

APLICAÇÃO DE REDES NEURAIAS ARTIFICIAIS NO ESTUDO DA CONFIABILIDADE ESTRUTURAL

JOSÉ MURILO F. SARAIVA
e
NELSON F. FAVILLA EBECKEN

COPPE/UFRJ Programa de Engenharia Civil
Caixa Postal 68506
CP 21945-970 Rio de Janeiro, Brasil
Tel.: + 55-21-560-8993, Fax: + 55-21-280-9545
E-mail: nelson@ntt.ufrj.br

SUMÁRIO

Apresenta-se nesse trabalho um estudo da aplicação de redes neurais na resolução do problema clássico da confiabilidade estrutural, ou seja, calcular a probabilidade de falha ou colapso de uma determinada estrutura. Esse estudo é de particular interesse na análise de estruturas em que as cargas predominantes são de natureza essencialmente probabilística, como é o caso das plataformas de exploração de petróleo sujeitas à ação do vento, ondas, correntezas, etc. Quando ocorrer um número de variáveis aleatórias elevado ou as funções de falha forem muito complexas, os métodos analíticos tradicionais, como o FORM e o SORM por exemplo, poderão não atingir a eficiência esperada. Nesses casos os métodos baseados em simulação, como o método de Monte Carlo, podem se tornar indispensáveis. Nos exemplos apresentados nesse texto avalia-se como o uso de redes neurais pode reduzir de forma significativa o número de simulações necessárias quando da utilização do método de Monte Carlo.

STRUCTURAL RELIABILITY ANALYSIS USING ARTIFICIAL NEURAL NETS

SUMMARY

This text presents a study of the use of neural networks for solving the classic structural reliability problem to obtain the probability of failure of collapse of a certain structure. This study is of particular interest when we analyse structures which prevailing loads are random variables, as it is the case of the Offshore Platforms that are exposed to wind, waves, streams, etc. When a considerable number of random variables occurs, or the limit state functions are too complex, the traditional analytic methods such as FORM and SORM sometimes are not as efficient as we could expect. In these cases, the simulation based methods like the Monte Carlo Technique can become the recommended tool. The examples presented in this text show how neural networks can considerably reduce the number of simulations needed when following the Monte Carlo Technique.

Recibido: Octubre 1996

INTRODUÇÃO

A análise da confiabilidade de estruturas grandes e complexas, por envolver um grande número de Variáveis Aleatórias ou exigir uma quantidade elevada de simulações, esbarra quase sempre em complicadas questões de ordem computacional e numérica.

As dificuldades numéricas resultam principalmente, do fato da confiabilidade estrutural se interessar pelo cálculo da *probabilidade de falha* (p_f), que é definida da seguinte maneira

$$p_f = \int_{g(X) \leq 0} f_X(x) dx \quad (1)$$

onde $X = (X_1, X_2, \dots, X_n)$ é variável aleatória n -dimensional. As variáveis X_1, \dots, X_n são também denominadas de *variáveis básicas* e envolvem todas as variáveis randômicas de interesse na análise (módulo de elasticidade, carga aplicada, resistência característica, etc.); $g(X)$ é a função de falha; $g(X) \leq 0$ o domínio de falha (O limite $g(X) = 0$ é chamado de superfície de falha); $f(x) = f_{x_1, x_2, \dots, x_n}(x_1, x_2, \dots, x_n)$ função densidade de probabilidade conjunta das variáveis básicas.

Verifica-se que fatores tais como um grande número de variáveis básicas, o domínio de falha envolvendo relações não lineares entre as variáveis, ou mesmo quando a função $f_X(x)$ não pode ser obtida diretamente através do produto das densidades marginais (como é o caso de variáveis independentes), podem tornar o cálculo analítico da integral (1) praticamente impossível.

Com o intuito de solucionar esse complicado problema numérico, poderosas ferramentas tem sido desenvolvidas nos últimos anos. Poderíamos citar algumas, tais como:

- Integração Numérica
- Simulação (Usando-se o método de Monte Carlo)
- Métodos Analíticos (FORM e SORM)

Todas as técnicas citadas acima já foram analisadas em várias situações, tendo-se verificado que a maior ou menor eficiência do método escolhido depende do tipo e complexidade do problema.

No caso da Integração numérica por exemplo, a dificuldade em alguns casos de se avaliar a função $g(X)$ tem limitado a sua aplicação a um número reduzido de variáveis. Alguns autores, como Melchers¹ e Madsen², apresentam a solução de alguns problemas de confiabilidade por integração numérica, mas sempre limitando o número de variáveis a um máximo de 8.

As técnicas envolvendo simulação através do método de Monte Carlo, embora sejam absolutamente gerais e de fácil implementação computacional, tornam-se em alguns exemplos, devido a convergência lenta e o grande número de simulações (em certos casos da ordem de 10^5), muito ineficientes, por exigirem um tempo de processamento elevado. Vários são os pesquisadores que tem se dedicado, através de engenhosos artifícios, em reduzir o número de simulações necessárias. Em seus trabalhos, Thoft³ e Melchers¹, apresentam técnicas como "Importance Sampling"

e “Conditional Expectation” (conhecidas como *técnicas de redução de variância*), que diminuem de forma considerável, quando comparados com o método de Monte Carlo clássico, o esforço exigido para se obter uma aproximação dentro dos critérios estabelecidos. Subia⁴ apresenta vários interessantes exemplos em que as técnicas citadas são aplicadas com sucesso.

Os métodos analíticos (FORM e SORM) são usados atualmente com bastante frequência, pois avaliam a probabilidade de falha através de criteriosas transformações que são feitas nas variáveis básicas que definem o problema, evitando assim a integração numérica. Na década de 70, Hasofer e Lind⁵, e posteriormente, Rackwitz e Fiessler⁶, propuseram um método de otimização (conhecido atualmente na literatura como método HL-RF) baseado no gradiente da função de falha e que tem sido utilizado desde então com relativo sucesso, constituindo-se num ponto de referência em quase todas as pesquisas feitas com confiabilidade estrutural. Vários trabalhos já foram publicados por diferentes pesquisadores com a finalidade de realizar refinamentos nos métodos FORM e SORM e diferentes programas comerciais já estão à disposição do engenheiro estrutural. Neste respeito podem ser citados os trabalhos de Kiureghian⁷ e Sagrilo⁸.

No entanto, os trabalhos de pesquisa nessa área ainda continuam em andamento, pois mesmo os métodos analíticos, apesar de bastante eficientes na maioria dos problemas estruturais envolvendo o cálculo de confiabilidade, podem ficar inviabilizados nos casos em que a superfície de falha é obtida através das interseções de várias funções $g_i(X)$, tornando muito complexo o problema de se encontrar o *ponto de projeto*, que vem a ser a base do método FORM. Com o intuito de se substituir $g(X)$ por uma função de falha aproximada $g^*(X)$, interessantes métodos já foram apresentados, como por exemplo o *Método da superfície de resposta* proposto por Holm⁹ e Sagrilo⁸ que fazem a aproximação da superfície exata (dada por $g(X)$) por uma superfície linear da forma

$$h(X) = a_0 + \sum a_i x_i$$

onde a_0, \dots, a_n são os coeficientes do hiperplano, ou através de uma superfície quadrática do tipo

$$p(X) = a_0 + \sum a_i x_i + \sum b_i x_i^2$$

No entanto, em algumas situações onde o ponto de projeto é difícil de ser avaliado, a superfície aproximada pode ser bastante imprecisa ocasionando erros grosseiros no cálculo da probabilidade de falha.

Pode-se deduzir, do que foi exposto até agora, a dificuldade de se conseguir um método absolutamente geral e que seja ao mesmo tempo de implementação computacional viável.

Portanto o objetivo principal do trabalho em curso não é o de se formular um novo algoritmo que venha a resolver todos os casos possíveis mas sim, colocar à disposição do analista uma ferramenta poderosa, de fácil entendimento e aplicação, e que tem sido objeto de exaustivos estudos nos principais centros de pesquisa no mundo inteiro, qual seja, a moderna técnica de *redes neurais artificiais*.

A teoria e a aplicação de redes neurais tem apresentado um extraordinário desenvolvimento nos últimos anos. Surpreendentes e interessantes aplicações foram desenvolvidas em áreas aparentemente sem nenhuma interligação. Algumas dessas aplicações podem ser vistas nos trabalhos recentes de Lopes¹⁰ e Arantes¹¹.

O presente trabalho tem por finalidade estudar como essa técnica pode auxiliar a solução de um problema de confiabilidade.

Há diversas definições para redes neurais artificiais mas, de forma resumida, pode-se afirmar que se trata de um modelo de computação paralela com aproximadores universais de funções, grafo direcionado, etc.

Sendo um aproximador universal de funções, com a sua convergência já demonstrada, (veja o teorema de existência de Kolmogorov¹²) e que exige somente alguns exemplos de "INPUTS" e "OUTPUTS" para o treinamento, pode-se configurar num ferramental poderoso nos casos em que a função de interesse é extremamente complexa. O conjunto contendo os "INPUTS" e "OUTPUTS", que é geralmente chamado de *conjunto de treinamento*, deve poder representar bem o domínio da função não havendo restrições rigorosas quanto ao número de elementos que deve conter. Como o método de Monte Carlo, embora bastante geral, tem como restrição básica o grande número de simulações exigido para se obter um resultado aceitável, pensou-se em conjugar os dois métodos: Monte Carlo + redes neurais, em que o primeiro forneceria o conjunto de treinamento, para se obter um algoritmo bastante simples e de fácil implementação.

Ao longo deste trabalho se irá comparar, através de exemplos com solução analítica conhecida, os resultados conseguidos com o algoritmo proposto com os já obtidos quando da utilização dos métodos clássicos. Para o treinamento e teste da rede neural foi utilizado o software Neuroshell¹³ que faz uso da técnica conhecida como *backpropagation*.

FORMULAÇÃO MATEMÁTICA

O Método de Monte Carlo

A probabilidade de falha p_f dada pela integral (1) também pode ser expressa na forma

$$p_f = \int_{g(x) \leq 0} I[g(X) \leq 0] f_x(x) dx \quad (2)$$

onde $I[.]$ é uma função que assume o valor 1 se $[.]$ é verdadeiro e o valor 0 se $[.]$ é falso.

Ao comparar a integral (2) com a expressão que fornece o valor esperado de uma variável aleatória X (ref.¹), chega-se a conclusão de que a mesma representa o valor esperado de $I[.]$. Se x_i^* representar a i -ésima observação da variável aleatória X então tem-se, como estimativa de (2), a expressão

$$p_f \approx (1/N) \cdot \sum_{i=1}^N I[g(x_i^*) \leq 0] \quad (3)$$

onde N representa o número de observações. A expressão (3) representa a base do método de Monte Carlo que será utilizado nos exemplos (“Exemplos Numéricos”).

O algoritmo *backpropagation*

A questão básica no início, quando do aparecimento de redes neurais multicamadas, era: Como “ensinar” (ou “treinar”) a rede artificial até que esta aprendesse a reconhecer a função desejada?

Com o intuito de se resolver essa questão foi proposto (na década de 80) por Rummelhart, Hinton e Williams¹⁴, um poderoso e eficiente algoritmo de aprendizagem, conhecido atualmente como “BACKPROPAGATION”, cuja principal característica é a de permitir que os neurônios da camada intermediária sejam capazes de modificar os pesos das suas conexões (sinapses) se receberem informações sobre os erros de cada neurônio pertencente a camada de saída.

O método pode ser resumido através das seguintes equações

$$\delta_j^{c-1} = (dx_j^{c-1}/dv_j^{c-1}) \cdot \sum_{i=1}^{N_c} w_{ij}^c \cdot \delta_i^c \quad (4)$$

$$\Delta w_{ij} = 2\eta x_j^{c-1} \cdot \delta_i^c \quad (5)$$

sendo i o índice de um neurônio da camada c , j o índice de um neurônio da camada $c - 1$, w_{ij}^c o peso ou junção sináptica entre o neurônio i da camada c e o neurônio j da camada $c - 1$, N_c o número de neurônios da camada c , δ_j^{c-1} o erro para o neurônio j da camada $c - 1$, δ_i^c o erro para o neurônio i da camada c , x_j^{c-1} o sinal de saída do neurônio j da camada $c - 1$, v_j^{c-1} o sinal de entrada do neurônio j da camada $c - 1$ e η tamanho do passo ao se usar a “regra delta”.

As equações (4) e (5) fornecem uma regra bastante eficiente para o treinamento de uma rede neural multicamada com o objetivo de se implementar uma relação funcional do tipo $j = \varphi(\underline{x})$ onde $\underline{x}^t = \{x_1, \dots, x_n\}$ é o sinal de entrada na rede e $\underline{y}^t = \{y_1, \dots, y_n\}$ o sinal de saída.

Para a implementação do algoritmo há a necessidade de se fornecer á rede um conjunto de treinamento $\psi[\underline{x}(s), \underline{y}(s)]$ que caracterize bem o domínio de operação.

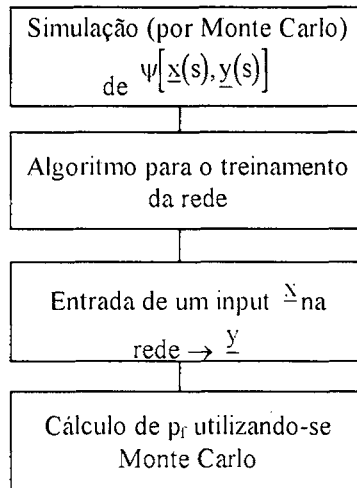
Conforme visto na introdução, o cálculo da probabilidade de falha apresenta como grande obstáculo, dentre outros, o fato da função $g(x)$, que representa o domínio de falha, não ser muito simples de ser avaliada na maioria das aplicações. Portanto, uma das idéias básicas ao se utilizar redes neurais no cálculo de p_f , é o de treinar uma rede para reconhecer a função $g(x)$ uma vez fornecido um conjunto de treinamento ψ , conjunto esse que poderá ser gerado utilizando-se uma técnica adequada de simulação.

Considerando-se o que foi exposto acima, conclui-se que a utilização de redes neurais para o cálculo de p_f poderá ser feita junto com o método de Monte Carlo, que fornecerá o conjunto de treinamento ψ . Essa sugestão de se “combinar” os dois algoritmos apresenta as seguintes vantagens:

- 1) Menor número de simulações (já que Monte Carlo será usado apenas para gerar o conjunto de treinamento).

- 2) Uma vez treinada a rede, não haverá restrições quanto ao número de 'INPUT's", representado pelo conjunto $\underline{x}(s)$.

Tem-se então, para o algoritmo proposto nesse trabalho, o seguinte organograma:



Nos exemplos apresentados a seguir os vetores $\underline{x}(s)$ e $\underline{y}(s)$ do conjunto ψ serão respectivamente $\underline{x}(s)$ o número aleatório $\in [0, 1]$ (ou valores de uma variável $N[0, 1]$), $\underline{y}(s)$ os valores de uma função $F(x)$ (ou de uma função de falha $g(x)$).

Vale destacar que os casos escolhidos pretendem ser principalmente ilustrativos.

EXEMPLOS NUMÉRICOS

Treliça com carga aplicada P aleatória

Seja a estrutura da (Figura 1):

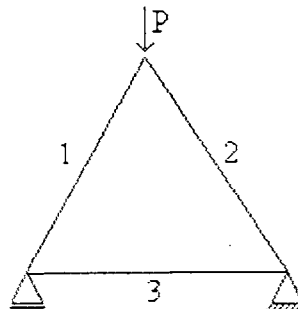


Figura 1

Variável	Distribuição	Média	Desvio Padrão
Carga(P)	normal	14	1,25
Resistência(R)	normal	11	1,50

Tabla I. Valores em t

Considere como Variáveis aleatórias básicas, a carga aplicada P e a resistência do material R . As distribuições de probabilidade das variáveis, bem como os valores de seus parâmetros, estão resumidos na Tabela I.

Sendo normal a distribuição de probabilidade das 2 variáveis, a probabilidade de falha na barra I será (veja Melchers¹ e Thoft³)

$$p_f = \Phi[-(\mu_r, \mu_{pi})/(\sigma_r^2 + \sigma_{pi}^2)^{1/2}] \tag{6}$$

sendo μ_r, μ_{pi} as médias das variáveis R e P_i (esforço normal na barra i), $\sigma_r^2, \sigma_{pi}^2$ as variâncias de R e P_i , Φ função de distribuição acumulada da variável normal padrão ($N[0, 1]$).

A utilização da expressão (6) fornecerá os valores de p_f para as barras 1, 2 e 3 (Tabela II).

Barra	p_f
1 = 2	0,040059
3	3,24E-06

Tabla II. Valores de p_f

Utilizando-se as regras clássicas de probabilidade (veja Arig and Tang¹⁵) tem-se que, a probabilidade do evento $W = \{A \text{ estrutura (treliça) falha}\}$, será igual a (a probabilidade da barra 3 foi desprezada devido a sua ordem de grandeza)

$$P(W) = 0,0785$$

Ao se realizar uma simulação, através do método de Monte Carlo, usaremos a seguinte expressão básica (Melchers¹)

$$p_f = P[(R, P) \leq 0] = \int_{-\infty}^{\infty} F_R(x) \cdot f_P(x) dx \tag{7}$$

sendo $F_R(x)$ a função de distribuição acumulada da variável R e $f_P(x)$ a função densidade de probabilidade da variável P .

A fórmula (7) equivale ao cálculo da média aritmética de $F_R(x)$. Ao se estimar $F_R(x)$ por Monte Carlo a seguinte seqüência de operações será realizada:

- Achar $r_i =$ número aleatório entre $[0 : 1]$
- Obter $x_i^* = \Phi^{-1}(r_i)$
- Obter X_i^* da expressão $x_i^* = (X_i^*, \mu_P)/\sigma_P$
- Calcular $r_i^* = (X_i^*, \mu_R)/\sigma_R$
- Calcular $F_R(x_i^*) = \Phi(r_i^*)$

É importante lembrar que os valores de x_i^* e $F_R(x_i^*)$ serão, respectivamente, os INPUT's e os OUTPUT's [ou seja os valores $(x(s), y(s))$] do conjunto de treinamento da rede neural. Vale ressaltar também que, caso se utilizasse o método de Monte Carlo para se estimar a probabilidade de falha da barra 1, por exemplo, seriam gerados N números aleatórios sendo $N = (z_\alpha/\varepsilon)^2 \cdot [(1 - p_f)/p_f]$, o que daria para $p_f = 0,040059$ (que é a probabilidade de falha da barra 1 = barra 2) um valor da ordem de 2.300 (uma estimativa grosseira seria $N = 100/p_f$, o que daria, nesse caso, $N = 2.500$). No entanto, para uma rede neural, não existe uma fórmula específica que determine o número de pares para o treinamento^{16,17,18}. A única recomendação de ordem prática é que o número de pares deve ser suficiente para caracterizar bem o domínio de operação, número esse que é, na maioria dos casos, bem inferior ao que seria exigido com a aplicação do método de Monte Carlo clássico. Essa é sem dúvida uma vantagem considerável quando levamos em conta aquelas situações em que a função $g(x)$ é bastante complexa e de avaliação onerosa em termos computacionais.

Verifica-se que dificilmente surge uma aplicação em confiabilidade estrutural em que seja necessária a utilização de mais de uma camada oculta. Em todos os exemplos do texto foram testadas redes com 2 e 3 camadas sem que houvesse qualquer ganho significativo.

Em relação ao número de neurônios na camada intermediária, não existe um dimensionamento rígido, sendo portanto uma característica do projeto de redes neurais. O fabricante do software utilizado, baseado em experiências anteriores, sugere a seguinte expressão

$$N' = [(N_{\text{ent}} + N_{\text{sai}}) + (N_{\text{amostras}})^{1/2}]/2 \cdot N_i$$

onde N_{ent} é o número de neurônios na camada de entrada, N_{sai} o número de neurônios na camada de saída, N_{amostras} o número de amostras para treinar a rede e N_i o número de camadas intermediárias.

Um outro critério, propõe uma determinação experimental da arquitetura ótima, a qual pode ser resumida no seguinte algoritmo:

1. Começar com dois neurônios na camada intermediária.
2. Para cada taxa de aprendizagem η , variando de $\eta = 0,1$ a $\eta = 1$.
 - 2.1 Treinar e fazer testes ao longo do treinamento.
 - 2.2 Calcular o erro máximo em módulo, junto com o erro quadrático médio.
3. Se o desempenho não for satisfatório, adicionar um neurônio e voltar ao passo 2.

No exemplo que esta sendo apresentado, o conjunto de treinamento constou de 30 pares $[x_i^*, F_R(x_i^*)]$ e foi utilizada, uma rede neural do tipo *backpropagation*. Após o treinamento forneceram-se com 50 INPUT's x_i^* obtendo-se, com a rede, 50 valores de $F_R(x_i^*)$.

A estimativa de p_f para a barra 1 será

$$p_f = (\sum F_R(x_i^*)) / 50 = 0,0380$$

com um erro relativo de $\varepsilon = 5,1 \%$.

Evidentemente que não haveria a menor dificuldade de se gerar os valores de x_i^* nesse caso pois, se tratando de uma variável normal reduzida, um par de valores $[x_1^*, x_2^*]$ seria obtido com o uso das expressões

$$\begin{aligned} x_1^* &= (-2 \cdot \ln(r_1))^{1/2} \cdot \sin(2\pi r_2) \\ x_2^* &= (-2 \cdot \ln(r_1))^{1/2} \cdot \cos(2\pi r_2) \end{aligned} \tag{8}$$

sendo r_1 e r_2 números aleatórios $\in [0 : 1]$.

O exemplo também foi resolvido com a utilização apenas do método de Monte Carlo. Uma tabela contendo o resumo dos resultados é apresentada a seguir (Tabela III).

Método	Número de simulações	p_f (barra 1)
* Monte Carlo	1.000	0,0389
* Monte Carlo + Rede neural	30 (para o treinamento) + 50 nos aleatórios par ao teste	0,0380

Tabla III. Resumo dos resultados

Função de falha qualquer

Adotou-se como exemplo a seguinte função de falha¹⁹

$$g(X) = X_1^{t_1}, X_2 \cdot t_2 + t_3$$

onde t_1, t_2 e t_3 são parâmetros determinísticos e os dados estatísticos das variáveis X_1 e X_2 são fornecidos na Tabela IV.

Variável	Distribuição	Média	Desvio Padrão
X_1	uniforme	50	28,9
X_2	exponencial	20	20

Tabla IV. Dados estatísticos das variáveis X_1 e X_2

Considerando-se as variáveis X_1 e X_2 independentes (a função de prob. conjunta será o produto das densidades marginais e igual a $f_{(x_1, x_2)} = (1/2.000) \cdot \exp(-0,05 x_2)$), e os valores dos parâmetros determinísticos respectivamente iguais a $t_1 = 1$, $t_2 = 2$ e $t_3 = 0$, o valor exato da probabilidade de falha será dado por

$$p_f = \int_0^{100} \int_0^{x_2} (1/2.000) \cdot \exp(-0,05 x_2) dx_1 dx_2 \quad (9)$$

logo $p_f = 0,1986$.

Segundo o manual do programa Calrel¹⁹, os valores da probabilidade de falha, calculada pelos métodos tradicionais, são $p_f = 0,1901$ (FORM), $p_f = 0,1969$ (SORM) e $p_f = 0,1990$ (Monte Carlo (com 1.000 simulações)).

Ao observar com atenção a integral (9) obtêm-se, após a integração em x_1 , o seguinte

$$p_f = (1/100) \cdot \int_0^{100} x_2 \cdot (0,05 \cdot \exp(-0,05 x_2)) dx_2 \quad (10)$$

Levando em conta a definição de valor esperado, junto com o fato de que a expressão entre parênteses no integrando é exatamente igual a densidade de probabilidade marginal da variável x_2 chega-se a conclusão de que (10) representa (a menos da constante $1/100$) o valor médio de x_2 . Isso sugere um procedimento semelhante ao que foi adotado no exemplo da Figura 1 para gerar o conjunto de treinamento, qual seja

- Achar r_i = número aleatório entre $[0 : 1]$
- Calcular $x_2^i = (-20) \cdot \ln(1 - r_i)$ [lembrando que x_2 tem uma dist. exponencial]
- Calcular $F_i = x_2^i / 100$

No caso do exemplo corrente, o procedimento descrito acima foi utilizado para se gerar um conjunto de treinamento com 50 pares $[r_i : F_i]$. Após a implementação da rede neural com 100 INPUT's r_i , obtêm-se 100 valores de F_i^* .

A avaliação da probabilidade de falha será feita da seguinte maneira

$$p_f = (\sum F_i^*) / 100 = 0,1927$$

(com um erro relativo de 3 % em relação ao valor exato).

Um resumo dos resultados pode ser visto na Tabela V.

Método	No de simulações	p_f
FORM	-	0,1901
SORM	-	0,1969
Monte Carlo (clássico)	10.000	0,1990
Monte Carlo + rede neural	50 (para o treinamento + 100 Nos aleatrios)	0,1927

Tabla V. Resumo dos resultados

Pórtico plano com 3 funções de falha

Seja o pórtico representado na Figura 2

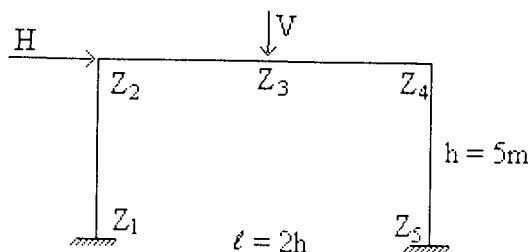


Figura 2. Pórtico plano

O cálculo da probabilidade de falha desse pórtico foi feito por Madsen² e posteriormente por Sagrilo⁸, que utilizou os algoritmos ALR e ANLR que operam com aproximações lineares para a superfície de falha no espaço das variáveis reduzidas.

São consideradas como variáveis aleatórias os momentos plásticos resistentes Z_1, \dots, Z_5 e as cargas aplicadas H e V . No seu estudo, Madsen² considera três mecanismos de colapso que são representados pelas seguintes funções de falha

$$\begin{aligned}
 g_1(X) &= Z_1 + Z_2 + Z_4 + Z_5, Hh \\
 g_2(X) &= Z_1 + 2Z_2 + 2Z_4 + Z_5, Hh, Vh \\
 g_3(X) &= Z_2 + 2Z_3 + Z_4, Vh
 \end{aligned}$$

onde as características das variáveis são apresentadas na Tabela VI.

Variável	Distribuição	Média	Desvio Padrão
Z_1, \dots, Z_5	lognormal	134,9	13,49
H	lognormal	50,0	15,00
V	lognormal	40,0	12,00

Tabla VI. Unidades: kN e m

O interesse no problema recai no fato de que, como qualquer mecanismo pode conduzir ao colapso da estrutura, a probabilidade de falha do pórtico será dada por

$$p_f = \cup_{i=1}^3 P(g_i \leq 0)$$

Esse é um exemplo do que, em confiabilidade estrutural, se denomina de “sistema” ou seja, a superfície de falha tem mais de um ponto de máxima densidade local. Embora o método FORM seja de aplicação perfeitamente viável nesse caso, seria oportuno destacar o fato de que a existência de várias funções de falha obrigaria a aplicação do

método para cada uma delas e na prática da engenharia não é incomum a ocorrência de situações em que aparecem na análise dezenas de funções de falha.

É claro que esse exemplo poderia ser resolvido com a utilização do método de Monte Carlo clássico. No entanto, o grande número de variáveis bem como a existência de 3 funções de falha, sugerem como sendo vantajosa a utilização conjunta do método com redes neurais. Isso foi feito para cada função g_i gerando-se, através de Monte Carlo, 3 conjuntos de 50 pares $[r_i, g_i(X)]$ sendo r_i um número aleatório $\in [0; 1]$. Após o treinamento da rede, foram obtidos 1000 valores de g_i e a probabilidade de falha estimada pela razão

$$p_f = (1/1.000) \cdot \sum I\{g_i(X) \leq 0\}, \text{ onde } I = 1 \text{ para } g_i(X) \leq 0 \text{ e } I = 0 \text{ para } g_i(X) > 0 \quad (11)$$

Os resultados (apenas para a função $g_2(X)$) são dados na Tabela VII.

$P(g_2(X) \leq 0)$		
FORM [14]	FORM [11]	Monte Carlo + rede neural
0,00197	0,00199	0,00200

Tabla VII. Os resultados para a função $g_2(X)$

CONCLUSÕES

Os exemplos numéricos apresentados tiveram como objetivo registrar a importância de se empregar redes neurais no estudo da confiabilidade estrutural, particularmente no cálculo de p_f . Verificou-se que a combinação dos dois métodos, Monte Carlo + redes neurais, o primeiro sendo utilizado para fornecer o conjunto $\Psi[x_s, y_s]$ como sendo os inputs e outputs iniciais para a implementação da rede, pode resultar numa ferramenta poderosa para se resolver aqueles casos em que a função de falha $g(x)$ é muito complexa ou exige uma análise estrutural muito dispendiosa para a sua avaliação. Além disso, o número de pares necessários para o treinamento é consideravelmente menor do que o número de simulações exigidas quando da aplicação isolada do método de Monte Carlo (da ordem de $100/p_f$).

Diversas alternativas para a utilização de redes neurais poderiam ser citadas, como por exemplo outras implementações do Método de Monte Carlo, mas sem dúvida alguma a análise de sistemas estruturais complexos envolvendo acentuadas não-linearidades e grande número de variáveis aleatórias será possível ser efetuada de forma eficiente com a utilização desta metodologia combinada.

REFERÊNCIAS

1. R.E. Melchers, "*Structural Reliability Analysis and Prediction*", Elis Horwood Limited, Chichester, England, (1987).
2. H.O. Madsen, S. Krenk e N.C. Lind, "*Methods of Structural Safety*", Prentice-Hall, Englewood Cliffs, New Jersey, (1986).
3. P. Thoft-Christensen e M.J. Baker, "*Structural Reliability Theory and its Applications*", Springer-Verlag, Berlin, (1982).
4. J.E.P. Subia, "Confiabilidade Estrutural Usando o Método de Monte Carlo com Técnicas de Redução de Variância", D.Sc. Thesis, Programa de Engenharia Civil, COPPE/UFRJ, Brasil, (1991).
5. A.M. Hasofer e N.C. Lind, "Exact and Invariant Second-Moment Code Format", *J. Engng. Mechanics (ASME)*, Vol. **100**, N.º. 1, pp. 111-121, (1974).
6. R. Rackwitz e B. Fiessler, "Structural Reliability under Combined Random Load Sequences", *Computer and Structures*, Vol. **9**, pp. 489-494, (1978).
7. A. Der Kiureghian, "Structural Component Reliability", Report N.º. CE229, Department of Civil Engineering, University of California at Berkeley, USA, (1995).
8. L.V.S. Sagrilo, "Análise de Confiabilidade Estrutural Utilizando os Metodos Analíticos FORM e SORM", D.Sc. Thesis, Programa de Engenharia Civil, COPPE/UFRJ, Brasil, (1994).
9. C.A. Holm, "Reliability Analysis of Structural Systems Using Nonlinear Finite Element Methods", D. Sc. Thesis, Division of Structural Engineering, the Norwegian Institute of Technology, the University of Trondheim, (1990).
10. T.A.P. Lopes, "Avaliação do Dano de Fadiga em Plataformas de Petróleo em Tempo Real", D.Sc. Thesis, Programa de Engenharia Civil, COPPE/UFRJ, Brasil, (1995).
11. J.A. Arantes, "Utilização de Técnicas de Inteligência Artificial na Elucidação de Características Desejáveis a um Sistema de Classificação de Embarcações por meio de suas Assinaturas Magnéticas", M.Sc. Thesis, Programa de Engenharia de Sistemas, COPPE/UFRJ, Brasil, (1995).
12. A.N. Kolmogorov, "On the Representation of Continuous Functions of Many Variables by Superposition of Continuous Functions of One Variable and Addition", *Dokl. Akad. Nauk.*, Vol. **114**, pp. 953-956, USSR, (1957).
13. NEUROHELL2, Ward Systems Group, Inc., (1993).
14. D.E. Rumelhart, G.E. Hinton e R.J. Williams, "Learning Internal Representations by Error Propagation, Parallel Distributed Processing: Explorations in the Microstructure of Cognition", Vol. **1**, D. Rumelhart e J. McClelland (Eds.), MA: MIT Press/Bradford Books, pp. 318-362, Cambridge, (1986).
15. A.H. Ang e W.H. Tang, "Probability Concepts in Engineering Planning and Design", Vol. **2**, John Wiley and Sons, New York, (1984).
16. J. Hertz, A. Krogh e R.G. Palmer, "Introduction to the Theory of Neural Computation", Addison-Wesley Publishing Company, New York, (1991).
17. Z.L. Kovacs, "Redes Neurais Artificiais", Edicao Academica, Brasil, (1996).
18. D.M. Skapura, "*Building Neural Networks*", Addison, Wesley Publishing Company, New York (1996).
19. P.L. Liu, H.Z. Lin e A.D. Kiureghian, "CALREL User Manual", Report N.º. UCB/SEMM, **89/18**, Structural Engineering, University of California at Berkeley, USA, (1989).