Proceedings of the V Brazilian Conference on Neural Networks - V Congresso Brasileiro de Redes Neurais
pp. 175-180, April 2-5, 2001 - Rio de Janeiro - RJ - Brazil

Otimização por Algoritmos Genéticos e Simulated Annealing após Tratamento de Dados por Sistemas Neuro-Fuzzy

Walter M. Cuenca[1, 2], Sandoval Carneiro Jr[2], Luis A. Cheim[3], Luis P. Calôba[2]
1. CEPEL/ACET, C. P. 68007, Rio de Janeiro, 21944-970, Brasil Tel: 21 598-6222
2. COPPE/UFRJ C. P. 68504, Rio de Janeiro, 21945-970, Brasil Tel: 21 260-5010
3. SIEMENS São Paulo, Brasil Tel: 11 4585 2113
E-mails: martin@cepel.br, sandoval@dee.ufrj.br, cheim@siemens.com.br, caloba@lps.ufrj.br

Abstract

A great part of the scientific applications and of the engineering problems, they can be considered as search and optimization problems. Due to need to obtain better results in these problems, several tools were developed looking for the optimal global. Search methods have been developed based on natural and physical processes. The most usual are: Genetic Algorithm (AG) and Simulated Annealing (SA). In this paper applies AG and SA separately, with the objective of reinforcing the performance of the Neuro-Fuzzy system in a function approximation problem, whose improvement and efficiency shown the results.

1. Introdução

O MANFIS (Mamdani Adaptive Fuzzy Inference System) [1] é uma técnica implementada a partir de um sistema *fuzzy* em uma estrutura de rede neural, tal como se mostra na Figura 1. Em sua essência é um Neuro-*fuzzy* Híbrido baseado na regra de inferência de Mamdani [2]. Este sistema foi aplicado para reconhecer padrões em "Descargas Parciais", com bastante sucesso [3]. Outras aplicações, como uma função de aproximação puramente teórica e um problema de predição de séries temporais, foram utilizadas para demonstrar a capacidade de identificação do sistema MANFIS.

O MANFIS se caracteriza por ter uma estrutura de quatro camadas. Na primeira camada os dados de entrada são fuzzificados, utilizando uma função contínua "*Bell Shape*". Na segunda camada se faz a montagem das regras, hierarquizadas convenientemente para ativar uma ou mais regras na camada seguinte, nesta camada (conseqüentes *fuzzy*) utilizam-se os critérios de inferência *fuzzy* de Mamdani (Max-Min) e na ultima camada implementa-se o defuzzificador. Os pesos em todas as conexões, entre camadas, são fixos e unitários.

O sistema de treinamento do MANFIS é do tipo supervisionado. O erro é medido na saída do defuzzificador (erro de saída), este erro é retropropagado para as camadas anteriores onde os parâmetros das funções *fuzzy* de fuzzificação são corrigidos, tornando-se funções adaptativas. Técnicas tradicionais, como: gradiente descendente, métodos de Newton, gradiente conjugado, Quase-Newton e estratégias de métrica variável usadas para achar o mínimo de suas funções objetivos (valor esperado), são capazes de encontrar ótimos locais, sendo ineficazes para os casos de otimização de funções multimodais, não-diferenciáveis e não-contínuas.

2. MANFIS

Esta técnica é uma estrutura neural simples tal como se ilustra na Figura 1, para um sistema de variáveis, entrada *(x,y)* e uma saída *(z)*. Os neurônios representados por círculos contendo *bellshapes* são os nós adaptativos enquanto que, aqueles representados por apenas um circulo, são as regras *fuzzy*.

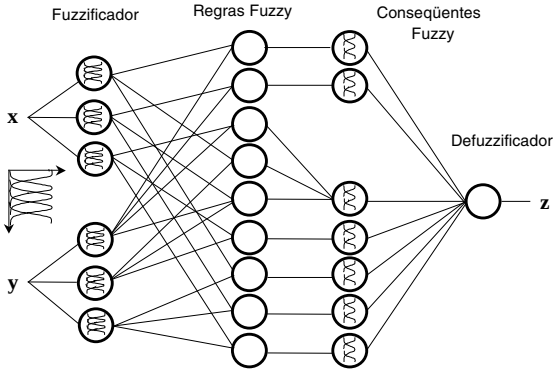

Figura 1: Sistema híbrido neuro-*fuzzy*

A primeira camada de neurônios é o fuzzificador, implicando a cada função de ativação do neurônio uma função de pertinência *fuzzy 'bellshapes'*. Para simplificar, cada espaço da variável de entrada é dividido em três regiões *fuzzy*, gerando nove possíveis regras após todas as combinações de antecedentes *fuzzy*, assim o número de nós ativados (regras disparadas) são apresentados na camada de conseqüentes *fuzzy*. Entre o fuzzificador e a camada conseqüentes *fuzzy* está a camada de mínimos (regras *fuzzy*).

O defuzzificador tem um neurônio único que avalia o centro de pesos, gerado pela combinação das funções de pertinência, alcançando a máxima saída z (*crisp*) [4], pela seguinte equação:

$$z = \frac{\sum_{i=1}^{n} (m_i c_i)}{\sum_{i=1}^{n} m_i} \qquad (1)$$

onde n é o número de nós da camada de conseqüentes *fuzzy* e m_i representa o máximo do nó i, em que a função *bellshape* é centralizada na vizinhança de c_i. O m_i é obtido na camada de conseqüentes *fuzzy* a través da regra de inferência composicional max-min de Mamdani [2]

O erro quadrático (*Erq)* associado ao nó de saída, é dado por:

$$Erq = \frac{1}{2}(z_d - z)^2 \tag{2}$$

onde z_d é a saída desejada num algoritmo de treinamento supervisionado. O termo erro ε é retropropagado pela rede de acordo com a derivada do *Erq* com respeito à rede de entrada e ao nó de saída, assim dado:

$$\varepsilon = -(z_d - z) f' \tag{3}$$

onde f é a função de ativação do nó e f' sua derivada com respeito a $m_i c_i$. Tomando derivadas parciais de f com respeito a m_i e c_i dados por:

$$\frac{\partial f}{\partial c_i} = \frac{m_i}{\sum_{j=1}^{n} m_j} \tag{4}$$

$$\frac{\partial f}{\partial m_i} = \frac{c_i \sum_{j=1}^{n} m_j - \sum_{j=1}^{n} m_j c_j}{\sum_{j=1}^{n} m_j} \tag{5}$$

Deve-se ter precaução neste ponto, na aplicação da retropropagação do erro, porque as camadas do MANFIS nem sempre estão totalmente conectadas, dependem das possíveis correlações existentes entre dados de entrada e saída do modelo físico que se quer simular.

No processo de retropropagação são atualizados os parâmetros de cada função de ativação *'bellshape'*, caraterizados pela função de pertinência $\mu(\cdot)$. Para *'x'* se tem:

$$\mu(x,a,b,c) = \frac{1}{1 + \left| \frac{x-c}{a} \right|^{2b}}, \tag{6}$$

onde o valor da função μ é definido no intervalo [0, 1], c é o centro da função *bellshape*, a é o ponto de cruzamento e b define a função *steepness* ($b/2a$ é a inclinação da função *bellshape* para $x = c - a$), x representa uma variável de entrada no nó.

Aplica-se, a um nó j, um parâmetro geral p_j da função de ativação bellshape que pode ser a, b ou c na Eq. (6). O termo atual deste parâmetro geral p_j é dado por:

$$\Delta p_j = -\eta \varepsilon_j \frac{\partial \mu}{\partial p_j}, \tag{7}$$

onde η é a taxa de aprendizagem e ε_j o termo erro propagado pelo nó j. O parâmetro geral é atualizado por:

$$p_j^{new} = p_j^{old} + \Delta p_j, \tag{8}$$

Então, desenvolvendo as derivadas parciais de μ com respeito aos parâmetros a, b e c dados por [4]:

$$\frac{\partial \mu}{\partial a} = \left(\frac{2b}{a}\right)\mu(x)(1-\mu(x)) \tag{9}$$

$$\frac{\partial \mu}{\partial b} = -2\ln\left|\frac{x-c}{a}\right|\mu(x)(1-\mu(x)) \tag{10}$$

$$\frac{\partial \mu}{\partial c} = \left(\frac{x-c}{a}\right)\mu(x)(1-\mu(x)) \tag{11}$$

3. Função Teste e Modelo Aproximado

O modelo matemático usado para testar o MANFIS é dado pela Eq. (12), para 121 pares de dados de entrada (x,y) para o treino. A Figura 2 apresenta as funções *bellshapes* que representam as variáveis x, y e z em seus respectivos espaços *fuzzy*. Repare que as variáveis x e y são representadas por um espaço *fuzzy* dividido apenas em três conjuntos *fuzzy*. A partir da Figura 1, pode-se facilmente ver que o número total de regras *fuzzy* possíveis é nove, e o espaço da variável z é dividido em nove conjuntos *fuzzy* ou nove possíveis combinações de conjuntos *fuzzy* x, y (ver Figura 2).

$$z = 3.\frac{\sin(x)\sin(y)}{xy} \tag{12}$$

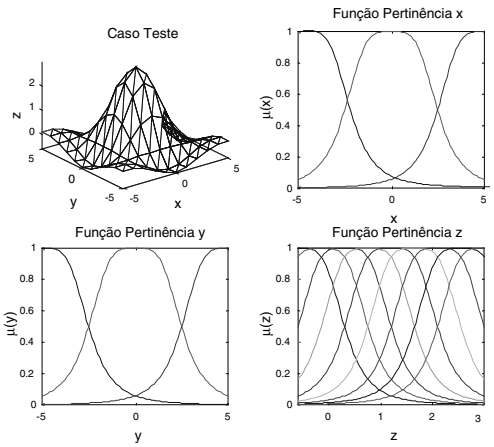

Figura 2: Funções de pertinência *fuzzy* antes do treino MANFIS.

A Figura 3 ilustra o desempenho do MANFIS. Observa-se que a primeira curva é o erro médio quadrático do treino (APE), esta decresce rapidamente. A segunda curva é a taxa de aprendizagem. Ambas curvas características, garantem um bom desempenho do MANFIS, mesmo usando só três conjuntos *fuzzy* por variável. Aumentando o número de conjuntos *fuzzy*, o MANFIS obtém uma melhor aproximação desta função. Já na Figura 4, observa-se que os conjuntos *fuzzy* se deslocaram em forma adaptativa em cada iteração nos domínios respectivos de seus parâmetros a, b e c. Na saída do MANFIS se tem uma concentração dos conjuntos *fuzzy* deslocados de suas origens antes do treino. Esta saída revela

que esta função de aproximação pode ser representada com menos de nove regras *fuzzy*.

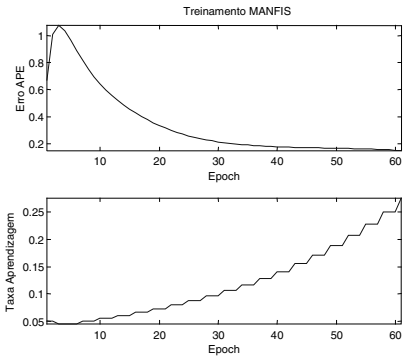

Figura 3: Erro de aprendizagem (APE) e taxa de aprendizagem do MANFIS com três conjuntos *fuzzy*

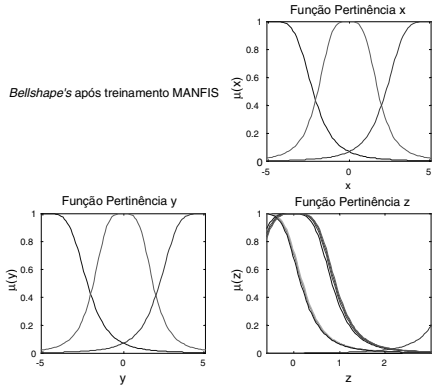

Figura 4: Funções de pertinência *fuzzy* após o treino MANFIS

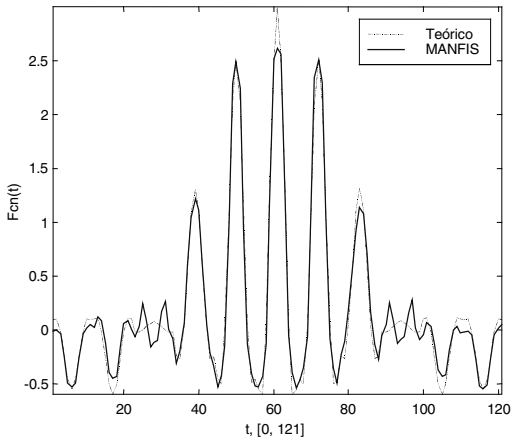

Figura 5: Função teste aproximada pelo MANFIS usando apenas três conjuntos *fuzzy*.

Na Figura 5 espera-se uma boa aproximação da função, porém alguns patamares não foram alcançados, razão pelo qual podem ser considerados como um problema de otimização.

O MANFIS adapta seus parâmetros (*a*, *b* e *c*) usando gradiente descendente na condição de garantir um mínimo valor (erro médio quadrático) para seu desempenho, esta estrutura faz com que o MANFIS dependa das derivadas parciais de seus parâmetros, que podem convergir facilmente para ótimos locais. Outra característica importante é que, nem todos os neurônios de uma camada estão conectados, e estes pesos (*sinapses)* servem só de conexão com valor unitário. Os pesos de cada conexão não são atualizados pela retropropagação do erro. Só os parâmetros de cada *bellshape* se atualizam a cada iteração do algoritmo, denominado processo adaptativo.

Os parâmetros *a, b, c* e *cz* variam a cada iteração, até chegarem a um equilibro e não se deslocarem mais. O certo é que nem toda função problema é bem comportada, podendo surgir casos não convergentes. Estes inconvenientes podem ser superados implantando-se uma rotina de otimização, usando AG's ou SA.

4. Otimização dos Resultados do MANFIS por AG's e SA

AG's e SA serão tratados separadamente, adequando-se os parâmetros do MANFIS para cada um deles. Os resultados serão comparados com a função teórica e com a função aproximada obtida pelo MANFIS.

Diversos trabalhos sobre SA citam vantagens desse método sobre os AG's, e vice-versa. É evidente que certos algoritmos adaptam-se melhor do que outros a determinados problemas, e isso contribui para acalorar essa discussão.

4.1. Otimização do MANFIS por AG's

Os AG's são métodos de busca estocásticos que emulam teorias evolucionárias biológicas para resolver problemas de otimização. Caracterizam-se por sua flexibilidade e facilidade de paralelização. As vantagens atribuídas aos algoritmos de SA são sua melhor fundamentação analítica, que permitem um maior controle sobre as características do algoritmo e, principalmente, a existência de uma prova de convergência para um ótimo global. Entretanto, os AG's são mais utilizados para administrar problemas de descontinuidade, não diferenciável, multimodais e não convexos, por sua facilidade de conviver com estes problemas [5].

Eles se adaptam mais naturalmente aos problemas de otimização combinatória, devido a sua natureza intrínseca discreta. Já os algoritmos de SA se adequam com menos esforço a problemas contínuos [5], [6]. Entre os componentes básicos dos AG's destacam-se:

- Representação genética das soluções viáveis do problema.
- Determinação de uma população inicial de cromossomos.
- Definição da função de avaliação dos cromossomos.
- Definição dos operadores genéticos eficazes na reprodução de novos cromossomos.
- Definição dos parâmetros que compreendem o tamanho da população, critérios evolutivos, estagnação e outros critérios de parada.

4.1.1. Adequação de Parâmetros MANFIS para AG's

As atualizações dos parâmetros *a*, *b*, *c* e *cz* serão geradas por critérios determinados pelos AG's, formando uma população de indivíduos.

O espaço de busca será determinado pelas fronteiras de cada parâmetro reunido na matriz **W**.

$$\mathbf{W} \rightarrow \begin{cases} a = 3\ x\ y \\ b = 3\ x\ y \\ c = 3\ x\ y \\ cz = 9\ z \end{cases} \quad (13)$$

Os elementos da matriz $\mathbf{W}_{[27x1]}$ formam o espaço de busca, estes serão sorteados aleatoriamente dentro dos domínios respectivos, de maneira que em cada geração se tenha uma população formada por um certo número de indivíduos. Os Domínios de cada parâmetro são:

$$a \rightarrow [0{,}1-5], \quad \Delta a_{máx} \leq 4{,}9 \quad (14)$$

$$b \rightarrow [0{,}1-5], \quad \Delta b_{máx} \leq 4{,}9 \quad (15)$$

$$c \rightarrow [^{-}5{,}1-5], \quad \Delta c_{máx} \leq 10{,}1 \quad (16)$$

$$cz \rightarrow [^{-}5{,}1-5], \quad \Delta cz_{máx} \leq 10{,}1 \quad (17)$$

Os AG's precisam da informação do valor de uma Função Objetivo do problema. O erro quadrático da saída do MANFIS assumirá este papel, muitas vezes nomeado como a energia de um estado.

$E(z) = f(z(a,b,c,cz))$ é a função minimante, onde, $z = \{z_1, z_2 ..., z_{121}\}$, então a energia do estado *i* pode ser representada por:

$$Erro_i = (zd - z_i)^2 = E_i \quad (18)$$

O MANFIS achará a energia para cada geração a que uma população é submetida.

4.1.2. Implementação do Algoritmo Genético

O algoritmo básico para AG apresenta a seguinte configuração:

- Gerar um conjunto de soluções viáveis para o problema dentro do espaço **W** de busca, restringidos pelas Eqs.(13 –17).
- A população é avaliada pela função objetivo (para cada indivíduo), quanto maior seja a adaptação do indivíduo no ambiente, maior será o valor da função objetivo e maiores serão as chances dele sobreviver no ambiente e se reproduzir, passando parte de seu material genético a gerações futuras.
- A seleção de indivíduos é feita pela probabilidade proporcional ao *fitness* relativa a cada indivíduo na população.
- Se faz o cruzamento entre indivíduos adotando um processo simples aleatório que ocorre com probabilidade fixa assumida.
- Realiza-se o processo de mutação selecionando uma posição num cromossomo e mudando o valor do gene correspondente aleatoriamente para um outro.
- O critério de parada assumido é de estagnação, um segundo critério é limitar um determinado número de gerações.

4.1.3. Resultados Obtidos com AG's e MANFIS

Os parâmetros do MANFIS foram implementados com bastante sucesso com 27 componentes as quais determinaram o espaço de busca.

O problema de otimização foi analisado para diversos tamanhos de populações. Obtendo um melhor comportamento e razoável custo computacional com uma população de 100 e 120 indivíduos. Um primeiro caso foi o de 100 indivíduos com uma função *fitness* inicial de 6 424,2388 atingindo até 12 662,4039.

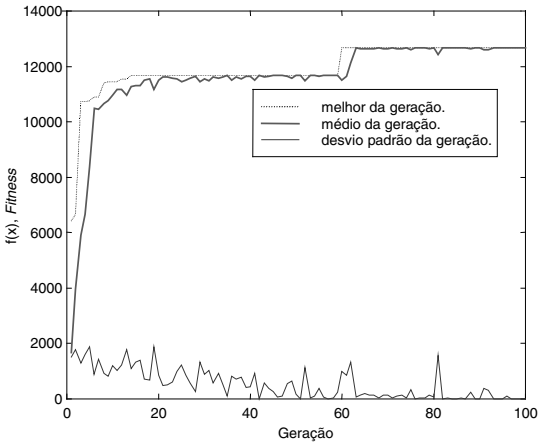

Figura 6: Desempenho do algoritmo ManFisAG

A Figura 6, mostra que o comportamento do algoritmo ManFisAG, nas primeiras gerações apresenta uma subida rápida das funções *fitness*, com tendências a estagnar-se no valor de ~11 300,0. Na figura, observa-se que a cada geração, a curva inferior corresponde ao desvio padrão, os picos indicam que a população foi alterada por mutações. Na geração 60 ocorre um caso especial, a mutação gerada mudou o patamar das funções *fitness* no sentido oposto que estas produzem. Neste caso, este indivíduo gerado é um super indivíduo significativo que altera a tendência do *fitness*.

Na Figura 7, a função teórica é superposta pela função aproximada MANFIS e ManFisAG (otimização por AG's). Observa-se que o ManFisAG apresenta melhoria na aproximação da função, com respeito ao próprio MANFIS (partes mais relevantes marcada por ovais).

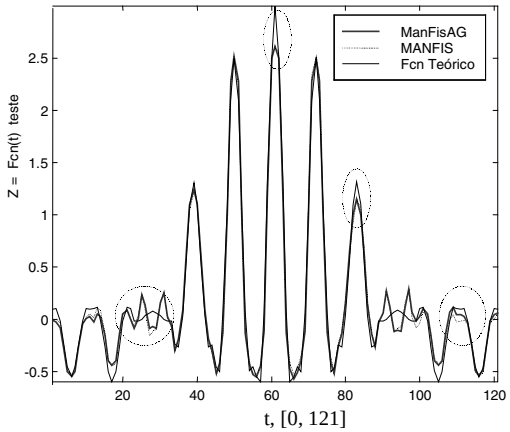

Figura 7: Comparação dos resultados pelo MANFIS e ManFisAG com a Fnc. teórica.

4.2. Otimização do MANFIS por SA

SA é uma ferramenta poderosa para problemas de otimização, usada em casos de grande complexidade. Foi introduzida por Metropolis et al [7], como um método de se determinar propriedades físico-químicas relacionadas a um grupo de átomos em transição para o equilíbrio térmico. Mais a frente Kirpatrick et al [5], tornou explícito o conceito de esquema de resfriamento. Já Geman e Geman [8], [5] implementaram limites inferiores para a velocidade de queda de temperatura, na condição de garantir a convergência do algoritmo para um mínimo global.

O processo de Metropolis é constituído basicamente de duas etapas. Na primeira procede-se à elevação inicial da temperatura a um estado de energia máxima e, na segunda, verifica-se o seu abaixamento sucessivo e suficientemente lento, para que as partículas do sistema se combinem de forma a atingirem o estado de energia mínima, isto é, de tal forma que seja atingido o equilíbrio térmico.

A simulação da evolução das soluções é baseada em técnicas de Monte Carlo e na geração de estados sucessivos. Supõe-se, como ponto de partida, um estado possuindo energia *Ei*, a partir do qual, recorrendo a um mecanismo apropriado é gerado um outro estado com energia E_j. Se a diferença de energias $E_j - E_i$ for menor ou igual a zero o novo estado é aceito como estado atual. Se a condição anterior não se verificar, o novo estado poderá ser ainda aceito, com uma probabilidade em função da diferença de energias entre estados sucessivos da temperatura do banho. A Eq. (19) apresenta o processo de cálculo dessa probabilidade:

$$p(Boltzmann) = e^{\frac{-(E_j - E_i)}{K_B T}} \qquad (19)$$

onde T é a temperatura e K_B é a constante de Boltzmann.

O processo de resfriamento simulado tem a particularidade de permitir o relaxamento transitório da otimalidade durante o processo de pesquisa. Este relaxamento é admitido na tentativa de evitar a convergência para mínimos locais.

4.2.1. Adequação dos Parâmetros MANFIS para SA

Sabe-se que o MANFIS adapta seus parâmetros (*a, b, c* e *cz*) pelo método de gradiente descendente e retro-propagação do erro. Estes determinam a função de saída (variável *z*), com um erro de aproximação. Os parâmetros *a, b,* c e *cz,* são os parâmetros que geram o estado *z(a,b,c,cz), cz* é o conjunto de centros de cada regra da camada de conseqüentes.

De modo que $\Delta z = f(\Delta a, \Delta b, \Delta c, \Delta cz)$, e o erro quadrático e será equivalente à função objetivo que se deseja minimizar $E(z) = f(z(a,b,c,cz))$.

$$z \rightarrow estado \quad S\,,\ z = \{z_1, z_2 ..., z_{121}\},$$

A matriz $\mathbf{W_{[27x1]}}$ é a mesma da equação (13) e contém os parâmetros que serão sorteados aleatoriamente dentro dos domínios respectivos conforme às equações (14)-(17). A energia de estado *i* 'E_i' é a mesma que na equação (18).

4.2.2. Implementação do Algoritmo SA

Um algoritmo básico para *Simulated Annealing* apresenta a seguinte configuração:

- Executa o MANFIS após o treino para gerar o estado inicial do sistema $z_{inicial}$ e a $E_{inicial}$.
- Assume uma temperatura inicial $T_{inicial.máx}>0$. A partir do estado e temperatura inicial, determina-se a constante de Boltzmann $K_B T = -\frac{\Delta E}{\ln(p)}$, assumindo uma probabilidade p alta e um ΔE pequeno.
- Enquanto não se faz o resfriamento do sistema:
  - Gera uma rotina para visitar os estados possíveis.
  - Escolhe um vizinho aleatório de um estado *z'* dentro dos domínios e restrições dos parâmetros. Neste caso foram sorteados os parâmetros para gerar o estado *z'*. Uma subrotina é ativada para gerar os Δ dos parâmetros, para atualizar e gerar os novos parâmetros e assim determinar o *z'*.
  - Determina-se, $\Delta E = f(z') - f(z) = E_j - E_i$.
  - Processo de aceitação de estados.
  - Se $\Delta E \leq 0$, fixar o estado $z \leftarrow z'$
  - No caso do $\Delta E > 0$, se aceita os estado $z \leftarrow z'$, mas com probabilidade $e^{\frac{-\Delta E}{T}}$.
- Ativa resfriamento após visitar a maioria dos estados.
- Finaliza por critério de parada.

4.2.3. Resultados Obtidos com SA e MANFIS

Os parâmetros do MANFIS foram implementados com bastante sucesso com 27 componentes, que podem ser tratados como variáveis aleatórias que implicam no

resultado final, na saída do defuzzificador, com 121 subestados em cada simulação.

A adequação dos parâmetros foi repassada convenientemente para gerar o estado inicial, assim como para implantar um esquema de resfriamento e o processo de visitação dos estados. Os casos de reinserção, reflexão, contração e extrapolação foram assistidos pelo método Downhill *simplex* [9], implicando em 27 vértices para resolver o *simplex*. Convergir o simplex implica em otimizar a saída do estado *z* e minimizar a função objetivo para zero.

O problema de otimização do MANFIS, para a função teste (Eq. 12), foi analisado para diversas temperaturas de começo e várias taxas de resfriamento. Um primeiro caso, com $T_{inicial} = 1{,}1$, $f(z) = E_i = 4{,}59$, assumindo K_B unitária e uma taxa de resfriamento por temperatura $T_{i+1} = 0{,}95T_i$. Atingindo o critério de parada com 1000 iterações, conseguindo convergir para um ótimo global tal como se mostra na Figura 8, a função objetivo alcançou 0,904 e a temperatura final foi $T_{final} = 0{,}06$.

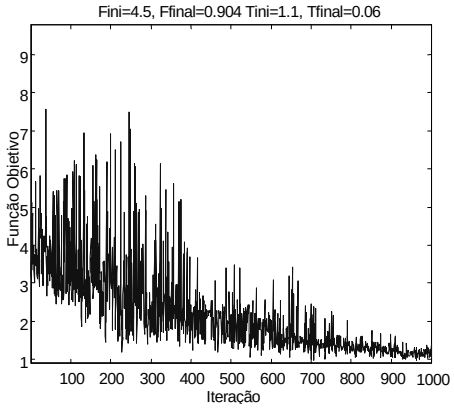

Figura 8: Desempenho do algoritmo ManFisSA

Na Figura 9, se comparam as funções teórica, MANFIS e aproximada pelo ManFisSA, onde a curva sólida é o resultado de aplicação SA no MANFIS. Observa-se que ela está bastante próxima à curva tracejada mais fina e que o ManFisSA teve melhor aproximação que o MANFIS.

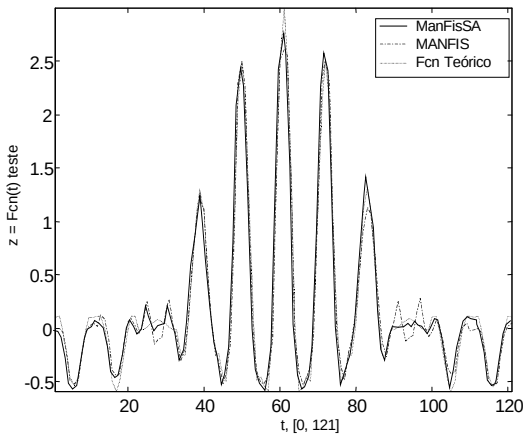

Figura 9: Comparação dos resultados pelo MANFIS e ManFisSA com a Fnc. teórica.

5. Conclusões

Os parâmetros do MANFIS *a*, *b*, *c* e *cz* (total 27) foram implementados com bastante sucesso tanto para adequação dos AG's como para SA.

Os resultados obtidos pelos AG's foram bastante satisfatórios. O desempenho do algoritmo ManFisAG, mostrado na Figura 6, onde o comportamento da função *Fitness* apresenta patamares bem definidos, revelam uma melhor aproximação da função teste, como mostra a Figura 7.

Os testes com SA também apresentaram resultados bastante satisfatórios. Foram analisados para diversos níveis de temperatura inicial e taxas de resfriamento. Em um dos casos analisados, Figura 8, o desempenho do algoritmo ManFisSA é qualificado pela minimização da função objetivo, onde se teve um valor inicial de 4,5 minimizado para 0,904.

SA e AG's podem ser aplicados para otimizar o MANFIS, cujos resultados mostram melhorias aceitáveis. Assim, o MANFIS associado com os AGs e/ou SA, torna-se uma técnica robusta para resolver problemas ainda mais complexos do que os analisados.

6. Agradecimentos

Os autores agradecem ao CEPEL, CNPq e SIEMENS (Brasil) pelo apoio fornecido para este trabalho.

Referências

[1] L. Cheim, W. Cuenca, S. Varricchio, 'A simple Mamdani Adaptive Neuro-Fuzzy Inference System (MANFIS)', *manuscrito Submetido à IEEE trans. on Fuzzy System*, 1998.

[2] Chin-Teng Lin, C. S. George Lee: *Neural Fuzzy Systems*, Prentice Hall, Upper Saddle River, NJ 07458, 1996.

[3] Huamán Cuenca W. "Aplicação de Sistemas Inteligentes no Reconhecimento de Padrões de Descargas Parciais em Transformadores de Potência". *Tese de Mestrado, Programa de Engenharia Elétrica* UFRJ COPPE 1998.

[4] Jang R., C.T. Sun, E. Muzitani: *Neuro-Fuzzy and Soft Computing*, Pretince Hall, USA 1997.

[5] Mendonça Paulo R. 'Novos Algoritmos de Simulated Annealing', *Tese*, UFRJ-COPPE 192/97.

[6] Tanumaro, J., ' Motivação, Fundamentos e Aplicações de Algoritmos Genéticos', Segundo Congresso Brasileiro de Redes Neurais, 1995.

[7] J. Pereira, et al. 'Identificação de Estratégias de Exploração de redes de Distribuição de Energia Eléctrica Utilizado Simulated Annealing', *4º. Encontro Luso-Afro-Brasileiro de Planejamento e Exploração de Redes de Energia*' ELAB'99.

[8] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, Inc., 2nd edition, 1999.

[9] Numerical Recipes in C. Section 10.4, 10.9 www.nr.com.