin{cases} +1, \text{se } v \geq +1 \\ v, \text{se} -1 < v < +1 \\ -1, \text{se } v \leq -1 \end{cases} \tag{3.5}$$

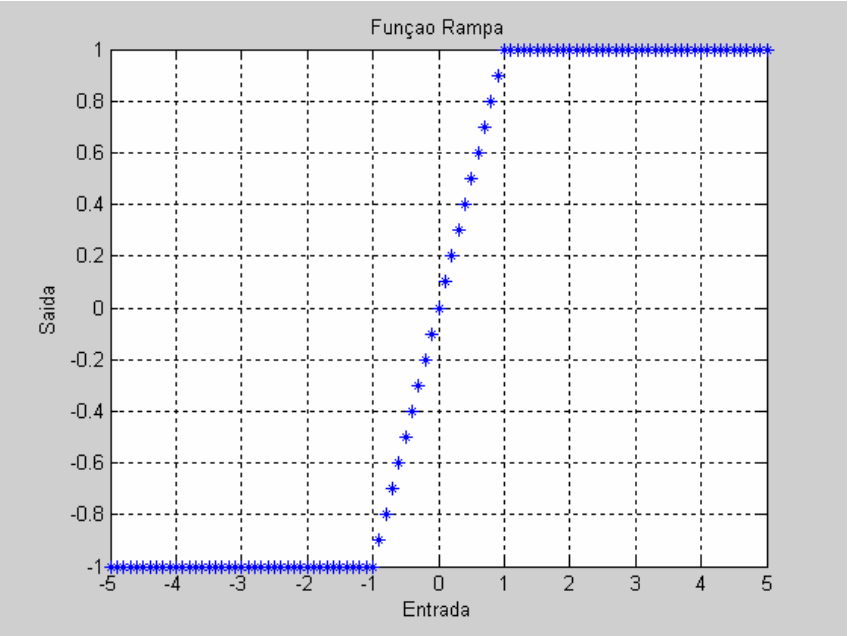

Figura 3.5: Função rampa.

3.5.3 FUNÇÃO DEGRAU

Esta função pode ser descrita pela equação 3.6 e graficamente pela figura 3.6:

$$f(v) = \begin{cases} +1, \text{se } v > 0 \\ -1, \text{se } v \leq 0 \end{cases}. \tag{3.6}$$

A função degrau unitário, que é uma particularização da função degrau, possui os limites superior e inferior iguais a 1 e 0, respectivamente, e foi utilizada no neurônio proposto por McCulloch e Pitts, que ficou conhecido como neurônio binário, pois representava a presença ou não de determinadas características nos dados. Esta função pode ser esboçada graficamente como mostrado na figura 3.7:

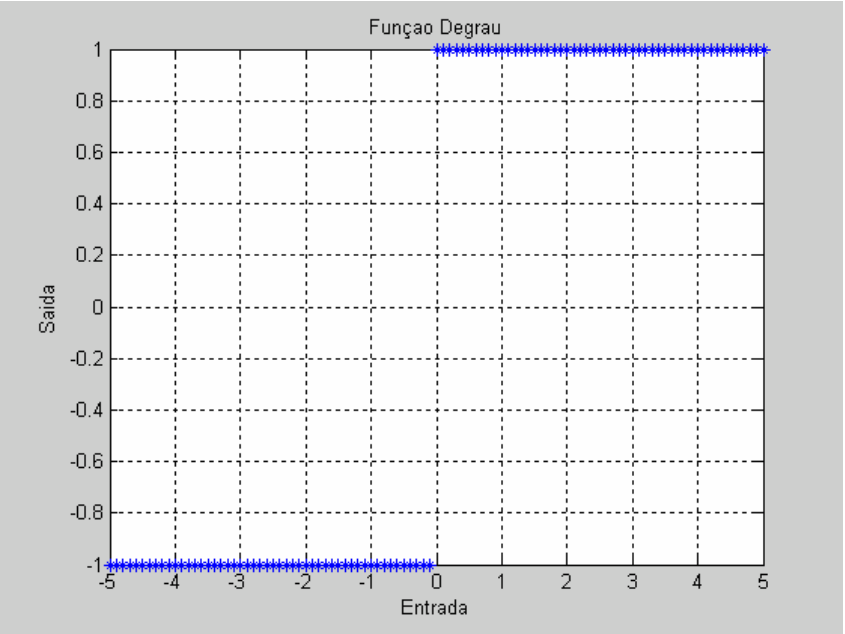

Figura 3.6: Função degrau.

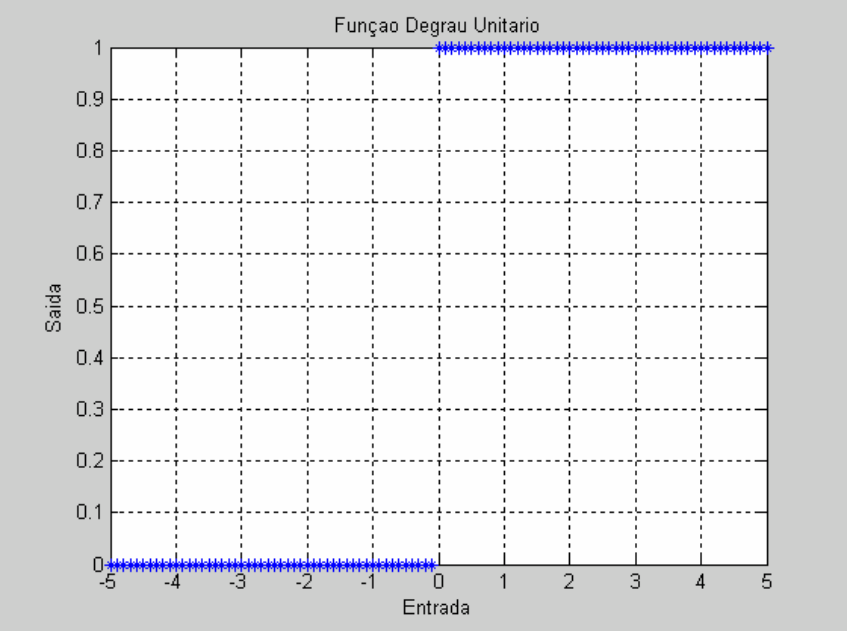

Figura 3.7: Função degrau unitário.

3.5.4 FUNÇÃO SIGMOIDAL

Esta função, também conhecida como S-shape e função logística, ilustrada na figura 3.8, é uma função que apresenta características como a de ser monotônica crescente, ou seja, seu valor cresce a medida que se aumenta o valor de sua entrada, a qual pode ser expressa pela equação 3.7:

$$f(v) = [1 + \exp(-\lambda.v)]^{-1} \tag{3.7}$$

com derivada expressa pela equação 3.8:

$$\frac{df(v)}{dv} = \lambda[1 - f(v)].f(v) \tag{3.8}$$

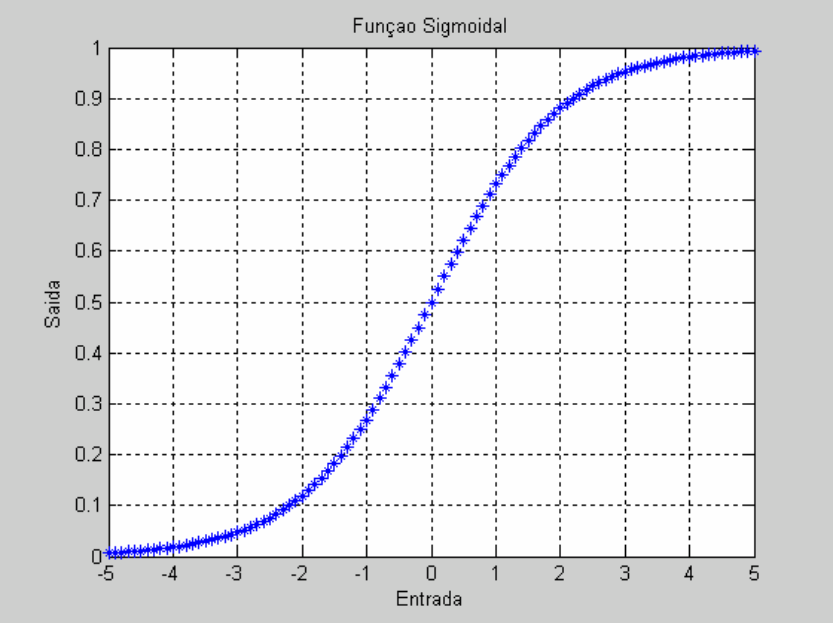

Figura 3.8: Função sigmoidal.

A função sigmoidal tem que possuir limites de valores superior e inferior para a sua saída. Para o caso da função logística, esta apresenta limites no intervalo [0;1].

A função sigmoidal depende diretamente do parâmetro λ, chamado de inclinação sigmoidal, que permite uma maior suavidade da curva na região de transição e tem seu valor usualmente igual a 1.

Para permitir que a função de ativação sigmoidal assuma valores negativos na saída, no intervalo de [-1;1], utiliza-se a forma correspondente a esta, a função tangente hiperbólica, definida pela equação 3.9 e ilustrada na figura 3.9.

$$f(v) = \tanh(v) \tag{3.9}$$

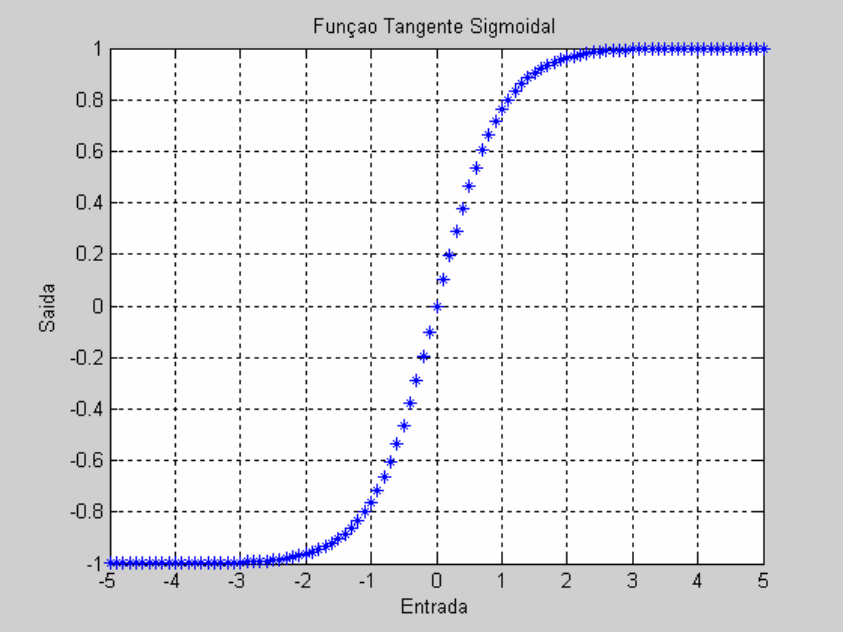

Figura 3.9: Função tangente hiperbólica.

3.6 TIPOS DE APRENDIZADO

Aprendizado certamente é a propriedade mais significativa de uma rede neural. As redes neurais possuem a capacidade de aprender por exemplos e fazer interpolações ou extrapolações do que aprenderam. O aprendizado é um processo gradual e iterado, onde os pesos são constantemente modificados, através de uma regra de aprendizado que fornece a forma como estes serão modificados. Isto é feito a partir de um conjunto de dados que serve como base de exemplo. Cada iteração deste processo de adaptação dos pesos é chamada de época de aprendizado.

Uma interessante definição do que é aprendizado para uma rede neural artificial é apresentada por BRAGA et al. (2000):

"Aprendizagem é o processo pelo qual os parâmetros de uma rede neural são ajustados através de uma forma continuada de estímulo pelo qual a rede está operando, sendo o tipo específico de aprendizagem realizada definido pela maneira particular como ocorrem os ajustes dos parâmetros".

Decorrem da definição alguns passos inerentes ao processo de aprendizagem que são:

- A rede é estimulada pelo ambiente;
- Esta sofre mudanças nos seus parâmetros livres a partir deste estímulo;
- Passa a responder de uma nova forma, devido a estas mudanças em sua estrutura interna.

Um algoritmo de aprendizagem é um conjunto de regras definidas para a solução do problema. Existem vários algoritmos utilizados, cada qual com suas peculiaridades, e suas vantagens e desvantagens.

Os métodos de aprendizado podem ser divididos em três formas básicas, segundo o nível de controle do usuário.

- Aprendizado Supervisionado:

O usuário dispõe de um comportamento de referência que ele deseja ensinar a rede, ou seja, ele tem conhecimento da entrada e da saída desejada. Com isso, a rede deve ser capaz de medir a diferença entre a saída atual e a saída desejada, e a partir desta diferença, corrigir os pesos para a redução do erro. A desvantagem deste tipo de aprendizado é que a rede não é capaz de aprender na ausência do supervisor.

Os algoritmos mais conhecidos são a regra Delta e o Backpropagation, sendo este último objeto de nosso estudo e posterior aprofundamento.

- Aprendizado Semi-Supervisionado:

Neste tipo de aprendizado, o usuário possui apenas dados imprecisos sobre o comportamento desejado. É também conhecido como aprendizado por reforço.

A diferença entre este e o aprendizado supervisionado é que o desempenho do supervisionado se baseia em um critério de erro conhecido, enquanto o por reforço se baseia em qualquer medida que possa ser fornecida ao sistema [BRAGA el al,2000].

- Aprendizado Não Supervisionado:

Este tipo é caracterizado pela ausência de supervisor, ou seja, somente os dados de entrada são fornecidos à rede, e então os pesos da rede são modificados em função de critérios internos, tais como, a repetição de padrões de ativação em paralelo de vários neurônios, não havendo uma função específica que a rede deverá aprender.

A partir do momento em que a rede estabelece uma harmonia com as regularidades estatísticas da entrada de dados, desenvolve-se nela uma habilidade de formar representações internas para codificar características de entrada e criar novas classes ou grupos automaticamente. Este tipo de aprendizado só se torna possível quando existe redundância nos dados de entrada, caso contrário, seria impossível encontrar quaisquer padrões ou características dos dados de entrada [BRAGA el al, 2000].

Os principais tipos de algoritmos que fazem uso deste tipo de aprendizado são o aprendizado por competição e o Hebbiano.

O aprendizado precisa de uma grande quantidade de dados que pode ser agrupado em uma base de dados. Uma rede pode se especializar de forma a memorizar e não mais generalizar a partir dos dados de treinamento, ou seja, a rede apresenta alto índice de acertos quando testada com pares de treinamento e quando testada com pares que estão fora de seu conjunto de treinamento este rendimento cai, não sabendo a rede

classificá-lo de forma correta. Este comportamento indesejável para as redes neurais é chamado de overfitting.

Uma técnica muito utilizada para se conseguir boas generalizações com a aplicação das redes neurais é o método da validação cruzada. Este método consiste na divisão do conjunto de dados em conjunto de treinamento, de validação e de teste. O ajuste dos pesos da rede é feito com o conjunto de treinamento e a capacidade de generalização é obtida com o uso do conjunto de validação. O melhor modelo de rede será aquele em que o desempenho conseguido com o conjunto de teste seja satisfatório.

A validação é utilizada para se fazer parar o treinamento, onde se procede o treinamento e a cada conjunto de iterações se testa o desempenho, acontecendo que quando o erro da rede for maior que a validação, este treinamento será interrompido pois é um indicativo de que a rede está perdendo a capacidade de generalização.

3.7 ARQUITETURAS DAS REDES

A forma com que os neurônios de uma rede estão interligados está diretamente ligado ao algoritmo de aprendizado, que será utilizado para treinar a rede. Pode-se caracterizar as arquiteturas de redes de três formas:

- Redes Single-Layer Feedforward

Também chamadas de redes progressivas de uma única camada, é a forma mais simples em que os neurônios estão organizados em camadas. Neste tipo de rede, tem-se uma camada de entrada conectada à camada de saída, como mostrado na figura 3.10.

Esta rede é estritamente progressiva devido ao fato de não haver conexões no sentido saída/entrada, ou seja, pode-se dizer que não há realimentação das camadas.

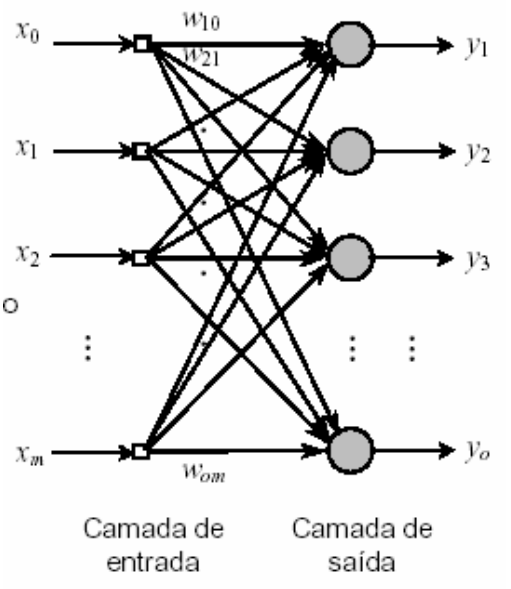

Figura 3.10: Rede progressiva single-layer.

- Redes Multilayer Feedforward

Estas redes tem por características o fato de serem também progressivas e possuir uma ou mais camadas ocultas, camadas estas situadas entre a entrada e a saída, onde os neurônios destas camadas são denominados neurônios ocultos ou intermediários. Sua função é intervir de forma a possibilitar à rede extrair informações de ordem superior.

Os neurônios da camada de entrada fornecem os sinais de entrada da segunda camada, que por sua vez fornecem a entrada para a camada seguinte, funcionando assim para o resto da rede.

A figura 3.11 mostra uma rede progressiva multicamadas, para um caso particular de duas camadas ocultas, em que há a conexão de todos os neurônios de uma camada com os da camada seguinte.

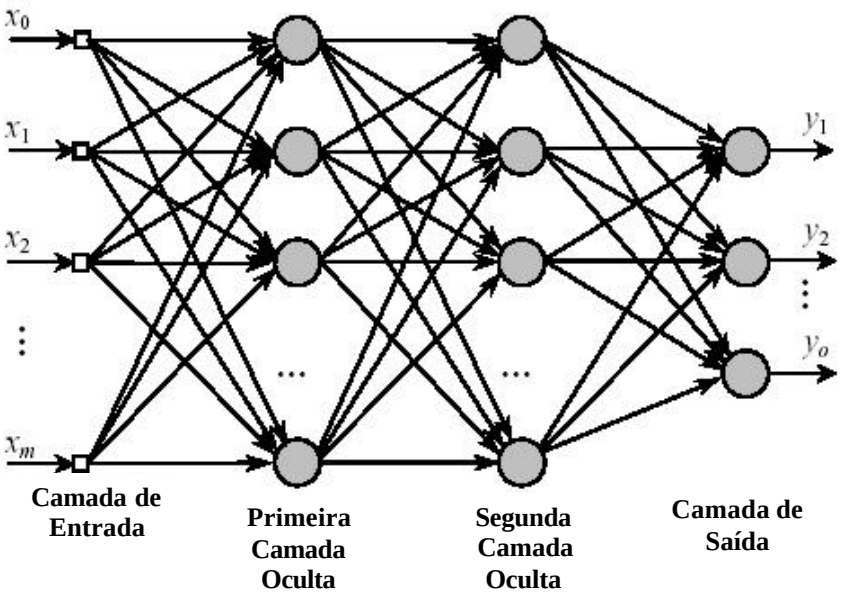

Figura 3.11: Rede progressiva multilayer.

Para o caso mostrado na figura 3.11, a rede é dita completamente conectada. Se porventura, não existirem algumas das conexões, a rede é dita parcialmente conectada.

- Redes Recorrentes

A principal diferença entre uma rede recorrente e uma rede progressiva é o fato da recorrente apresentar pelo menos um “loop” de realimentação. O modelo mais conhecido deste tipo de rede é a rede de Hopfield.

Na figura 3.12 esboça-se uma rede de uma única camada de neurônios, onde cada um deles alimenta seu sinal de saída de volta para as entradas de todos os neurônios.

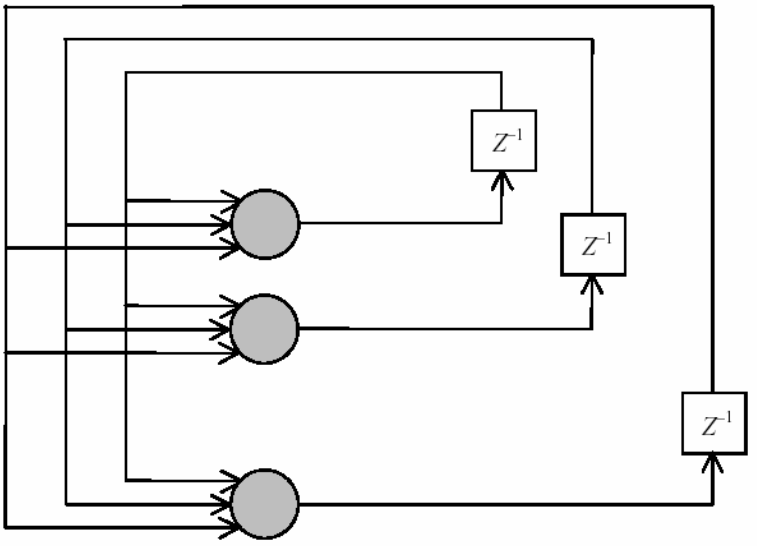

Figura 3.12: Rede recorrente.

3.8 ALGORITMO DE TREINAMENTO

Existem muitos tipos de algoritmos de treinamento específicos para determinados modelos de redes, o que torna a aplicação destas mais eficientes. Estes algoritmos diferem entre si pela forma com que os pesos são modificados.

Os algoritmos de treinamento visam, a partir do conjunto de treinamento, fazer com que a rede produza uma saída o mais próximo do valor exato, através de um processo iterativo que corrige os pesos dos neurônios de cada camada, utilizando para tal algum critério de convergência, seja ele um erro máximo para a saída da rede ou um número máximo de épocas de treinamento.

Entre os mais conhecidos e eficientes algoritmos de treinamento, e que será objeto deste estudo, o algoritmo backpropagation, será melhor explanado no item seguinte juntamente com algoritmos de ajuste dos pesos sinápticos utilizados neste trabalho.

3.8.1 ALGORITMO BACKPROPAGATION

O algoritmo backpropagation é o principal algoritmo de treinamento de rede utilizado e mais conhecido para treinamento de redes multicamadas. É frequentemente usado como o algoritmo de aprendizado em redes neurais estruturadas em camadas devido à sua eficiência (HIROSE et al., 1991).

Embora existam diversos algoritmos de treinamento de redes disponíveis, o algoritmo backpropagation tem apresentado excelentes resultados na análise e resolução de problemas de confiabilidade estrutural (SARAIVA, 1997).

O algoritmo backpropagation baseia-se no princípio do aprendizado por correção de erro, no qual o erro é retropropagado da camada de saída para as camadas intermediárias da rede neural. O desenvolvimento deste algoritmo representa um ponto fundamental em redes neurais, pois trata-se de um método computacionalmente eficiente para o treinamento de redes.

Basicamente, pode-se dividir o algoritmo em dois passos, conforme pode ser observado na figura 3.13: o direto e o reverso.

No passo direto, o vetor de entrada é aplicado e seu efeito se propaga através das camadas da rede neural produzindo uma saída, com os pesos todos fixos. No passo reverso, os pesos são reajustados de acordo com a regra de aprendizado por correção de erro. A resposta fornecida pela rede é subtraída da resposta desejada produzindo um sinal de erro, sendo este sinal propagado de volta através dos mesmos neurônios do passo direto, mas no sentido contrário do fluxo de sinais das conexões, fato este do qual advém o nome backpropagation. Os pesos são ajustados de forma que a resposta a ser produzida pela rede se aproxime da resposta esperada.

Ao passo que o conjunto de dados para treinamento da rede passa a ser representativo do problema, a rede neural multicamadas treinada com este algoritmo desenvolverá a capacidade de generalizar.

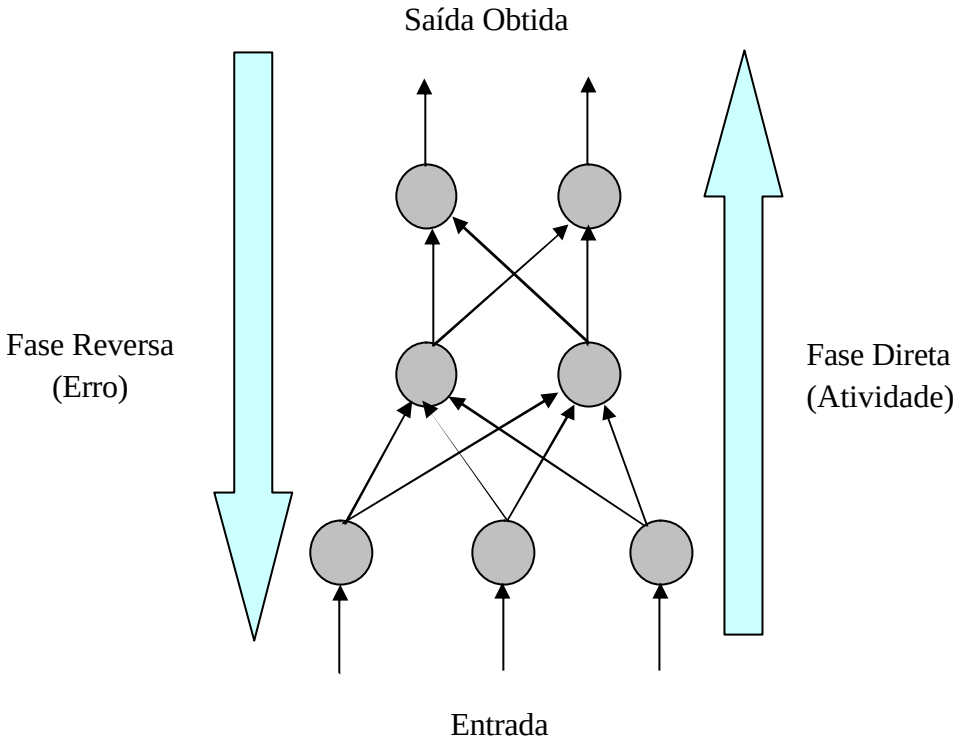

Figura 3.13: Fases do algoritmo backpropagation.

Através de um conjunto de treinamento, mostrado na equação 3.10, composto por um vetor **x** com n entradas e um vetor de saída exato **y**:

$$\psi = \{x_i, y_{ei}\} \quad i = 1,..,p \tag{3.10}$$

e pode-se definir o erro quadrático médio sobre este, apresentado pela equação 3.11:

$$E_R = \frac{1}{p}\sum_{i=1}^{p}(y_i - y_{ei})^2 \tag{3.11}$$

onde y_i é a saída fornecida pela rede e y_{ei} é o valor exato correspondente a saída da rede.

A equação 3.11 pode ser reescrita em função dos pesos, como apresentado na equação 3.12:

$$E_R = \frac{1}{p}\sum_{i=1}^{p}(f(\mathbf{W}^T.x) - y_{ei})^2 . \tag{3.12}$$

Visando determinar o conjunto de pesos que gere o mínimo erro para a função descrita na equação 3.12, existem vários algoritmos de otimização que são aplicados em conjunto com o algoritmo backpropagation, dos quais pode se destacar os algoritmos de Levenberg – Marquardt e o do gradiente descendente com momentum, também conhecida como regra delta generalizada ou algoritmo backpropagation tradicional, os quais serão utilizados em nossas aplicações.

3.8.2 ALGORITMO DO GRADIENTE DESCENDENTE COM MOMENTUM

Este algoritmo busca minimizar este erro quadrático produzido pela rede, que é expresso em função dos pesos (ver equação 3.12), o qual se almeja um conjunto de pesos otimizado que encerrará o processo de treinamento, tornando a rede apta a fornecer resultados aceitáveis.

Na regra delta generalizada, o ajuste dos pesos do neurônio é realizado numa direção ótima, sendo este processo repetido para cada par do conjunto de treinamento, fazendo-se uso dos pesos atualizados no passo anterior. Após este processo de obtenção de uma matriz dos pesos ótima, a rede se torna capaz de realizar um mapeamento do conjunto de entrada/saída.

A idéia básica deste processo pode ser compreendida tomando em consideração um único neurônio e a função de transferência sigmoidal (equação 3.7). Parte-se do conjunto de treinamento descrito na equação 3.10, no qual o sinal de entrada fornecido pela rede, apresentado na equação 3.13 e na figura 3.14:

$$\mathbf{v}=\mathbf{W}^{\mathbf{T}}.\mathbf{x} \tag{3.13}$$

que descreve o somatório das entradas multiplicadas pelos respectivos pesos e f é a função de ativação que fornece a saída da rede (y), como detalha a figura 3.14:

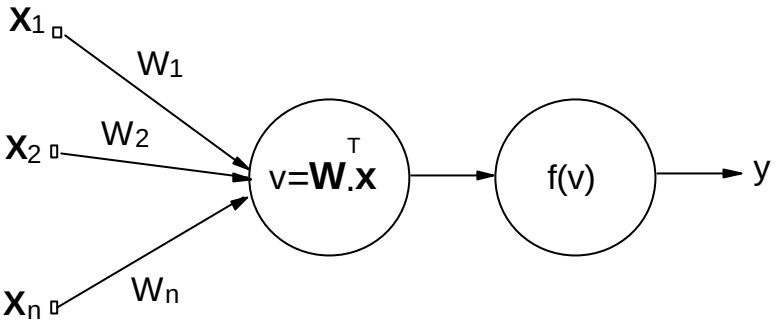

Figura 3.14: Regra delta para um neurônio isolado.

Por se tratar de um problema de otimização, torna-se necessário a determinação do gradiente do erro em relação aos pesos, mostrado na equação 3.14:

$$\frac{dE_R(\mathbf{W})}{d\mathbf{W}} = 2\sum_{s=1}^{p}\left(f(\mathbf{W}^T x_s) - y_{es}\right).\frac{df(\mathbf{W}^T x_s)}{d\mathbf{W}}. \tag{3.14}$$

Fazendo uso da regra da cadeia e da relação mostrada na equação 3.13, chega-se as equações 3.15 e 3.16:

$$\frac{dE_R(\mathbf{W})}{d\mathbf{W}} = 2\sum_{s=1}^{p}\left(f(v) - y_{es}\right).\frac{df(v)}{dv}.\frac{dv}{d\mathbf{W}} \tag{3.15}$$

$$\frac{dE_R(\mathbf{W})}{d\mathbf{W}} = 2\sum_{s=1}^{p}\left(f(v) - y_{es}\right).\frac{df(v)}{dv}.x_s\ . \tag{3.16}$$

Denotando-se o gradiente local como sendo o produto do erro pela derivada da função de ativação, mostrado na equação 3.17:

$$\delta_s = \left(f(v) - y_{es}\right).\frac{df(v)}{dv} \tag{3.17}$$

tem-se a expressão final do gradiente igual a equação 3.18:

$$\frac{dE_R(\mathbf{W})}{d\mathbf{W}} = 2\sum_{s=1}^{p}\delta_s.\mathbf{x}_s\ . \tag{3.18}$$

Para a determinação do ponto de mínimo basta igualar a equação 3.16 a zero, processo este que pode ser inviabilizado devido ao tamanho do vetor de entrada,

principalmente nos problemas de engenharia que, na maioria dos casos, engloba um grande número de variáveis.

De acordo com a regra do gradiente descendente, sendo W(k) um ponto sobre a superfície de erro, o ajuste a ser aplicado a este ponto é expresso na equação 3.19:

$$\Delta \mathbf{W}(\mathrm{k}) = \eta \frac{\partial \mathrm{E}_{\mathrm{R}}(\mathbf{W})}{\partial \mathbf{W}} \tag{3.19}$$

onde η é uma constante positiva, definida no intervalo [0;1], denominada taxa de aprendizado.

Efetuado o processo de ajuste, o valor atualizado do peso é descrito na equacão 3.20:

$$\mathbf{W}(\mathrm{k}+1) = \mathbf{W}(\mathrm{k}) - \Delta \mathbf{W}(\mathrm{k}). \tag{3.20}$$

É de particular interesse a determinação do parâmetro η, que é diretamente responsável pela rapidez do processo de aprendizado.

A aplicação desta regra pode ser agora generalizada para uma rede multicamadas, tornando-se necessário seguir uma determinada notação para facilitar a compreensão, descrita abaixo e mostrada na figura 3.15:

- Sinal de saída do neurônio i da camada L = x_i^L;
- Sinal de entrada do neurônio i da camada L = v_i^L;
- Peso entre os neurônios i da camada L e j da camada (L-1) = W_i^L.

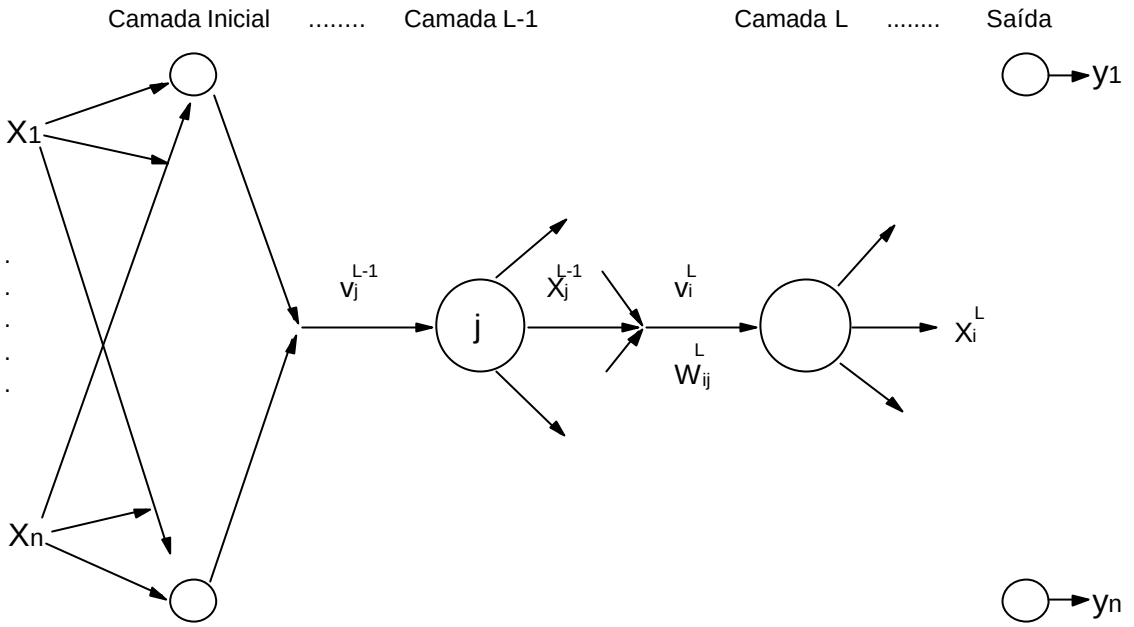

Figura 3.15: Regra Delta para uma rede multicamada.

Observando o neurônio i da camada L, os sinais de entrada e de saída respectivos a este neurônio são dados pelas equações 3.21 e 3.22:

$$v_i^L = \sum_{j=1}^{N_{L-1}} W_{ij}^L . x_j^L \tag{3.21}$$

$$x_i^L = f\left(v_i^L\right). \tag{3.22}$$

Considerando o conjunto de treinamento mostrado na equação 3.10 e começando pela última camada, o erro quadrático na saída é dado pela equação 3.23:

$$E_R\left(\mathbf{W}^L\right) = \sum_{s=1}^{p} \sum_{i=1}^{N_L} \left(f\left(\sum_{j=1}^{N_{L-1}} W_{ij}^L . x_{js}^{L-1} \right) - y_{eis} \right)^2 . \tag{3.23}$$

Os pesos serão atualizados da mesma forma com a qual foi realizada para um neurônio isolado com o uso da regra delta, assumindo a forma mostrada na equação 3.24:

$$\left[\mathbf{W}^L\right](k+1) = \left[\mathbf{W}^L\right](k) - \eta . \frac{dE_R\left[\mathbf{W}^L\right]}{d\left[\mathbf{W}^L\right]} . \tag{3.24}$$

A expressão final da regra de atualização dos pesos em função do gradiente local é mostrada na equação 3.25:

$$\left[\mathbf{W}^{L}\right](k+1) = \left[\mathbf{W}^{L}\right](k) + 2.\eta.\sum_{s=1}^{p} \delta_{s}^{L}.\left(\mathbf{x}_{s}^{L-1}\right)^{T}. \tag{3.25}$$

O algoritmo backpropagation provê uma aproximação da trajetória de movimento sobre a superfície de erro a qual, a cada ponto da superfície, segue a direção do ponto mais íngreme em busca do ponto de mínimo global.

Quanto menor for a taxa de aprendizado, menores vão ser as correções a serem aplicadas aos pesos entre cada iteração, ocasionando uma processo de convergência lento. Caso contrário, se o valor deste parâmetro for alto, pode-se obter uma aceleração no processo de convergência, mas tornando o algoritmo instável pelo fato deste poder oscilar em torno do ponto de mínimo local.

Uma forma simples de garantir a estabilidade e acelerar a convergência é a utilização da regra delta acrescida do fator de momentum. Esta é representada na equação 3.26:

$$\Delta\mathbf{W}(k) = \beta_{m}\Delta\mathbf{W}(k-1) + \eta\frac{\partial E_{R}(\mathbf{W})}{\partial\mathbf{W}} \tag{3.26}$$

onde β_{m} é denominado fator de momentum e possui a variação $0 < \beta_{m} < 1$. O efeito desta constante é aumentar a velocidade na direção do ponto de mínimo.

O que se deseja com a inserção do termo de momentum é a redução no tempo de treinamento, a melhora na estabilidade do processo e com isso aumentar a possibilidade de encontrar o mínimo global. Este processo pode ser melhor visualizado na figura 3.16, mostrada por BRAGA et al.(2000), onde está representada uma superfície de erro obtida para uma rede MultiLayer Perceptron com e sem a presença do fator de Momentum, respectivamente. A inclusão deste fator no método visa também evitar regiões onde a derivada é nula ou quase nula, também chamadas de platôs, região esta que pode ser observada na figura 3.17.

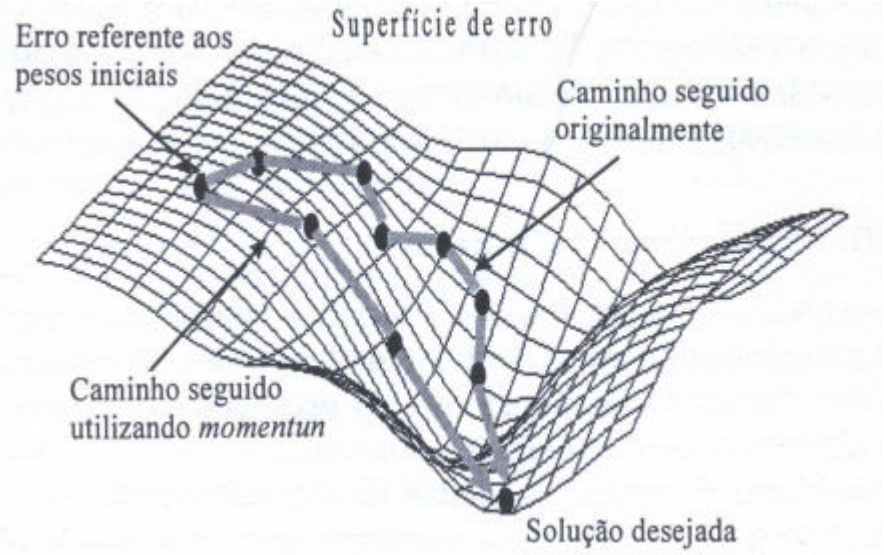

Figura 3.16: Superfície de erro com e sem o fator
de Momentum (BRAGA et al., 2000).

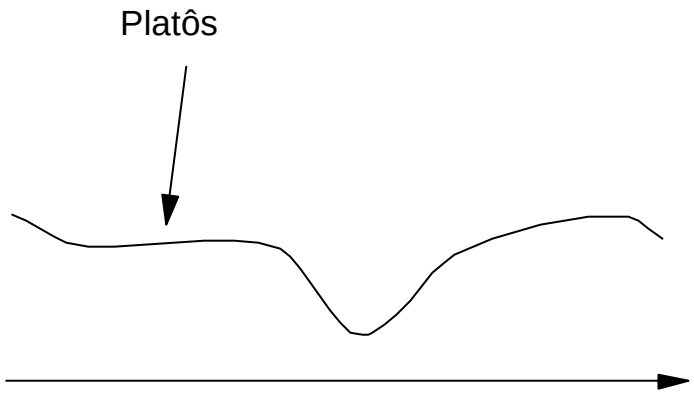

Figura 3.17: Região da superfície de erro com derivada nula ou quase nula.

Quando treinada, uma rede que faz uso do algoritmo backpropagation e da regra delta generalizada poderá necessitar de um tempo de treinamento razoavelmente longo, ou seja, se for encontrado um ponto de mínimo local na superfície de erro, este tende a parar de diminuir e, como consequência, permanecer em um valor maior que o considerado aceitável.

CALOBA (1995) e EBEHART (1990) apud SARAIVA (1997) sugerem algumas práticas para a implementação computacional deste algoritmo, as quais podem ser citadas:

- Representatividade dos dados: O conjunto de treinamento deve ser representativo da população, pois a rede depende da forma com que os dados lhe foram apresentados;
- Dimensionamento da rede: O número de neurônios em cada camada e de quantas camadas vai ser composta a rede podem ser determinados através de experimentos numéricos, podendo-se adotar como critérios: escolher apenas uma camada oculta, exceto em casos onde duas sejam preferíveis e escolher o menor número possível de neurônios na camada oculta e ir acrescentando até a função ser representada de forma adequada;
- Taxa de aprendizagem e fator de momentum: Adota-se usualmente na prática valores de de η e β_m iguais a 0.1, sendo os valores ótimos para estas constantes determinados através de experimentos numéricos;
- Valores iniciais das sinapses: Adotar pesos iniciais com valores entre –0.5 e 0.5, pois valores pequenos gerariam um treinamento muito demorado e valores maiores poderiam gerar valores de gradiente próximos a zero;
- Divisão do conjunto de treinamento: Sugere-se dividir o conjunto de dados em um conjunto de treinamento e um conjunto de testes, pois esta divisão evita que a rede memorize e perca a capacidade de generalizar.

3.8.3 ALGORITMO DE LEVENBERG – MARQUARDT

Este algoritmo é considerado o método mais rápido para treinamento de redes feedforward que possuem uma quantidade moderada de pesos sinápticos. Este se baseia, para a aceleração do treinamento, na determinação das derivadas de segunda ordem do erro em relação aos pesos, diferindo do algoritmo backpropagation tradicional que considera as derivadas de primeira ordem.

O algoritmo de Levenberg – Marquardt se baseia no método de otimização de Newton, que faz uso da matriz Hessiana. Esta matriz contém as derivadas de segunda ordem dos erros em relação aos pesos. No método de Levenberg – Marquardt se faz uma aproximação para esta matriz, mostrada na equação 3.27, determinada em função

da matriz Jacobiana, que contém as primeiras derivadas dos pesos em função dos pesos sinápticos, expressa na equação 3.28:

$$\mathbf{H} = \frac{\partial^2 E_R(\mathbf{W})}{\partial \mathbf{W}^2} \tag{3.27}$$

$$\mathbf{J} = \frac{\partial e(\mathbf{W})}{\partial \mathbf{W}} \tag{3.28}$$

onde e(**W**) é definido conforme a equação 3.29:

$$e(\mathbf{W}) = \sum_{i=1}^{p} (y_i - y_{ei}) . \tag{3.29}$$

A determinação da matriz Jacobiana é muito mais simples que a determinação da matriz Hessiana. Como, para uma rede neural, a performance de treinamento é expressa em função da soma dos erros quadráticos, a matriz Hessiana pode ser aproximada para o algoritmo de Levenberg – Marquardt pela equação 3.30:

$$\mathbf{H} = \mathbf{J}^T(\mathbf{W}).\mathbf{J}(\mathbf{W}) . \tag{3.30}$$

O método de Newton atualiza os pesos segundo a equação 3.31:

$$\mathbf{W}(k+1) = \mathbf{W}(k) - \mathbf{H}^{-1}.g_k \tag{3.31}$$

onde g_k pode ser escrito conforme a equação 3.32:

$$g_k = 2.\mathbf{J}^T(\mathbf{W}).e(\mathbf{W}) . \tag{3.32}$$

A matriz Hessiana pode ser singular, mas pode ter sua inversa calculada usando a matriz auxiliar mostrada na equação 3.33:

$$\mathbf{D} = \mathbf{H} + \mu_K \mathbf{I} . \tag{3.33}$$

Se os autovalores e os autovetores da matriz **H** são $\{\lambda_1, \lambda_2, ..., \lambda_n\}$ e $\{\phi_1, \phi_2, ..., \phi_n\}$ respectivamente, então os autovalores e autovetores da matriz **D** são determinados conforme mostrado na equação 3.34:

$$\mathbf{D}\phi_i = (\mathbf{H} + \mu_K \mathbf{I})\phi_i = \mathbf{H}\phi_i + \mu_K \phi_i = \lambda_i \phi_i + \mu_K \phi_i = (\lambda_i + \mu_K)\phi_i \tag{3.34}$$

Então, o algoritmo de Levenberg – Marquardt procede a atualização dos pesos baseado na mesma expressão do método de Newton (equação 3.31), realizando as modificações para a determinação da matriz Hessiana, mostrada na equação 3.35:

$$\mathbf{W}(k+1) = \mathbf{W}(k) - \left[\mathbf{J}^T(\mathbf{W}).\mathbf{J}(\mathbf{W}) + \mu_K \mathbf{I}\right]^{-1}.\mathbf{J}^T(\mathbf{W}).e(\mathbf{W}) \qquad (3.35)$$

onde **I** é a matriz identidade e μ_K é uma constante que transforma o algoritmo em gradiente descendente com um passo pequeno (quando μ_K é grande) e o método Gauss-Newton usando a aproximação para a matriz Hessiana (quando μ_K é zero), em que se supõe uma rápida convergência.

O parâmetro μ_K funciona como um fator de estabilização do treinamento, ajustando a aproximação de forma a utilizar a rápida convergência do método de Newton e evitando passos muito grandes que possam levar a um erro de convergência

Este algoritmo gera uma relação entre a velocidade do método de Gauss – Newton e a garantia de convergência do método do gradiente descendente.

O algoritmo pode ser resumido da seguinte maneira:

1. Se apresenta todas as entradas a rede e se calculam as respectivas saídas determinando os erros individuais e(**W**) e os quadráticos para cada entrada $E_R(\mathbf{W})$;
2. Calcula-se a matriz Jacobiana;
3. Determina-se $\Delta\mathbf{W}_k$;
4. Calcula-se a soma dos erros quadráticos usando $\mathbf{W}_k + \Delta\mathbf{W}_k$. Se esta soma é menor que o valor calculado no item 1, então se divide μ_k por υ, calcula-se $\mathbf{W}_{k+1} = \mathbf{W}_k + \Delta\mathbf{W}_k$ e retorna-se ao passo 1. Se a soma não se reduz, então se multiplica μ_k por υ e retorna-se ao passo 3. Assim o erro é sempre reduzido a cada iteração.

O algoritmo deve atingir a convergência quando a norma do gradiente for menor que algum valor pré-determinado, ou quando a soma dos erros quadráticos seja reduzida a um valor traçado como meta.

Este método apresenta convergência em menos iterações, mas requer mais cálculos por iteração devido ao cálculo de matrizes inversas. Apesar do grande esforço computacional, este segue sendo o algoritmo de treinamento mais rápido para redes neurais quando se trabalha com um número moderado de parâmetros na rede, se este número é elevado a utilização deste algoritmo é pouco prática.

3.9 EXEMPLO DE TREINAMENTO

O objetivo desta aplicação é o de analisar alguns parâmetros que influenciam no treinamento de uma rede neural com o algoritmo backpropagation utilizando a técnica do gradiente descendente com momentum e o algoritmo de Levenberg – Marquardt. Cada um destes casos será avaliado de forma separada dos outros, os quais serão fixados utilizando os valores sugeridos em alguns trabalhos.

Neste exemplo se apresenta o treinamento de uma rede neural que visa calcular o deslocamento máximo de uma viga biapoiada submetida ao carregamento mostrado na figura 3.18. O valor deste deslocamento é determinado pela equação 3.36, considerando o valor do comprimento da viga igual a 400 cm.

$$d_{Max} = \frac{5.q.L^4}{384.E.I} \tag{3.36}$$

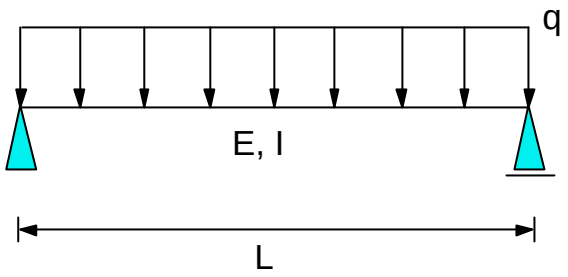

Figura 3.18: Viga biapoiada analisada no exemplo de treinamento.

Para a utilização da rede, selecionou-se como parâmetros de entrada o módulo de elasticidade E, o momento de inércia I, o carregamento distribuído q, respectivamente. Como saída, a rede determina o valor do deslocamento no meio do vão da viga. Definiu-se o intervalo de variação destes parâmetros para fornecer à rede, mostrados na tabela 3.2:

Tabela 3.2: Domínio dos parâmetros de entrada e saída da rede para o exemplo de treinamento.

Variável	**Variação**
E	20000 – 21000 (kN/cm^2)
I	6000 – 18000 (cm^3)
q	2 – 4 (kN/cm)
d_{MAX}	1.764 – 11.111 (cm)

Estes valores foram escalonados no intervalo [0,1] e fornecidos como treinamento para a rede, formando-se um conjunto de treinamento contendo 90 pares, onde estes pares foram gerados a partir do intervalo pré-definido para as variáveis mostrado na tabela 3.2.

Para se poder proceder a análise, adotou-se a seguinte variação nos parâmetros das redes neurais a serem treinadas com as estratégias de atualização de pesos propostas:

- Taxa de aprendizagem η: 0.01, 0.1 e 0.3;

- Fator de "Momentum" β_m: 0.1, 0.3 e 0.5;
- Constante μ_k: 0.001, 0.01 e 0.1
- Número de camadas intermediárias: 1 e 2;
- Número de neurônios na camada intermediária: 5 e 8.

Sobre o número de camadas intermediárias e número de neurônios em cada uma destas camadas, não existe um número ideal, sendo este determinado de forma experimental numérica. MASTERS (1994) apud SARAIVA (1997) sugere o seguinte:

- Escolher apenas uma camada oculta, que é capaz de fazer aproximações de funções contínuas de ordem qualquer, exceto em situações em que duas são preferíveis a uma;
- Usar o menor número possível de neurônios na camada intermediária, iniciando em dois e ir acrescentando até que o objetivo seja representado de forma adequada.

Já para os parâmetros taxa de aprendizagem, fator de "Momentum" e constante μ_k, toma-se o valor 0.1 como o valor ideal.

Optou-se também por analisar a influência exercida pela matriz dos pesos e pelas funções de ativação na resposta fornecida pela rede neural, onde os valores iniciais dos pesos serão avaliados através da geração de números aleatórios em intervalos que serão posteriormente citados.

Foram configuradas 16 redes para a determinação do deslocamento máximo da viga, cujas características são listadas na tabela 3.3, tomando-se como base os dados discutidos anteriormente.

Tabela 3.3: Configuração das redes para o exemplo de treinamento.

	Nci	Nnc	h	b_m	m_k	Funções de transferência
RN1	1	5	0.01	0.1	-	Sigm. – Sigm.
RN2	1	5	0.1	0.1	-	Sigm. – Sigm.
RN3	1	5	0.3	0.1	-	Sigm. – Sigm.
RN4	1	5	0.1	0.3	-	Sigm. – Sigm.
RN5	1	5	0.1	0.5	-	Sigm. – Sigm.
RN6	1	5	-	-	0.001	Sigm. – Sigm.
RN7	1	5	-	-	0.01	Sigm. – Sigm.
RN8	1	5	-	-	0.1	Sigm. – Sigm.
RN9	1	8	0.1	0.1	-	Sigm. – Sigm.
RN10	1	8	-	-	0.01	Sigm. – Sigm.
RN11	2	5	0.1	0.1	-	Sigm. – Sigm. – Sigm.
RN12	2	5	-	-	0.01	Sigm. – Sigm. – Sigm.
RN13	1	5	0.1	0.1	-	Sigm. – Sigm.
RN14	1	5	-	-	0.01	Sigm. – Sigm.
RN15	1	5	0.1	0.1	-	Sigm. – Lin.
RN16	1	5	-	-	0.01	Sigm. – Lin.

As redes RN13 e RN2 e também as redes RN14 e RN7 possuem as mesmas características citadas na tabela 3.3, exceto as matrizes de pesos, que diferem de um caso para o outro. Para as redes RN2 e RN7 consideram-se os valores dos pesos gerados de forma aleatória sem a definição de um intervalo, já para os casos das redes RN13 e RN14, os valores dos pesos são gerados de forma aleatória no intervalo de [-0.5;0.5], segundo o que sugere SARAIVA (1997).

Por mera organização e para uma melhor comparação, as redes dispostas na tabela 3.3 serão classificadas em grupos de acordo com a variação de uma determinada característica, podendo-se observar estes grupos na tabela 3.4.

Tabela 3.4: Grupos de redes analisadas no exemplo de treinamento

Grupo	Redes	Parâmetro analisado
I	RN1, RN2 e RN3	Taxa de aprendizagem
II	RN2,RN4 e RN5	Fator de Momentum
III	RN6,RN7 e RN8	Constante μ_k
IV	RN2, RN7, RN9 e RN10	Nº de neurônios por camada
V	RN2, RN7, RN11 e RN12	Nº de camadas intermediárias
VI	RN2, RN7, RN13 e RN14	Matriz de pesos
VII	RN2, RN7, RN15 e RN16	Função de transferência

Para se fazer uma análise mais precisa de como está se processando o treinamento da rede, serão apresentados gráficos que mostram a performance de treinamento em função do número de épocas escolhidas. O número de épocas representa cada iteração do processo de adaptação dos pesos, e seu valor foi escolhido para todos os casos igual a 1000.

Não serão apresentados todos os gráficos de performance das redes neurais pois seria um processo demasiadamente longo, sendo somente exibidos os gráficos mais relevantes para cada grupo analisado. Os gráficos das figuras 3.19 a 3.25 mostram as performances de treinamento para as redes RN2, RN3, RN7, RN10, RN12, RN14 e RN16, respectivamente. Estes gráficos relacionam o erro quadrático médio fornecido pela saída da rede em função do número de épocas de aprendizado para o conjunto de treinamento. Também é exposto em cada uma destas figuras o valor do erro que se pretende alcançar com o treinamento da rede neural, tomado para todos os casos como igual a 1e-7.

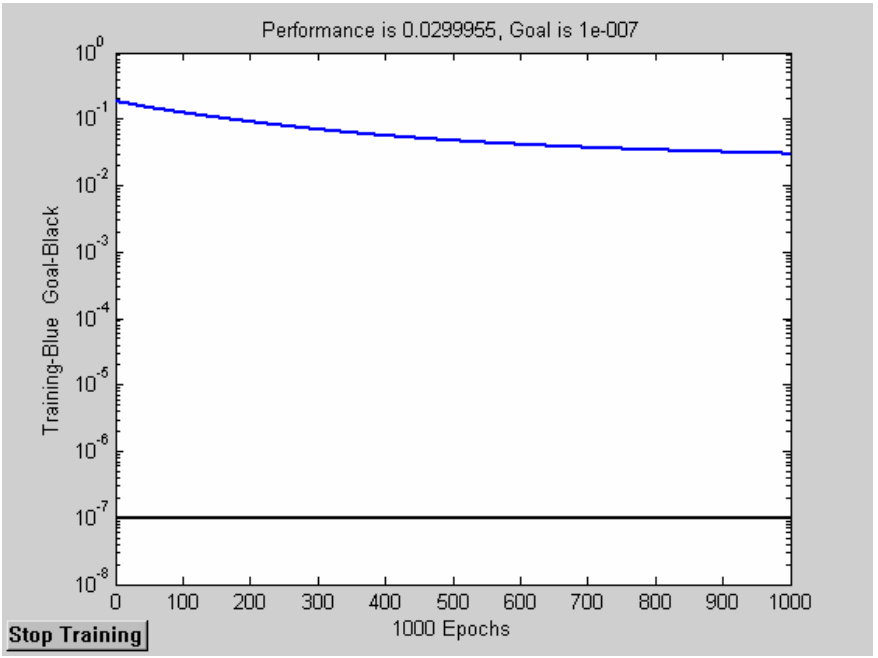

Figura 3.19: Performance de treinamento para a rede RN2.

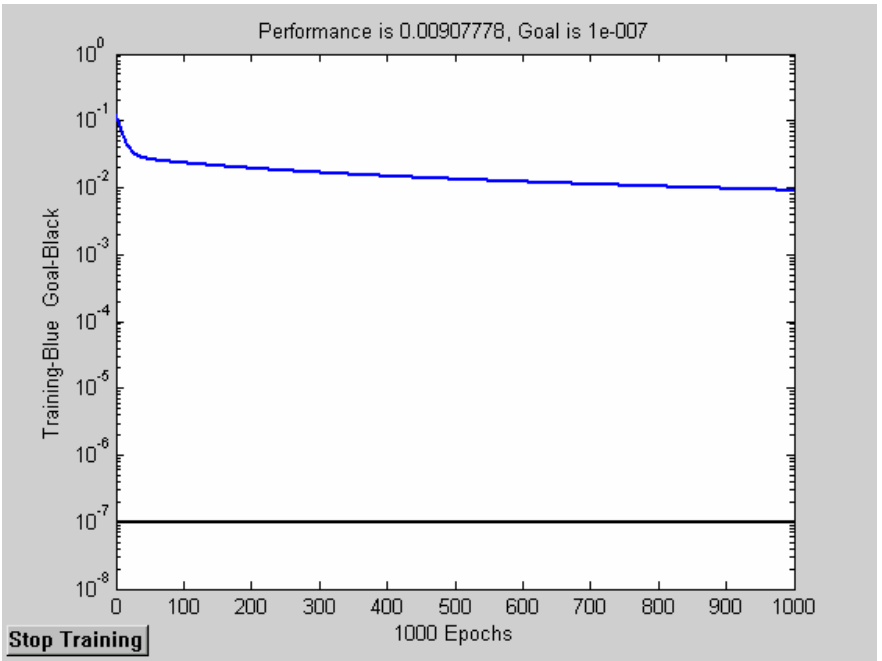

Figura 3.20: Performance de treinamento para a rede RN3.

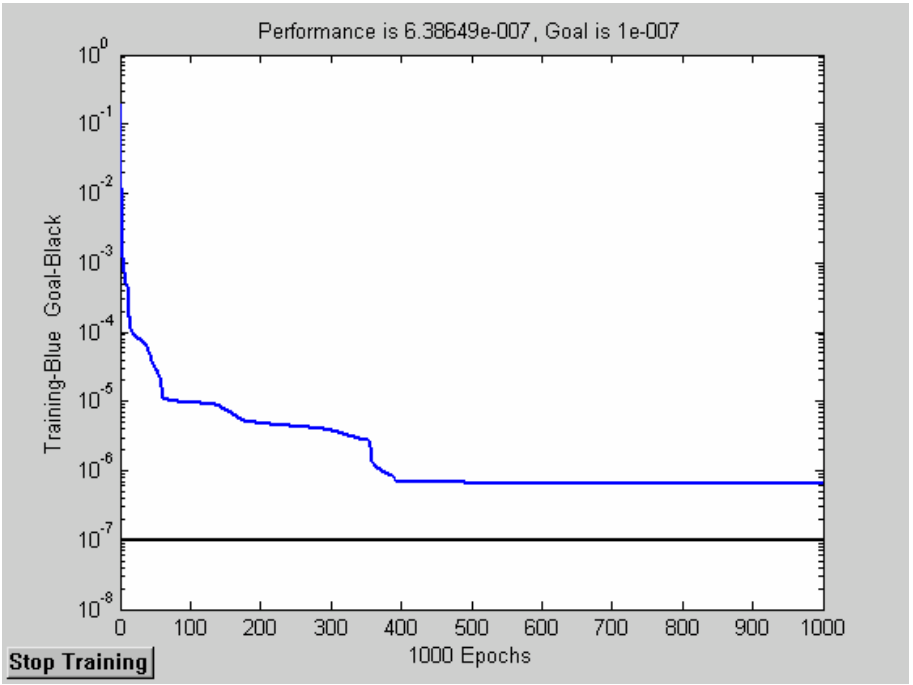

Figura 3.21: Performance de treinamento para a rede RN7.

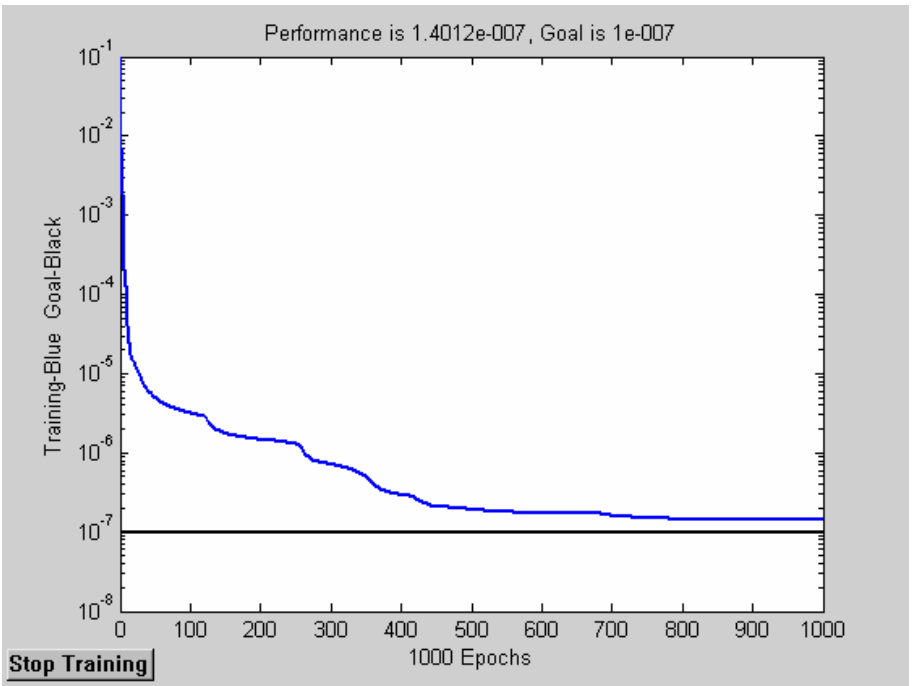

Figura 3.22: Performance de treinamento para a rede RN10.

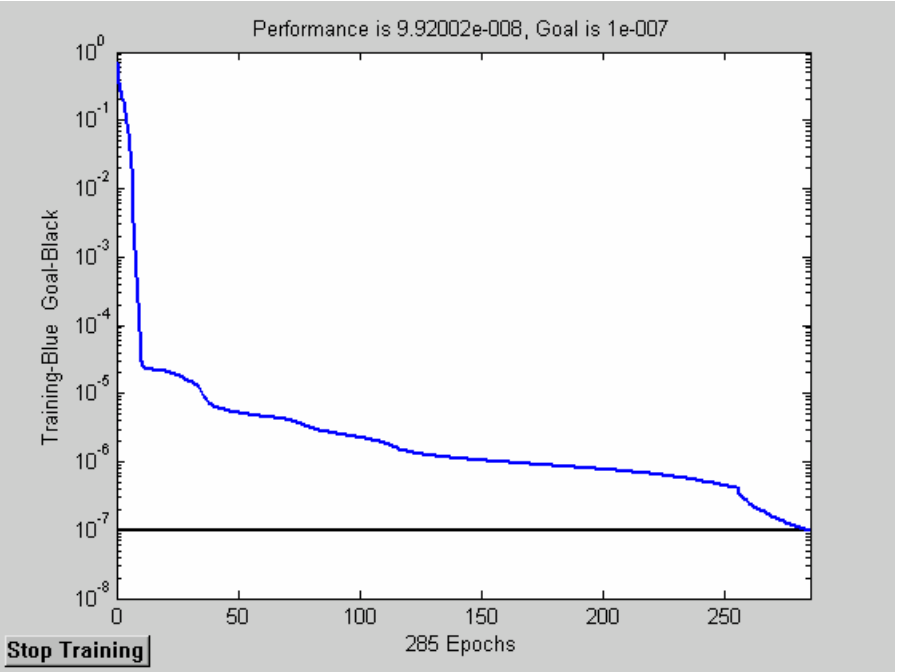

Figura 3.23: Performance de treinamento para a rede RN12.

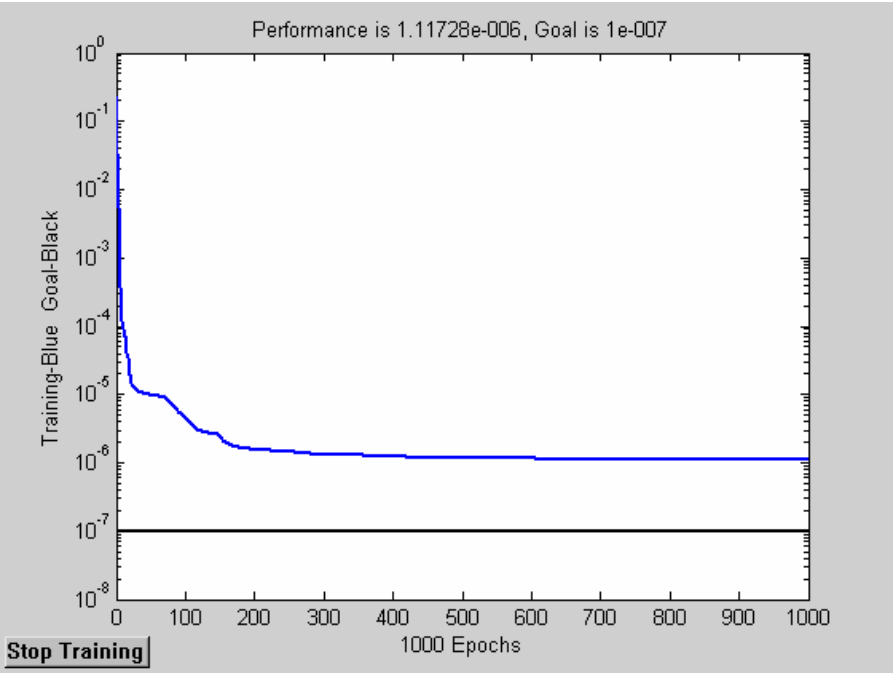

Figura 3.24: Performance de treinamento para a rede RN14.

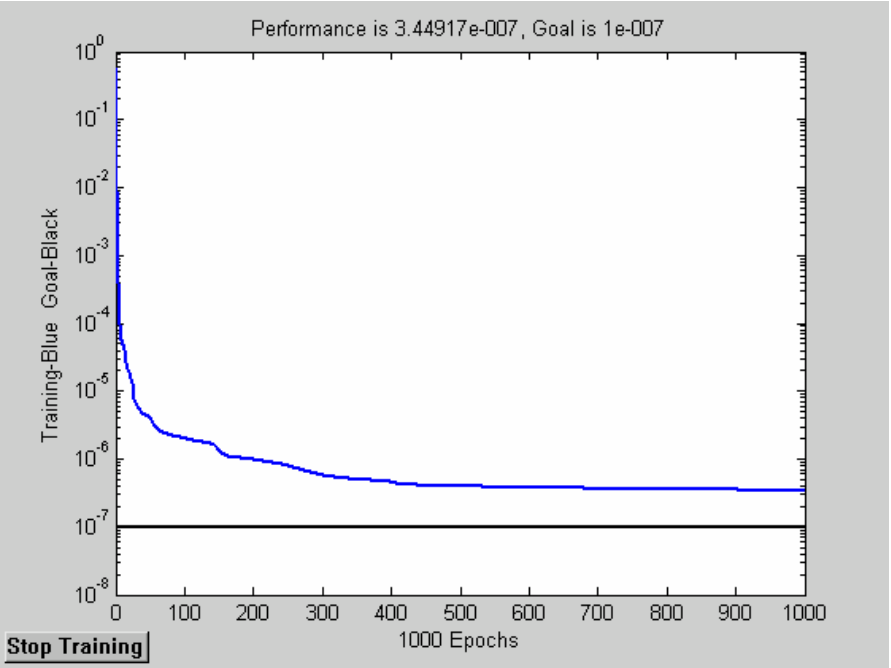

Figura 3.25: Performance de treinamento para a rede RN16.

Para cada caso avaliado nas figuras acima, considerou-se como critério de parada do treinamento a meta de erro previamente definida ou o número máximo de épocas determinado. Verifica-se que as redes treinadas com o algoritmo de Levenberg – Marquardt apresentam uma melhor performance em relação às treinadas com o algoritmo do gradiente descendente com momentum, fato que era esperado devido a sua maior rapidez do treinamento.

A tabela 3.5 mostra os valores dos pares de entrada para teste das redes configuradas no exemplo de treinamento, entradas estas que são escalonadas no intervalo [0,1] antes de serem fornecidas à rede.

Tabela 3.5: Pares de entrada para teste das redes do exemplo de treinamento.

Pares	Teste 1	Teste 2	Teste 3
E	20080	20840	20235
I	12000	17400	6780
q	2.90	2.45	3.85

Na tabela 3.6 estão apresentados os resultados da simulação das redes para cada par de teste mostrado na tabela 3.5, no qual se pode fazer uma comparação entre as redes analisadas e o resultado exato do deslocamento. Também se apresenta nesta tabela o erro fornecido para cada rede neural analisada em relação ao valor exato do deslocamento máximo.

Tabela 3.6: Resultados da simulação das redes para o exemplo de treinamento.

	Teste 1	**Erro (%)**	**Teste 2**	**Erro (%)**	**Teste 3**	**Erro (%)**
Exata	4.012	-	2.252	-	9.354	-
RN1	4.998	24.58	2.548	13.14	7.271	22.27
RN2	5.060	26.12	1.983	11.94	8.676	7.25
RN3	3.212	19.94	2.525	12.22	9.695	3.65
RN4	4.343	8.25	2.422	7.55	7.890	15.65
RN5	4.152	3.49	3.203	42.23	7.078	24.33
RN6	4.035	0.57	2.201	2.26	9.363	0.096
RN7	4.013	0.025	2.257	0.22	9.339	0.16
RN8	3.992	0.50	2.192	2.66	9.332	0.24
RN9	2.430	39.43	2.119	5.91	9.429	0.80
RN10	4.014	0.05	2.258	0.27	9.349	0.05
RN11	4.714	17.50	2.310	2.58	8.701	6.98
RN12	4.008	0.099	2.232	0.89	9.348	0.064
RN13	2.528	36.99	2.759	22.51	9.361	0.075
RN14	3.999	0.32	2.248	0.18	9.338	0.17
RN15	6.040	50.55	2.191	2.71	8.065	13.78
RN16	4.018	0.15	2.263	0.49	9.361	0.075

As redes classificadas no grupo I mostram uma tendência de diminuição do erro da resposta da rede quando o valor da taxa de aprendizagem é aumentado, apresentando a rede RN3 uma melhor performance em relação às outras.

Já para o grupo II, nota-se que para valores altos e para valores baixos do fator de "Momentum" a rede não faz boas aproximações das respostas, mostrando a rede RN4 um melhor rendimento.

No grupo III todas as redes apresentaram bons resultados, sendo estes bastante semelhantes, não se podendo citar um valor ideal para a constante μ_k, destacando a rede RN7 que indicou os menores erros.

As redes do grupo IV mostram que a inserção de neurônios na camada intermediária para o algoritmo de Levenberg – Marquardt proporciona resultados ainda melhores. As redes treinadas com o algoritmo do gradiente descendente mostram uma tendência de diminuição do erro, apesar destas redes apresentarem resultados não tão satisfatórios. A adição de neurônios na camada intermediária deve ser realizada com relativo cuidado pois a rede pode perder a sua capacidade de generalização quando este número é acrescido em demasia.

A verificação avaliada para o grupo V revela que o aumento no número de camadas intermediárias tende a reduzir o erro da resposta para ambas as redes, devendo-se ter o mesmo cuidado referido no parágrafo anterior pois um número elevado de camadas intermediárias pode prejudicar a generalização da rede.

A definição da matriz de pesos com valores aleatórios no intervalo de [-0.5;0.5] para treinamento da rede, definida no grupo VI, não proporcionou bons resultados para as redes treinadas com o algoritmo do gradiente descendente, o mesmo não acontecendo com as treinadas com o algoritmo de Levenberg – Marquardt, que apesar de não indicar os melhores resultados, estes se revelaram satisfatórios.

A proposta de mudança de função de ativação, definida no grupo VII, revelou as mesmas conclusões que para a matriz de pesos para os dois tipos de algoritmos de treinamento. Para o algoritmo de Levenberg – Marquardt, a aplicação tanto da função linear quanto sigmoidal não exerce significativa influência sobre a resposta da rede.

Apesar de não apresentarem resultados com erros aceitáveis, as redes configuradas com a técnica do gradiente descendente com momentum , a depender do problema a ser analisado pode vir a apresentar uma melhor performance, através de um aumento no número de pares do conjunto de treinamento ou na mudança de qualquer outro parâmetro de ajuste. Também é válido mencionar que para este algoritmo de atualização dos pesos é necessário um número elevado de épocas de treinamento para que apresente boas aproximações de resposta e que este é susceptível a ficar preso a um mínimo local, fato este que pode explicar a qualidade nos resultados obtidos.

Já as redes configuradas com a técnica de Levenberg – Marquardt revelaram uma maior rapidez na convergência de treinamento, além de mostrarem resultados com erros aceitáveis. Baseando-se neste fato, as redes neurais treinadas com este algoritmo serão empregadas nas aplicações posteriores deste trabalho, assim como as treinadas com o algoritmo do gradiente descendente para se poderem comparar os resultados fornecidos por estas duas técnicas.

Como já mencionado anteriormente, a escolha dos parâmetros da rede neural é feita de forma experimental numérica, portanto, os valores obtidos acima são característicos para este tipo de problema, não valendo a mesma consideração para outros problemas, podendo estes valores serem tomados como ponto de partida.

Um dos aspectos que vale ressaltar é o potencial das redes, independente do algoritmo que se esteja utilizando, em aprender a partir do treinamento e generalizar para a obtenção de respostas com aproximações aceitáveis, o que estimula a sua aplicação em diversas áreas, inclusive na área da confiabilidade estrutural.

Capítulo 4

MÉTODO DE MONTE CARLO COM REDE NEURAL

4.1 INTRODUÇÃO

Os métodos de análise de confiabilidade podem apresentar algumas restrições na determinação da probabilidade de falha de modo a inviabilizar a sua aplicação. Como já mencionado anteriormente, os métodos analíticos FORM e SORM podem apresentar problemas na determinação dos pontos de mínimo. Já o método de Monte Carlo normalmente exige um grande esforço computacional devido ao grande número de simulações requerido para se ter uma boa aproximação da probabilidade de falha.

O foco principal deste capítulo é a apresentação de uma metodologia de aplicação do método de simulação de Monte Carlo em conjunto com redes neurais artificiais. Objetiva-se com esta aplicação a substituição de etapas necessárias à determinação da probabilidade de falha pelo método de simulação de Monte Carlo, bem como fazer uso da rede neural para a substituição do conjunto análise estrutural mais análise de confiabilidade.

Torna-se necessário para a avaliação da confiabilidade, como passo primordial, a definição dos critérios a serem atendidos quanto à falha como, por exemplo, critérios de tensão máxima, de deslocamento máximo, de flambagem, entre outros, através do qual se pode definir a função de performance do problema.

Para se poder fazer uso da rede neural, torna-se necessário a escolha dos valores que serão fornecidos como conjunto de treinamento, que dependem principalmente do problema a ser tratado e do algoritmo de aprendizagem utilizado. Estes dados devem ser selecionados de forma adequada e devem cobrir todo o domínio do problema, dentro de um determinado intervalo, efetuando-se em seguida um pré-processamento de forma a tentar reduzir o espaço das variáveis envolvidas.

Este conjunto de dados de treinamento deve ser separado em conjuntos de treinamento, teste e validação, através dos quais deles se pode verificar o desempenho da rede ao longo do treinamento. O conjunto de validação pode ser importante como critério de parada pois quando o erro fornecido por este conjunto for maior que o de teste, é um indicativo de que a rede está perdendo a sua capacidade de generalização.

Os parâmetros da rede neural, como por exemplo, número de camadas e número de neurônios por camada, taxa de aprendizagem entre outros, serão determinados através de testes numéricos, seguindo-se como sugestão inicial os parâmetros encontrados na literatura. O tipo de rede neural a ser utilizada para a análise dos exemplos será uma rede feedforward multilayer perceptron com o algoritmo de treinamento backpropagation, que é considerado o mais eficiente, fazendo-se uso de dois algoritmos de otimização para a correção dos pesos que são o algoritmo do gradiente descendente com momentum e o algoritmo de Levenberg – Marquardt.

O software utilizado para a configuração das redes foi o Matlab, que possui uma base de arquiteturas de rede e de algoritmos de treinamento, além de uma interface que torna fácil o seu manuseio. Este software também possibilita a implementação de rotinas computacionais, que foram necessárias na solução de alguns problemas em estudo.

Para se fazer a análise da confiabilidade, foi utilizado o programa **FERUM** (**F**inite **E**lement **R**eliability **U**sing **M**atlab), desenvolvido por HAUKAAS (2001). Este programa é capaz de realizar análise estrutural linear elástica e a análise de confiabilidade estrutural, tendo como opções para a determinação da probabilidade de

falha os métodos analíticos FORM/SORM e o método de Monte Carlo com a técnica de Amostragem por Importância.

Os resultados obtidos com os problemas analisados serão posteriormente comparados com os resultados fornecidos por outros programas de análise de confiabilidade e também com exemplos encontrados na literatura. A utilização destes programas de análise de confiabilidade também serão úteis na obtenção do conjunto de treinamento necessário ao aprendizado da rede.

As estratégias de aplicação das redes neurais serão abordadas de três formas, as quais passam pela escolha do parâmetro a ser substituído pela rede e dependerá da forma como a função de performance estiver disposta no problema. Também serão avaliadas a solução do problema utilizando o método de Monte Carlo Clássico e as técnicas de redução de variância Amostragem por Importância e Esperança Condicionada. Estes métodos servirão para comprovar a viabilidade de aplicação das redes neurais.

4.2 MÉTODO DE MONTE CARLO COM REDE NEURAL

Uma das principais características da aplicação das redes neurais no método de Monte Carlo é a redução do esforço computacional que esta tende a proporcionar, produzindo resultados aceitáveis. Outro ponto importante é que caso os resultados fornecidos pela rede neural não sejam satisfatórios, ou seja, caso se acompanhe os erros fornecidos e este se apresente acima do valor esperado, a rede poderá ser treinada novamente com a inserção de novos pontos em seu conjunto de treinamento que poderão propiciar uma melhor performance.

A aplicação das redes será definida segundo três estratégias. Estes três casos serão abordados a seguir, onde se fará menção dos detalhes da aplicação das redes no problema de confiabilidade estrutural.

4.2.1 ALGORITMO A1

Para a utilização deste algoritmo é necessário o conhecimento prévio da função de performance, das variáveis aleatórias envolvidas na análise e de suas características estatísticas.

A rede é utilizada em conjunto com a técnica de Esperança Condicionada para gerar valores da função de distribuição acumulada da variável X_m, variável esta com maior dispersão entre as variáveis envolvidas, quando isso for possível de ser feito, como comentado anteriormente, no qual o treinamento da rede se dará através do conjunto mostrado na equação 4.2:

$$\psi = [r_i ; F_{Xm}]_S \quad s = 1..p \tag{4.2}$$

onde: F_{Xm} = Valor da distribuição acumulada da variável X_m,

r_i = Vetor com valores aleatórios representativos das variáveis,

p = Número de pares de treinamento.

Após o processo de treinamento, a partir de um conjunto de N valores aleatórios representativos das variáveis envolvidas no problema, a rede neural fornecerá N valores da função de distribuição acumulada da variável X_m, através do qual se poderá calcular a probabilidade de falha utilizando a equação 2.40.

Os passos para a determinação da probabilidade de falha utilizando o algoritmo A1 estão mostrados na figura 4.1:

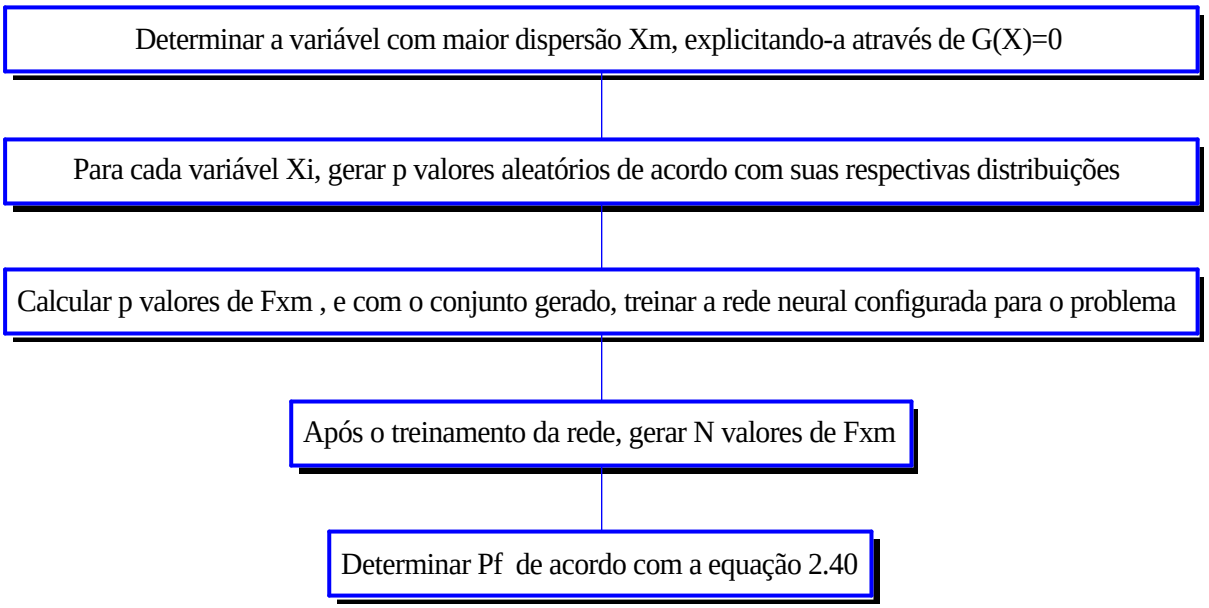

Figura 4.1: Fluxograma do algoritmo A1.

4.2.2 ALGORITMO A2

Este algoritmo é utilizado quando a função de falha do problema é expressa de forma implícita, necessitando realizar-se uma análise estrutural para a determinação do parâmetro que ocasionará a falha, análise esta que pode ser linear ou não linear, a depender da natureza do problema.

O treinamento da rede neural é realizado através do conjunto apresentado na equação 4.3:

$$\psi = [r_i ; CF_i]_s \quad s = 1..p \tag{4.3}$$

onde: CF_i = Critério de falha,

r_i = Vetor com valores aleatórios representativos das variáveis,

p = Número de pares de treinamento.

Como para se proceder a aplicação deste algoritmo é necessário se fazer uma análise estrutural, a aplicação do método de Monte Carlo pode se tornar cara devido ao grande número de simulações necessário.

Neste processo, utiliza-se uma rede para a determinação da relação complementar da função de falha (análise estrutural). A partir desta determinação, calcula-se a probabilidade de falha utilizando o método de Monte Carlo através da equação 2.28.

Os passos para a determinação de P_f fazendo uso do algoritmo A2 estão apresentados no fluxograma da figura 4.2:

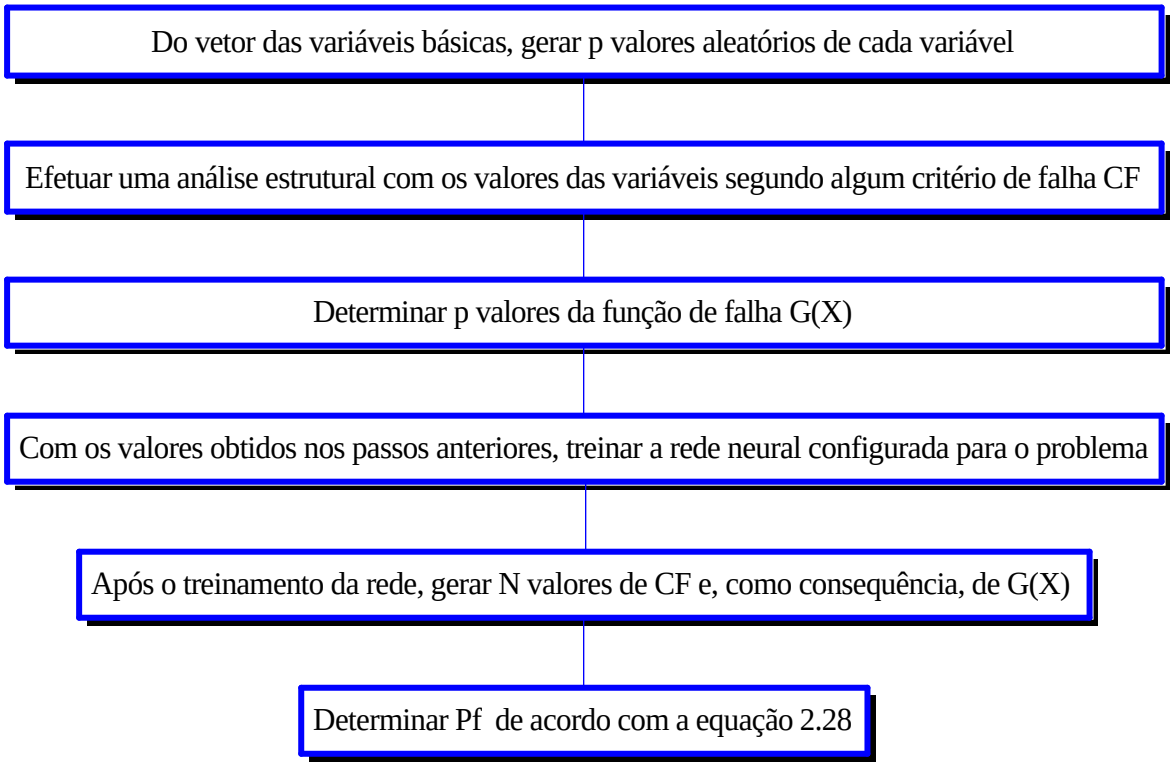

Figura 4.2: Fluxograma do algoritmo A2.

4.2.3 ALGORITMO A3

O algoritmo A3 visa a substituição da aplicação dos processos de análise estrutural e de confiabilidade, utilizando a rede neural para este fim. Procurar-se-á, através do uso de um programa que proceda estas análises, simultaneamente, obter o conjunto de treinamento, e em seguida, treinar e dispor a rede para a determinação da

probabilidade de falha, onde este conjunto deve ser gerado através da definição de um intervalo pré-determinado para uma variável, entre as variáveis envolvidas no problema, denominada variável de contole. Para as demais variáveis serão mantidas as suas características.

O treinamento da rede neural com o uso deste algoritmo é realizado através do conjunto apresentado na equação 4.4:

$$\psi = [r_i ; P_f]_S \quad s = 1..p \tag{4.4}$$

onde: P_f = Probabilidade de falha,

r_i = Vetor com valores aleatórios representativo das variáveis,

p = Número de pares de treinamento.

Para as aplicações deste algoritmo se fez uso do programa FERUM, utilizando o método de amostragem por importância para a geração do conjunto de treinamento da rede neural.

A determinação de P_f através do algoritmo A3 segue os passos apresentados no fluxograma da figura 4.3:

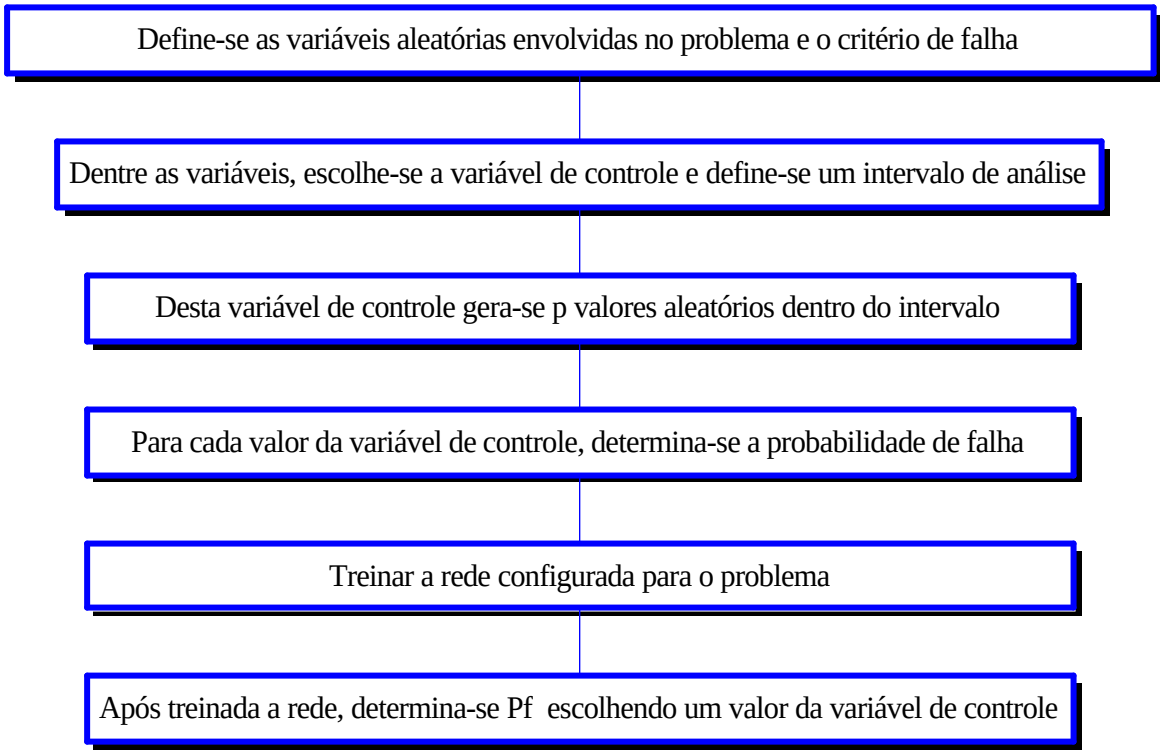

Figura 4.3: Fluxograma do algoritmo A3.

Capítulo 5

APLICAÇÕES

5.1 CONSIDERAÇÕES INICIAIS

Neste capítulo serão mostradas as aplicações desenvolvidas durante a elaboração deste trabalho. Todas estas tomam como base os algoritmos apresentados no capítulo anterior e são detalhadas a problemas de confiabilidade estrutural.

As redes avaliadas nas aplicações que seguem são todas do tipo Multilayer Perceptron. Ao longo das aplicações, estas receberão denominações no qual possuirão duas terminações, LM e GDM, que denotarão a utilização dos algoritmos de Levenberg – Marquardt e do gradiente descendente com momentum, respectivamente, posto para facilitar a distinção entre os dois métodos.

Também receberão denominações no decorrer das aplicações o método de simulação de Monte Carlo, representado por MCC, o método de Amostragem por Importância, representado por MCIS e o método da Esperança Condicionada, denominado de MCCE. Estes métodos serão úteis para a comparação dos valores obtidos nas análises.

Para todos os casos analisados neste capítulo considerou-se a matriz de pesos das redes neurais geradas de forma aleatória pelo programa Matlab. Considerou-se como critério de parada para o treinamento das redes avaliadas nestas aplicações o número máximo de épocas igual a 1000 ou o erro mínimo esperado de 1e-10, obtido da comparação da saída da rede com relação a saída exata, prevalecendo o que primeiro acontecer.

A geração do conjunto de treinamento para os casos analisados foi procedida tomando como base os valores aleatórios oriundos das variáveis do problema, onde para cada caso se procurou determinar os valores limites para se poder definir um intervalo de análise para a rede neural.

Em todos os casos analisados, efetuou-se a divisão do conjunto de pares para treinamento das redes em conjuntos de treinamento, validação e teste. O conjunto de treinamento é responsável pela modificação dos pesos da rede. Já o conjunto de validação tem a finalidade de garantir a capacidade de generalização e o conjunto de testes verifica se a rede não se ajustou de forma excessiva os dados apresentados no treinamento e na validação, qualificando a melhor rede aquela que apresentar o menor erro para o conjunto de teste.

O conjunto de validação é utilizado para se fazer a interrupção antecipada, que visa treinar a rede com o conjunto de treinamento e de tantas em tantas iterações testar o seu desempenho da rede com o conjunto de validação. Caso o erro da rede com relação ao conjunto de validação aumente, o treinamento é interrompido, pois a rede está perdendo a capacidade de generalização.

Os resultados obtidos, tanto para a análise da performance das redes, quanto para a análise da confiabilidade estrutural, serão dispostos em gráficos e tabelas para facilitar a visualização e compreensão dos resultados.

5.2 APLICAÇÕES DO ALGORITMO A1

5.2.1 APLICAÇÃO 1

A aplicação consiste de um pórtico plano que possui três modos de falha, representado na figura 5.1, onde são consideradas como variáveis aleatórias os momentos plásticos resistentes, denominados de Z1, Z2, Z3, Z4 e Z5, dados em kN.m, e

as cargas aplicadas na direção horizontal e vertical H e V, respectivamente, dadas em kN.

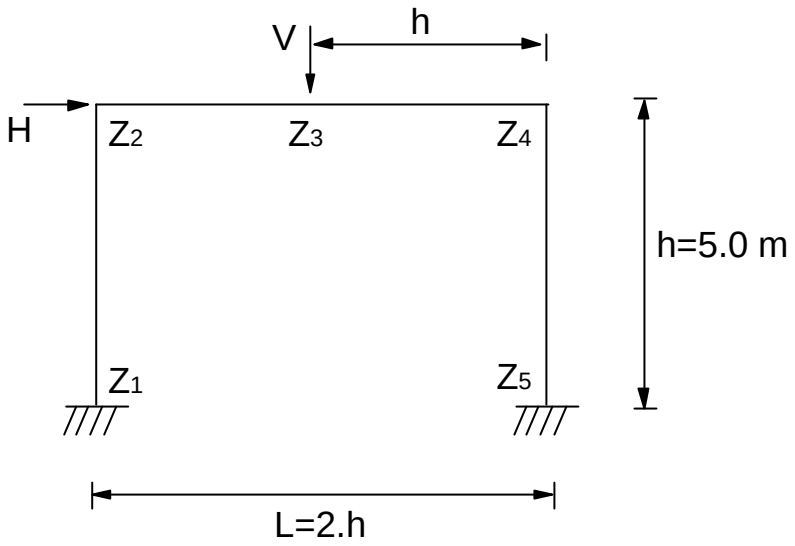

Figura 5.1: Pórtico plano com três mecanismos de falha da aplicação 1.

A tabela 5.1 mostra os valores que representam as características de cada variável envolvida no problema. O parâmetro Cov representa a covariância da variável e, λ e ξ são os parâmetros da distribuição lognormal.

Tabela 5.1: Características das variáveis aleatórias da aplicação 1.

Variável	**Distribuição**	m	s	**COV**	l	x
Z1,…, Z5	Lognormal	134.9	13.49	0.1	4.9	0.099
H	Lognormal	50	15	0.3	3.87	0.294
V	Lognormal	40	12	0.3	3.65	0.294

Para a análise deste exemplo são considerados três mecanismos de colapso, representado na figura 5.2, cujas funções de performance estão mostradas na equação 5.1, podendo cada uma delas provocar a falha da estrutura, determinando com que a probabilidade da estrutura como um todo falhar seja expressa em função da falha dos três mecanismos, apresentada na equação 5.2, que denota a possibilidade de ocorrência de algum dos modos.

$$
\begin{aligned}
G_1(\mathbf{X}) &= Z_1 + Z_2 + Z_4 + Z_5 - H.h \\
G_2(\mathbf{X}) &= Z_1 + 2.Z_2 + 2.Z_4 + Z_5 - H.h - V.h \\
G_3(\mathbf{X}) &= Z_2 + 2.Z_3 + Z_4 - V.h
\end{aligned}
\tag{5.1}
$$

$$
P_f = \bigcup_{i=1}^{3} P(G_i \leq 0)\,. \tag{5.2}
$$

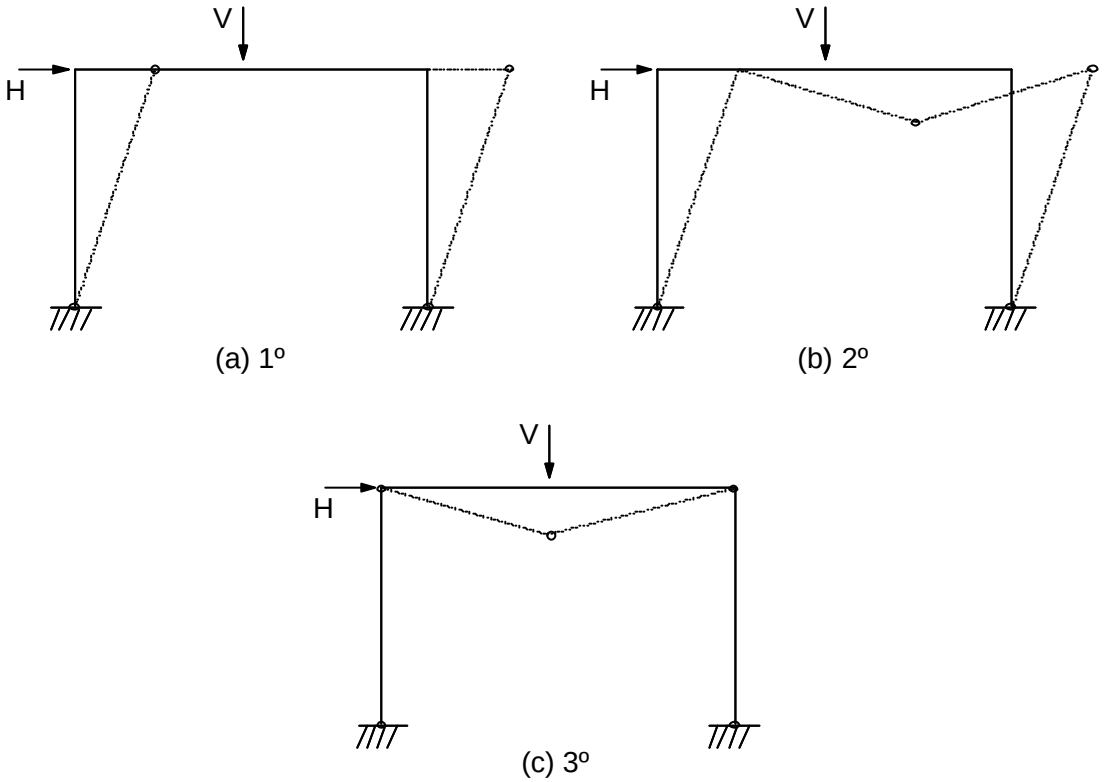

Figura 5.2: Representação dos mecanismos de falha do pórtico plano.

Nos três casos considerados na equação 5.1, a determinação da probabilidade de falha é feita através da técnica de Esperança Condicionada, no qual se escolhe a variável com maior dispersão e através dela se quantifica a falha. Para os casos da função de falha 1 e 2, escolheu-se a variável H, já para o caso 3, a variável selecionada foi V.

Para a função de falha 1, a relação pode ser agora escrita como mostrado na equação 5.3:

$$
\begin{aligned}
P_{f1} &= P[G_1(\mathbf{X}) \le 0] \\
P_{f1} &= P[Z_1 + Z_2 + Z_4 + Z_5 - H.h \le 0] \\
P_{f1} &= P[H \ge (Z_1 + Z_2 + Z_4 + Z_5)/h] \\
P_{f1} &= 1 - F_H[(Z_1 + Z_2 + Z_4 + Z_5)/h]
\end{aligned}
\tag{5.3}
$$

Para os demais modos de falha análise semelhante é procedida, podendo-se reescrever as expressões como apresentado nas equações 5.4 e 5.5:

$$
\begin{aligned}
P_{f2} &= P[G_2(\mathbf{X}) \le 0] \\
P_{f2} &= P[Z_1 + 2.Z_2 + 2.Z_4 + Z_5 - H.h - V.h \le 0] \\
P_{f2} &= 1 - F_H[(Z_1 + 2.Z_2 + 2.Z_4 + Z_5 - V.h)/h]
\end{aligned}
\tag{5.4}
$$

$$
\begin{aligned}
P_{f3} &= P[G_3(\mathbf{X}) \le 0] \\
P_{f3} &= P[Z_2 + 2.Z_3 + Z_4 - V.h \le 0] \\
P_{f3} &= 1 - F_V[(Z_2 + 2.Z_3 + Z_4)/h]
\end{aligned}
\tag{5.5}
$$

onde F_H e F_V representam as funções de distribuição acumulada das variáveis H e V respectivamente.

Foram configuradas seis redes neurais para a determinação da falha dos três mecanismos de colapso, cujas características podem ser observadas na tabela 5.2, onde o termo Nci representa o número de camadas intermediárias e Nnc representa o número de neurônios nestas camadas.

Tabela 5.2: Configuração das redes neurais da aplicação 1.

Rede	**Nci**	**Nnc**	h	b_m	m_k	**Função de Transferência**
RNPORLM1	1	5	-	-	0.01	Tangente – Linear
RNPORLM2	1	5	-	-	0.01	Tangente – Linear
RNPORLM3	1	5	-	-	0.01	Tangente – Linear
RNPORGDM1	1	5	0.1	0.1	-	Tangente – Linear
RNPORGDM2	1	5	0.1	0.1	-	Tangente – Linear
RNPORGDM3	1	5	0.1	0.1	-	Tangente – Linear

As redes neurais enumeradas em 1, 2 e 3 são utilizadas para a determinação dos valores das funções de distribuição acumulada das variáveis com maior dispersão representativa dos respectivos modos, valores que são necessários à determinação da falha pela técnica de redução de variância, possuindo as mesmas características, exceto o número de neurônios na camada de entrada que varia em função do número de termos fornecidos a entrada da rede. A geração de números aleatórios é realizada pelo programa Matlab.

O conjunto de treinamento constou de 80 pares, onde foi realizado um pré-processamento dos dados do vetor de entrada, no qual eles são normalizados para um conjunto de dados possuindo média 0 e desvio padrão igual a 1 e possuía a dimensão do vetor de entrada de 4, 5 e 3 para as funções de falha G_1, G_2 e G_3 respectivamente. O mesmo procedimento foi realizado para o conjunto de saída da rede.

Após simulada as redes é efetuado um pós-processamento, fazendo com que a saída fornecida pela rede retorne ao intervalo original.

Também foi implementada uma rotina que determina a probabilidade de falha para os três mecanismos de colapso utilizando a técnica de Esperança Condicionada, denominada de MCCE.

Nas figuras 5.3 e 5.4 são apresentadas as performances de treinamento para a rede RNPORLM1 e RNPORGDM1, respectivamente, para a determinação do modo de falha 1. Os gráficos apresentados possuem a distinção entre treinamento, teste e validação, dos quais estes dados constam do conjunto de treinamento.

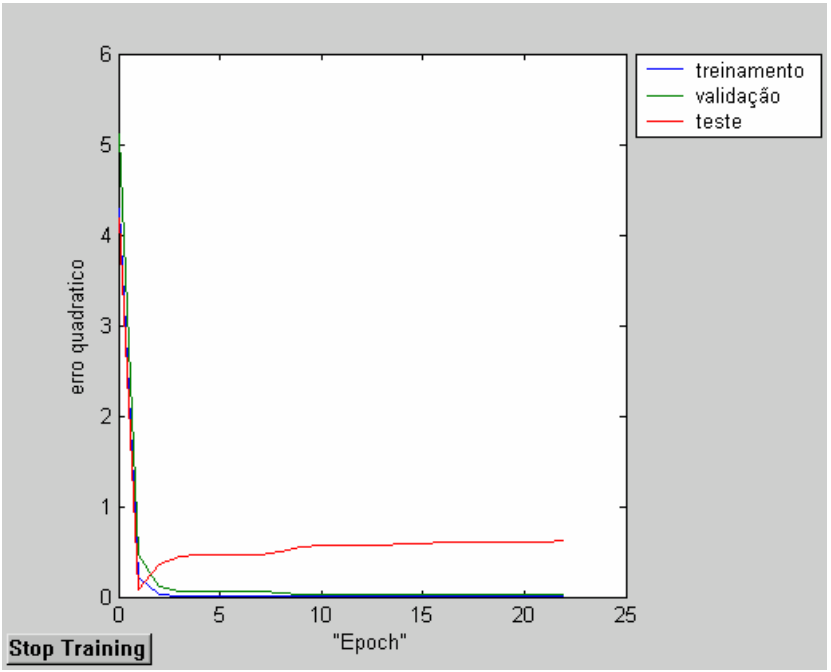

Figura 5.3: Performance de treinamento da rede RNPORLM1.

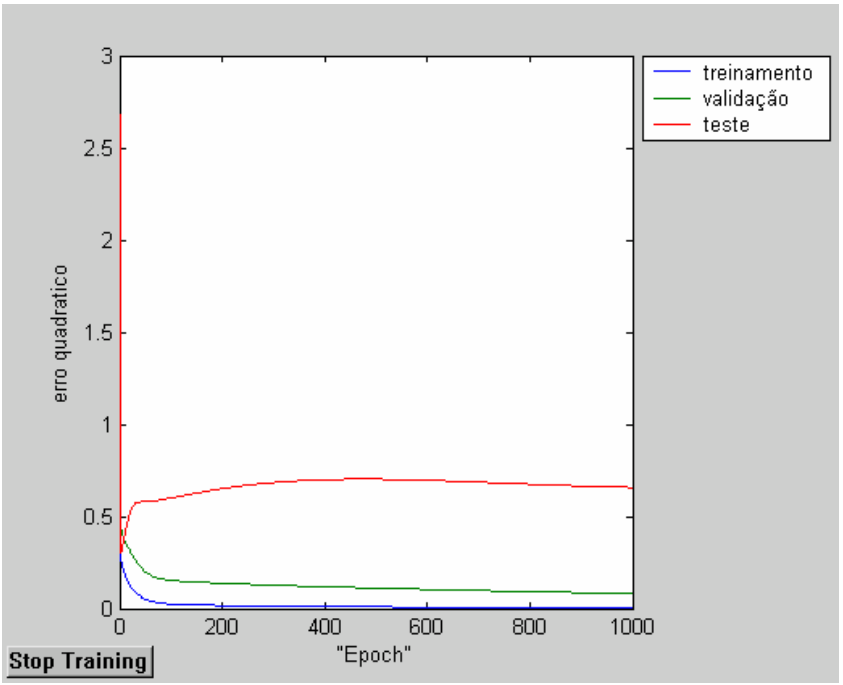

Figura 5.4: Performance de treinamento da rede RNPORGDM1.

A tabela 5.3 expõe os resultados obtidos com a utilização da rede neural para a determinação da probabilidade de falha. Observa-se nesta tabela os valores fornecidos por MADSEN (1986) apud SARAIVA (1997) para os métodos FORM e SORM

convencionais, os valores fornecidos por SAGRILO (1994) utilizando os algoritmos de análise linear e não linear ALR e ANLR, que fazem a consideração de uma superfície de resposta linear e quadrática respectivamente, para os métodos FORM e SORM.

Observa-se também o valor fornecido por SARAIVA (1997) utilizando uma rede neural, denominada de MCRN1, para substituir o cálculo da função de distribuição acumulada, rede cujos parâmetros estão apresentados a seguir:

- Número de camadas ocultas = 1;
- Neurônios na camada oculta = 7;
- Taxa de aprendizagem $\eta = 0.1$;
- Fator de Momentum $\beta_m = 0.1$;
- Função de ativação = Sigmoidal.

Tabela 5.3: Valores das probabilidades de falha para os diferentes mecanismos de falha do pórtico plano da aplicação 1.

Método	P_{f1}	P_{f2}	P_{f3}
FORM (Madsen)	0.00360	0.00199	0.000291
SORM (Madsen)	0.00322	0.00267	0.000283
FORM (ALR)	0.00335	0.00197	0.000294
FORM (ANLR)	0.00335	0.00197	0.000294
SORM (ANLR)	0.00318	0.00281	0.000275
MCRN1 (500 sim.)	0.00316	0.00275	0.000260
MCCE (500 sim.)	0.00326	0.00246	0.000293
RNPORLM (500 sim.)	0.00310	0.00230	0.000232
RNPORGDM (500 sim.)	0.00300	0.00240	0.000231

Os valores mostrados na tabela 5.3 para a análise das redes configuradas se mostram perfeitamente comparáveis para um número total de 500 simulações realizadas.

Tanto as redes treinadas com o algoritmo de Levenberg-Marquardt quanto com o algoritmo do gradiente descendente com momentum se revelaram bastante semelhantes no que concerne aos resultados fornecidos, apesar do primeiro método atingir a convergência do treinamento para um número de épocas menor.

Verificam-se também os bons resultados apresentados pela função MCCE, que também serviu como parâmetro de comparação para atestar os resultados obtidos pelas redes neurais.

5.2.2 APLICAÇÃO 2

Esta aplicação se refere a uma treliça isostática representada na figura 5.5, onde são consideradas como variáveis básicas a carga aplicada P e a resistência R, cujas características estatísticas destas variáveis estão apresentadas na tabela 5.4.

Este é um problema conhecido na literatura como problema básico de confiabilidade, tendo sido analisado por inúmeras vezes (SARAIVA,1997).

Os valores dos esforços normais das barras da treliça são os seguintes:

- Barras 1 e 2: $P\frac{\sqrt{3}}{3}$;
- Barra 3: $P\frac{\sqrt{3}}{6}$.

Tabela 5.4: Características das variáveis aleatórias da aplicação 2.

Variável	Distribuição	m	s
Carga (P)	Normal	14	1.25
Resistência (R)	Normal	11	1.50

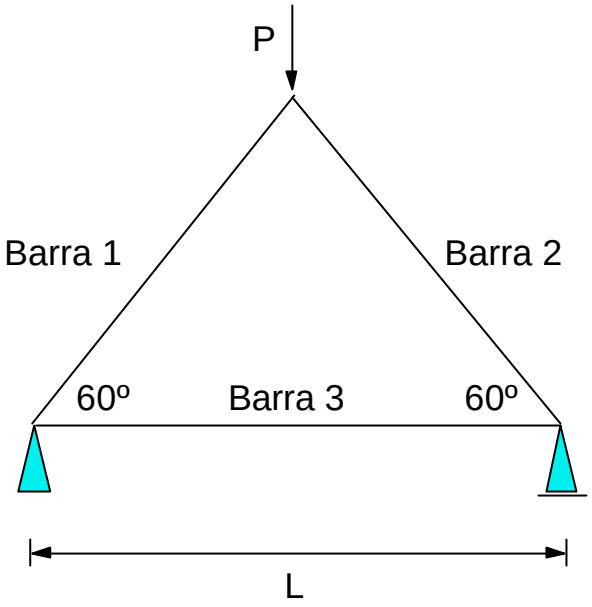

Figura 5.5: Treliça isostática da aplicação 2.

Neste problema é considerado o efeito da carga P contra a resistência R, sendo a função de falha definida pela equação 5.6:

$$G(\mathbf{X}) = R - P \,. \tag{5.6}$$

A probabilidade de falha para este caso pode ser calculada, para uma barra qualquer da treliça, utilizando-se a equação 5.7:

$$P_f\,(\text{Barra i}) = P[(R - P) \le 0] = \int_{-\infty}^{\infty} F_R\,(\mathbf{X}).f_P\,(\mathbf{X}).d\mathbf{X} \tag{5.7}$$

onde: $F_R(\mathbf{X})$ = Função de distribuição acumulada da variável R;

$f_P(\mathbf{X})$ = Função densidade de probabilidade da variável P.

No caso destas variáveis possuírem distribuições normais, a integral definida na equação 5.7 pode ser calculada através da equação 5.8:

$$P_f\,(\text{Barra i}) = \Phi\left(-(\mu_R - \mu_{Pi}) / \left(\sigma_R^2 + \sigma_{Pi}^2\right)^{1/2} \right) \tag{5.8}$$

onde: μ_R, μ_{Pi} são as médias das variáveis R e P na barra i;

σ_R, σ_{Pi} são os desvios padrão de R e P na barra i;

Φ é a função de distribuição acumulada normal padrão.

Fazendo uso da equação 5.8, chega-se a probabilidade de falha para as barras da treliça: Barra 1 e 2 – P_f = 0.039848; Barra 3 – P_f = 3.24e-6.

Para a utilização das redes, aplicou-se a técnica de redução de variância Esperança Condicionada, no qual se escolhe a variável com maior dispersão e a explicita em função das outras para a redução da ordem da integral que define a falha.

O conjunto de treinamento para este exemplo apresenta 100 pares, que consistem em valores da resistência R, no qual é realizado um pré-processamento para normalização destas entradas, e a saída fornece valores da função de distribuição acumulada da variável R.

A rede configurada por SARAIVA (1997), denominada de MCRN1, possui as seguintes características:

- Rede Feedforward sem realimentação com técnica Backpropagation;
- Número de camadas intermediárias = 1;
- Número de neurônios na camada intermediária = 5;
- Taxa de aprendizagem $\eta = 0.1$;
- Fator de "Momentum" $\beta_m = 0.1$.

Além desta rede, SARAIVA (1997) utilizou os métodos de Monte Carlo Clássico e o método de Monte Carlo com a técnica de redução de variância "Antithetic Variable", denominado de MCRV, que possibilitarão uma comparação mais precisa dos valores fornecidos pelas redes estudadas.

As redes neurais configuradas nesta aplicação, denominadas de RNTRELM e RNTREGDM possuem os parâmetros listados na tabela 5.5, sendo todas elas do tipo Feedforward Backpropagation.

Tabela 5.5: Configuração das redes neurais da aplicação 2.

	Nci	**Nnc**	h	b_m	m_k	**Funções de transferência**
RNTRELM1	1	5	-	-	0.01	Tangente – Linear
RNTRELM3	1	5	-	-	0.01	Tangente – Linear
RNTREGDM1	1	8	0.1	0.1	-	Tangente – Linear
RNTREGDM3	1	8	0.1	0.1	-	Tangente – Linear

As redes RNTRELM1 e RNTREGDM1 foram utilizadas para a determinação da falha das barras 1 e 2, e RNTRELM3 e RNTREGDM3 para a determinação da falha na barra 3 da treliça.

A figura 5.6 mostra a performance de treinamento para a rede treinada com o algoritmo de Levenberg – Marquardt para a determinação da falha da barra 1.

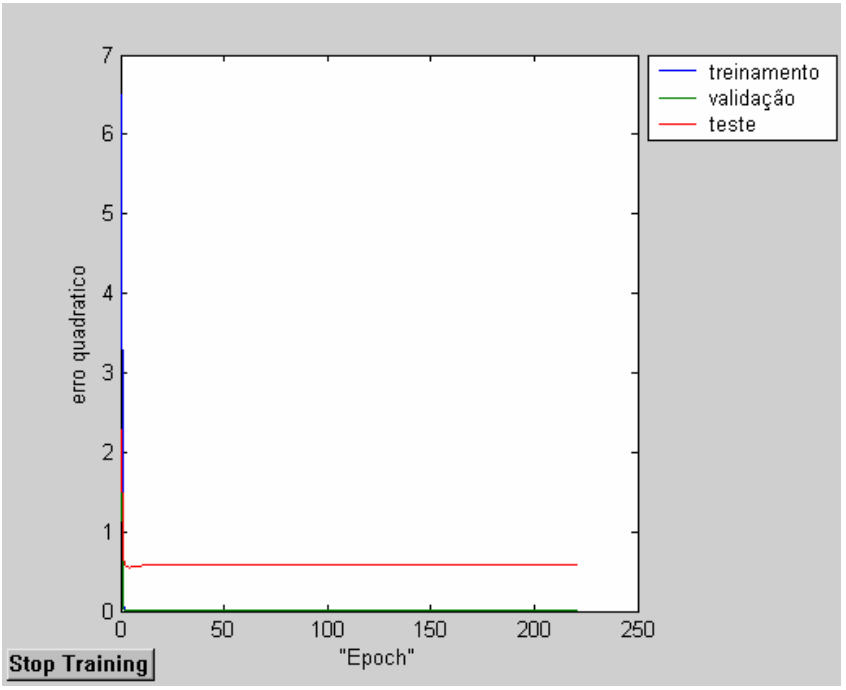

Figura 5.6: Performance de treinamento da rede RNTRELM1.

No gráfico da figura 5.7 revela-se a performance de treinamento para a rede RNTREGDM1.

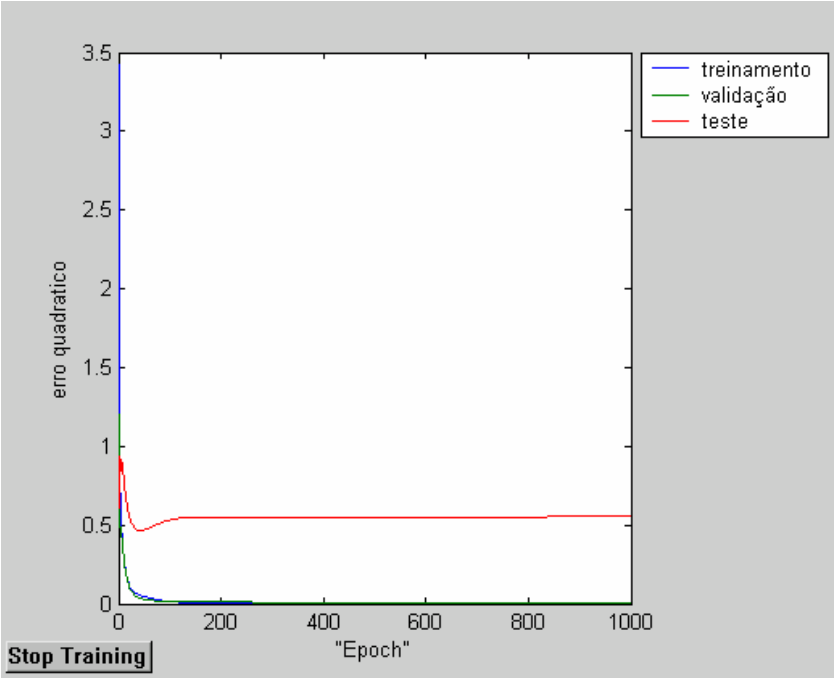

Figura 5.7: Performance de treinamento para a rede RNTREGDM1.

Dos gráficos apresentados nas figuras acima, observa-se um bom desempenho tanto para as redes GDM quanto para as LM, valendo-se ressaltar a rapidez de convergência desta última.

Os resultados da análise da probabilidade de falha da barra 1 estão mostrados na tabela 5.6, que mostra os valores obtidos para 100, 300, 500, 1000 e 3000 simulações, tendo como valor exato da probabilidade de falha ($P_{f\,1}$= 0.039848).

Tabela 5.6: Valores da probabilidade de falha para as barras 1 e 2 da treliça isostática da aplicação 2.

	Número de Simulações				
	100	**300**	**500**	**1000**	**3000**
MCC	0.0300	0.0333	0.0300	0.0430	0.0410
MCRV	0.0404	0.0408	0.0409	0.0408	0.0402
MCRN1	0.0376	0.0425	0.0413	0.0404	0.0392
RNTRELM1	0.0389	0.0391	0.0380	0.0389	0.0396
RNTREGDM1	0.0408	0.0393	0.0407	0.0404	0.0402

O gráfico da figura 5.8 ilustra o erro gerado pelos métodos apresentados na tabela 5.6 para o número de simulaçoes lá mencionados, comprovando os bons resultados fornecidos pelas redes neurais RNTRELM1 e RNTREGDM1, que ficaram abaixo de 1% para um número de 3000 simulações, e apresentando baixos erros ao longo das simulações.

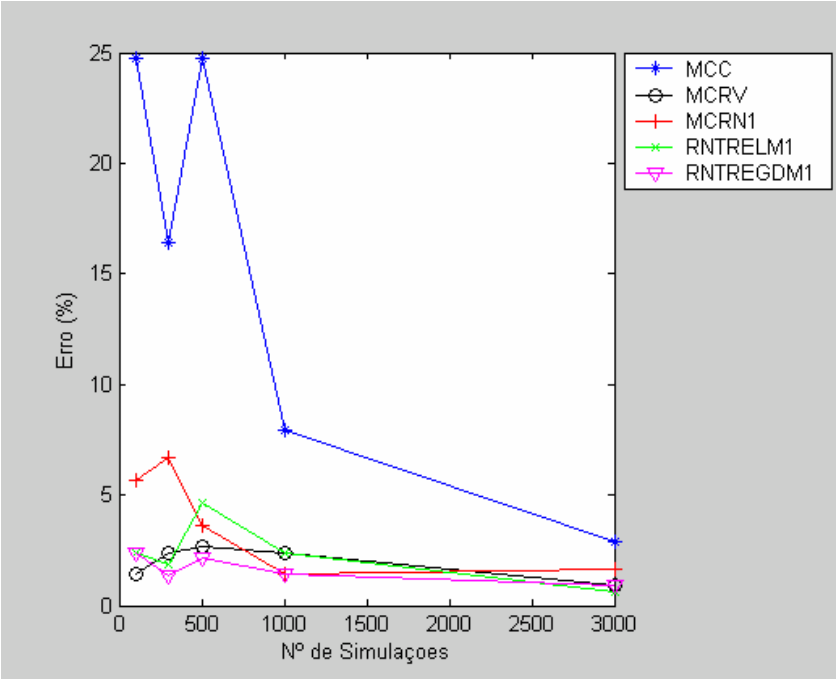

Figura 5.8: Comparação dos erros de P_f das barras 1 e 2 da treliça.

A tabela 5.7 mostra os valores da probabilidade de falha obtidos utilizando as redes RNTRELM e RNTREGDM para a barra 3 da treliça isostática, cujo valor exato é P_{f3} = 3.24e-6.

Tabela 5.7: Valores da probabilidade de falha para a barra 3
da treliça isostática da aplicação 2.

	Nº de Simulações				
	100	**300**	**500**	**1000**	**3000**
RNTRELM3	3.1091e-6	3.2950e-6	3.3013e-6	3.1456e-6	3.1791e-6
RNTREGDM3	2.9551e-6	3.0943e-6	3.0644e-6	3.3064e-6	3.0736e-6

O gráfico apresentado na figura 5.9 mostra a variação do erro para a determinação, pela rede, da probabilidade de falha para a barra 3 da treliça isostática.

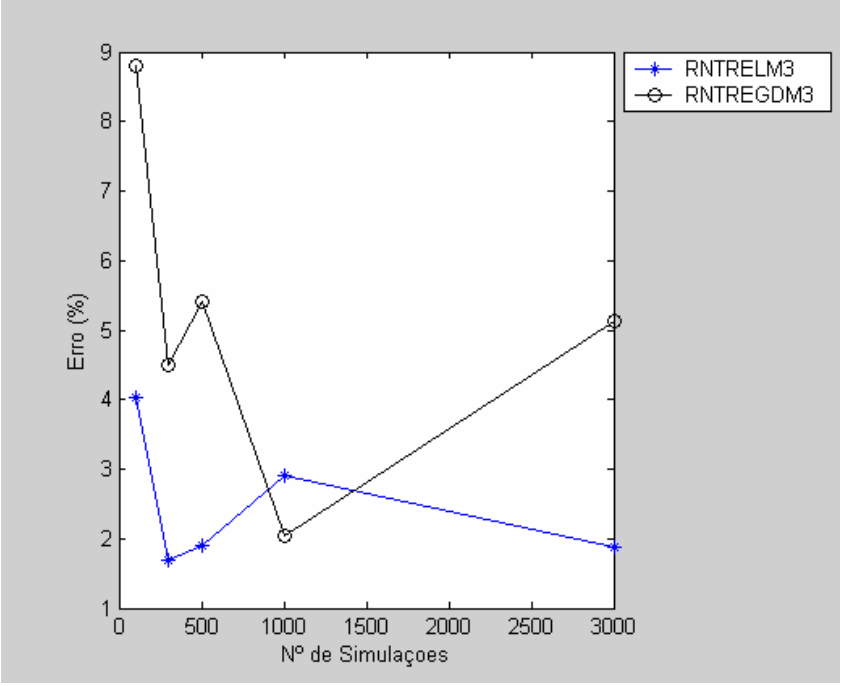

Figura 5.9: Comparação dos erros de P_f da barra 3 da treliça.

Os valores mostrados na tabela 5.7 e na figura 5.9 implicam erros de 1.88% e 5.14% em relação à resposta exata para o problema para 3000 simulações, o que revela a boa performance dos dois algoritmos para a aproximação da resposta. Apesar da rede

RNTREGDM3 atingir um erro maior em relação à outra rede analisada, esta apresentou um erro de aproximadamente 2% para 1000 simulações, o que atesta o bom resultado por ela fornecido.

Os resultados apresentados nas tabelas 5.6 e 5.7 mostram que as redes fornecem valores da probabilidade de falha com erros aceitáveis, atestando a viabilidade de sua aplicação. Esta característica se deve ao bom treinamento e a escolha correta dos parâmetros da rede neural.

5.2.3 APLICAÇÃO 3

Nesta aplicação será analisada uma viga em balanço de seção retangular submetida a um carregamento distribuído e apresentando um comportamento linear elástico, exemplo este analisado por GOMES e AWRUCH (2004).

A falha desta viga ocorrerá para um deslocamento máximo em sua extremidade livre. Este deslocamento não deve exceder o limite de serviço de L/325, onde L é o comprimento da viga. A viga é apresentada na figura 5.10:

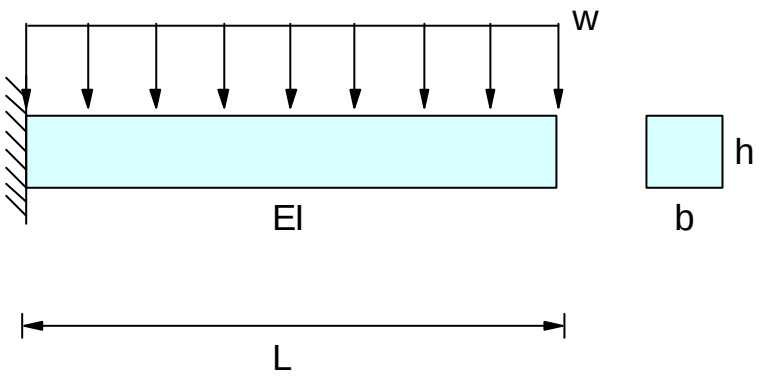

Figura 5.10: Viga em balanço com comportamento linear elástico da aplicação 3.

A função de falha pode ser escrita como mostrado na equação 5.9:

$$G(\mathbf{X}) = \frac{L}{325} - \frac{w.b.L^4}{8.E.I} \tag{5.9}$$

onde w é o carregamento por unidade de área, E é o módulo de elasticidade, I é o momento de inércia, b é a largura e L é o comprimento da viga.

As variáveis L e E são consideradas determinísticas com valores 6 m e $2.6.10^4$ MPa, respectivamente. A função de falha passa então a ser escrita conforme a equação 5.10, em função das variáveis w e h, onde h é a altura da viga (em mm), cujos valores que descrevem as suas características estatísticas são mostrados na tabela 5.8. Estas variáveis são tomadas como não correlacionadas.

$$G(\mathbf{X}) = 18.46154 - 7.476923.10^{10}.\frac{w}{h^3} \tag{5.10}$$

Tabela 5.8: Características das variáveis aleatórias da aplicação 3.

Variável	**Distribuição**	m	**COV**	**Unidade**
w	Normal	1000	0.2	N/m^2
h	Normal	250	0.15	mm

A solução exata para este problema é fornecida por RAJASHEKHAR e ELLINGWOOD (1993) utilizando o método de amostragem por importância, obtendo o valor da probabilidade de falha de $0{,}9607.10^{-2}$ (índice de confiabilidade β = 2.341) para um número de 1000 simulações.

GOMES e AWRUCH (2004) citam diversos métodos utilizados para comparação descritos em seu trabalho, dos quais se escolheu como valores comparativos para este trabalho os seguintes métodos relacionados a seguir:

1- Método de Monte Carlo com “Antithetic Variables” e função de estado limite real;
2- Método de Monte Carlo com “Importance sampling” e “Antithetic Variables” e função de estado limite real (função de amostragem atualizada a cada 150 simulações);

3- Método FORM com gradientes da função de estado limite dados explicitamente;
4- Método de Monte Carlo com "Antithetic variables" e uma rede neural multicamada treinada como função de estado limite;
5- Método de Monte Carlo com "Importance sampling", "Antithetic variables" e uma rede neural multicamada treinada como função de estado limite (função de amostragem atualizada a cada 150 simulações).

Utilizou-se para esta aplicação duas redes neurais, denominadas de RNVBLM e RNVBGDM, cujas características podem ser observadas na tabela 5.9. O conjunto de treinamento para estas redes foi composto de 100 pares.

Tabela 5.9: Configuração das redes neurais da aplicação 3.

	Nci	**Nnc**	h	b_m	m_k	**F. Transferência**
RNVBLM	1	7	-	-	0.1	Tangente – Linear
RNVBGDM	1	8	0.1	0.5	-	Tangente – Linear

Na figura 5.11 é mostrada a performance de treinamento para a rede treinada com o algoritmo de Levenberg – Marquardt.

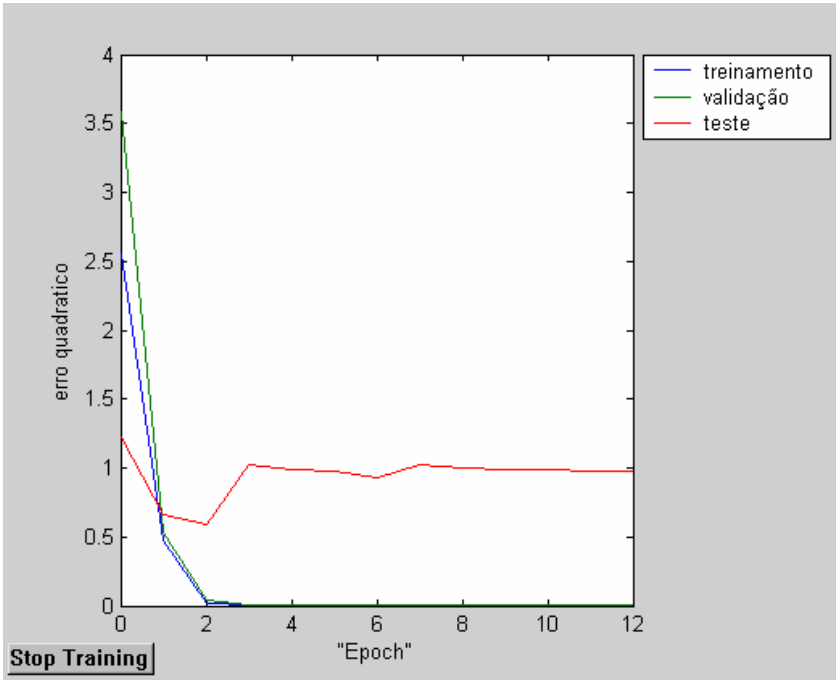

Figura 5.11: Performance de treinamento para a rede RNVBLM.

Na figura 5.12 é mostrada a performance de treinamento para a rede treinada com o algoritmo do gradiente descendente.

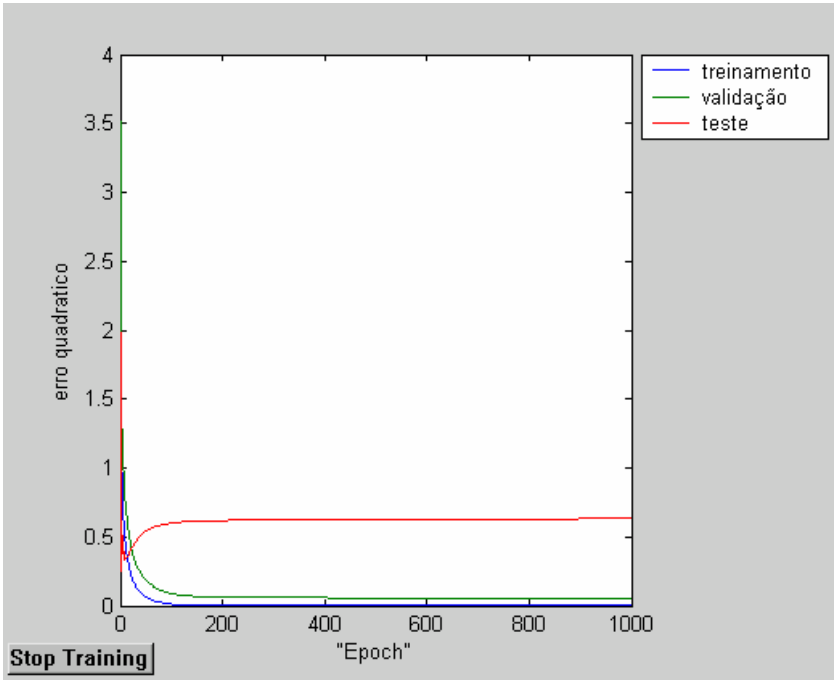

Figura 5.12: Performance de treinamento para a rede RNVBGDM.

Na tabela 5.10 estão apresentados os resultados da simulação das redes utilizadas para a determinação da falha, encontrando-se também nesta tabela o valor exato e os métodos descritos anteriormente citados por GOMES e AWRUCH (2004).

Tabela 5.10: Valores da probabilidade de falha para a aplicação 3.

	P_f	nsim
RAJASHEKHAR (1993)	0.009607	1000
1	0.009690	100000
2	0.009642	17850
3	0.009789	33
4	0.009163	50000
5	0.009394	17850
MCC	0.009280	25000
MCCE	0.009506	5000
RNVBLM	0.009313	5000
RNVBGDM	0.008877	10000

Uma análise de erro é efetuada para os dados fornecidos na tabela 5.10, análise esta mostrada na figura 5.13, onde se comparam os resultados obtidos com a resposta encontrada por RAJASHEKHAR e ELLINGWOOD (1993).

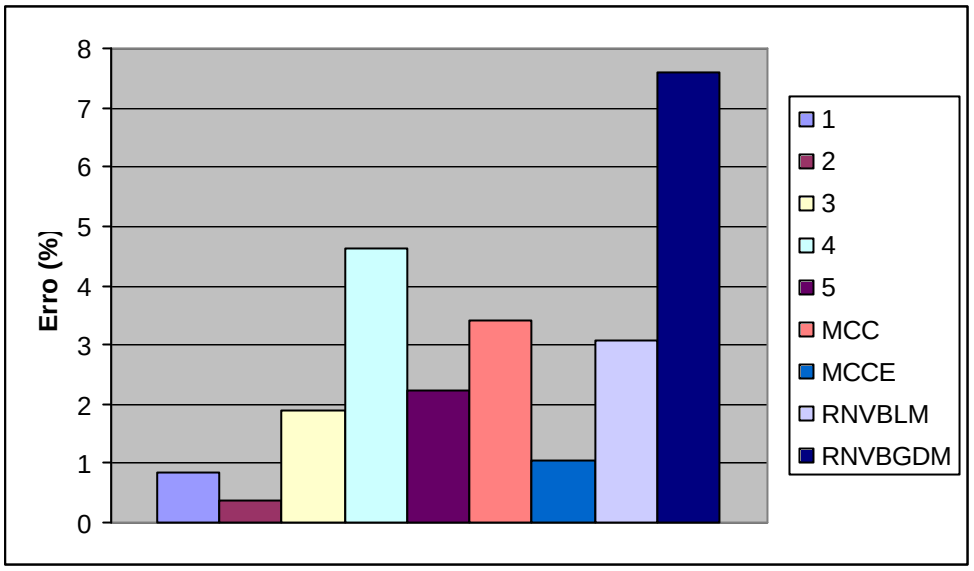

Figura 5.13: Comparação dos erros de P_f da viga em balanço.

Dos resultados mostrados na tabela 5.10 e na figura 5.13, observa-se a boa aproximação fornecida pela rede RNVBLM a um tempo computacional baixo, apresentando para um número de 5000 simulações um erro de 3,06%. Já a rede RNVBGDM apresentou um erro 7.6% para um número de 10000 simulações, que se mostra também um resultado favorável.

O método de Monte Carlo com a técnica de Esperança Condicionada apresentou uma melhor performance que as redes neurais configuradas, mostrando um erro 1,05% para um total de 5000 simulações realizadas.

Já o método de Monte Carlo, apresentou um alto custo computacional para alcançar uma resposta com um erro aceitável (erro de 3,4% para 25000 simulações realizadas). Os bons resultados apresentados pela rede em comparação aos outros métodos citados tornam-se indicativos da qualidade advinda da aplicação desta rede em conjunto com método de simulação de Monte Carlo.

5.3 APLICAÇÕES DO ALGORITMO A2

5.3.1 APLICAÇÃO 4

A aplicação 4 apresenta uma viga biapoiada, representada na figura 5.14, onde são consideradas como variáveis aleatórias básicas a carga concentrada aplicada P e o módulo de elasticidade E.

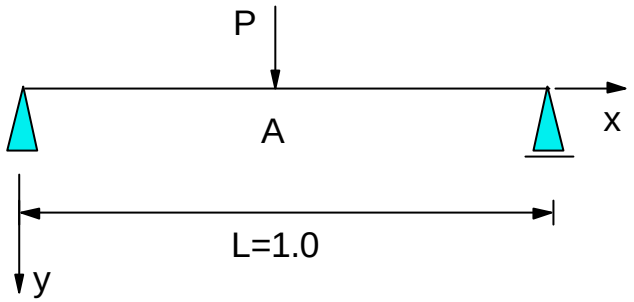

Figura 5.14: Viga isostática com carga concentrada da aplicação 4.

Os dados referentes as variáveis aleatórias estão mostrados na tabela 5.11. O critério de falha está relacionado com o deslocamento do ponto A segundo o eixo y, tomando-se como valor limite d = 4,8 cm. A função de falha pode ser representada então pela equação 5.11:

$$G(\mathbf{X}) = 0.048 - d(\mathbf{X}) \tag{5.11}$$

onde $d(\mathbf{X})$ é o deslocamento vertical do ponto A e $\mathbf{X}$ é o vetor contendo as variáveis aleatórias (P,E).

Tabela 5.11: Características das variáveis aleatórias da aplicação 4.

Variável	**Distribuição**	m	s	**COV**	l	x
$P(X_1)$	Lognormal	0.16667	$1.67x10^{-2}$	0.1	-1.7967	0.0998
$E(X_2)$	Lognormal	2400	240	0.1	7.7782	0.0998

O conjunto de treinamento para o referido exemplo consta de 80 pares. Para este conjunto é realizado um pré-processamento dos dados, e em seguida, após o

treinamento, é efetuado o pós-processamento para a retomada dos valores na escala original.

Na tabela 5.12 estão apresentadas as configurações das redes neurais analisadas nesta aplicação, denominadas de RNVCCLM e RNVCCGDM.

Tabela 5.12: Configuração das redes neurais da aplicação 4.

	Nci	**Nnc**	h	b_m	m_k	**F. Transferência**
RNVCCLM	1	3	-	-	0.1	Tangente – Linear
RNVCCGDM	1	6	0.1	0.1	-	Tangente – Linear

Utilizou-se rotinas auxiliares para a determinação da probabilidade de falha, visto que a rede é treinada para fornecer apenas os deslocamentos do ponto A. As figuras 5.15 e 5.16 mostram as performances de treinamento das redes RNVCCLM e RNVCCGDM, respectivamente.

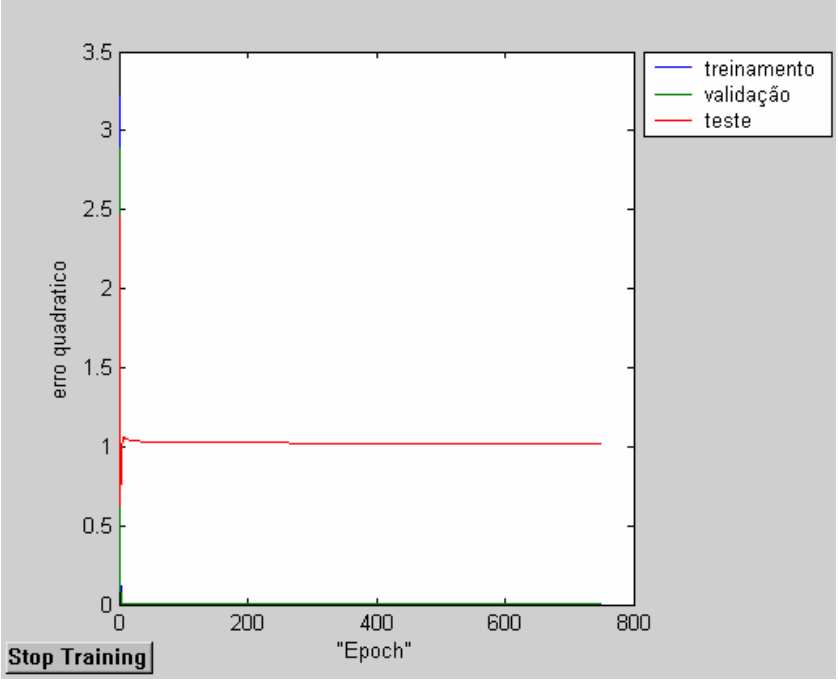

Figura 5.15: Performance de treinamento da rede RNVCCLM.

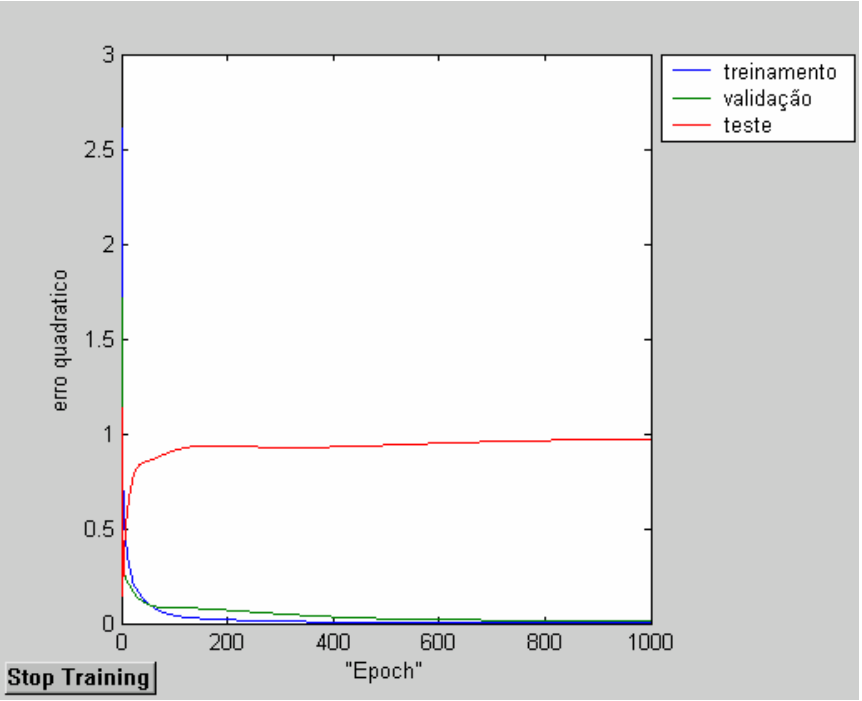

Figura 5.16: Performance de treinamento da rede RNVCCGDM.

As redes utilizadas por SARAIVA (1997), cujos resultados servirão de base para comparação, possuem as seguintes características:

- Rede neural do tipo Feedforward Backpropagation;
- Número de camadas ocultas = 1;
- Número de neurônios na camada oculta = 6;
- Taxa de aprendizagem $\eta = 0.1$;
- Fator de "Momentum" $\beta_m = 0.1$;
- Funções de transferência = Sigmoidal.

Estas redes foram denominadas de MCRN2 e MCRN2*, ambas possuindo as mesmas características acima citadas, e tem como principal diferencial a forma de calcular a probabilidade de falha: MCRN2 determina a probabilidade de falha utilizando a técnica de redução de variância Amostragem por Importância, enquanto MCRN2* gera os valores dos deslocamentos e calcula a probabilidade de falha através do método de Monte Carlo Clássico. Determinou-se também os valores da probabilidade de falha

para o método de Monte Carlo Clássico sem a utilização das redes neurais, denominado de MCC.

A tabela 5.13 apresenta a comparação dos resultados obtidos para a referida aplicação. A utilização do método de Monte Carlo Clássico visa apresentar uma estimativa do que seria o valor exato da probabilidade de falha para servir de comparação para os resultados obtidos pelas redes estudadas.

Tabela 5.13: Valores da probabilidade de falha para a aplicação 4.

	Número de Simulações				
	500	**1000**	**3000**	**5000**	**10000**
MCRN2	0.0200	0.0151	0.0126	0.0108	0.0101
MCRN2*	0.0040	0.0040	0.0143	0.0110	0.0106
MCC	0.0040	0.0050	0.0090	0.0098	0.0092
RNVCCGDM	0.0080	0.0100	0.0107	0.0108	0.0106
RNVCCLM	0.0120	0.0100	0.0090	0.0102	0.0095

Dos dados obtidos na tabela 5.13, verifica-se que a rede analisada foi capaz de produzir valores da probabilidade de falha com aproximação razoáveis, acompanhando a tendência do método de Monte carlo, que seria o valor considerado exato para o problema.

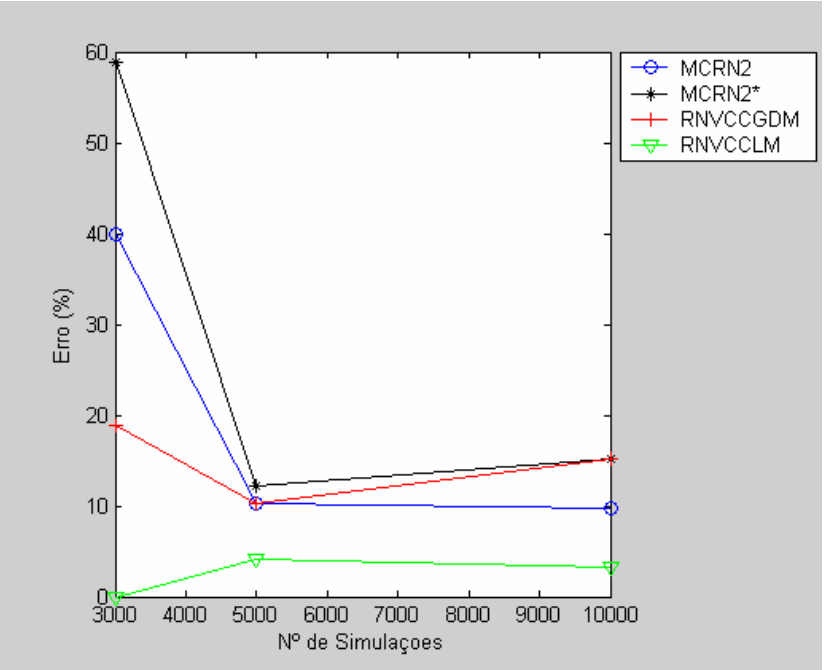

Figura 5.17: Comparação dos erros de P_f da viga biapoiada.

O gráfico da figura 5.17 apresenta uma comparação dos erros obtidos com a simulação das redes empregadas neste exemplo, tomando como base o valor obtido para o método de Monte Carlo. Nota-se que a rede treinada com o algoritmo de Levenberg – Marquardt apresentou bons resultados, chegando a atingir um erro de 3.26% para 10000 simulações. Já a rede treinada com o algoritmo do gradiente descendente, apesar de atingir um erro superior a 10%, seus resultados se mostram comparáveis com os resultados obtidos por SARAIVA (1997).

5.3.2 APLICAÇÃO 5

Nesta aplicação analisa-se um pórtico elástico plano, mostrado na figura 5.18, o qual possui as dimensões e está submetido aos carregamentos lá apresentados.

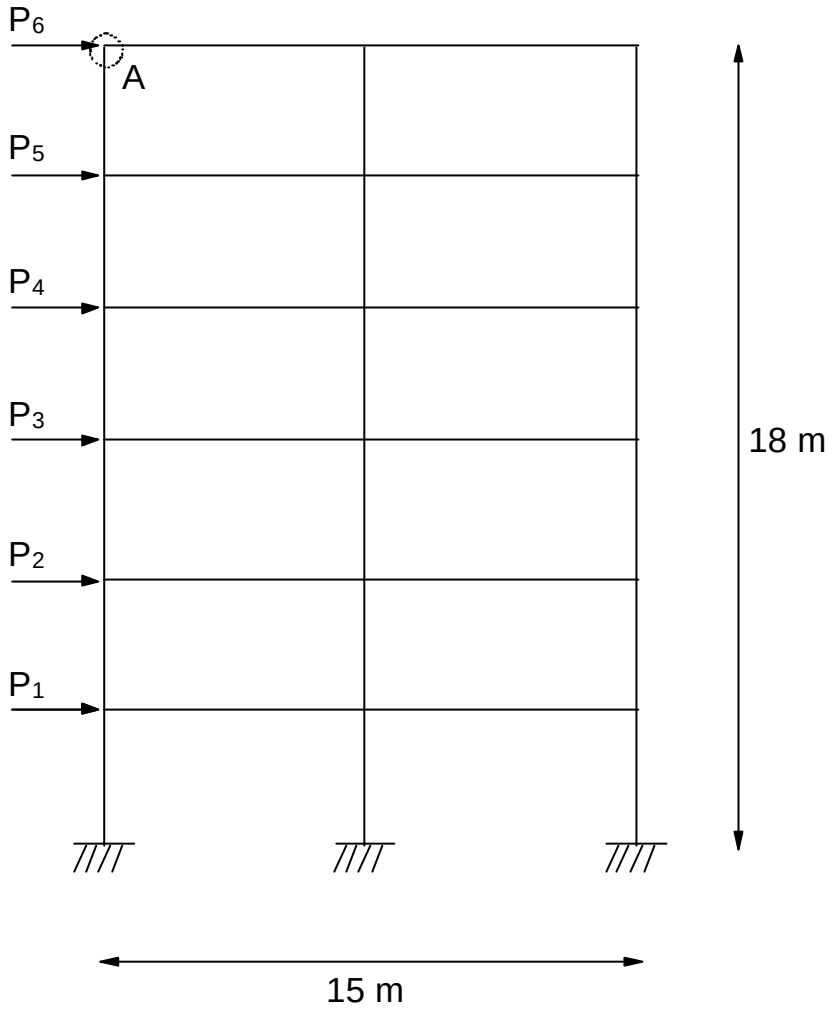

Figura 5.18: Pórtico elástico plano da aplicação 5.

A função de falha é definida através do deslocamento máximo do ponto A, definido na figura 5.18, tendo como valor limite 12 cm na direção horizontal, podendo ser expressa através da equação 5.12:

$$G(\mathbf{X}) = 0.12 - d_A(\mathbf{X}). \tag{5.12}$$

As variáveis contidas no vetor $\mathbf{X}$ são os módulos de elasticidade das colunas e das vigas, E_C e E_V, os momentos de inércia das colunas e das vigas, I_C e I_V, e as cargas aplicadas P_1, P_2, P_3, P_4, P_5 e P_6. As características estatísticas destas variáveis são mostradas na tabela 5.14:

Tabela 5.14: Características das variáveis aleatórias
da aplicação 5 (kN e m).

Variável	Distribuição	m	s	l	e
E_C	Lognormal	1.96e7	1.96e6	16.7861	0.0998
E_V	Lognormal	1.96e7	1.96e6	16.7861	0.0998
I_C	Lognormal	1.0e-3	1.0e-4	-6.9127	0.0998
I_V	Lognormal	1.5e-3	1.5e-4	-6.5073	0.0998
P_1	Normal	24.50	6.125	-	-
P_2	Normal	27.44	6.860	-	-
P_3	Normal	28.42	7.105	-	-
P_4	Normal	29.40	7.350	-	-
P_5	Normal	30.38	7.595	-	-
P_6	Normal	31.36	7.840	-	-

Os parâmetros λ e ε, mostrados na tabela 5.13, representam os parâmetros da distribuição lognormal.

HURTADO e ALVAREZ (2001) fornecem a probabilidade de falha para o problema, cujo valor é de 0.00545, obtida para um número de 350000 simulações utilizando o método de Monte Carlo.

Este pórtico foi analisado por HURTADO e ALVAREZ (2001), que obtiveram resultados para a aplicação de redes neurais na determinação da probabilidade de falha, fazendo uso de redes neurais MultiLayer Perceptron (MLP), segundo os métodos do gradiente descendente com fator de momentum e taxa de aprendizagem variável (BPX) e método de Gauss – Newton Levenberg – Marquardt (GNLM) , e utilizando também redes com função de base radial (RBF).

A análise efetuada nesta aplicação versou sobre a rede RNPLM, cujos parâmetros estão listados a seguir:

- Número de camadas ocultas: 1;
- Número de neurônios na camada oculta: 18;
- Funções de Transferência: Tangente Hiperbólica – Linear;
- Constante μ_K: 0.01.

Compôs-se de 175 pares o conjunto de treinamento para a rede utilizada na aplicação. O gráfico apresentado na figura 5.19 mostra a performance de treinamento para a rede RNPLM.

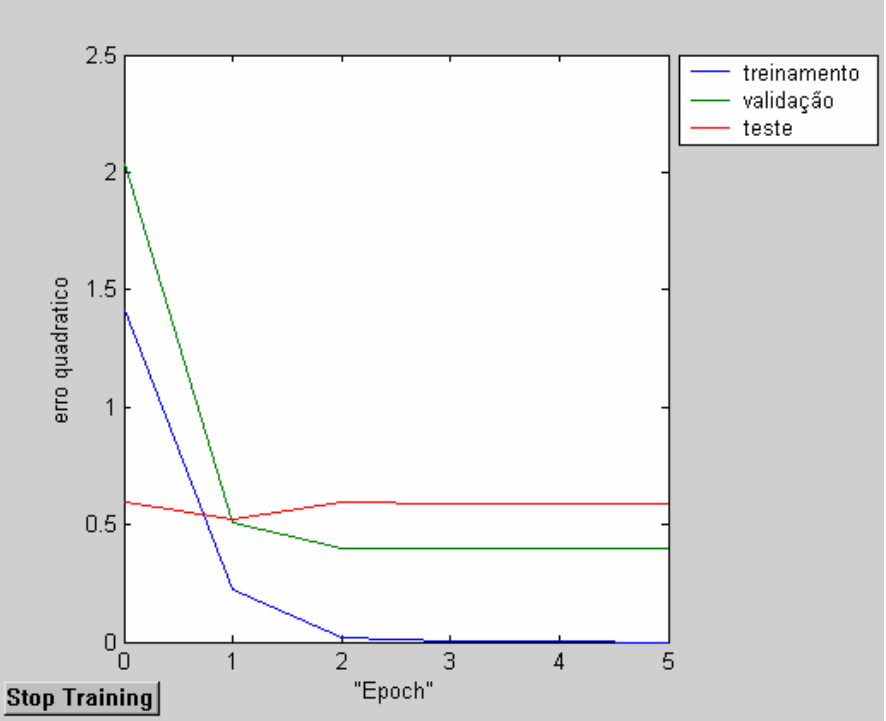

Figura 5.19: Performance de treinamento para a rede RNPLM.

A tabela 5.15 fornece os resultados da aplicação da rede estudada, assim como os resultados fornecidos por HURTADO e ALVAREZ (2001). As redes expostas por HURTADO e ALVAREZ (2001) visam minimizar a soma dos erros quadráticos das saídas (SSE) e utilizam dois processos de amostragem: geração de números aleatórios da função de distribuição das variáveis básicas (CDF) e distribuição uniforme das variáveis básicas com k desvios-padrão sobre a média (Unk).

Tabela 5.15: Resultados obtidos na aplicação 5.

	Estratégia	P_f
Exata	-	0.00545
SSE - BPX	CDF	0.00500
	UN3	0.00514
	UN5	0.00967
SSE - GNLM	CDF	0.00543
	UN3	0.00539
	UN5	0.00592
RBF	CDF	0.00515
	UN3	0.00530
	UN5	0.00572
RNPLM	-	0.00500

A figura 5.20 revela os erros obtidos da aplicação dos métodos propostos na tabela 5.15, não se computando o erro para o método BPX – UN5, cujo valor do erro superou 73%.

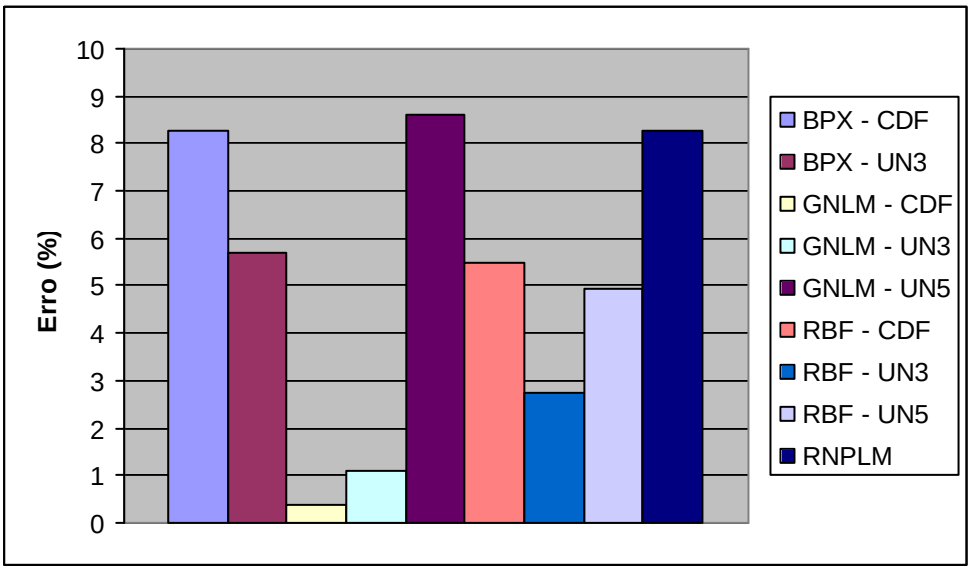

Figura 5.20: Comparação dos erros de P_f do pórtico elástico plano.

Verifica-se que os métodos mostrados na figura 5.22 se revelam bastante razoáveis, dando-se destaque ao método GNLM – CDF, cujo erro foi inferior a 0.5%. Já para a rede neural analisada nesta aplicação, embora o erro tenha sido um pouco elevado, superior a 8%, a resposta fornecida permite que esta possa ser aplicada na resolução do problema.

5.3.3 APLICAÇÃO 6

Nesta aplicação estudar-se-á a confiabilidade de uma coluna bidimensional treliçada, apresentada na figura 5.21, a qual é composta de 20 nós e 45 elementos. A referida coluna foi estudada por SAGRILO (1994), Liu e Der Kiureghian (1989) e HOLM (1990).

As características geométricas das barras da treliça estão apresentadas na tabela 5.16. As variáveis aleatórias básicas do problema são os módulos de elasticidade E_1 e E_2 e as cargas aplicadas R_1 e R_2. As variáveis E_1 e E_2 são correlacionadas com valor ?=0.30. As características estatísticas das variáveis do problema estão mostradas na tabela 5.17.

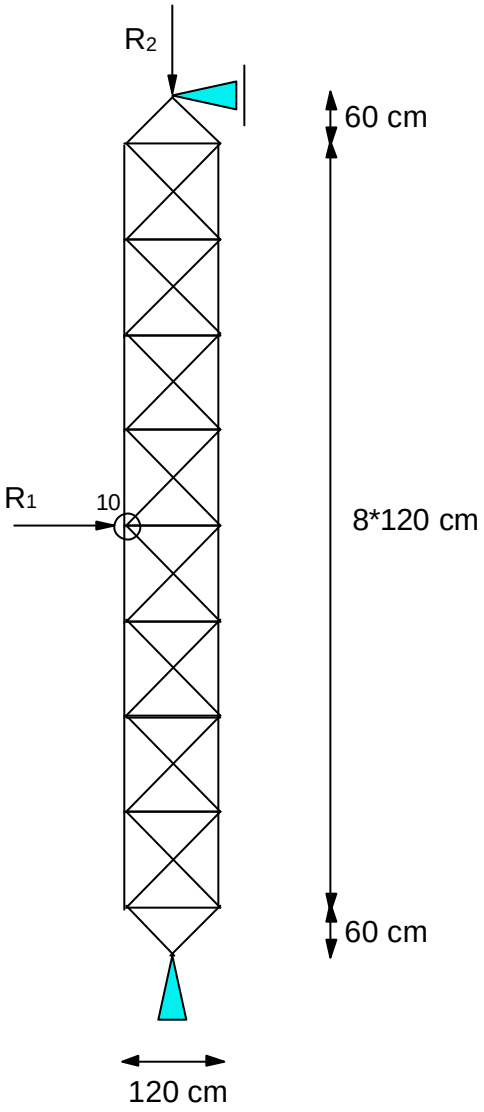

Figura 5.21: Coluna bidimensional treliçada da aplicação 6.

Tabela 5.16: Características geométricas das barras da coluna da aplicação 6.

Barras	**Área (cm^2)**	**Módulo de Elasticidade**
Horizontais	0.938	E_2
Verticais	1.59	E_1
Inclinadas [1]	0.938	E_2
Inclinadas [2]	1.59	E_1
[1] Barras que não chegam nos apoios [2] Barras que chegam nos apoios		

Tabela 5.17: Características das variáveis aleatórias da aplicação 6.

Variável	Distribuição	μ	COV	?	?
E_1	Lognormal	3000	0.08	8.0032	0.0799
E_2	Lognormal	3000	0.08	8.0032	0.0799
R_1	Lognormal	20	0.1	2.9908	0.0998
R_2	Lognormal	1500	0.1	7.3082	0.0998
Unidades em kN e cm					

A falha para o problema é caracterizada pelo deslocamento horizontal do nó 10, tomando-se como valor limite u_{10} = 30 cm, sendo sua função de falha expressa pela equação 5.13:

$$G(\mathbf{X}) = u_{10} - d(\mathbf{X}) \tag{5.13}$$

onde X é o vetor contendo as variáveis aleatórias do problema (E_1,E_2,F_1,F_2).

Para a determinação do deslocamento se utilizou o programa **SAP 2000** , em que se realizou uma análise não linear para a coluna.

A aplicação da rede neural versou sobre duas configurações, RNCOLLM e RNCOLGDM, cujos parâmetros estão listados na tabela 5.18. O conjunto de treinamento para estas redes neurais constou de 80 pares.

Tabela 5.18: Configuração das redes neurais da aplicação 6.

	Nci	Nnc	?	$ß_m$	$μ_k$	F. Transferência
RNCOLLM	1	6	-	-	0.01	Tangente - Linear
RNCOLGDM	1	12	0.1	0.1	-	Tangente - Linear

Nos gráficos das figuras 5.22 e 5.23 são apresentadas as performances de treinamento para as redes RNCOLLM e RNCOLGDM, respectivamente.

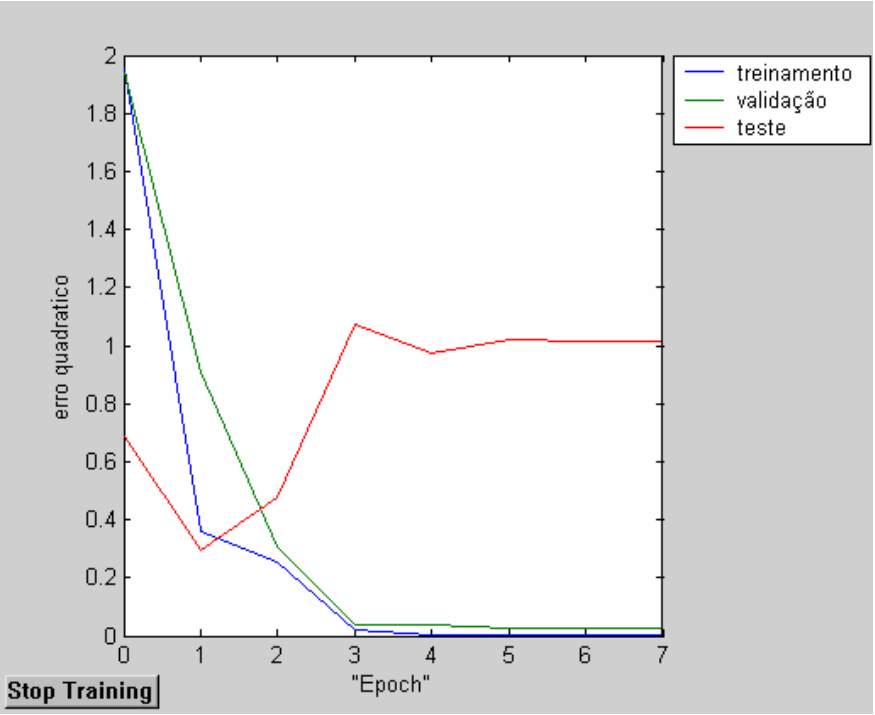

Figura 5.22: Performance de treinamento da rede RNCOLLM.

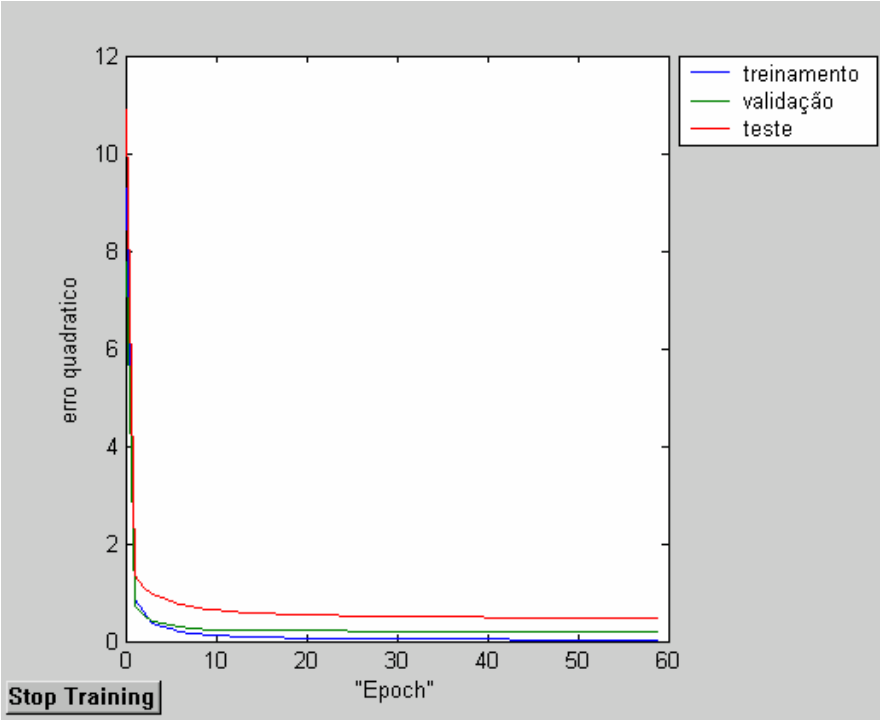

Figura 5.23: Performance de treinamento da rede RNCOLGDM.

Na tabela 5.19 estão apresentados os valores obtidos com a simulação das redes empregadas neste estudo. Como comparação se utilizou apenas os valores obtidos por SAGRILO (1994), visto que HOLM (1990) e LIU e Der KIUREGHIAN (1991) só forneceram informações sobre o ponto de projeto, que não foi objeto deste estudo.

SAGRILO (1994) analisou o problema utilizando os métodos FORM e SORM, fazendo uso também de aproximações lineares (ALR) e quadráticas (ANLR) para a superfície de resposta.

Os valores produzidos pelas redes analisadas, constantes na tabela 5.19, foram obtidos com um número de 5000 simulações.

Tabela 5.19: Resultados obtidos da aplicação 6.

	P_f
FORM	6.47e-3
FORM (ALR)	6.47e-3
FORM (ANLR)	6.47e-3
SORM	5.93e-3
SORM (ANLR)	6.37e-3
RNCOLLM	6.40e-3
RNCOLGDM	6.20e-3

Verifica-se uma boa concordância dos valores obtidos pelas redes neurais estudadas com os valores obtidos por SAGRILO (1994), tanto para o método FORM quanto para SORM.

5.4 APLICAÇÕES DO ALGORITMO A3

5.4.1 APLICAÇÃO 7

Na aplicação 7, pretende-se analisar a confiabilidade de uma treliça com as dimensões e cargas mostradas na figura 5.24. Todas as barras da treliça possuem área de 1000 mm^2.

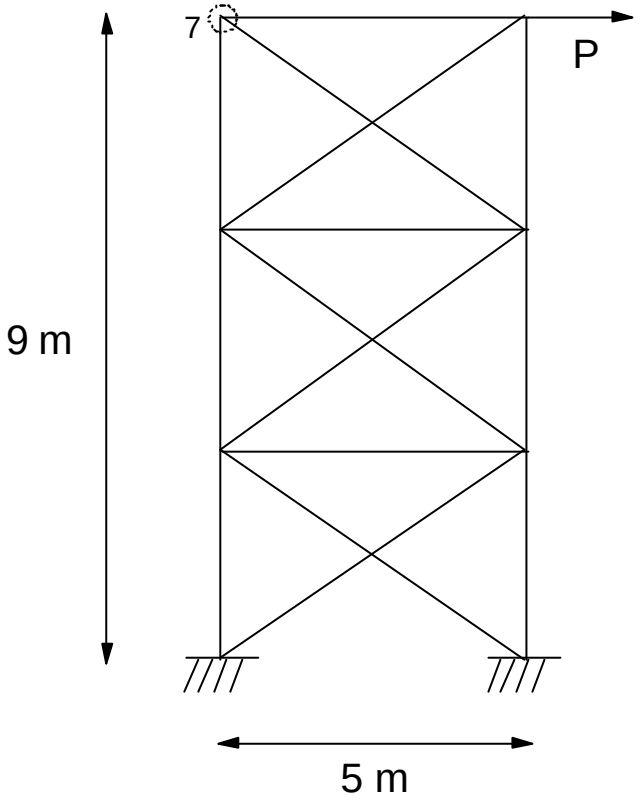

Figura 5.24: Treliça plana da aplicação 7.

A função de falha para o problema está descrita na equação 5.14, a qual é caracterizada pelo deslocamento máximo de 70 mm para o nó 7, apresentado na figura 5.24:

$$G(\mathbf{X}) = 70.0 - d_7(\mathbf{X}) \tag{5.14}$$

onde $d_7(\mathbf{X})$ representa o deslocamento do nó 7 calculado em função das variáveis aleatórias.

Considera-se como variáveis aleatórias a carga aplicada e o módulo de elasticidade E, expresso em kN/mm^2, cujas características estatísticas desta ultima variável podem ser observadas na tabela 5.20.

Tabela 5.20: Característica da variável aleatória da aplicação 7.

	m	s	COV	Distribuição
E	200000	60000	0.3	Normal

Para a variável P, expressa em kN, considera-se uma distribuição uniforme no intervalo [250 – 350 kN], variável que é tomada como parâmetro de controle para a aplicação.

O conjunto de treinamento para a rede neural configurada para este problema foi gerado no programa **FERUM** utilizando o método de Amostragem por Importância, no qual se variou o valor da carga P e determinou-se a respectiva probabilidade de falha. Este conjunto constou de 100 pares.

Foi configurada uma única rede, denominada de RNTLM, para a análise da confiabilidade da treliça plana, cujos parâmetros pertinentes a esta são listados abaixo:

- Número de camadas intermediárias = 1;
- Número de neurônios na camada intermediária = 5;
- Funções de Transferência = tangente hiperbólica – linear;
- Constante $\mu_K = 0.1$.

A performance de treinamento para a rede RNTLM é apresentada no gráfico da figura 5.25.

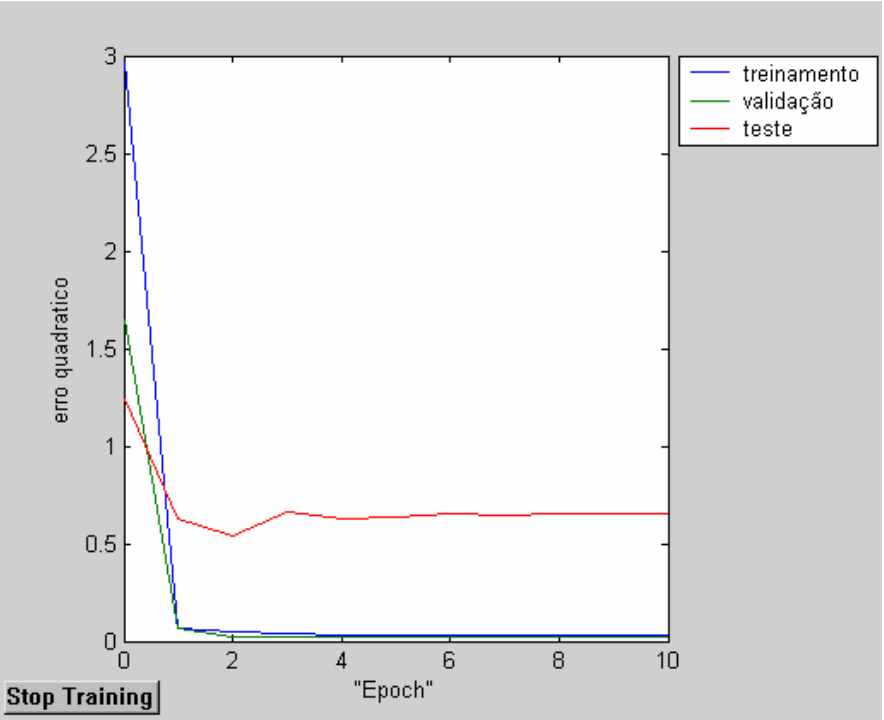

Figura 5.25: Performance de treinamento para a rede RNTLM.

Para a comparação dos resultados, utilizou-se o próprio programa **FERUM**, fazendo uso do método FORM e do método de Amostragem por Importância.

Na tabela 5.21 estão mostrados os resultados obtidos pela rede neural RNTLM para 3 valores de carga escolhidos dentro do intervalo de análise.

Tabela 5.21: Resultados obtidos na aplicação 7.

	P = 322.56 kN			**P = 268.4 kN**			**P = 347.5 kN**		
	P_f	**nsim**	**Tempo**	**P_f**	**nsim**	**Tempo**	**P_f**	**nsim**	**Tempo**
FORM	9.746e-3	-	5.797	1.705e-3	-	5.656	2.181e-2	-	5.765
MCIS	4.299e-2	10000	-	1.119e-2	10000	-	7.836e-2	10000	-
RNTLM	4.026e-2	-	7.515	1.050e-2	-	10.484	8.350e-2	-	3.547

Observa-se nos resultados obtidos com a aplicação da rede neural que esta apresentou erros de 6.35%, 6.2% e 6.56%, respectivamente, em relação ao método de Amostragem por Importância para os casos descritos na tabela 5.21.

Em relação ao tempo computacional, a rede obteve um custo um pouco maior que para o método FORM, método este que não apresentou bons resultados, sempre destoando dos demais. Já o método de amostragem por importância teve sua resposta obtida, para todos os casos, a um número de simulações muito elevado, tornando a rede passível de ser aplicada em sua substituição devido à qualidade de sua resposta.

5.4.2 APLICAÇÃO 8

Nesta aplicação será analisado um pórtico com material de propriedades lineares, apresentado na figura 5.26, estando suas dimensões lá especificadas. As variáveis envolvidas nesta análise são o módulo de elasticidade E, o momento de inércia I e a área da seção transversal A das barras do pórtico, cujas características estatísticas são mostradas na tabela 5.22. Também considera-se como variável aleatória a carga aplicada P.

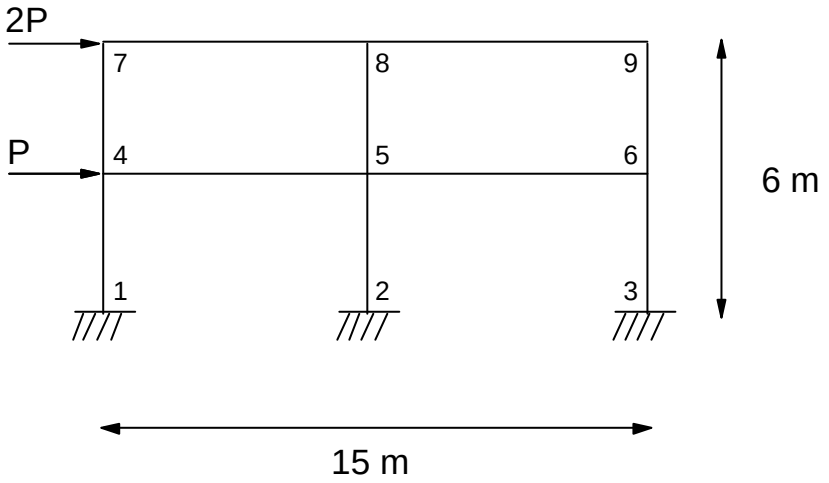

Figura 5.26: Pórtico plano com características lineares da aplicação 8.

Tabela 5.22: Características das variáveis aleatórias da aplicação 8.

	Média	COV	Distribuição
E	200000 (kN/mm^2)	0.1	Normal
I	0.5e9 (mm$^{4)}$	0.1	Normal
A	5000 (mm^2)	0.1	Normal

A falha da estrutura ocorrerá para um deslocamento máximo de 45 mm para o nó 7, representado na figura 5.26, sendo sua função de performance definida na equação 5.15:

$$G(\mathbf{X}) = 45 - d_7(\mathbf{X}) \tag{5.15}$$

onde $d_7(\mathbf{X})$ representa o deslocamento do nó 7 calculado em função das variáveis aleatórias.

Para a aplicação da rede, tomou-se como parâmetro de controle o valor médio da carga P aplicada. Esta variável tem um coeficiente de variação de 0.1 e distribuição Normal. Propõe-se dar uma variação uniforme em seu valor médio no intervalo [270 – 350 kN].

Para a geração do conjunto de treinamento, fez-se uso do programa **FERUM** utilizando o método de Amostragem por Importância para a determinação da probabilidade de falha. O conjunto de treinamento para esta aplicação constou de 100 pares.

Foi analisada uma configuração de rede, denominadas de RNPLM, sendo seus parâmetros listados abaixo:

- Número de camadas intermediárias = 1;
- Número de neurônios na camada intermediária = 3;
- Funções de Transferência = Tangente Hiperbólica – Linear;
- Constante μ_k = 0.038.

A performance de treinamento da rede RNPLM é apresentada no gráfico da figura 5.27.

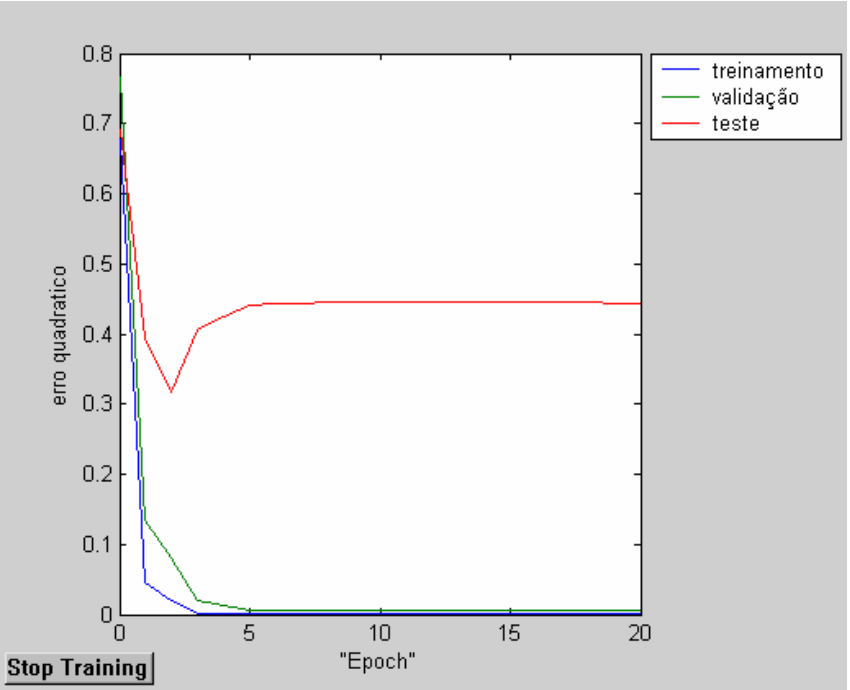

Figura 5.27: Performance de treinamento para a rede RNPLM.

Os resultados obtidos com a aplicação das redes são apresentados na tabela 5.23. Nesta tabela são expostos os valores obtidos pela rede neural e pelo programa **FERUM**, utilizando o método FORM, o método SORM e o método de Amostragem por Importância. Também é exposta uma comparação com o custo computacional requerido por cada um destes métodos analisados.

Para o método SORM são avaliadas três estratégias de determinação da probabilidade de falha. Utilizou-se o método da curvatura ajustada (Curvature Fitted), sendo fornecido os resultados pela fórmula de Breitung, pela fórmula de Breitung melhorada e pela integral exata Tvedt, denominados de SORM1, SORM2 e SORM3, respectivamente.

Também é fornecido na tabela 5.23 o tempo computacional utilizado na análise para os métodos FORM e SORM e para a simulação da rede neural, e para o método de

Amostragem por Importância é apresentado o número de simulações necessário à determinação da falha. Nos caso das redes não está computado o tempo gasto no treinamento.

Tabela 5.23: Resultados obtidos na aplicação 8.

	P = 273.9635 kN			**P = 304.8 kN**			**P = 336.9 kN**		
	P_f	**nsim**	**Tempo**	**P_f**	**nsim**	**Tempo**	**P_f**	**nsim**	**Tempo**
FORM	5.893e-7	-	5.828	1.078e-4	-	0.485	4.034e-3	-	5.672
SORM1	9.231e-7	-	10.656	1.477e-4	-	10.438	4.966e-3	-	10.704
SORM2	9.407e-7			1.509e-4			5.092e-3		
SORM3	9.351e-7			1.502e-4			5.073e-3		
MCIS	9.499e-7	2641	-	1.559e-4	1627	-	4.850e-3	1164	-
RNPLM	1.074e-6	-	2.953	1.465-4	-	2.239	5.099e-3	-	1.656

Atesta-se a qualidade dos resultados fornecidos pela rede, apresentados na tabela 5.23, nos quais se obteve um baixo custo computacional. Apesar de apresentar um erro de pouco menos de 12%, o primeiro caso analisado na tabela permite que se possa quantificar a falha através do valor fornecido pela rede neural.

Com relação ao custo computacional, observa-se que a rede fornece seus resultados a um baixo custo computacional, assim como o método FORM, mas este apresenta solução que se distancia dos resultados obtidos pelos demais métodos.

Já para o método de Amostragem por Importância, abordado nesta aplicação, a solução é alcançada para um número de simulações relativamente pequeno, que é um dos objetivos de sua aplicação. O método SORM apresenta uma convergência para as três formas de determinar a falha a um tempo computacional semelhante ao requerido pela rede para cada uma destas formas.

Pode-se notar da aplicação, que é possível a substituição simultânea das etapas de análise estrutural e da determinação da probabilidade de falha, mas requer cuidados

no que concerne à determinação do intervalo de análise e no tamanho do conjunto de treinamento, pois a escolha destes pode afetar na resposta alcançada pela rede neural.

Capítulo 6

CONCLUSÕES E SUGESTÕES DE PESQUISA

As aplicações analisadas revelam os bons resultados originados da utilização das redes neurais artificiais tanto em conjunto com o método de simulação de Monte Carlo como também na substituição de todo o processo de análise. O fato de os métodos de análise de confiabilidade apresentarem algumas restrições no processo de solução ou exigirem uma tempo computacional demasiado para a convergência à uma solução torna os resultados obtidos da solução com as redes neurais uma boa alternativa de utilização.

Um dos pontos principais deste trabalho foi a abordagem dos conceitos fundamentais sobre as redes neurais artificiais e como se processa o seu treinamento, dando enfoque aos algoritmos de Levenberg – Marquardt e do gradiente descendente com momentum, aplicados para a atualização dos parâmetros ajustáveis da rede em conjunto com o algoritmo backpropagation.

Pode-se destacar da aplicação das redes neurais algumas características com relação ao seu desempenho, as quais são:

- A rápida convergência de treinamento, permitindo que caso a aproximação fornecida pela rede não tenha sido satisfatória, a inserção de novos pontos em seu conjunto de treinamento ou a mudança de algum de seus parâmetros possa ser realizada para um novo treinamento da rede neural;

- A capacidade da rede neural em fazer aproximações de funções de ordem qualquer e a generalização que esta faz para estas funções;
- Esta aplicação tende a reduzir o esforço computacional exigido na análise requerido para o método de Monte Carlo.

Uma questão de fundamental interesse na utilização das redes neurais é quanto a representatividade do conjunto de treinamento, devendo este abranger de forma significativa os casos em que a falha acontece ou não dentro do domínio do problema, necessitando muito cuidado na escolha destes valores.

A utilização das redes na análise de confiabilidade foi efetuada segundo algumas estratégias, definidas no capítulo 4, no qual se definiu como e a que partes do processo de simulação a rede foi inserida para a determinação da probabilidade de falha.

Para se poder realizar a aplicação das redes neurais foi necessário a escolha da sua arquitetura ideal, esta determinada de forma experimental, tomando como ponto de partida os valores encontrados na literatura. Esta arquitetura deve ser grande o suficiente para extrair os resultados necessários e, ao mesmo passo, pequena o suficiente para que seu treinamento seja rápido, devendo-se acompanhar a cada treinamento a resposta fornecida pela rede para a verificação se esta está dentro dos resultados esperados.

Com relação aos métodos de análise de confiabilidade, verificou-se que o método de simulação de Monte Carlo, em todas as aplicações em que foi empregado, proporcionou resultados satisfátorios a um custo computacional elevado, não acontecendo o mesmo quando se aplicou as técnicas de redução de variância, estudadas neste trabalho, que reduziram de forma significativa o esforço requerido.

Os algoritmos de atualização dos pesos estudados apresentaram bons resultados, valendo-se salientar que o algoritmo de Levenberg – Marquardt gera melhores resultados e a rede converge, no treinamento, para um número de épocas menor que o outro algoritmo estudado, que requer um número de épocas elevado para atingir um

nível de erro aceitável, apesar deste ter proporcionado resultados aceitáveis nas aplicações analisadas.

A aplicação das redes neurais se revelou um método passível de ser utilizado em conjunto com o método de Monte Carlo, pois a utilização destes em conjunto proporcionou ótimos resultados tendo como consequência um baixo esforço computacional, fato que era objetivado neste trabalho.

Já para a aplicação das redes em substituição à análise estrutural mais confiabilidade, apesar de não se terem dados na literatura para posterior comparação dos resultados obtidos, esta foi bem sucedida no que concerne aos resultados fornecidos, comparados aos determinados pelo programa **FERUM**.

Num âmbito geral, o sucesso resultante da aplicação das redes neurais artificiais em conjunto com o método de Monte Carlo e as técnicas de redução de variância permite que esta possa ser aplicada, com mais frequência, para a resolução de determinados problemas de confiabilidade estrutural.

Uma sugestão para pesquisas futuras seria a utilização de algoritmos de atualização de pesos diferentes dos referenciados neste trabalho como, por exemplo, o algoritmo do gradiente conjugado, e também a utilização do algoritmo de Gauss-Newton, que servirão para a averiguação de qual a estratégia que melhor se adequa a solução de problemas de confiabilidade estrutural. Os métodos sugeridos não são os únicos encontrados na literatura, podendo ser usado qualquer outro destes para a posterior comparação.

Uma segunda sugestão propõe a utilização outros tipos de arquiteturas de redes, fazendo uso das redes de base radial (Radial Basis Networks), redes auto-organizáveis (Self – Organizing Networks) e redes recorrentes (Recurrent Networks), onde se enquadram as redes de Hopfield e de Elman, em que esta variação pode ser benéfica para o estudo destas diversas arquiteturas de rede na aplicação de problemas de confiabilidade.

Numa terceira opção, sugere-se a aplicação das redes neurais artificiais aplicadas em conjunto com os métodos analíticos FORM e SORM para a constatação da qualidade dos resultados, propondo-se também o estudo da confiabilidade estrutural com a utilização de outros métodos de análise como a técnica de superfície de resposta, ou com outras técnicas de redução de variância, para a verificação dos resultados fornecidos da aplicação das redes neurais.

REFERÊNCIAS BIBLIOGRÁFICAS

- Almeida, M. A. F. (2000). **Introdução ao estudo das redes neurais artificiais**, Universidade Federal de Santa Catarina, Florianopólis – SC.

- Ang, A. H.; Tang, W. H. (1984). **Probability concepts in engineering planning and design**, Vol. 2, Wiley.

- Ayyub, B. M.; Chia, C. -Y. (1992). **Generalized conditional expectation for structural reliability assessment**, Structural Safety, vol. 11, pp. 131-146.

- Baldi, P.; Hornik, K. (1989). **Neural networks and principal component analysis: Learning from examples without local minima**, Neural Networks, vol. 2, pp. 53-58.

- Bauchspiess, A. (2002). **Sistemas inteligentes: redes neurais e lógica fuzzy**, Mini-curso, 6ª Semana de Engenharia Elétrica, Universaidade de Brasília, Brasília – DF.

- Braga, A. P.; Ludemir, T. B.; Carvalho, A. C. P. L. F. (2000). **Redes neurais artificiais: Teoria e aplicações**, LTC Editora.

- Demuth, H.; Beale, M. (1998). **Neural Networks Toolbox User's Guide - MATLAB**, Versão 3.0.

- Devraignes, D.; Neto, F. M. (2003). **Dimensionamento de seções tranversais de concreto armado com a utilização de redes neurais artificiais (RNA)**, CILAMCE 2003, Ouro Preto – MG, CD-ROM.

- Engelund, S.; Rackwitz, R. (1993). **A benchmark study on importance sampling techniques in structural reliability**, Structural Safety, vol. 12, pp. 255-276.

- Frangopol, D. M. (1984). **Interactive reliability – based structural optimization**, Computer and Structures, vol. 19, pp. 559-563.

- Gomes, H. M. (2003). **Artificial neural networks and response surfaces for structural reliability analysis**, CILAMCE 2003, Ouro Preto – MG, CD-ROM.

- Gomes, H. M.; Awruch, A. M. (2004). **Comparison of response surface and neural network with others methods for structural reliability analysis**, Structural safety, vol. 26, pp. 49-67.

- Haukaas, T. (2001a). **A new computational framework for nonlinear finite element reliability and sensitivity analysis**, Qualifying Examination, University of Califórnia – Berkeley.

- Haukaas, T. (2001b). **Structural reliability analysis in OpenSees**, University of Califórnia – Berkeley.

- Hirose, Y.; Yamashita, K.; Hijiya, S. (1991). **Back-Propagation algorithm which varies the number of hidden units**, Neural Networks, vol. 4, pp. 61-66.

- Hurtado, J. E.; Alvarez, D. A. (2001). **Neural network – based reliability analysis: a comparative study**, Computer Methods in Applied Mechanics Engineering, vol. 191, pp. 113-132.

- Kamal, H. A.; Ayyub, B. M. (2000). **Variance reduction techniques for simulation – based structural reliability assessment of systems**, 8th ASCE Specially Conference on Probabilistic Mechanics and Structural Reliability.

- Karamchandrani, A. (1990). **Limitations of some of the approximate structural analysis methods that are used in structural system reliability**, Structural Safety, vol. 7, pp. 115 – 127.

- Liu, P-L.; Kiureghian, A. D. (1991). **Optimization algorithms for structural reliability**, Structural Safety, vol. 9, pp. 161 – 177.

- Machado, E. R. (2000). **Avaliação da confiabilidade de estruturas em concreto armado**, Dissertação de Mestrado, Universidade Federal de Minas Gerais – UFMG, Belo Horizonte – MG.

- Melchers, R. E. (1989). **Importance sampling in structural systems**, Structural safety, vol. 6, pp. 3-10.

- Melchers, R. E. (1987). **Structural Reliability: Analysis and Prediction**, John Wiley & Sons.

- Melchers, R. E. (1994). **Structural system reliability assessment using directional simulation**, Structural safety, vol. 16, pp. 23 - 37.

- Nascimento Jr., C. L.; Yoneyama, T. (2002). **Inteligência artificial em controle e automação**, Editora Edgard Blucher, 1ª Reimpressão.

- Oliveira, R. A. (1997). **Confiabilidade de sistemas estruturais pelo método de integração Monte Carlo com amostragem por importância**, Tese de Doutorado, Programa de Pós-Graduação em Engenharia Civil, COPPE/UFRJ, Rio de Janeiro – RJ.

- Osório, F. S.; Bittencourt, J. R. (2000) **Sistemas inteligentes baseados em redes neurais artificiais aplicados ao processamento de imagens**, I Workshop de Inteligência Artificial, Universidade de Santa Cruz do Sul.

- Papadrakakis, M.; Papadopoulos, V.; Lagaros, N. D. (1996). **Structural reliability analysis of elastic-plastic structures using neural networks and Monte Carlo simulation**, Computer Methods in Applied Mechanics Engineering, vol. 136, pp. 145-163.

- Papadrakakis, M.; Lagaros, N. D. (2002). **Reliability – based structural optimization using neural networks and Monte carlo simulation**, Computer Methods in Applied Mechanics Engineering, vol. 191, pp. 3491-3507.

- Pulido, J. E. (1991). **Confiabilidade de sistemas estruturais pelo método de Monte Carlo com técnica de redução de variância**, Tese de Doutorado, Programa de Pós-Graduação em Engenharia Civil, COPPE/UFRJ, Rio de Janeiro – RJ.

- Pulido, J. E.; Jacobs, T. L.; Prates de Lima, E. C. (1992). **Structural reliability using Monte carlo simulation with variance reduction techniques on elastic-plastic structures**, Computer and Structures, vol. 43, pp. 419-430.

- Rajashekhar, M. R.; Ellingwood, B. R. (1993). **A new look at the response surface approach for reliability analysis**, Structural Safety, vol. 12 pp. 205 - 220.

- Sagrilo, L. V. S. (1994). **Análise de confiabilidade estrutural utilizando os métodos analíticos FORM e SORM**, Tese de Doutorado, Programa de Pós-Graduação em Engenharia Civil, COPPE/UFRJ, Rio de Janeiro – RJ.

- SAP2000 Nonlinear Version 8.1.2 (1995). **User's Manual**, Version 8.1.2. Computer and Structures, Inc., Berkeley, CA.

- Saraiva, J. M. F. (1997). **A utilização de redes neurais em conjunto com o método de Monte Carlo na análise da confiabilidade de estruturas**, Tese de Doutorado, Programa de Pós-Graduação em Engenharia Civil, COPPE/UFRJ, Rio de Janeiro – RJ.

- Schuëller, G. I.; Pradlwater, H. J.; Koutsourclakis, P. S. (2003). **A comparative study of reliability estimation procedures for high dimensions**, 16th ASCE Engineering Mechanics Conference.

- Sciuva, M.; Lomario, D. (2003). **A comparison between Monte Carlo and FORMs in calculating the reliability of a composite structure**, Composite Structures, vol. 59, pp. 155 – 162.