



UNIVERSIDADE FEDERAL DO PIAUÍ  
CAMPUS MINISTRO PETRÔNIO PORTELLA  
CENTRO DE TECNOLOGIA  
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA  
RÔMULLO RANDELL MACEDO CARVALHO

ANÁLISE DA NEURO-OTIMIZAÇÃO EM ALGORITMOS GLOBAIS  
SUPERVISIONADOS E APLICAÇÃO NA MODELAGEM DA  
DINÂMICA DE LASERS SEMICONDUTORES

TERESINA  
2019

RÔMULLO RANDELL MACEDO CARVALHO

ANÁLISE DA NEURO-OTIMIZAÇÃO EM ALGORITMOS GLOBAIS  
SUPERVISIONADOS E APLICAÇÃO NA MODELAGEM DA  
DINÂMICA DE LASERS SEMICONDUTORES

Dissertação apresentada como requisito parcial à obtenção do grau de mestre em Engenharia Elétrica, no Programa de Pós-Graduação em Engenharia Elétrica (Sistemas de Controle e Automação), na Universidade Federal do Piauí.

Orientador: Prof. Dr. Luis Gustavo Mota Souza.

TERESINA

2019

FICHA CATALOGRÁFICA  
Universidade Federal do Piauí  
Biblioteca Comunitária Jornalista Carlos Castello Branco  
Serviço de Processamento Técnico

C331a Carvalho, Rômulo Randell Macedo.  
Análise da neuro-otimização em algoritmos globais supervisionados e aplicação na modelagem da dinâmica de lasers semicondutores / Rômulo Randell Macedo Carvalho. – 2019.  
126 f.

Dissertação (Mestrado em Engenharia Elétrica) –  
Universidade Federal do Piauí, Teresina, 2019.  
“Orientador: Prof. Dr. Luis Gustavo Mota Souza”.

1. Algoritmo genético. 2. Lasers de pontos quânticos.  
3. Máquina de aprendizado extremo. 4. Otimização por enxame de partículas. 5. Perceptron de múltiplas camadas. I. Título.

CDD 621.3

RÔMULLO RANDELL MACEDO CARVALHO

ANÁLISE DA NEURO-OTIMIZAÇÃO EM ALGORITMOS GLOBAIS  
SUPERVISIONADOS E APLICAÇÃO NA MODELAGEM DA  
DINÂMICA DE LASERS SEMICONDUCTORES

Dissertação apresentada como requisito parcial à obtenção do grau de mestre em Engenharia Elétrica, no Programa de Pós-Graduação em Engenharia Elétrica (Sistemas de Controle e Automação), na Universidade Federal do Piauí.

Orientador: Prof. Dr. Luis Gustavo Mota Souza.

Aprovada em: \_\_/\_\_/\_\_\_\_.

BANCA EXAMINADORA

---

Prof. Dr. Luís Gustavo Mota Souza (Orientador)  
Universidade Federal do Piauí

---

Prof. Dr. José Maria Pires de Menezes Júnior (Interno)  
Universidade Federal do Piauí

---

Prof. Dr. Guilherme de Alencar Barreto (Interno)  
Universidade Federal do Ceará

---

Prof. Dr. George André Pereira Thé (Externo)  
Universidade Federal do Ceará

*A todos que ainda acreditam  
e investem na ciência.*

## AGRADECIMENTOS

A Deus, pelo dom da vida e pelo êxito neste desafio mais.

À minha família, em especial, a meus pais, Manoel e Roberta, e à minha irmã, Ramonyelle. Os anos de convívio trouxeram perseverança, fé e humildade.

À minha quase noiva Luciana, porque um poema pode ser escrito e recitado com apenas quatro palavras: nó, bagunça, sorriso e abraço.

Aos amigos Herbert e Breno. A vontade já estava lá, mas foi com vocês que dei o pontapé inicial neste vasto universo de neurônios e algoritmos.

Ao Programa de Pós-Graduação em Engenharia Elétrica da UFPI, pela receptividade e, sobretudo, pela oportunidade.

À Fundação de Amparo à Pesquisa do Estado do Piauí (FAPEPI) e à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pela concessão de bolsa durante doze dos vinte e sete meses de mestrado.

Ao meu orientador, Prof. Dr. Luis Gustavo Mota Souza, pela paciência e pela estima durante os anos de iniciação científica, trabalho de conclusão de curso e orientação de mestrado.

Aos Prof. Dr. José Maria Pires de Menezes Júnior, Prof. Dr. George André Pereira Thé e Prof. Dr. Guilherme de Alencar Barreto por participarem da banca de defesa dessa dissertação.

A uma infinidade de pessoas que compartilharam seu vasto conhecimento e me puseram sobre seus ombros, entre eles: Aguirre, Haykin, Huang, e seus pares.

A todos os amigos e familiares, conhecidos e desconhecidos, que, de alguma forma, ajudaram-me nessa caminhada, seus nomes foram omitidos (porém não a estima) por razões da finitude dos recursos papel, tinta e memória.

*“The Universe is not only queerer than we suppose,  
but queerer than we can suppose”  
— J. B. S. Haldane*

## RESUMO

O presente estudo associa as redes máquina de aprendizagem extremo (ELM) e *perceptron* de múltiplas camadas (MLP) a mecanismos inteligentes de otimização, o Algoritmo Genético e a Otimização por Enxame de Partículas, de maneira tal a obter modelos globais supervisionados mais eficientes (com menos neurônios na camada oculta). A modelagem da potência óptica de lasers semicondutores de pontos quânticos por meio de redes neurais com diferentes estratégias de aprendizagem tem se apresentado como uma alternativa eficaz à abordagem clássica a partir de métodos massivamente físico-experimentais, como as equações de taxa e a descrição por microestados. Por se tratarem de paradigmas do tipo caixa preta, as redes supervisionadas dispensam o profundo conhecimento do processo e o tempo de teorização demandado dos métodos clássicos, ao mesmo tempo que consegue entregar resultados satisfatórios, com base em recentes trabalhos. Para tanto, duas abordagens são estabelecidas para estudo dos modelos: a primeira com um valor fixo reduzido da quantidade de neurônios e a segunda com um valor variável dentro do treinamento e determinado pelo próprio algoritmo. Ambas as abordagens são estudadas para a aproximação de funções de uma e de duas variáveis, com análise dos impactos da otimização nas redes MLP e ELM, como em relação ao sobreajuste e subajuste. Por fim, os modelos elaborados são aplicados à modelagem da potência óptica de lasers de ponto quântico em função de degrau de corrente na entrada, usando-se autorregressão com entrada exógenas, para um banco de dados com variações de estado (fundamental e excitado) e temperatura. Todos os modelos são submetidos à validação pelo erro quadrático médio e por análise dos resíduos.

**Palavras-chave:** Lasers de pontos quânticos. *Perceptron* de múltiplas camadas. Máquina de aprendizado extremo. Otimização por enxame de partículas. Algoritmo genético.

## ABSTRACT

The present study associates Extreme learning machine (ELM) and multilayer perceptron (MLP) networks with intelligent optimization mechanisms, Genetic Algorithm and Particle Swarm Optimization, in order to obtain more efficient supervised global models (with fewer hidden layer neurons). Modeling the optical power of semiconductor quantum dot lasers by neural networks with different learning strategies has been an effective alternative to the classical approach from experimental methods such as rate equations and microstate description. Because they are black box paradigms, supervised networks dispense deep knowledge of the process and the theorizing time required of classical methods, while delivering satisfactory results based on recent work. Therefore, two approaches are established to study the models: the first with a reduced fixed amount of neurons and the second with a variable value within the training and determined by the algorithm itself. Both approaches are studied for the approximation of functions of one and two variables, with analysis of the impacts of optimization in MLP and ELM networks, as in relation to overfitting and underfitting. Finally, the elaborated models are applied to the modeling of the optical power of quantum dot lasers as a function of input current step, using regressive model with an external input, for a database with state (fundamental and excited) and temperature variations. All models are subjected to validation by the mean square error and residual analysis.

**Keywords:** Quantum dot lasers. Multilayer perceptron. Extreme learning machine. Particle swarm optimization. Genetic Algorithm.

## LISTA DE FIGURAS

|   |    |
|---|----|
| Figura 2.1 — Cavityde óptica de um laser e modos de cavityde.....   | 24 |
| Figura 2.2 — Exemplo de diagrama de bandas (a) de condução e (b) de valência.....                             | 27 |
| Figura 2.3 — Potência óptica parametrizada (20°C e 40°C).....   | 29 |
| Figura 2.4 — Potência óptica parametrizada (50°C).....  | 30 |
| Figura 2.5 — Representação em malha da potência óptica parametrizada.....                                     | 31 |
| Figura 2.6 — Função de uma variável (Problema 1). .....   | 33 |
| Figura 2.7 — Função de duas variáveis (Problema 2). .....   | 33 |
| Figura 3.1 — Esquema genérico da identificação de sistemas.....   | 35 |
| Figura 3.2 — Etapas da identificação de sistemas. ....  | 36 |
| Figura 3.3 — Redes <i>feedforward</i> . .....   | 41 |
| Figura 3.4 — Modelo de neurônio genérico atual.....   | 42 |
| Figura 3.5 — Função logística ( $a = 1, b = 1$ )......  | 43 |
| Figura 4.1 — Cruzamento uniforme. ....  | 56 |
| Figura 5.1 — Aproximação de função do Problema 1 pela ELM e pela GA-ELM ( $N_{hid}$ constante).....           | 68 |
| Figura 5.2 — Aproximação de função do Problema 1 pela PSO-ELM ( $N_{hid}$ constante). ....                    | 70 |
| Figura 5.3 — Evolução do erro de treinamento para o Problema 1 ( $N_{hid}$ constante). .....                  | 71 |
| Figura 5.4 — Aproximação de função do Problema 2 pela ELM e pela GA-ELM ( $N_{hid}$ constante).....           | 73 |
| Figura 5.5 — Aproximação de função do Problema 2 pela PSO-ELM ( $N_{hid}$ constante). ....                    | 74 |
| Figura 5.6 — Problema 1 solucionado pela GA-ELM e pela PSO-ELM ( $N_{hid}$ variável).....                     | 76 |
| Figura 5.7 — Problema 2 solucionado pela GA-ELM e pela PSO-ELM ( $N_{hid}$ variável).....                     | 76 |
| Figura 5.8 — Evolução do MSE, de $N_{hid}$ e de $f_{apt}$ no treinamento sem minimização (Problema 1). .....  | 77 |
| Figura 5.9 — Evolução do MSE, de $N_{hid}$ e de $f_{apt}$ no treinamento com minimização (Problema 1). .....  | 78 |
| Figura 5.10 — Evolução do MSE, de $N_{hid}$ e de $f_{apt}$ no treinamento com minimização (Problema 1). ..... | 79 |
| Figura 5.11 — Evolução do MSE, de $N_{hid}$ e de $f_{apt}$ no treinamento com minimização (Problema 2). ..... | 80 |
| Figura 5.12 — Validação do modelo ELM para o Problema 1 ( $N_{hid} = 5$ ).....                                | 81 |
| Figura 5.13 — Validação do modelo ELM para o Problema 1 ( $N_{hid} = 50$ ).....                               | 81 |
| Figura 5.14 — Validação do modelo GA-ELM para o Problema 1 ( $N_{hid}$ minimizado).....                       | 82 |
| Figura 5.15 — Validação do modelo PSO-ELM para o Problema 1 ( $N_{hid}$ minimizado).....                      | 82 |
| Figura 5.16 — Validação do modelo ELM para o Problema 2 ( $N_{hid} = 10$ ).....                               | 83 |

|   |     |
|---|-----|
| Figura 5.17 — Validação do modelo ELM para o Problema 2 ( $Nhid = 250$ ). .....                     | 83  |
| Figura 5.18 — Validação do modelo GA-ELM para função de duas variáveis ( $Nhid$ minimizado). .....  | 84  |
| Figura 5.19 — Validação do modelo GA-ELM para função de duas variáveis ( $Nhid$ minimizado). .....  | 84  |
| Figura 5.20 — Aproximação de função do Problema 1 pelas redes MLP, GA-MLP e PSO-MLP. ....           | 88  |
| Figura 5.21 — Aproximação de função do Problema 2 pelas redes MLP, GA-MLP e PSO-ELM. ....           | 89  |
| Figura 5.22 — Evolução do MSE da MLP (Problemas 1). .....   | 89  |
| Figura 5.23 — Evolução do MSE, de $Nhid$ e de $fapt$ nas redes GA-MLP e PSO-MLP (Problema 1). ..... | 90  |
| Figura 5.24 — Evolução do MSE da MLP (Problemas 2). .....   | 90  |
| Figura 5.25 — Evolução do MSE, de $Nhid$ e de $fapt$ nas redes GA-MLP e PSO-MLP (Problema 2). ..... | 91  |
| Figura 6.1 — Modelagem do laser QD a 20°C e ES por ELM, GA-ELM e PSO-ELM. ....                      | 100 |
| Figura 6.2 — Modelagem do laser QD a 20°C e GS por ELM, GA-ELM e PSO-ELM. ....                      | 101 |
| Figura 6.3 — Modelagem do laser QD a 40°C e ES por ELM, GA-ELM e PSO-ELM. ....                      | 102 |
| Figura 6.4 — Modelagem do laser QD a 40°C e GS por ELM, GA-ELM e PSO-ELM. ....                      | 103 |
| Figura 6.5 — Modelagem do laser QD a 50°C e ES por ELM, GA-ELM e PSO-ELM. ....                      | 104 |
| Figura 6.6 — Validação das redes ELM, GA-ELM e PSO-ELM. ....  | 105 |
| Figura 6.7 — Validação das redes GA-ELM e PSO-ELM com $Nhid$ minimizado. ....                       | 106 |
| Figura 6.8 — Modelagem do laser QD a 20°C e ES por MLP, GA-MLP e PSO-MLP. ....                      | 107 |
| Figura 6.9 — Modelagem do laser QD a 20°C e GS por MLP, GA-MLP e PSO-MLP. ....                      | 108 |
| Figura 6.10 — Modelagem do laser QD a 40°C e ES por MLP, GA-MLP e PSO-MLP. ....                     | 109 |
| Figura 6.11 — Modelagem do laser QD a 40°C e GS por MLP, GA-MLP e PSO-MLP. ....                     | 110 |
| Figura 6.12 — Modelagem do laser QD a 50°C e ES por MLP, GA-MLP e PSO-MLP. ....                     | 111 |
| Figura 6.13 — Validação das redes MLP, GA-MLP e PSO-MLP. ....                                       | 112 |
| Figura 6.14 — Validação das redes GA-MLP e PSO-MLP com $Nhid$ minimizado. ....                      | 113 |

## LISTA DE QUADROS

|   |    |
|---|----|
| Quadro 3.1 — Algoritmo MLP (duas camadas, treinamento local)..... | 47 |
| Quadro 3.2 — Algoritmo ELM. ....                                  | 51 |
| Quadro 4.1 — Algoritmo de seleção por torneio. ....               | 55 |
| Quadro 4.2 — Algoritmos de cruzamento. ....                       | 56 |
| Quadro 4.3 — Algoritmos de mutação.....                           | 57 |
| Quadro 4.4 — Algoritmo genético. ....                             | 59 |
| Quadro 4.5 — Algoritmo PSO. ....                                  | 63 |

## LISTA DE TABELAS

|   |     |
|---|-----|
| Tabela 2.1 — Características de corrente para cada temperatura.....               | 28  |
| Tabela 2.2 — Características das séries temporais da amostra. ....                | 28  |
| Tabela 2.3 — Características da amostra para as funções.....                      | 33  |
| Tabela 3.1 — Definição das etapas de modelagem deste estudo. ....                 | 52  |
| Tabela 5.1 — Parâmetros gerais dos algoritmos MLP, ELM e versões otimizadas. .... | 66  |
| Tabela 5.2 — Quantidade de neurônios escondidos. ....                             | 67  |
| Tabela 5.3 — Parâmetros reais de treinamento da ELM por AG (GA-ELM). ....         | 67  |
| Tabela 5.4 — Parâmetros reais de treinamento da ELM por PSO (PSO-ELM). ....       | 67  |
| Tabela 5.5 — Parâmetros comuns a GA-ELM e PSO-ELM.....                            | 74  |
| Tabela 5.6 — Parâmetros de treinamento binário da ELM por AG (GA-ELM).....        | 75  |
| Tabela 5.7 — Parâmetros de treinamento da ELM por PSO (PSO-ELM).....              | 75  |
| Tabela 5.8 — Comparativo entre ELM e variantes da GA-ELM. ....                    | 85  |
| Tabela 5.9 — Parâmetros do treinamento por <i>backpropagation</i> . ....          | 86  |
| Tabela 5.10 — Parâmetros de treinamento da MLP por AG (GA-MLP). ....              | 86  |
| Tabela 5.11 — Parâmetros de treinamento da MLP por PSO (PSO-MLP). ....            | 87  |
| Tabela 5.12 — Tempo de treinamento e MSE de teste médios (Problema 1). ....       | 92  |
| Tabela 5.13 — Tempo de treinamento e MSE de teste médios (Problema 2). ....       | 93  |
| Tabela 6.1 — Parametrização dos estados do laser nos ensaios. ....                | 95  |
| Tabela 6.2 — Parametrização das temperaturas dos ensaios. ....                    | 95  |
| Tabela 6.3 — Janela de regressão do modelo. ....                                  | 96  |
| Tabela 6.4 — Parâmetros gerais dos algoritmos. ....                               | 96  |
| Tabela 6.5 — Parâmetros comuns ao treinamento por AG e PSO.....                   | 97  |
| Tabela 6.6 — Parâmetros de treinamento por AG. ....                               | 97  |
| Tabela 6.7 — Parâmetros de treinamento por PSO.....                               | 98  |
| Tabela 6.8 — Parâmetros do treinamento do MLP por <i>backpropagation</i> . ....   | 98  |
| Tabela 6.9 — Tempo de treinamento e MSE de teste médios. ....                     | 113 |

## LISTA DE ABREVIATURAS E SIGLAS

|         |   |
|---------|---|
| AG      | Algoritmo genético  |
| AR      | <i>Autoregressive</i>                                       |
| ARX     | <i>Autoregressive with exogenous inputs</i>                 |
| ARMAX   | <i>AutoRegressive Moving Average with eXogenous inputs</i>  |
| ELM     | <i>Extreme learning machine</i>                             |
| FAC     | Função de autocorrelação                                    |
| GA-ELM  | <i>Genetic algorithm Extreme learning machine</i>           |
| GA-MLP  | <i>Genetic algorithm Multilayer perceptron</i>              |
| MISO    | <i>Multiple Inputs, Single Output</i>                       |
| MLP     | <i>Multilayer perceptron</i>                                |
| MMQ     | Método dos mínimos quadrados                                |
| MSE     | <i>Mean squared error</i>                                   |
| NAR     | <i>Nonlinear autoregressive</i>                             |
| NARX    | <i>Nonlinear autoregressive with exogenous inputs</i>       |
| NNARX   | <i>Neural Network autoregressive with exogenous inputs</i>  |
| PRBS    | <i>Pseudo random binary signal</i>                          |
| PSO     | <i>Particle swarm optimization</i>                          |
| PSO-ELM | <i>Particle swarm optimization Extreme learning machine</i> |
| PSO-MLP | <i>Particle swarm optimization Multilayer perceptron</i>    |
| QD      | <i>Quantum-dot</i>  |
| RNA     | Rede neural artificial                                      |
| SISO    | <i>Single input, single output</i>                          |

## LISTA DE SÍMBOLOS

|                      |   |
|----------------------|---|
| $\Delta w_{kj}$      | Atualização do peso sináptico $w_{kj}$  |
| $b_k$                | <i>Bias</i> do neurônio $k$   |
| $v_k$                | Campo local induzido do neurônio $k$  |
| $\varphi'(\cdot)$    | Derivada da função de ativação  |
| $u(t)$               | Entrada observada no instante $t$   |
| $\xi(t)$             | Erro da saída estimada no instante $t$  |
| $b$                  | Fator multiplicativo da amplitude da função de aptidão                                      |
| $f_{apt}(\cdot)$     | Função de aptidão   |
| $\varphi(\cdot)$     | Função de ativação  |
| $\delta_k$           | Gradiente local do neurônio $k$   |
| $a$                  | Inclinação da função de ativação  |
| $\mathbf{W}_{out}$   | Matriz de pesos sinápticos da camada de saída   |
| $\mathbf{W}_{hid}$   | Matriz de pesos sinápticos da camada escondida  |
| $\mu_X$              | Média da variável aleatória $X$   |
| $\mathbf{z}_{p,i}$   | Melhor vetor posição da partícula $i$ ( <i>personal best</i> )                              |
| $\mathbf{z}_g$       | Melhor vetor posição de todas as partículas da população ( <i>global best</i> )             |
| $\mathbf{Z}_p(n)$    | Matriz de melhor posição particular de cada indivíduo da população ( <i>personal best</i> ) |
| $U(\mu_X, \sigma^2)$ | Número aleatório gerado com média $\mu_X$ e variância $\sigma^2$ em distribuição uniforme   |
| $N(\mu_X, \sigma^2)$ | Número aleatório gerado com média $\mu_X$ e variância $\sigma^2$ em distribuição normal     |
| $N_{amt}$            | Número de amostras total  |
| $N_e$                | Número de épocas de treinamento do MLP  |
| $N_g$                | Número de gerações  |
| $N_{it}$             | Número de iterações   |
| $N_{out}$            | Número de neurônios na camada de saída  |
| $N_{hid}$            | Número de neurônios na camada oculta  |
| $N_{in}$             | Número de nós de entrada  |
| $w_{kj}$             | Peso sináptico do neurônio $k$ para a informação proveniente do nó $j$                      |
| $\mathbf{z}_i$       | Vetor posição da partícula $i$ da população, com dimensão $N_{dim}$                         |
| $y_k$                | Resposta a estímulos do neurônio $k$  |
| $\hat{y}(t)$         | Saída estimada pelo modelo no instante $t$  |
| $y(t)$               | Saída observada no instante $t$   |

|                 |   |
|-----------------|---|
| $\eta$          | Taxa de aprendizagem do MLP                                     |
| $E(X)$          | Valor esperado de uma variável aleatória $X$                    |
| $\sigma_X^2$    | Variância da variável aleatória $X$                             |
| $v_i$           | Velocidade da partícula $i$ da população                        |
| $\mathbf{Z}(n)$ | Matriz da posição das partículas da população, de tamanho $N_p$ |
| $\mathbf{V}(n)$ | Matriz da velocidade dos indivíduos da população                |
| $\mathbf{x}(t)$ | Vetor de entrada no instante $t$                                |

## SUMÁRIO

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>INTRODUÇÃO.....</b>   | <b>18</b> |
| 1.1      | CONTEXTUALIZAÇÃO DO TEMA .....                                       | 18        |
| 1.1.1    | Otimização do MLP.....   | 18        |
| 1.1.2    | Otimização da ELM.....   | 19        |
| 1.1.3    | Modelagem clássica de laser de pontos quânticos.....                 | 20        |
| 1.1.4    | Modelagem de lasers semicondutores com aprendizagem de máquina ..... | 21        |
| 1.2      | MOTIVAÇÃO .....  | 21        |
| 1.3      | OBJETIVOS .....  | 22        |
| 1.3.1    | Objetivo geral .....   | 22        |
| 1.3.2    | Objetivos específicos .....  | 22        |
| 1.4      | PRODUÇÃO CIENTÍFICA .....  | 23        |
| 1.5      | ESTRUTURA DO TRABALHO .....  | 23        |
| <b>2</b> | <b>LASERS SEMICONDUCTORES DE PONTO QUÂNTICO.....</b>                 | <b>24</b> |
| 2.1      | LASERS: DEFINIÇÃO E APLICAÇÕES.....                                  | 24        |
| 2.2      | MATERIAL SEMICONDUTOR EM LASERS .....                                | 25        |
| 2.3      | LASERS DE PONTO QUÂNTICO .....                                       | 26        |
| 2.4      | DESCRIÇÃO DO PROCESSO E DO BANCO DE DADOS .....                      | 27        |
| 2.5      | PROBLEMAS COMPLEMENTARES: A APROXIMAÇÃO DE FUNÇÕES .....             | 32        |
| <b>3</b> | <b>MODELOS GLOBAIS SUPERVISIONADOS .....</b>                         | <b>35</b> |
| 3.1      | ETAPAS DA IDENTIFICAÇÃO DE SISTEMAS.....                             | 35        |
| 3.1.1    | Aquisição de dados .....   | 36        |
| 3.1.2    | Representação matemática.....  | 36        |
| 3.1.3    | Estrutura do modelo.....   | 37        |
| 3.1.4    | Estimação de parâmetros .....  | 38        |
| 3.1.5    | Validação de modelos.....  | 38        |
| 3.1.5.1  | <i>Mensuração do erro na saída.....</i>                              | <i>38</i> |
| 3.1.5.2  | <i>Análise dos resíduos .....</i>                                    | <i>39</i> |
| 3.2      | <i>PERCEPTRON</i> DE MÚLTIPLAS CAMADAS .....                         | 41        |
| 3.2.1    | A aprendizagem em redes neurais .....                                | 41        |
| 3.2.2    | Descrição matemática do neurônio.....                                | 42        |
| 3.2.3    | Algoritmo de retropropagação do erro .....                           | 43        |
| 3.2.4    | Aspectos ajustáveis do MLP.....                                      | 45        |
| 3.3      | MÁQUINA DE APRENDIZADO EXTREMO .....                                 | 48        |
| 3.3.1    | Representação matricial na propagação de informações .....           | 48        |
| 3.3.2    | Algoritmo de aprendizagem da ELM .....                               | 49        |

|          |  |           |
|----------|--|-----------|
| 3.3.3    | Considerações sobre a ELM .....                            | 50        |
| 3.4      | MODELO NNARX .....   | 52        |
| <b>4</b> | <b>MECANISMOS DE OTIMIZAÇÃO.....</b>                       | <b>53</b> |
| 4.1      | ALGORITMO GENÉTICO E SEUS OPERADORES .....                 | 53        |
| 4.1.1    | Representação do cromossomo.....                           | 53        |
| 4.1.2    | Métodos de seleção em AGs.....                             | 54        |
| 4.1.2.1  | <i>Pressão de seleção</i> .....                            | 54        |
| 4.1.2.2  | <i>Seleção por torneio</i> .....                           | 54        |
| 4.1.3    | Cruzamento e mutação .....                                 | 55        |
| 4.1.3.1  | <i>Cruzamento uniforme</i> .....                           | 55        |
| 4.1.3.2  | <i>Método BLX-alfa</i> .....                               | 55        |
| 4.1.3.3  | <i>Mutação</i> .....                                       | 57        |
| 4.1.4    | Elitismo.....  | 57        |
| 4.1.5    | Convergência do algoritmo.....                             | 58        |
| 4.2      | OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS .....                  | 59        |
| 4.2.1    | Posição e velocidade das partículas .....                  | 59        |
| 4.2.2    | Atualização do enxame .....                                | 60        |
| 4.2.3    | Considerações sobre convergência .....                     | 61        |
| 4.3      | O TREINAMENTO NEURO-OTIMIZADO DAS REDES NEURAIS.....       | 64        |
| 4.3.1    | Pesos (codificação real) .....                             | 64        |
| 4.3.2    | Número de neurônios ocultos (codificação binária).....     | 65        |
| <b>5</b> | <b>NEURO-OTIMIZAÇÃO PARA APROXIMAÇÃO DE FUNÇÃO .....</b>   | <b>66</b> |
| 5.1      | ELM COM TREINAMENTO DA CAMADA OCULTA: GA-ELM E PSO-ELM.... | 66        |
| 5.1.1    | <i>Nhid</i> constante .....                                | 66        |
| 5.1.2    | <i>Nhid</i> variável.....                                  | 74        |
| 5.1.3    | Evolução de MSE e <i>Nhid</i> .....                        | 75        |
| 5.1.4    | Validação por análise dos resíduos.....                    | 81        |
| 5.2      | MLP: CAMADAS OCULTA E DE SAÍDA TREINADAS POR AG E PSO..... | 85        |
| 5.3      | O CUSTO DA OTIMIZAÇÃO .....                                | 91        |
| 5.4      | CONSIDERAÇÕES E RESSALVAS .....                            | 94        |
| <b>6</b> | <b>NEURO-OTIMIZAÇÃO APLICADA A LASERS QD.....</b>          | <b>95</b> |
| 6.1      | CONSIDERAÇÕES DE MODELO E DO BANCO DE DADOS .....          | 95        |
| 6.2      | PARÂMETROS DE TREINAMENTO DOS MODELOS .....                | 97        |
| 6.3      | MODELAGEM POR ELM, GA-ELM E PSO-ELM .....                  | 99        |
| 6.3.1    | Modelagem por MLP, GA-MLP e PSO-MLP.....                   | 106       |
| 6.4      | O CUSTO DA OTIMIZAÇÃO .....                                | 113       |

|          |                                   |            |
|----------|-----------------------------------|------------|
| 6.5      | CONSIDERAÇÕES E RESSALVAS .....   | 114        |
| <b>7</b> | <b>CONSIDERAÇÕES FINAIS .....</b> | <b>115</b> |
| 7.1      | CONCLUSÃO .....                   | 115        |
| 7.2      | LIMITAÇÕES .....                  | 117        |
| 7.3      | TRABALHOS FUTUROS .....           | 118        |
|          | <b>REFERÊNCIAS.....</b>           | <b>119</b> |

# 1 INTRODUÇÃO

*"Se vi mais longe foi por estar de pé sobre ombros de gigantes."*<sup>1</sup> — Isaac Newton

A modelagem de lasers ópticos a partir de técnicas de aprendizagem de máquinas tem ainda um vasto potencial a ser explorado na literatura. Os métodos clássicos demandam alto grau de conhecimento do processo e da composição do material com o qual o laser é produzido (THÉ, 2010); por outro lado, métodos inteligentes de identificação de sistemas dependem de pouco ou nenhum desse conhecimento prévio (BRAGA, 2000). A seguir é discutido sucintamente o contexto atual da modelagem de lasers semicondutores, de modelos a partir de redes neurais com treinamento otimizado e da aplicação destas naquela.

## 1.1 CONTEXTUALIZAÇÃO DO TEMA

### 1.1.1 Otimização do MLP

O *perceptron de múltiplas camadas (MultiLayer Perceptron, MLP)* teve seu algoritmo primeiramente descrito por Rumelhart, Hinton e Williams (1985, 1986). O método tradicional de aprendizagem da rede é por retropropagação do erro, descrito na Seção 3.2.

Da década de 1980 até hoje, a recorrência do MLP em trabalhos acadêmicos segue uma de duas abordagens: uso do algoritmo com o treinamento por retropropagação do erro para solucionar algum problema de identificação ou classificação; ou aplicação de nova metodologia de aprendizagem, porém utilizando a rede treinada tradicionalmente como parâmetro. A seguir, elencam-se trabalhos recentes que usam otimização como treinamento, por algoritmo genético (AG), ou por otimização por enxame de partículas (*Particle Swarm Optimization, PSO*):

#### a) MLP otimizado por AG:

- Luo et al (2017) elaboraram modelo para classificar exemplos de sinal de eletromiografia de superfície;
- ALTobi et al (2019) aplicaram à diagnóstico de falha de bomba centrífuga;
- Lima et al (2018) abordaram a classificação de movimentos de mão a partir de sinais eletromagnéticos;
- Kar, Nayak e Nayak (2016) elaboraram modelo de previsão de séries temporais financeiras; entre outros estudos.

#### b) MLP otimizado por PSO:

---

<sup>1</sup> Traduzido do inglês: *"If I have seen further it is by standing on the shoulders of Giants"*. Trecho de carta de Isaac Newton (1643-1727) para Robert Hooke (1635-1703), datada de 5 de fevereiro de 1676..

- Das et al (2015) aplicaram à tarefa de classificação de mineração de dados;
- Chatterjee et al (2017) abordaram classificação da dengue usando dados de expressão gênica;
- Kanungo et al (2015) utilizaram uma versão melhorada da otimização por partícula de enxame para MLP aplicada à classificação; entre outros trabalhos.

### 1.1.2 Otimização da ELM

A máquina de aprendizagem extrema (*Extreme Learning Machine*, ELM) foi proposta por Huang, Zhu e Siew (2004), uma RNA com aprendizado alternativo ao treinamento iterativo típico das redes multicamadas até então. Contudo, para tanto, o algoritmo não atualiza os pesos da camada oculta e, por isso, apresenta algumas limitações, como precisar de mais neurônios<sup>2</sup>.

Com base nas primeiras descrições da ELM, por Huang, Zhu e Siew (2004, 2006), outros autores têm, por diversas vezes, proposto variações, inclusive com associações a algoritmos evolutivos. Essas novas versões alteram distintos aspectos da rede, como a otimização do número de neurônios — para a produção de redes compactas —, a dos pesos e do bias da camada escondida e o suporte a entradas complexas (CHRISTOU, 2018).

Apenas um ano após o primeiro trabalho sobre a ELM, Zhu et al (2005) sugeriram otimizar bias e pesos da camada escondida de uma rede ELM pelo algoritmo de evolução diferencial (*Differential Evolution*, DE), a fim de resolver quatro problemas de classificação e um de aproximação de função. A rede E-ELM (*Evolutionary ELM*), como foi alcunhada, comparada a algoritmos iterativos de redes neurais (MLP treinada por *backpropagation*), apresentou velocidade de aprendizagem mais rápida e erro no teste mais acurado; comparada à ELM convencional, obteve uma arquitetura de rede mais compacta. Outros algoritmos híbridos com a ELM têm sido desenvolvidos para solucionar problemas específicos:

#### a) ELM otimizada por AG:

- Alexandre et al (2015) propuseram modelo para a reconstrução de dados de ondas ausentes em atividades marítimas;
- Yang et al (2013) abordaram o problema de despacho econômico;
- Alencar, Rocha Neto e Gomes (2016) utilizaram o algoritmo genético como método de poda de neurônios da rede;
- Aybar-Ruiz et al (2016) elaboraram modelo de previsão de radiação solar a partir de entradas climáticas; entre outros trabalhos.

---

<sup>2</sup> Mais detalhes na seção 3.3.

## b) ELM otimizada por PSO:

- Xu e Shu (2006) propuseram o algoritmo;
- Gritti et al (2015) apresentaram associação com identificação por meio de modelo de Hammerstein-Wiener;
- Ahila, Sadasivam e Manimala (2015) aplicaram na classificação de distúrbios no sistema de potência;
- Zhang e Mei (2016), utilizaram para a avaliação em tempo real da qualidade da água dos rios; entre outros estudos.

## 1.1.3 Modelagem clássica de laser de pontos quânticos

As abordagens clássicas de identificação de lasers semicondutores de pontos quânticos utilizam equações de taxa ou cálculos cinéticos microscópicos, baseadas massivamente em um tratamento físico-experimental (THÉ, 2010). Para certos lasers, a descrição por microestados produz modelos mais precisos, em decorrência de um íntimo estudo da matéria que os constitui, enquanto o método por equação de taxas, mais antigo, exige menor grau de conhecimento do processo, sob pena de redução na precisão dos resultados, apesar de, em geral, satisfatórios.

As equações de taxa, originalmente operadores de flutuação para descrição da natureza quântica das transições, são ajustáveis ao comportamento de lasers de pontos quânticos, por meio de modelos com eficiência de injeção unitária (HAUG, 1969; FAIST, 2003). Yokoyana e Brorson (1989) e Bjork e Yamamoto (1991) usaram equações de taxa para descrever, nessa ordem, as propriedades de saída de lasers ópticos e de lasers semicondutores de microcavidade; Simondi-Teisseire et al (1996) usaram o formalismo do método para prever a proporção populacional de compostos em laser semicondutor e sua influência na resposta do processo.

Baseado em um hamiltoniano<sup>3</sup> microscópico, Gies et al (2007) descreveram um modelo de semicondutores para pontos quânticos incorporados em microcavidades, com desvios em relação às abordagens atômicas mais tradicionais. Na mesma toada, Mal et al (2018) analisaram a relação entre fotoluminescência, temperatura e potência, da estrutura de banda eletrônica de pontos quânticos (no estado fundamental e no primeiro excitado) com um hamiltoniano, usando o modelo  $k \cdot p$  de oito bandas. Outros trabalhos recentes que usam hamiltoniano ou o modelo  $k \cdot p$  de oito bandas para estruturas de pontos quânticos são: Xiong e Zhang (2017), Jung, Kum e Hwang (2018) e Ghadi et al (2018).

---

<sup>3</sup> Em mecânica clássica, um sistema hamiltoniano é um sistema físico no qual as forças são invariantes da velocidade. Esse mesmo modelo pode ser utilizado matematicamente em outras abordagens, desde que o sistema possa ser descrito por equações diferenciais na forma das equações de Hamilton.

### 1.1.4 Modelagem de lasers semicondutores com aprendizagem de máquina

Os dados experimentados do laser modelado nesse estudo foram obtidos por Stefan Breuer e Wolfgang Elsaesser, do Semiconductor Optics Group da Technical University of Darmstadt, durante a pesquisa de Thé (2010). O banco de dados possui conjuntos de séries temporais para diferentes estados de excitação e temperaturas do material do laser e é apresentado em maiores detalhes na Seção 2.4.

Carvalho (2017) e Santos (2017) utilizaram esses dados para modelagem por meio de redes neurais considerando cada par temperatura-estado como um modelo distinto a ser obtido: o primeiro com modelos supervisionados MLP e ELM; e o segundo com não supervisionados LLM e KSOM.

Carvalho (2018) estendeu o estudo propondo a modelagem por meio da rede ELM com os pesos da camada oculta da rede otimizada por um AG, com um número de neurônios constante predefinida. Essa adaptação para treinamento dos pesos ocultos da ELM proporcionou modelos com estruturas menores e mais eficientes (menos neurônios).

Santos (2018), por outro lado, reuniu os conjuntos de séries temporais para diferentes temperaturas e estados em um mesmo modelo. As redes utilizadas no estudo foram: MLP, ELM e rede com estados do eco (*Echo State Network*, ESN), além de modelos híbridos entre si.

## 1.2 MOTIVAÇÃO

A modelagem clássica de lasers semicondutores de ponto quântico é realizada a partir de uma de duas metodologias: por equações de taxa ou por descrição em microestados, ambas baseadas na apreciação do problema por uma abordagem física do processo, a partir de conhecimentos teórico e experimental. Esse tratamento apresenta dois vieses notórios:

- c) complexo conhecimento do processo;
- d) análise minuciosa e demorada do sistema e de suas partes.

Por outro lado, em estudos anteriores (CARVALHO, 2017; SANTOS, 2017; CARVALHO, 2018; SANTOS, 2018), modelos autorregressivos baseados nas redes supervisionadas MLP e ELM, entre outras, foram obtidos satisfatoriamente para identificação do banco de dados da pesquisa de Thé (2010) de laser semicondutor de ponto quântico.

A área temática parece promissora, com bastante espaço ainda a ser explorado na literatura. Essa dissertação pretende dar continuidade com o estudo da identificação do banco de dados a partir da inclusão de modelos híbridos entre redes neurais e algoritmos otimizadores inteligentes.

### 1.3 OBJETIVOS

#### 1.3.1 Objetivo geral

Este estudo pretende avaliar os impactos da neuro-otimização por AG e por PSO no treinamento dos algoritmos globais supervisionados ELM e MLP, bem como utilizar essa abordagem na identificação da dinâmica de lasers semicondutores de ponto quântico e para modelagem de suas características. Almeja-se responder a seguinte indagação: os modelos neurais otimizados são capazes de generalizar o comportamento do sistema com mais eficiência que seus pares treinados convencionalmente?

Para a pergunta norteadora, considera-se mais eficiente o modelo com resultados satisfatórios que utilize estruturas menos complexas e tenha menor custo computacional na etapa de teste. Especificamente neste trabalho, usa-se de critério para esta avaliação a quantidade de neurônios na camada oculta das redes: quanto menor, maior a eficiência.

#### 1.3.2 Objetivos específicos

Para desenvolvimento do objetivo geral deste trabalho, algumas etapas específicas devem ser cumpridas:

- a) estudar as etapas da identificação de sistemas dinâmicos para construção e validação de modelos;
- b) discutir a teoria e o algoritmo das redes MLP e ELM, bem como dos mecanismos de otimização AG e PSO;
- c) reproduzir os algoritmos tradicionais da MLP e da ELM como parâmetro de comparação para os demais modelos;
- d) desenvolver aproximadores de função baseados em RNAs supervisionadas otimizadas por AG e PSO;
- e) avaliar os impactos da neuro-otimização nas redes ELM e MLP: eficiência, tempo de processamento, entre outros;
- f) mapear a resposta de potência óptica de lasers semicondutores de ponto quântico em estado fundamental e excitado em diferentes temperaturas para degraus de corrente na entrada de distintas amplitudes;
- g) realizar a modelagem da potência óptica de lasers quânticos a partir dos algoritmos neuro-otimizados desenvolvidos;
- h) promover a divulgação acadêmica dos resultados obtidos por meio de publicação em congressos ou periódicos.

## 1.4 PRODUÇÃO CIENTÍFICA

Até o momento de submissão à avaliação por banca do Programa de Pós Graduação em Engenharia Elétrica, da Universidade Federal do Piauí, o presente estudo contribuiu ativamente para publicação do seguinte artigo:

CARVALHO, R. R. M.; SOUZA, L. G. M.; THÉ, G. A. P. Mapeamento da dinâmica de laser semiconductor por meio de sistema híbrido neuro-evolutivo. In: **XXII Congresso Brasileiro de Automática**. João Pessoa, 2018.

## 1.5 ESTRUTURA DO TRABALHO

Além desta primeira seção introdutória, o trabalho apresenta mais outras seis — incluso considerações finais. Procurou-se dividir esse estudo em um arcabouço teórico das temáticas indispensáveis ao seu escopo nas três seções subsequentes — a identificação de sistemas, a configuração de redes neurais e a otimização por meio de algoritmos inteligentes —, bem como construção e impacto, aplicação em estudo de caso e conclusão a respeito dos algoritmos implementados nas três seções finais.

Na **primeira seção**, a Introdução, situa-se o presente estudo no contexto da literatura atual, bem como elucida os objetivos pleiteados ao longo da pesquisa. O texto da **Seção 2** trata de uma breve introdução à teoria de modelagem e suas etapas, com destaque para a validação de modelos e os bancos de dados utilizados neste trabalho.

Na **Seção 3**, são apresentados noções gerais e implementação dos algoritmos neurais supervisionados (MLP e ELM), enquanto, na **Seção 4**, discutem-se os mecanismos de otimização (AG e PSO). Na **Seção 5**, as redes neurais otimizadas são utilizadas para a aproximação de função, uma aplicação controlável e útil para analisar o impacto da otimização nos modelos globais supervisionados.

A **Seção 6** dispõe o estudo de caso; pela abordagem caixa-preta, a discussão a respeito de lasers semicondutores cede espaço aos modelos obtidos e à sua análise, gráfica e estatística. Na **Seção 7** — a última —, expõe-se um conciso encerramento do trabalho: conclusões relativas ao estudo apresentado, limitações dos resultados elencados e possíveis trabalhos futuros.

## 2 LASERS SEMICONDUTORES DE PONTO QUÂNTICO

*“E aqueles que foram vistos dançando foram julgados insanos por aqueles que não podiam escutar a música.”<sup>1</sup>*  
— Friedrich Wilhelm Nietzsche

Nesta seção, discorre-se sucintamente a respeito dos principais aspectos de lasers semicondutores de pontos quânticos (*Quantum Dots*, QD): composição, aplicações e características. Além disso, apresenta-se o banco de dados para modelagem na Seção 6, assim como os problemas de aproximações utilizados para pré-avaliar os modelos neuro-otimizados na Seção 5.

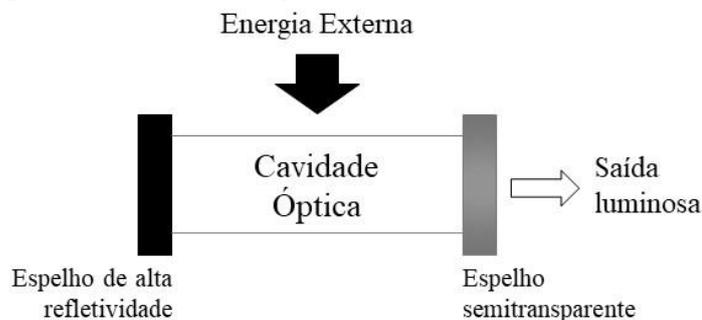
### 2.1 LASERS: DEFINIÇÃO E APLICAÇÕES

O termo laser tem origem no acrônimo de amplificação da luz pela emissão estimulada de radiação (em inglês: *Light Amplification by Stimulated Emission of Radiation*) e pode ser definido como

[...] um amplificador óptico capaz de gerar um feixe de luz intenso, direcionado e com uma frequência bem-definida (luz monocromática). Ele consiste, basicamente, de um meio ativo, que pode ser sólido líquido ou gasoso, onde a luz é gerada e amplificada, inserido entre dois espelhos paralelos ou arranjos [...] (VALADARES; ALVES; CHAVES, 2005, p. 35-36).

A importância de dois espelhos paralelos para a estrutura do laser (Figura 2.1) é a reflexão contínua entre as duas superfícies de forma a confinar e, conseqüentemente, concentrar e amplificar a luz. Um dos espelhos é semitransparente para que haja a passagem de uma fração do feixe incidente ao meio (e produzindo a saída óptica do laser) enquanto o outro deve possuir uma alta refletividade.

Figura 2.1 — Cavityde óptica de um laser e modos de cavityde.



Fonte: Próprio autor.

<sup>1</sup> Aforismo célebre de Nietzsche. Original em alemão: “*Diejenigen, die tanzten wurden für verrückt gehalten, von denen, die die Musik nicht hören konnten*”.

As características luminosas de emissão dos lasers propiciam que eles sejam úteis em diversas áreas: telecomunicações, indústria, medicina, metrologia e pesquisa científica. Atualmente, três tipos tem maior aplicação industrial:

a) CO<sub>2</sub>:

- soldagem, cortes e usinagem de materiais;
- tratamento térmico de metais;
- cortes cirúrgicos, cauterização e remoção de camadas da pele.

b) neodímio YAG (Nd:YAG):

- processamento de materiais: usinagem, pontos de solda e marcação;
- aplicações cirúrgicas;
- bombeamento de lasers sintonizáveis operando no visível, espectroscopia e sensoriamento remoto.

c) excimer (conhecido como laser frio, por não gerar calor):

- cirurgias oculares para remoção de camadas extremamente finas de tecido da córnea sem afetar regiões adjacentes;
- fonte ultravioleta em processos litográficos.

A depender do tamanho, da sofisticação e da versatilidade, esses equipamentos podem chegar a custar milhões de dólares (VALADARES; ALVES; CHAVES, 2005).

## 2.2 MATERIAL SEMICONDUTOR EM LASERS

Entretanto, para que funcione, o material do laser deve ser capaz de excitar um número mínimo de elétrons para um nível de energia superior e, em geral, estes tendem a ocupar os níveis de menor energia do átomo. Nesse sentido, materiais semicondutores são atrativos para confecção de lasers, uma vez que são passíveis de um excelente controle de condutibilidade, entre os metais (em que os elétrons se excitam muito facilmente) e isolantes elétricos (que exigiriam muita energia para se excitarem).

A partir de algum fator externo, um elétron pode ser transferido para o nível superior de energia e, posteriormente retornar, emitindo fótons. Essa emissão induz outros elétrons excitados a também emitirem um fóton idêntico ao incidente (emissão estimulada), criando um efeito cascata.

Entretanto, caso haja poucos elétrons excitados, os fótons emitidos são absorvidos por outros não-excitados e haverá um processo de decaimento pela excitação eletrônica e não de

amplificação. Para que este se torne relevante a ponto de criar uma reação em cadeia maciça, a quantidade de átomos no estado excitado deve ser maior que a de não-excitados: esse processo de inversão de população é o que permite o funcionamento de um laser.

### 2.3 LASERS DE PONTO QUÂNTICO

A fabricação de um laser semicondutor como aquele cujo banco de dados é explorado no presente estudo emprega em sua fabricação heteroestruturas de ponto quântico (*quantum dot*, QD) (Thé, 2010). De acordo com Dias (2011), essas estruturas são oriundas de um método que tem sido bastante desenvolvido, a Epitaxia por Feixe Molecular (*Molecular-Beam Epitaxy*, MBE), a qual pode originar ainda os poços quânticos (*quantum wells*, QWs) e os fios quânticos (*quantum wires*, QWs).

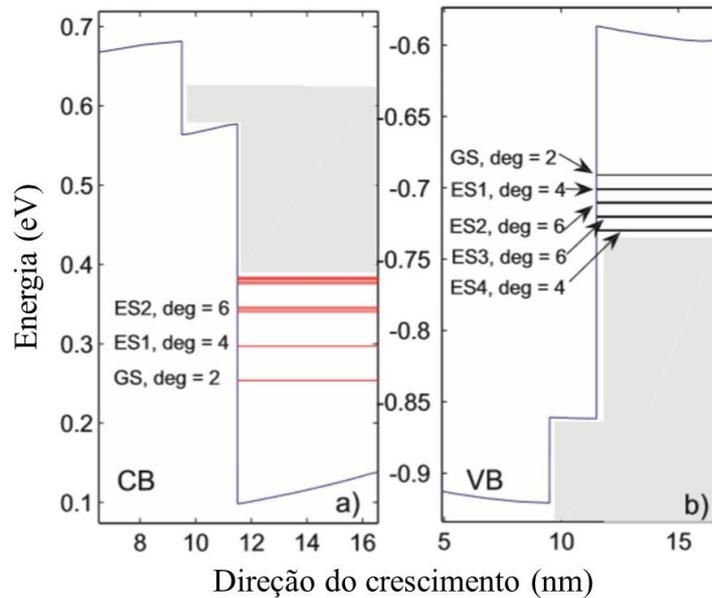
A MBE é um processo para deposição epitaxial de filmes finos, com uma ou mais fontes de feixes moleculares incidindo sobre uma superfície cristalina, o substrato. O método foi desenvolvido no final dos anos 1960 na Bell Telephone Laboratories por J. R. Arthur e Alfred Y. Cho, sendo de uso amplo na fabricação de semicondutores, como transistores, e considerado fundamental para o desenvolvimento de equipamentos eletrônicos e nanotecnologias (CHO; ARTHUR, 1975; WANG; LU, 2013).

Semicondutores QD, em síntese, são cristais nanométricos de banda proibida (*bandgap*) estreita inseridos em uma matriz semicondutora de banda proibida maior. O objetivo consiste em ter maior controle sobre a condução de elétrons, apesar de a tornar mais custosa; nesse sentido, Lasers utilizam materiais semicondutores em prol de maior precisão do processo, ou seja, para que a condução de corrente no material seja até certo ponto controlável (THÉ, 2010).

Esse processo decorre da formação de uma barreira de potencial tridimensional no material, em que se passa a conduzir em um espectro discreto de estados de energia. Na banda proibida, existem duas zonas de energia: a banda de valência (*valence band*, VB) e a banda de condução (*conduction band*, CB); os elétrons podem se encontrar em qualquer uma das duas, sendo necessário uma energia mínima para que saiam da primeira em direção à segunda.

Na Figura 2.2, é apresentado um diagrama de CB e VB desenvolvido por Thé (2010) em simulações. No diagrama é possível observar que o material semicondutor do laser simulado apresenta um estado fundamental (*ground state*, GS) e dois excitados (*excited states*, ES<sub>1</sub> e ES<sub>2</sub>) para a CB, além de um GS e quatro excitados (ES<sub>1</sub>, ES<sub>2</sub>, ES<sub>3</sub> e ES<sub>4</sub>) para a VB; os demais estados excitados (de ordem superior para a CB e inferior para a VB), na área sombreada, são considerados formantes de um *quasicontinuum* (QC) — uma região aproximadamente contínua em quase todos os aspectos.

Figura 2.2 — Exemplo de diagrama de bandas (a) de condução e (b) de valência.



Fonte: adaptada de Thé (2010).

#### 2.4 DESCRIÇÃO DO PROCESSO E DO BANCO DE DADOS

O conjunto de dados do laser modelado nesse estudo é oriundo do trabalho de Thé (2010) e inclui:

- resposta em potência óptica;
- estímulo em degrau abrupto de corrente com patamares inferior  $I_{bottom}$  e superior  $I_{top}$ , ambos em mA;
- três perfis de temperatura de operação: 20°C, 40°C e 50°C; e
- dois perfis de estado de excitação: GS e ES<sub>1</sub>. Como os demais estados excitados superiores a ES<sub>1</sub> não são abordados, este é denotado apenas por ES deste ponto em diante deste trabalho.

Nesse estudo, os modelos consideram o sistema de múltiplas entradas (corrente, temperatura e estado) e uma saída (potência óptica) (*Multiple Inputs, Single Output, MISO*). O banco de dados do laser semiconductor inclui cinco disposições em função da temperatura e do estado: para 20°C e 40°C, há amostras de ES e GS; para 50°C, apenas de ES.

Em cada disposição, realizaram-se variados ensaios, cada um com um degrau de corrente diferente, mantendo-se aproximadamente o mesmo valor de  $I_{bottom}$  para cada temperatura. Apesar disso (e de  $I_{bottom}$  ser considerada constante neste trabalho), por exemplo, a 20°C, em alguns ensaios, o laser chegou a ser alimentado por correntes em torno de 50 mA no momento anterior ao degrau abrupto.

A distinção entre um degrau e outro se dá exatamente em seu patamar de corrente  $I_{top}$ : procuraram-se realizar ensaios consecutivos com uma diferença de 1mA entre si. Dessa forma, para 20°C, por exemplo, o primeiro ensaio ocorreu com um degrau nominal abrupto de 60 a 65mA; o segundo, de 60 e 66mA; o terceiro, de 60 e 67mA; e assim por diante, até o último degrau, de 60 a 140mA, totalizando uma quantidade de 76 séries temporais para essa temperatura de operação.

Na Tabela 2.1, são apresentados: a quantidade de séries temporais coletadas para cada temperatura (quantidades iguais entre os estados); os valores de  $I_{bottom}$ , aproximadamente fixos; e o intervalo de  $I_{top}$ , com valores incrementados de unidade em unidade. Cada série temporal corresponde a um total de 510 valores de pares de entrada e saída entre 36,30 e 41,39 nanossegundos após o degrau abrupto de corrente, conforme Tabela 2.2.

Tabela 2.1 — Características de corrente para cada temperatura.

| Temperatura durante a amostragem | Quantidade de séries temporais | $I_{bottom}$ [mA] | Intervalo de $I_{top}$ [mA] |
|----------------------------------|--------------------------------|-------------------|-----------------------------|
| 20°C                             | 76                             | 60                | [65; 140]                   |
| 40°C                             | 71                             | 110               | [115; 185]                  |
| 50°C                             | 30                             | 170               | [171; 200]                  |

Fonte: próprio autor.

Tabela 2.2 — Características das séries temporais da amostra.

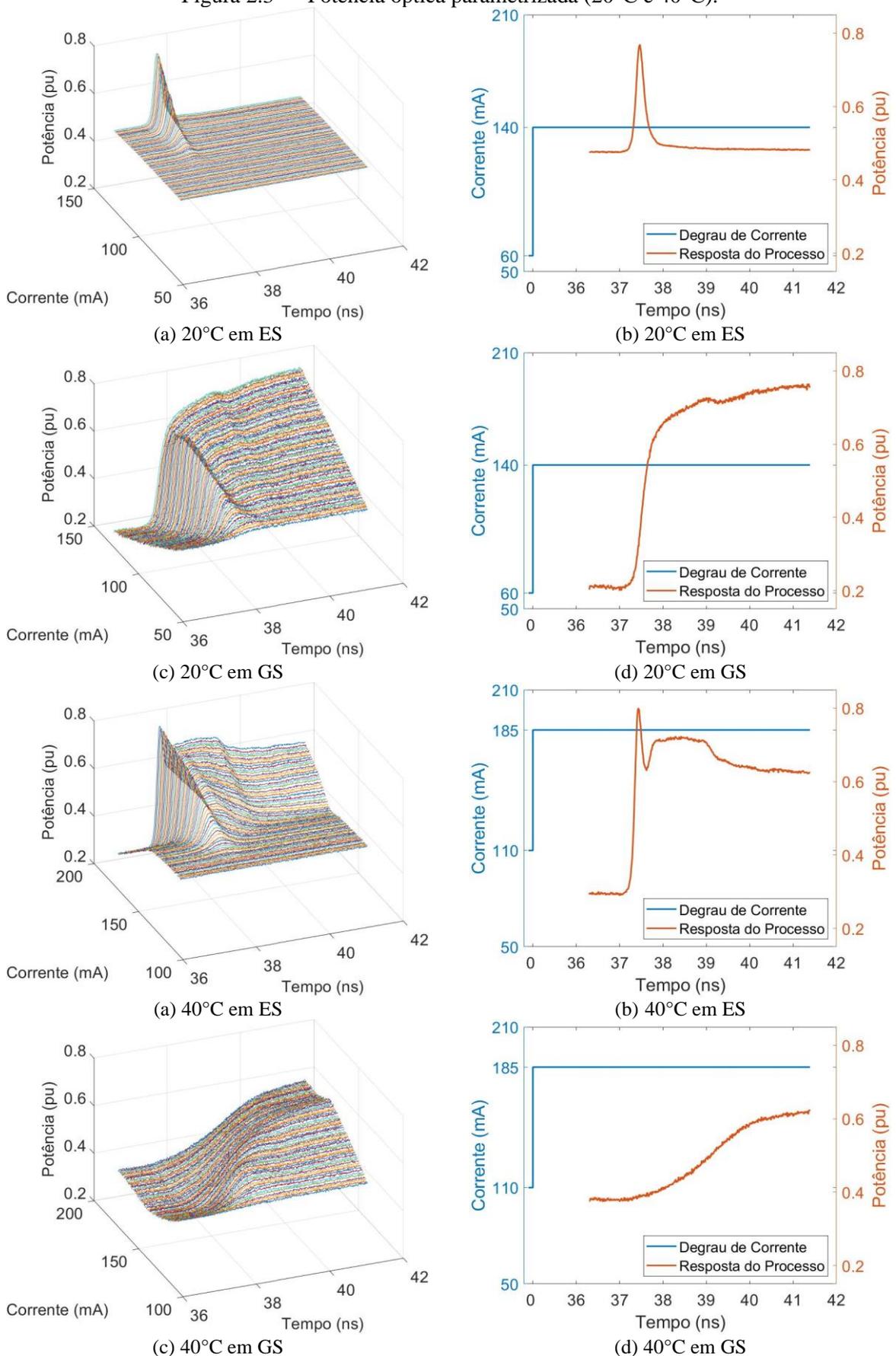
| Parâmetro                          | Valores         |
|------------------------------------|-----------------|
| Quantidade de amostras temporais   | $N_{amt} = 510$ |
| Intervalo temporal da amostra (ns) | [36,30; 41,39]  |

Fonte: próprio autor.

Nas Figura 2.3 (20°C e 40°C) e Figura 2.4 (50°C), são apresentados esquematicamente a resposta em potência óptica ao degrau de corrente (em miliampères) para cada configuração temperatura-estado do banco de dados. Nessa representação, cada linha representa uma série temporal distinta a partir de um degrau de corrente; destaca-se o exemplar de com maior  $I_{top}$  em cada configuração.

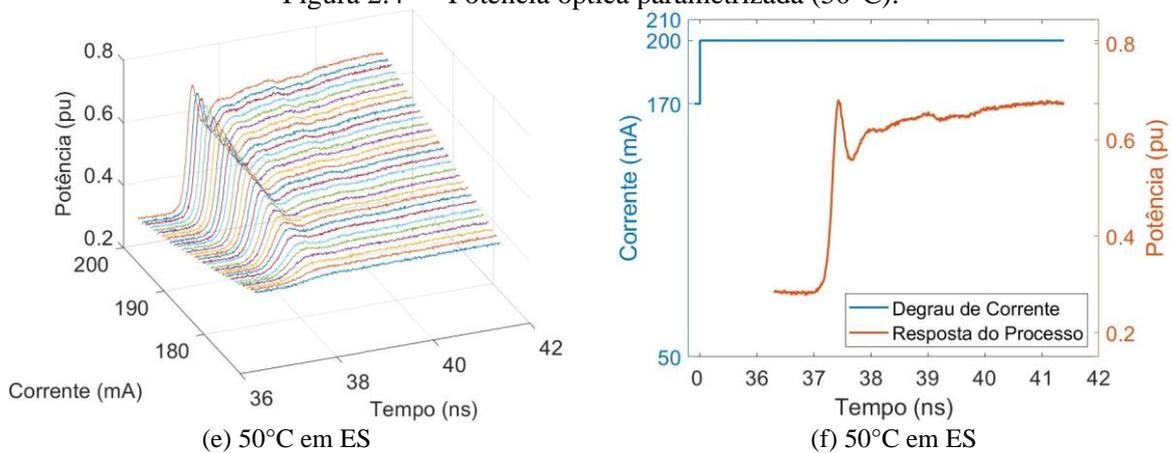
Vale ressaltar que a potência óptica está parametrizada no intervalo [0,2; 0,8], por maior adequação aos modelos propostos na Seção 6. Nos gráficos bidimensionais com o maior degrau para cada configuração, este é apresentado no instante  $t = 0$ , enquanto a primeira amostra da resposta do laser só foi colhida em  $t = 36,3\text{ns}$ , até  $t = 41,39\text{ns}$ .

Figura 2.3 — Potência óptica parametrizada (20°C e 40°C).



Representação em linhas (esquerda) vs. degrau de corrente (direita). Fonte: próprio autor.

Figura 2.4 — Potência óptica parametrizada (50°C).



Representação em linhas (esquerda) vs. degrau de corrente (direita). Fonte: próprio autor.

Cada configuração apresenta um comportamento característico graficamente distinto:

a) 20°C/ES (Figura 2.3a, b): o laser experimentado é incapaz de manter a potência em uma faixa ativa:

- $I_{top} > 120\text{mA}$ : resposta aproximadamente impulsiva, com a potência decaindo rapidamente com o tempo;
- $I_{top} < 120\text{mA}$ : resposta praticamente inalterada em relação a antes da aplicação do degrau.

b) 20°C/GS (Figura 2.3c, d): crescimento da potência com o tempo até um valor estável.

Para alguns degraus de corrente, a característica é semelhante a um sistema de segunda ordem;

c) 40°C/ES (Figura 2.3e, f): semelhante a 20°C/ES, porém, para valores maiores de  $I_{top}$ , existe uma estabilização da resposta acima de seu valor inicial:

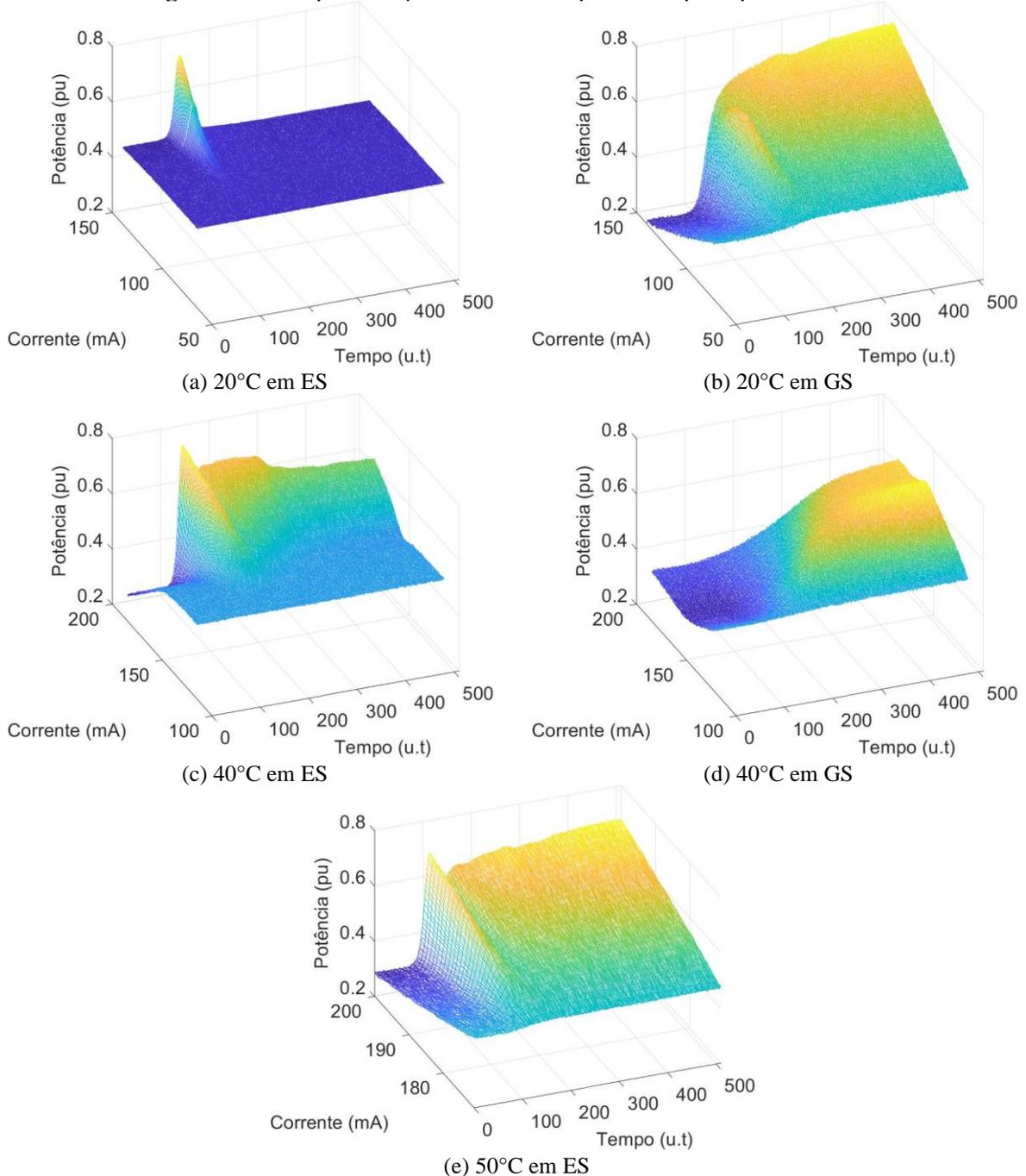
- o aumento do degrau também provoca uma tendência impulsiva inicial, entretanto, com característica de *overshoot* e posterior estabilização;
- baixos valores de  $I_{top}$  provocam pouco efeito na saída.

d) 40°C/GS (Figura 2.3g, h): comportamento semelhante a um processo de segunda ordem entre superamortecido (menor  $I_{top}$ ) e subamortecido (maior  $I_{top}$ ) (OGATA, 2010);

e) 50°C/ES (Figura 2.4a, b): assim como nas duas outras temperaturas de operação no mesmo estado, existe uma curva de declínio da potência a partir do pico inicial, no entanto, essa tendência é revertida em direção novamente ao pico.

Apesar de visualmente mais direta, a representação bidimensional do processo não tem muito atrativo além desse primeiro contato de exemplo, uma vez que é infactível apresentar os resultados obtidos para cada série temporal. Mesmo a representação tridimensional em linhas não é ideal; a esquematização adotada mais adiante para comparações entre os modelos obtidos é em malha, apresentada na Figura 2.5, sendo particularmente útil para reunir todos os pares de saída-entrada, evidenciando os pontos de distorções.

Figura 2.5 — Representação em malha da potência óptica parametrizada.



(e) 50°C em ES  
Fonte: próprio autor.

Essa representação é utilizada tanto para as ilustrações dos modelos do laser (Seção 6) quanto para o problema de aproximação de função de duas variáveis (Seção 5). Nela, os valores de tempo são utilizados em função da amostra dentro da série ( $N_{amt} = 510$ ) em vez de pelo instante em que foi colhido; u.t representa, nesse caso, abreviação para unidade de tempo, o que representaria uma unidade do tempo de amostragem ( $t_{amt} = 0,01\text{ns}$ ).

## 2.5 PROBLEMAS COMPLEMENTARES: A APROXIMAÇÃO DE FUNÇÕES

Uma aplicação notoriamente semelhante à modelagem em identificação de sistemas, tanto visualmente em saídas gráficas quanto computacionalmente, é a aproximação de função — uma identificação, inclusive, pode ser entendida simplificada como a estimação de uma função recursiva em que entradas e saídas possivelmente são realimentadas. Ambas as metodologias fazem uso de reconhecimento de padrões em um conjunto de amostras de entrada e saída correspondentes, em que as respostas podem variar livremente dentro de um intervalo parametrizado, em oposição, por exemplo, ao problema de classificação, no qual se estabelecem classes numéricas discretas ou estabelecidas em faixas de valores.

Neste estudo, desenvolveram-se duas funções matemáticas para prévia análise dos algoritmos propostos antes da modelagem da dinâmica do sistema pretendido: uma de apenas uma variável, de análise gráfica potencialmente mais direta; e outra de duas variáveis, visualmente mais próxima do sistema a modelar. A primeira (Figura 2.6a) apresenta característica senoidal, exponencial inversa e polinomial, conforme

$$y = f(x_1) = \text{sen}(2x_1) \cdot e^{\frac{0,1}{0,1+x_1}} + 0,1(x_1 - 3,5)^3 ; \quad (2.1)$$

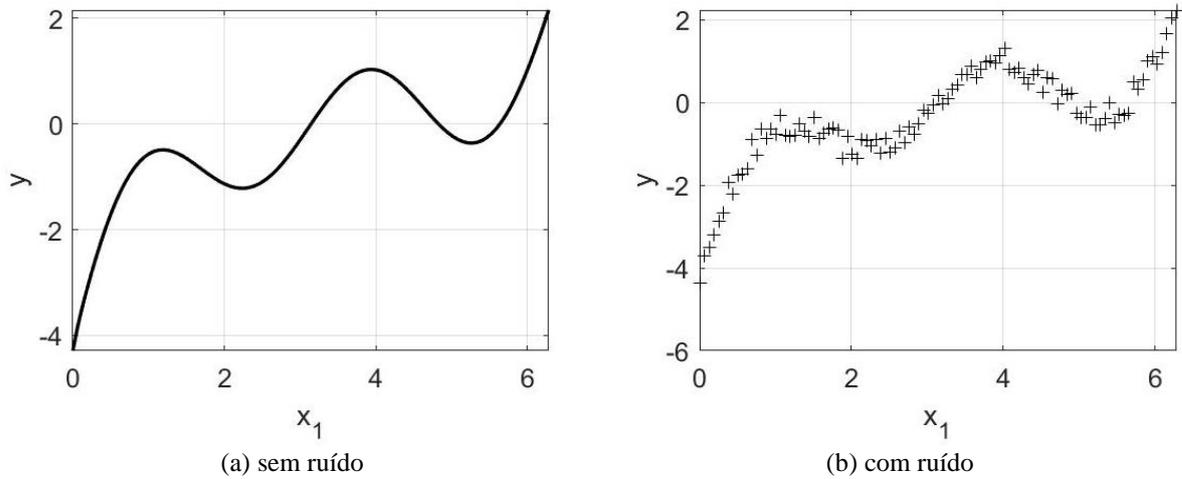
a segunda (Figura 2.7a) é preponderantemente exponencial, de acordo com

$$y = f(x_1, x_2) = x_1 x_2 e^{-x_1^2 - x_2^2} \quad (2.2)$$

— ambas apresentam dois pontos de máximo e dois de mínimo locais no intervalo definido, conforme Tabela 2.3.

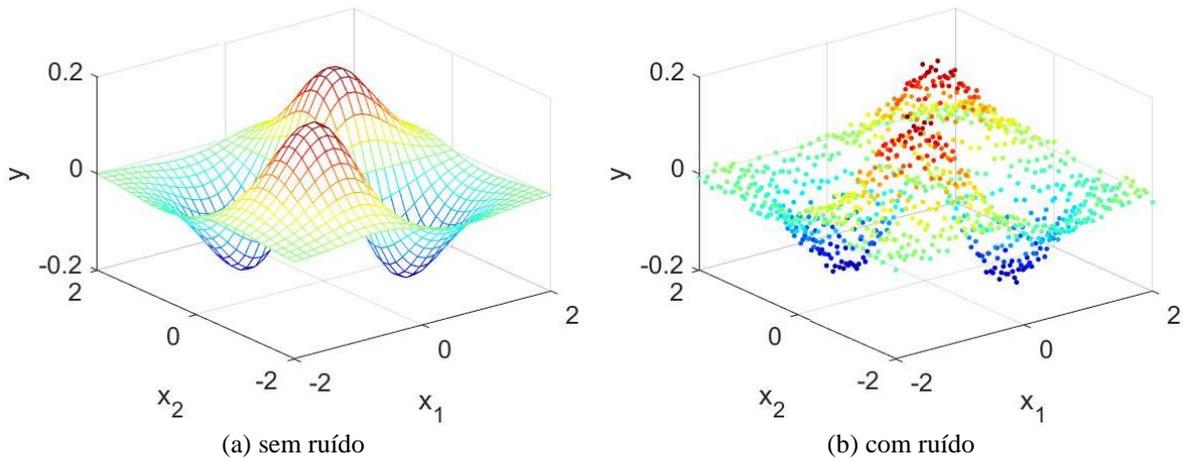
Em ambas as funções, acrescentou-se, ainda, um sinal ruidoso gaussiano branco  $e_g$ , com amplitude de até 10% da função original, a fim de simular a coleta de dados reais com distúrbios inerentes ao procedimento. A partir desse ponto, os dados obtidos das Equações (2.1) e (2.2) após inclusão de ruído são tratados como Problema 1 (Figura 2.6b) e Problema 2 (Figura 2.7b), respectivamente.

Figura 2.6 — Função de uma variável (Problema 1).



Fonte: próprio autor.

Figura 2.7 — Função de duas variáveis (Problema 2).



Fonte: próprio autor.

Tabela 2.3 — Características da amostra para as funções.

| Parâmetro                          | Símbolo    | Valores           |                        |
|------------------------------------|------------|-------------------|------------------------|
|                                    |            | Problema 1        | Problema 2             |
| Quantidade de atributos de entrada | $N_{atr}$  | $N_{atr} = 1$     | $N_{atr} = 2$          |
| Quantidade de amostras             | $N_{amt}$  | $N_{amt} = 101$   | $N_{amt} = 961$        |
| Intervalo de entrada da função     | —          | $x_1 = [0, 2\pi]$ | $x_1 = x_2 = [-2, +2]$ |
| $SNR_{dB}$ [dB]                    | $SNR_{dB}$ | 16,82             | 17,44                  |

Fonte: próprio autor.

A razão sinal-ruído (*Signal-to-Noise Ratio*, SNR) é de 16,82 dB, para o Problema 1, e de 17,44 dB, para o Problema 2 — realiza-se o cálculo da SNR a partir de

$$SNR_{dB} = 10 \log(SNR) = 20 \log\left(\frac{A_{sinal}}{A_{ruído}}\right), \quad (2.3)$$

conforme apresentado em Price e Goble (2014), em que  $A_{sinal}$  e  $A_{ruído}$  são as amplitudes (medidas em RMS<sup>2</sup>) do sinal e do ruído, respectivamente, e o índice dB indica que o resultado é apresentado em decibéis.

Na Tabela 2.3, algumas informações relevantes para levantamento do banco de dados para cada problema são descritas: quantidade de atributos de entrada do modelo, intervalo das variáveis da função utilizado para coleta e quantidade de amostras colhidas. No Problema 2,  $N_{amt} = 961$  representa 31 valores distribuídos uniformemente no intervalo  $[-2, +2]$  para cada uma das duas variáveis  $x_1$  e  $x_2$  ( $31 \cdot 31 = 961$ ).

Ambos os problemas de aproximação de função são utilizados na Seção 5 para uma primeira avaliação dos modelos neuro-otimizados desenvolvidos, por constituírem bancos de dados altamente controláveis pelo pesquisador. A maior parte da discussão constitui o alicerce para a modelagem de laser semicondutor QD, na Seção 6.

---

<sup>2</sup> RMS: *root mean square*.

### 3 MODELOS GLOBAIS SUPERVISIONADOS

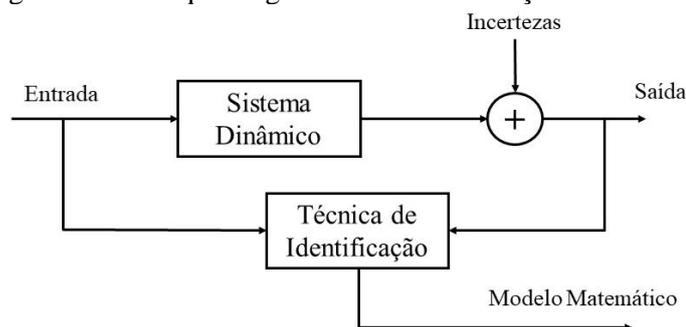
“The neural network is this kind of technology that is not an algorithm.”<sup>1</sup> — Howard Rheingold

Esta seção aborda as etapas envolvidas na identificação de um sistema, com especial destaque para duas: a representação matemática (pelas redes neurais MLP e ELM) e a validação dos modelos. Segundo Aguirre (2004), a identificação, por vezes, constitui-se de um procedimento relativamente simples, que obtém um modelo matemático, em geral, a partir apenas dos sinais de entrada e saída do sistema real que se pretende modelar.

Nesse caso, o processo em si é desconhecido — ou ignorado —, de tal forma que o modelo tem seus parâmetros e sua estrutura determinados empiricamente (caixa preta, *black box*), em contrapartida à modelagem realizada por caixa branca (*white box*). Nesta abordagem, a construção do modelo pode levar anos, uma vez que o sistema não somente deve ser familiar, como também o seu comportamento, de descrição possível e clara por meio de equações e parâmetros regidos por leis da ciência (NELLES, 2013).

Na Figura 3.1, apresenta-se um esquema genérico de identificação de sistemas dinâmicos: uma técnica pré-determinada relaciona entrada e saída, produzindo um modelo matemático — deve-se considerar incertezas na saída em virtude de fatores externos que influem sobre medições e sobre o próprio processo. Por se tratar de um problema em que não se conhece o sistema profundamente, com certo grau de imprecisão na relação entrada-saída, o modelo obtível não é único e deve ser avaliado.

Figura 3.1 — Esquema genérico da identificação de sistemas.



Fonte: próprio autor.

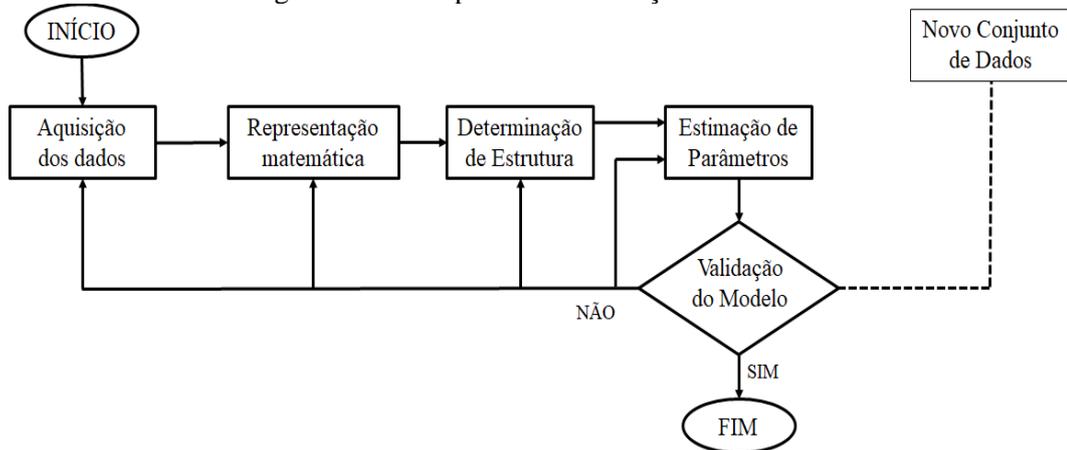
#### 3.1 ETAPAS DA IDENTIFICAÇÃO DE SISTEMAS

Segundo Aguirre (2004), a identificação de sistemas é constituída de cinco etapas distintas e consecutivas (Figura 3.2): aquisição dos dados, definição da representação

<sup>1</sup> Tradução nossa: “A rede neural é esse tipo de tecnologia que não é um algoritmo” (GARCIA, 1990).

matemática, determinação da estrutura do modelo, estimação de parâmetros e validação. Após a última, caso o modelo obtido seja insatisfatório, deve-se efetuar alguma alteração em uma ou mais etapas anteriores e, então, realizar a modelagem novamente. Adiante, debate-se concisamente cada etapa.

Figura 3.2 — Etapas da identificação de sistemas.



Fonte: próprio autor.

### 3.1.1 Aquisição de dados

O êxito na modelagem em caixa preta depende essencialmente da qualidade dos dados coletados do sistema: bons modelos só são obtidos se as séries temporais utilizadas contiverem toda a informação do processo (ou da parcela deste que se quer modelar); e isto pode ser custoso ou mesmo nem possível, mas essencial (CORRÊA; AGUIRRE, 2004).

A coleta de informações do processo deve considerar dois pontos: o sinal de excitação aplicado ao sistema e o tempo de amostragem. A excitação precisa estar dentro das faixas de interesse e limitado às próprias condições do processo; sinais comuns para este fim são o degrau, a senóide, o sinal binário pseudoaleatório (*pseudo random binary signal*, PRBS) e o ruído branco, entre outros (DANTAS, 2013; COELHO; COELHO, 2004). Por outro lado, a amostragem deve ser inferior à rápida demais, a ponto de não considerar o tempo mínimo para que algumas características sejam percebidas, e superior à lenta em excesso, a ponto de distorcer

### 3.1.2 Representação matemática

O modelo linear mais genérico é o autorregressivo com média móvel e com entradas externas (*AutoRegressive Moving Average with eXogenous inputs*, ARMAX), no entanto, processos em geral são satisfatoriamente modelados sem considerar a média móvel ou entradas exógenas. A ordem desses modelos é determinada pelo valor máximo de regressão da saída  $y(t)$ ,  $n_y$ , e da entrada  $u(t)$ ,  $n_u$ , além de  $\tau_d$ , o atraso em quantidade de instantes da

amostragem<sup>2</sup>; para um modelo autorregressivo com entradas externas (*AutoRegressive with eXogenous inputs*, ARX)<sup>3</sup>, a resposta é obtida conforme

$$y(t) = a_1 y(t-1) + \dots + a_{n_y} y(t-n_y) + b_0 u(t-\tau_d) + \dots + b_{n_u-1} u(t-\tau_d-n_u+1) + e(t), \quad (3.1)$$

em que  $a_1, \dots, a_{n_y}$  e  $b_0, \dots, b_{n_u-1}$  são coeficientes pertencentes ao conjunto dos números reais e  $e(t)$  é um ruído gaussiano branco.

Entretanto, sistemas reais, em geral, são não lineares e seus respectivos modelos também precisam admitir não linearidades. O modelo não linear autorregressivo com entradas exógenas (*Non-linear AutoRegressive with eXogenous inputs*, NARX) é descrito matematicamente por:

$$y(t) = f^l \left( y(t-1), \dots, y(t-n_y), u(t-\tau_d), \dots, u(t-\tau_d-n_u+1) \right) + e(t), \quad (3.2)$$

com a adição de uma função  $f^l$ , cujo grau de não-linearidade é igual a  $l$ .

Vale ressaltar que as Equações (3.1) e (3.2) se referem a modelos monovariáveis, ou SISO (*Single Input Single Output*), caracterizados por apenas um sinal de entrada e um de saída. Em caso de múltiplas entradas, acrescentam-se estas e seus respectivos coeficientes na Equação (3.1) ou suas relações não-lineares na Equação (3.2); já para múltiplas saídas, cada uma seria representada por uma equação particular.

Em última análise, as redes neurais adotadas neste estudo podem ser consideradas simplificadaamente modelos do tipo NARX com um encadeamento de pesos de ponderação às entradas e de funções que promovem a não-linearidade na modelagem. Mais detalhes dessa análise são discutidos na Seção 3.4.

### 3.1.3 Estrutura do modelo

Para um modelo ARX simples, a estrutura restringe-se basicamente a se determinar a ordem do modelo ( $n_y$ ,  $n_u$  e  $\tau_d$ ) (AGUIRRE, 2004). Um sistema não linear inclui, ainda, toda a configuração da função que confere não-linearidade à modelagem; no caso de uma rede MLP, por exemplo, precisa-se determinar, ainda, a quantidade de neurônios na camada escondida, a taxa de aprendizagem, a função de ativação e seus parâmetros, entre outros. Cada modelo tem uma estrutura própria.

<sup>2</sup> O atraso  $\tau_d$  (ou tempo morto) pode ainda ser entendido como o intervalo de tempo entre um estímulo na entrada e seu respectivo impacto nas respostas do sistema; normalmente medido em unidades do tempo de amostragem.

<sup>3</sup> Para um modelo autorregressivo (*AutoRegressive*, AR), sem entradas exógenas ou média móvel, removem-se os coeficientes e componentes relativos ao sinal de entrada da Equação (3.2).

### 3.1.4 Estimação de parâmetros

A estimação consiste em adequar os valores numéricos dos coeficientes da representação matemática delimitada à estrutura de modelo. Essa etapa está intimamente relacionada à representação adotada: técnicas de estimação de parâmetros, apesar de adaptáveis, a princípio, atendem a modelos específicos. A modelagem clássica é arcabouçada no método dos mínimos quadrados (MMQ) e seus derivados; mesmo algumas técnicas de aprendizagem de máquina utilizam conceitos advindos dele — como na camada de saída da ELM.

### 3.1.5 Validação de modelos

A pergunta mais importante a ser respondida após a modelagem é se esta apresenta resultados satisfatórios. Uma forma relativamente simples de validar um modelo consiste em verificar o comportamento estimado em relação ao esperado: aplicar entradas cujas respostas do sistema modelado são conhecidas e checar se as saídas correspondem. Cabe, no entanto, uma relevante ressalva: os dados utilizados na validação devem ser distintos dos utilizados na modelagem, uma vez que o modelo pode ter aprendido apenas as repostas das entradas de treinamento e não a reproduzir o comportamento do sistema real em si.

Na prática, idealmente, efetuam-se dois experimentos independentes na observação do sistema, de modo a obter dois conjuntos de dados distintos, um voltado à obtenção do modelo e outro à sua validação; se dois experimentos não podem ser realizados no sistema, então, divide-se o conjunto de dados daquele realizado em dois (não necessariamente iguais), com o mesmo objetivo. A este método denomina-se *holdout* (AGUIRRE, 2004).

Para modelos autorregressivos, a predição de saída apresentada em simulação mais comum é a de um passo a frente, em que há a estimação apenas da próxima saída: o modelo não utiliza sua resposta  $\hat{y}(t)$  atual para processar e produzir a seguinte,  $\hat{y}(t + 1)$ , mas sim a resposta  $y(t)$ , obtida pelo sistema real. Em oposição, a predição de infinitos passos à frente (ou simulação livre) utiliza todas as estimativas já realizadas para previsão do próximo passo.

#### 3.1.5.1 Mensuração do erro na saída

Durante a simulação, além de qualificar os desvios do modelo em relação às respostas obtidas do sistema, é importante quantificá-los pelo erro na saída,  $\xi(t)$ , conforme

$$\xi(t) = y(t) - \hat{y}(t), \quad (3.3)$$

em que  $\hat{y}(t)$  é o sinal obtido a partir do modelo e  $y(t)$ , o desejado, a partir do sistema real.

Como a avaliação do modelo é normalmente fornecida pela média entre as amostras, essa métrica não é confiável: o erro médio pode ser baixo mesmo para desvios grosseiros, desde que estejam bem distribuídos entre valores negativos e positivos. Para evitar esse falso positivo na avaliação, uma variação bem comum é o erro quadrático médio (*mean squared error*, MSE), conforme (o traço acima dos termos representa o valor médio do sinal):

$$\text{MSE} = \overline{\xi^2(t)} = \overline{(y(t) - \hat{y}(t))^2} . \quad (3.4)$$

### 3.1.5.2 Análise dos resíduos

O erro  $\xi(t)$  pode ainda ser visto como um fator residual. Uma vez que os dados obtidos do sistema real estão expostos a uma série de possíveis erros e incertezas de medições, dificilmente, o modelo é capaz de reproduzir com exatidão todas as saídas do sistema real, porque parte da informação contida na medição não faz parte da dinâmica do processo.

A pergunta a ser respondida, então, resume-se a: o quão residual o erro deve ser para que tudo explicável nos dados seja, de fato, explicado? Para que contenham apenas aquilo que não é próprio da dinâmica do sistema, o resíduo deve ser uma variável aleatória branca. Caso seja, porém, não há garantias que a simulação livre do modelo seja adequada, mas somente que a simulação um passo à frente é boa (AGUIRRE, 2004).

Segundo Billings e Voon (1986), o sinal de resíduo  $\xi(t)$  é uma série temporal linearmente branca se sua função de autocorrelação (FAC) for nula para qualquer atraso  $\tau$  não nulo:

$$\Phi_{\xi\xi}(\tau) = E[\xi(t) \cdot \xi(t + \tau)] = \delta(\tau) , \quad (3.5)$$

em que  $E[\cdot]$  é o operador de esperança matemática e  $\delta(\tau)$ , o impulso unitário. Em trabalhos semelhantes, porém, Billings e Voon (1983) e Billings e Tao (1991) subtraíram o valor médio de resíduo da análise, de forma a minimizar a influência do *offset* do sinal:

$$\Phi_{\xi'\xi'}(\tau) = E[(\xi(t) - \overline{\xi(t)}) \cdot (\xi(t + \tau) - \overline{\xi(t)})] = \delta(\tau) , \quad (3.6)$$

em que o sinal de apóstrofo no índice indica a subtração da média do sinal e o traço superior ao sinal indica o valor médio.

As Equações (3.5) e (3.6), no entanto, só verificam correlações lineares. A fim de procurar possíveis informações advindas de não linearidades presentes no resíduo, em ambos os trabalhos de Billings e Tao (1991) e de Billings e Zhu (1995), propuseram-se duas outras equações de validação (respectivamente, linear-quadrada e quadrada-quadrada):

$$\Phi_{\xi'\xi'^2}(\tau) = E[(\xi(t) - \overline{\xi(t)}) \cdot (\xi^2(t + \tau) - \overline{\xi^2(t)})] = 0 , \quad (3.7)$$

$$\Phi_{\xi^2 \xi^2}(\tau) = E[(\xi^2(t) - \overline{\xi^2(t)}) \cdot (\xi^2(t + \tau) - \overline{\xi^2(t)})] = \delta(\tau). \quad (3.8)$$

Por outro lado, o resíduo da saída do modelo, além de descorrelacionado de si, deve não conter informação correlacionada à entrada. Billings e Voon (1983) propuseram a verificação entre entrada e resíduo em segunda ordem e em terceira ordem, respectivamente:

$$\Phi_{\xi u}(\tau) = E[\xi(t) \cdot u(t + \tau)] = 0, \quad (3.9)$$

$$\Phi_{\xi(\xi u)}(\tau) = E[\xi(t) \cdot \xi(t + \tau - 1) \cdot u(t + \tau - 1)] = 0. \quad (3.10)$$

Em busca de correlações não lineares entre entrada e saída, Billings e Voon (1986) estenderam o cálculo aos termos quadráticos do resíduo e da entrada:

$$\Phi_{u^2 \xi}(\tau) = E[(u^2(t) - \overline{u^2(t)}) \cdot \xi(t + \tau)] = 0, \quad (3.11)$$

$$\Phi_{u^2 \xi^2}(\tau) = E[(u^2(t) - \overline{u^2(t)}) \cdot \xi^2(t + \tau)] = 0. \quad (3.12)$$

Outras verificações podem ser utilizadas, inclusive utilizando o sinal de saída, como (BILLINGS; ZHU, 1995):

$$\Phi_{(y\xi)' \xi^2}(\tau) = E[(y(t)\xi(t) - \overline{y(t)\xi(t)}) \cdot (\xi^2(t + \tau) - \overline{\xi^2(t)})] = 0, \quad (3.13)$$

Apesar de propostas nas décadas de 1980 e 1990, essas equações de correlação ou variações delas repercutiram posteriormente e permanecem em uso ainda hoje para validação de modelos de sistemas diversos. A saber, alguns exemplos das pesquisas na literatura que utilizaram dessas proposições nos últimos cinco anos: Saad, Osman e Mashor (2014); He et al (2014); Barreto e Souza (2016); Gritti (2015); Pour (2015); Zhu et al (2015); Binkowski (2016); da Cruz et al (2016); Guan, Zhao e Zhu (2016); He et al (2016); Solares, Wei e Billings (2017); Ayala, Rakotondrahe e Coelho (2018); Zakaria et al (2018); Inaoka et al (2019).

Por fim, de forma clara ou subentendida, os autores dessas estratégias de destacam que, na prática, o termo de correlação obtido ainda é normalizado. Neste trabalho, a fim de evitar possíveis confusões, o termo  $\Phi$  das Equações (3.5) a (3.13) é substituído por  $r$  nas posteriores seções, evidenciando se tratar do coeficiente de correlação, conforme:

$$r_{XY} = \frac{\Phi_{X'Y'}}{\sigma_X \sigma_Y}. \quad (3.14)$$

Há de se destacar que, além de confinado no intervalo de  $-1$  e  $1$ ,  $r_{XY}$  é considerado aproximadamente nulo (ou não significativo) caso esteja dentro de um intervalo de confiança em torno de zero. Billings e Voon (1983), bem como a literatura em geral, sugerem um intervalo que garanta 95% de confiança; para uma quantidade  $N$  grande de amostras, os limites desse intervalo são definidos em  $\pm 1,96/\sqrt{N}$ .

### 3.2 PERCEPTRON DE MÚLTIPLAS CAMADAS

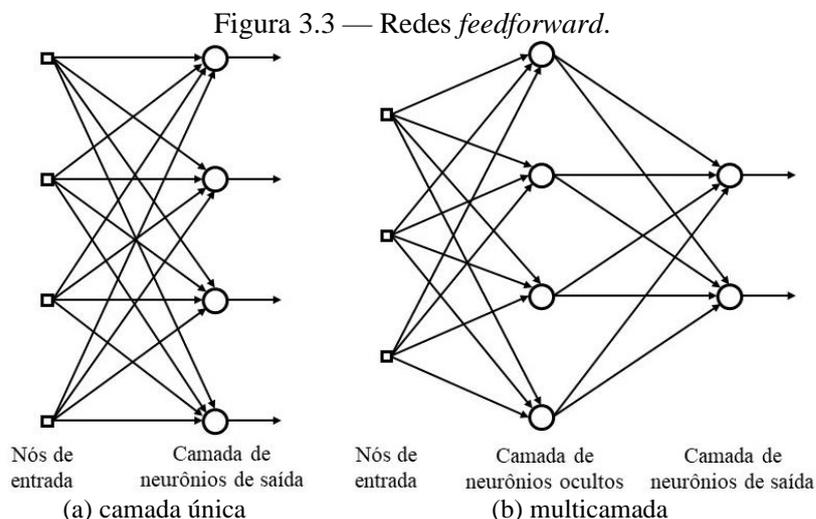
O *perceptron* de múltiplas camadas (*MultiLayer Perceptron*, MLP) é uma rede supervisionada *feedforward* como apresentado na Figura 3.3, em que todos os seus neurônios seguem o modelo da Figura 3.4. O treinamento clássico do MLP ocorre pela retropropagação do erro (ou *error backpropagation*), descrito a seguir.

#### 3.2.1 A aprendizagem em redes neurais

A capacidade de aprender e generalizar o conhecimento aprendido a partir de um conjunto de amostras reduzido é, sem dúvida, um dos maiores atrativos das por redes neurais (BRAGA, 2000). De forma objetiva, RNAs são treinadas com base em um de dois paradigmas, o supervisionado ou o não supervisionado. A supervisão, nesse caso, depende de um ente que treine a rede a partir do erro da saída estimada e, mesmo se apresentando novos dados após a fase de treinamento (como na etapa de teste), a RNA não está apta a aprender nenhuma informação a mais, mas tão somente fornecer suas respostas estimadas (HAYKIN, 2001).

Em RNAs *feedforward* (acíclica, ou alimentada adiante), como no MLP ou na ELM, a informação percorre a rede sempre em direção à sua saída, de um neurônio de uma camada para a outro da camada posterior, conforme Figura 3.3, em que cada círculo representa um neurônio e cada quadrado, uma entrada; as setas representam a informação circulando ao longo da rede (BRAGA, 2000; WELCH; RUFFING; VENAYAGAMOORTHY, 2009).

A quantidade de camadas em uma RNA interfere na complexidade da solução que ela é capaz de fornecer. Redes monocamadas, por exemplo, se limitam a solucionar problemas linearmente separáveis; em contrapartida, a adição de camadas intermediárias (ocultas) incrementa a capacidade de aprender estatísticas de ordem elevada dos dados.



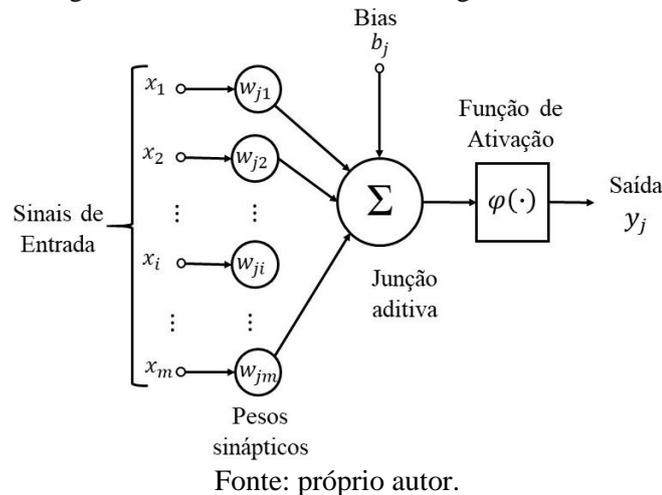
Fonte: próprio autor.

A quantidade de entradas,  $N_{in}$ , e de neurônios de saída,  $N_{out}$ , são definidos em geral pela própria aplicação da rede. Por outro lado, a quantidade de neurônios nas camadas ocultas,  $N_{hid}$ , é outro fator importante na arquitetura da rede e de sua complexidade — de praxe, determinada empiricamente, por tentativas e erros<sup>4</sup>, ou por métodos práticos específicos.

### 3.2.2 Descrição matemática do neurônio

Um neurônio  $j$  qualquer com  $m$  nós de entrada é apresentado de forma genérica na Figura 3.4 (esquema típico de um *perceptron*), em que cada nó de entrada  $i$  é excitado por um valor  $x_i$  e apresenta peso sináptico próprio  $w_{ji}$ . À semelhança de seu par biológico, porém com avaliação matemática imediata, o modelo artificial é excitado pelas diferentes entradas, posteriormente ponderadas e somadas e, enfim, submetidas a uma função de ativação  $\varphi(\cdot)$  antes de retransmitidas (BARRETO, 2002) — denomina-se  $v_j$  de campo local induzido.

Figura 3.4 — Modelo de neurônio genérico atual.



Além das excitações do meio, existe uma entrada fixa  $b_j$  própria do neurônio, o *bias*. Matematicamente, o modelo da Figura 3.4 é descrito pelo somatório

$$v_j = b_j + \sum_{i=1}^m w_{ji}x_i = \sum_{i=0}^m w_{ji}x_i \quad (3.15)$$

e pela posterior ativação

$$y_j = \varphi(v_j), \quad (3.16)$$

em que se considera o *bias* como um peso de entrada unitária ( $w_{k0} = b_k$  e  $x_0 = +1$  ou  $-1$ ).

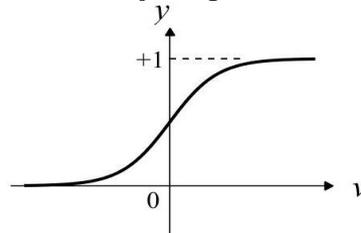
<sup>4</sup> Esse processo de tentativa e erro pode ser incluído no próprio algoritmo. Em alguns dos modelos das seções 5 e 6, o AG e o PSO são responsáveis não só pelo ajuste dos pesos da rede, mas também pela otimização de  $N_{hid}$ .

A função de ativação é quem produz em última instância a resposta propriamente dita aos estímulos, dentro de um intervalo restrito, normalmente entre 0 e 1 ou entre  $-1$  e 1. No MLP e na ELM, a função de ativação dos neurônios é tipicamente uma sigmoide, como a logística ou a tangente hiperbólica, para conferir algum grau de não linearidade na saída, além do fato de serem deriváveis, o que é necessário ao método de treinamento do *backpropagation* (Seção 3.2.3). A função logística é a adotada neste trabalho e é descrita por

$$y(t) = \varphi(v(t)) = \frac{1}{1 + e^{-av(t)}}, \quad (3.17)$$

em que  $a$  é a inclinação da curva ( $a > 0$ ) e  $b$ , um fator multiplicativo da amplitude. Para  $b = 1$ , a imagem (Figura 3.5) dessa função é delimitada no intervalo  $[0, 1]$ .

Figura 3.5 — Função logística ( $a = 1, b = 1$ ).



Fonte: Carvalho (2017).

O problema de reconhecimento de padrões em redes supervisionadas se resume, então, a determinar os valores numéricos dos pesos em cada um dos neurônios de cada uma das camadas da rede de modo que a saída final esteja de acordo com o esperado. Essa solução é obtida de modo particular para cada RNA, podendo haver mudanças no modelo genérico do neurônio da Figura 3.4 ou de outros aspectos da rede como um todo.

### 3.2.3 Algoritmo de retropropagação do erro

A aprendizagem por retropropagação do erro é caracteristicamente supervisionada: o erro da saída estimada pelo neurônio é comparado à desejada e a diferença entre elas (erro), utilizada no ajuste dos pesos de entrada do neurônio. Normalmente, os neurônios são agrupados e o processo de retropropagação se dá de camada a camada e não de neurônio a neurônio.

Existe, entretanto, um empecilho a esse raciocínio: na modelagem (caixa preta) de qualquer sistema, de praxe, conhecem-se previamente apenas os estímulos e suas respectivas respostas e, portanto, não há como saber com exatidão os valores adequados para as respostas das camadas intermediárias. O algoritmo contorna isso por meio de gradientes locais: na camada de saída, é calculado a partir do próprio erro; nas intermediárias, a partir de uma estimativa, obtida pelo erro da camada vizinha subsequente.

A aprendizagem por retropropagação do erro, então, é dividida em duas etapas: a de propagação direta (*forward propagation*), em que todas as camadas da rede são ordenadamente estimuladas e estimam suas saídas, enquanto os pesos sinápticos permanecem inalterados; e a de propagação reversa (*backward propagation*), em que as respostas estimadas são usadas no ajuste dos pesos. Para um MLP de uma camada oculta, a propagação direta ocorre na sequência:

a) cada neurônio oculto  $j$ :

- recebe estímulo  $x_i(t)$  de cada nó de entrada  $i$ , ponderando-o por um peso  $w_{ji}^{hid}$ ;
- produz uma resposta  $\hat{y}_j^{hid}(t)$ , conforme Equações (3.15) e (3.16).

b) cada neurônio da camada de saída  $k$ :

- recebe estímulo  $\hat{y}_j^{hid}(t)$  de cada nó oculto  $j$ , ponderando-o por um peso  $w_{kj}^{out}$ ;
- produz uma resposta  $\hat{y}_k(t)$ , conforme Equações (3.15) e (3.16).

A etapa de propagação reversa utiliza as respostas estimadas da rede para produzir um erro  $\xi_k(t)$  para cada neurônio  $k$  na saída, conforme Equação (3.3), reproduzida a seguir:

$$\xi_k(t) = y_k(t) - \hat{y}_k(t), \quad (3.18)$$

em que  $y_k$  é a resposta desejada, previamente conhecida. O gradiente local  $\delta_k(t)$  do neurônio é calculado a partir de  $\xi_k(t)$  e da derivada da função de ativação  $\varphi'(v_k(t))$  do próprio neurônio:

$$\delta_k(t) = \xi_k(t)\varphi'(v_k(t)). \quad (3.19)$$

Por fim, o ajuste  $\Delta w_{kj}$ , que deve ser acrescido ao peso  $w_{kj}$ , Equação (3.21), é obtido de  $\hat{y}_j(t)$ , a resposta do neurônio  $j$  da camada anterior, e de  $\eta$ , uma taxa de aprendizagem arbitrária:

$$\Delta w_{kj}(t) = \eta\delta_k(t)\hat{y}_j(t). \quad (3.20)$$

$$w_{kj}(t+1) = w_{kj}(t) + \Delta w_{kj}(t). \quad (3.21)$$

As Equações (3.19), (3.20) e (3.21) são também válidas para o ajuste dos pesos da camada oculta da rede, realizando-se as devidas alterações de índices, respectivamente:

$$\delta_j(t) = \xi_j(t)\varphi'(v_j(t)), \quad (3.22)$$

$$\Delta w_{ji}(t) = \eta\delta_j(t)x_i(t), \quad (3.23)$$

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t). \quad (3.24)$$

Entretanto, como não existe conhecimento prévio da resposta esperada para os neurônios da camada oculta, o erro nessa camada é estimado pelo gradiente local de saída<sup>5</sup>:

<sup>5</sup> Para detalhes da obtenção das Equações (3.19) a (3.24): Carvalho (2017), Haykin (2001), ou os artigos originais do algoritmo de Rumelhart, Hinton e Williams (1985, 1986).

$$\xi_j(t) = \sum \delta_k(n)w_{kj} . \quad (3.25)$$

Denomina-se época cada ciclo de aprendizagem do algoritmo de treinamento, em que a informação contida em todas as amostras destinadas a esta etapa é propagada diretamente para a saída e, então, retropropagada em erro para atualizar os pesos. Não necessariamente ocorre apenas um ajuste dos pesos dentro de uma época, como apresentado na alínea g da Seção 3.2.4.

### 3.2.4 Aspectos ajustáveis do MLP

No MLP, como em qualquer RNA, a configuração equivocada de algum parâmetro de treino gera uma rede incapaz de reconhecer os padrões do sistema identificável de interesse e resulta em uma saída estimada não compatível. Essa discrepância se dá de duas formas:

- a) subajuste (ou *underfitting*): treinamento insuficiente para que os pesos consigam representar satisfatoriamente o padrão a ser reconhecido;
- b) sobreajuste (ou *overfitting*): a rede aprende, além da informação útil, uma parcela de ruídos contidos nos dados. Decorre, normalmente de treinamento excessivo, que obriga o modelo a ser mais complexo do que o necessário.

Uma série de fatores na configuração do MLP e de seu treinamento podem resultar em sobreajuste ou subajuste, ou pelo menos influenciar na velocidade em que o ajuste adequado é obtido. A seguir, alguns são elencados:

- a) **taxa de aprendizagem,  $\eta$** , que dita a velocidade de atualização dos pesos sinápticos:
  - quanto maior, mais rápido o treino (maior passo de atualização dos pesos), porém, menor acurácia e maior chance de instabilidade no ajuste dos pesos;
  - estratégia sugerida: decaimento exponencial de  $\eta$  em função da época atual do algoritmo.
- b) **inicialização dos pesos**: em geral, uma inicialização aleatória (uniforme ou gaussiana) no espaço de busca é suficiente;
- c) **função de ativação,  $\varphi(\cdot)$** : por ser antissimétrica, a tangente hiperbólica, normalmente, apresenta desempenho superior à logística;
- d) **normalização de entradas e de saídas**: leva o conjunto de entradas e saídas cada qual a uma mesma faixa de amplitude, o que, em geral, reduz o espaço de busca e a discrepância entre os valores dos pesos;

- e) **critério de parada**, para finalizar o algoritmo; em geral, mais de um é usado. Possui forte relação ao problema solucionado e dificilmente é conclusivo. Comuns:
- quantidade máxima de épocas;
  - erro na saída reduzido; entre outros.
- f) **quantidade máxima de épocas**,  $N_e$ , típico critério de parada por garantir o encerramento eventual do algoritmo. Não há números consensuais, normalmente, determinada empiricamente:
- valores pequenos provocam em subajuste;
  - valores altos propiciam gasto computacional desnecessário e sobreajuste.
- g) a quantidade de amostras utilizada a cada atualização dos pesos:
- **treinamento por lote**: as propagações direta e reversa acontecem uma única vez a cada época; a atualização se dá em função do erro médio da rede para todas amostras;
  - **treinamento local**: ocorre atualização a cada amostra; os pesos sinápticos são ajustados  $N$  vezes por época, em que  $N$  é o número total de amostras de treino;
  - a abordagem **por lote** garante a convergência do treinamento, entretanto, é mais suscetível a convergir para um mínimo local; mais estável, é recomendado em casos de alta taxa de aprendizagem;
  - a abordagem **local**, pela natureza estocástica, é passível de instabilidade, porém, mais rápida e menos onerosa em termos de memória; ideal para taxas de aprendizagem reduzidas ou banco de dados extensos ou redundantes.
- h) **número de neurônios ocultos**,  $N_{hid}$ : quanto maior, mais extração de informações da entrada. Sobre o ajuste:
- em valores razoavelmente elevados, acelera drasticamente o sobreajuste<sup>6</sup>;
  - $N_{hid}$  insuficiente implica em subajuste.

Após as considerações anteriores, cabe elencar a sequência de passos do algoritmo MLP (Quadro 3.1), restringindo o pseudocódigo a camada oculta simples, camada de saída sem *bias* e treinamento local. Os termos  $U(0,1)$  e  $N(0, \sigma^2)$  representam inicializações aleatórias uniforme e normal, respectivamente.

---

<sup>6</sup> Redes neurais *feedforward* com  $N_{hid}$  nós escondidos podem aprender com exatidão  $N_{hid}$  observações distintas (HUANG et al, 2015; TAMURA; TATEISHI, 1997). Dessa forma, para quantidades elevadas de  $N_{hid}$ , o MLP estipula, em seu treinamento, pesos sinápticos que relacionam individualmente os pares de entrada e saída, porém não representam o padrão a ser reconhecido.

Quadro 3.1 — Algoritmo MLP (duas camadas, treinamento local).

| <b>ALGORITOMO MLP</b>   |  |
|---|--|
| <b>Entrada</b>  |  |
| $x(t)$ : entrada da rede  | $y(t)$ : saída observada                                 |
| <b>Saídas</b>   |  |
| $\hat{y}(t)$ : saída estimada   | $\xi(t)$ : erro  |
| <b>Configuração da rede</b>   |  |
| $N_{hid}$ : número de neurônios ocultos   | $\eta_0, \eta_f$ : taxas de aprendizagem inicial e final |
| $N_e$ : número de épocas de treinamento   |  |
| <b>Algoritmo</b>  |  |
| I. Inicialização aleatória dos pesos ( $t = 0$ )  |  |
| $w_{ij}^{hid} \sim U(0,1)$ ou $w_{ij}^{hid} \sim N(0, \sigma^2), i = 1,2, \dots, N_{in} + 1, j = 1,2, \dots, N_{hid}$ |  |
| $w_{kj}^{out} \sim U(0,1)$ ou $w_{kj}^{out} \sim N(0, \sigma^2), j = 1,2, \dots, N_{hid}, j = 1,2, \dots, N_{out}$    |  |
| II. Etapa de treino. Para cada época ( $n = 1,2, \dots, N_e$ )  |  |
| i. Atualização de taxa de aprendizagem: $\eta = \eta_0 \left(\frac{\eta_f}{\eta_0}\right)^{\frac{t}{N_e}}$            |  |
| ii. Para cada amostra ( $t = 1,2, \dots, N_{tr}$ )  |  |
| — cálculo da ativação dos neurônios ocultos (propagação)  |  |
| $v_j^{hid}(t) = \sum w_{ij}^{hid} x_i(t)$   |  |
| $\hat{y}_j^{hid}(t) = \varphi(v_j^{hid}(t))$  |  |
| — cálculo da ativação dos neurônios da camada de saída (propagação)   |  |
| $v_k^{out}(t) = \sum w_{kj}^{out} \hat{y}_j^{hid}(t)$   |  |
| $\hat{y}_k(t) = \varphi(v_k^{out}(t))$  |  |
| — cálculo do erro na saída:   |  |
| $\xi_k(t) = y_k(t) - \hat{y}_k(t)$  |  |
| — atualização dos neurônios da camada de saída (retropropagação)  |  |
| $\delta_k^{out}(t) = \xi_k(t) \varphi'(v_k^{out}(t))$   |  |
| $\Delta w_{kj}^{out}(t) = \eta \delta_k^{out}(t) \hat{y}_j^{hid}(t)$  |  |
| $w_{kj}^{out}(t+1) = w_{kj}^{out}(t) + \Delta w_{kj}^{out}(t)$  |  |
| — atualização dos neurônios da camada de saída (retropropagação)  |  |
| $\delta_j^{hid}(t) = \varphi'(v_j^{hid}(t)) \sum \delta_k^{out}(t) w_{kj}^{out}$                                      |  |
| $\Delta w_{ij}^{hid}(t) = \eta \delta_j^{hid}(t) x_i(t)$  |  |
| $w_{ij}^{hid}(t+1) = w_{ij}^{hid}(t) + \Delta w_{ij}^{hid}(t)$  |  |
| iii. Verificação de critérios de parada   |  |
| III. Etapa de teste ( $t = 1,2, \dots, N_{tt}$ )  |  |
| i. cálculo da ativação dos neurônios ocultos (propagação)   |  |
| $v_j^{hid}(t) = \sum w_{ij}^{hid} x_i(t)$   |  |
| $\hat{y}_j^{hid}(t) = \varphi(v_j^{hid}(t))$  |  |
| ii. cálculo da ativação dos neurônios da camada de saída (propagação)   |  |
| $v_k^{out}(t) = \sum w_{kj}^{out} \hat{y}_j^{hid}(t)$   |  |
| $\hat{y}_k(t) = \varphi(v_k^{out}(t))$  |  |
| iii. Cálculo de erro: $\xi_k(t) = y_k(t) - \hat{y}_k(t)$  |  |

Fonte: próprio autor.

### 3.3 MÁQUINA DE APRENDIZADO EXTREMO

Proposta por Huang, Zhu e Siew (2004), a máquina de aprendizado extremo (*Extreme Learning Machine*, ELM) é uma rede neural alimentada adiante de camada oculta simples (*single-hidden layer feedforward neural networks*, SLFN), assim como o MLP analisado na Seção 3.2.3. Aliás, a ELM e o MLP apresentam arquiteturas bastantes semelhantes, apesar das discrepâncias entre os algoritmos de aprendizagem (SOUZA, 2012).

A ELM pode ser esquematizada de forma idêntica à SLFN da Figura 3.3b. Estruturalmente, a diferença para um MLP de apenas uma camada oculta se encontra na saída da rede: os neurônios desta camada, na ELM, não apresentam função de ativação (em oposição ao modelo da Figura 3.4), de tal maneira que a resposta obtida em cada nó é linear e corresponde ao próprio campo local induzido ( $\hat{y}_k = v_k^{out}$ ).

Dessa forma, a não linearidade da ELM é conferida exclusivamente pela camada oculta, cujos neurônios seguem completamente o esquema da Figura 3.4 e o cálculo de ativação da Equação (3.16). A aprendizagem não iterativa da rede (Seção 3.3.2), inclusive, só é possível exatamente em decorrência da linearidade da camada de saída; essa característica disruptiva evita uma etapa de treinamento demorado para ajuste dos pesos, mantendo boa generalização, porém, com execução “extremamente rápida” (HUANG; ZHU; SIEW, 2006).

#### 3.3.1 Representação matricial na propagação de informações

O processo de propagação direta de informações na ELM ocorre de forma idêntica à do MLP, à exceção da ativação na de saída. Cabe ressaltar que a *bias*, não obrigatório e, por vezes, dispensado nos neurônios ocultos do MLP, em geral, é usado em ambas as camadas da ELM.

A representação matricial do equacionamento facilita o desenvolvimento do algoritmo da ELM — e da implementação do próprio MLP, se realizado em ambientes de programação voltados para matrizes, como o MATLAB®. Nesse sentido, baseado nas Equações (3.15) e (3.16), o vetor contendo o campo local induzido e contendo a saída estimada de todos os neurônios da camada oculta são obtidos, respectivamente, por

$$\mathbf{v}_{hid}(t) = \mathbf{W}_{hid}\mathbf{x}(t), \quad (3.26)$$

$$\hat{\mathbf{y}}_{hid}(t) = \varphi(\mathbf{v}_{hid}(t)), \quad (3.27)$$

em que  $\mathbf{x}(t)$  representa o vetor coluna de entradas

$$\mathbf{x}(t) = [-1 \quad x_1(t) \quad x_2(t) \quad \cdots \quad x_{N_{in}}(t)]^T, \quad (3.28)$$

e  $\mathbf{W}_{hid}$ , a matriz de pesos ocultos

$$\mathbf{W}_{hid} = \begin{bmatrix} w_{1,1}^{hid} & w_{1,2}^{hid} & \dots & w_{1,N_{in}+1}^{hid} \\ w_{2,1}^{hid} & w_{2,2}^{hid} & \dots & w_{2,N_{in}+1}^{hid} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N_{hid},1}^{hid} & w_{N_{hid},2}^{hid} & \dots & w_{N_{hid},N_{in}+1}^{hid} \end{bmatrix}_{N_{hid} \times (N_{in}+1)}. \quad (3.29)$$

Antes da informação ser propagada para a de saída,  $\hat{\mathbf{y}}_{hid}(t)$  é acrescido de novo *bias*:

$$\mathbf{y}_{hid}(t) = \begin{bmatrix} -1 \\ \hat{\mathbf{y}}_{hid}(t) \end{bmatrix} = \begin{bmatrix} -1 \\ y_1^{hid}(t) \\ y_2^{hid}(t) \\ \vdots \\ y_{N_{hid}}^{hid}(t) \end{bmatrix}. \quad (3.30)$$

O novo vetor obtido,  $\mathbf{y}_{hid}(t)$ , é utilizado na estimação da resposta da rede, sem ativação:

$$\hat{\mathbf{y}}(t) = \mathbf{v}_{out}(t) = \mathbf{W}_{out} \mathbf{y}_{hid}(t), \quad (3.31)$$

em que  $\mathbf{W}_{out}$  é a matriz de pesos de saída

$$\mathbf{W}_{out} = \begin{bmatrix} w_{1,1}^{out} & w_{1,2}^{out} & \dots & w_{1,N_{hid}}^{out} \\ w_{2,1}^{out} & w_{2,2}^{out} & \dots & w_{2,N_{hid}}^{out} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N_{out},1}^{out} & w_{N_{out},2}^{out} & \dots & w_{N_{out},N_{hid}}^{out} \end{bmatrix}_{N_{out} \times N_{hid}}. \quad (3.32)$$

### 3.3.2 Algoritmo de aprendizagem da ELM

Da mesma forma que no MLP, a aprendizagem se resume a encontrar pesos nas camadas oculta e de saída tais que a estimativa produzida pela rede seja condizente com a resposta observada no sistema real. Sem o uso da realimentação do erro, o algoritmo consiste em manter os pesos ocultos inicializados aleatoriamente e mapear os pesos de saída por meio de uma técnica clássica: o método dos mínimos quadrados (MMQ).

O MMQ é linear e se associa, tipicamente, a pseudoinversa de Moore-Penrose<sup>7</sup>, por conta da restrição da definição de inversa apenas a matrizes quadradas. A matriz  $\mathbf{W}_{out}$  é determinada segundo

$$\mathbf{W}_{out} = (\mathbf{Y}_{hid} \mathbf{Y}_{hid}^T)^{-1} \mathbf{Y}_{hid} \mathbf{Y}_{out}, \quad (3.33)$$

<sup>7</sup> A pseudoinversa foi descrita primeiramente por Moore (1920) e, posteriormente, redescoberta por Penrose (1955). Mais detalhes em Barata e Hussen (2012).

em que  $\mathbf{Y}_{hid}$  é a matriz acumulada de respostas obtidas na camada oculta para todas as amostras

$$\begin{aligned} \mathbf{Y}_{hid} &= [\mathbf{y}_{hid}(1) \quad \mathbf{y}_{hid}(2) \quad \cdots \quad \mathbf{y}_{hid}(N_{tr})] \\ &= \begin{bmatrix} -1 & -1 & \cdots & -1 \\ y_1^{hid}(1) & y_1^{hid}(2) & \cdots & y_1^{hid}(N_{tr}) \\ y_2^{hid}(1) & y_2^{hid}(2) & \cdots & y_2^{hid}(N_{tr}) \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_{hid}}^{hid}(1) & y_{N_{hid}}^{hid}(2) & \cdots & y_{N_{hid}}^{hid}(N_{tr}) \end{bmatrix}, \end{aligned} \quad (3.34)$$

e  $\mathbf{Y}_{out}$ , a de saídas observadas

$$\mathbf{Y}_{out} = [\mathbf{y}(1) \quad \mathbf{y}(2) \quad \cdots \quad \mathbf{y}(N_{tr})] = \begin{bmatrix} y_1(1) & y_1(2) & \cdots & y_1(N_{tr}) \\ y_2(1) & y_2(2) & \cdots & y_2(N_{tr}) \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_{out}}(1) & y_{N_{out}}(2) & \cdots & y_{N_{out}}(N_{tr}) \end{bmatrix}, \quad (3.35)$$

cujo termo  $\mathbf{y}$  representa a observação de todos os neurônios para cada amostra

$$\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_{N_{out}}(t) \end{bmatrix}. \quad (3.36)$$

O algoritmo completo da ELM é esquematizado no Quadro 3.2.

### 3.3.3 Considerações sobre a ELM

Um comparativo imediato ao MLP remete a ausência da aprendizagem não linear adaptativa nas camadas da ELM, uma vez que, na oculta, os neurônios são mantidos com os pesos inicializados aleatoriamente e, na de saída, são mapeados linearmente pelo MMQ. Outros pontos de destaque da ELM:

- a) há poucos parâmetros ajustáveis e consegue bons resultados de generalização (HUANG et al, 2012);
- b) rapidez notoriamente maior que algoritmos iterativos baseados em gradiente, porém, requer mais neurônios dos que as RNAs tradicionais, treinadas por *backpropagation* (HUANG et al, 2015);
- c) quanto maior  $N_{hid}$ , melhor a aprendizagem; não há problemas de *overfitting* (YANG; WANG; YUAN, 2012). Existe, entretanto, um ponto em que a melhoria se torna muito pequena em relação ao incremento de processamento computacional;
- d) os pesos ocultos limitam a aprendizagem da rede: os valores gerados aleatoriamente provavelmente não constituem um conjunto ótimo;

- e) o “erro de treinamento” é naturalmente minimizado pela pseudoinversa (HUANG; WANG; LAN, 2011) e não converge a mínimos locais devido à sua natureza estocástica (VAPNIK; BOTTOU, 2004).

Quadro 3.2 — Algoritmo ELM.

| <b>ALGORITOMO ELM</b>   |                          |
|---|--------------------------|
| <b>Entrada</b>  |                          |
| $\mathbf{x}(t)$ : vetor de entradas da rede   | $y(t)$ : saída observada |
| <b>Saídas</b>   |                          |
| $\hat{y}(t)$ : saída estimada   | $\xi(t)$ : erro          |
| <b>Configuração da rede</b>   |                          |
| $N_{hid}$ : número de neurônios ocultos   |                          |
| <b>Algoritmo</b>  |                          |
| I. Inicialização aleatória dos pesos ocultos ( $t = 0$ )  |                          |
| $w_{ij}^{hid} \sim U(0,1)$ ou $w_{ij}^{hid} \sim N(0, \sigma^2)$ , $i = 1, 2, \dots, p + q + 1$ , $j = 1, 2, \dots, N_{hid}$  |                          |
| II. Etapa de treino   |                          |
| i. Cálculo da ativação dos neurônios ocultos  |                          |
| — Para cada amostra ( $t = 1, 2, \dots, N_{tr}$ )   |                          |
| $\mathbf{v}_{hid}(t) = \mathbf{W}_{hid}\mathbf{x}(t)$   |                          |
| $\hat{\mathbf{y}}_{hid}(t) = \varphi(\mathbf{v}_{hid}(t))$  |                          |
| ii. Inclusão do bias: $\mathbf{y}_{hid}(t) = \begin{bmatrix} -1 \\ \hat{\mathbf{y}}_{hid}(t) \end{bmatrix}$   |                          |
| iii. Matrizes de amostras acumuladas  |                          |
| $\mathbf{Y}_{hid} = \begin{bmatrix} -1 & -1 & \dots & -1 \\ y_1^{hid}(1) & y_1^{hid}(2) & \dots & y_1^{hid}(N_{tr}) \\ y_2^{hid}(1) & y_2^{hid}(2) & \dots & y_2^{hid}(N_{tr}) \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_{hid}}^{hid}(1) & y_{N_{hid}}^{hid}(2) & \dots & y_{N_{hid}}^{hid}(N_{tr}) \end{bmatrix}$ e $\mathbf{Y}_{out} = \begin{bmatrix} y_1(1) & y_1(2) & \dots & y_1(N_{tr}) \\ y_2(1) & y_2(2) & \dots & y_2(N_{tr}) \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_{out}}(1) & y_{N_{out}}(2) & \dots & y_{N_{out}}(N_{tr}) \end{bmatrix}$ |                          |
| iv. Cálculo da matriz de pesos de saída $\mathbf{W}_{out}$  |                          |
| $\mathbf{W}_{out} = (\mathbf{Y}_{hid}\mathbf{Y}_{hid}^T)^{-1}\mathbf{Y}_{hid}\mathbf{Y}_{out}$  |                          |
| v. Cálculo do erro (com os dados de treino)   |                          |
| — Para cada amostra ( $t = 1, 2, \dots, N_{tr}$ )   |                          |
| $\hat{\mathbf{y}}(t) = \mathbf{v}_{out}(t) = \mathbf{W}_{out}\mathbf{y}_{hid}(t)$   |                          |
| $\xi_{tr}(t) = \mathbf{y}(t) - \hat{\mathbf{y}}(t)$   |                          |
| III. Teste ( $t = 1, 2, \dots, N_{tt}$ )  |                          |
| i. cálculo da ativação dos neurônios ocultos  |                          |
| $\mathbf{v}_{hid}(t) = \mathbf{W}_{hid}\mathbf{x}(t)$   |                          |
| $\hat{\mathbf{y}}_{hid}(t) = \varphi(\mathbf{v}_{hid}(t))$  |                          |
| ii. Inclusão do bias: $\mathbf{y}_{hid}(t) = \begin{bmatrix} -1 \\ \hat{\mathbf{y}}_{hid}(t) \end{bmatrix}$   |                          |
| iii. cálculo da saída   |                          |
| $\hat{\mathbf{y}}(t) = \mathbf{v}_{out}(t) = \mathbf{W}_{out}\mathbf{y}_{hid}(t)$   |                          |
| iv. Cálculo de erro:  |                          |
| $\xi_{tt}(t) = \mathbf{y}(t) - \hat{\mathbf{y}}(t)$   |                          |

Fonte: próprio autor.

## 3.4 MODELO NNARX

Como possuem pelo menos uma componente (camada) de aprendizagem não linear, os modelos obtidos a partir das redes MLP e ELM são essencialmente não lineares: autorregressivos ou não, com ou sem entradas exógenas. A não linearidade do modelo é conferida pela função de ativação.

Tomando por base o modelo NARX, o vetor de entradas da rede é definido segundo

$$\mathbf{x}(t) = [-1 \quad y(t-1) \quad \cdots \quad y(t-n_y) \quad u(t-\tau_d) \quad \cdots \quad u(t-\tau_d-n_u+1)]^T, \quad (3.37)$$

contendo as saídas autorregressivas e as entradas regressivas atrasadas pelo transporte. Por meio das representações matriciais, a propagação direta de informação (estimação da resposta da rede) pelo algoritmo adotado do MLP (Quadro 3.1) é, de forma simplificada, equacionada por

$$\hat{y}(t) = \varphi(\mathbf{W}_{out}\varphi(\mathbf{W}_{hid}\mathbf{x}(t))). \quad (3.38)$$

Enquanto isso, para o da ELM (Quadro 3.2), resume-se a:

$$\hat{y}(t) = \mathbf{W}_{out} \left[ \varphi^{-1}(\mathbf{W}_{hid}\mathbf{x}(t)) \right]. \quad (3.39)$$

Ambas as Equações (3.38) e (3.39) são proposições consonantes com a Equação (3.2). Pela obtenção a partir de redes neurais, ambos os modelos podem ser denominados também como NNARX (*Neural Network AutoRegressive with eXogenous inputs*) — a Tabela 3.1 contém a especificação dos padrões adotados para cada etapa de identificação neste estudo.

Tabela 3.1 — Definição das etapas de modelagem deste estudo.

| <b>Etapas de identificação</b> | <b>Especificação</b>   |
|--------------------------------|--|
| Aquisição de dados             | Thé (2010)<br>Problemas 1 e 2 (Seção 2.5)  |
| Representação matemática       | MLP e ELM  |
| Estrutura do modelo            | Descrito nas Seções 5 e 6  |
| Estimação dos parâmetros       | Algoritmo de retropropagação do erro (MLP)<br>Pseudo-inversa na saída (ELM)<br>Algoritmo genético (MLP e ELM)<br>Otimização por enxame de partículas (MLP e ELM) |
| Validação                      | MSE<br>Análise de resíduos   |

Fonte: próprio autor.

## 4 MECANISMOS DE OTIMIZAÇÃO

*“Uma lei geral, levando ao avanço de todos os seres orgânicos, ou seja, multiplicar, variar, deixar os mais fortes viverem e os mais fracos morrerem.”<sup>1</sup>*  
— Charles Darwin

Neste trabalho, dois mecanismos são utilizados para otimizar o treinamento de ambas as redes MLP e a ELM: o algoritmo genético e a otimização por enxames de partículas. Esta Seção se dedica a conceituar estes mecanismos e discutir sua integração às RNAs em questão.

### 4.1 ALGORITMO GENÉTICO E SEUS OPERADORES

Baseado nos estudos de Darwin sobre evolução e seleção natural, o algoritmo genético (AG) foi pioneiramente descrito por Holland (1975) e, então, por seus alunos, como Jong (1975). De forma ampla, um AG é qualquer modelo baseado em população que usa operadores de seleção e recombinação para gerar novos pontos de amostra em um espaço de busca.

De modo geral, a implementação de um AG é baseada em um ciclo evolutivo: a partir de uma população inicial, de candidatos a solução tipicamente aleatórios, avalia-se a aptidão e se seleciona os melhores, denominados de pais (seleção); posteriormente, novos candidatos (filhos) são gerados misturando-se características de diferentes pais (recombinação); ocorre nova avaliação e o ciclo se repete. Dessa maneira, as soluções mais precárias, ao longo do processo evolutivo, são paulatinamente descartadas, enquanto as melhores se perpetuam, em coerência à teoria da seleção natural das espécies mais adaptadas.

A semelhança biológica prossegue nas nomenclaturas: os candidatos a soluções são cunhados de cromossomos e suas características, de genes; os novos candidatos são gerados por operadores de cruzamento e mutação; denomina-se geração cada ciclo seleção-combinação; entre outros aspectos. Cabe ressaltar que o algoritmo é eficaz não simplesmente por sua base de inspiração, mas pela sua natureza probabilística e estocástica, como Goldberg (2013) extensivamente analisa para AGs sem mutação e com representação exclusivamente binários.

#### 4.1.1 Representação do cromossomo

A codificação de um cromossomo pode ser binária (os genes assumem valores 0 ou 1) ou real (valores reais quaisquer dentro de uma faixa definida) — ambas as representações são apreciadas neste estudo (Seção 4.3). Um cromossomo pode ser representado por um vetor, em

---

<sup>1</sup> Tradução nossa de trecho do capítulo VII do livro *The Origin of Species*, originalmente publicado em 1859. Citação em inglês: “*One general law, leading to the advancement of all organic beings, namely, multiply, vary, let the strongest live and the weakest die*” (DARWIN, 1859, p. 244).

que cada elemento representa um gene, binário ou real, conforme, respectivamente ( $N_b$  é o tamanho do cromossomo binário e  $N_r$ , do real):

$$\mathbf{z}_i^b = [z_{i1}^b \quad z_{i2}^b \quad \cdots \quad z_{iN_b}^b], \quad (4.1)$$

$$\mathbf{z}_i^r = [z_{i1}^r \quad z_{i2}^r \quad \cdots \quad z_{iN_r}^r], \quad (4.2)$$

#### 4.1.2 Métodos de seleção em AGs

Para que haja a seleção, primeiramente, os indivíduos da população devem ser avaliados segundo alguma métrica, mensurada pela função de aptidão,  $f_{apt}$ , por vezes associada a um ou mais objetivos de otimização. Após avaliação, os indivíduos devem ser escolhidos de acordo com a aptidão, sendo privilegiados aqueles de maior valor.

Para manter o tamanho da população,  $N_{pop}$ , constante ao longo do algoritmo, repete-se o método de seleção  $N_{pop}$  vezes, obtendo-se  $N_{pop}$  pais — cada par de pais geram normalmente dois filhos (Seção 4.1.3). Alguns métodos de seleção são: torneio, proporcional (roleta) e truncamento, entre outros. Neste trabalho, utiliza-se apenas o primeiro.

##### 4.1.2.1 Pressão de seleção

A pressão de seleção consiste na influência que o meio ambiente exerce na perpetuação dos genes ao longo das gerações (CAMARGO, 2006), mas pode ser entendida também como a tendência das características (genes) de um indivíduo ou de um grupo predominar na população. A pressão seletiva é um fator que depende do método de seleção e da métrica de avaliação da função de aptidão: se a chance de seleção for aproximadamente a mesma para todos os indivíduos, a pressão seletiva é fraca; do contrário, se um ou mais indivíduos tiverem uma tendência notoriamente maior a serem selecionados, diz-se que ela é forte.

A diversidade da população, por outro lado, depende da proximidade das características dos indivíduos. Quando há baixa diversidade na população, a pressão seletiva tende a ser menor, no entanto, métodos de seleção como o torneio conseguem, mesmo em baixa diversidade, forçar maior pressão seletiva.

##### 4.1.2.2 Seleção por torneio

Na seleção por torneio, sorteia-se aleatoriamente  $N_{tor}$  indivíduos da população e se seleciona aquele de maior aptidão (Quadro 4.1). O sorteio pode ocorrer com repetição, em que o indivíduo pode ser sorteado mais de uma vez, ou sem repetição. Vale destacar que a pressão de seleção neste método é proporcional ao número de indivíduos no torneio,  $N_{tor}$ .

Quadro 4.1 — Algoritmo de seleção por torneio.

| <b>ALGORITMO TORNEIO</b>   |
|--|
| I. Sortear $N_{tor}$ indivíduos da população;<br>II. Avaliar os indivíduos sorteados;<br>III. Armazenar o cromossomo do indivíduo de maior aptidão ( <i>j-ésimo</i> pai).<br>IV. Repetir os passos I a III até que a população de pais tenha tamanho $N_{pop}$ . |

Fonte: próprio autor.

#### 4.1.3 Cruzamento e mutação

Os operadores apresentam uma limitação dentro do algoritmo, uma taxa que delimita uma porcentagem de pais que são cruzados (probabilidade de cruzamento,  $p_c$ ) e de filhos que sofrem mutação (probabilidade de mutação,  $p_m$ ). Essas taxas são necessárias para que durante o treinamento do AG haja equilíbrio entre perpetuar as melhores características e manter uma população diversa o suficiente para troca de informações.

O cruzamento (ou *crossover*) consiste na troca de informações do cromossomo de um pai com o de outro, formando um novo cromossomo com características de ambos os pais e, portanto, precisa de uma taxa relativamente alta — porém, quando excessivamente elevada, causa efeitos destrutivos, sobretudo, em AGs sem elitismo (Seção 4.1.4). Enquanto isso, a mutação consiste em substituir um componente por outro aleatório— e deve ser aplicado a uma taxa bem mais baixa, para não haver perda considerável de informações importantes.

Existem vários métodos para ambos os operadores, que apresentam forte dependência com a representação do cromossomo. Neste trabalho, destacam-se os cruzamentos uniforme (binário) e o método BLX-alfa (real).

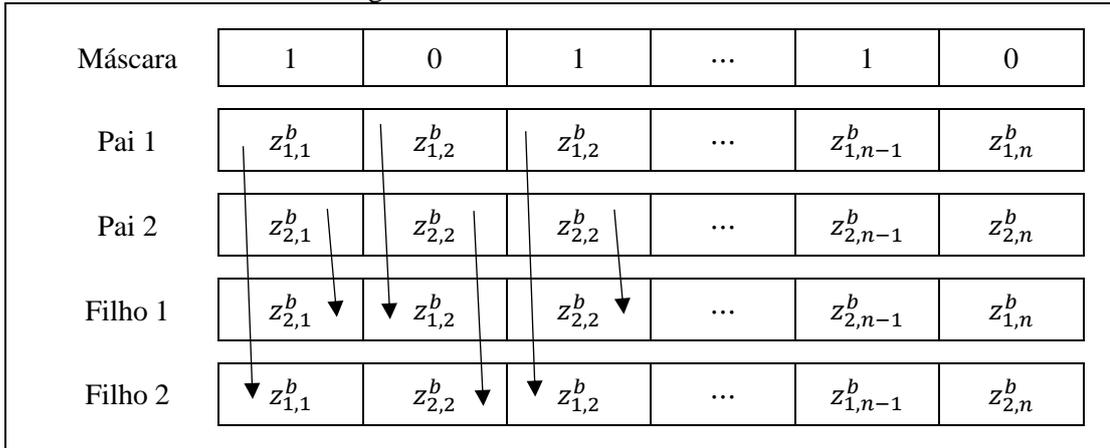
##### 4.1.3.1 Cruzamento uniforme

No cruzamento uniforme, selecionam-se dois cromossomos dentre o conjunto de pais e gera-se um vetor aleatório (máscara) de mesmo tamanho: nas posições nulas da máscara, o filho herda o gene do primeiro pai; nas unitárias, herda o do segundo — os mesmos pais podem ainda gerar um segundo filho com a mesma máscara (Figura 4.1). O algoritmo está no Quadro 4.2.

##### 4.1.3.2 Método BLX-alfa

Na representação real, uma forma simples de realizar o cruzamento é a utilização de alguma média entre os valores dos genes dos pais, entretanto, dessa forma, a diversidade da população e a região efetiva do espaço de busca tendem a diminuir geração a geração. O método BLX-alfa elimina esse viés ao introduzir um parâmetro  $\beta$ , gerado aleatoriamente a cada cruzamento e pertencente ao intervalo  $[-\alpha, 1 + \alpha]$  (algoritmo no Quadro 4.2).

Figura 4.1 — Cruzamento uniforme.



Fonte: próprio autor.

Quadro 4.2 — Algoritmos de cruzamento.

| <b>ALGORITMO CRUZAMENTO UNIFORME</b>  |
|---|
| I. Gerar valor aleatório $r$ ;<br>II. Se $r < p_c^b$ :<br>i. Gerar máscara aleatória de tamanho $N_c$ ( $N_c$ é o número de genes no cromossomo)<br>$[b_1 \ b_2 \ \dots \ b_n]$<br>ii. Para $i = 1, 2, \dots, n$<br>— se $b_i = 0 \rightarrow z_i^{b,filho1} = z_i^{b,pai1}$ e $z_i^{b,filho2} = z_i^{b,pai2}$<br>— se $b_i = 1 \rightarrow z_i^{b,filho1} = z_i^{b,pai2}$ e $z_i^{b,filho2} = z_i^{b,pai1}$<br>III. Senão, os pais são levados à próxima geração sem cruzamento<br>$\mathbf{z}^{b,filho1} = \mathbf{z}^{b,pai1}$ e $\mathbf{z}^{b,filho2} = \mathbf{z}^{b,pai2}$<br>IV. Repetir passos I a III até completar a nova população. |
| <b>ALGORITMO MÉTODO BLX-ALFA</b>  |
| I. Gerar valor aleatório $r$ ;<br>II. Se $r < p_c^r$ :<br>i. Gerar $\beta$ , tal que $\beta \in [-\alpha, 1 + \alpha]$<br>ii. Cruzar os pais<br>$\mathbf{z}_1^r = \mathbf{z}_1^{p,r} + \beta(\mathbf{z}_2^{p,r} - \mathbf{z}_1^{p,r})$<br>$\mathbf{z}_2^r = \mathbf{z}_2^{p,r} + \beta(\mathbf{z}_1^{p,r} - \mathbf{z}_2^{p,r})$<br>III. Senão, os pais são levados à próxima geração sem cruzamento<br>$\mathbf{z}^{r,filho1} = \mathbf{z}^{r,pai1}$ e $\mathbf{z}^{r,filho2} = \mathbf{z}^{r,pai2}$<br>IV. Repetir passos I a III até completar a nova população.   |

Fonte: próprio autor.

O cromossomo do filho é obtido a partir do dos pais, conforme

$$\mathbf{z}_1^r = \mathbf{z}_1^{p,r} + \beta(\mathbf{z}_2^{p,r} - \mathbf{z}_1^{p,r}); \tag{4.3}$$

um segundo filho pode ainda ser gerado com os mesmos pais e mesmo parâmetro  $\beta$ :

$$\mathbf{z}_2^r = \mathbf{z}_2^{p,r} + \beta(\mathbf{z}_1^{p,r} - \mathbf{z}_2^{p,r}). \quad (4.4)$$

A diversidade da população de filhos em relação a dos pais é tão maior quanto maior for  $\alpha$ ; cabe ressaltar que esse parâmetro deve estar entre permitir algum afastamento dos pais, porém sem perdas expressivas de suas características. Um valor usual para  $\alpha$  é 0,3.

#### 4.1.3.3 Mutação

O operador de mutação consiste em modificar um gene específico de um cromossomo — mais de um gene pode ser alterado, inclusive, desde que respeitando a taxa de mutação. Na representação binária, se o valor do gene é 0, passa a 1 e vice-versa; na representação real, o valor do gene deve ser substituído por um valor aleatório dentro do espaço de busca — o algoritmo para ambas as representações está no Quadro 4.3.

Quadro 4.3 — Algoritmos de mutação.

| <b>ALGORITMO MUTAÇÃO BINÁRIA</b>   |
|--|
| I. Para $i = 1, 2, \dots, N_c$ ( $N_c$ é o número de genes no cromossomo) <ul style="list-style-type: none"> <li>i. Gerar valor aleatório <math>r</math>;</li> <li>ii. Se <math>r &lt; p_m^b</math>:               <ul style="list-style-type: none"> <li>— se <math>z_i^b = 0 \rightarrow z_i^b = 1</math></li> <li>— se <math>z_i^b = 1 \rightarrow z_i^b = 0</math></li> </ul> </li> </ul> II. Repetir passo I para toda a população. |
| <b>ALGORITMO MUTAÇÃO REAL</b>  |
| I. Para $i = 1, 2, \dots, N_c$ ( $N_c$ é o número de genes no cromossomo) <ul style="list-style-type: none"> <li>i. Gerar valor aleatório <math>r</math>;</li> <li>ii. Se <math>r &lt; p_m^r</math>:               <ul style="list-style-type: none"> <li><math>z_i^r =</math> valor aleatório dentro do espaço de busca.</li> </ul> </li> </ul> II. Repetir passo I para toda a população.  |

Fonte: próprio autor.

#### 4.1.4 Elitismo

Por vezes, um algoritmo genético, mesmo que estruturado de forma adequada, não consegue convergir por conta do contínuo processo de construção e desconstrução de soluções mais próximas da ótima em função da contínua troca de informações do cruzamento (e, em bem menor influência, da inovação também contínua da mutação). Para evitar essa situação, pode-se acrescentar o elitismo ao algoritmo: de forma ampla, consiste em conservar o(s) melhor(es) indivíduos da população na geração diretamente para a seguinte. Vale ressaltar que a “elite” também deve participar do processo de seleção e de cruzamento (uma vez que são os melhores, possuem informação valiosa a ser compartilhada com os demais).

#### 4.1.5 Convergência do algoritmo

Assim como as redes neurais, o AG precisa de critérios que sinalizem o momento de parada do algoritmo, entre eles, pode se citar alguns típicos:

- a) **número máximo de gerações**,  $N_g$ , o qual garanta que, mesmo sem convergir, o algoritmo seja eventualmente encerrado;
- b) **convergência da função objetivo**, quando, entre consecutivas gerações, não mais houver melhoria significativa;
- c) **aptidão ótima alcançada**, quando se conhecer previamente seu valor para comparativo.

No entanto, à exceção do caso não usual da alínea c, esses critérios de parada não garantem que o AG obteve resultados satisfatórios após o treinamento; no caso específico da alínea b, caso o algoritmo não tenha obtido, ocorre uma convergência prematura. O algoritmo converge prematuramente quando há uma dominância de um ou mais indivíduos ainda não ótimos sobre a população.

A pressão seletiva e a diversidade da população são fatores fortemente relacionados à velocidade de convergência do algoritmo. Em casos de convergência prematura, devem ser utilizados mecanismos de diversificação da população (como: o aumento das taxas de mutação e crossover e não inserção de filhos duplicados na população de novas gerações) ou de aumento da pressão de seleção.

Além disso, a população inicial preferencialmente deve cobrir todo o espaço de busca: se ela ficar restrita a um nicho, provavelmente, o algoritmo convergirá para um ótimo local dentro da região. Posteriormente, a cada nova geração, a população deve permanecer dentro do espaço de busca definido, mesmo que já não o cubra tão extensivamente. Uma das formas de manter os filhos de codificação real no espaço de busca é testá-los para a faixa de valor que este cobre,  $z_{ij}^r \in [z_{j,min}^r, z_{j,max}^r]$ , e limitá-los as extremidades, conforme

$$\begin{cases} z_{ij}^r(n+1) < z_{j,min}^r \rightarrow z_{ij}^r(n+1) = z_{j,min}^r \\ z_{ij}^r(n+1) > z_{j,max}^r \rightarrow z_{ij}^r(n+1) = z_{j,max}^r \end{cases} \quad (4.5)$$

Como cada gene representa uma característica do cromossomo, cada um possui valores máximo e mínimo próprio.

No Quadro 4.4, é apresentado uma sequência de passos de um AG genérico a partir dos algoritmos de seleção (Quadro 4.1), cruzamento (Quadro 4.2) e mutação (Quadro 4.3). Esse passo a passo está suscetível a adaptações específicas a depender do problema a aplicar.

Quadro 4.4 — Algoritmo genético.

| <b>ALGORITOMO GENÉTICO</b>   |                                     |
|--|-------------------------------------|
| <b>Configuração do AG</b>  |                                     |
| $N_{pop}$ : número de cromossomos na população   | $p_c$ : probabilidade de cruzamento |
| $N_g$ : número de gerações de treinamento  | $p_m$ : probabilidade de mutação    |
| <b>Algoritmo</b>   |                                     |
| I. Etapa pré-treino ( $n = 0$ )  |                                     |
| i. encapsular o problema a ser resolvido dentro da função de aptidão, $f_{apt}$ .  |                                     |
| ii. inicializar população dentro do espaço de busca:   |                                     |
| $z_{ij} \sim U(\mu, \sigma^2), i = 1, 2, \dots, N_{pop}, j = 1, 2, \dots, N_c.$  |                                     |
| II. Etapa de treino. Para cada geração ( $n = 1, 2, \dots, N_g$ )  |                                     |
| i. para cada cromossomo ( $i = 1, 2, \dots, N_{pop}$ )   |                                     |
| — avaliar a aptidão: $f_{apt}(z_i)$ .  |                                     |
| ii. realizar seleção de pais da próxima geração:   |                                     |
| — roleta ou torneio (Quadro 4.1).  |                                     |
| iii. produzir os cromossomos dos filhos a partir dos operadores:   |                                     |
| — cruzamento (Quadro 4.2) e mutação (Quadro 4.3);  |                                     |
| — verificar e adequar os novos cromossomos ao espaço de busca:   |                                     |
| $\begin{cases} z_{ij}^r(n+1) < z_{j,min}^r \rightarrow z_{ij}^r(n+1) = z_{j,min}^r \\ z_{ij}^r(n+1) > z_{j,max}^r \rightarrow z_{ij}^r(n+1) = z_{j,max}^r \end{cases}$ |                                     |
| iv. atualizar a população:   |                                     |
| — substituir pelos cromossomos dos filhos;   |                                     |
| — preservar os melhores indivíduos da geração anterior.  |                                     |
| v. verificar critérios de parada.  |                                     |

Fonte: próprio autor.

## 4.2 OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS

O algoritmo de otimização por enxame de partículas (do inglês, *Particle Swarm Optimization*, PSO), assim como o AG, apresenta base biológica de inspiração: o comportamento social e cooperativo apresentado por inúmeras espécies em enxames, de maneira a realizar as suas necessidades no espaço de pesquisa.

### 4.2.1 Posição e velocidade das partículas

Como o AG, a PSO é baseada em um algoritmo de população, cujos indivíduos são atualizados a cada iteração em busca da melhor solução. No entanto, enquanto o AG utiliza operadores de seleção, cruzamento e mutação nos cromossomos, na PSO, cada candidato a solução do problema proposto é tratado como uma partícula  $i$  em um enxame, que ocupa uma posição  $\mathbf{z}_i$ . Esta é representada por um vetor dentro de um espaço multidimensional, em que cada dimensão corresponde a uma característica da solução (à semelhança do gene no AG):

$$\mathbf{z}_i = [z_{i,1} \quad z_{i,2} \quad \cdots \quad z_{i,N_D}], \quad (4.6)$$

em que  $N_D$  é a quantidade de dimensões (tamanho da partícula). Cada valor desse vetor posição é ajustado por uma velocidade própria, formando um conjunto de valores de mesma dimensão:

$$\mathbf{v}_i = [v_{i,1} \quad v_{i,2} \quad \cdots \quad v_{i,N_D}]. \quad (4.7)$$

A posição e a velocidade das partículas são ajustadas a cada iteração  $n$  e podem ser agrupadas em matrizes que incluam toda a população, conforme, respectivamente:

$$\mathbf{Z}(n) = \begin{bmatrix} \mathbf{z}_1(n) \\ \mathbf{z}_2(n) \\ \vdots \\ \mathbf{z}_{N_{pop}}(n) \end{bmatrix}, \quad (4.8)$$

$$\mathbf{V}(n) = \begin{bmatrix} \mathbf{v}_1(n) \\ \mathbf{v}_2(n) \\ \vdots \\ \mathbf{v}_{N_{pop}}(n) \end{bmatrix}. \quad (4.9)$$

Além deles, para ajuste da população, o algoritmo deve armazenar a melhor posição individual  $\mathbf{z}_{p,i}$  para cada partícula  $i$  (*personal best*) — matricialmente representada por

$$\mathbf{Z}_p = \begin{bmatrix} \mathbf{z}_{p,1} \\ \mathbf{z}_{p,2} \\ \vdots \\ \mathbf{z}_{p,N_{pop}} \end{bmatrix} \quad (4.10)$$

—, além da melhor solução entre todas  $\mathbf{z}_g$  (*global best*). Essa avaliação ocorre baseada na função de aptidão,  $f_{apt}$ .

#### 4.2.2 Atualização do enxame

O ajuste da posição depende da representação da partícula, se real ou se binária, porém ocorre em direção às melhores posições já encontradas (tanto as individuais quanto a global), em proporções predefinidas no algoritmo. Para codificação real, velocidade e posição são ajustados, respectivamente, por

$$v_{ij}(n+1) = c_0 v_{i,j}(n) + c_1 r_1 (z_{p,i,j} - z_{i,j}(n)) + c_2 r_2 (z_g - x_{ij}(n)), \quad (4.11)$$

$$z_{ij}^r(n+1) = z_{ij}^r(n) + v_{ij}^r(n+1), \quad (4.12)$$

em que os coeficientes são definidos a seguir:

- a)  $c_1$ , parâmetro cognitivo: atualização em relação a melhor posição individual da partícula;
- b)  $c_2$ , parâmetro social: atualização em relação a melhor posição global da população;
- c)  $c_0$ , ponderação de inércia: manutenção da velocidade da partícula;
- d)  $r_1$ , ponderação aleatória do parâmetro cognitivo, entre 0 e 1;
- e)  $r_2$ , ponderação aleatória do parâmetro social, entre 0 e 1.

Os valores aleatórios  $r_1$  e  $r_2$  devem ser gerados a cada iteração do algoritmo para cada dimensão  $j$  da partícula  $i$ , enquanto os parâmetros  $c_0$ ,  $c_1$  e  $c_2$  são únicos para toda a população. Uma forma prática de se estimar estes parâmetros dá-se por meio de

$$\omega = \phi - 1 + \frac{1}{\sqrt{\phi^2 - 2\phi}}, \quad (4.13)$$

$$c_0 = \omega, \quad (4.14)$$

$$c_1 = c_2 = \omega \cdot \phi, \quad (4.15)$$

em que  $\omega$  e  $\phi$  são variáveis auxiliares, em que geralmente  $\phi \geq 2$ .

Por outro lado, para a população binária, apesar de o ajuste da velocidade das partículas permanecer conforme Equação (4.11), a posição requer atualização diferente:

$$z_{ij}^b(n+1) = \begin{cases} 1, & \text{se } R < S(v_{ij}^b(n+1)) \\ 0, & \text{demais casos} \end{cases}, \quad (4.16)$$

em que  $R$  é um valor aleatório entre 0 e 1 e  $S(\cdot)$ , uma função sigmoide, tipicamente a logística (a mesma utilizada como ativação em redes neurais, Seção 3.2.2). Nesse caso, o ajuste de  $z_{ij}^b$  é dito probabilístico.

#### 4.2.3 Considerações sobre convergência

As considerações sobre convergência para o AG (Seção 4.1.5) se aplicam também à PSO, respeitando-se as devidas equivalências. Os critérios de parada típicos são os mesmos: número máximo de iterações,  $N_{it}$ ; convergência da função objetivo; e aptidão ótima alcançada.

A convergência prematura deve ser prevenida pela configuração dos parâmetros  $c_0$ ,  $c_1$  e  $c_2$ ; a população inicial também deve preferencialmente cobrir todo o espaço de busca. Quanto à atualização das partículas, as posições devem permanecer dentro do espaço de busca,  $z_{ij}^r \in [z_{j,min}^r, z_{j,max}^r]$ , porém também a velocidade deve ser corrigida nesse caso, conforme

$$\begin{cases} z_{ij}^r(n+1) < z_{j,min}^r \rightarrow \begin{cases} z_{ij}^r(n+1) = z_{j,min}^r \\ v_{ij}^r(n+1) = 0 \end{cases} \\ z_{ij}^r(n+1) > z_{j,max}^r \rightarrow \begin{cases} z_{ij}^r(n+1) = z_{j,max}^r \\ v_{ij}^r(n+1) = 0 \end{cases} \end{cases} \quad (4.17)$$

Além disso, a velocidade deve respeitar os limites de sua própria faixa do espaço de busca, tal que  $v_{ij}^r \in [v_{j,min}^r, v_{j,max}^r]$ , conforme

$$\begin{cases} v_{ij}^r(n+1) < v_{j,min}^r \rightarrow v_{ij}^r(n+1) = v_{j,min}^r \\ v_{ij}^r(n+1) > v_{j,max}^r \rightarrow v_{ij}^r(n+1) = v_{j,max}^r \end{cases} \quad (4.18)$$

Na codificação binária, a limitação da velocidade também é respeitada:  $v_{ij}^b \in [v_{j,min}^b, v_{j,max}^b]$ ; a posição, entretanto, por assumir somente os valores 0 e 1, por definição, não precisa de um artifício limitador para mantê-la no espaço de busca.

Por fim, para controlar a diversidade da população ao longo das iterações, os parâmetros  $c_1$  e  $c_2$  podem ser mantidos constantes, enquanto  $c_0$  varia conforme alguma função linear, como

$$c_0(n+1) = c_{0i} - (c_{0i} - c_{0f}) \frac{n}{N_{it}}, \quad (4.19)$$

em que  $c_{0i}$  e  $c_{0f}$  são os valores inicial e final para a ponderação de inércia. Por exemplo, os termos inicial e final podem variar 10% ou 20% do valor  $\omega$ , obtido na Equação (4.13).

Essa variação de  $c_0$  evita a convergência prematura e torna a aprendizagem mais precisa nas últimas iterações. Vale ressaltar, no entanto, que a ponderação de inércia para indivíduos reais  $c_0^r$  e para binários  $c_0^b$  devem variar distintamente. Para  $c_0^r$ , o valor inicial deve ser maior,

$$c_{0i}^r > c_{0f}^r, \quad (4.20)$$

a fim de evitar mudanças bruscas de direção na velocidade e, portanto, manter as partículas  $\mathbf{z}_i$  espalhadas pelo espaço de busca inicialmente, conforme Equação (4.12).

Por outro lado, a posição em partículas binárias tem ajuste não linear em função da velocidade, conforme Equação (4.16). Assim, para  $S(\cdot)$  uma função logística, uma maior velocidade promove menor diversificação da população, mantendo as características das partículas nos extremos 1 ou 0 sem mudanças em direção contrária. Portanto, para  $c_0^b$ , o valor inicial deve ser menor:

$$c_{0i}^b < c_{0f}^b. \quad (4.21)$$

No Quadro 4.5, apresenta-se os passos sequenciais de uma PSO genérica.

Quadro 4.5 — Algoritmo PSO.

| <b>ALGORITOMO PSO</b>   |  |
|---|--|
| <b>Configuração da PSO</b>  |  |
| $N_{pop}$ : número de partículas na população   | $c_0^b, c_0^r$ : ponderação de inércia (binária e/ou real) |
| $N_{it}$ : número de iterações de treinamento   | $c_1^b, c_1^r$ : parâmetro cognitivo (binária e/ou real)   |
|   | $c_2^b, c_2^r$ : parâmetro social (binária e/ou real)      |
| <b>Algoritmo</b>  |  |
| I. Etapa pré-treino ( $n = 0$ ):  |  |
| i. encapsular o problema a ser resolvido dentro da função de aptidão, $f_{apt}$ .   |  |
| ii. inicializar população dentro do espaço de busca:  |  |
| $z_{ij} \sim U(\mu, \sigma^2), i = 1, 2, \dots, N_{pop}, j = 1, 2, \dots, N_D.$   |  |
| II. Etapa de treino. Para cada geração ( $n = 1, 2, \dots, N_{it}$ ):   |  |
| i. para cada partícula ( $i = 1, 2, \dots, N_{pop}$ ):  |  |
| — avaliar a aptidão: $f_{apt}(\mathbf{z}_i)$ .  |  |
| ii. Determinar melhores partículas:   |  |
| — <i>personal best</i> :  |  |
| $\mathbf{z}_p = \begin{bmatrix} \mathbf{z}_{p,1} \\ \mathbf{z}_{p,2} \\ \vdots \\ \mathbf{z}_{p,N_{pop}} \end{bmatrix}.$  |  |
| — <i>global best</i> : $\mathbf{z}_g$ .   |  |
| iii. atualizar posições e velocidades. Para cada partícula ( $i = 1, 2, \dots, N_{pop}$ ):  |  |
| — se binário:   |  |
| $\mathbf{v}_i(n+1) = c_0 \mathbf{v}_i(n) + c_1 r_1 (\mathbf{z}_{p,i} - \mathbf{z}_i(n)) + c_2 r_2 (\mathbf{z}_g^b - \mathbf{z}_i(n));$  |  |
| $z_{ij}^b(n+1) = \begin{cases} 1, & \text{se } R < S(v_{ij}^b(n+1)) \\ 0, & \text{demais casos} \end{cases} \text{ (para cada dimensão } j);$   |  |
| — se real:  |  |
| $\mathbf{v}_i^r(n+1) = c_0 \mathbf{v}_i^r(n) + c_1 r_1 (\mathbf{z}_{p,i}^r - \mathbf{z}_i^r(n)) + c_2 r_2 (\mathbf{z}_g^r - \mathbf{z}_i^r(n));$  |  |
| $\mathbf{z}_i^r(n+1) = \mathbf{z}_i^r(n) + \mathbf{v}_i^r(n+1).$  |  |
| iv. verificar e adequar as novas partículas ao espaço de busca. Para cada dimensão ( $j = 1, 2, \dots, N_{pop}$ ) de cada partícula ( $i = 1, 2, \dots, N_{pop}$ ):   |  |
| — se binário:   |  |
| $\begin{cases} v_{ij}^b(n+1) < v_{j,min}^b \rightarrow v_{ij}^b(n+1) = v_{j,min}^b \\ v_{ij}^b(n+1) > v_{j,max}^b \rightarrow v_{ij}^b(n+1) = v_{j,max}^b \end{cases}.$   |  |
| — se real:  |  |
| $\begin{cases} z_{ij}^r(n+1) < z_{j,min}^r \rightarrow \begin{cases} z_{ij}^r(n+1) = z_{j,min}^r \\ v_{ij}^r(n+1) = 0 \end{cases} \\ z_{ij}^r(n+1) > z_{j,max}^r \rightarrow \begin{cases} z_{ij}^r(n+1) = z_{j,max}^r \\ v_{ij}^r(n+1) = 0 \end{cases} \end{cases};$ |  |
| $\begin{cases} v_{ij}^r(n+1) < v_{j,min}^r \rightarrow v_{ij}^r(n+1) = v_{j,min}^r \\ v_{ij}^r(n+1) > v_{j,max}^r \rightarrow v_{ij}^r(n+1) = v_{j,max}^r \end{cases}.$   |  |
| v. verificar critérios de parada.   |  |

Fonte: próprio autor.

### 4.3 O TREINAMENTO NEURO-OTIMIZADO DAS REDES NEURAIAS

Os modelos neuro-otimizados abordados neste trabalho são desenvolvidos com o MLP treinado por AG (*Genetic Algorithm* MLP, GA-MLP) ou por PSO (*Particle Swarm Optimization* MLP, PSO-MLP) e a ELM treinada por AG (*Genetic Algorithm* ELM, GA-ELM) ou por PSO (*Particle Swarm Optimization* ELM, PSO-ELM), em vez de por seus métodos tradicionais apresentados na Seção 3. As redes neurais, nesse caso, funcionam como a função de aptidão dos mecanismos de otimização e o algoritmo é idêntico ao apresentado nos Quadros 4.4 (AG) e Quadro 4.5 (PSO), em que avaliar a aptidão pode ser entendido como a propagação direta das informações pelas redes, conforme Quadro 3.1 (MLP) e Quadro 3.2 (ELM).

Esse treinamento neuro-otimizado é abordado de três formas no presente estudo. A primeira delas tem foco apenas na atualização dos pesos (codificação exclusivamente real), com um valor constante de neurônios ocultos; a aptidão, nesse caso, é dada pelo inverso de alguma métrica de erro na resposta estimada da rede durante o treino, de acordo com

$$f_{apt} = \frac{1}{erro} ; \quad (4.22)$$

As outras duas permitem a variação de  $N_{hid}$  entre iterações. Na segunda abordagem, a aptidão permanece sendo obtida pela Equação (4.22), enquanto, na terceira, promove-se a minimização de  $N_{hid}$ , para uma rede mais eficiente, conforme

$$f_{apt} = \frac{1}{(erro)^{\gamma_e} \cdot (N_{hid})^{\gamma_h}} , \quad (4.23)$$

em que os expoentes  $\gamma_e$  e  $\gamma_h$  ponderam de forma arbitrária os fatores para a aptidão — notoriamente,  $\gamma_e > \gamma_h$ , uma vez que a minimização do erro é o objetivo principal do algoritmo.

#### 4.3.1 Pesos (codificação real)

O cromossomo (no AG) ou a partícula (na PSO) de codificação real são utilizados para representar os pesos nas camadas oculta da ELM e oculta e de saída do MLP. Dessa forma, a representação é dada por uma matriz de dimensão  $N_{hid} \times N_{in}$  para camada oculta e  $N_{hid} \times N_{out}$  para a saída.

Tanto a GA-ELM quanto a PSO-ELM permanecem com sua camada de saída sendo obtida pela pseudoinversa (Seção 3.3.2) e somente os pesos ocultos são objeto de otimização. Apesar disso, o problema de otimização da ELM envolve muitos parâmetros particulares (pesos ocultos) a serem estimados e, em geral, uma solução subótima, porém próxima o bastante da ótima, normalmente, é tido como satisfatória.

O treinamentos da GA-ELM e da PSO-ELM com  $N_{hid}$  constante concentram esforços exclusivamente no ajuste dos pesos da camada escondida, de forma a encontrar a matriz ótima  $\mathbf{W}_{hid}$  tal que a pseudoinversa da Equação (3.33) obtenha o conjunto de pesos  $\mathbf{W}_{out}$  que produza a saída de menor erro. Portanto, cada cromossomo ou partícula da população representa uma matriz candidata a melhor  $\mathbf{W}_{hid}$ .

Por outro lado, GA-MLP e PSO-MLP têm ajuste dos pesos exclusivamente pelos seus respectivos mecanismos de otimização: não somente da camada oculta, como também da de saída. Isso torna a o treinamento neuro-otimizado da MLP caracteristicamente mais complexo que o da ELM. Dessa forma, cada indivíduo da população apresenta duas matrizes  $\mathbf{W}_{hid}$  e  $\mathbf{W}_{out}$ .

#### 4.3.2 Número de neurônios ocultos (codificação binária)

Nos quatro algoritmos (GA-ELM, PSO-ELM, GA-MLP e PSO-MLP), em caso de a quantidade de  $N_{hid}$  ser variável, cada indivíduo da população ainda teria um vetor binário no treinamento.

Essa codificação binária é destinada à manipulação do número de neurônios ocultos e está presente apenas nos algoritmos que permitem a variação de  $N_{hid}$  entre as iterações, havendo ou não minimização explícita por meio da função de aptidão. Para manipulação da quantidade de neurônios  $N_{hid}$  se estabelece um vetor em que cada *bit* representa um nó oculto: o valor unitário para aqueles ativados e nulo para os desativados.

Um neurônio desativado tem seus pesos removidos das matrizes de pesos  $\mathbf{W}_{hid}$  e  $\mathbf{W}_{out}$ , ou seja, o sinal de entrada não chega a um nó nesse estado, nem sua possível resposta é considerada na camada de saída (e, portanto, não participa da soma de informações de outros neurônios). A restrição de variação de  $N_{hid}$  é tal que  $N_{hid}^{min} < N_{hid} \leq N_{hid}^{max}$ , em que:

- a)  $N_{hid} \leq N_{hid}^{max}$ : estabelecida pela própria definição do código, em “ativar” e “desativar” neurônios do indivíduo real pelo indivíduo binário;  $N_{hid}^{max}$  é o próprio tamanho do vetor de *bits*;
- b)  $N_{hid} > N_{hid}^{min}$ : aplicada após operadores de cruzamento e de mutação (no AG) ou da atualização da posição pela velocidade (na PSO), verificando se, na nova população, qualquer indivíduo apresenta uma quantidade inferior a  $N_{hid}^{min}$  de valores binários simultaneamente em zero; em caso positivo, a configuração binária de  $N_{hid}$  do indivíduo na geração ou na iteração anterior é mantida e o ajuste se dá apenas na codificação real.

## 5 NEURO-OTIMIZAÇÃO PARA APROXIMAÇÃO DE FUNÇÃO

*“By far the greatest danger of Artificial Intelligence is that people conclude too early that they understand it.”<sup>1</sup>*  
 — Eliezer Yudkowsky

Nesta Seção, apresentam-se os resultados dos modelos GA-ELM, PSO-ELM, GA-MLP e PSO-MLP aplicados aos dois problemas de aproximação de função denotados na Seção 2.5. Analisam-se os impactos da neuro-otimização em relação às redes MLP e ELM com suas metodologias tradicionais, bem como se realiza comparativo de desempenho.

Cada um dos algoritmos propostos é executado 100 vezes, obtendo-se uma faixa de resposta para cada um deles, a fim de se verificar a confiabilidade de uma execução aleatória produzir uma rede ajustada satisfatoriamente. No geral, a análise de qualidade abordada dos modelos obtidos leva em consideração a média e a dispersão das execuções. As RNAs utilizam como função de ativação a logística e normalização da saída no intervalo  $[0, +1]$  (Tabela 5.1).

Tabela 5.1 — Parâmetros gerais dos algoritmos MLP, ELM e versões otimizadas.

| Parâmetro               | Símbolo          | Valor adotado                                |
|-------------------------|------------------|--|
| Quantidade de execuções | —                | 100  |
| Função de ativação      | $\varphi(\cdot)$ | Logística: $\varphi(v) = \frac{1}{1+e^{-v}}$ |
| Normalização da saída   | —                | $[0, +1]$                                    |

Fonte: próprio autor.

### 5.1 ELM COM TREINAMENTO DA CAMADA OCULTA: GA-ELM E PSO-ELM

A seguir, são apresentados separadamente os resultados para os três vieses de neuro-otimização da ELM:  $N_{hid}$  constante;  $N_{hid}$  variável, porém não explicitamente minimizado; e  $N_{hid}$  minimizado. Para comparativo, os resultados dos algoritmos da GA-ELM e da PSO-ELM são discutidos juntos para cada abordagem.

#### 5.1.1 $N_{hid}$ constante

Nos algoritmos com  $N_{hid}$  constante, estabeleceram-se os valores de 5, 10 e 20 neurônios para o Problema 1 e de 10, 20, 50 e 250 para o Problema 2 (Tabela 5.2), para todos os modelos. Os parâmetros de treinamento correspondentes para a GA-ELM (Tabela 5.3) e para a PSO-

<sup>1</sup> Tradução nossa: “De longe, o maior perigo da Inteligência Artificial é que as pessoas concluem muito cedo que a entendem”.

ELM (Tabela 5.4) são adotados de forma semelhante ( $N_g = N_{it} = 100$  e  $N_{pop} = 150$ ). As demais configurações se definiram empiricamente a partir dos valores típicos apresentados e das discussões sobre os algoritmos otimizadores na Seção 4.

Tabela 5.2 — Quantidade de neurônios escondidos.

| Parâmetro                       | Símbolo   | Valores Típicos | Valor adotado |                   |
|---------------------------------|-----------|-----------------|---------------|-------------------|
|                                 |           |                 | Problema 1    | Problema 2        |
| Quantidade de neurônios ocultos | $N_{hid}$ | $N_{hid} > 1$   | [5; 10; 20]   | [10; 20; 50; 250] |

Fonte: próprio autor.

Tabela 5.3 — Parâmetros reais de treinamento da ELM por AG (GA-ELM).

| Parâmetro                        | Símbolo   | Valores Típicos    | Valor adotado   |            |
|----------------------------------|-----------|--------------------|-----------------|------------|
|                                  |           |                    | Problema 1      | Problema 2 |
| Quantidade de gerações           | $N_g$     | $N_g \geq 100$     | $N_g = 100$     |            |
| Tamanho da população             | $N_{pop}$ | $N_{pop} \geq 100$ | $N_{pop} = 150$ |            |
| Probabilidade de cruzamento real | $p_c^r$   | $p_c^r \geq 0,7$   | $p_c^r = 0,9$   |            |
| Probabilidade de mutação real    | $p_m^r$   | $p_m^r \leq 0,05$  | $p_m^r = 0,05$  |            |
| Operador BLX-alfa                | $\alpha$  | $\alpha \leq 1$    | $\alpha = 0,3$  |            |
| Número de indivíduos no torneio  | $N_{tor}$ | potências de 2     | $N_{tor} = 32$  |            |

Fonte: próprio autor.

Tabela 5.4 — Parâmetros reais de treinamento da ELM por PSO (PSO-ELM).

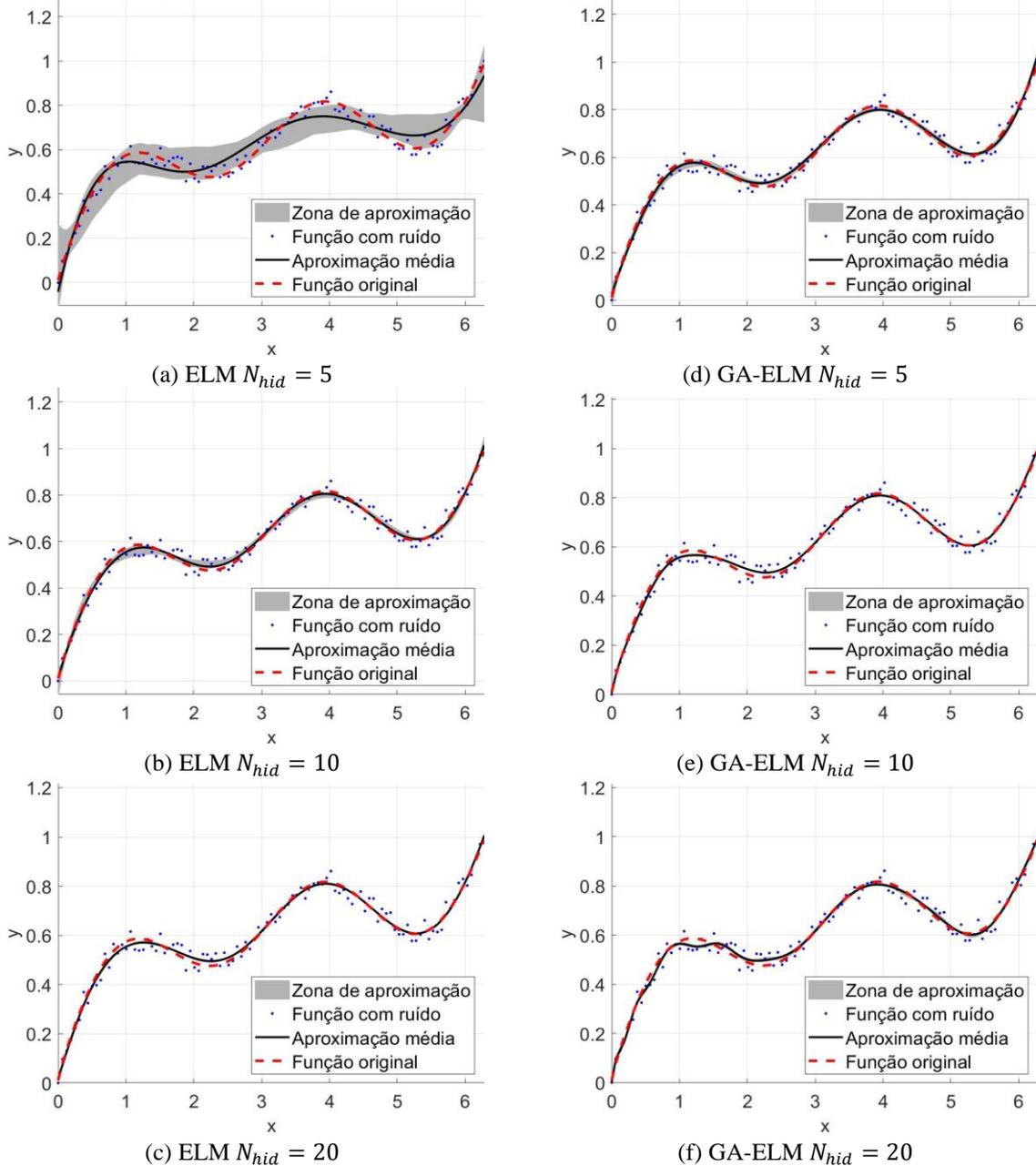
| Parâmetro                    | Símbolo                   | Valores Típicos    | Valor adotado                          |                |
|------------------------------|---------------------------|--------------------|--|----------------|
|                              |                           |                    | Problema 1                             | Problema 2     |
| Quantidade de iterações      | $N_{it}$                  | $N_{it} \geq 100$  | $N_{it} = 100$                         | $N_{it} = 200$ |
| Tamanho da população         | $N_{pop}$                 | $N_{pop} \geq 100$ | $N_{pop} = 150$                        |                |
| Peso cognitivo real          | $c_1^r$                   | Equação (4.15)     | $c_1^r = 1,35$                         |                |
| Peso social real             | $c_2^r$                   | Equação (4.15)     | $c_2^r = 1,35$                         |                |
| Coefficiente de inércia real | $c_{0i}^r$ e $c_{0f}^r$   | Equação (4.14)     | $c_{0i}^r = 0,7$ e $c_{0f}^r = 1,1$    |                |
| Posição das partículas       | $z_{max}^r$ e $z_{min}^r$ | —                  | $z_{max}^r = 10$ e $z_{min}^r = -10$   |                |
| Velocidade das partículas    | $v_{max}^r$ e $v_{min}^r$ | —                  | $v_{max}^r = 0,5$ e $v_{min}^r = -0,5$ |                |

Fonte: próprio autor.

Na Figura 5.1, são apresentados os resultados gráficos da GA-ELM para aproximação de função monovariável do Problema 1, bem como os da ELM para as mesmas quantidades de

neurônio a fim de comparação. Para cada modelo, são mostrados a média das soluções obtidas nas 100 execuções, a zona de aproximação e os dados da função original, Equação (2.1), antes e depois da inclusão de ruído.

Figura 5.1 — Aproximação de função do Problema 1 pela ELM e pela GA-ELM ( $N_{hid}$  constante).



Fonte: próprio autor.

Neste estudo, entenda-se por zona de aproximação o intervalo gráfico que compreende a solução obtida pelo algoritmo em todas as execuções. Esse critério de avaliação indica o grau de previsibilidade no treinamento da rede, o quão próximas são as soluções oferecidas pelo algoritmo em execuções independentes, apesar da inicialização aleatória de parâmetros. Quanto

maior o intervalo, menor a confiabilidade de que, repetido o treinamento do algoritmo, ele apresente a mesma solução.

Alguns pontos que merecem destaque para os resultados da GA-ELM em relação aos da ELM:

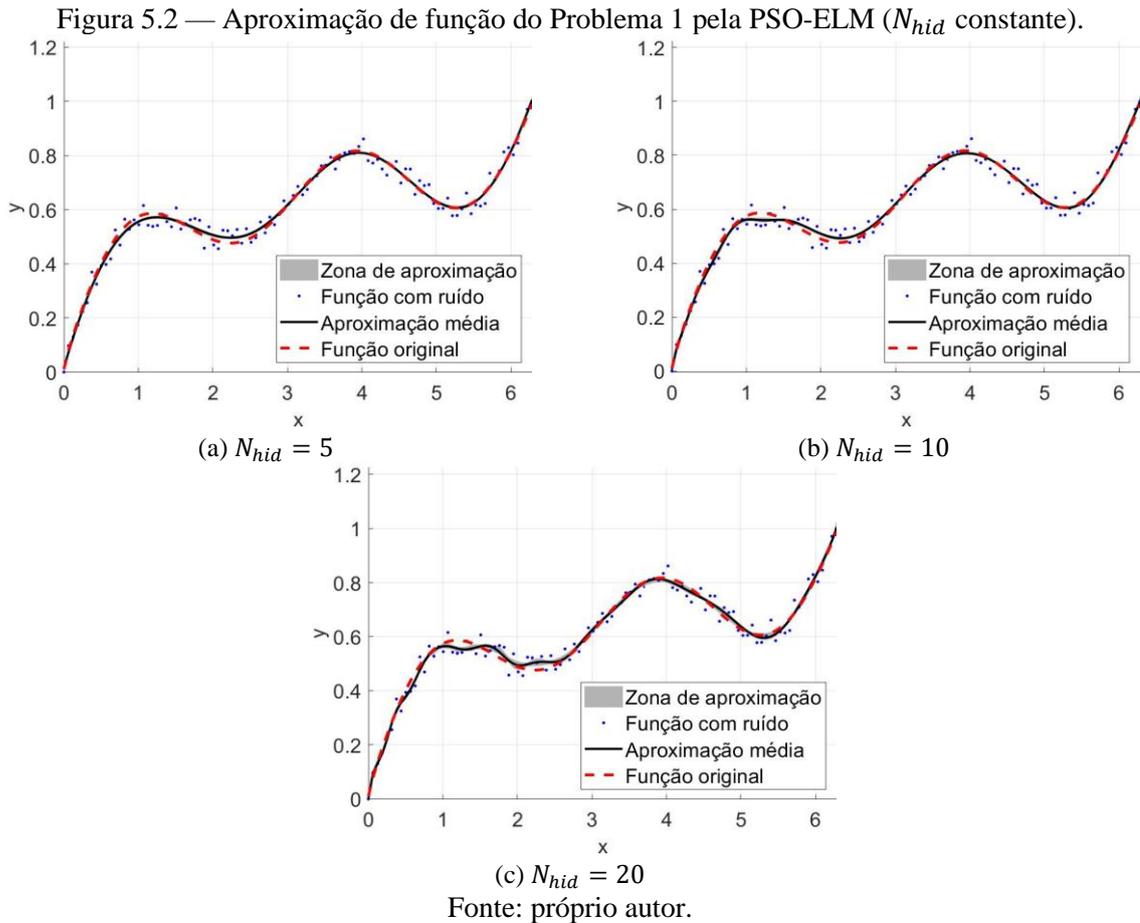
- c) redução da zona de aproximação, sobretudo, para valores de  $N_{hid}$  menores<sup>2</sup>;
- d) sobreajuste, para valores de  $N_{hid}$  maiores (em que a ELM convencional consegue bons ajustes)<sup>3</sup>;
- e) maior eficiência: resultado médio da GA-ELM para  $N_{hid}$  reduzido é semelhante aos melhores resultados obtidos pelo algoritmo convencional (com maior  $N_{hid}$ ).

Os três pontos abordados acima (alíneas a, b e c) também são válidos para a comparação entre os modelos ELM (Figura 5.1a, b, c) e PSO-ELM (Figura 5.2). Destaca-se, no entanto, que, apesar do efeito resultante ser semelhante, as aprendizagens das GA-ELM e PSO-ELM se dão de formas distintas e, portanto, existem sutis diferenças gráficas a serem analisadas:

- a) a zona de aproximação, mesmo notoriamente estreita em ambos os modelos, é ainda mais na PSO-ELM — um efeito interessante é o surgimento de uma estreita zona para  $N_{hid} = 20$ , praticamente inexistente para valores menores, fenômeno parcialmente explicável pela diminuição da diversidade, como visto mais à frente sob análise da evolução do erro;
- b) o sobreajuste afeta mais a PSO-ELM do que a GA-ELM:
  - para  $N_{hid} = 10$ , por exemplo, a PSO-ELM (Figura 5.2b) apresenta uma aproximação com sutis desvios em direção a amostras específicas para valores da abscissa no gráfico entre  $x = 1$  e  $x = 2$ , enquanto o resultado gráfico da GA-ELM não parece ser afetado por *overfitting* ainda;
  - para  $N_{hid} = 20$ , ambas GA-ELM e PSO-ELM apresentam desvios do comportamento esperado do modelo por sobreajuste, novamente com a segunda aparentemente sendo mais afetada (valores da abscissa  $0 \leq x \leq 3$ ) do que a primeira ( $0 \leq x \leq 2$ ).

<sup>2</sup> Como discutido na alínea e da seção 3.3.3, para valores pequenos de  $N_{hid}$ , o algoritmo convencional da ELM apresenta, em geral, característico *underfitting* em decorrência da manutenção da aleatoriedade na camada oculta da rede após a fase de aprendizagem.

<sup>3</sup> Como discutido na alínea c da seção 3.3.3, destaca-se que o algoritmo puro da ELM não apresenta *overfitting* mesmo para valores elevados de  $N_{hid}$ .



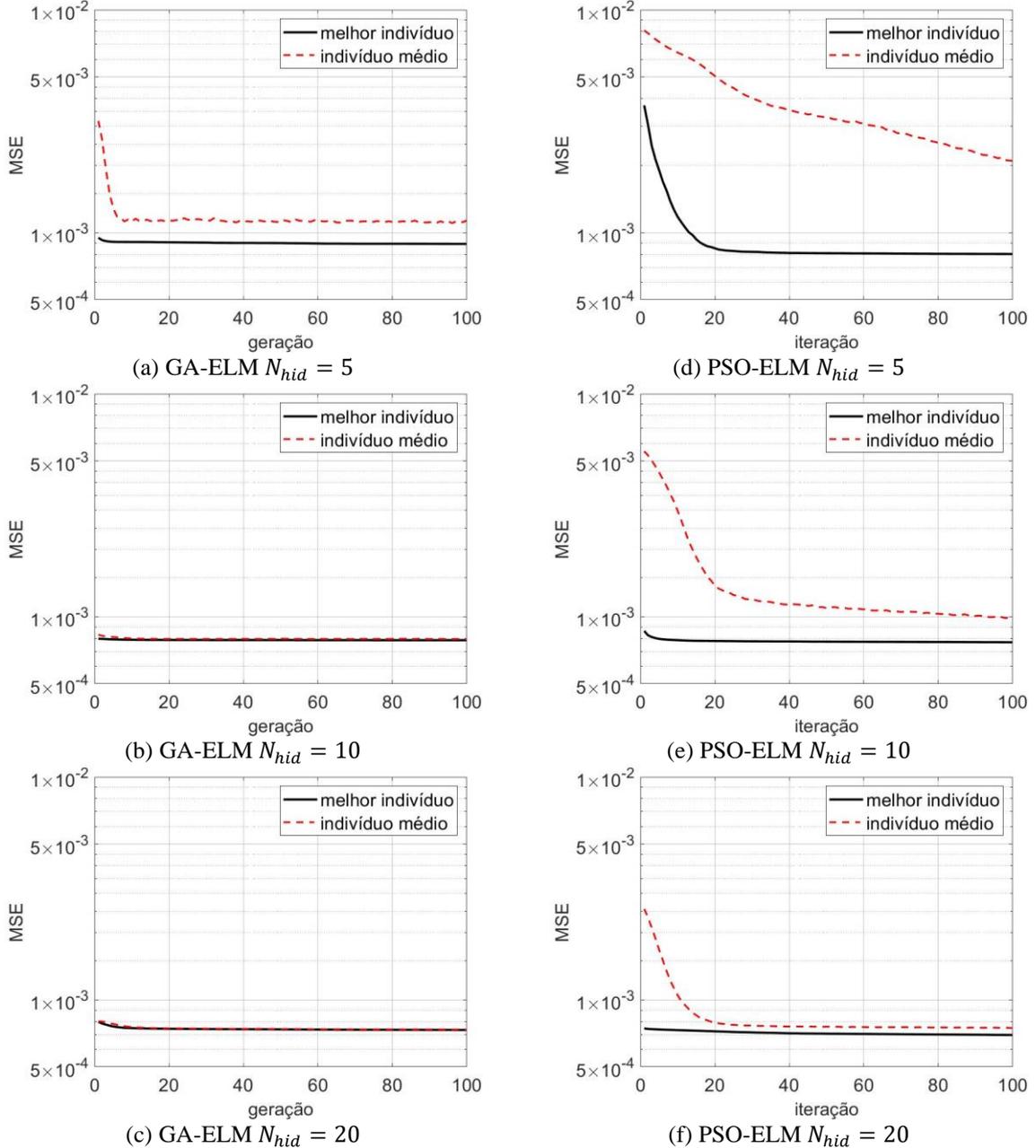
A fim de investigar as diferenças nos impactos da otimização na ELM pelo AG e pela PSO, apresenta-se, na Figura 5.3, a evolução do erro de treinamento (tanto do melhor indivíduo quanto da média da população) ao longo das gerações ou iterações dos algoritmos GA-ELM e PSO-ELM para cada configuração de  $N_{hid}$ . Destaca-se:

- o erro do melhor indivíduo se estabiliza mais rapidamente do que o da média da população — o que é, de fato, esperado: uma vez que a rede alcance a melhor solução, os demais indivíduos tendem a convergir em sua direção;
- o erro obtido para a GA-ELM é menor e tende a convergir mais rapidamente — o que não é necessariamente positivo.

Em relação à convergência do erro do melhor indivíduo, ela só é aprazível para  $N_{hid} = 5$  e, ainda assim, pouco menos de 20 gerações mais rápida — e, apesar do erro ligeiramente menor da GA-ELM com  $N_{hid} = 10$  ou 20, a essa ordem de grandeza, pequenas diferenças no MSE não são um fator tão relevante, uma vez que as redes já caminham para o *overfitting* (Figura 5.1). Por outro lado, a média da população é nitidamente mais rápida em se aproximar do melhor indivíduo na GA-ELM, o que implica em diminuição da diversidade e

possível convergência prematura — impactos que não afetam visivelmente o resultado médio da aproximação do Problema 1, porém ajudam a explicar a zona de aproximação ligeiramente maior, se comparada a da PSO-ELM (Figura 5.1).

Figura 5.3 — Evolução do erro de treinamento para o Problema 1 ( $N_{hid}$  constante).



Fonte: próprio autor.

É possível presumir uma diminuição da diversidade da população quanto mais próximos forem os gráficos do melhor indivíduo e da média populacional, porque isso indicaria que os indivíduos da população convergiram às proximidades da melhor solução. Por si só, esse fator não garante perda da diversidade, pois indivíduos diferentes podem gerar MSE semelhante; porém, é um bom indicio, considerando-se tratar de toda a população.

Agora a relação entre diversidade da população e a zona de aproximação: quando reduzida, a diversidade impacta na capacidade da rede continuar procurando indivíduos mais diferentes do que os que já encontrou, o que é importante nas últimas gerações, para que o algoritmo convirja. No entanto, caso essa redução ocorra cedo, como para a GA-ELM, é possível que dentro de 100 execuções da rede, algumas apresentem convergência prematura a um mínimo local e, por isso, a zona de aproximação tende ser maior que a da PSO-ELM, que se mantém mais diversa por algum tempo.

É cabível creditar o ligeiro aumento da zona de aproximação da PSO-ELM ( $N_{hid} = 20$ , Figura 5.2) à redução mais rápida da diversidade da população. Não se pode presumir, contudo, que esse fenômeno se amplificaria com o aumento de neurônios na camada oculta, à semelhança da intensificação do clara *overfitting* para valores elevados de  $N_{hid}$  em algoritmos iterativos.

Para o Problema 2, na Figura 5.4, comparam-se as respostas médias obtidas para ELM e GA-ELM, bem como na Figura 5.5, para PSO-ELM: a coloração no gráfico indica a intensidade do MSE em relação à resposta esperada no ponto conforme a indicação na barra de cores (escala de 0 a 0,1; valores superiores a 0,1 recebem a mesma coloração deste). Para ambos os algoritmos, adotaram-se valores de estudo para  $N_{hid} = 10, 20, 50$  e 250.

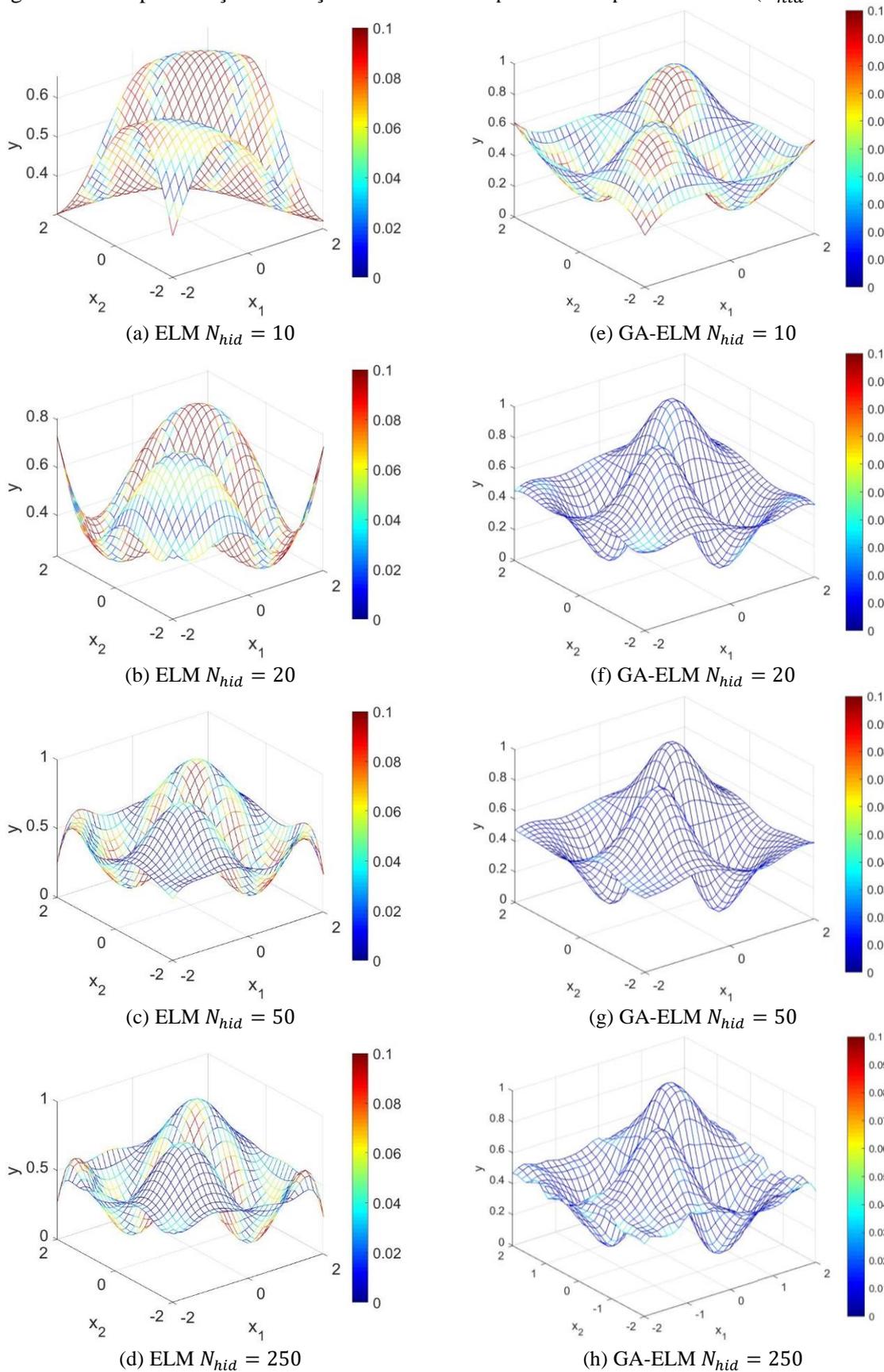
Para realizar a aproximação do Problema 2, a ELM apresenta ainda mais dificuldade na aprendizagem: com dez neurônios ocultos (Figura 5.4a), o modelo obtido é precário; com 30 e 50 (Figura 5.4b,c), razoável; e mesmo para valor bem mais elevado, como 250 (Figura 5.4d), apresenta erros significativos em algumas regiões da função. Em contrapartida, a GA-ELM com 20 nós escondidos (Figura 5.4f) já demonstra solução mais satisfatória.

Com  $N_{hid} = 50$  (Figura 5.4g), o modelo da GA-ELM é praticamente monocolor próximo a indicação de cor nula. A partir de então, o crescimento no número de neurônios proporciona tendência a *overfitting* (semelhante ao Problema 1); com 250 nós escondidos (Figura 5.4h), o modelo apresenta alguns perceptíveis erros pela mudança de cor.

A PSO-ELM apresenta os melhores resultados: já para  $N_{hid} = 10$  (Figura 5.5a), a aproximação é visivelmente próxima, progredindo ainda mais para  $N_{hid} = 20$  e 50 (Figura 5.5b, c). Com 250 neurônios ocultos (Figura 5.5d), à semelhança da GA-ELM, ocorrem sobreajustes.

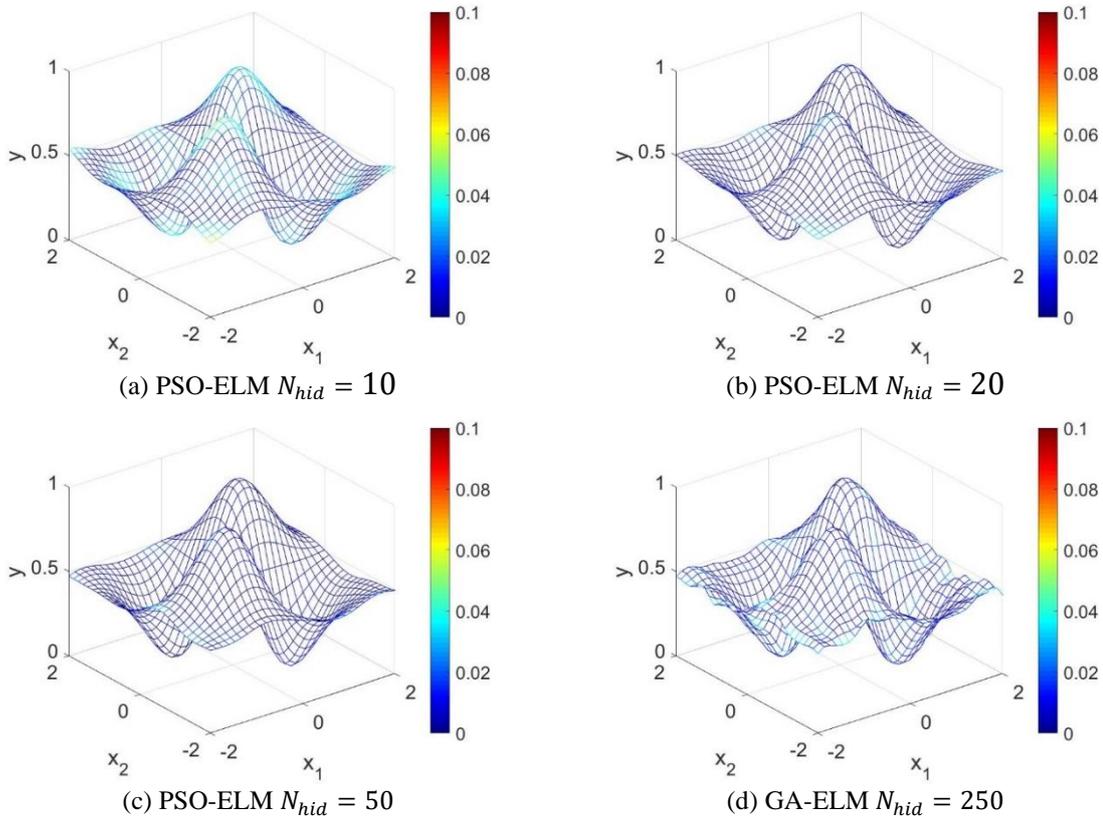
Em síntese, o impacto do AG e da PSO sobre a ELM, mantendo constante o número de neurônios ocultos, é o de potencializar o ajuste da curva: se a rede pura apresenta *underfitting* (exemplo: baixa quantidade de neurônios ocultos), o otimizador proporciona melhor ajuste; se apresenta ajuste adequado, o otimizador sobreajusta (*overfitting*). Essa atuação se altera quando  $N_{hid}$  passa a variar durante o treinamento.

Figura 5.4 — Aproximação de função do Problema 2 pela ELM e pela GA-ELM ( $N_{hid}$  constante).



Fonte: próprio autor.

Figura 5.5 — Aproximação de função do Problema 2 pela PSO-ELM ( $N_{hid}$  constante).



Fonte: próprio autor.

### 5.1.2 $N_{hid}$ variável

Conforme Seção 4.3.2, os algoritmos com variação de  $N_{hid}$  apresentam parâmetros para atualização binária, em complementação aos da codificação real da Tabela 5.3 (GA-ELM) e da Tabela 5.4 (PSO-ELM). Os parâmetros binários comuns a GA-ELM e PSO-ELM (quantidade de neurônios ocultos e expoentes  $\gamma_e$  e  $\gamma_h$  da função de aptidão) estão definidos na Tabela 5.5, enquanto os restritos a cada modelo estão na Tabela 5.6 e na Tabela 5.7, nessa ordem.

Tabela 5.5 — Parâmetros comuns a GA-ELM e PSO-ELM.

| Parâmetro                        | Símbolo         | Valores Típicos     | Valor adotado        |                       |
|----------------------------------|-----------------|---------------------|----------------------|-----------------------|
|                                  |                 |                     | Problema 1           | Problema 2            |
| Quantidade máxima de $N_{hid}$   | $N_{hid}^{max}$ | —                   | $N_{hid}^{max} = 50$ | $N_{hid}^{max} = 250$ |
| Quantidade mínima de $N_{hid}$   | $N_{hid}^{min}$ | $N_{hid}^{min} > 1$ | $N_{hid}^{min} = 5$  |                       |
| Expoente do erro na aptidão      | $\gamma_e$      | —                   | $\gamma_e = 1$       |                       |
| Expoente de $N_{hid}$ na aptidão | $\gamma_h$      | —                   | $\gamma_h = 0,25$    | $\gamma_h = 0,5$      |

Fonte: próprio autor.

Tabela 5.6 — Parâmetros de treinamento binário da ELM por AG (GA-ELM).

| Parâmetro                           | Símbolo | Valores Típicos   | Valor adotado  |
|-------------------------------------|---------|-------------------|----------------|
| Probabilidade de cruzamento binário | $p_c^b$ | $p_c^b \geq 0,7$  | $p_c^b = 0,9$  |
| Probabilidade de mutação binário    | $p_m^b$ | $p_m^b \leq 0,05$ | $p_m^b = 0,05$ |

Fonte: próprio autor.

Tabela 5.7 — Parâmetros de treinamento da ELM por PSO (PSO-ELM).

| Parâmetro                       | Símbolo                   | Valores Típicos | Valor adotado                       |
|---------------------------------|---------------------------|-----------------|-------------------------------------|
| Peso cognitivo binário          | $c_1^b$                   | Equação (4.15)  | $c_1^b = 1,35$                      |
| Peso social binário             | $c_2^b$                   | Equação (4.15)  | $c_2^b = 1,35$                      |
| Coefficiente de inércia binário | $c_0^b$                   | Equação (4.14)  | $c_{0i}^b = 0,9$ e $c_{0f}^b = 1,1$ |
| Velocidade das partículas       | $v_{max}^b$ e $v_{min}^b$ | —               | $v_{max}^b = 5$ e $v_{min}^b = -5$  |

Fonte: próprio autor.

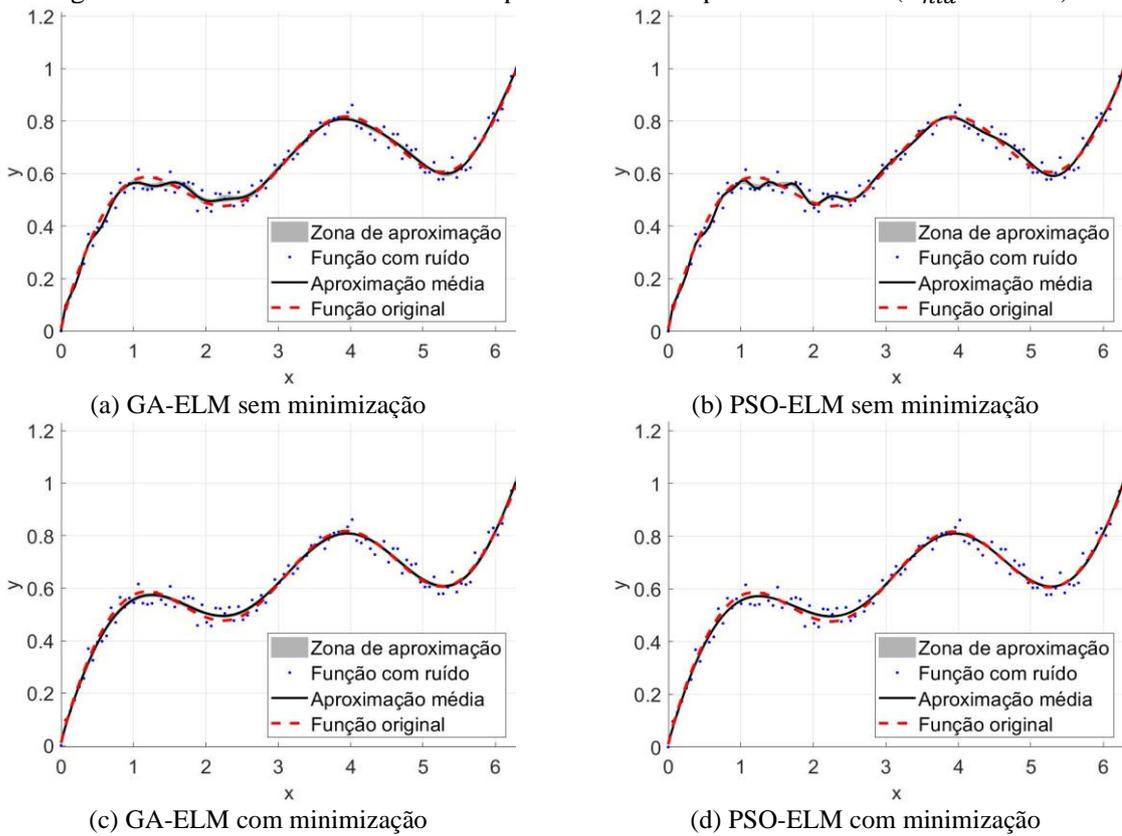
Os valores  $N_{hid}^{max}$  e  $N_{hid}^{min}$  adotados (Tabela 5.5) estão em linha com a verificação apresentada para a abordagem com  $N_{hid}$  constante, permitindo o algoritmo percorrer as mesmas quantidades de neurônios realizada manualmente e ainda valores intermediários. Os demais parâmetros estão em linha com valores típicos da literatura.

Na Figura 5.6, são apresentados o resultado médio e a zona de aproximação relativos ao Problema 1 para os modelos GA-ELM e PSO-ELM, sem e com minimização explícita de  $N_{hid}$ ; de forma semelhante, na Figura 5.7, são apresentados o resultado médio e a barra de cores em solução ao Problema 2. Os desvios em direção a amostras específicas em vez de na direção do sinal original nos algoritmos sem minimização de  $N_{hid}$  (Figura 5.6a, b e Figura 5.7a, b) decorrem de *overfitting*, não observado na Figura 5.6c, d e na Figura 5.7c, d, com minimização.

### 5.1.3 Evolução de MSE e $N_{hid}$

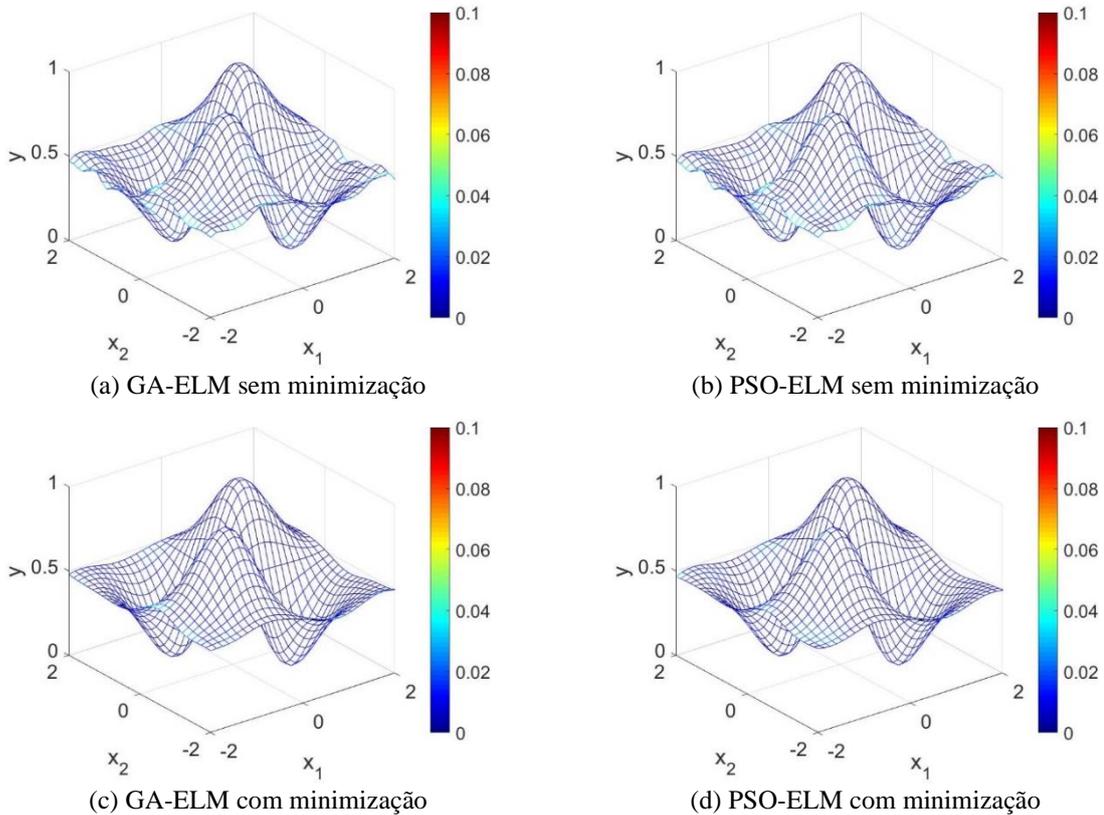
Na Figura 5.8, são apresentados a evolução média (dentro das 100 execuções) do MSE, de  $N_{hid}$  e de  $f_{apt}$  ao longo das gerações ou iterações (durante o treinamento) para o melhor indivíduo da população, bem como para o indivíduo médio, dos modelos GA-ELM e PSO-ELM sem minimização da quantidade de neurônios escondidos para o Problema 1. O MSE decresce e a função de aptidão cresce inversamente; a quantidade de neurônios aumenta até próximo do valor máximo ( $N_{hid}^{max} = 50$ ).

Figura 5.6 — Problema 1 solucionado pela GA-ELM e pela PSO-ELM ( $N_{hid}$  variável).



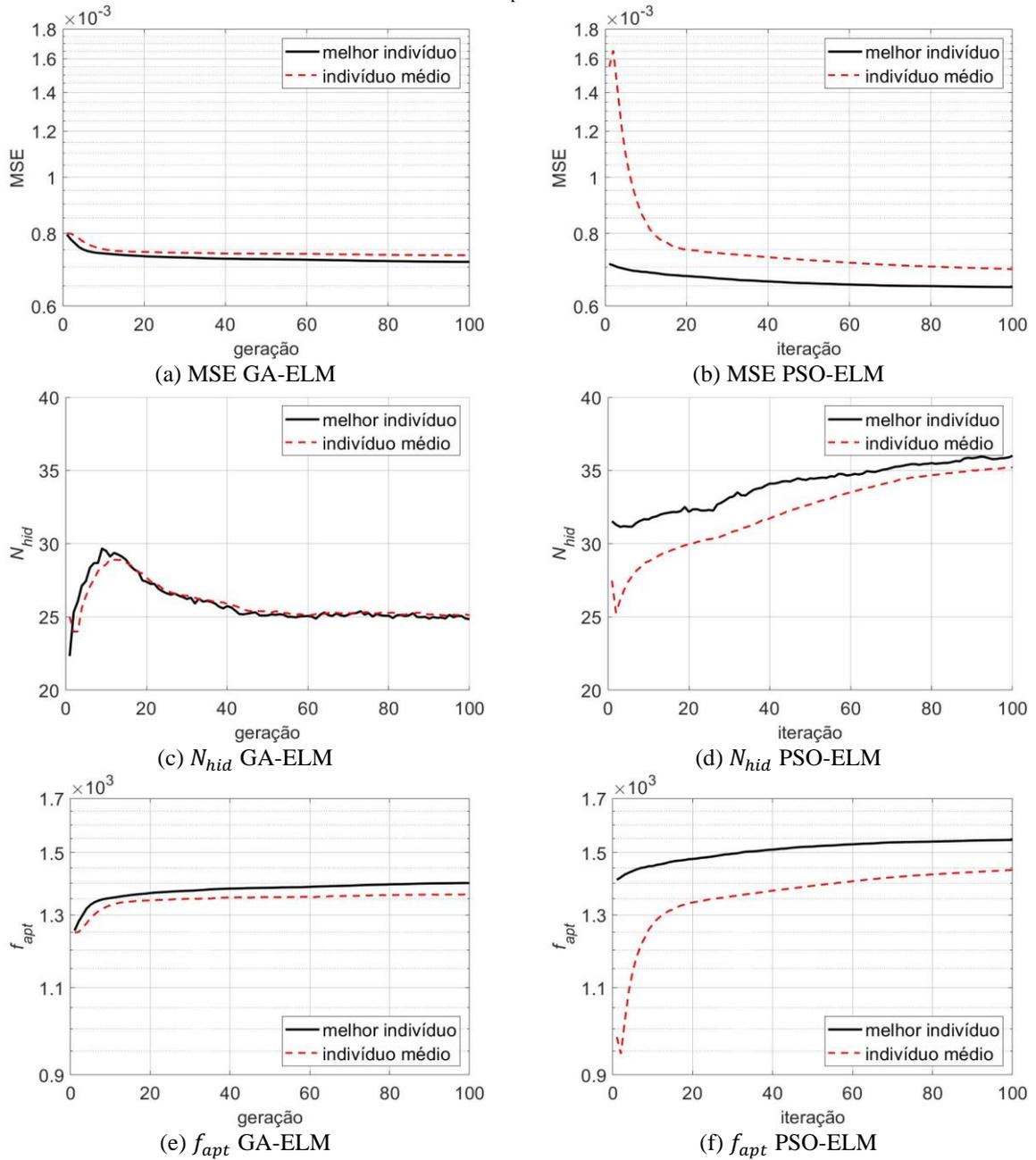
Fonte: próprio autor.

Figura 5.7 — Problema 2 solucionado pela GA-ELM e pela PSO-ELM ( $N_{hid}$  variável).



Fonte: próprio autor.

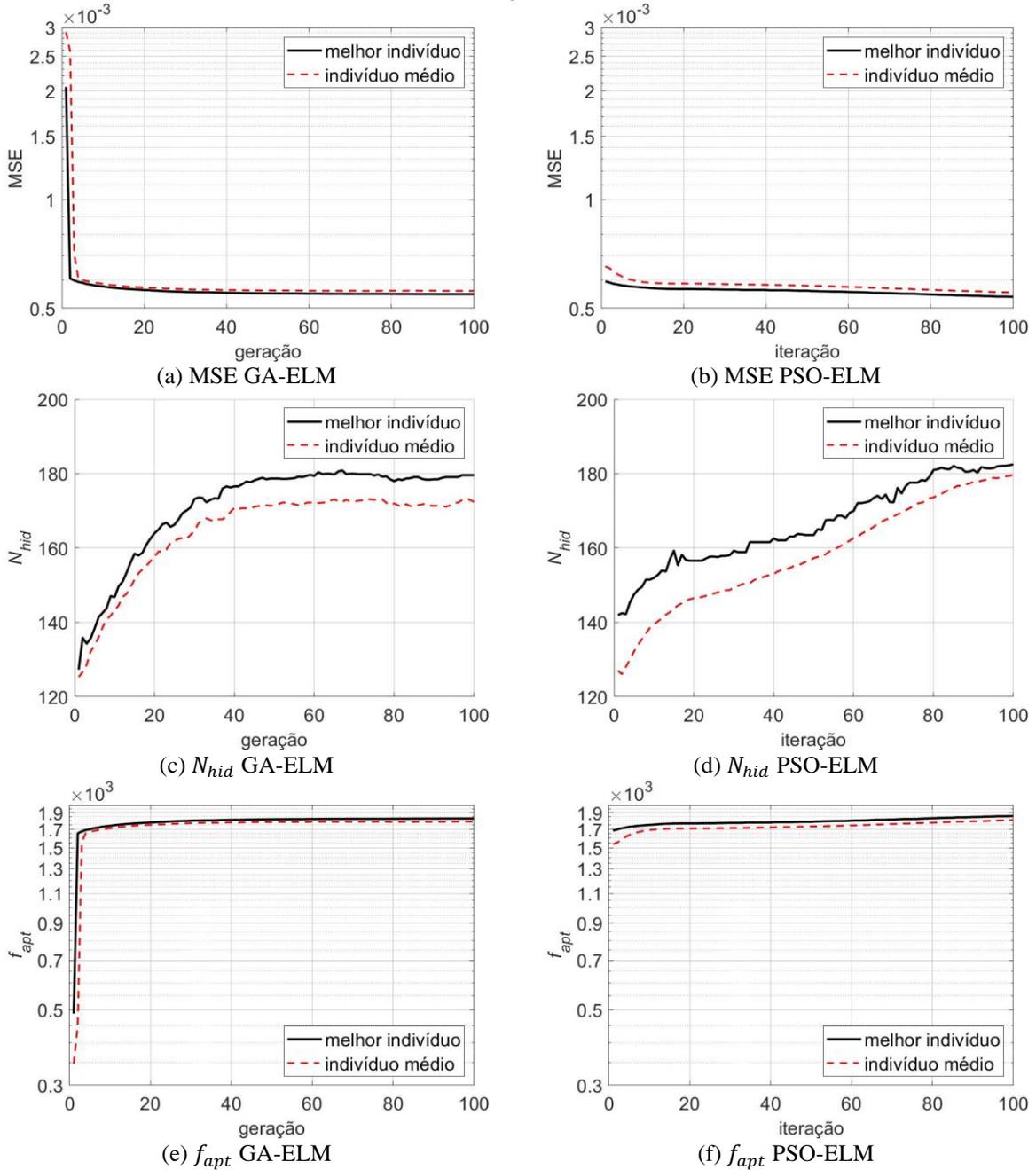
Figura 5.8 — Evolução do MSE, de  $N_{hid}$  e de  $f_{apt}$  no treinamento sem minimização (Problema 1).



Fonte: próprio autor.

A semelhança entre os sobreajustes da abordagem sem minimização e da com  $N_{hid}$  constante elevado se explica por esse aumento de  $N_{hid}$  ao longo da execução: em busca do melhor ajuste à curva, uma maior quantidade de neurônios implica em saídas mais próximas das dos dados de treinamento mesmo que absorvendo informação de erros advindos da coleta. Nesse caso, a eficiência (e a eficácia) da rede é prejudicada, porém o algoritmo entende isso positivamente pela redução do MSE. Corrobora-se com essas observações a análise para o Problema 2, com outro valor máximo de neurônios ( $N_{hid}^{max} = 250$ ), de acordo com a Figura 5.9: o número de neurônios ocultos é crescente.

Figura 5.9 — Evolução do MSE, de  $N_{hid}$  e de  $f_{apt}$  no treinamento sem minimização (Problema 2).

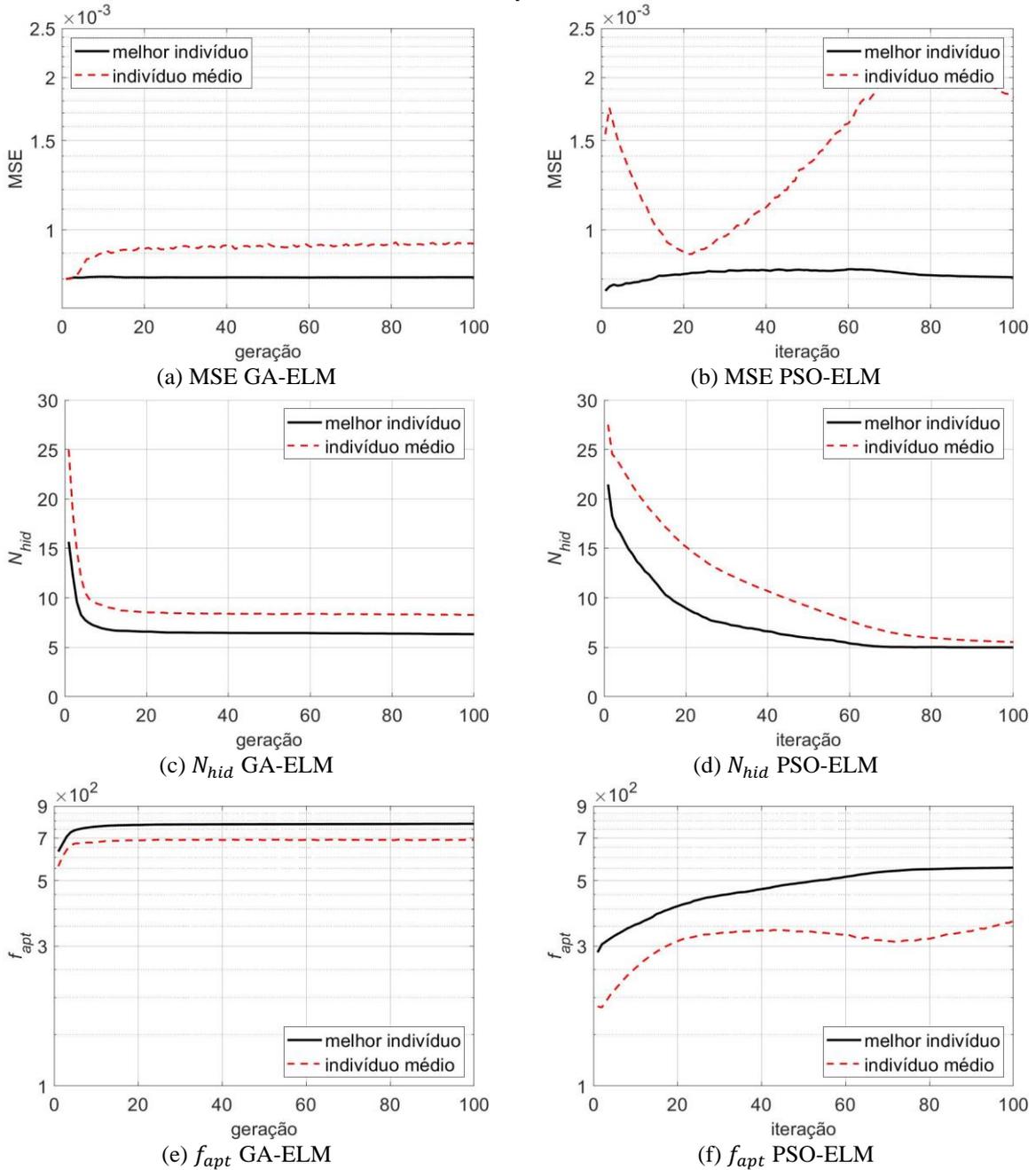


Fonte: próprio autor.

Por outro lado, na Figura 5.10 (Problema 1,  $N_{hid}^{max} = 50$ ) e na Figura 5.11 (Problema 2,  $N_{hid}^{max} = 250$ ), são apresentados os mesmos gráficos de evolução do MSE, de  $N_{hid}$  e de  $f_{apt}$  para os modelos GA-ELM e PSO-ELM com minimização da quantidade de neurônios escondidos. Destaca-se que, em ambas, a solução ótima da população é encontrada com valores  $N_{hid}$  bem inferiores ao limite máximo  $N_{hid}^{max}$ , a cargo do algoritmo localizar o valor cujo modelo otimizado apresente baixo MSE, maximizando a função de aptidão.

A minimização de  $N_{hid}$  proporciona um modelo mais versado mais eficiente, com o mínimo de neurônios necessários para se obter um ajuste satisfatório (Figura 5.6c, d e na Figura 5.7c, d) e comparável aos melhores resultados da ELM convencional e da GA-ELM e da PSO-ELM com a quantidade ideal de neurônios. A contrapartida dessa abordagem, utilizando a Equação (4.23), é que as ponderações de  $N_{hid}$ ,  $\gamma_h$ , e do erro,  $\gamma_e$ , dentro da função são arbitrárias e empíricas e deve haver uma melhor relação para outros casos específicos — e possivelmente até mesmo para essa aplicação.

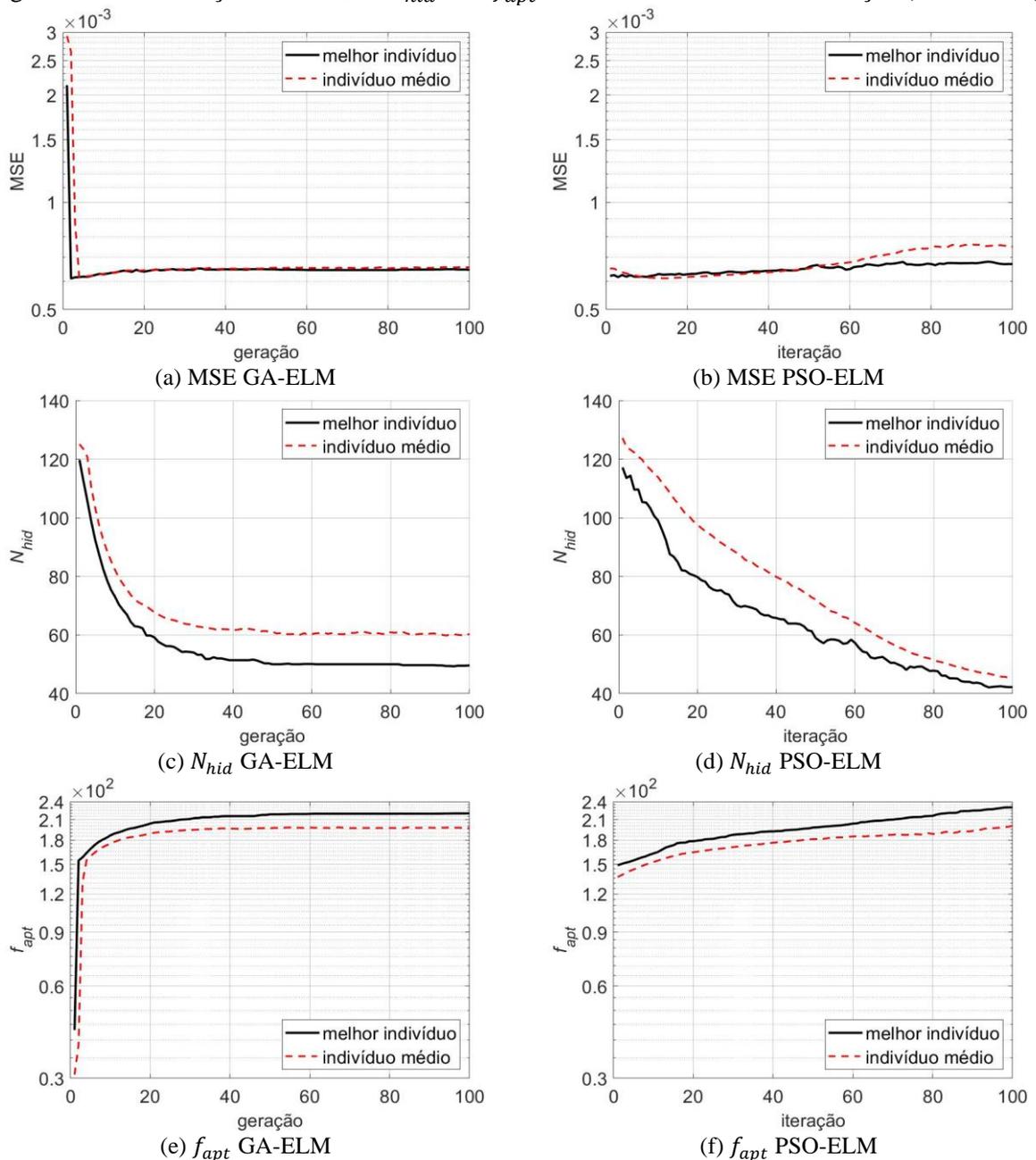
Figura 5.10 — Evolução do MSE, de  $N_{hid}$  e de  $f_{apt}$  no treinamento com minimização (Problema 1).



Fonte: próprio autor.

Apesar de contra-intuitivo, o fato do MSE do melhor indivíduo da população apresentar um leve aumento e do indivíduo médio, um leve ou um acentuado, faz todo o sentido nessa estratégia de minimização de  $N_{hid}$  logo após um ponto de mínimo nas primeiras gerações ou iterações. Isso porque, inicialmente, o algoritmo busca minimizar principalmente o erro, uma vez que esse apresenta um expoente maior  $\gamma_e$ , porém, chega-se a um ponto em que a melhora do MSE é limitada pela redução em menor escala, mas simultaneamente, de  $N_{hid}$ . Essa reação de estabilização ou leve alta do MSE do menor indivíduo é essencial não somente para rede obter maior eficiência, mas também para evitar a armadilha do *overfitting*.

Figura 5.11 — Evolução do MSE, de  $N_{hid}$  e de  $f_{apt}$  no treinamento com minimização (Problema 2).

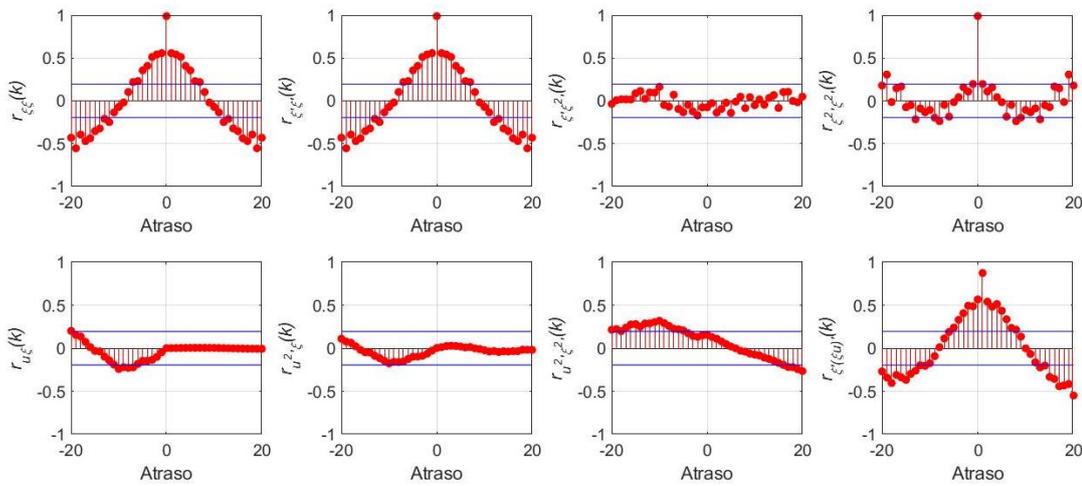


Fonte: próprio autor.

5.1.4 Validação por análise dos resíduos

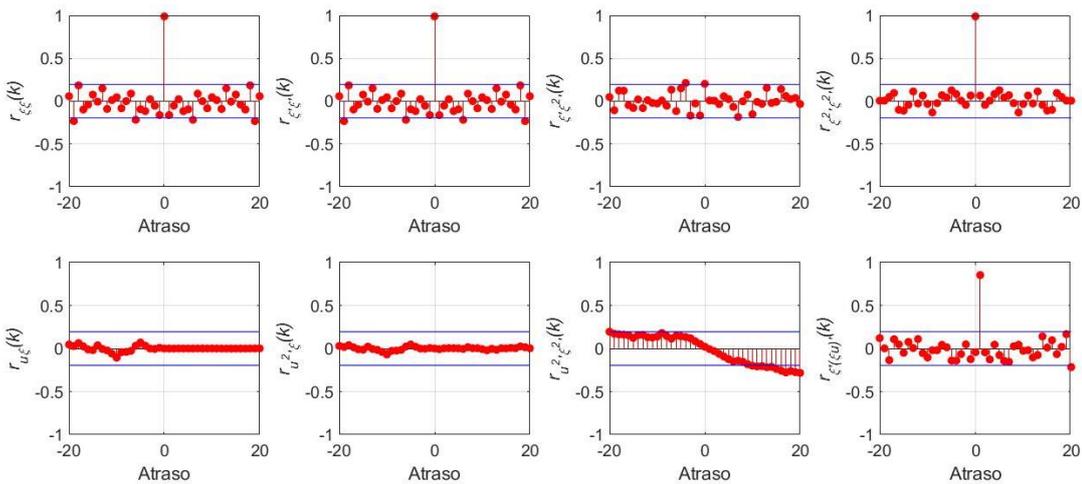
A avaliação dos modelos pela análise dos resíduos, como apresentado na Seção 3.1.5.2, especialmente nas Equações (3.5) a (3.12)<sup>4</sup>, corroboram, bem como conferem maior precisão, a qualidade da modelagem da ELM e variantes orimizadas desta seção; o intervalo de confiança adotado é de 95%<sup>5</sup>. Para o Problema 1, o modelo ELM com  $N_{hid} = 5$  falha em sua validação (Figura 5.12) para os quesitos  $r_{\xi\xi}(\tau) = \delta(\tau)$ ,  $r_{\xi'\xi'}(\tau) = \delta(\tau)$ ,  $r_{u^2\xi^2}(\tau) = 0$  e  $r_{\xi(\xi u)}(\tau) = 0$ ; em contrapartida, com um valor de  $N_{hid}$  dez vezes maior (Figura 5.13), o modelo é satisfatório.

Figura 5.12 — Validação do modelo ELM para o Problema 1 ( $N_{hid} = 5$ ).



Fonte: próprio autor.

Figura 5.13 — Validação do modelo ELM para o Problema 1 ( $N_{hid} = 20$ ).



Fonte: próprio autor.

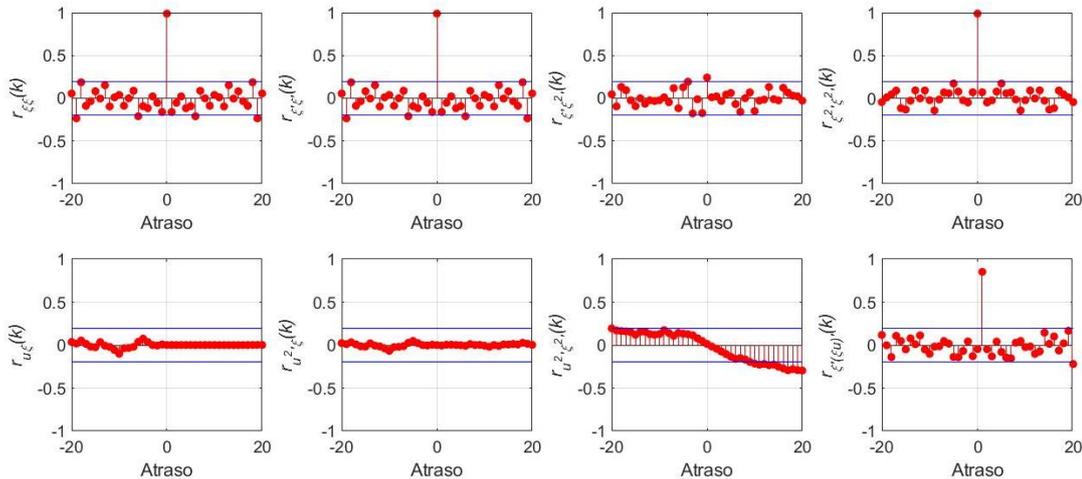
<sup>4</sup> A validação por análise de resíduos é bem sucedida quando:  $r_{\xi\xi}(\tau) = \delta(\tau)$ ,  $r_{\xi'\xi'}(\tau) = \delta(\tau)$ ,  $r_{\xi'\xi^2}(\tau) = 0$ ,  $r_{\xi^2\xi^2}(\tau) = \delta(\tau)$ ,  $r_{\xi u}(\tau) = 0$ ,  $r_{\xi(\xi u)}(\tau) = 0$ ,  $r_{u^2\xi}(\tau) = 0$ ,  $r_{u^2\xi^2}(\tau) = 0$ .

<sup>5</sup> Como já mencionado na seção 3.1.5.2, os limites do intervalo de confiança de 95% são definidos em  $\pm 1,96/\sqrt{N}$ .

O aumento de neurônios na ELM sempre é positivo para melhor precisão na modelagem (maior eficácia), mesmo que o tempo de processamento aumente (menor eficiência). No entanto, os modelos otimizados da ELM conseguem os dois atributos: assim como a ELM com  $N_{hid} = 50$ , a GA-ELM (Figura 5.14) e a PSO-ELM (Figura 5.15) com  $N_{hid}$  minimizado (resultados entre 5 e 10) atendem aos oito critérios de validação propostos, observando falha sem expressividade notória para  $r_{u^2\xi^2}(\tau) = 0$  próximo a  $\tau = 20$ , também observável na Figura 5.13.

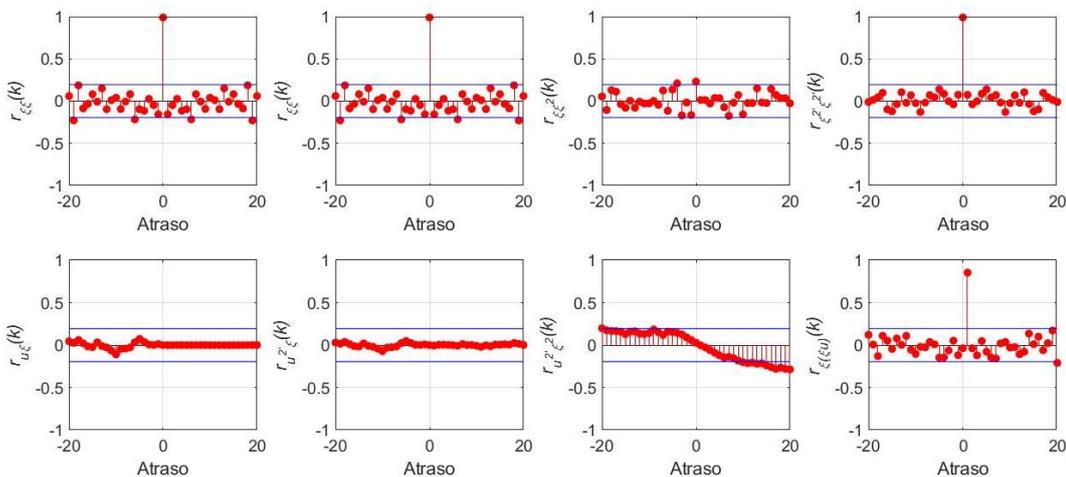
Em ordem de evitar repetidos gráficos de alta similaridade, omitem-se os gráficos de validação para os algoritmos da GA-ELM e da PSO-ELM com  $N_{hid}$  fixo e variável sem minimização explícita. Nestas configurações, os resultados da validação foram semelhantes.

Figura 5.14 — Validação do modelo GA-ELM para o Problema 1 ( $N_{hid}$  minimizado).



Fonte: próprio autor.

Figura 5.15 — Validação do modelo PSO-ELM para o Problema 1 ( $N_{hid}$  minimizado).

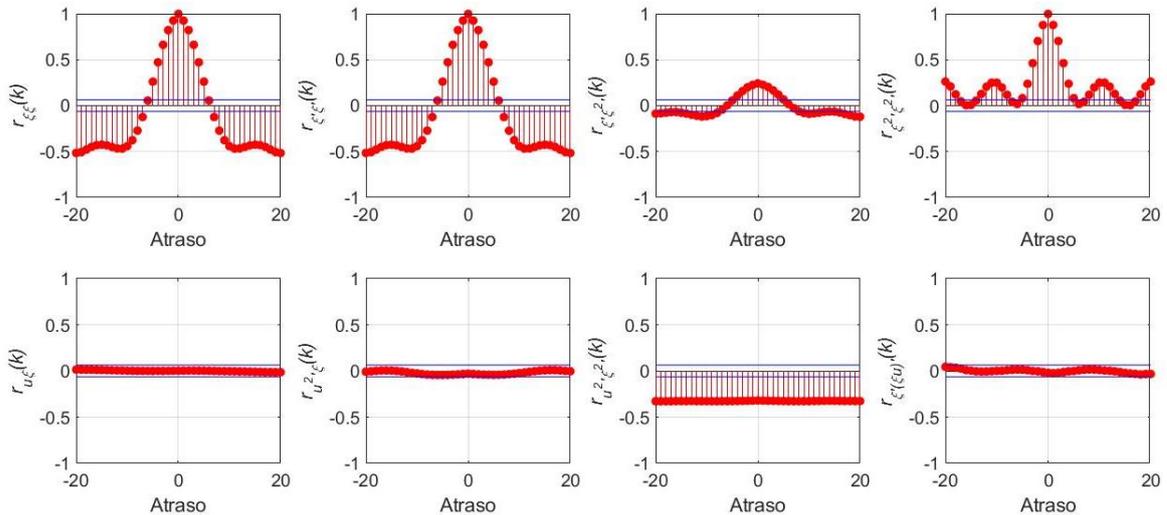


Fonte: próprio autor.

Para o Problema 2, o modelo ELM com  $N_{hid} = 10$  falha para validação (Figura 5.16) dos quesitos  $r_{\xi\xi}(\tau) = \delta(\tau)$ ,  $r_{\xi'\xi'}(\tau) = \delta(\tau)$  e  $r_{u^2\xi^2}(\tau) = 0$ , além de  $r_{\xi'\xi^2}(\tau) = 0$  e  $r_{\xi^2\xi^2}(\tau) = \delta(\tau)$ . O aumento de neurônios ( $N_{hid} = 250$ , Figura 5.17) novamente melhora os dados de validação, no entanto, insuficientemente; as falhas de validação persistem em quatro critérios:  $r_{\xi\xi}(\tau) = \delta(\tau)$ ,  $r_{\xi'\xi'}(\tau) = \delta(\tau)$ ,  $r_{\xi^2\xi^2}(\tau) = \delta(\tau)$  e  $r_{u^2\xi^2}(\tau) = 0$ .

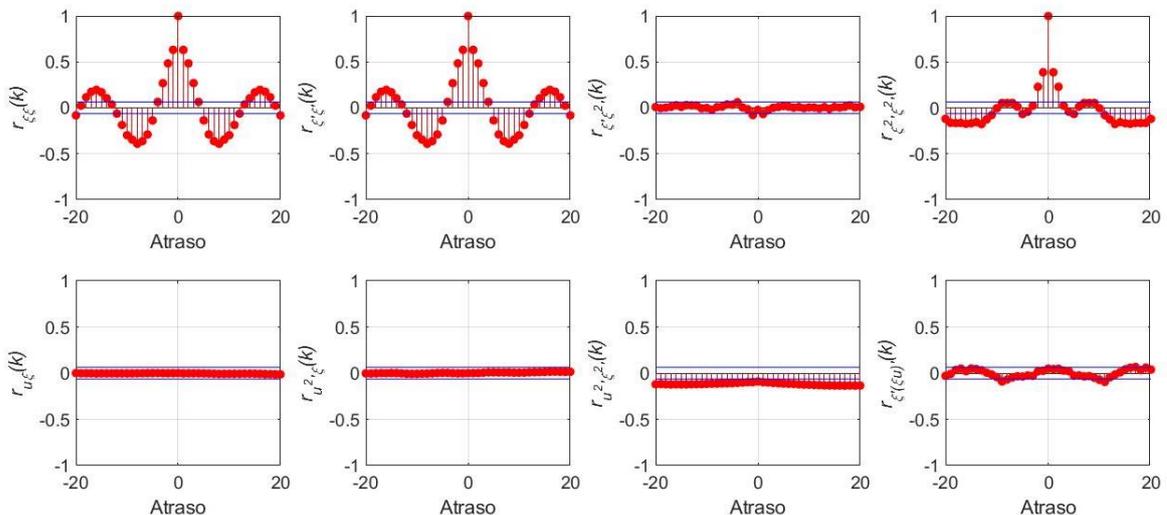
Por outro lado, os modelos GA-ELM e PSO-ELM com  $N_{hid}$  minimizado para o Problema 2 atendem bem aos oito critérios (Figura 5.18 e Figura 5.19, respectivamente), ratificando a eficácia e eficiência no paradigma de redução de neurônios ocultos mesmo para processos em que a ELM sem otimização não consegue uma identificação satisfatória.

Figura 5.16 — Validação do modelo ELM para o Problema 2 ( $N_{hid} = 10$ ).



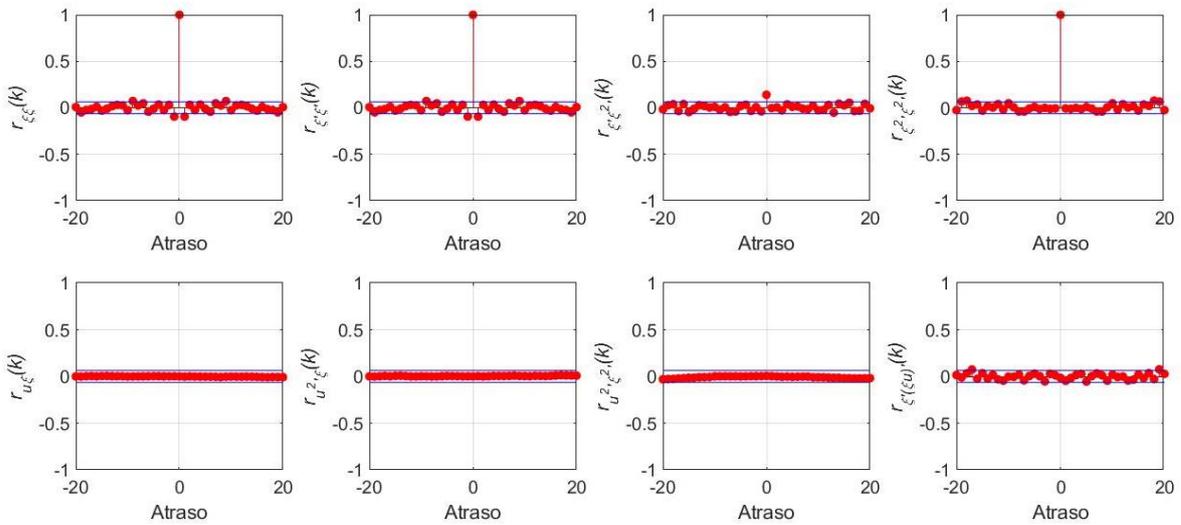
Fonte: próprio autor.

Figura 5.17 — Validação do modelo ELM para o Problema 2 ( $N_{hid} = 250$ ).



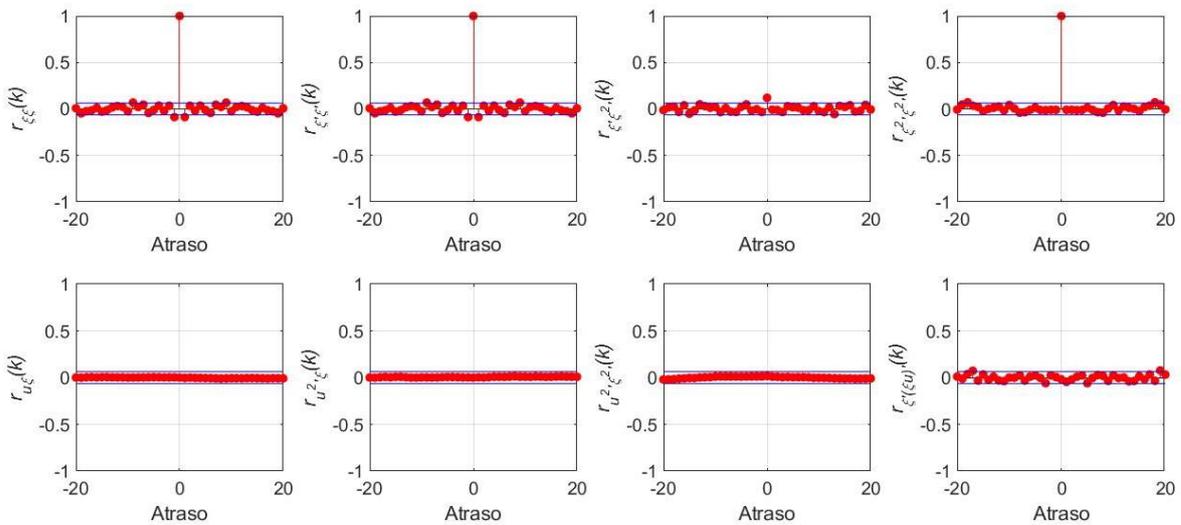
Fonte: próprio autor.

Figura 5.18 — Validação do modelo GA-ELM para função de duas variáveis ( $N_{hid}$  minimizado).



Fonte: próprio autor.

Figura 5.19 — Validação do modelo GA-ELM para função de duas variáveis ( $N_{hid}$  minimizado).



Fonte: próprio autor.

Por fim, na Tabela 5.8, é apresentado um breve comparativo entre a ELM e as suas variantes otimizadas, agrupando-se GA-ELM e PSO-ELM pelo tratamento a  $N_{hid}$  (constante, não minimizado ou minimizado)<sup>6</sup>; a informação a respeito do tempo de processamento é mais objetivamente discutida na Seção 5.3. Destaca-se que o modelo ELM obtido não é capaz de aproximar a função de duas variáveis (Problema 2). Por outro lado, suas adaptações obtiveram melhores resultados, com destaque para a GA-ELM e a PSO-ELM com  $N_{hid}$  minimizado.

<sup>6</sup> Na Tabela 5.8, o comparativo não se atém às diferenças entre os algoritmos por conta do mecanismo otimizador, mas tão somente pela abordagem da otimização, uma vez que tanto AG e PSO desempenham impactos similares na rede neural para a discussão dessa pesquisa.

Tabela 5.8 — Comparativo entre ELM e variantes da GA-ELM.

| <b>Características do Algoritmo</b>    |                                |                                       |                                |  |
|--|--------------------------------|---------------------------------------|--------------------------------|--|
| <b>Algoritmo</b>                       | ELM                            | GA-ELM /<br>PSO-ELM                   | GA-ELM /<br>PSO-ELM            | GA-ELM /<br>PSO-ELM                      |
| $N_{hid}$                              | Constante                      | Constante                             | Variável                       | minimizado                               |
| <b>Resultados Obtidos</b>              |                                |                                       |                                |  |
| <b>Treinamento<br/>Camada de Saída</b> | EMQ                            | EMQ                                   | EMQ                            | EMQ                                      |
| <b>Treinamento<br/>Camada Oculta</b>   | Sem<br>treinamento             | AG/PSO                                | AG/PSO                         | AG/PSO                                   |
| <b>Função de aptidão</b>               | —                              | $f_{apt} = \frac{1}{erro}$            | $f_{apt} = \frac{1}{erro}$     | $f_{apt} = \frac{1}{N_{hid} \cdot erro}$ |
| <b>Resultado Médio</b>                 | Razoável a<br>satisfatório     | Satisfatório<br>( $N_{hid}$ não alto) | Razoável a<br>satisfatório     | Satisfatório                             |
| <b>Zona de<br/>Aproximação</b>         | Espaçado<br>( $N_{hid}$ baixo) | Reduzido                              | Reduzido                       | Reduzido                                 |
| <b>Tempo de<br/>processamento</b>      | Baixíssimo                     | Médio                                 | Médio                          | Médio                                    |
| <b>Overfitting</b>                     | NÃO                            | SIM<br>( $N_{hid}$ alto)              | SIM<br>( $N_{hid}^{máx}$ alto) | NÃO                                      |
| <b><math>N_{hid}</math> final</b>      | Constante                      | Constante                             | Alto                           | Baixo                                    |

Fonte: próprio autor.

## 5.2 MLP: CAMADAS OCULTA E DE SAÍDA TREINADAS POR AG E PSO

Assim como para a GA-ELM e a PSO-ELM, as três distintas abordagens ( $N_{hid}$  constante,  $N_{hid}$  não minimizado e  $N_{hid}$  minimizado) são de interesse de estudo para o GA-MLP e PSO-MLP. No entanto, pela concisão do trabalho e por, em muitos aspectos, a reprodução de conceitos ser idêntica aos das versões da ELM, são omitidos os dois primeiros paradigmas.

Na Tabela 5.9, são apresentados os parâmetros de treinamento do MLP (quantidade de épocas e taxa de aprendizagem), enquanto nas Tabela 5.10 e Tabela 5.11, os das redes GA-MLP e PSO-MLP, respectivamente, tanto reais quanto binários. Vale destacar que a quantidade de épocas utilizadas para o Problema 2 é inferior ao Problema 1; isso decorre do treinamento local adotado: a quantidade de atualizações para o Problema 1 é de  $50000 \cdot 101 = 5,05 \cdot 10^6$ , enquanto que para o Problema 2,  $10000 \cdot 961 = 9,61 \cdot 10^6$  e, portanto, a deste ainda é maior<sup>7</sup>.

<sup>7</sup> No MLP com treinamento local, a quantidade de atualizações é igual ao produto de épocas por amostras. 101 e 961 são, respectivamente a quantidade de amostras dos Problemas 1 e 2.

Percebe-se que a quantidade de gerações (Tabela 5.10) e de iterações (Tabela 5.11) são notoriamente maiores do que nas versões de otimização da ELM, por conta da segunda camada a ser treinada (o tamanho da população também é maior pelo mesmo motivo); os demais parâmetros do PSO-MLP se mantêm em linha com as versões equivalentes. A quantidade de neurônios ocultos e os expoentes  $\gamma_e$  e  $\gamma_h$  da função de aptidão estão definidos na Tabela 5.5.

Vale destacar que os algoritmos otimizados do MLP, assim como os da ELM, consideram todas as amostras simultaneamente para o treinamento, diferentemente do MLP por retropropagação do erro com treinamento local. Esse paradigma para o GA-MLP e o PSO-MLP se assemelha a uma espécie de treino por lote (a Seção 3.2.4 mostra detalhes entre a diferença entre os paradigmas) e influencia nos resultados desses algoritmos, sobretudo, no GA-MLP.

Tabela 5.9 — Parâmetros do treinamento por *backpropagation*.

| Parâmetro            | Símbolo | Valores Típicos   | Valor adotado                    |             |
|----------------------|---------|-------------------|----------------------------------|-------------|
|                      |         |                   | Problema 1                       | Problema 2  |
| Quantidade de épocas | $N_e$   | $E \geq 1000$     | $E = 50000$                      | $E = 10000$ |
| Taxa de aprendizagem | $\eta$  | $0 < \eta \leq 1$ | $\eta_0 = 0,5$ e $\eta_f = 0,05$ |             |
| Treinamento          | —       | local ou por lote | local                            |             |

Fonte: próprio autor.

Tabela 5.10 — Parâmetros de treinamento da MLP por AG (GA-MLP).

| Parâmetro                           | Símbolo   | Valores Típicos    | Valor adotado   |                |
|-------------------------------------|-----------|--------------------|-----------------|----------------|
|                                     |           |                    | Problema 1      | Problema 2     |
| Quantidade de gerações              | $N_g$     | $N_g \geq 100$     | $N_g = 10^4$    | $N_g = 10^4$   |
| Tamanho da população                | $N_{pop}$ | $N_{pop} \geq 100$ | $N_{pop} = 400$ |                |
| Probabilidade de cruzamento real    | $p_c^r$   | $p_c^r \geq 0,7$   | $p_c^r = 0,9$   | $p_c^r = 0,9$  |
| Probabilidade de mutação real       | $p_m^r$   | $p_m^r \leq 0,05$  | $p_m^r = 0,05$  | $p_m^r = 0,05$ |
| Probabilidade de cruzamento binário | $p_c^b$   | $p_c^b \geq 0,7$   | $p_c^b = 0,9$   | $p_c^b = 0,9$  |
| Probabilidade de mutação binária    | $p_m^b$   | $p_m^b \leq 0,05$  | $p_m^b = 0,05$  | $p_m^b = 0,05$ |
| Operador BLX-alfa                   | $\alpha$  | $\alpha \leq 1$    | $\alpha = 0,7$  | $\alpha = 0,7$ |
| Número de indivíduos no torneio     | $N_{tor}$ | potências de 2     | $N_{tor} = 2$   | $N_{tor} = 2$  |

Fonte: próprio autor.

Tabela 5.11 — Parâmetros de treinamento da MLP por PSO (PSO-MLP).

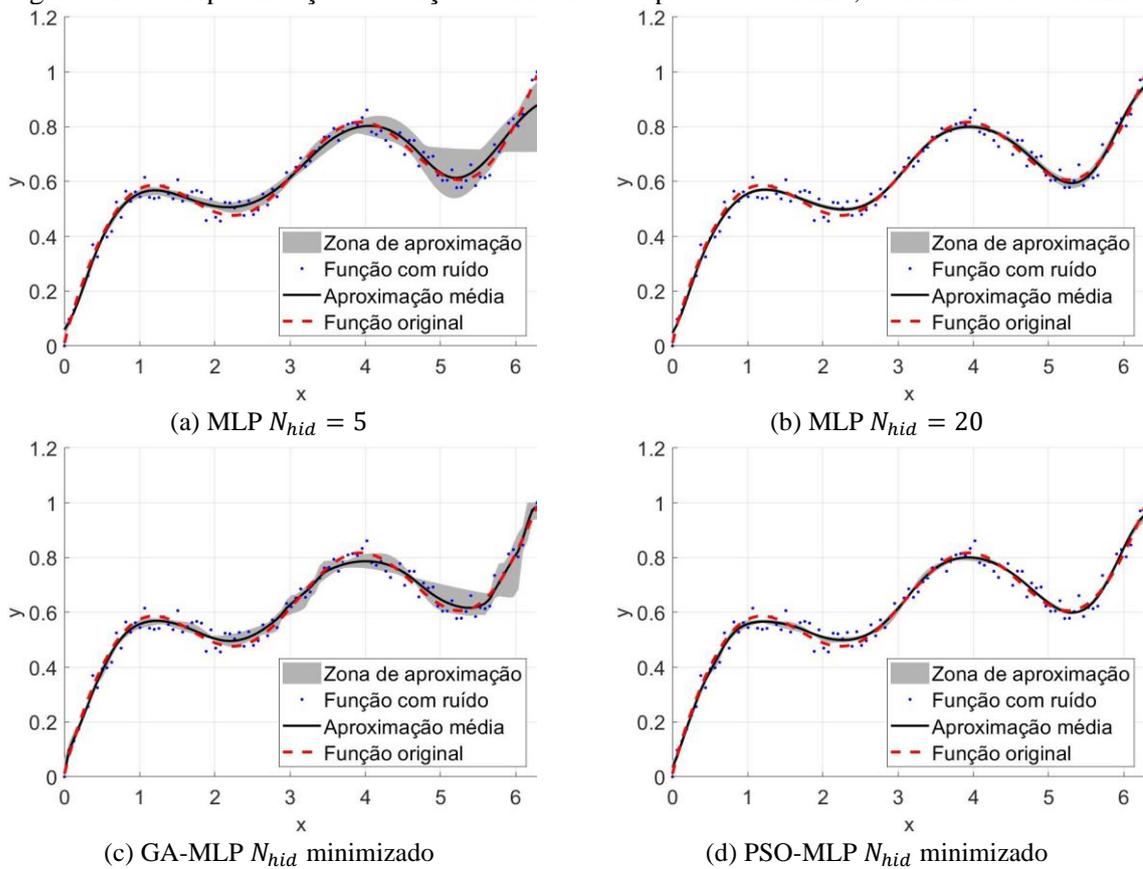
| Parâmetro                         | Símbolo                   | Valores Típicos    | Valor adotado                          |                  |
|-----------------------------------|---------------------------|--------------------|--|------------------|
|                                   |                           |                    | Problema 1                             | Problema 2       |
| Quantidade de iterações           | $N_{it}$                  | $N_{it} \geq 100$  | $N_{it} = 10000$                       | $N_{it} = 10000$ |
| Tamanho da população              | $N_{pop}$                 | $N_{pop} \geq 100$ | $N_{pop} = 400$                        |                  |
| Pesos cognitivo e social reais    | $c_1^r$ e $c_2^r$         | Equação (4.15)     | $c_1^r = c_2^r = 1,35$                 |                  |
| Coefficiente de inércia real      | $c_{0i}^r$ e $c_{0f}^r$   | Equação (4.14)     | $c_{0i}^r = 0,7$ e $c_{0f}^r = 1,1$    |                  |
| Pesos cognitivo e social binários | $c_1^b$ e $c_2^b$         | Equação (4.15)     | $c_1^b = c_2^b = 1,35$                 |                  |
| Coefficiente de inércia binário   | $c_{0i}^b$ e $c_{0f}^b$   | Equação (4.14)     | $c_{0i}^b = 0,9$ e $c_{0f}^b = 1,1$    |                  |
| Posição das partículas            | $z_{max}^r$ e $z_{min}^r$ | —                  | $z_{max}^r = 10$ e $z_{min}^r = -10$   |                  |
| Velocidade real das partículas    | $v_{max}^r$ e $v_{min}^r$ | —                  | $v_{max}^r = 0,5$ e $v_{min}^r = -0,5$ |                  |
| Velocidade binária das partículas | $v_{max}^b$ e $v_{min}^b$ | —                  | $v_{max}^b = 5$ e $v_{min}^b = -5$     |                  |

Fonte: próprio autor.

Com uma configuração em linha com a da GA-ELM, o GA-MLP não consegue alcançar uma aproximação satisfatória, então, uma estratégia de elitismo mais robusta é incorporada ao algoritmo. Em vez da preservação do melhor indivíduo (como para a GA-ELM), reúnem-se todos os indivíduos da população anterior e os filhos gerados pelos operadores em um mesmo conjunto de maneira tal que os  $N_{pop}$  melhores cromossomos desse conjunto passam a integrar a nova população. Com isso, a aprendizagem pode se tornar rápida e direcionada em excesso, por isso, alguns parâmetros de treinamento precisam ser configurados a manter a diversidade populacional: maior  $\alpha$  (maior janela de atualização do cromossomo durante o cruzamento) e menor  $N_{tor}$  (menor pressão em direção ao melhor indivíduo da população) (Tabela 5.10).

Na Figura 5.20, são apresentados o resultado médio e a zona de aproximação para a aproximação do Problema 1 pelas redes MLP, GA-MLP e PSO-MLP com minimização explícita de  $N_{hid}$ ; de forma semelhante, na Figura 5.21, são apresentados o resultado médio e a barra de cores em solução ao Problema 2. Com um menor valor de  $N_{hid}$  (Figura 5.20a), apesar de um bom resultado médio, a aproximação do MLP apresenta desvios de maior proporção a partir da abscissa  $x = 4$ ; não obstante, para  $N_{hid} = 20$  (Figura 5.20b), a zona de aproximação é notoriamente estreita, além do bom resultado médio.

Figura 5.20 — Aproximação de função do Problema 1 pelas redes MLP, GA-MLP e PSO-MLP.



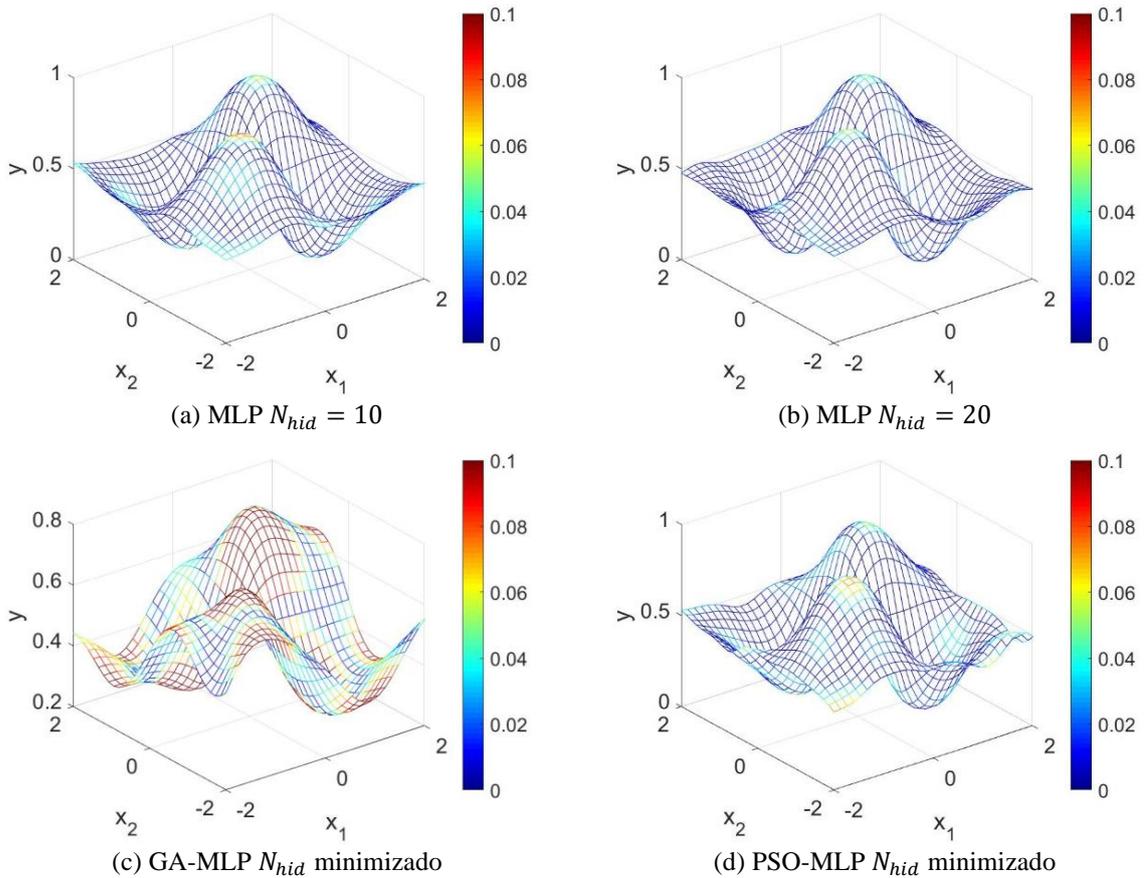
O GA-MLP com  $N_{hid}$  minimizado (Figura 5.20c) apresenta dificuldade no aprendizado dos dados a partir de valores da abscissa de  $x = 3$  em diante, com resultado médio satisfatório, porém, com desvios perceptíveis entre as execuções (zona pouco estreita), enquanto o PSO-MLP é satisfatório em ambos os quesitos, com zona de aproximação notoriamente reduzida, para toda a faixa de dados. Não obstante, para o Problema 2 (Figura 5.21), os resultados médios do MLP com  $N_{hid} = 20$  e do do PSO-MLP são semelhantes, enquanto o GA-MLP apresenta aproximação precária.

No geral, para o Problema 1 (Figura 5.22), o MSE é minimizado até um patamar próximo a ordem de  $10^{-2}$  e depois rompe em direção a um patamar inferior: para  $N_{hid} = 5$ , esse rompimento é lento e não chega a valores muito menores; para  $N_{hid} = 20$ , ocorre rapidamente em direção a  $10^{-3}$ , região dos melhores resultados com a ELM e suas variantes otimizadas. Para o Problema 2 (Figura 5.24), não ocorre a formação de dois patamares, porém, o MLP com mais neurônios converge a um MSE bem mais próximo de  $10^{-3}$ .

Nas Figura 5.23 e Figura 5.25, são apresentados o MSE, o número de neurônios escondidos e a função de aptidão durante o treinamento do GA-MLP e PSO-MLP para os Problemas 1 e 2, respectivamente. Em ambos os casos, o GA-MLP converge a um patamar de

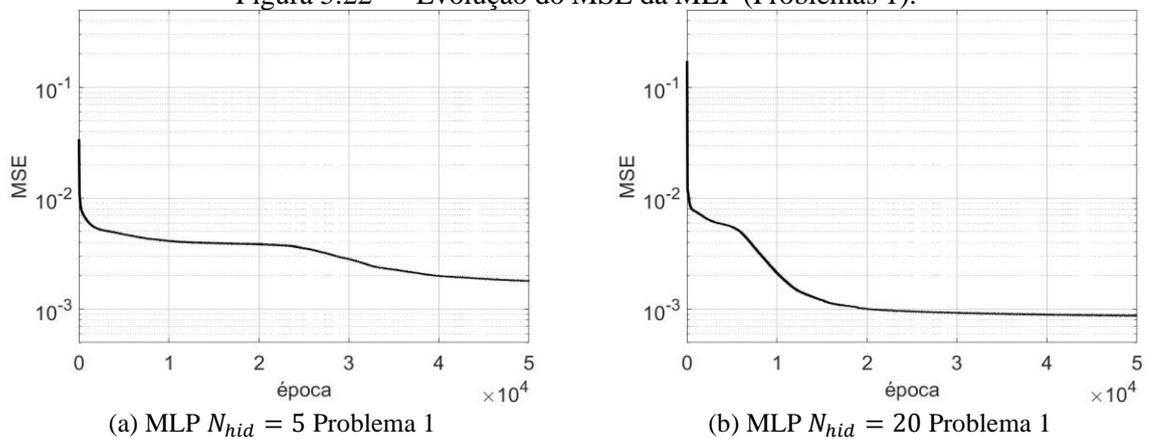
erro próximo a  $10^{-2}$ ; uma vez acomodado e mantido precocemente nesse valor, o algoritmo busca maximizar a função de aptidão pela redução de  $N_{hid}$  até seu valor mínimo  $N_{hid}^{min} = 5$ . Não obstante, o algoritmo do PSO-MLP mantém a redução do MSE depois de se manter algumas milhares de iterações aproximadamente constante; dessa forma,  $N_{hid}$  é reduzida paulatinamente.

Figura 5.21 — Aproximação de função do Problema 2 pelas redes MLP, GA-MLP e PSO-ELM.



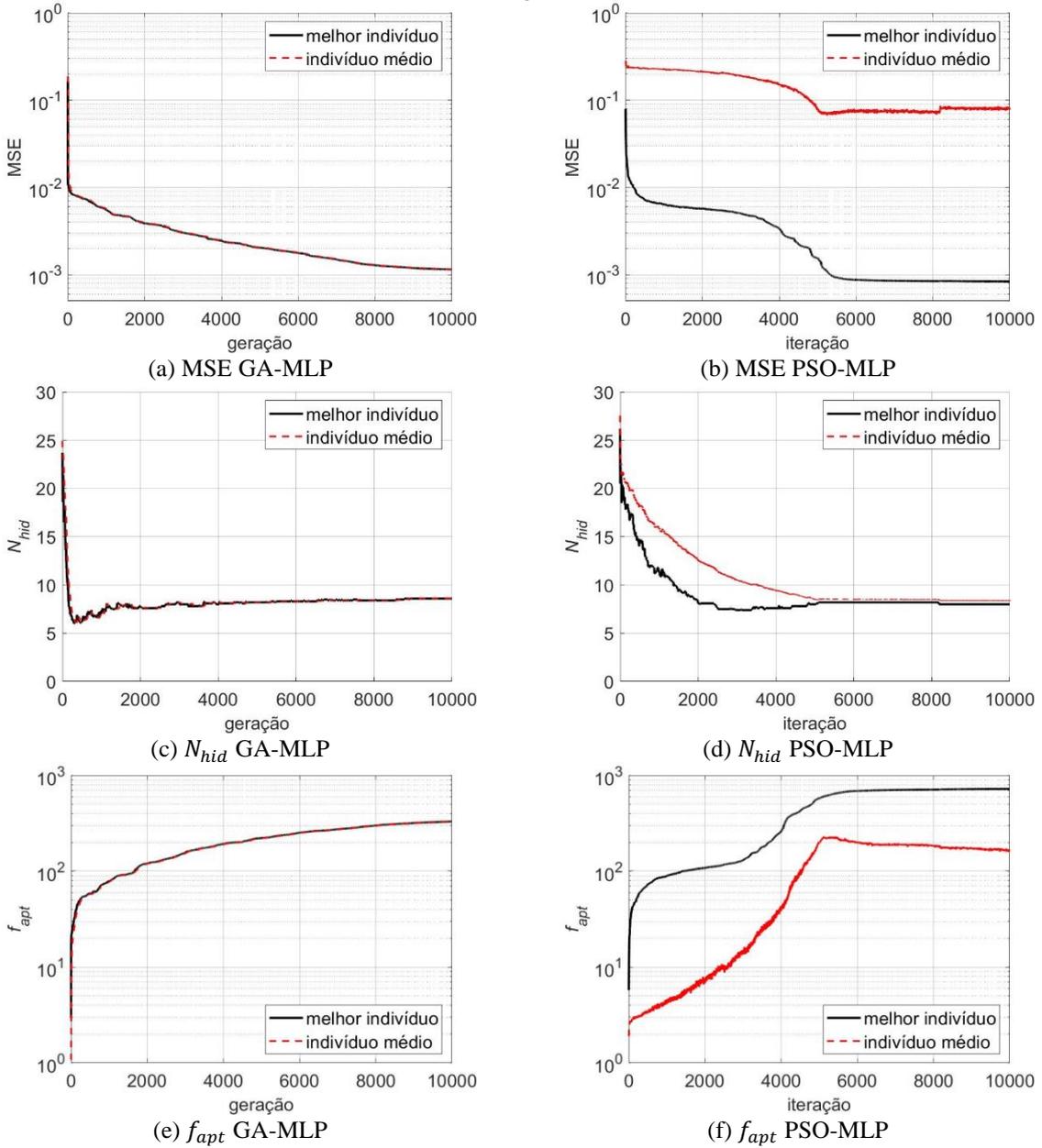
Fonte: próprio autor.

Figura 5.22 — Evolução do MSE da MLP (Problemas 1).



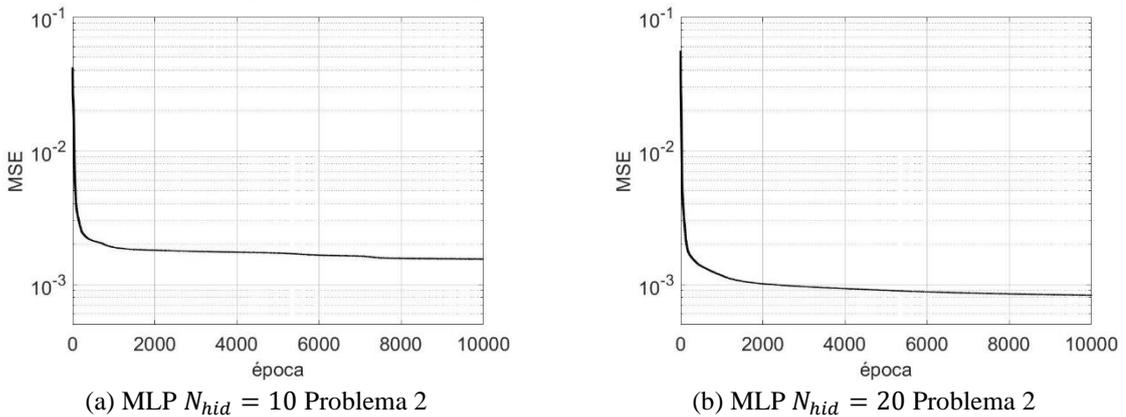
Fonte: próprio autor.

Figura 5.23 — Evolução do MSE, de  $N_{hid}$  e de  $f_{apt}$  nas redes GA-MLP e PSO-MLP (Problema 1).



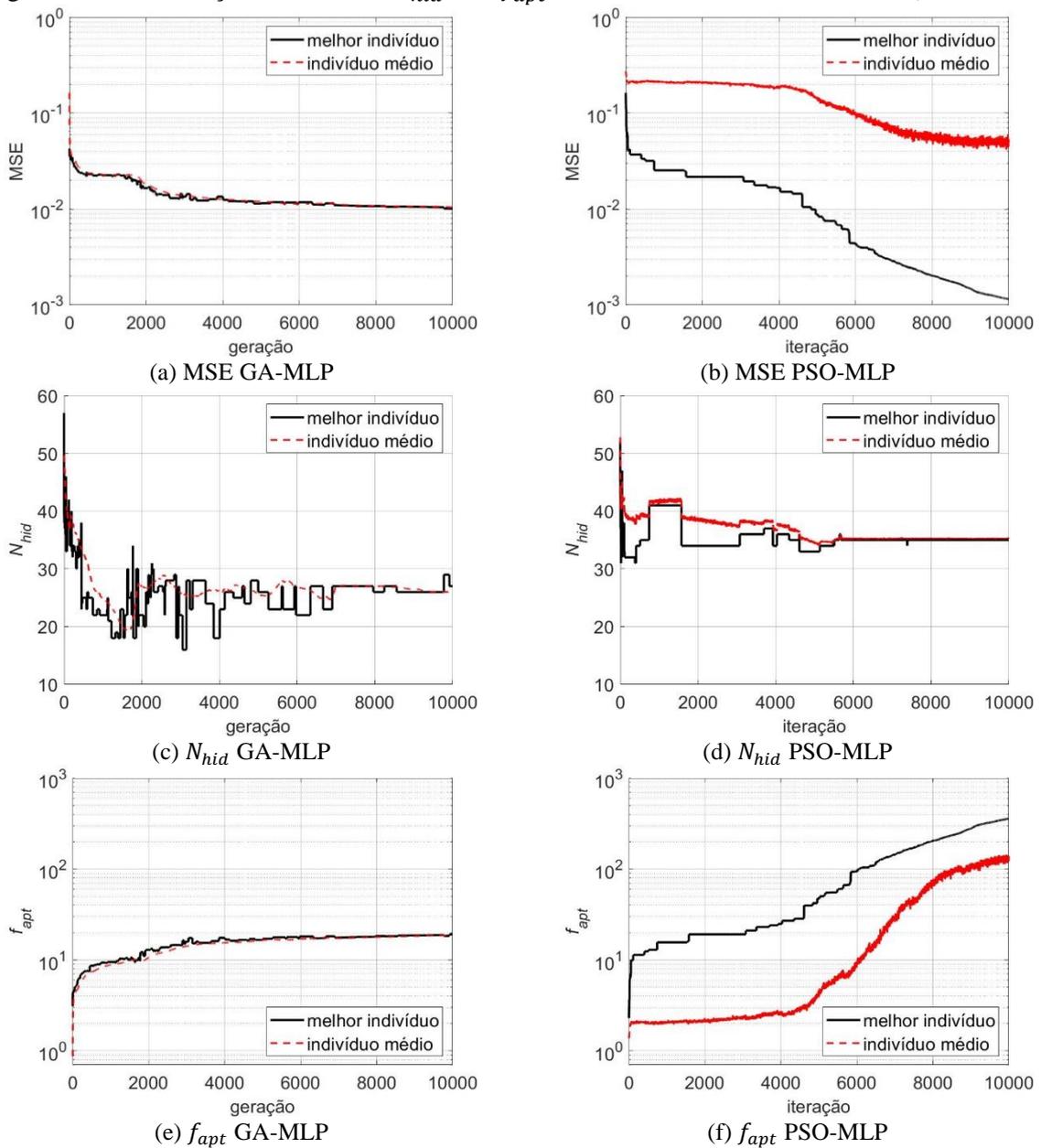
Fonte: próprio autor.

Figura 5.24 — Evolução do MSE da MLP (Problemas 2).



Fonte: próprio autor.

Figura 5.25 — Evolução do MSE, de  $N_{hid}$  e de  $f_{apt}$  nas redes GA-MLP e PSO-MLP (Problema 2).



Fonte: próprio autor.

### 5.3 O CUSTO DA OTIMIZAÇÃO

Além dos impactos já discutidos, existe um ponto importante de análise: o custo de processamento da otimização e, conseqüentemente, o tempo de treinamento dos algoritmos. Na Tabela 5.12, são apresentados o MSE e o tempo de treinamento<sup>8,9</sup> de todos os algoritmos

<sup>8</sup> Ambos os dados (MSE e tempo) são valores médios em relação às 100 execuções de cada algoritmo. Dessa forma, os dados são referentes a apenas uma execução.

<sup>9</sup> Para referência de tempo de processamento, os algoritmos foram implementados no ambiente de programação MATLAB®, versão R2018a, e o computador utilizado para executar o software é um notebook Dell Inspiron 15 série 3000, modelo I15-3576-A70C, sistema operacional Windows 10, processador 8ª geração Core™ i7 e 8Gb de RAM.

apresentados (inclusive as variações otimizadas do MLP omitidas de maior discussão na Seção 5.2) para o Problema 1. Na Tabela 5.13, os mesmos dados para o Problema 2. Destaca-se que a quantidade neurônios apresentada para os algoritmos com variação de  $N_{hid}$  é o valor mediano  $N_{hid}^{med}$  entre as 100 execuções.

Tabela 5.12 — Tempo de treinamento e MSE de teste médios (Problema 1).

| Algoritmo                     | $N_{hid}$              | MSE<br>[ $10^{-3}$ ] | Tempo<br>[s] | Algoritmo                     | $N_{hid}$               | MSE<br>[ $10^{-3}$ ] | Tempo<br>[s] |
|-------------------------------|------------------------|----------------------|--------------|-------------------------------|-------------------------|----------------------|--------------|
| ELM                           | 5                      | 3,22                 | 0,0011       | MLP                           | 5                       | 1,77                 | 62,83        |
|                               | 10                     | 0,83                 | 0,0016       |                               | 10                      | 0,87                 | 65,73        |
|                               | 20                     | 0,80                 | 0,0029       |                               | 20                      | 0,85                 | 77,56        |
| GA-ELM                        | 5                      | 0,89                 | 1,31         | GA-MLP                        | 5                       | 3,88                 | 38,5         |
|                               | 10                     | 0,78                 | 2,13         |                               | 10                      | 1,43                 | 49,3         |
|                               | 20                     | 0,73                 | 3,54         |                               | 20                      | 0,95                 | 57,2         |
| GA-ELM<br>sem<br>minimização  | $N_{hid}^{md}$<br>= 25 | 0,72                 | 4,59         | GA-MLP<br>sem<br>minimização  | $N_{hid}^{máx}$<br>= 32 | 0,83                 | 319,2        |
| GA-ELM<br>com<br>minimização  | $N_{hid}^{md}$<br>= 6  | 0,81                 | 2,04         | GA-MLP<br>com<br>minimização  | $N_{hid}^{md}$<br>= 8   | 1,33                 | 268,9        |
| PSO-ELM                       | 5                      | 0,80                 | 1,57         | PSO-MLP                       | 5                       | 3,40                 | 66,07        |
|                               | 10                     | 0,77                 | 2,81         |                               | 10                      | 1,21                 | 92,5         |
|                               | 20                     | 0,70                 | 4,42         |                               | 20                      | 0,80                 | 122,6        |
| PSO-ELM<br>sem<br>minimização | $N_{hid}^{md}$<br>= 36 | 0,64                 | 7,17         | PSO-MLP<br>sem<br>minimização | $N_{hid}^{md}$<br>= 26  | 0,79                 | 199,1        |
| PSO-ELM<br>com<br>minimização | $N_{hid}^{md}$<br>= 5  | 0,81                 | 2,80         | PSO-MLP<br>com<br>minimização | $N_{hid}^{md}$<br>= 8   | 0,83                 | 142,5        |

Fonte: próprio autor.

Antes de maiores discussões, vale destacar que o tempo colhido a cada execução de um algoritmo, individualmente, tem influência de diversos fatores circunstanciais em função do poder de processamento da máquina utilizada e de possíveis processos inerentes do próprio sistema operacional e que sofrem oscilações. Entretanto, uma vez que os valores apresentados nas Tabela 5.12 e Tabela 5.13 são médios para as 100 execuções, há minimização desses efeitos

e, se levados em consideração apenas de forma comparativa entre os algoritmos, os valores de tempo estimam bem as variações de processamento necessário.

Tabela 5.13 — Tempo de treinamento e MSE de teste médios (Problema 2).

| Algoritmo                     | $N_{hid}$            | MSE<br>[ $10^{-3}$ ] | Tempo<br>[s] | Algoritmo                     | $N_{hid}$            | MSE<br>[ $10^{-3}$ ] | Tempo<br>[s] |
|-------------------------------|----------------------|----------------------|--------------|-------------------------------|----------------------|----------------------|--------------|
| ELM                           | 10                   | 25,4                 | 0,0013       | MLP                           | 10                   | 1,54                 | 123,1        |
|                               | 20                   | 12,7                 | 0,0018       |                               | 20                   | 0,81                 | 133,8        |
|                               | 50                   | 3,9                  | 0,0032       |                               | 250                  | 2,8                  | 0,0182       |
|                               | 250                  | 2,8                  | 0,0182       |                               |                      |                      |              |
| GA-ELM                        | 10                   | 6,5                  | 7,17         | GA-MLP                        | 10                   | 14,2                 | 543,2        |
|                               | 20                   | 1,8                  | 13,80        |                               | 20                   | 12,3                 | 631,4        |
|                               | 50                   | 0,66                 | 41,85        |                               |                      |                      |              |
|                               | 250                  | 0,49                 | 304,65       |                               |                      |                      |              |
| GA-ELM<br>sem<br>minimização  | $N_{hid}^{md} = 180$ | 0,55                 | 138,7        | GA-MLP<br>sem<br>minimização  | $N_{hid}^{máx} = 73$ | 10,2                 | 2121,6       |
| GA-ELM<br>com<br>minimização  | $N_{hid}^{md} = 48$  | 0,65                 | 39,29        | GA-MLP<br>com<br>minimização  | $N_{hid}^{máx} = 27$ | 18,3                 | 1668,2       |
| PSO-ELM                       | 10                   | 2,34                 | 6,35         | PSO-MLP                       | 10                   | 3,41                 | 296,5        |
|                               | 20                   | 0,92                 | 9,42         |                               | 20                   | 1,31                 | 338,9        |
|                               | 50                   | 0,63                 | 24,07        |                               |                      |                      |              |
|                               | 250                  | 0,48                 | 384,8        |                               |                      |                      |              |
| PSO-ELM<br>sem<br>minimização | $N_{hid}^{md} = 180$ | 0,54                 | 135,1        | PSO-MLP<br>sem<br>minimização | $N_{hid}^{máx} = 81$ | 1,14                 | 2428,44      |
| PSO-ELM<br>com<br>minimização | $N_{hid}^{md} = 43$  | 0,67                 | 46,42        | PSO-MLP<br>com<br>minimização | $N_{hid}^{md} = 35$  | 1,26                 | 996,3        |

Fonte: próprio autor.

Em ambos os problemas de aproximação, a otimização da ELM de  $N_{hid}$  constante provoca um incremento substancial no tempo de treinamento da ELM de mais de 5000 vezes e de potencial ainda maior com o aumento da quantidade de neurônios ocultos. Contudo, quando comparados ao tempo da MLP, os das redes GA-ELM e PSO-ELM, são menores em pelo menos 10 vezes (para  $N_{hid} = 5, 10$  ou  $20$ ); o processo de aprendizagem da ELM sem

otimização é que, de fato, chama a atenção pela sua velocidade de execução, resultante da ausência de uma fase iterativa de treinamento.

Por outro lado, o MLP e suas variantes otimizadas de  $N_{hid}$  constante não apresentam grandes diferenças no tempo de treinamento entre si, com mesma ordem de grandeza ou próxima, pelo menos — destaca-se que o aumento da quantidade de neurônios ocultos impacta mais nas versões otimizadas e, por isso, em eventuais configurações de maior  $N_{hid}$  o tempo deve ser notoriamente mais elástico para GA-ELM, PSO-ELM, GA-MLP e PSO-MLP, como com  $N_{hid} = 250$  já indica.

Quando analisados os tempos das versões dos algoritmos otimizados com  $N_{hid}$  variável, percebe-se que eles dependem da progressão da quantidade de neurônios ocultos: quando sem minimização, o tempo aumenta, quando com minimização, diminui. É interessante notar que, em todos os casos, em menor ou maior escala, o aumento de  $N_{hid}$  é acompanhado da elevação do tempo de treinamento.

Em relação ao MSE, as versões otimizadas de  $N_{hid}$  constante, em todos os casos, apresentaram um valor menor do que as originais de mesma quantidade de neurônios ocultos, porém, ressalta-se que para  $N_{hid}$  mais alto, esse menor MSE é acompanhado de *overfitting*. Isso também é destaque para as versões sem minimização de  $N_{hid}$ , que apresentam os menores MSE; as versões com minimização demonstram MSE bastante reduzido e são menos propensas ao *overfitting*. GA-MLP não conseguiu aproximar adequadamente o Problema 2.

#### 5.4 CONSIDERAÇÕES E RESSALVAS

A otimização das redes provoca impactos positivos e negativos: se, por um lado, aumentam a eficiência da rede, por outro, elevam o processamento computacional necessário para o treinamento. A aplicação prática dos modelos otimizados deve levar em consideração o que pesa mais para a identificação em questão: a eficiência ou o tempo de treinamento.

As análises realizadas na Seção 5 como um todo constituem um preâmbulo extenso para a aplicação dos modelos obtidos no sistema de identificação alvo: o banco de dados de potência lasers semicondutores da pesquisa de Thé (2010). Na Seção 6, essa modelagem é descrita e discutida.

## 6 NEURO-OTIMIZAÇÃO APLICADA A LASERS QD

*"Então, o universo não é exatamente como você pensava que era. É melhor reorganizar suas crenças. Porque você certamente não pode reorganizar o universo."*<sup>1</sup> — Isaac Asimov

A modelagem clássica de lasers quânticos semicondutores é do tipo caixa branca, realizada por meio de equações de taxa que regem o processo ou a descrição deste em microestados. Os modelos são obtidos, em geral, em um trabalho extenso, demorado e complexo, e exigem conhecimento específico das características do sistema.

Nesta seção, discorre-se sobre os algoritmos obtidos na Seção 5 aplicados ao banco de dados de laser QD apresentado na Seção 2.4. Os modelos são analisados conforme sua capacidade de compreender e reproduzir a dinâmica; os algoritmos supervisionados clássicos são novamente utilizados como parâmetro de avaliação dos propostos a partir dos otimizadores.

### 6.1 CONSIDERAÇÕES DE MODELO E DO BANCO DE DADOS

As três entradas consideradas para o processo são: o estado de observação do laser (fundamental ou excitado), a temperatura (20°C, 40°C e 50°C) e o valor do patamar superior  $I_{top}$  do degrau de corrente aplicado no sistema. Cada estado e temperatura recebem valor numérico específico para servir de identificação na entrada do modelo, conforme Tabela 6.1 e Tabela 6.2, respectivamente.

Os modelos adotados são de base não linear, discretos, paramétricos e MISO: a janela de regressão é determinada empiricamente e, apresentada na Tabela 6.3. Por conta de as entradas serem indicativas e constantes dentro de cada série temporal, não há muito sentido em usar regressão nelas ou um tempo de atraso não nulo.

Tabela 6.1 — Parametrização dos estados do laser nos ensaios.

| <b>Estado</b> | ES  | GS  |
|---------------|-----|-----|
| <b>Valor</b>  | 0,4 | 0,8 |

Fonte: próprio autor.

Tabela 6.2 — Parametrização das temperaturas dos ensaios.

| <b>Temperatura</b> | 20°C | 40°C | 50°C |
|--------------------|------|------|------|
| <b>Valor</b>       | 0,2  | 0,5  | 0,8  |

Fonte: próprio autor.

<sup>1</sup> Tradução nossa de trecho do capítulo nove do livro de ficção *Ao Cair da Noite (Nightfall)*, de Isaac Asimov (1920-1992) e de Robert Silverberg (1935-). A frase é do personagem Athor: “*So the universe is not quite as you thought it was. You’d better rearrange your beliefs, then. Because you certainly can’t rearrange the universe*”.

Tabela 6.3 — Janela de regressão do modelo.

| Parâmetro        | Símbolo  | Valor |
|------------------|----------|-------|
| Ordem da saída   | $n_y$    | 4     |
| Ordem da entrada | $n_u$    | 1     |
| Tempo de atraso  | $\tau_d$ | 0     |

Fonte: próprio autor.

Os dados de resposta em potência óptica do sistema tem normalização dentro do intervalo  $[0,2; 0,8]$ , para se evitar a região de saturação próxima às extremidades (0 e 1) da função de ativação logística (Tabela 6.4). Isso é importante porque os dados possuem quantidade razoável de valores de saída próximos aos mínimos e máximos do banco de dados e a saturação na ativação dos neurônios poderia causar distorções no reconhecimento de padrões esperado. Destaca-se que do total de 324 séries temporais, apenas 97 (cerca de 30%) são utilizadas para treinamento, isso porque o banco de dados é extenso e apresenta muitas informações similares.

Tabela 6.4 — Parâmetros gerais dos algoritmos.

| Parâmetro                        | Símbolo          | Valor adotado                       |
|----------------------------------|------------------|-------------------------------------|
| Quantidade de execuções          | —                | 100                                 |
| Função de ativação               | $\varphi(\cdot)$ | $\varphi(v) = \frac{1}{1 - e^{-v}}$ |
| Normalização dos dados           | —                | $[0,2; 0,8]$                        |
| Percentual dos dados para treino | —                | 30%                                 |

Fonte: próprio autor.

O vetor de entradas da rede é definido, então, conforme (já incluso o bias)

$$\mathbf{x}(t) = [-1 \quad y(t-1) \quad y(t-2) \quad y(t-3) \quad y(t-4) \quad I_{top} \quad K_E \quad K_T]^T, \quad (6.1)$$

em que  $K_E$  e  $K_T$  são respectivamente, o estado e a temperatura. Uma boa prática é trazer todos os componentes do vetor para a mesma faixa de valores e, por isso, a normalização de todas as três entradas se dá dentro do intervalo  $[0,2; 0,8]$ , inclusive para  $I_{top}$ .

Assim como para os problemas de aproximação de função, os algoritmos propostos para a modelagem do laser QD são executados 100 vezes pelos mesmos fins estatísticos. No geral, a análise de qualidade dos modelos leva em consideração a média e a dispersão das execuções.

## 6.2 PARÂMETROS DE TREINAMENTO DOS MODELOS

Dos modelos abordados na Seção 5, são utilizados para identificação do banco de dados do laser: ELM; MLP; GA-ELM, PSO-ELM, GA-MLP e PSO-MLP com  $N_{hid}$  constante; GA-ELM, PSO-ELM, GA-MLP e PSO-MLP com  $N_{hid}$  minimizado. Os demais não são utilizados por serem tidos como ineficazes na proposta desta aplicação.

Os parâmetros de treinamento desses modelos são elencados a seguir da seguinte forma: comuns ao AG e ao PSO (Tabela 6.5); por AG somente (Tabela 6.6); por PSO somente (Tabela 6.7); e por retropropagação do erro (Tabela 6.8). A quantidade de neurônios escondidos utilizados na ELM é o mesmo do MLP (Tabela 6.8).

Tabela 6.5 — Parâmetros comuns ao treinamento por AG e PSO.

| Parâmetro                        | Símbolo         | Valores Típicos        | Valor adotado        |                  |
|----------------------------------|-----------------|------------------------|----------------------|------------------|
|                                  |                 |                        | GA-ELM / PSO-ELM     | GA-MLP / PSO-MLP |
| Quantidade máxima de $N_{hid}$   | $N_{hid}^{max}$ | —                      | $N_{hid}^{max} = 50$ |                  |
| Quantidade mínima de $N_{hid}$   | $N_{hid}^{min}$ | $N_{hid}^{min} > 1$    | $N_{hid}^{min} = 5$  |                  |
| Expoente do erro na aptidão      | $\gamma_e$      | —                      | $\gamma_{erro} = 1$  |                  |
| Expoente de $N_{hid}$ na aptidão | $\gamma_h$      | —                      | $\gamma_{hid} = 0,5$ |                  |
| Quantidade de iterações/gerações | $N_{it}, N_g$   | $N_{it}, N_g \geq 100$ | 100                  | 1000             |
| Tamanho da população             | $N_{pop}$       | $N_{pop} \geq 100$     | 150                  | 400              |

Fonte: próprio autor.

Tabela 6.6 — Parâmetros de treinamento por AG.

| Parâmetro                           | Símbolo   | Valores Típicos   | Valor adotado  |                |
|-------------------------------------|-----------|-------------------|----------------|----------------|
|                                     |           |                   | GA-ELM         | GA-MLP         |
| Probabilidade de cruzamento real    | $p_c^r$   | $p_c^r \geq 0,7$  | $p_c^r = 0,9$  | $p_c^r = 0,9$  |
| Probabilidade de mutação real       | $p_m^r$   | $p_m^r \leq 0,05$ | $p_m^r = 0,05$ |                |
| Probabilidade de cruzamento binário | $p_c^b$   | $p_c^b \geq 0,7$  | $p_c^b = 0,9$  | $p_c^b = 0,9$  |
| Probabilidade de mutação binária    | $p_m^b$   | $p_m^b \leq 0,05$ | $p_m^b = 0,05$ |                |
| Operador BLX-alfa                   | $\alpha$  | $\alpha \leq 1$   | $\alpha = 0,3$ | $\alpha = 0,7$ |
| Número de indivíduos no torneio     | $N_{tor}$ | potências de 2    | $N_{tor} = 32$ | $N_{tor} = 2$  |

Fonte: próprio autor.

Tabela 6.7 — Parâmetros de treinamento por PSO.

| Parâmetro                         | Símbolo                   | Valores Típicos | Valor adotado                             |         |
|-----------------------------------|---------------------------|-----------------|---|---------|
|                                   |                           |                 | PSO-ELM                                   | PSO-MLP |
| Pesos cognitivo e social reais    | $c_1^r$ e $c_2^r$         | Equação (4.15)  | $c_1^r = c_2^r = 1,35$                    |         |
| Coefficiente de inércia real      | $c_0^r$                   | Equação (4.14)  | $c_{0i}^r = 0,7$ e<br>$c_{0f}^r = 1,1$    |         |
| Pesos cognitivo e social binários | $c_1^b$ e $c_2^b$         | Equação (4.15)  | $c_1^b = c_2^b = 1,35$                    |         |
| Coefficiente de inércia binário   | $c_0^b$                   | Equação (4.14)  | $c_{0i}^b = 0,9$ e<br>$c_{0f}^b = 1,1$    |         |
| Posição das partículas            | $z_{max}^r$ e $z_{min}^r$ | —               | $z_{max}^r = 10$ e<br>$z_{min}^r = -10$   |         |
| Velocidade real das partículas    | $v_{max}^r$ e $v_{min}^r$ | —               | $v_{max}^r = 0,5$ e<br>$v_{min}^r = -0,5$ |         |
| Velocidade binária das partículas | $v_{max}^b$ e $v_{min}^b$ | —               | $v_{max}^b = 5$ e<br>$v_{min}^b = -5$     |         |

Fonte: próprio autor.

Tabela 6.8 — Parâmetros do treinamento convencional.

| Parâmetro                             | Símbolo   | Valores Típicos   | Valor adotado                       |
|---------------------------------------|-----------|-------------------|-------------------------------------|
| Quantidade de épocas (MLP)            | $N_e$     | $E \geq 1000$     | $E = 50000$                         |
| Taxa de aprendizagem (MLP)            | $\eta$    | $0 < \eta \leq 1$ | $\eta_0 = 0,5$ e<br>$\eta_f = 0,05$ |
| Treinamento (MLP)                     | —         | local ou por lote | local                               |
| Quantidade de neurônios ocultos (MLP) | $N_{hid}$ | $N_{hid} > 1$     | [5; 20]                             |
| Quantidade de neurônios ocultos (ELM) | $N_{hid}$ | $N_{hid} > 1$     | [5; 50]                             |

Fonte: próprio autor.

No geral, os valores adotados empiricamente para os parâmetros de treinamento estão em linha com as perspectivas já abordadas para aproximação de função, na Seção 5. Outra contribuição importante dessa análise anterior é a contribuição assertiva para configuração dos algoritmos, uma vez que, por serem bancos de dados controláveis e com bem menos amostras, possibilitam com menos esforços repetidas tentativas para a parametrização.

Destacam-se o número de indivíduos no torneio  $N_{tor}$  e o operador  $\alpha$  do método BLX-alfa, respectivamente, 2 e 0,7, valores relativamente fora de linha aos modelos abordados na Seção 5. Esses valores são tomados a fim de prevenir uma convergência prematura, reduzindo a pressão de seleção (por meio de  $N_{tor}$ ) e aumentando a diversidade da população (pelo  $\alpha$ ).

### 6.3 MODELAGEM POR ELM, GA-ELM E PSO-ELM

Apesar de cada um dos algoritmos ter sido treinado para aprender a dinâmica do banco de dados para todas as temperaturas e estados, em um mesmo modelo, visualmente, é mais racional apresentar os resultados por temperatura e estado. Na Figura 6.1, estão apresentados os modelos da rede ELM e suas variantes otimizadas para o caso de 20°C/ES; de forma semelhante, nas Figura 6.2, Figura 6.3, Figura 6.4 e Figura 6.5 para 20°C/GS, 40°C/ES, 40°C/GS e 50°C/ES, respectivamente — os gráficos apresentam os resultados médios das 100 execuções coloridos em proporção de até 0,05, conforme barra de cores.

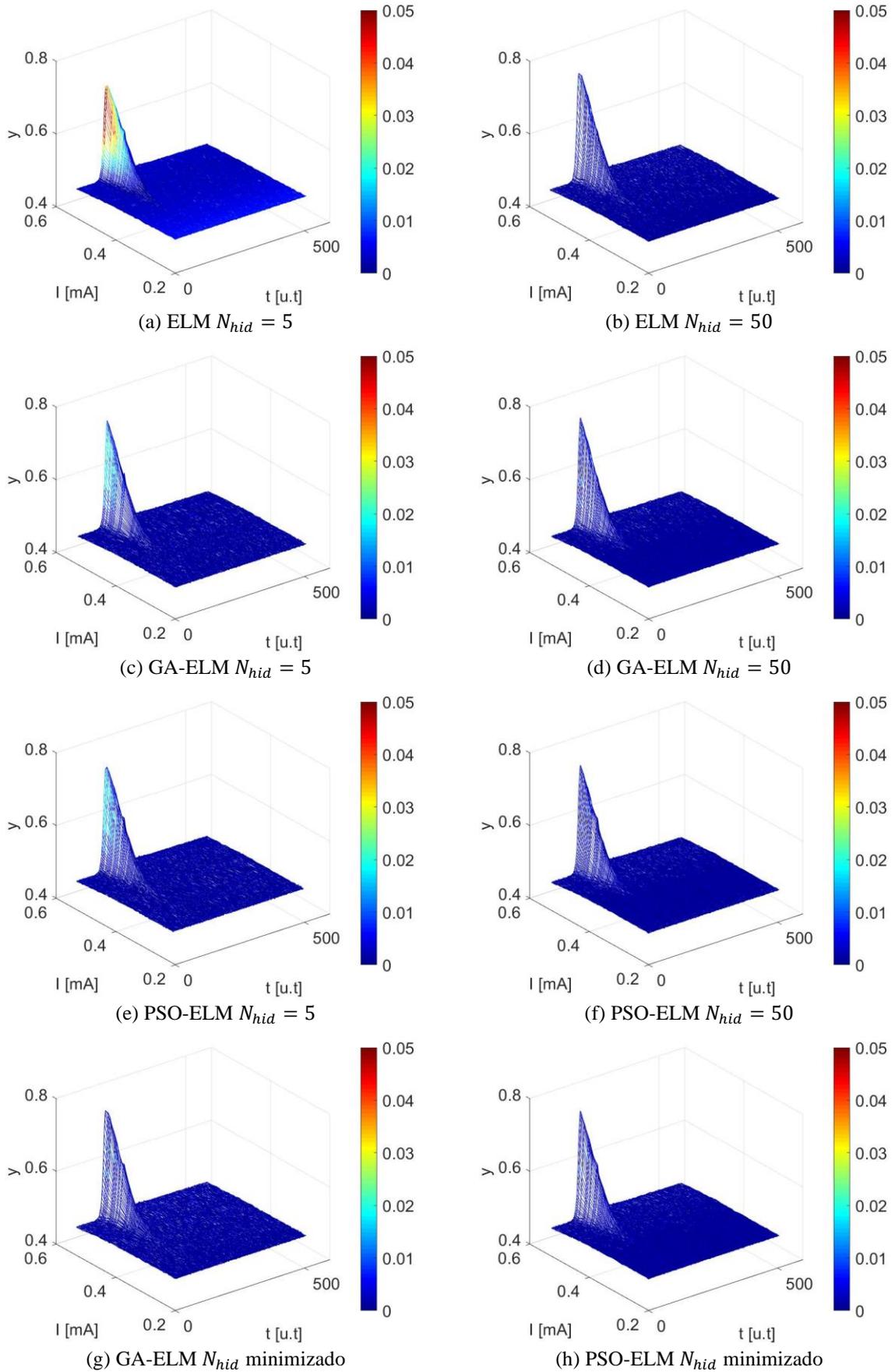
Alguns pontos importantes a serem notados:

- a) para todos os pares de temperatura-estado, a ELM com  $N_{hid} = 5$  apresenta a pior identificação entre os apresentados;
- b) para  $N_{hid} = 5$ , a GA-ELM e a PSO-ELM, em geral, são notavelmente superiores à sua correlata ELM;
- c) para  $N_{hid} = 50$ , a melhora dos algoritmos otimizadas é pouca efetiva;
- d) os resultados da GA-ELM e PSO-ELM são os mais satisfatórios ou próximos dele em relação ao MSE, além do maior grau de eficiência;
- e) a identificação é mais fácil para o caso de 40°C e estado fundamental do que para qualquer outro;
- f) a mediana da quantidade final de neurônios  $N_{hid}^{md}$  na GA-ELM e na PSO-ELM é igual a 6 e 7, respectivamente.

Nas Figura 6.6 e Figura 6.7, são apresentadas as validações para cada um dos modelos ELM e seus variantes otimizadas; testes realizados:  $r_{\xi'\xi'}(\tau) = \delta(\tau)$ ,  $r_{\xi'\xi^{2'}}(\tau) = 0$ ,  $r_{\xi^{2'}\xi^{2'}}(\tau) = \delta(\tau)$  e  $r_{(y\xi)'\xi^{2'}}(\tau) = 0$ . A análise de resíduos que envolve entradas, nesse caso, não faz sentido, pelos motivos já mencionados em justificativa da ordem unitária de entrada e do tempo de atraso nulo.

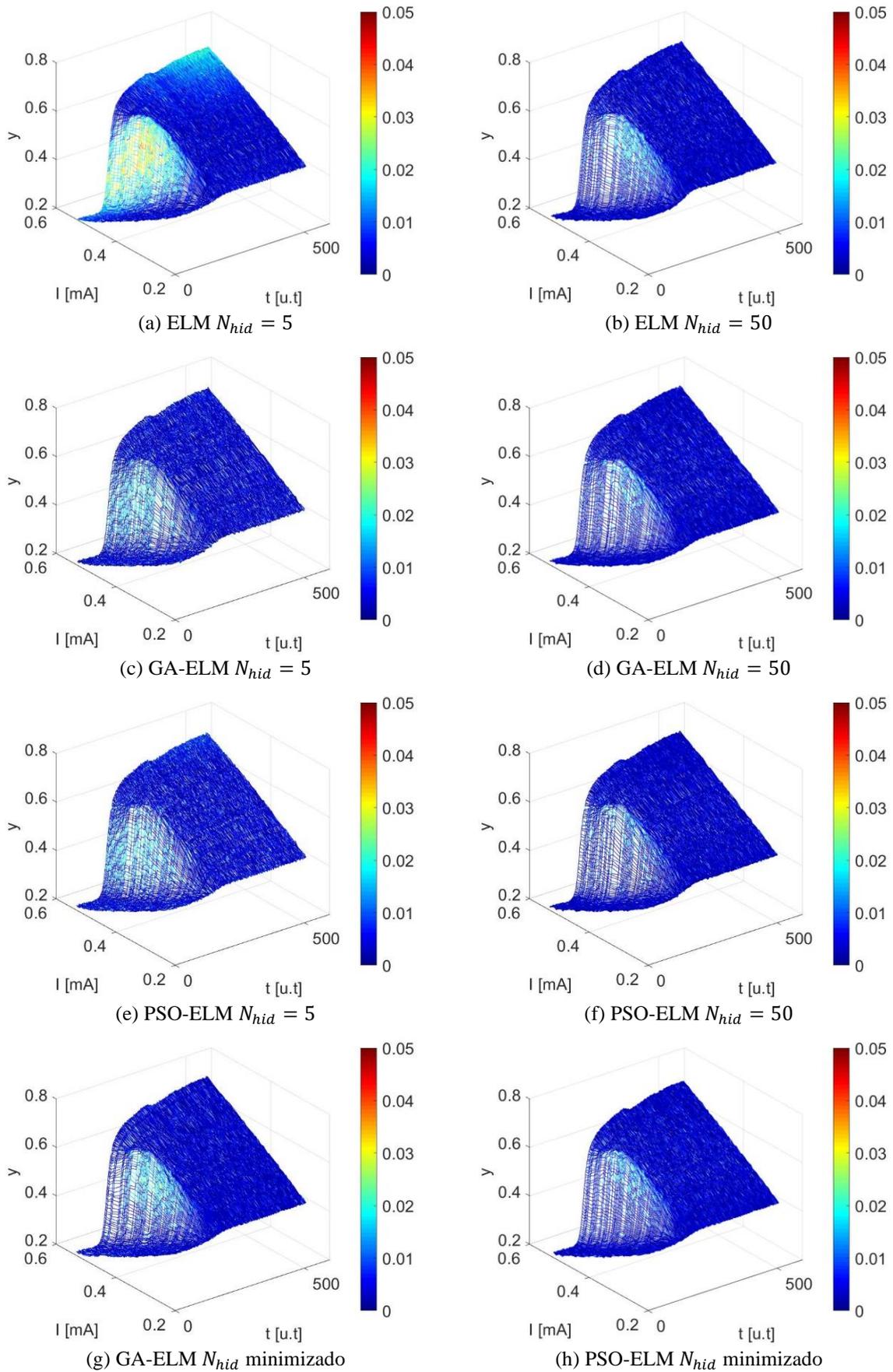
À exceção de ELM com  $N_{hid} = 5$ , os demais modelos responderam satisfatoriamente à análise. A GA-ELM e a PSO-ELM minimizaram o número de neurônios a valores de próximos deste e obtiveram validação semelhante (ou melhor) que a ELM de  $N_{hid} = 50$ .

Figura 6.1 — Modelagem do laser QD a 20°C e ES por ELM, GA-ELM e PSO-ELM.



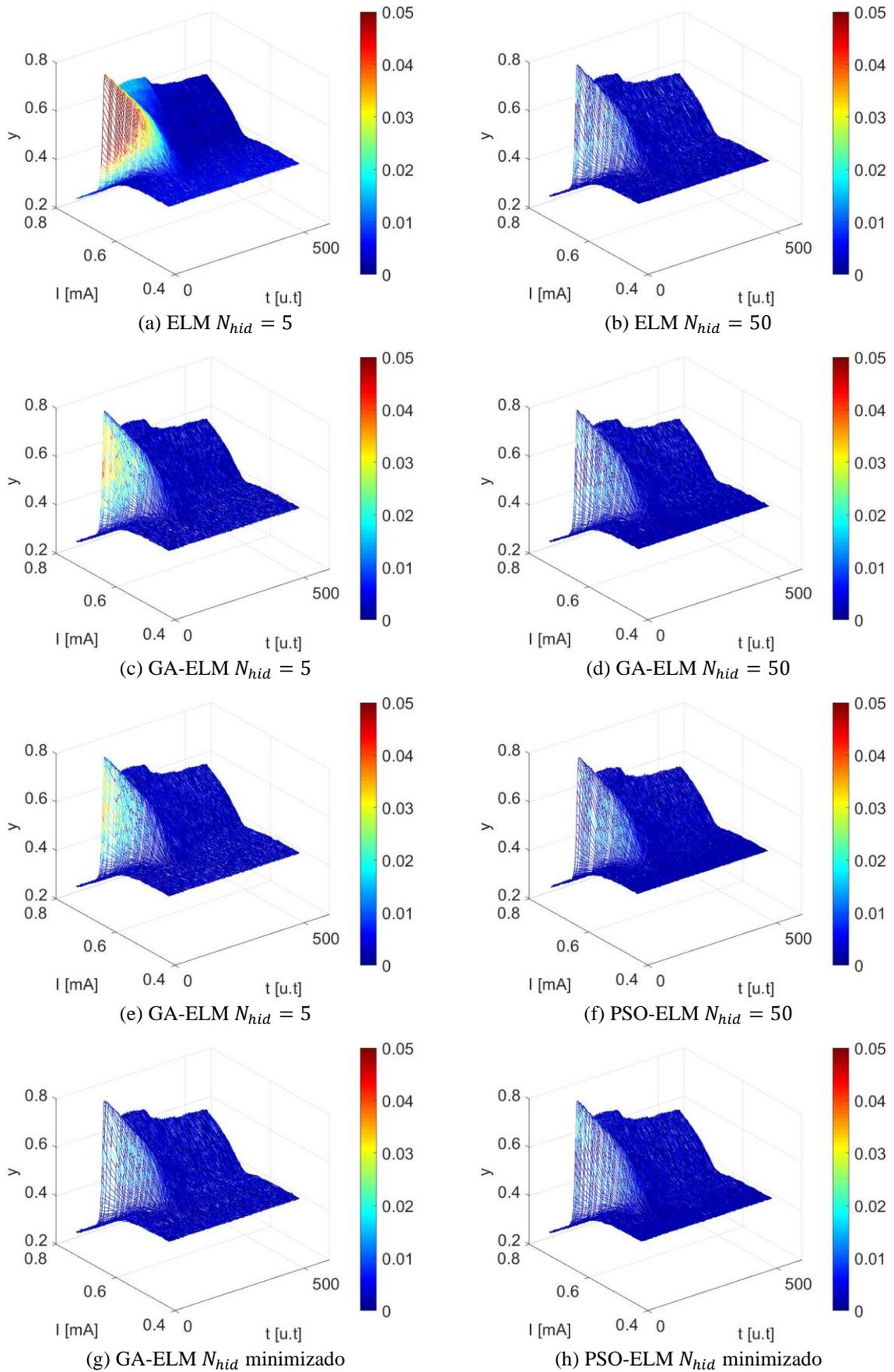
Fonte: próprio autor.

Figura 6.2 — Modelagem do laser QD a 20°C e GS por ELM, GA-ELM e PSO-ELM.



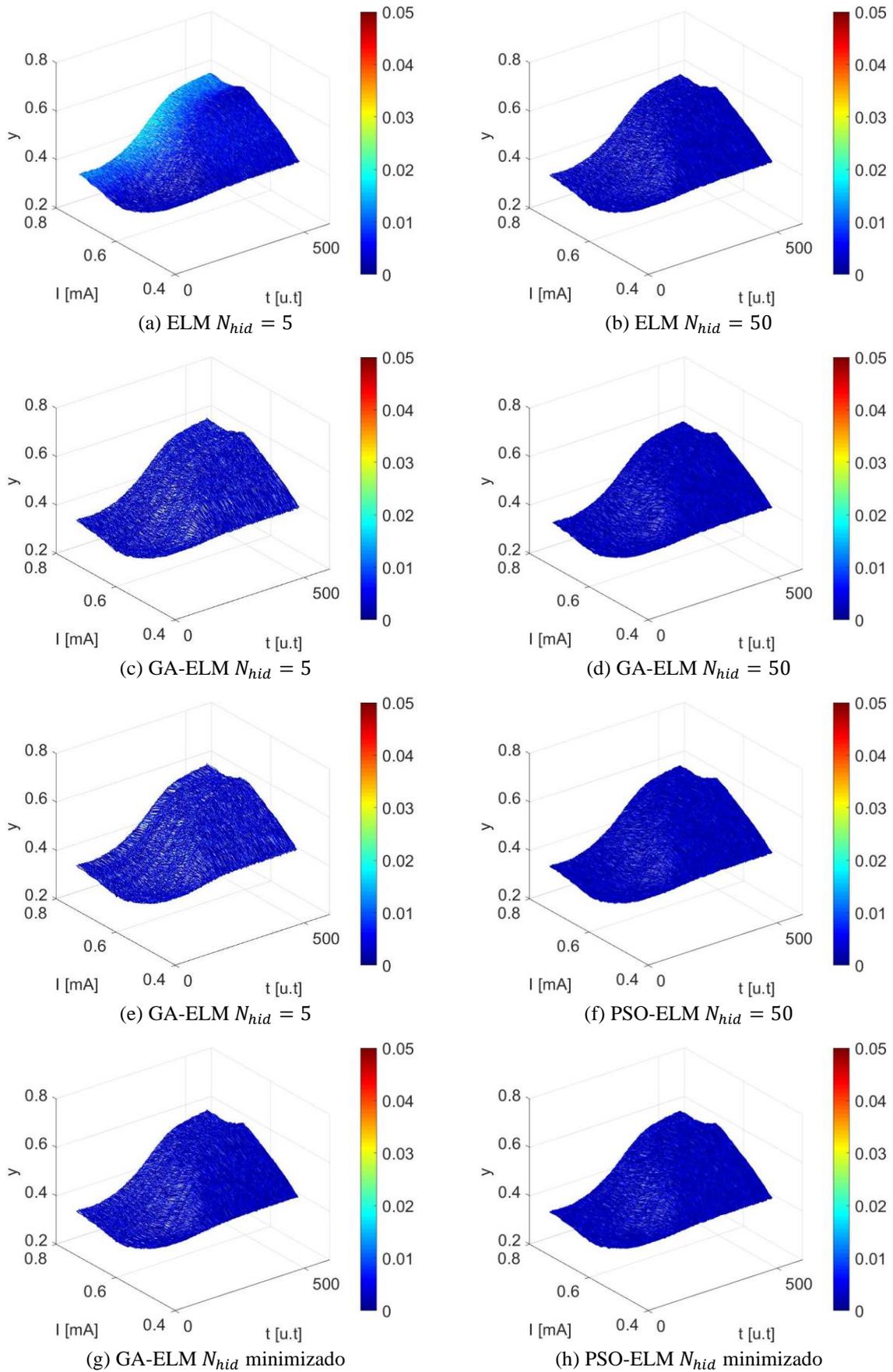
Fonte: próprio autor.

Figura 6.3 — Modelagem do laser QD a 40°C e ES por ELM, GA-ELM e PSO-ELM.



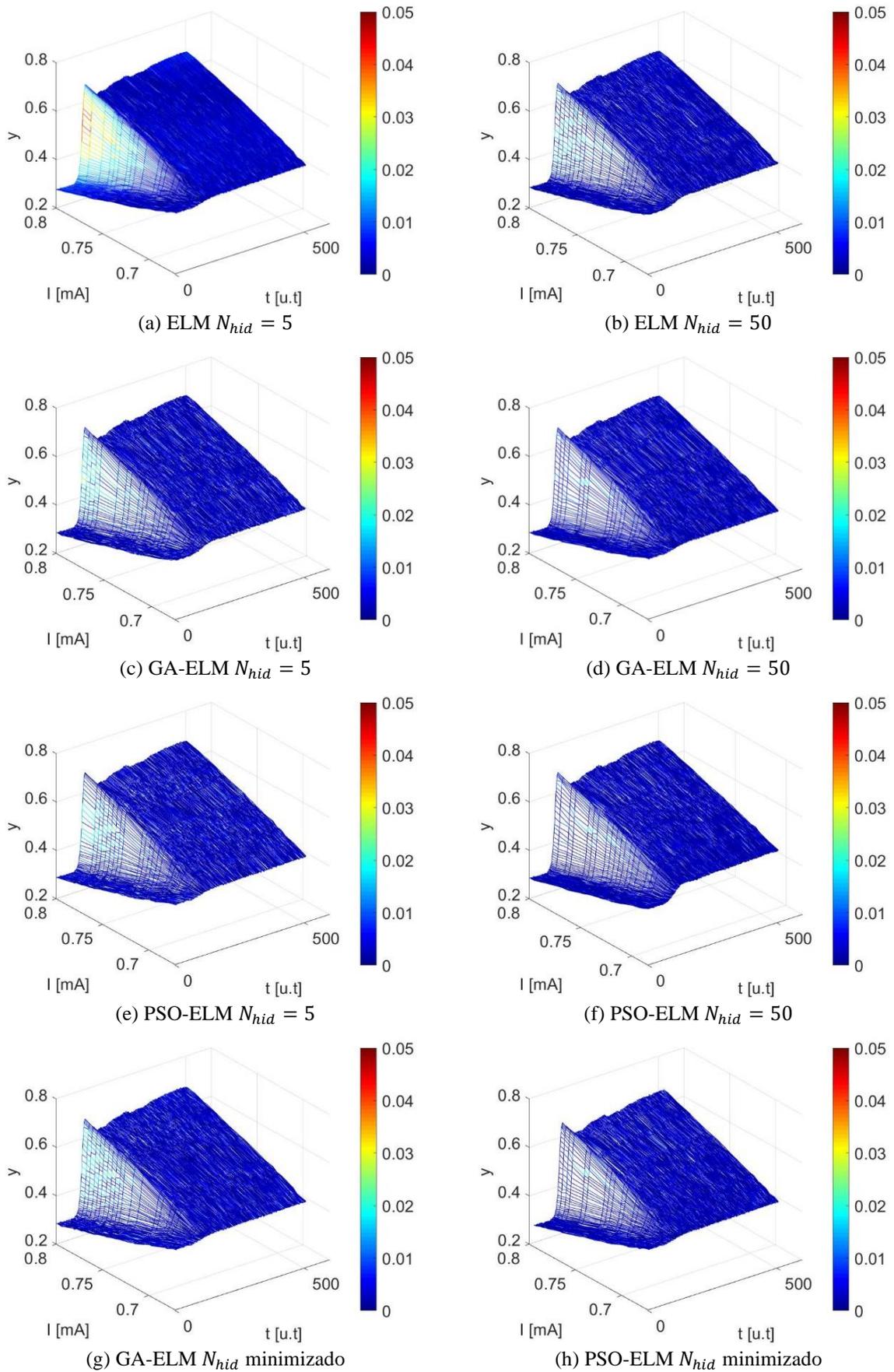
Fonte: próprio autor.

Figura 6.4 — Modelagem do laser QD a 40°C e GS por ELM, GA-ELM e PSO-ELM.



Fonte: próprio autor.

Figura 6.5 — Modelagem do laser QD a 50°C e ES por ELM, GA-ELM e PSO-ELM.



Fonte: próprio autor.

Figura 6.6 — Validação das redes ELM, GA-ELM e PSO-ELM.

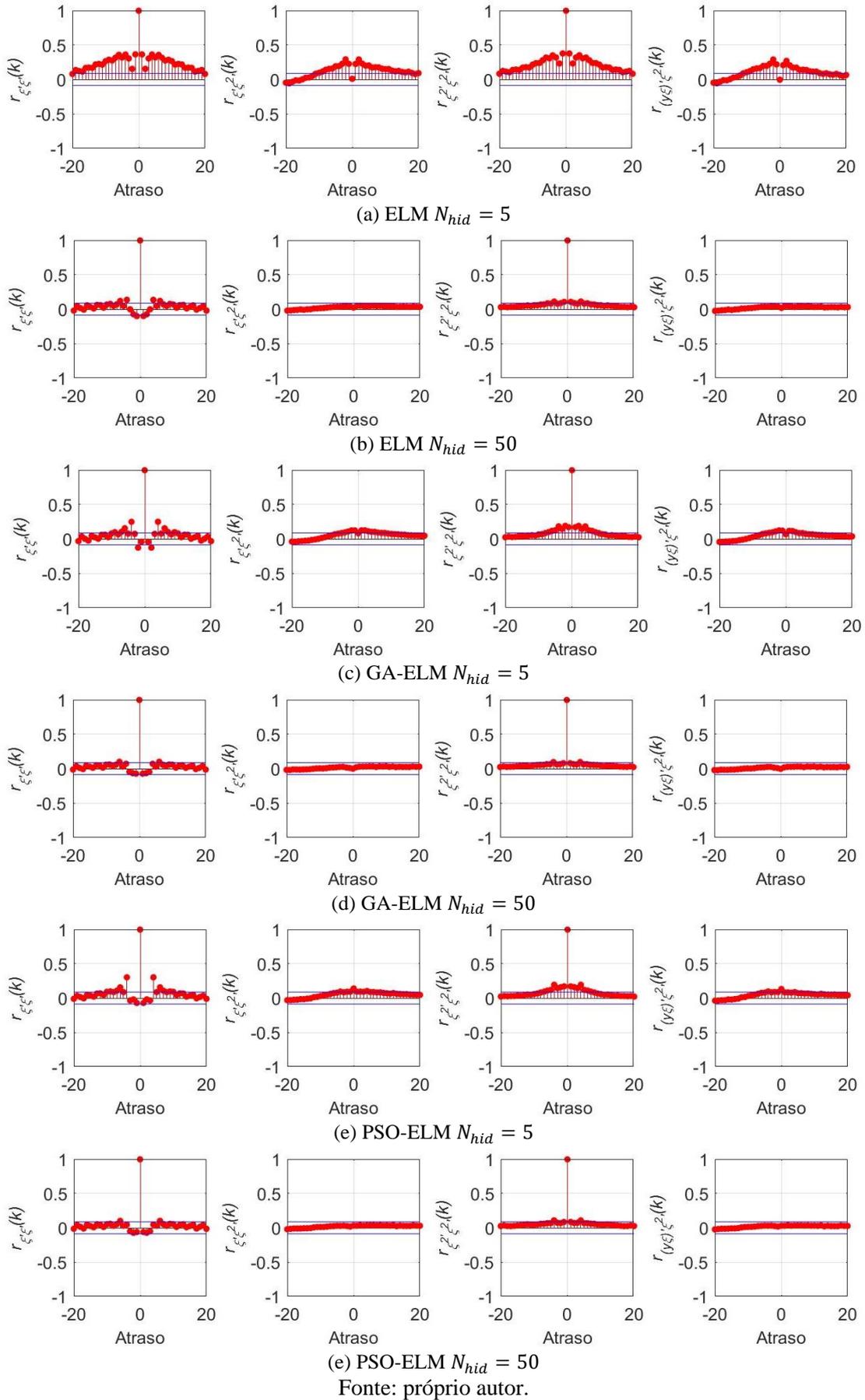
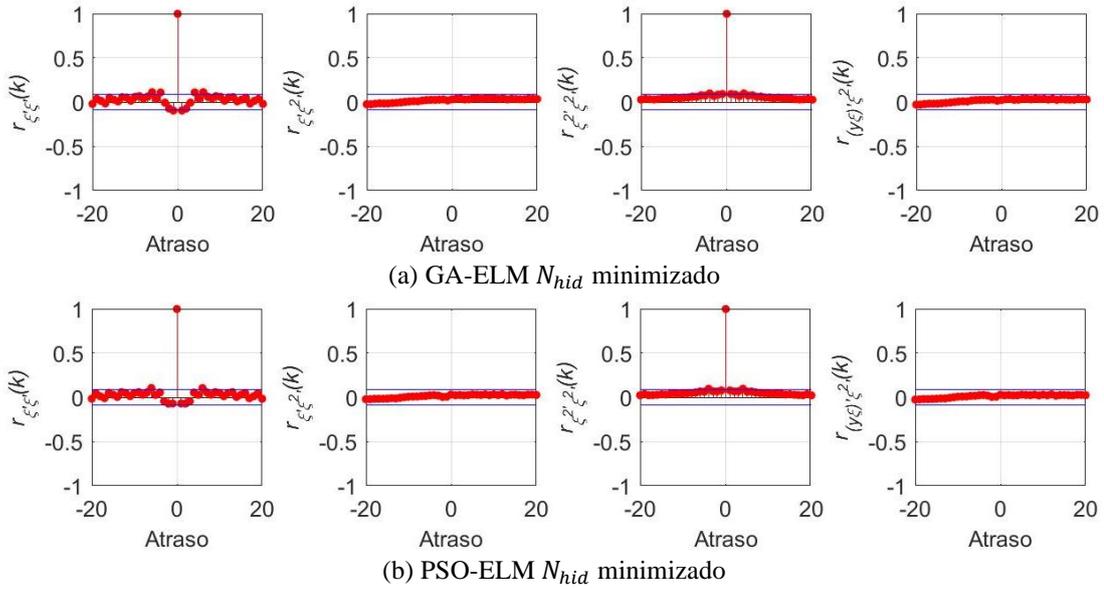


Figura 6.7 — Validação das redes GA-ELM e PSO-ELM com  $N_{hid}$  minimizado.



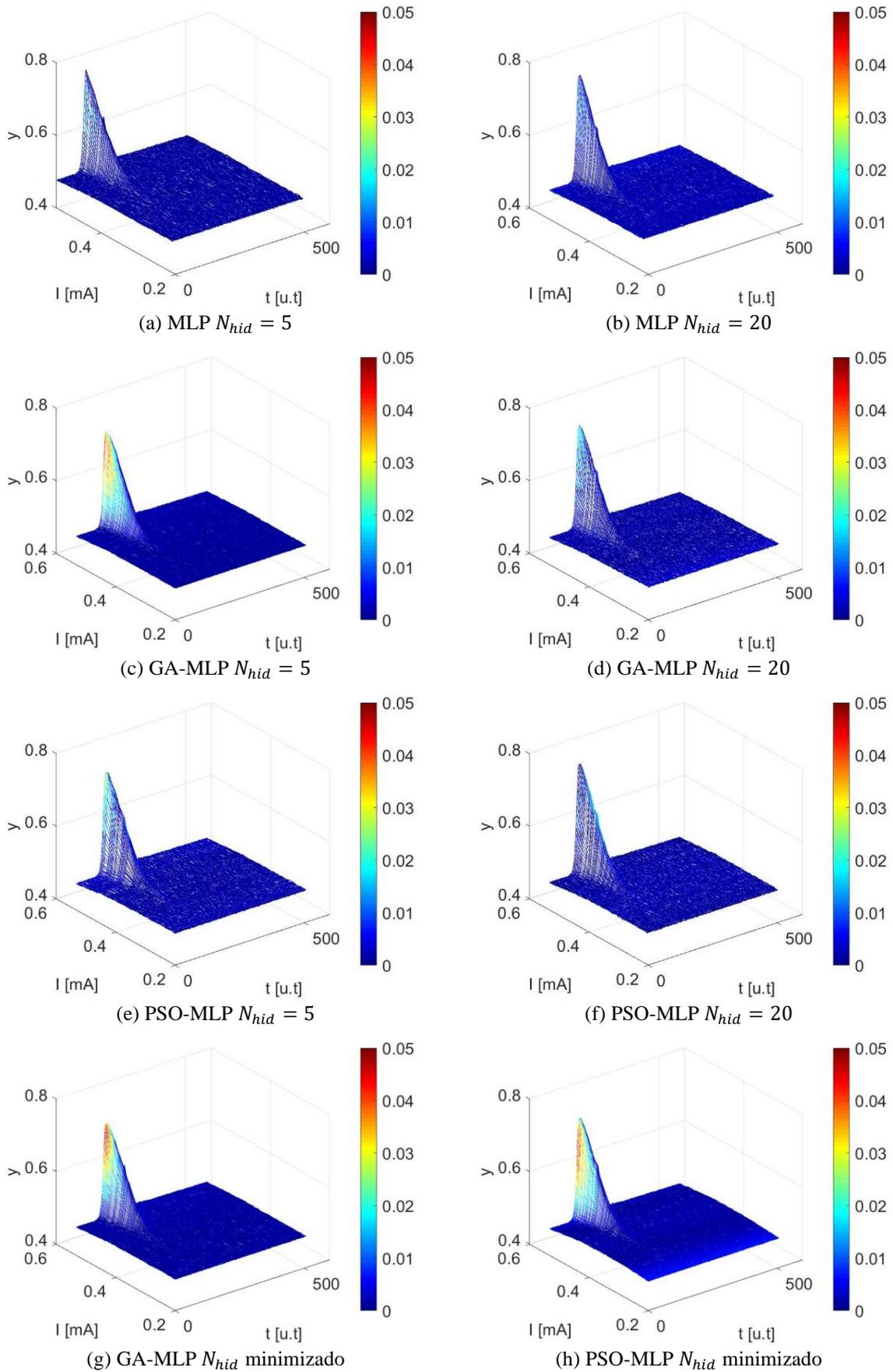
### 6.3.1 Modelagem por MLP, GA-MLP e PSO-MLP

Seguindo o mesmo racional de apresentação dos resultados da ELM e suas variantes, na Figura 6.8, estão apresentados os modelos da rede MLP e suas versões otimizadas para o caso de 20°C/ES; de forma semelhante, nas Figura 6.9, Figura 6.10, Figura 6.11 e Figura 6.12 para 20°C/GS, 40°C/ES, 40°C/GS e 50°C/ES, respectivamente. Alguns pontos importantes:

- a) o MLP, em geral, apresenta resultados tão ou mais satisfatórios que suas versões otimizadas;
- b) no geral, quando há, a melhora dos algoritmos otimizados é pouca efetiva;
- c) os resultados do GA-MLP e do PSO-MLP não são satisfatórios, apresentando zonas de maior MSE, apesar do maior grau de eficiência;
- d) a identificação é mais fácil para o caso de 40°C/GS e mais difícil para 40°C/ES e 20°C/GS;
- e) a mediana da quantidade final de neurônios  $N_{hid}^{md}$  na GA-ELM e na PSO-ELM é igual a 23 e, respectivamente.

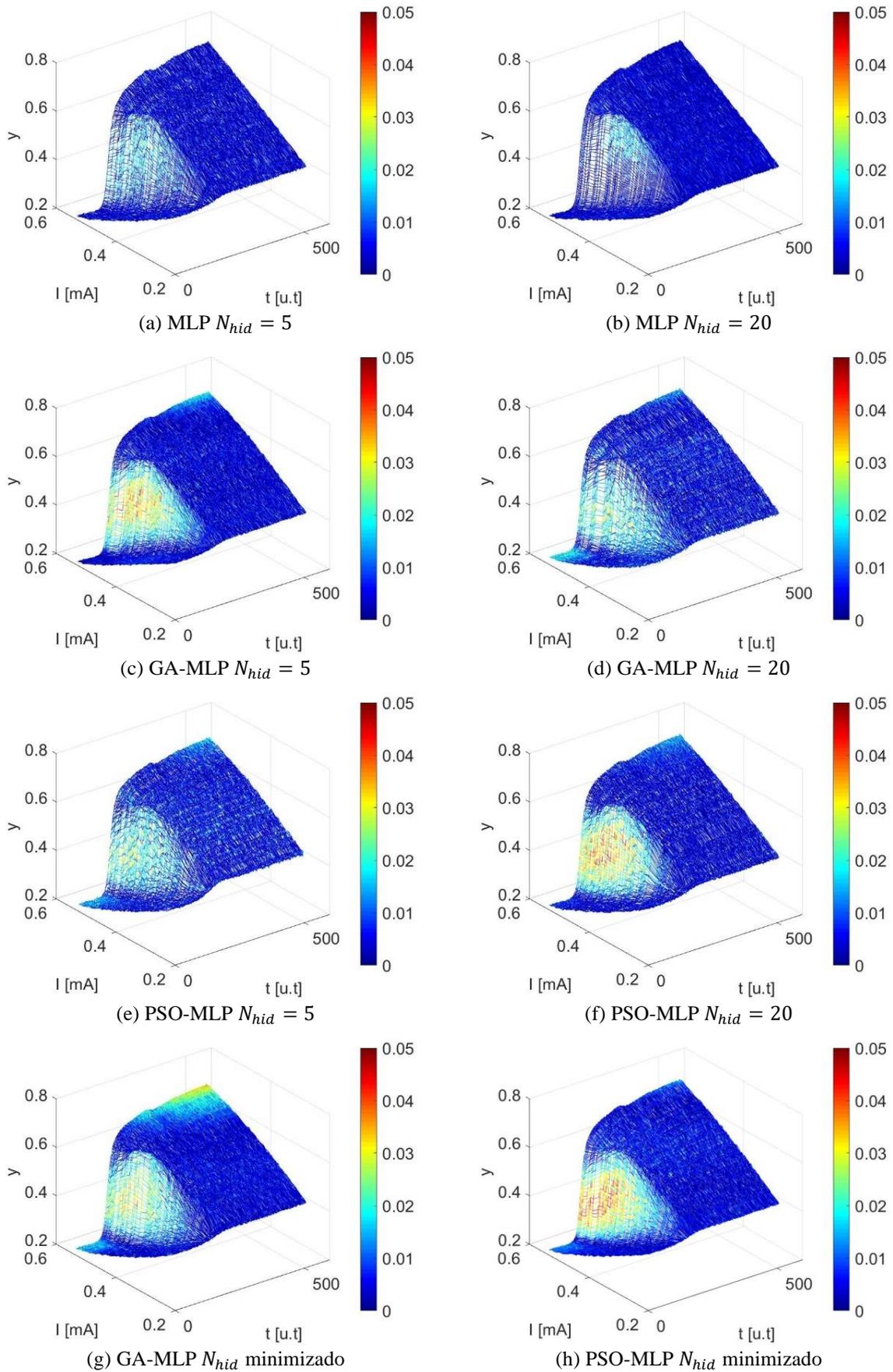
Nas Figura 6.13 e Figura 6.14, são apresentadas as validações para cada um dos modelos MLP e seus variantes otimizados:  $r_{\xi' \xi'}(\tau) = \delta(\tau)$ ,  $r_{\xi' \xi^{2'}}(\tau) = 0$ ,  $r_{\xi^{2'} \xi^{2'}}(\tau) = \delta(\tau)$  e  $r_{(y\xi)' \xi^{2'}}(\tau) = 0$ ; assim como para a ELM, a análise de resíduos que envolve entradas não faz sentido. Todos os algoritmos otimizados não apresentam validação consistente; a MLP treinada por *backpropagation* se acomoda muito melhor no intervalo de confiança e, ainda assim, apresenta discrepância em  $\tau = 0$  para  $r_{\xi' \xi^{2'}}(\tau)$  e  $r_{(y\xi)' \xi^{2'}}(\tau)$ , de relação linear-quadrática.

Figura 6.8 — Modelagem do laser QD a 20°C e ES por MLP, GA-MLP e PSO-MLP.



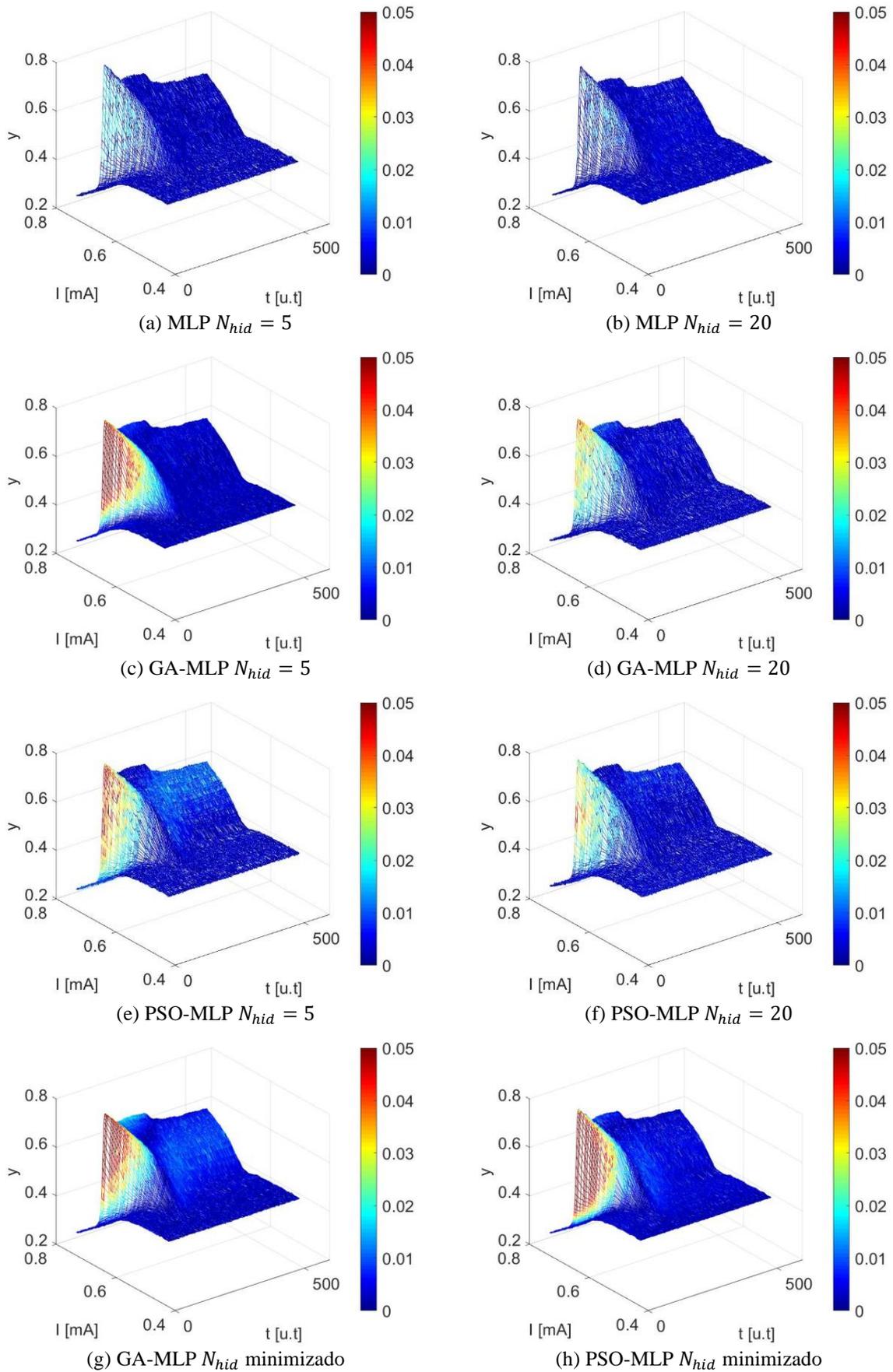
Fonte: próprio autor.

Figura 6.9 — Modelagem do laser QD a 20°C e GS por MLP, GA-MLP e PSO-MLP.



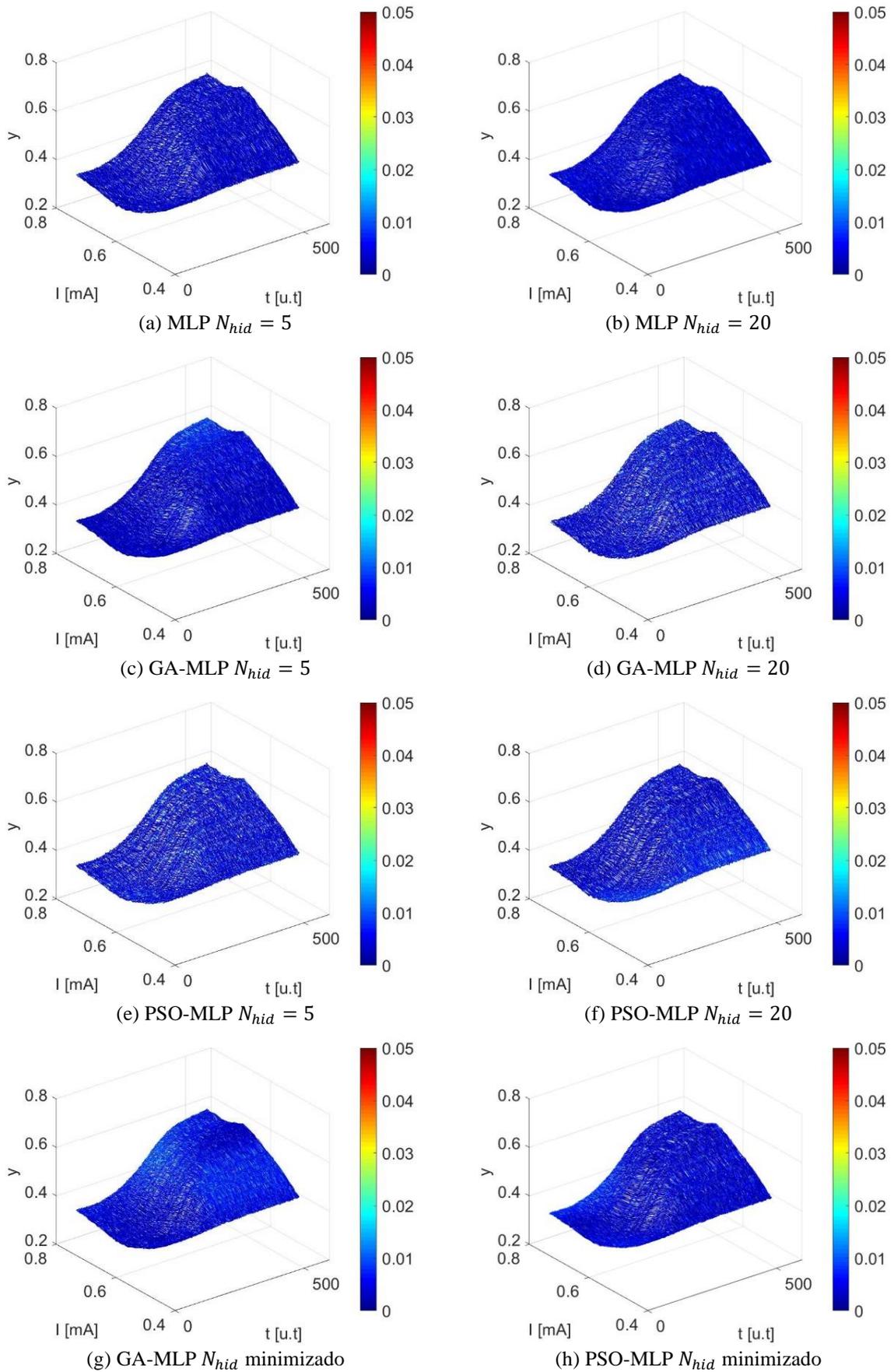
Fonte: próprio autor.

Figura 6.10 — Modelagem do laser QD a 40°C e ES por MLP, GA-MLP e PSO-MLP.



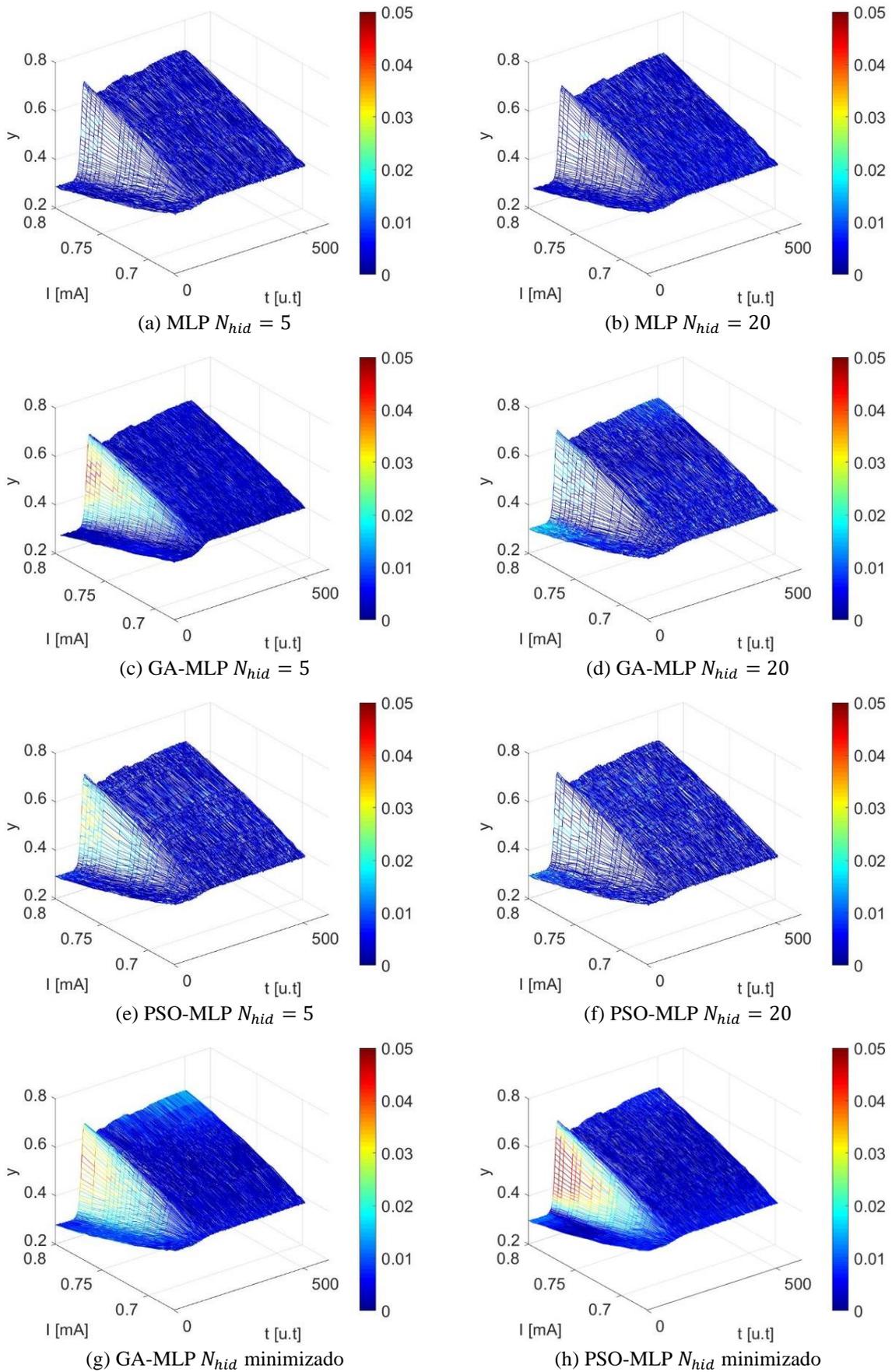
Fonte: próprio autor.

Figura 6.11 — Modelagem do laser QD a 40°C e GS por MLP, GA-MLP e PSO-MLP.



Fonte: próprio autor.

Figura 6.12 — Modelagem do laser QD a 50°C e ES por MLP, GA-MLP e PSO-MLP.



Fonte: próprio autor.

Figura 6.13 — Validação das redes MLP, GA-MLP e PSO-MLP.

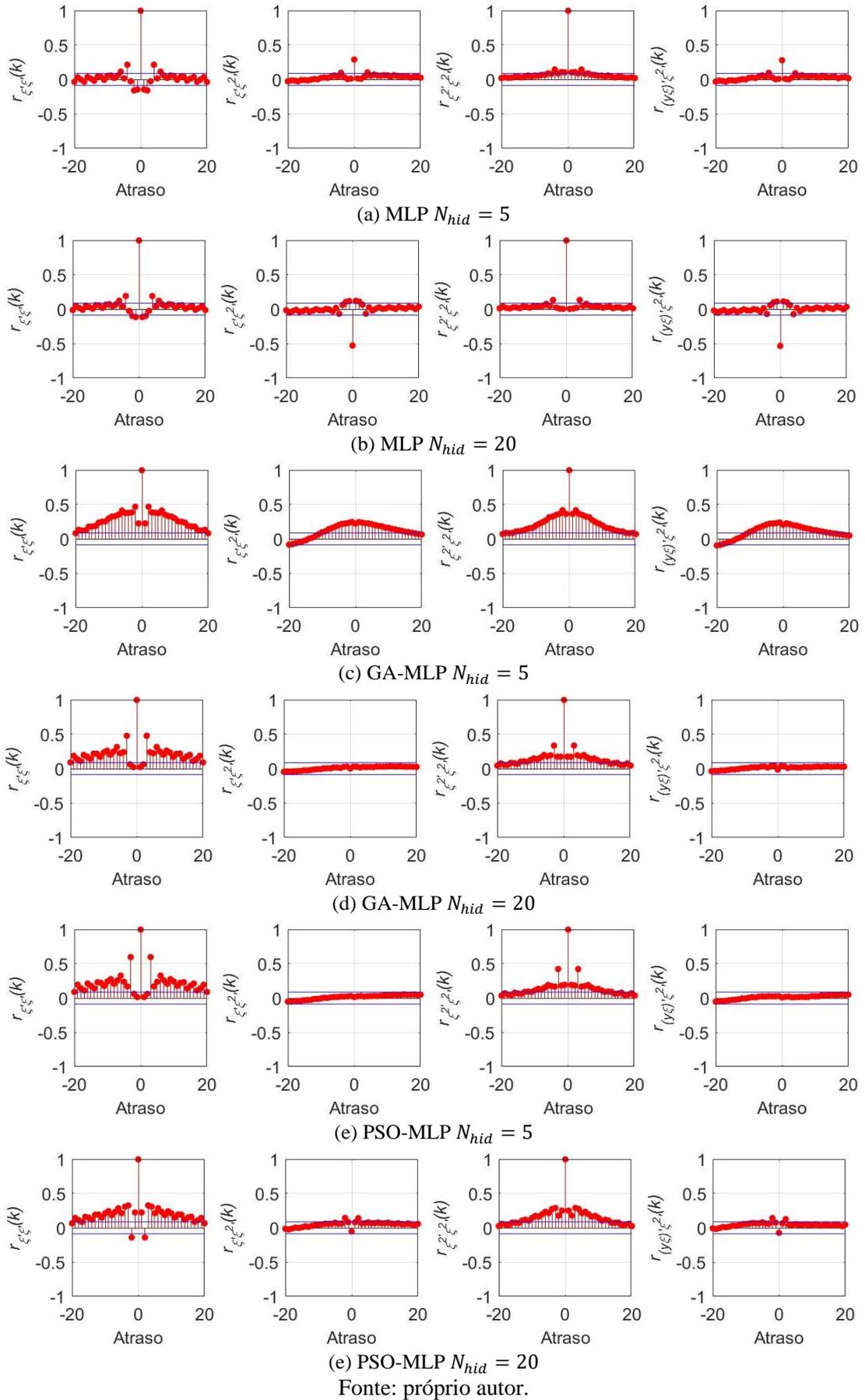
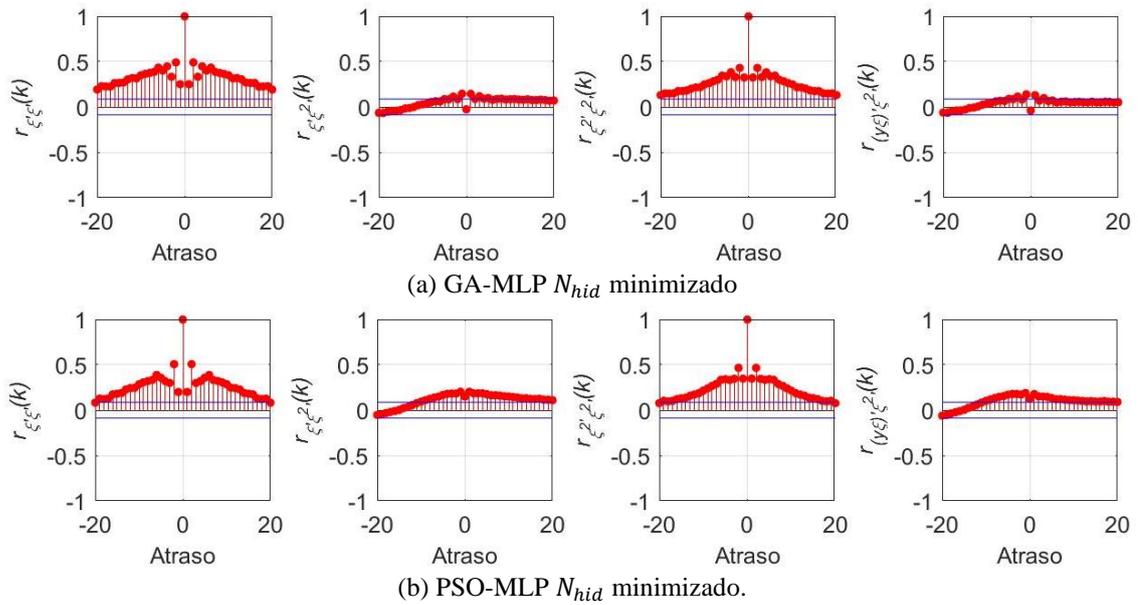


Figura 6.14 — Validação das redes GA-MLP e PSO-MLP com  $N_{hid}$  minimizado.



#### 6.4 O CUSTO DA OTIMIZAÇÃO

As ponderações acerca das limitações e de como o tempo de treinamento apresentado deve ser visto realizadas na Seção 5.3 são pertinentes na discussão que segue. Na Tabela 6.9, são mostrados o tempo de treinamento e o MSE de teste médios entre as repetidas execuções dos algoritmos.

Tabela 6.9 — Tempo de treinamento e MSE de teste médios.

| Algoritmo                     | $N_{hid}$           | MSE<br>[ $10^{-4}$ ] | Tempo<br>[s] | Algoritmo                     | $N_{hid}$            | MSE<br>[ $10^{-4}$ ] | Tempo<br>[s] |
|-------------------------------|---------------------|----------------------|--------------|-------------------------------|----------------------|----------------------|--------------|
| ELM                           | 5                   | 1,01                 | 0,016        | MLP                           | 5                    | 0,136                | 5167,1       |
|                               | 50                  | 0,11                 | 0,059        |                               | 20                   | 0,146                | 5848,6       |
| GA-ELM                        | 5                   | 0,16                 | 160,0        | GA-MLP                        | 5                    | 0,348                | 1970,8       |
|                               | 50                  | 0,10                 | 956,7        |                               | 20                   | 0,322                | 7118,5       |
| GA-ELM<br>com<br>minimização  | $N_{hid}^{med} = 6$ | 0,15                 | 314,1        | GA-MLP<br>com<br>minimização  | $N_{hid}^{med} = 23$ | 0,501                | 6452,2       |
| PSO-ELM                       | 5                   | 0,17                 | 128,1        | PSO-MLP                       | 5                    | 0,341                | 2520,1       |
|                               | 50                  | 0,10                 | 1218,2       |                               | 20                   | 0,335                | 6155,0       |
| PSO-ELM<br>com<br>minimização | $N_{hid}^{med} = 7$ | 0,13                 | 356,5        | PSO-MLP<br>com<br>minimização | $N_{hid}^{med} = 13$ | 0,401                | 5965,2       |

Fonte: próprio autor.

Apesar do elevado aumento no tempo de processamento (10000 ou mais vezes maior), as versões otimizadas da ELM conseguem produzir modelos mais eficientes com MSE comparável ao da ELM com  $N_{hid} = 50$ . Enquanto os modelos otimizados do MLP são menos eficazes e eficientes do que a rede treinada por retropropagação do erro: mesmo com  $N_{hid}$  maiores ou minimizados, apresentam maior MSE.

Para as mesmas quantidades de neurônios na cama oculta, as versões otimizadas da ELM são entre 6 e 32 vezes mais rápidas que o MLP (relação com intervalo mais elástico do que o da averiguada nos problemas de aproximação de função da Seção 5 porém ainda razoavelmente próxima). Por conta da quantidade elevada de amostras disponíveis para treinamento do banco de dados do laser, os algoritmos otimizados estão suscetíveis a variações mais bruscas no tempo de treinamento em decorrência de alguma mudança em um  $N_{hid}$  fixo.

Em consonância com a Seção 5.3, o tempo de treinamento e o MSE apresentam dependência de  $N_{hid}$  fixo, porém, quando este aumenta, o tempo é perceptível mais impactado (aumento significativo), enquanto o MSE reduz de forma mais paulatina. Isso pode ser observado explicitamente para a PSO-ELM, cuja variação de  $N_{hid} = 5$  para 50 chega a ter impacto linearmente proporcional no tempo: o tempo também é multiplicado por 10; enquanto isso, o MSE só reduz aproximadamente pela metade.

Por fim, a otimização nos modelos MLP não foram eficazes na identificação deste banco de dados: o MSE dos algoritmos otimizados é em torno de três vezes maiores para o GA-MLP e para o PSO-MLP. Possivelmente — e provavelmente — há uma configuração dos parâmetros de treinamento diferente ou alguma heurística a ser incluída nos algoritmos que provoque uma melhora e estes entreguem o resultado esperado de maior eficiência da rede.

## 6.5 CONSIDERAÇÕES E RESSALVAS

A otimização é um processo custoso e deve ser analisado com esmero. Para esta aplicação específica, enquanto para a ELM, pode-se dizer que houve vantagem significativa na eficiência (não necessariamente proporcional ao aumento do custo computacional), para o MLP, as estratégias e abordagens trabalhadas foram falhas em tornar o algoritmo mais eficiente.

À primeira vista, pode-se concluir — acertadamente — que um percentual de treino maior causaria algum impacto no treinamento das redes otimizadas e poderia agregar algum valor aos modelos. Cabe ressaltar que, se o MLP treinado por *backpropagation* consegue um desempenho superior e razoavelmente validado, independente do percentual ser 30% ou maior, os algoritmos de otimização para terem sido bem sucedidos necessitariam de um resultado ao menos compatível.

## 7 CONSIDERAÇÕES FINAIS

*“As convicções são inimigas mais perigosas da verdade do que as mentiras.”<sup>1</sup>*  
— Friedrich Wilhelm Nietzsche

Esta é a última seção do trabalho. Aqui são tecidos os comentários derradeiros a respeito de toda a abordagem realizada ao longo da dissertação, compilando o que os resultados permitem inferir, dentro das limitações adotadas na pesquisa, e realizando prospecções de sua contribuição e de que forma ela pode ser continuada posteriormente, não somente pelo autor, como também por qualquer outro membro da comunidade científica que dispor de interesse.

### 7.1 CONCLUSÃO

Em consonância com Carvalho (2017), as redes MLP e ELM treinadas por seus algoritmos tradicionais apresentam resultados satisfatórios e, pelo menos, razoavelmente validados pela análise de resíduos (esta última não desenvolvida no trabalho anterior). Esses dois algoritmos funcionam como balizadores para os algoritmos propostos neste estudo, de maneira tal que estes somente são considerados eficientes caso sejam mais eficientes que seus precursores sem otimização.

A hipótese inicial é de afirmativa para a pergunta “os modelos neurais otimizados são capazes de generalizar o comportamento do sistema com mais eficiência que seus pares treinados convencionalmente?”<sup>2</sup>. Parcialmente ela já havia sido respondida por outro estudo nosso (CARVALHO, 2018) para o treinamento otimizado por AG (com  $N_{hid}$  constante) da rede ELM, porém, não para as versões com  $N_{hid}$  variável pelo próprio algoritmo, ou para treinamento por PSO, ou para otimização da MLP.

Em Carvalho (2018), é apresentado que um AG é capaz de ajustar a camada oculta da ELM de tal maneira que esta consiga resultados melhores mesmo com menor quantidade de neurônios. Por consequência de sua própria definição, o algoritmo tradicional da ELM requer mais neurônios e tem desempenho crescente (mas não proporcional) com o aumento de  $N_{hid}$ .

Para a modelagem da potência óptica de lasers semicondutores QD, em especial, dos dados fornecidos por Thé (2010), uma GA-ELM com  $N_{hid} = 5$  é capaz de entregar resultados semelhantes ou superiores ao de uma ELM com  $N_{hid} = 20$ . Essa constatação obteve validação por análise de resíduos, além de gráfica pela saída estimada do modelo sendo comparada

---

<sup>1</sup> Frase retirada do livro *Human, All Too Human: A Book for Free Spirits*, de Nietzsche (NIETTSCHÉ; LEVY, 1924). Tradução nossa; original em alemão: “Überzeugungen sind gefährlichere Feinde der Wahrheit als Lügen”.

<sup>2</sup> Pergunta apresentada no objetivo geral (seção 1.3.1).

visualmente pela resposta esperada em pelo menos dois conjuntos de temperatura e estado (CARVALHO, 2018).

O presente trabalho amplia o estudo das seguintes formas:

- a) adota os cinco subconjuntos de dados em um mesmo modelo;
- b) define três abordagens para a otimização, uma com  $N_{hid}$  fixa e duas com variável (explicitamente minimizado ou não);
- c) acrescenta o algoritmo PSO para ajuste dos pesos da camada oculta;
- d) estende a análise ao MLP.

Para a otimização da rede ELM, nos três problemas estudados (os dois de aproximação de função e a identificação do laser), obteve-se modelo mais eficiente: no Problema 1 com cinco ou dez neurônios a GA-ELM e a PSO-ELM apresentam resultados melhores que a ELM correlata e iguais ou superiores aos da ELM com 20 e 50 neurônios. Mais: os algoritmos minimizadores de  $N_{hid}$  determinaram valor intermediário entre cinco e dez, em que se mantivesse um MSE satisfatório em uma rede mais eficiente.

De forma semelhante, comportaram-se a GA-ELM e a PSO-ELM para os Problema 2 e para a modelagem do laser. Outros impactos em destaque são:

- a) ELM otimizada perde a característica aleatória na camada oculta;
- b) suscetibilidade a *overffiting* para valores elevados de  $N_{hid}$  (à exceção do modelo com minimização de  $N_{hid}$ );
- c) incremento significativo no tempo de treinamento da rede (ordem de  $10^4$  vezes mais lento, variando em função de  $N_{hid}$ ,  $N_{pop}$  e  $N$ ).

A quantidade de amostras  $N_{amt}$  se mostrou nitidamente determinante na lentidão de treinamento quando variou de  $N_{amt} = 101$  ou  $961$  dos problemas de aproximação para as mais de 49 mil amostras de treinamento do laser (cerca de 30% do total de séries temporais).

Por outro lado, os modelos otimizados do MLP mostraram resultados menos unânimes. Ao GA-MLP, adicionou-se outra estratégia de elitismo para que se equiparasse aos demais e, ainda assim, apresentou resultados insatisfatórios e inferiores aos do PSO-MLP nas três abordagens da quantidade de neurônios no Problema 2 e na identificação do laser, principalmente, no primeiro. O PSO-MLP também obteve validação apenas pelos resíduos para o Problema 1.

Destaca-se que isso não representa a invalidação da hipótese inicial em um universo geral: a otimização do MLP apresentou falhas em tornar o algoritmo mais eficiente pelas

estratégias e abordagens trabalhadas. Não há fundamentos pelo presente estudo que levem a crer que não exista uma configuração dos parâmetros de treinamento diferente ou alguma heurística a ser incluída nos algoritmos que provoque uma melhora e estes entreguem o resultado esperado de maior eficiência da rede.

Em uma análise comparativa entre os resultados da ELM e do MLP otimizados, atribui uma complexidade maior a solução de um modelo otimizado do MLP e não inteiramente imprevisto a necessidade de estratégias mais contundentes, que podem ser aplicadas em posteriores estudos. A otimização é um processo de custo computacional razoável e deve ser avaliado o grau de interesse na redução de neurônios.

Ressalta-se novamente que o menor percentual de treino na identificação do laser (cerca de 30%) não é um empecilho definitivo para a otimização do MLP, uma vez que seu algoritmo tradicional por retropropagação do erro se apresentou satisfatório. Estudo futuro pode incluir algum método de redução mais eficiente do banco de dados do que a simples diminuição no percentual de dados para treino.

Sobre o banco de dados do laser em si, alguns destaques:

- a) os dados referentes a temperatura de 40°C e estado fundamental aparentemente são modeláveis mais facilmente;
- b) para 20°C/GS e 40°C/ES, as regiões de aclave em resposta ao degrau são as que mais apresentaram dificuldade de serem modeladas;
- c) o menor subconjunto de dados (20°C/ES) é suficiente pra modelagem das características apresentadas mesmo em percentuais de treino baixo.

## 7.2 LIMITAÇÕES

As heurísticas e os parâmetros de treinamento adotados para a otimização do MLP se apresentaram limitados para a aprendizagem da rede, sobretudo, para o GA-MLP. Outra característica, esta para todos os algoritmos otimizados abordados, apesar de alguns parâmetros do treinamento convencional das redes serem suprimidos (no caso do MLP, sobretudo), o ajuste inicial do AG e do PSO apresentam vários coeficientes, definidos, em geral, empiricamente.

O mal ajuste desses parâmetros em pequenas proporções têm impactos no treinamento das redes que podem impossibilitar completamente o aprendizado, sobretudo para o MLP, que depende exclusivamente do ajuste dos otimizadores — a camada de saída da ELM permanece sendo determinada pela pseudoinversa de Moore-Penrose.

As versões da ELM otimizadas são notoriamente menos rápidas na determinação dos pesos das camadas do que o algoritmo convencional — o grande atrativo da rede é exatamente

essa velocidade na execução não característica na maioria dos algoritmos de aprendizagem de máquina. Por outro lado, a ELM não permanece refém da aleatoriedade de seus pesos ocultos inicializados e mantidos aleatórios.

Outro destaque é que, mesmo a MLP tendo um algoritmo mais demorado, eventualmente, ela pode identificar algum sistema mais complexo modificando seus parâmetros de treinamento ou aumentando a quantidade de épocas, enquanto há casos em que a ELM não consegue, mesmo com o aumento indefinido de neurônios, como é mostrado para o Problema 2.

Por fim, mesmo que fosse a proposta do trabalho, utilizar percentuais de treinamento mais altos para o banco de dados do laser não seria factível pelo prazo da pesquisa e pelo poder de processamento disponível. Levando em conta somente o MLP com  $N_{hid} = 5$ , a qual se destaca não ser o algoritmo de maior custo computacional, gastaram-se, em média, 5167,1 segundos a cada execução, o que representa aproximadamente seis dias consecutivos do computador utilizado em dedicação, sem processos simultâneos não inerentes aos do próprio sistema operacional.

### 7.3 TRABALHOS FUTUROS

O estudo desenvolvido ainda tem uma gama abrangente de possibilidades de análise das redes neurais MLP e ELM com otimização na identificação do sistema dinâmico proposto. Várias técnicas de treinamento podem ser incorporadas aos algoritmos para otimizar a identificação e formulação de novos modelos. Algumas sugestões em trabalhos futuros são:

- a) buscar estratégias e novas configurações para os algoritmos GA-MLP e PSO-MLP em razão de torná-los aptos à otimização inicialmente proposta neste trabalho e somente parcialmente alcançada;
- b) aplicar métodos de redução de dados de entrada para o banco de dados do laser antes de realizar o treinamento das redes;
- c) ampliar o estudo de otimização para outras redes neurais (inclusive com ausência de supervisão) e para outros mecanismos de otimização, como os algoritmos de busca gravitacional ou baseados em colônias de formigas, ou mesmo incluir lógica nebulosa;
- d) utilizar outras técnicas de validação; entre outros.

## REFERÊNCIAS

- AGUIRRE, Luis Antonio. Introdução à identificação de sistemas: técnicas lineares e não-lineares aplicadas a sistemas reais. Belo Horizonte: UFMG, 2004.
- AHILA, R.; SADASIVAM, V.; MANIMALA, K. An integrated PSO for parameter determination and feature selection of ELM and its application in classification of power system disturbances. **Applied Soft Computing**, v. 32, p. 23-37, 2015.
- ALENCAR, Alisson S. C.; ROCHA NETO, Ajalmar R.; GOMES, João Paulo P. A new pruning method for extreme learning machines via genetic algorithms. **Applied Soft Computing**, v. 44, p. 101-107, 2016.
- ALEXANDRE, E.; CUADRA, L.; NIETO-BORGE, J. C.; CANDIL-GARCIA, G.; DEL PINO, M.; SALCEDO-SANZ, S. A hybrid genetic algorithm: extreme learning machine approach for accurate significant wave height reconstruction. **Ocean Modelling**, v. 92, p. 115-123, 2015.
- ALTOBI, M. A. S.; BEVAN, G.; WALLACE, P.; HARRISON, D.; RAMACHANDRAN, K. P. Fault diagnosis of a centrifugal pump using MLP-GABP and SVM with CWT. **Engineering Science and Technology, an International Journal**, 2019.
- AYALA, Helon Vicente Hultmann; RAKOTONDRAHE, Micky; COELHO, Leandro dos Santos. Modeling of a 2-DOF Piezoelectric Micromanipulator at High Frequency Rates through Nonlinear Black-box System Identification. In: **2018 Annual American Control Conference (ACC)**. IEEE, 2018. p. 4354-4359.
- AYBAR-RUIZ, A.; JIMÉNEZ-FERNÁNDEZ, S.; CORNEJO-BUENO, L.; CASANOVA-MATEO, C.; SANZ-JUSTO, J.; SALVADOR-GONZÁLEZ, P.; SALCEDO-SANZ, S. A novel grouping genetic algorithm–extreme learning machine approach for global solar radiation prediction from numerical weather models inputs. **Solar Energy**, v. 132, p. 129-142, 2016.
- BARATA, João Carlos Alves; HUSSEIN, Mahir Saleh. The Moore–Penrose pseudoinverse: A tutorial review of the theory. **Brazilian Journal of Physics**, v. 42, n. 1-2, p. 146-165, 2012.
- BARRETO, Guilherme A.; SOUZA, Luís Gustavo M. Novel approaches for parameter estimation of local linear models for dynamical system identification. **Applied Intelligence**, v. 44, n. 1, p. 149-165, 2016.
- BARRETO, J. M. **Introdução às redes neurais artificiais**. V Escola Regional de Informática. Sociedade Brasileira de Computação, Regional Sul, Santa Maria, Florianópolis: 2002.
- BILLINGS, S. A.; TAO, Q. H. Model validity tests for non-linear signal processing applications. **International Journal of Control**, v. 54, n. 1, p. 157-194, 1991.
- BILLINGS, S. A.; VOON, W. S. F. A prediction-error and stepwise-regression estimation algorithm for non-linear systems. **International Journal of Control**, v. 44, n. 3, p. 803-822, 1986.
- BILLINGS, S. A.; VOON, W. S. F. Structure detection and model validity tests in the identification of nonlinear systems. In: **IEE Proceedings D-Control Theory and Applications**. IET, 1983. p. 193-199.
- BILLINGS, S. A.; ZHU, Q. M. Model validation tests for multivariable nonlinear models including neural networks. **International Journal of Control**, v. 62, n. 4, p. 749-766, 1995.

BINKOWSKI, C. **Síntese das técnicas de identificação de sistemas não lineares: estruturas de modelo de Hammerstein-Wiener e NARMAX**. Dissertação (Engenharia Elétrica). São Leopoldo: Universidade do Vale do Rio dos Sinos (UNISINOS), 2016.

BJORK, Gunnar; YAMAMOTO, Yoshihisa. **Analysis of semiconductor microcavity lasers using rate equations**. IEEE Journal of Quantum Electronics, v. 27, n. 11, p. 2386-2396, 1991.

BRAGA, A. de P.; CARVALHO, A. C. P. L. F.; LUDERMIR, Teresa Bernarda. **Redes neurais artificiais: teoria e aplicações**. Rio de Janeiro: LTC, 2000.

CAMARGO, Gilberto de Menezes. Controle da pressão seletiva em algoritmo genético aplicado a otimização de demanda em infraestrutura aeronáutica. Tese de Doutorado. São Paulo: USP, 2006.

CARVALHO, R. R. M. Aplicação de redes neurais supervisionadas para modelagem de lasers quânticos. Monografia (engenharia elétrica). Teresina: UFPI, 2017.

CARVALHO, R. R. M.; SOUZA, L. G. M.; THÉ, G. A. P. Mapeamento da dinâmica de laser semiconductor por meio de sistema híbrido neuro-evolutivo. In: **XXII Congresso Brasileiro de Automática**. João Pessoa, 2018.

CHATTERJEE, S.; HORE, S.; DEY, N.; CHAKRABORTY, S.; ASHOUR, A. S. Dengue fever classification using gene expression data: a PSO based artificial neural network approach. In: **Proceedings of the 5th international conference on frontiers in intelligent computing: theory and applications**. Springer, Singapore, 2017. p. 331-341.

CHO, Al Y.; ARTHUR, J. R. Molecular beam epitaxy. **Progress in solid state chemistry**, v. 10, p. 157-191, 1975.

CHRISTOU, V.; TSIPOURAS, M. G.; GIANNAKEAS, N.; TZALLAS, A. T. Hybrid extreme learning machine approach for homogeneous neural networks. **Neurocomputing**, v. 311, p. 397-412, 2018.

COELHO, A. A. R.; COELHO, L. dos S. **Identificação de sistemas dinâmicos lineares**. [S.l.]: UFSC, 2004.

CORRÊA, Marcelo V.; AGUIRRE, Luis A. Identificação não-linear caixa-cinza: uma revisão e novos resultados. **SBA: Controle & Automação Sociedade Brasileira de Automatica**, v. 15, n. 2, pp. 109-126, 2004.

DA CRUZ, L. F.; FREIRE, R. Z.; REYNOSO-MEZA, G.; PINTO, F. B.; COELHO, L. D. S. RBF Neural Network combined with self-adaptive MODE and Genetic Algorithm to identify velocity profile of swimmers. In: **2016 IEEE Symposium Series on Computational Intelligence (SSCI)**. IEEE, 2016. p. 1-7.

DANTAS, A. D. O. da S. Identificação de modelos polinomiais NARX utilizando algoritmos combinados de detecção de estrutura e estimação de parâmetros com aplicações práticas. Dissertação de Mestrado em Engenharia Elétrica e de Computação. Natal: UFRN, 2013.

DIAS, Diego Borelli. **Propriedades ópticas de pontos quânticos semicondutores de InAs/GaAs**. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná, 2011.

FAIST, J.; SIRTORI, C.; HOFSTETTER, D. **Quantum cascade laser**. Science-AAAS-Weekly Paper Edition-including Guide to Scientific Information, v. 264, n. 5158, p. 553-555, 1994.

GARCIA, Jerry. **Interview with Jerry Garcia (& Bear)** [jun. 1990]. Entrevistador: Howard Rheingold. 37 min. [S.l.: s.n.], 1990.

GHADI, H.; SINGH, P. K.; DUBEY, S.; BHATT, M.; CHAKRABARTI, S. Co-relation of theoretical simulation with experimental results for InAs quantum-dot heterostructures with different capping material. In: **Quantum Dots and Nanostructures: Growth, Characterization, and Modeling XV**. International Society for Optics and Photonics, 2018. p. 105430R.

GIES, Christopher; WIERSIG, Jan; LORKE, Michael; JAHNKE, Frank. **Semiconductor model for quantum-dot-based microcavity lasers**. *Physical Review A*, v. 75. 2007.

GOLDBERG, David E. **The design of innovation: lessons from and for competent genetic algorithms**. Springer Science & Business Media, 2013.

GRITTI, M., AYALA; H. V. H.; MARIANI, V. C.; COELHO, L. Abordagem de computação natural aplicada à identificação de um processo fan and plate. **XII Simpósio Brasileiro de Automação Inteligente**. Natal, RN: SBA, 2015.

GUAN, Xing Chen; ZHAO, Dong Ya; ZHU, Quan Min. NARMAX modelling and U-model control design for continuous stirred tank reactor (CSTR). In: **2016 35th Chinese Control Conference (CCC)**. IEEE, 2016. p. 1964-1969.

HAUG, H. Quantum-mechanical rate equations for semiconductor lasers. *Physical Review*, v. 184, n. 2, p. 338, 1969.

HAYKIN, Simon. **Redes neurais: princípios e prática**. trad. Paulo Martins Engel. 2. ed. Porto Alegre: Bookman, 2001.

HE, F.; SARRIGIANNIS, P. G.; BILLINGS, S. A.; WEI, H.; ROWE, J.; ROMANOWSKI, C.; HOGGARD, N.; HADJIVASSILLIOU, M.; RAO, D. G.; GRÜNEWALD, R.; KHAN, A.; YIANNI, J. Nonlinear interactions in the thalamocortical loop in essential tremor: a model-based frequency domain analysis. *Neuroscience*, v. 324, p. 377-389, 2016.

HE, F.; WEI, H. L.; BILLINGS, S. A.; SARRIGIANNIS, P. G. A nonlinear generalization of spectral granger causality. *IEEE Transactions on Biomedical Engineering*, v. 61, n. 6, p. 1693-1701, 2014.

HODGE, Victoria; AUSTIN, Jim. A survey of outlier detection methodologies. *Artificial intelligence review*, v. 22, n. 2, p. 85-126, 2004.

HOLLAND, J. H. An efficient genetic algorithm for the traveling salesman problem. *European Journal of Operational Research*, v. 145, p. 606-617, 1975.

HUANG, G.; HUANG, G. B.; SONG, S.; YOU, K. Trends in extreme learning machines: a review. *Neural Networks*, v. 61, p. 32-48, 2015.

HUANG, Guang-Bin; WANG, Dian Hui; LAN, Yuan. **Extreme learning machines: a survey**. *International Journal of Machine Learning and Cybernetics*, v. 2, n. 2, p. 107-122, 2011.

HUANG, Guang-Bin; ZHOU, Hongming; DING, Xiaojian; ZHANG, Rui. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, v. 42, n. 2, p. 513-529, 2012.

HUANG, Guang-Bin; ZHU, Qin-Yu; SIEW, Chee-Kheong. Extreme learning machine: a new learning scheme of feedforward neural networks. In: *IEEE International Joint Conference on Neural Networks*. IEEE, 2004. p. 985-990.

HUANG, Guang-Bin; ZHU, Qin-Yu; SIEW, Chee-Kheong. **Extreme learning machine: theory and applications**. Neurocomputing, v. 70, n. 1, p. 489-501, 2006.

INAOKA, H.; KOBAYASHI, K.; NEBUYA, S.; KUMAGAI, H.; TSURUTA, H.; FUKUOKA, Y. Derivation of NARX models by expanding activation functions in neural networks. **IEEE Transactions on Electrical and Electronic Engineering**, 2019.

JONG, Kenneth Alan De. Analysis of the behavior of a class of genetic adaptive systems. Dissertação. University of Michigan: Ann Arbor, 1975.

JUNG, Hyuna; KUM, Hyun; HWANG, Jinyoung. Band Alignment Transition from Type I to Type II of InP/In 0.48 Ga 0.52 P quantum Dots. In: **TENCON 2018-2018 IEEE Region 10 Conference**. IEEE, 2018. p. 1045-1048.

KANUNGO, D. P.; NAIK, B.; NAYAK, J.; BABOO, S.; BEHERA, H. S. An improved PSO based back propagation learning-MLP (IPSO-BP-MLP) for classification. In: **Computational Intelligence in Data Mining-Volume 1**. Springer, New Delhi, 2015. p. 333-344.

KAR, B. P.; NAYAK, S. K.; NAYAK, S. C. Opposition-Based GA Learning of Artificial Neural Networks for Financial Time Series Forecasting. In: **Computational Intelligence in Data Mining—Volume 2**. Springer, New Delhi, 2016. p. 405-414.

LIMA, A. A. M.; ARAUJO, R. M.; DOS SANTOS, F. A. G.; YOSHIZUMI, V. H.; DE BARROS, F. K.; SPATTI, D. H.; LIBONI, L. H. B.; DAJER, M. E. Classification of Hand Movements from EMG Signals using Optimized MLP. In: **2018 International Joint Conference on Neural Networks (IJCNN)**. IEEE, 2018. p. 1-7.

LUO, W.; ZHANG, Z.; WEN, T.; LI, C.; LUO, Z. Features extraction and multi-classification of sEMG using a GPU-Accelerated GA/MLP hybrid algorithm. **Journal of X-ray Science and Technology**, v. 25, n. 2, p. 273-286, 2017.

MAL, I.; PANDA, D. P.; TONGBRAM, B.; SAMAJDAR, D. P.; CHAKRABARTI, S. Analytical modeling of temperature and power dependent photoluminescence (PL) spectra of InAs/GaAs quantum dots. **Journal of Applied Physics**, v. 124, n. 14, p. 145701, 2018.

MOORE, Eliakim H. On the reciprocal of the general algebraic matrix. **Bull. Am. Math. Soc.**, v. 26, p. 394-395, 1920.

NELLES, Oliver. Nonlinear system identification: from classical approaches to neural networks and fuzzy models. **Springer Science & Business Media**, 2013.

NIETZSCHE, Friedrich Wilhelm; LEVY, Oscar. **Human, all too human: A book for free spirits**. Macmillan, 1924.

OGATA, K. **Engenharia de Controle Moderno**. 5. ed. Boston: Pearson, 2010.

PENROSE, Roger. A generalized inverse for matrices. In: Mathematical proceedings of the Cambridge philosophical society. **Cambridge University Press**, 1955. p. 406-413.

- POUR, Najmeh Kamyab. **Energy efficiency in wireless sensor networks**. Tese de doutorado. Sydney: Faculty of Engineering and Information Technology at The University of Technology Sydney, 2015.
- PRICE, J.; GOBLE, T. Signals and noise. In: MAZDA, Fraidoon (Ed.). **Telecommunications engineer's reference book**. Butterworth-Heinemann, 2014.
- RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Cognitive modeling**, v. 5, n. 3, p. 1, 1986.
- RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. **California Univ San Diego La Jolla Inst for Cognitive Science**, 1985.
- SAAD, Z.; OSMAN, M. K.; MASHOR, M. Y. Modelling and Forecasting of Car Speed Using Detrended Inputs and Hybrid Multilayered Perceptron Network. **Contemporary Engineering Sciences**, v. 7, n. 13, p. 603-610, 2014.
- SANTOS, H. G. da S. **Estudo das aplicações de modelos lineares locais baseados na rede Kohonen para modelagem de lasers quânticos semicondutores**. Monografia (engenharia elétrica). Teresina: UFPI, 2017.
- SANTOS, W. F. da S. **Predição da Dinâmica de Lasers de Pontos Quânticos Semicondutores Utilizando Redes Neurais Artificiais**. Dissertação de mestrado. UFBA: Salvador, 2018.
- SHCHERBAKOV, M. V.; BREBELS, A.; SHCHERBAKOVA, N. L.; TYUKOV, A. P.; JANOVSKY, T. A.; KAMAEV, V. A. E. A survey of forecast error measures. **World Applied Sciences Journal**, v. 24, n. 24, p. 171-176, 2013.
- SIMONDI-TEISSEIRE, B.; VIANA, B.; VIVIEN, D.; LEJUS, A. M. Yb<sup>3+</sup> to Er<sup>3+</sup> energy transfer and rate-equations formalism in the eye safe laser material Yb: Er: Ca<sub>2</sub>Al<sub>2</sub>SiO<sub>7</sub>. **Optical Materials**, v. 6, n. 4, p. 267-274, 1996.
- SOLARES, J. R. A.; WEI, H.-L.; BILLINGS, S. A. A novel logistic-NARX model as a classifier for dynamic binary classification. **Neural Computing and Applications**, v. 31, n. 1, p. 11-25, 2017.
- SOUZA, L. G. M. Modelos lineares locais para identificação de sistemas dinâmicos usando redes neurais competitivas. Tese de doutorado. Fortaleza: UFCE, 2012.
- TAMURA, S.; TATEISHI, M. Capabilities of a four-layered feedforward neural network: four layers versus three. **IEEE Transactions on Neural Networks**, v. 8, n. 2, p. 251-255, 1997.
- THÉ, G. A. P. **Improved modeling and simulation of quantum dot lasers**. Tese de doutorado. Politecnico di Torino, 2010.
- TYUKOV, A.; BREBELS, A.; SHCHERBAKOV, M. Automatic two way synchronization between server and multiple clients for HVAC system. In: **Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services**. ACM, 2011. p. 467-470.
- VALADARES, E. de C.; ALVES, E. G.; CHAVES, A. S. **Aplicações da física quântica do transistor à nanotecnologia**. Editora Livraria da Física, 2005.

- VAPNIK, V. N.; BOTTOU, L. Stochastic learning. In: **Advanced Lectures on Machine Learning**, LNAI 3176. Springer Verlag, 2004. p. 146-168.
- WANG, G.-C.; LU, T.-M. RHEED transmission mode and pole figures: thin film and nanostructure texture analysis. **Springer Science & Business Media**, 2013.
- WELCH, R. L.; RUFFING, S. M.; VENAYAGAMOORTHY, G. K. Comparison of feedforward and feedback neural network architectures for short term wind speed prediction. In: **2009 International Joint Conference on Neural Networks**. IEEE, 2009. p. 3335-3340.
- XIONG, Y.; ZHANG, X. An accurate method of modeling self-assembled InAs/InGaAsP/InP (001) quantum dot with double-capping procedure. **IEEE Journal of Quantum Electronics**, v. 53, n. 6, p. 1-11, 2017.
- XU, Y.; SHU, Y. Evolutionary extreme learning machine-based on particle swarm optimization. In: **International Symposium on Neural Networks**. Springer, Berlin, Heidelberg, 2006. p. 644-652.
- YANG, H.; YI, J.; ZHAO, J.; DONG, Z. Extreme learning machine based genetic algorithm and its application in power system economic dispatch. **Neurocomputing**, v. 102, p. 154-162, 2013.
- YANG, Y.; WANG, Y.; YUAN, X. Bidirectional extreme learning machine for regression problem and its learning effectiveness. **IEEE Transactions on Neural Networks and Learning Systems**, v. 23, n. 9, p. 1498–1505, 2012.
- ZAKARIA, M. Z.; MANSOR, Z.; NOR, A. M.; SAAD, M. S., BAHARUDIN, M. E.; AHMAD, R. NARMAX Model Identification Using Multi-Objective Optimization Differential Evolution. **International Journal of Integrated Engineering**, v. 10, n. 7, 2018.
- ZHANG, Ying; MEI, L. I. A novel evaluation model of water quality based on PSO-ELM method. **Environmental Science & Technology**, n. 5, p. 26, 2016.
- ZHU, Q. Y.; QIN, A. K.; SUGANTHAN, P. N.; HUANG, G. B. Evolutionary extreme learning machine. **Pattern recognition**, v. 38, n. 10, p. 1759-1763, 2005.
- ZHU, Q.; WANG, Y.; ZHAO, D.; LI, S.; BILLINGS, S. A. Review of rational (total) nonlinear dynamic system modelling, identification, and control. **International journal of systems science**, v. 46, n. 12, p. 2122-2133, 2015.