

Universidade Federal de Minas Gerais
Programa de Pós-Graduação em Engenharia Elétrica

**DIAGNÓSTICO DE FALHAS BASEADO EM SISTEMA
INTELIGENTE EVOLUTIVO**

Maurílio José Inácio

TESE DE DOUTORADO N° 195

**DIAGNÓSTICO DE FALHAS BASEADO EM SISTEMA
INTELIGENTE EVOLUTIVO**

Maurílio José Inácio

DATA DA DEFESA: 01/12/2014

Universidade Federal de Minas Gerais
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

**DIAGNÓSTICO DE FALHAS BASEADO EM SISTEMA
INTELIGENTE EVOLUTIVO**

Maurílio José Inácio

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Orientador: Prof. Walmir Matos Caminhas

Belo Horizonte - MG

Dezembro de 2014

"Diagnóstico de Falhas Baseado em Sistema Inteligente Evolutivo"

Maurilio Jose Inacio

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Aprovada em 01 de dezembro de 2014.

Por:



Prof. Dr. Walmir Matos Caminhas
DELT (UFMG) - Orientador



Prof. Dr. André Paim Lemos
DELT (UFMG) - Coorientador



Prof. Dr. Benjamim Rodrigues de Menezes
DELT (UFMG)



Prof. Dr. Fernando Antonio Campos Gomide
DCA (UNICAMP)



Prof. Dr. Ivan Nunes da Silva
EESC (USP-SC)



Prof. Dr. Pyramo Pires da Costa Jr.
DEE (PUC-MG)



Prof. Dr. Fernando de Oliveira Souza
DELT (UFMG)

Dedico este trabalho à minha família, meus pais, meus irmãos, minha esposa Deniz, meu filho Matheus e minha filha Ana Luiza.

Agradecimentos

A DEUS, por me conceder os bens mais preciosos que poderíamos ter, vida e saúde, e com eles a capacidade para superar nossos desafios e realizar nossos sonhos.

Ao Prof. Walmir Caminhas, pela orientação, a qual permitiu a realização desse trabalho, pelo apoio e incentivo, desde os tempos do mestrado, a continuar meus estudos visando alcançar o doutorado. E principalmente pela amizade de sempre e pelos bons momentos que compartilhamos.

Ao Prof. André Lemos, pela co-orientação, que muito contribuiu para a realização desse trabalho. E também pela amizade e companheirismo ao longo dos meus anos de estudo no CPDEE.

A todos os professores do CPDEE com os quais convivi, pelos muitos conhecimentos compartilhados e pelos ensinamentos transmitidos durante as aulas, que foram de fundamental importância para minha formação acadêmica.

Aos colegas do DIFCOM (Laboratório de Detecção de Falhas, Controle, Otimização e Modelagem) da UFMG, pela amizade, pelo companheirismo e pelos agradáveis momentos que passamos juntos. Em especial aos colegas Renato, Alisson e Guilherme, pelas muitas contribuições, as quais me ajudaram a superar dificuldades.

Aos meus colegas da FEMC (Fundação Educacional Montes Claros) e da FACIT (Faculdade de Ciência e Tecnologia de Montes Claros), pelo apoio e motivação. Em especial aos colegas Ângela, Gisele, Haroldo, Renato, Leonardo, Paulo e Murilo, pela amizade ao longo dos muitos anos de trabalho.

À minha família, meus pais e irmãos, pelo apoio e incentivo incondicional para realização dos meus objetivos. Em especial à minha esposa Deniz, meu filho Matheus e minha filha Ana Luiza, pelo amor, pela compreensão e pela paciência nos inúmeros momentos de ausência, sem os quais eu jamais conseguiria completar essa difícil e longa jornada.

“Os computadores são incrivelmente rápidos, precisos e burros; os homens são incrivelmente lentos, imprecisos e brilhantes; juntos, seus poderes ultrapassam os limites da imaginação”.

Albert Einstein

Resumo

Em muitas áreas, os avanços tecnológicos levaram ao surgimento de máquinas e equipamentos complexos, dificultando seu gerenciamento e manutenção e, conseqüentemente, ocasionando o aumento de falhas. Nesse contexto, esse trabalho propõe uma metodologia de diagnóstico de falhas baseada em sistema inteligente evolutivo, visando sua aplicação em sistemas dinâmicos não estacionários que exijam a realização dessas tarefas em modo *on-line* e em tempo real. Sistemas inteligentes evolutivos são sistemas capazes de determinar gradualmente tanto a sua estrutura quanto os seus parâmetros, extraindo conhecimento a partir dos dados de entrada. De acordo com o conceito de manutenção inteligente, a aplicação de sistemas inteligentes evolutivos para realizar diagnóstico de falhas tem-se mostrado promissora. Nesse trabalho é proposto um classificador *fuzzy* evolutivo para aplicação em diagnóstico de falhas, baseado em uma nova abordagem que combina um algoritmo de agrupamento recursivo e um método de detecção de *drift*. Essa abordagem confere ao classificador *fuzzy* evolutivo a capacidade de aprendizado contínuo e incremental em modo *on-line*, sem a necessidade do conhecimento prévio do sistema dinâmico em questão, e ainda proporcionando maior robustez a *outliers* e ruído presentes nos dados de entrada. O classificador proposto nesse trabalho é avaliado em dois problemas tipicamente encontrados na indústria, que são o diagnóstico de falhas em sistema de acionamento de uma máquina de corrente contínua, e o diagnóstico de falhas em sistema de tanques interativos. Os resultados das simulações realizadas demonstraram que o classificador proposto tem um desempenho promissor, sugerindo-o como uma alternativa viável para aplicação em problemas reais de diagnóstico de falhas.

Palavras-chave: Diagnóstico de Falhas, Sistemas Evolutivos, Classificador *fuzzy*.

Abstract

In many areas, the advance of technology has resulted in the emergence of machinery and complex equipments, which impose great challenges for its management and maintenance, and consequently, increase the fault occurrences. In this context, this work proposes a methodology for fault diagnosis based on evolving intelligent system aiming at its application in non-stationary dynamic systems which require performing these tasks in online mode and real time. Evolving intelligent systems are systems capable to gradually determine both its structure and its parameters, by extracting knowledge from input data. According to intelligent maintenance concept, the application of evolving intelligent systems to perform fault diagnosis has shown promising. In this work we propose an evolving fuzzy classifier for fault diagnosis application, based on a new approach that combines a recursive clustering algorithm and a drift detection method. This approach gives the evolving fuzzy classifier the ability of continuous and incremental learning in online mode, without need a prior knowledge about of the dynamic system in question, and providing greater robustness to outliers and noise present in input data. The classifier proposed in this work is evaluated on two problems typically found in the industry, which are fault diagnosis in DC drive system, and fault diagnosis in interactive tanks system. The results of simulations demonstrate that the proposed classifier has promising performance, suggesting it as a feasible alternative for application to real problems of fault diagnosis.

Keywords: Fault diagnosis, Evolving Systems, *Fuzzy* Classifier.

Sumário

Sumário	vii
Lista de Figuras	xi
Lista de Tabelas	xiii
Lista de Acrônimos	xvi
1 Introdução	1
1.1 Contexto e Motivação	1
1.2 Objetivos	3
1.3 Contribuições da Tese	4
1.3.1 Trabalhos em Periódicos	5
1.3.2 Capítulo de Livro	5
1.3.3 Trabalhos em Congressos Nacionais	5
1.3.4 Trabalho em Congresso Internacional	6
1.4 Organização do texto	6
Lista de Símbolos	1
2 Diagnóstico de Falhas	8
2.1 Introdução	8
2.2 Definições	9

2.3	Histórico	10
2.4	Sistema de Diagnóstico de Falhas	11
2.5	Métodos de Diagnóstico de Falhas	14
3	Sistemas Inteligentes Evolutivos	16
3.1	Introdução	16
3.2	Conceitos	17
3.3	Histórico	19
3.4	Modelos	23
3.5	Aplicações	38
4	Algoritmo de Agrupamento Recursivo e Detecção de <i>Drift</i>	45
4.1	Introdução	45
4.2	Algoritmo de Agrupamento Recursivo	46
4.3	Detecção de <i>Drift</i>	50
4.4	Algoritmo proposto	52
5	Classificador <i>Fuzzy</i> Evolutivo	61
5.1	Introdução	61
5.2	Classificador <i>Fuzzy</i>	62
5.3	Classificador proposto	64
5.4	Aplicação do classificador proposto em diagnóstico de falhas . .	65
6	Resultados Numéricos	78
6.1	Introdução	78
6.2	Diagnóstico de falhas em sistema de acionamento C.C.	80
6.3	Diagnóstico de falhas em sistema de tanques interativos	95
6.3.1	Resultados de simulações	97
6.3.2	Validação do classificador	107
7	Conclusão	117
	Referências bibliográficas	132

Lista de Figuras

2.1	Aplicação de sistema de diagnóstico de falhas em processos. . .	11
3.1	Representação da estrutura dos sistemas inteligentes evolutivos.	19
3.2	Número de publicações sobre sistemas inteligentes evolutivos nos últimos anos.	23
4.1	Agrupamento de dados.	58
4.2	Identificação do primeiro grupo de dados.	58
4.3	Atualização dos parâmetros do primeiro grupo de dados. . . .	58
4.4	Primeiro grupo de <i>Outliers</i> não identificados como um grupo. .	59
4.5	Identificação e atualização dos parâmetros do segundo grupo de dados.	59
4.6	Segundo grupo de <i>Outliers</i> não identificados como um grupo. . .	59
4.7	Identificação e atualização dos parâmetros do terceiro grupo de dados.	59
4.8	Terceiro grupo de <i>Outliers</i> não identificados como um grupo. . .	60
5.1	Diagnóstico de falhas com Classificador Fuzzy Evolutivo.	67
5.2	Aplicativo desenvolvido para demonstração de funcionamento do classificador <i>fuzzy</i> evolutivo.	69
5.3	Deteção de nova classe e criação da primeira regra com rótulo da classe fornecido pelo operador.	70
5.4	Atualização dos parâmetros da primeira regra.	70
5.5	Deteção de nova classe e criação da segunda regra com rótulo da classe fornecido pelo operador.	71
5.6	Atualização dos parâmetros da segunda regra.	71

5.7	Detecção e criação da terceira regra com rótulo da classe fornecido pelo operador.	72
5.8	Atualização dos parâmetros da terceira regra.	72
5.9	Detecção de nova classe e criação da quarta regra com rótulo da classe fornecido pelo operador.	73
5.10	Atualização dos parâmetros da quarta regra.	73
5.11	Detecção de nova classe e criação da quinta regra com rótulo da classe fornecido pelo operador.	74
5.12	Atualização dos parâmetros da quinta regra.	74
5.13	Amostras de dados da classe 0 classificadas automaticamente. .	75
5.14	Amostras de dados da classe 3 classificadas automaticamente. .	75
5.15	Amostras de dados da classe 1 classificadas automaticamente. .	76
5.16	Amostras de dados da classe 4 classificadas automaticamente. .	76
5.17	Amostras de dados da classe 2 classificadas automaticamente. .	77
6.1	Ilustração dos tempos de detecção e isolamento da falha.	81
6.2	Representação do Sistema de Acionamento C.C.	81
6.3	Simulação de falha: abertura da bobina de armadura.	84
6.4	Simulação de falha: curto circuito no conversor de campo. . . .	84
6.5	Simulação de falha: curto circuito entre as espiras da bobina da armadura.	85
6.6	Simulação de falha: falha de lubrificação dos mancais.	85
6.7	Simulação de falha: falha no sensor de corrente de campo. . . .	86
6.8	Simulação das falhas vistas em espaço 3D.	86
6.9	Saída desejada e saída estimada pelo classificador em cenário de 3 falhas simuladas no sistema de acionamento C.C.	88
6.10	Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de acionamento C.C.	89
6.11	Saída desejada e saída estimada pelo classificador em cenário de 7 falhas simuladas no sistema de acionamento C.C.	89
6.12	Saída desejada e saída estimada pelo classificador em cenário de 9 falhas simuladas no sistema de acionamento C.C.	90
6.13	Saída desejada e saída estimada pelo classificador em cenário de 11 falhas simuladas no sistema de acionamento C.C.	90

6.14	Simulação das falhas no sistema de acionamento C.C. com presença de <i>outliers</i>	93
6.15	Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de acionamento C.C. com presença de <i>outliers</i>	93
6.16	Diagrama em blocos do sistema de tanques interativos.	95
6.17	Fluxo de massa do sistema de tanques interativos.	96
6.18	Simulação de falha: agarramento da válvula FCV-1.	99
6.19	Simulação de falha: desligamento da bomba BA-2.	99
6.20	Simulação de falha: entupimento no duto entre TQ-2 e TQ-3.	100
6.21	Simulação de falha: vazamento no duto entre TQ-2 e TQ-3.	100
6.22	Simulação de falha: falha no sensor de nível de TQ-3.	101
6.23	Saída desejada e saída estimada pelo classificador em cenário de 3 falhas simuladas no sistema de tanques interativos.	103
6.24	Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de tanques interativos.	103
6.25	Saída desejada e saída estimada pelo classificador em cenário de 7 falhas simuladas no sistema de tanques interativos.	104
6.26	Saída desejada e saída estimada pelo classificador em cenário de 9 falhas simuladas no sistema de tanques interativos.	104
6.27	Saída desejada e saída estimada pelo classificador em cenário de 11 falhas simuladas no sistema de tanques interativos.	105
6.28	Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de tanques interativos com presença de <i>outliers</i>	108
6.29	Falha na planta real: agarramento da válvula FCV-1.	110
6.30	Falha na planta real: desligamento da bomba BA-2.	111
6.31	Falha na planta real: entupimento no duto entre TQ-2 e TQ-3.	111
6.32	Falha na planta real: vazamento no duto entre TQ-2 e TQ-3.	112
6.33	Falha na planta real: falha no sensor de nível de TQ-3.	112
6.34	Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 3 falhas no sistema de tanques interativos.	113
6.35	Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 5 falhas no sistema de tanques interativos.	114

6.36	Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 7 falhas no sistema de tanques interativos.	114
6.37	Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 9 falhas no sistema de tanques interativos.	115
6.38	Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 11 falhas no sistema de tanques interativos.	115

Lista de Tabelas

6.1	Matriz de decisão para avaliação da detecção de falhas	79
6.2	Tipos de falhas no sistema de acionamento C.C.	83
6.3	Desempenho na detecção de falhas simuladas no sistema de acionamento C.C.	91
6.4	Desempenho na classificação de falhas simuladas no sistema de acionamento C.C.	92
6.5	Tempos de detecção e classificação por tipo de falha simulada no sistema de acionamento C.C.	92
6.6	Desempenho na detecção de falhas simuladas no sistema de acionamento C.C. com presença de <i>outliers</i>	94
6.7	Desempenho na classificação de falhas simuladas no sistema de acionamento C.C. com presença de <i>outliers</i>	94
6.8	Tipos de falhas no sistema de tanques interativos	98
6.9	Desempenho na detecção de falhas simuladas no sistema de tanques interativos.	106
6.10	Desempenho na classificação de falhas simuladas no sistema de tanques interativos.	106
6.11	Tempos de detecção e classificação por tipo de falha simulada no sistema de tanques interativos.	107
6.12	Desempenho na detecção de falhas simuladas no sistema de tanques interativos com presença de <i>outliers</i>	108
6.13	Desempenho na classificação de falhas simuladas no sistema de tanques interativos com presença de <i>outliers</i>	108
6.14	Desempenho na validação de detecção de falhas do sistema de tanques interativos.	116

6.15	Desempenho na validação de classificação de falhas do sistema de tanques interativos.	116
------	--	-----

Lista de Acrônimos

ACC	-	<i>Accuracy</i>
CART	-	<i>Classification and Regression Tress</i>
CBM	-	<i>Condition-Based Maintenance</i>
DDM	-	<i>Drift Detection Method</i>
DDT	-	<i>Detection Delay Time</i>
DENFIS-		<i>Dynamic Evolving Neuro-Fuzzy Inference Systems</i>
ECM	-	<i>Evolving Clustering Method</i>
ECOS	-	<i>Evolving Connectionist Systems</i>
EFuNN	-	<i>Evolving Fuzzy Neural Network</i>
eGKL	-	<i>evolving GK-like</i>
eMG	-	<i>Multivariable Gaussian Evolving Fuzzy Modeling</i>
eNFN	-	<i>Evolving Neo-Fuzzy Neural Network</i>
ePL	-	<i>Evolving Fuzzy Participatory Learning</i>
eR	-	<i>Evolving Rule-Base Models</i>
ESOM	-	<i>Evolving Self-Organising Map</i>
eTS	-	<i>Evolving Takagi-Sugeno Models</i>
eVQ	-	<i>Evolving Vector Quantization</i>
FBeM	-	<i>Fuzzy Set Based Evolving Modeling</i>
FCM	-	<i>Fuzzy C-Means</i>
FIR	-	<i>Fault Isolation Rate</i>
FLEXFIS		<i>Flexible Fuzzy Inference Systems</i>
GK	-	<i>Gustafson-Kessel</i>
IDT	-	<i>Isolation Delay Time</i>
KNN	-	<i>K-Nearest Neighbor</i>
MBPC	-	<i>Model-based Predictive Control</i>
MIMO	-	<i>Multi-Input-Multi-Output</i>

MISO	-	<i>Multi-Input-Single-Output</i>
OIR	-	<i>Operator Intervention Rate</i>
PAC	-	<i>Probability Approximately Correct</i>
PC	-	<i>Personal Computere</i>
PHM	-	<i>Prognostics and Health Management</i>
PLC	-	<i>Programmable Logic Controller</i>
POD	-	<i>Probability of Detection</i>
POFA	-	<i>Probability of False Alarm</i>
SAFIS	-	<i>Sequencial Adaptive Fuzzy Inference System</i>
SECoS	-	<i>Simple Evolving Connectionist System</i>
SOFNN	-	<i>Self-organizing Fuzzy Neural Network</i>
TS	-	<i>Takagi-Sugeno</i>
xTS	-	<i>eXtended Takagi-Sugeno</i>

Lista de Símbolos

N	-	número de amostras
n	-	dimensão do espaço de entrada
k	-	índice da amostra
x_k	-	k -ésima amostra de dados, de dimensão $1 \times n$
c	-	número de grupos
v_i	-	vetor centro do grupo i , de dimensão $1 \times n$
d_{ik}^2	-	distância entre o centro do grupo i e a k -ésima amostra
A_i	-	matriz de adaptação da norma do grupo i , de dimensão $n \times n$
ρ_i	-	volume do grupo i
F_i	-	matriz de dispersão do grupo i , de dimensão $n \times n$
F_{init}	-	matriz de dispersão inicial
u_{ik}	-	grau de pertinência no grupo i da k -ésima amostra
β	-	nível de significância
$\chi_{n,\beta}^2$	-	intervalo de confiança superior da distribuição Qui-Quadrado com n graus de liberdade
M_i	-	número de amostras atribuídas ao grupo i
M_{min}	-	número mínimo de amostras atribuídas a um grupo
p_k	-	probabilidade do erro de previsão
s_k	-	desvio padrão do erro de previsão
z_1	-	nível de alerta
z_2	-	nível de <i>drift</i>
α_i	-	taxa de aprendizado do grupo i
α_{init}	-	taxa de aprendizado inicial
M_{dis}	-	número de dissimilaridades
W_j	-	janela de dados
R_i	-	regra <i>fuzzy</i> do grupo i

y_i	-	saída da regra i
L_i	-	saída correspondente ao rótulo da classe
K	-	número de classes
μ_{ij}	-	função de pertinência Gaussiana do grupo i
σ_{ij}^2	-	variância função de pertinência i
$H(x)$	-	função de pertinência Gaussiana multivariável
Σ	-	matriz de dispersão, de dimensão $n \times n$

Capítulo 1

Introdução

1.1 Contexto e Motivação

Atualmente, em muitas áreas, o avanço da tecnologia tem resultado no surgimento de máquinas e equipamentos complexos, o que impõem grandes desafios para seu gerenciamento e manutenção. Nas indústrias, por exemplo, o diagnóstico de falhas em processos em geral é de vital importância para garantir a operação normal da planta. No caso de processos onde a complexidade é elevada, torna-se difícil para os operadores humanos diagnosticar uma situação anormal (falha) em tempo hábil, levando-os a tomar decisões erradas. Estudos estatísticos na área industrial indicam que aproximadamente 70% dos acidentes são causados por erro humano, o que pode causar perdas econômicas, diminuição da segurança e danos ambientais (Venkatasubramanian, 2005).

Estes fatos levaram ao surgimento de novos conceitos de gerenciamento e manutenção, tais como a manutenção baseada em condição (*Condition-Based Maintenance* – CBM) e o prognóstico e gerenciamento de saúde (*Prognostic and Health Management* – PHM). CBM refere-se à utilização de dados da máquina ou equipamento obtidos em tempo real para determinar sua condição de operação como normal ou falha, permitindo o agendamento da manutenção ou reparo antes que ocorra uma parada. PHM refere-se ao processo em que é realizada a previsão do comportamento futuro da máquina ou equipamento, permitindo estimar sua vida útil em função dos estados de operação atual e passados (Vachtsevanos et al., 2006). Tendo como alicerce os conceitos CBM/PHM, surgiu

o conceito de manutenção inteligente, que emprega sistemas de diagnóstico e prognóstico de falhas avançados. Dessa forma, a manutenção inteligente se torna vital para máquinas e equipamentos atuais.

Para atender a essa necessidade, ao longo das últimas décadas, foram propostos na literatura diversos métodos e técnicas de diagnóstico de falhas, baseados em diferentes teorias e abordagens, e utilizados nas mais variadas aplicações. Em geral, os métodos utilizam modelos (matemáticos ou estatísticos), experiência acumulada, ou dados do histórico do processo para realização do diagnóstico de falhas (Venkatasubramanian, 2005). Os métodos baseados em modelos ou em experiência, apesar de eficientes, têm a desvantagem de exigir conhecimento prévio sobre o sistema dinâmico em questão. Por outro lado, os métodos baseados em dados do processo não necessitam do conhecimento prévio e o conhecimento é extraído a partir dos dados obtidos diretamente da planta relacionada ao processo.

Recentemente, os métodos de diagnóstico de falhas baseados em dados do processo têm recebido maior ênfase na literatura, uma vez que a aquisição de dados por meio de sensores é amplamente utilizada nos sistemas de automação atuais (Abellan-Nebot and Subirón, 2010), o que torna a extração de conhecimento de um sistema dinâmico uma tarefa mais fácil do que elaborar um modelo ou acumular experiência. Nesse tipo de diagnóstico, os autores propõem métodos que empregam os chamados “Sistemas Inteligentes”, ferramentas derivadas da área de Inteligência Computacional, principalmente as redes neurais artificiais, sistemas *fuzzy* e as redes *neurofuzzy*, dentre outras (Jardine et al., 2006).

Entretanto, apesar do diagnóstico de falhas baseados em sistemas inteligentes alcançar um bom desempenho, muitas vezes até superior aos métodos tradicionais, esses enfrentam dificuldades quando o problema envolve o diagnóstico de falhas em sistemas dinâmicos não estacionários, o que representa a maioria dos casos atualmente. Nesses sistemas, parâmetros físicos, características de funcionamento e comportamento das falhas se alteram ao longo do tempo, exigindo que o sistema de diagnóstico de falhas seja adaptativo, de forma a manter sua eficiência mesmo com as alterações no sistema dinâmico. A fim de solucionar esse problema, vários trabalhos encontrados na literatura propõem o uso dos chamados “Sistemas Inteligentes Evolutivos” em diagnóstico de falhas (Chivala et al., 2010; Filev et al., 2010; Lemos et al., 2013; Lughofer and Guardiola, 2008; Petković et al., 2012).

A maioria dos sistemas inteligentes evolutivos, baseados em algoritmos de agrupamento recursivos, adota um mecanismo para atualização da estrutura e parâmetros do sistema (criação/modificação/remoção de grupos) que utiliza alguma medida de similaridade entre as amostras de dados de entrada e os grupos existentes. Esse tipo de mecanismo, apesar de funcional, pode levar a uma definição errônea da estrutura, uma vez que dados com *outliers*¹ ou ruído (como normalmente são os dados adquiridos por meio de sensores em ambientes industriais, por exemplo) que excedam a medida de similaridade, podem gerar grupos que não representam efetivamente um agrupamento de dados. Alguns sistemas inteligentes evolutivos adotam mecanismos mais eficazes para atualização da estrutura e parâmetros do sistema, como os modelos propostos em Lima et al. (2010) e em Lemos et al. (2011a), dessa forma, tendem a ser mais robustos.

Para aplicação em diagnóstico de falhas, a implementação de um sistema inteligente evolutivo baseado em algoritmo de agrupamento recursivo que não seja robusto a *outliers* e ruído é algo crítico, uma vez que estes, sendo interpretados como dados válidos, levam o algoritmo a definir grupos que representam condições de funcionamento ou falha inexistentes, ocasionando posteriormente um aumento no índice de alarme falso e diminuição no índice de isolamento de falhas. Nesse contexto, este trabalho busca o desenvolvimento de algoritmos de aprendizado para implementação de sistemas inteligentes evolutivos com maior robustez a *outliers* e ruído, tendo como foco principal a sua aplicação em diagnóstico de falhas em sistemas dinâmicos não estacionários.

1.2 Objetivos

Este trabalho tem como objetivo principal propor uma metodologia de diagnóstico de falhas baseado em um sistema inteligente evolutivo com emprego de nova abordagem para aprendizagem da estrutura e parâmetros do sistema que permita, a partir de dados de entrada adquiridos em modo *on-line* e, se ne-

¹Na literatura há muitas definições para o termo *outlier*, mas aparentemente não há uma definição universalmente aceita. De forma geral, *outlier* pode ser descrito como um dado (ou um conjunto de dados) que parece ser inconsistente com o restante do conjunto de dados. Um *outlier* é também descrito como novidade, anormalidade ou desvio (Hodge and Austin, 2004).

cessário, em tempo real, realizar o diagnóstico de falhas em sistemas dinâmicos não estacionários. Essa abordagem consiste na utilização de um novo algoritmo de aprendizado que combina um algoritmo de agrupamento recursivo com um método de detecção de *drift*, o que confere ao sistema inteligente evolutivo, além da capacidade de aprendizado contínuo e incremental em modo *on-line*, maior robustez a *outliers* e ruído. Em função das características do algoritmo proposto, os principais benefícios a serem alcançados pelo sistema inteligente evolutivo no diagnóstico de falhas são:

- aprendizado das falhas do sistema dinâmico em modo *on-line* e, se necessário, em tempo real, dispensando a necessidade de conhecimento prévio do modelo do sistema;
- capacidade de adaptação às alterações no sistema dinâmico, permitindo a aplicação em problemas reais.
- baixo índice de alarme falso e alto índice de isolamento de falhas devido a robustez a *outliers* e ruído, aumentando a confiabilidade do diagnóstico.

Outro objetivo desse trabalho consiste em demonstrar a maior aplicabilidade do diagnóstico de falhas baseado sistemas inteligentes evolutivos em relação a outros métodos e técnicas encontradas na literatura, quando aplicados à problemas de engenharia do mundo real. Para atender a esse objetivo, o sistema inteligente evolutivo proposto nesse trabalho é avaliado em dois problemas comumente encontrados na indústria, mais especificamente, o diagnóstico de falhas em sistema de acionamento C.C., e o diagnóstico de falhas em um sistema de tanques interativos. Com os resultados a serem obtidos espera-se demonstrar que a abordagem proposta é uma alternativa eficaz no diagnóstico de falhas em sistemas dinâmicos não estacionários, onde outras abordagens demonstram ser menos vantajosas ou pouco eficazes devido às características de tais sistemas.

1.3 Contribuições da Tese

Algumas publicações derivadas desse trabalho já foram publicadas em congressos e em capítulo de livro. Além disso um artigo foi aceito para publicação em periódico e outro encontra-se em fase de desenvolvimento. A lista completa de trabalhos é apresentada a seguir.

1.3.1 Trabalhos em Periódicos

1. INÁCIO, M.; LEMOS, A.; CAMINHAS, W. *Fault Diagnosis with Evolving Fuzzy Classifier based on Clustering Algorithm and Drift Detection. Mathematical Problems in Engineering*, vol. 2014 (Aceito).
2. SILVA, A.; INACIO, M.; LEMOS, A.; CAMINHAS, W.; GOMIDE, F. *Evolving Fuzzy Systems: A Review and Perspectives* (Em desenvolvimento).

O primeiro item descreve detalhadamente a metodologia de diagnóstico de falhas proposta nesse trabalho, assim como a sua aplicação em diagnóstico de falhas de sistemas dinâmicos não-estacionários, onde foi empregado como exemplo o sistema de acionamento C.C. O segundo item é uma revisão da literatura sobre sistemas *fuzzy* evolutivos apresentando o estado da arte, os desafios da pesquisa, e as perspectivas no desenvolvimento e aplicação desses sistemas em problemas do mundo real.

1.3.2 Capítulo de Livro

1. INÁCIO, M.; MAIA, R.; CAMINHAS, W. *Evolving Fuzzy Classifier Based on the Modified ECM Algorithm for Pattern Classification*. In: Yin, Hujun; Costa, José; Barreto, Guilherme. (Org.). *Lecture Notes in Computer Science - Intelligent Data Engineering and Automated Learning - IDEAL 2012*, 2012, v. 7435, p. 612-621.

Descreve uma proposta inicial de algoritmo de agrupamento evolutivo e sua aplicação em classificação de padrões em modo *on-line* e em tempo real.

1.3.3 Trabalhos em Congressos Nacionais

1. INÁCIO, M.; MAIA, R.; CAMINHAS, W. Diagnóstico de Falhas *on-line* Baseado em um Sistema Inteligente Evolutivo. In: XIX Congresso Brasileiro de Automática, 2012, Campina Grande. Anais do XIX Congresso Brasileiro de Automática, CBA 2012, p. 2000–2007.
2. INÁCIO, M.; MAIA, R.; CAMINHAS, W. Prognóstico de Falhas *on-line* baseado em um Sistema *Fuzzy* Evolutivo. In: II Congresso Brasileiro de

Sistemas *Fuzzy*, 2012, Natal. Anais do II Congresso Brasileiro de Sistemas *Fuzzy*, 2012. p. 504-521.

O primeiro trabalho descreve uma proposta inicial de aplicação de algoritmos de agrupamento evolutivos para implementação de sistemas inteligentes evolutivos (classificador *fuzzy* evolutivo), bem como o emprego desses sistemas em diagnóstico de falhas de sistemas dinâmicos não-estacionários. Da mesma forma, o segundo trabalho descreve uma proposta inicial de implementação de sistemas inteligentes evolutivos (sistema *fuzzy* evolutivo) e seu emprego em prognóstico de falhas de sistemas dinâmicos.

1.3.4 Trabalho em Congresso Internacional

1. INÁCIO, M.; LEMOS, A.; CAMINHAS, W. *Evolving Fuzzy Classifier based on Clustering Algorithm and Drift Detection for Fault Diagnosis Applications. In Annual Conference of the Prognostics and Health Management Society, Forth Worth, TX, USA. Proceedings of the Annual Conference of the Prognostics and Health Management Society*, 2014. p. 399-408.

Descreve a implementação do classificador *fuzzy* evolutivo a partir do algoritmo de agrupamento recursivo com detecção de *drift* proposto nesse trabalho, assim como a sua aplicação em diagnóstico de falhas de sistemas dinâmicos não-estacionários, onde foi empregado como exemplo o sistema de tanques interativos.

1.4 Organização do texto

O restante deste trabalho é composto por 6 capítulos. O capítulo 2 apresenta as definições e o histórico da área de diagnóstico de falhas, bem como o modelo de sistema de diagnóstico de falhas e os principais métodos e técnicas reportados pela literatura. O capítulo 3 apresenta os conceitos teóricos relacionados aos sistemas inteligentes evolutivos, seu histórico, principais modelos propostos na literatura e aplicações desses sistemas. O capítulo 4 apresenta os conceitos teóricos relacionados a algoritmos de agrupamento recursivo, métodos de detecção de *drift*, e detalha o algoritmo de agrupamento recursivo proposto

nesse trabalho. O capítulo 5 apresenta o classificador *fuzzy* evolutivo proposto nesse trabalho e sua aplicação em diagnóstico de falhas de sistemas dinâmicos. O capítulo 6 descreve as simulações de diagnóstico de falhas realizados e os resultados alcançados na avaliação dos algoritmos propostos nesse trabalho. Finalmente, o capítulo 7 apresenta as conclusões sobre esse trabalho, a análise dos resultados e as propostas de continuidade dos estudos.

Capítulo 2

Diagnóstico de Falhas

2.1 Introdução

A tarefa de diagnóstico de falhas é fundamental para o processo de gerenciamento e manutenção de máquinas e equipamentos nas mais diversas áreas, tais como indústria, medicina, militar, aeroespacial, dentre outras.

O diagnóstico de falhas é uma tarefa importante do processo pois, nessas áreas, falhas não diagnosticadas corretamente podem ocasionar graves consequências, tanto no aspecto econômico, quanto nos aspectos de segurança e de meio ambiente. Dessa forma, a implantação de um sistema de diagnóstico de falhas eficiente, que seja capaz de diagnosticar falhas com rapidez e precisão, minimizando os impactos causados pela ocorrência das falhas, é de grande importância (Venkatasubramanian, 2005).

Durante as últimas décadas, diversos métodos e técnicas de diagnóstico de falhas foram propostos na literatura, com contribuições que vão desde os métodos clássicos baseados em modelos aos métodos modernos baseados em dados do processo com emprego de algoritmos derivados da área de inteligência computacional (Vachtsevanos et al., 2006).

Nesse capítulo, a seção 2.2 apresenta as definições de termos comuns na área de diagnóstico de falhas, a seção 2.3 apresenta um breve histórico da evolução do diagnóstico de falhas, a seção 2.4 apresenta um modelo de sistema de diagnóstico de falhas, suas características e princípio de funcionamento. A seção 2.5 descreve os principais métodos de diagnóstico de falhas reportadas pela

literatura.

2.2 Definições

Ao longos dos últimos anos, as definições de termos na área de pesquisa de diagnóstico de falhas em sistemas dinâmicos têm mudado, conforme são incorporados novos métodos e técnicas de diagnóstico em máquinas e equipamentos dentro do *framework* CBM/PHM. Apesar de ainda não existir um vocabulário universal na comunidade acadêmica, nas áreas industrial, militar e outras são aceitas as seguintes definições de falha e diagnóstico de falha (Venkatasubramanian, 2005):

- Falha: o termo “falha” é geralmente definido como um afastamento da variável observada, ou do parâmetro calculado associado com o processo, em relação à faixa de valores aceitáveis.
- Diagnóstico de falha: geralmente definido como o processo de detecção, isolamento e identificação de uma condição iminente ou incipiente que afeta um componente (sistema ou subsistema).

Os termos detecção, isolamento e identificação de falha são tipicamente empregados para transmitir os seguintes significados:

- Detecção de falha: ação de detectar e reportar uma condição anormal de operação.
- Isolamento de falha: ação de determinar qual componente (sistema ou subsistema) está falhando ou falhou.
- Identificação de falha: ação de estimar a natureza ou a extensão da falha.

Outros termos são associados ao conceito de diagnóstico de falhas, tal como o termo prognóstico de falha, que é definido como o processo de estimar a vida útil remanescente da máquina, ou definido ainda como a previsão do tempo de evolução de uma falha desde o seu início até a parada total de funcionamento de um componente. Na literatura há pouca discordância sobre a relação de

dependência entre diagnóstico e prognóstico de falhas, porém, a exata demarcação entre esses dois campos não é muito bem definida. De acordo com as definições de cada abordagem, diagnóstico envolve a detecção, identificação e a quantificação da falha que está ocorrendo ou ocorreu, enquanto prognóstico lida com previsão da falha que ainda vai ocorrer. Embora as saídas do diagnóstico sejam mais úteis em uma aplicação prática, o prognóstico depende das saídas do diagnóstico (por exemplo, indicadores de falha) e, geralmente, não pode ser feito isoladamente (Sikorska et al., 2011).

2.3 Histórico

A busca por métodos e técnicas de diagnóstico de falhas eficientes remota à época em que os profissionais dessas áreas começaram a lidar com máquinas e equipamentos mais complexos, onde a simples observação do seu funcionamento não era mais suficiente para detectar ou prever falhas (Vachtsevanos et al., 2006).

Há uma longa história de tentativa de desenvolvimento e implementação de diagnóstico de falhas em vários níveis, do básico ao avançado. A principal motivação para isso foi a necessidade de maximizar o tempo de uso dos equipamentos e minimizar os custos de manutenção. Durante as décadas de 1950 e 1960, os sistemas de diagnóstico de falhas consistiam de simples indicadores analógicos, capazes apenas de sinalizar a ocorrência de determinadas anormalidades (alarmes). Na década de 1970 os sistemas digitais e computadorizados foram introduzidos, permitindo a implementação de sistemas de diagnóstico mais avançados, capazes não apenas de indicar mas também isolar as falhas. A partir da década de 1980, os sistemas de diagnóstico tornaram-se cada vez mais avançados e integrados, com capacidade de diagnosticar múltiplas falhas simultaneamente, oferecendo ainda informações adicionais para dar suporte ao gerenciamento e manutenção de máquinas baseados nos modernos conceitos CBM/PHM.

Em termos históricos, os métodos de diagnóstico de falhas são mais conhecidos do que os métodos de prognóstico de falhas, pois vem sendo investigados ao longo das últimas décadas por pesquisadores não somente da área de engenharia, mas também de outras áreas tais como medicina, economia, finanças e

ciências em geral (Vachtsevanos et al., 2006). Durante muitos anos, a pesquisa e o desenvolvimento de sistemas de diagnóstico de falhas ficaram concentradas nas áreas industrial, militar, naval e aeroespacial, mas recentemente têm surgido aplicações de diagnóstico e prognóstico de falhas em outras áreas tais como o diagnóstico de falhas em sistemas de aquecimento/refrigeração em construção civil (Katipamula and Brambley, 2005).

2.4 Sistema de Diagnóstico de Falhas

De forma geral, um sistema de diagnóstico de falhas é um sistema capaz de detectar, isolar, e identificar a falha. Para isso, o sistema de detecção e diagnóstico de falhas deve receber tanto informações de controle quanto informações do estado atual do processo. Essas informações normalmente são fornecidas pelo controlador e por sensores, respectivamente. A Figura 2.1 mostra de forma genérica a aplicação dos sistemas de diagnóstico de falhas em processos, onde x corresponde às variáveis de entradas, e y corresponde às variáveis de saídas do processo. Nesse caso, as fontes de falha podem ser desde mau funcionamento do controlador, distúrbios no processo, falhas estruturais na planta, falhas nos sensores, ou ainda falhas nos atuadores.

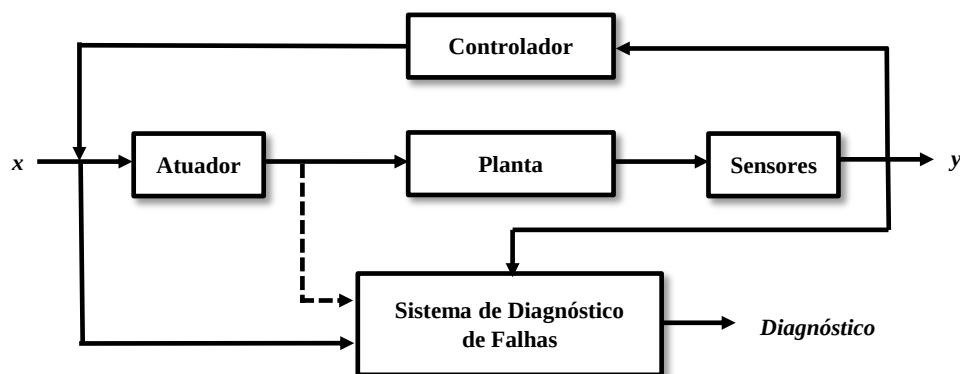


Figura 2.1: Aplicação de sistema de diagnóstico de falhas em processos.

Na aplicação de diagnóstico de falhas em um processo qualquer, normalmente o sistema de diagnóstico deve lidar com as seguintes classes de mau funcionamento ou falhas (Venkatasubramanian et al., 2003a):

- Alterações de parâmetros do processo: falhas causadas por alteração de parâmetros do processo ocorrem quando um distúrbio entra no processo a partir do ambiente por meio de uma ou mais variáveis exógenas (independentes). Como exemplos desse tipo de falha podem-se citar a alteração da concentração de um reagente em relação ao seu estado normal na alimentação de um processo químico, ou a alteração no coeficiente de transferência de calor em trocadores de calor devido à formação de depósitos de material.
- Alterações estruturais: falhas causadas por alterações estruturais referem-se a alterações no processo em si. Essas ocorrem devido aos danos nos componentes e que resultam em mau funcionamento da planta. Como exemplos desse tipo de falha podem-se citar o mau funcionamento de um controlador de processo ou agarramento de uma válvula, ou ainda uma tubulação entupida ou com vazamento.
- Mau funcionamento em sensores e atuadores: falhas causadas por sensores e atuadores usualmente ocorrem quando há uma falha fixa no componente ou uma alteração no seu funcionamento. Alguns instrumentos fornecem sinais de realimentação que são essenciais para o controle do processo e quando há uma falha nesses instrumentos isso pode causar um desvio das variáveis de estado da planta para além dos limites aceitáveis, a menos que a falha seja detectada imediatamente e que ações corretivas sejam realizadas a tempo.

No mundo real, os sistemas de diagnóstico de falhas também precisam lidar com falhas causadas por fatores que normalmente estão fora do escopo do diagnóstico, tais como as incertezas não estruturadas, os ruídos do processo e os ruídos de medição. Incertezas não estruturadas são, principalmente, falhas que não estão modeladas *a priori*. Ruídos do processo referem-se ao descompasso entre o estado atual do processo e as previsões dadas pelas equações do modelo, enquanto ruídos de medição refere-se à presença de sinais espúrios de alta frequência nas medições dos sensores.

Em sistemas de diagnóstico de falhas, diversas características são desejáveis, tais como (Venkatasubramanian et al., 2003a):

- Rapidez: detectar e diagnosticar a falha no menor tempo possível.
- Robustez: imunidade a *outliers* e ruído.
- Isolabilidade: distinguir diferentes falhas.
- Adaptabilidade: manter a eficiência mesmo quando ocorre mudanças no processo.
- Identificação de novidade: distinguir em certa condição, a operação normal de falha. E em caso de falha, identificar se é uma falha conhecida ou uma nova falha.
- Identificação de múltiplas falhas: capacidade de identificar múltiplas falhas simultaneamente no processo.
- Estimação da taxa de classificação: fornecer a *priori* uma estimativa do erro de classificação que pode ocorrer.
- Interpretabilidade: fornecer explicações de como a falha se originou e se propagou.
- Requisitos de modelagem: quantidade de esforço de modelagem necessário para desenvolvimento do sistema.
- Requisitos computacionais e de armazenamento: quantidade da capacidade computacional e espaço de armazenamento necessários para execução do sistema.

Muitas das características desejadas em um sistema de diagnóstico de falhas são contraditórias (por exemplo, diagnóstico rápido e confiável) e dificilmente todas elas são satisfeitas por um único sistema. Entretanto, conhecer estas características é útil para avaliação dos vários métodos de diagnósticos de falhas, permitindo a comparação desses em termos de eficiência, confiabilidade, generalidade, dentre outros aspectos.

2.5 Métodos de Diagnóstico de Falhas

Diferentes métodos de diagnóstico de falhas em sistemas dinâmicos são encontrados na literatura, sendo que, a principal diferença entre eles está na forma como o conhecimento do processo é utilizado para o diagnóstico. Basicamente, os métodos para diagnóstico de falhas são divididos em duas categorias (Venkatasubramanian, 2005):

- métodos baseados em modelos;
- métodos baseados no histórico do processo.

Em ambas as categorias, pressupõe-se a existência de algum tipo de conhecimento prévio sobre o sistema em questão, seja na forma de um modelo (quantitativo ou qualitativo), seja na forma de dados do histórico do processo (dados dos quais se pode extrair o modelo do sistema).

Os métodos de diagnóstico de falhas baseados em modelos quantitativos utilizam um conjunto de relações matemáticas quantitativas que representam o processo físico (Venkatasubramanian et al., 2003a). Esses métodos utilizam a redundância analítica para detectar e diagnosticar falhas. A redundância analítica consiste em comparar as variáveis do processo medidas por sensores com valores calculados analiticamente a partir do modelo. Como exemplos de métodos de diagnóstico de falhas que pertencem a essa categoria estão os métodos que empregam observadores de estado (Chen and Patton, 1999; Frank, 1994; Takahashi and Peres, 1999; Takahashi et al., 1999), relações de paridade (Ploix and Adrot, 2006), filtros de Kalman (Chang and Hwang, 1994; Gao et al., 2008; Zolghadri, 1996) e estimação de parâmetros (Isermann, 1994).

Os métodos de diagnóstico de falhas baseados em modelos qualitativos, diferentemente dos métodos baseados em modelos matemáticos, utilizam um conjunto de relações qualitativas ou bases de conhecimento para extrair conclusões acerca do estado do processo ou sistema (isto é, se a operação é normal ou falha) (Venkatasubramanian et al., 2003b). Dentre os métodos de diagnóstico de falhas que pertencem a essa categoria estão, por exemplo, os métodos que empregam dígrafos (Maurya et al., 2006; Vedam and Venkatasubramanian, 1997), árvores de falhas (Dutuit and Rauzy, 2005; Lee et al., 1985) e sistemas *fuzzy* (Detroja et al., 2006; El-Shal and Morris, 2000; Ragot and D.Maquin, 2006).

Em contraste com os métodos anteriores que necessitam de um tipo de conhecimento prévio do processo (quantitativo ou qualitativo), os métodos de diagnóstico de falhas baseados no histórico do processo utilizam uma grande quantidade de dados para extrair o conhecimento acerca do processo (Venkatasubramanian et al., 2003c). Os dados são obtidos a partir das variáveis de entrada e saída medidas por sensores, processados e utilizados no diagnóstico de falhas. O processamento de dados é também conhecido como extração de características ou extração de parâmetros. Os métodos que empregam sistemas especialistas (Cho and Park, 1997; Laurentys et al., 2011a), análise de tendência qualitativa (D’Angelo et al., 2011a,b; Maurya et al., 2007), análise de componentes principais (Dunia et al., 1996; Li et al., 2000), máquinas de vetores suporte (Baccarini et al., 2011; Widodo and Yang, 2007), redes neurais artificiais (Calado et al., 2001; D’Angelo and Costa, 2001; Leonard and Kramer, 1993), redes *neurofuzzy* (Caminhas and Takahashi, 2001; Frank and Koppen-Seliger, 1997; Zio and Gola, 2009), redes Bayesianas (D’Angelo et al., 2014; Li et al., 2013; Verron et al., 2010) e os sistemas imunes artificiais (Laurentys et al., 2011b; Silva et al., 2012) pertencem à categoria dos métodos de diagnóstico de falhas baseados em dados do histórico do processo.

Apesar de existirem diferentes abordagens para o diagnóstico de falhas em sistemas dinâmicos, um único método pode não ser adequado para atender a todos os requisitos de uma aplicação. De forma geral, a qualidade dos resultados de qualquer um dos métodos depende da qualidade da informação fornecida. Uma forma de melhorar o sistema de diagnóstico de falhas é integrar as características de um método com outro, de forma a superar as limitações de cada estratégia individual (Venkatasubramanian, 2005).

Capítulo 3

Sistemas Inteligentes Evolutivos

3.1 Introdução

Nos dias atuais, sistemas computacionais são utilizados em praticamente todas as áreas que envolvam trabalho humano. Desde pequenos sistemas utilizados em residências, até grandes sistemas utilizados nas empresas, os sistemas computacionais tornaram-se quase imprescindíveis para que as pessoas consigam realizar suas atividades com eficiência. Em muitos casos, os sistemas computacionais substituem a ação humana para realização de algumas tarefas, tornando o processo em questão automatizado, com o objetivo de oferecer às pessoas mais segurança, conforto ou então liberá-las para a realização de outras tarefas. Entretanto, com a evolução tecnológica, muitos problemas do mundo real se tornam cada vez mais complexos, exigindo para sua solução sistemas computacionais com características “inteligentes”, tais como aprendizado e raciocínio baseado em conhecimento.

Na tentativa de viabilizar o desenvolvimento de sistemas computacionais inteligentes nas últimas décadas, foram propostos na literatura sistemas computacionais baseados em redes neurais artificiais, sistemas *fuzzy* e algoritmos evolucionários. Mais recentemente, pesquisadores do campo de Inteligência Computacional perceberam que as soluções de problemas complexos e em tempo real exigiam um aspecto dos sistemas inteligentes ainda não explorado: a adaptabilidade. Com tal característica, os sistemas inteligentes seriam capazes de resolver problemas em ambientes dinâmicos e não estacionários como os tipicamente en-

contrados nas áreas da indústria, automotiva, medicina, militar, aeroespacial, dentre outras. Desta forma, surgiu então um novo campo de pesquisa derivado da área de Inteligência Computacional dedicado ao estudo dos chamados sistemas inteligentes evolutivos (Angelov and Kasabov, 2006; Kasabov and Filev, 2006).

Iniciando o capítulo, seção 3.2 apresenta os principais conceitos teóricos relacionados aos sistemas inteligentes evolutivos, em seguida, a seção 3.3 apresenta um breve histórico da evolução desses sistemas e a seção 3.4 apresenta alguns dos seus principais modelos propostos na literatura. Finalizando o capítulo, a seção 3.5 apresenta exemplos de aplicações dos sistemas inteligentes evolutivos em problemas reais.

3.2 Conceitos

Os chamados sistemas inteligentes evolutivos são sistemas inteligentes geralmente baseados em redes neurais artificiais, regras *fuzzy*, ou redes *neurofuzzy*, capazes de, a partir de dados de entrada adquiridos em modo *on-line* e muitas vezes em tempo real, determinar gradualmente tanto a sua estrutura quanto seus parâmetros (Angelov and Kasabov, 2006). A aplicação dos sistemas inteligentes evolutivos tem crescido nos últimos anos, sendo que muitos trabalhos na literatura apresentam casos de aplicação com sucesso em problemas complexos do mundo real envolvendo modelagem, controle, classificação ou previsão (Angelov et al., 2010). Tais sistemas são diferentes dos sistemas adaptáveis, que visam apenas de adaptar os seus parâmetros internos ao problema em questão (Aström and Wittenmark, 1995). Os sistemas inteligentes evolutivos também são diferentes dos sistemas que empregam os algoritmos evolucionários, nos quais um processo baseado em operadores de seleção, cruzamento e mutação é capaz de evoluir um indivíduo melhor adaptado ao problema a ser resolvido, a partir de uma população de indivíduos (Abraham and Grosan, 2008). Sistemas adaptativos e evolucionários são mais apropriados para aplicações em que as mudanças no ambiente são relativamente lentas e, geralmente, esses sistemas se adaptam/evoluem em modo *off-line* (Angelov and Kasabov, 2006).

Um aspecto importante dos sistemas inteligentes evolutivos é que há diferentes abordagens teóricas e práticas que podem ser utilizadas para a sua imple-

mentação, tais como os sistemas baseados em redes neurais artificiais, em regras *fuzzy*, em agentes inteligentes ou ainda sistemas híbridos (Angelov et al., 2010). Independentemente da abordagem a ser utilizada, as principais características dos sistemas inteligentes evolutivos são:

- a sua estrutura não é fixa e não é definida *a priori*: ela se desenvolve (expande ou retrai) naturalmente à medida que o sistema evolui;
- os seus parâmetros são ajustados (adaptados) conforme o sistema evolui;
- o seu funcionamento é contínuo, ocorrendo um aprendizado em modo *on-line* e, se necessário, em tempo real.

O desenvolvimento dos sistemas inteligentes evolutivos adota como modelo a forma de evolução natural dos indivíduos, especialmente a dos seres humanos. Na natureza, os indivíduos evoluem durante a sua vida de forma gradual e contínua, a partir do aprendizado e da utilização do conhecimento acumulado. As experiências vividas pelo indivíduo permitem que ele se torne cada vez mais apto para superar as dificuldades impostas pelas mudanças no ambiente e assim ser capaz de resolver problemas cada vez mais difíceis.

A Figura 3.1 ilustra de forma genérica os sistemas inteligentes evolutivos. O principal componente da estrutura é o sistema inteligente em si (por exemplo, uma rede neural artificial, um sistema *fuzzy* ou uma rede *neurofuzzy*), responsável pelas tarefas de raciocínio e tomada de decisão. Essas tarefas são realizadas a partir dos dados de entrada e utilizando o conhecimento acumulado na base de conhecimento. Para atualizar a base de conhecimento, é utilizado um mecanismo de aprendizado *on-line*, o que confere ao sistema a característica de evolutivo, uma vez que ele se torna capaz de adquirir novos conhecimentos com a experiência, aumentando dessa maneira sua capacidade de resolver novos problemas.

De acordo com Angelov et al. (2010) os sistemas inteligentes evolutivos podem ser vistos como uma sinergia entre sistemas inteligentes com estrutura expansível (evolutiva) para representação da informação (como as estruturas dos sistemas *fuzzy* e das redes *neurofuzzy*, mas não necessariamente essas) e métodos aprendizado de máquina *on-line*. A pesquisa para o desenvolvimento de sistemas inteligentes evolutivos tem como foco os processos não estacionários, algoritmos de aprendizado computacionalmente eficientes para aplicações

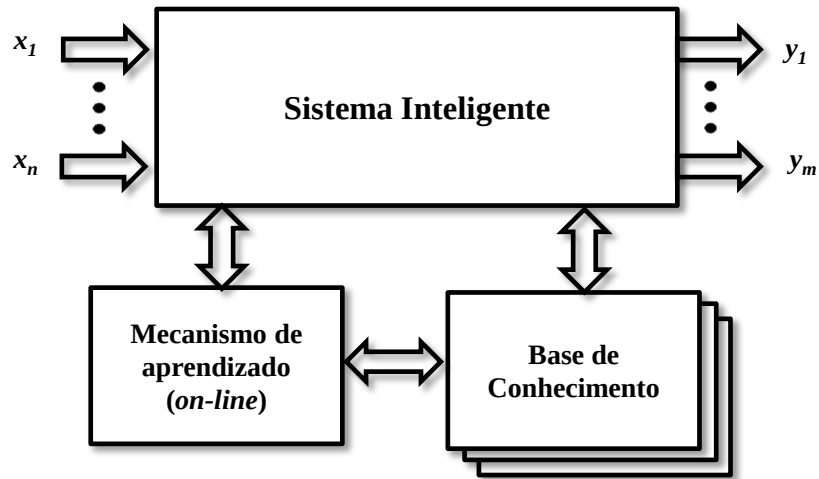


Figura 3.1: Representação da estrutura dos sistemas inteligentes evolutivos.

em tempo real e modelos evolutivos que possam ser empregados nas tarefas tais como modelagem, controle, classificação ou previsão.

Concluindo, pode-se afirmar que os principais objetivos a serem alcançados na implementação dos sistemas inteligentes evolutivos são o aprendizado contínuo, a capacidade de auto-adaptação e auto-organização (evolução da estrutura do sistema) de forma que tais sistemas possam se adaptar à ambientes dinâmicos, não estacionários e complexos.

3.3 Histórico

O paradigma dos sistemas inteligentes evolutivos surgiu a partir de estudos que tinham como objetivo descobrir novos métodos e técnicas que permitissem tornar os sistemas inteligentes conhecidos (especialmente as redes neurais artificiais e os sistemas *fuzzy*, ou a combinação de ambos, as redes *neurofuzzy*) em sistemas cuja estrutura fosse expansível (evolutiva), de forma a se ajustar ao problema em questão, visando maior desempenho.

Os primeiros trabalhos publicados na literatura na década de 1990 se baseavam nas redes neurais artificiais (Fritzke, 1994) ou redes *neurofuzzy* (Kasabov, 1998) para desenvolvimento de sistemas inteligentes evolutivos. Nesses modelos

de redes, a estrutura é flexível e alterada durante o processo evolutivo. Basicamente, duas abordagens para construção da rede foram propostas: os nós (ou nodos) da rede já existem antes do treinamento e somente as conexões entre os nós são criadas durante o treinamento, ou tanto os nós quanto as conexões são criadas durante o treinamento. Esses primeiros modelos foram aperfeiçoados em trabalhos seguintes, assim como deram origem a outros modelos baseados em redes neurais artificiais evolutivas (Kasabov, 2001; Kasabov and Filev, 2006; Kasabov and Song, 2002).

Outros trabalhos publicados em meados dos anos 2000 propuseram sistemas inteligentes evolutivos baseados em modelos de regras flexíveis (Angelov, 2002; Angelov and Buswell, 2001). Em tais modelos, um conjunto flexível de regras *fuzzy* realiza o mapeamento não linear entre entradas e saídas, utilizados o modelo *fuzzy* tipo Mamdani ou tipo Takagi-Sugeno. Um algoritmo de aprendizado *on-line* realiza a identificação do modelo, isso significa que tanto a estrutura do modelo (base de regras) quanto os seus parâmetros são modificados durante o processo evolutivo. Nos anos seguintes, surgiram na literatura outros trabalhos com aperfeiçoamentos dos sistemas inteligentes evolutivos baseados em modelos de regras flexíveis (Angelov and Filev, 2004, 2005), e sistemas com características semelhantes aos anteriores, mas com novos métodos de aprendizado (Leite and Gomide, 2011; Lemos et al., 2011a,b; Leng et al., 2005; Lima et al., 2010; Lughofer, 2008).

Nos últimos anos, muitos outros trabalhos sobre sistemas inteligentes evolutivos foram publicados, tratando tanto do desenvolvimento teórico quanto das aplicações desses sistemas em problemas do mundo real (Lughofer, 2011). Uma demonstração do grande interesse despertado pelo tópico Sistemas Inteligentes Evolutivos na comunidade científica é o grande número de simpósios, *workshops*, tutoriais e seções especiais, realizados em diversos eventos, tais como:

- *Annual Meeting of the North American Fuzzy Information Processing Society* (NAFIPS).
- *Conference on Adaptive and Intelligent Systems* (ICAIS).
- *Conference on European Society of Fuzzy Logic and Technology Conference* (EUSFLAT).
- *Conference on Evolving and Adaptive Intelligent Systems* (EAIS).

- *Conference on Information Processing and Management of Uncertainty for Knowledge-Based Systems* (IPMU).
- *Conference on Machine Learning and Applications* (ICMLA).
- *Conference on Neural Information Processing* (ICONIP).
- *Conference on Systems, Man, and Cybernetics* (IEEE-SMC).
- *Conference on Fuzzy Systems* (FUZZ-IEEE).
- *International Fuzzy Systems Association World Congress* (IFSA).
- *International Joint Conference on Neural Networks* (IJCNN).
- *International Symposium Intelligent Systems* (IEEE-IS).
- *Symposium on Evolving and Autonomous Learning System* (EALS).
- *Symposium on Evolving Fuzzy Systems* (EFS).
- *Symposium on Evolving Intelligent Systems* (EIS).
- *Symposium Series on Computational Intelligence* (SSCI).
- *Workshop on Evolving and Self-Developing Intelligent Systems* (ESDIS).
- *Workshop on Genetic and Evolving Fuzzy Systems* (GEFS).
- *Workshop on Neurocomputing and Evolving Intelligence* (NCEI).
- *World Congress on Evolutionary Computation* (WCCI).

Além dos eventos científicos, outra forma de demonstração da importância que os sistemas inteligentes evolutivos têm alcançado na comunidade são os diversos periódicos e livros publicados, onde se destacam os seguintes:

- *Applied Soft Computing: Special issue on Evolving Soft Computing Techniques and Applications* (periódico).
- *Evolving Connectionist Systems: Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines* (livro).

- *Evolving Connectionist Systems: The Knowledge Engineering Approach* (livro).
- *Evolving Fuzzy Systems: Methodologies, Advanced Concepts and Applications* (livro).
- *Evolving Intelligent Systems: Methodology and Applications* (livro).
- *Evolving Rule-based Models: A Tool for Design of Flexible Adaptive Systems* (livro).
- *Evolving Systems* (periódico).
- *IEEE Transactions on Fuzzy Systems: Special issue on Evolving Fuzzy Systems* (periódico).
- *Information Sciences: Special section on Online fuzzy machine learning and data mining* (periódico).
- *Learning in Non-Stationary Environments: Methods and Applications* (livro).

A Figura 3.2 mostra um gráfico do crescimento do número de publicações no campo de pesquisa dos sistemas inteligentes evolutivos entre os anos de 1990 a 2013, onde pode-se comprovar que esse campo teve um rápido desenvolvimento. Os números apresentados no gráfico foram obtidos através de pesquisa na ferramenta de busca *ISI Web of Science* (Reuters, 2014), utilizando as seguintes palavras-chave: (*adaptive OR evolving OR self-organizing*) AND (*neural network OR fuzzy rules OR fuzzy neural network*).

Entretanto, apesar do grande desenvolvimento alcançado no campo de pesquisa dos sistemas inteligentes evolutivos e dos inúmeros casos relatados na literatura de aplicação desses sistemas com sucesso em problemas de modelagem, controle, previsão, classificação e processamento de dados em geral, ainda há muita demanda para aplicações em diversas áreas onde os problemas são complexos e o ambiente é dinâmico e não estacionário, como por exemplo: sistemas autônomos, manufatura avançada, processos industriais complexos, finanças, economia, ciências sociais, biomedicina, biotecnologia e nanotecnologia. Dessa

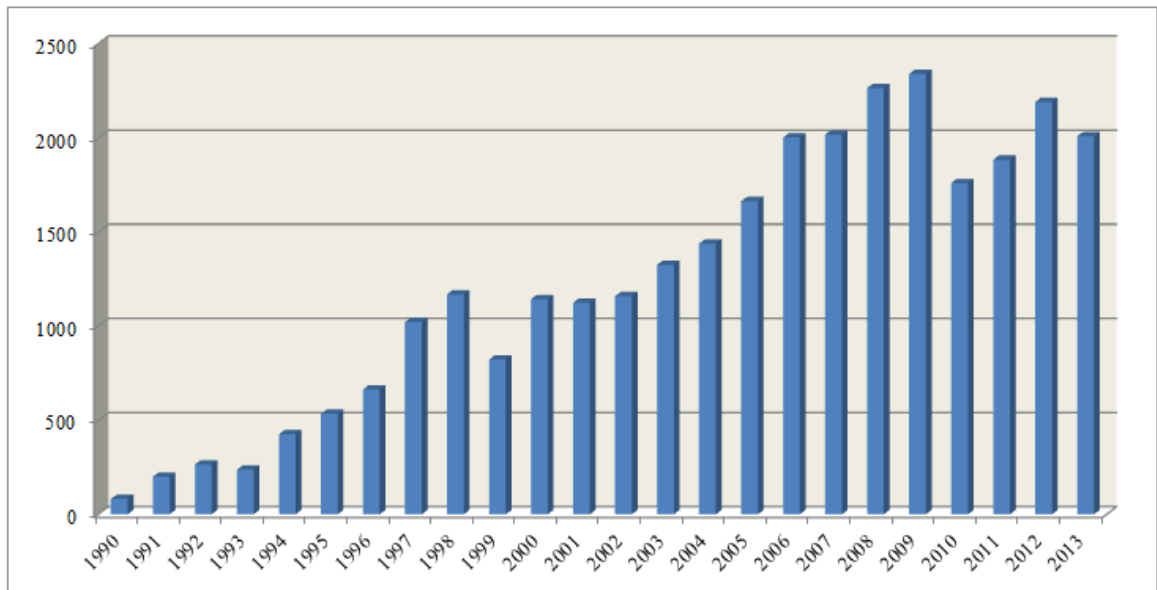


Figura 3.2: Número de publicações sobre sistemas inteligentes evolutivos nos últimos anos.

forma, pode-se afirmar que o campo de pesquisa dos sistemas inteligentes evolutivos continua em crescimento, não está saturado e continua demandando o desenvolvimento teórico e metodológico. (Angelov et al., 2010).

3.4 Modelos

Os modelos de sistemas inteligentes evolutivos propostos na literatura seguem três abordagens básicas: redes neurais artificiais, sistemas *fuzzy* ou redes *neurofuzzy*. Apesar das várias abordagens, esses modelos têm em comum a característica de não possuir a estrutura do sistema fixa e definida *a priori*, a estrutura e seus parâmetros evoluem durante um processo de aprendizado em modo *on-line* e, se necessário, em tempo real. O que diferencia os modelos de sistemas inteligentes evolutivos são as estratégias e os mecanismos adotados para realizar a evolução do sistema, o que confere ao modelo maior ou menor grau de flexibilidade, robustez e confiabilidade. A seguir são descritos os modelos propostos na literatura que se destacam.

Evolving Connectionist Systems (ECOS)

Os modelos de sistemas inteligentes evolutivos do tipo *Evolving Connectionist Systems* (ECOS) são sistemas tipo conexionistas modulares (redes), capazes de evoluir sua estrutura e funcionalidade de maneira contínua, auto-organizada, *on-line*, adaptativa, e interativa a partir da entrada de dados (Kasabov and Filev, 2006). Implementações de sistemas ECOS requerem estruturas conexionistas que incorporem os princípios de tais sistemas, como os modelos *Evolving Fuzzy Neural Network* (EFuNN) (Kasabov, 2001) e o *Dynamic Evolving Neuro-Fuzzy Inference Systems* (DENFIS) (Kasabov and Song, 2002). Os sistemas ECOS realizam o aprendizado de modelos locais do processo a partir do agrupamento de dados e da associação de cada grupo a uma função de saída local. Os grupos de dados são criados conforme a similaridade das amostras dos dados apenas no espaço de entrada (no caso do modelo DENFIS) ou, simultaneamente, no espaço de entrada e no espaço de saída (no caso do modelo EFuNN). As amostras que têm maior similaridade em relação a um grupo existente, são alocadas ao grupo. As amostras não pertencentes a nenhum grupo existente formam novos grupos ao longo do tempo, conforme a entrada de dados. A medida de similaridade entre as amostras e os grupos existentes pode ser realizada de várias formas, sendo que a mais comum é a distância Euclidiana normalizada entre a amostra e os centros dos grupos.

O modelo EFuNN consiste de uma rede do tipo *feedforward* de cinco camadas, onde cada camada executa uma função específica, e com uma camada de realimentação (opcional). Nessa rede, os nós da camada de entrada representam os valores dos dados de entrada e os nós da camada *fuzzy* de entrada representam os graus de pertinência de cada variável em relação às funções de pertinência de entrada predefinidas. Os nós das regras evoluem através de aprendizado não supervisionado ou supervisionado, e representam os centros dos grupos das associações entre os dados de entrada e dados de saída. Cada nó representando uma regra é definido por dois vetores de pesos. O primeiro é ajustado através de um aprendizado não supervisionado baseado na similaridade entre amostras de dados no espaço de entrada do problema, o segundo é ajustado através de um aprendizado supervisionado baseado no erro da saída. Os nós da camada *fuzzy* de saída representam os graus de pertinência das variáveis de saída em relação às funções de pertinência de saída predefinidas, e os nós da camada de saída representam os valores das variáveis de saída. A

regra de aprendizado empregada no modelo EFuNN leva em consideração tanto a similaridade espacial quanto a correlação temporal entre os dados e, na implementação desse modelo, podem ser acrescentadas regras de eliminação e de agregação de nós.

Diferentemente do modelo EFuNN, que consiste numa rede *neurofuzzy* que evolui incrementalmente sua estrutura e funcionalidade utilizando algoritmo de agrupamento, o modelo DENFIS é um sistema de inferência *fuzzy* do tipo *Takagi-Sugeno* (TS). O sistema de inferência *fuzzy* empregado no modelo DENFIS é do tipo TS de primeira ordem (Takagi and Sugeno, 1985). No modelo DENFIS todas as regras *fuzzy* são criadas e atualizadas durante um processo de treinamento de “um passo”, baseado no método de agrupamento *Evolving Clustering Method* (ECM). O ECM é um algoritmo de agrupamento evolutivo rápido, de modo *on-line*, baseado em distância máxima, que implementa um particionamento do espaço de entrada para o propósito de criar e atualizar regras de inferência *fuzzy*. Nos antecedentes das regras são utilizados conjuntos *fuzzy* definidos por funções de pertinência triangulares, cujos parâmetros são determinados pelo agrupamento de dados e pelo valor de limiar. Nos consequentes das regras são utilizadas funções lineares, nas quais os parâmetros são estimados e atualizados através de uma versão recursiva do algoritmo de mínimos quadrados.

Outros modelos de sistemas inteligentes evolutivos do tipo ECOS foram propostos na literatura e são descritos a seguir. O modelo *Simple Evolving Connectionist System* (SECoS) (Watts and Kasabov, 2000) consiste em uma rede de arquitetura simplificada, como apenas 3 camadas de neurônios e um número mínimo de neurônios em cada uma delas, apenas o necessário para o aprendizado. A primeira camada é a camada de entrada, com funções de transferência linear. A segunda camada é a camada oculta e é a camada que evolui durante o aprendizado, sendo essa equivalente à camada de regras do modelo EFuNN. E a terceira camada é a camada de saída. Nas camadas oculta e de saída são utilizadas funções de ativação linear saturada. O modelo SECoS pode ser visto como uma versão minimalista do modelo EFuNN, com os componentes de fuzzificação e defuzzificação removidos. A vantagem do modelo SECoS é sua implementação mais simples e, como há poucas conexões e um menor número de neurônios, é necessário muito menos processamento para simulação da rede. O modelo *Evolving Self-Organising Map* (ESOM) (Deng and Kasabov, 2000)

é semelhante ao modelo clássico de redes neurais tipo mapa auto-organizável com duas camadas e conexões ponderadas entre elas. A segunda camada é a camada que evolui nesse modelo. Diferentemente de outros modelos tipo ECOS, esse modelo utiliza aprendizado não supervisionado baseado no conceito de formação dinâmica de grupos espaciais. O algoritmo de aprendizado do modelo ESCOM atualiza a rede de modo *on-line* criando novos nós quando necessário, removendo as conexões mais fracas bem como eliminando os neurônios que tiveram todas as suas conexões removidas.

Evolving Takagi-Sugeno Models (eTS)

O modelo *Evolving Takagi-Sugeno Models* (eTS) consiste em um sistema inteligente evolutivo baseado na identificação de sistema *fuzzy* tipo TS, utilizando um algoritmo de aprendizado contínuo e de modo *on-line*, responsável pela identificação tanto da estrutura do modelo (base de regras), quanto dos seus parâmetros (Angelov, 2002; Angelov and Filev, 2004). Desenvolvido a partir do conceito *Evolving Rule-Base Models* (eR), esse sistema utiliza o potencial informativo (informação de proximidade espacial acumulada) das amostras de dados de entrada como medida para atualizar a base de regras. O algoritmo de aprendizado avalia recursivamente a relação entre esses potenciais e dinamicamente atualiza as regras *fuzzy* e os parâmetros dos antecedentes, combinado com um processo de atualização recursiva dos parâmetros dos consequentes.

No modelo eTS, a base de regras constitui um grupo de regras com antecedentes *fuzzy*, nos quais são utilizados conjuntos *fuzzy* definidos por funções de pertinência Gaussianas, e consequentes funcionais que seguem o método de inferência de TS. A identificação do modelo pode ser dividido em duas subtarefas:

1. Aprendizado dos antecedentes das regras, que consiste na determinação do ponto focal das regras, isto é, seus centros e os parâmetros das funções de pertinência.
2. Aprendizado dos parâmetros das funções lineares dos consequentes das regras.

O aprendizado dos antecedentes das regras é realizado por agrupamento no espaço de entrada, empregando o método de agrupamento *Subtractive Cluste-*

ring. Nesse método, os pontos desse espaço são candidatos a centro dos grupos, sendo a capacidade de um ponto ser escolhido como centro determinada pelo seu potencial, uma medida da proximidade espacial entre um ponto em particular e todos os outros pontos. O valor do potencial de um ponto será alto se esse ponto é circundado por um grande número de outros pontos e, dessa forma, é razoável que este seja estabelecido como centro de um grupo. O potencial de todos outros pontos de dados é reduzido por uma quantidade proporcional ao potencial do ponto escolhido como centro e inversamente proporcional à distância desse centro. O próximo centro de grupo é definido (após o processo de redução) também como o ponto de maior potencial. O procedimento se repete até que o potencial de todos os pontos de dados seja reduzido abaixo de certo limiar. O aprendizado dos consequentes das regras é tratado como um problema de mínimos quadrados, utilizando um algoritmo recursivo, o vetor de parâmetros dos modelos lineares é encontrado minimizando-se uma função objetivo global. Outra forma de aprendizado dos consequentes das regras é encontrar os parâmetros dos modelos lineares minimizando-se uma função objetivo local para cada regra.

A partir do modelo eTS, outros modelos de sistemas inteligentes evolutivos foram propostos, como o modelo Simple_eTS (Angelov and Filev, 2005). A principal modificação nesse modelo em relação ao modelo eTS original foi a inclusão de um mecanismo para eliminação de regras que se tornam irrelevantes durante o aprendizado. O modelo Simple_eTS usa o conceito de dispersão derivado do potencial baseado na densidade dos dados, mas com uma redução na complexidade dos cálculos de potencial, o que torna o algoritmo computacionalmente mais eficiente. Nesse modelo são utilizadas funções de pertinência do tipo Cauchy nos antecedentes das regras, ao invés de funções Gaussianas, e o algoritmo recursivo de mínimos quadrados para aprendizado dinâmico dos parâmetros e estrutura do modelo TS. O modelo *eXtended Takagi-Sugeno* (xTS) (Angelov and Zhou, 2006) consiste numa versão estendida do modelo eTS, onde as principais modificações foram a inclusão do cálculo do raio de influência de cada grupo de forma recursiva e uma medida de relevância dos grupos baseada no conceito de idade. Outra versão estendida do modelo eTS é o modelo eTS+ (Angelov et al., 2010). Nesse modelo é empregada uma nova abordagem para atualização dos parâmetros e da base de regras. Nessa abordagem a aprendizagem é guiada por critérios como idade, utilidade, densidade local e zona

de influência, caracterizando-a como uma técnica robusta, flexível e autônoma. Outras características desse modelo são a possibilidade da base de regras iniciar vazia e a não necessidade de definição de limites e parâmetros pelo usuário. Um modelo baseado no eTS+ foi proposto posteriormente (Simple_eTS+) (Angelov, 2011).

Flexible Fuzzy Inference Systems (FLEXFIS)

O modelo *Flexible Fuzzy Inference Systems* (FLEXFIS) é outro exemplo de sistema inteligente evolutivo baseado em sistemas *fuzzy* tipo TS, utilizando aprendizado incremental a partir dos dados de entrada e com foco na conexão entre a adaptação dos parâmetros não lineares dos antecedentes das regras e dos parâmetros lineares dos consequentes das regras (Lughofer, 2008, 2011). Esse modelo emprega um treinamento incremental/evolutivo robusto, onde o aprendizado é baseado em um método de agrupamento que encontra o grupo mais próximo calculando a distância da nova amostra de dados de entrada até a superfície dos grupos existentes (ao invés dos centros, como nos métodos de agrupamento convencionais). A estratégia de treinamento *on-line* da parte não linear do modelo é combinada com uma adaptação recursiva dos parâmetros lineares existentes nos consequentes das regras *fuzzy*.

O aprendizado dos parâmetros dos antecedentes das regras no modelo FLEXFIS é baseado no agrupamento dos dados no espaço de entrada/saída em partes locais e projeção desses grupos de modo a definir a parte dos antecedentes das regras (isto é, os conjuntos *fuzzy*) e as regras em si. O método de agrupamento utilizado é uma extensão do método *Vector Quantization* (VQ) denominado *Evolving Vector Quantization* (eVQ). No método de agrupamento eVQ, alguns conceitos são responsáveis por suas características. Um deles é o ganho de aprendizagem, que define o grau de deslocamento dos centros e é responsável pela convergência do processo de aprendizagem. No método eVQ não há um ganho de aprendizagem global que decresce com o número de iterações, mas sim diferentes ganhos de aprendizagem para os diferentes grupos de acordo com sua significância. A significância do *i*-ésimo grupo é calculada em função do número de amostras de dados que formam esse grupo. Outro conceito é representado por um parâmetro denominado de “vigilante”, que define a evolução dos grupos. Esse parâmetro guia o processo de criação de novos grupos sem provocar

alterações em nenhum dos grupos existentes (característica de estabilidade) e alteração dos grupos existentes para adaptar a nova informação (característica de plasticidade).

O aprendizado dos parâmetros dos consequentes das regras no modelo FLEX-FIS emprega a abordagem de aprendizado local, onde cada regra é tratada separadamente e os respectivos parâmetros lineares dos consequentes são otimizados utilizando um algoritmo recursivo de mínimos quadrados ponderados. Nessa abordagem, o vetor de parâmetros é atualizado a cada nova amostra de dados de entrada obtida. A estratégia de combinação entre o aprendizado das antecedentes e evolução das regras com o aprendizado dos consequentes ocorre da seguinte forma:

- Se um novo grupo é formado, automaticamente uma nova regra é definida sem alterar os parâmetros dos consequentes das outras regras. Somente os parâmetros dos consequentes da nova regra são estimados.
- Se nenhum novo grupo é formado, o centro de grupo e a zona de influência do grupo vencedor são atualizados, e antes de atualizar os parâmetros dos consequentes da regra vencedora, é adicionado um vetor de correção aos parâmetros lineares e uma matriz de correção à matriz Hessiana inversa. Esse procedimento tem como objetivo balancear uma situação de não-otimalidade causada pela atualização nos parâmetros da parte dos antecedentes das regras.

Uma versão estendida do modelo FLEXFIS, denominada FLEXFIS++, foi proposta em Lughofer (2012), onde a principal modificação foi a inclusão de um mecanismo de união de regras visando a redução da complexidade desnecessária do modelo, alcançando uma redução da dimensionalidade de maneira suave e gradual. Dentre outros benefícios obtidos pelo modelo FLEXFIS++ está uma maior confiabilidade e interpretabilidade do modelo evoluído.

Self-organizing Fuzzy Neural Network (SOFNN)

O modelo *Self-organizing Fuzzy Neural Network* (SOFNN) consiste em um sistema inteligente evolutivo do tipo rede *neurofuzzy* e que implementa o modelo de sistema *fuzzy* TS em modo *on-line* (Leng et al., 2005). A primeira camada

dessa rede é formada por neurônios de função base elipsoidal, sendo que cada neurônio dessa camada possui um vetor centro e um vetor largura. Diferentemente de outras redes *neurofuzzy*, os neurônios de função base elipsoidal são organizados pela própria rede, não necessitando do conhecimento prévio do número de neurônios. O algoritmo de aprendizado do modelo SOFNN é dividido em duas partes: aprendizado dos parâmetros e aprendizado da estrutura. O aprendizado dos parâmetros utiliza uma versão modificada do algoritmo recursivo de mínimos quadrados. O aprendizado da estrutura inclui técnicas de inclusão e eliminação de neurônios. As abordagens propostas no modelo SOFNN garantem a completeza- ϵ das regras *fuzzy* e contribuem para aliviar o dilema entre interpretabilidade e precisão.

A arquitetura do modelo SOFNN é formada por cinco camadas, definidas da seguinte forma:

- Camada 1: cada neurônio dessa camada representa uma variável de entrada.
- Camada 2: os neurônios dessa camada são neurônios de base elipsoidal e representam a premissa das regras *fuzzy*. A saída desses neurônios é calculada como o produto das pertinências dadas por funções de pertinência Gaussiana.
- Camada 3: nessa camada, a saída de cada neurônio é calculada como o grau de pertinência de cada neurônio da camada anterior normalizado pela soma de todos os graus de pertinência.
- Camada 4: cada neurônio dessa camada possui duas entradas, uma recebe a saída do neurônio correspondente da camada 3 e a outra recebe um vetor de *bias* ponderado (produto de um vetor de *bias* pelo vetor de parâmetros). Esse é, portanto, o consequente das regras *fuzzy*. A saída de cada neurônio é calculada como o produto das duas entradas.
- Camada 5: nessa camada, cada neurônio representa uma variável de saída, calculada como o somatório dos sinais de entrada recebidos da camada 4.

A adaptação dos parâmetros dos consequentes das regras *fuzzy* em modo *on-line*, no modelo SOFNN, é realizada por aprendizado global de todos os parâmetros, utilizando uma versão modificada do algoritmo recursivo de mínimos

quadrados. A modificação consiste na inclusão de um parâmetro de ponderação na equação de atualização do vetor de parâmetros. Tal parâmetro é definido em função do erro de estimação e do erro de aproximação. Quando o erro de estimação é menor que o erro de aproximação, o parâmetro de ponderação será definido como zero, e o vetor de parâmetros não é ajustado. Isto garante uma convergência assintótica do erro de estimação e a convergência do vetor de parâmetros para valores finitos.

O processo de aprendizado do modelo SOFNN inclui o aprendizado da estrutura, que procura alcançar o tamanho ideal da rede através de um mecanismo de auto-organização. O número de neurônios na segunda camada é aumentado ou reduzido dinamicamente durante o processo de aprendizado. A estratégia para incluir um novo neurônio (nova regra *fuzzy*) é baseada em dois critérios: o primeiro é um critério de erro (calculado como o desvio entre a saída estimada e a saída real), e o segundo é um critério que verifica se as novas amostras de dados de entrada são suficientemente cobertas por algum neurônio (regra *fuzzy*) da rede (por comparação da saída de cada neurônio da segunda camada com um limiar). Esses dois critérios combinados produzem quatro condições que devem ser avaliadas para inclusão de neurônios. A estratégia para eliminação de neurônios é baseada na ideia de eliminação dos neurônios menos importantes, verificando se a performance da rede como um todo é aceitável mesmo quando tais neurônios são excluídos. Nesse caso, a exclusão de um neurônio equivale a definir os seus parâmetros como zero.

Sequential Adaptive Fuzzy Inference System (SAFIS)

O modelo *Sequential Adaptive Fuzzy Inference System* (SAFIS) é outro exemplo de sistema inteligente evolutivo do tipo rede *neurofuzzy* (Rong et al., 2006). O SAFIS emprega um método baseado nas redes neurais tipo rede de funções de base radiais (*Radial Basis Function* - RBF) para construção da base de regras de um sistema *fuzzy* TS de ordem zero e pode ser representado por uma rede *feedforward* com cinco camadas. Esse sistema de inferência *fuzzy* utiliza um critério baseado na distância Euclidiana entre as amostras e os centros dos grupos, em conjunto com o conceito de influência para adicionar e remover regras de forma incremental. Nesse modelo, o conceito de influência de uma regra *fuzzy* é definido como sua contribuição para o sistema *fuzzy* como um todo

em termos estatísticos. O modelo SAFIS utiliza funções de pertinência Gaussianas nos antecedentes das regras e constantes (*singletons*) nos consequentes. Nesse modelo, os parâmetros das regras são atualizados por um algoritmo do tipo filtro de Kalman modificado (*Extended Kalman Filter* - EKF). Uma versão estendida do modelo SAFIS, denominada *Extended SAFIS* (ESAFIS), foi proposta em Rong et al. (2011), onde as modificações propostas visam principalmente a melhoria na acurácia da previsão e aumento da velocidade de processamento em relação ao modelo original. No modelo ESAFIS foi introduzida uma modificação no conceito de influência, que é utilizado no mecanismo de inclusão e remoção das regras. Outra modificação introduzida nesse modelo é a utilização do algoritmo recursivo de mínimos quadrados para atualização dos consequentes das regras quando não há alteração no número de regras, o que contribui para redução da complexidade computacional.

Evolving Fuzzy Participatory Learning (ePL)

O modelo *Evolving Fuzzy Participatory Learning* (ePL) (Lima et al., 2010) permite o desenvolvimento de sistemas inteligentes evolutivos baseados em regras *fuzzy* utilizando uma abordagem que combina o conceito de aprendizado participativo (Yager, 2006) com a ideia de modelagem de sistemas *fuzzy* evolutivos. Nessa aplicação, o conceito de aprendizado participativo é empregado em um algoritmo de agrupamento não supervisionado utilizado para encontrar a estrutura da base de regras durante o processo de modelagem evolutivo. O modelo ePL emprega o modelo de sistema *fuzzy* TS e, de forma semelhante aos modelos SimplLeTS e eTS, realiza a identificação do modelo por estimação do ponto focal das regras (parâmetros dos antecedentes), exceto que o modelo ePL utiliza agrupamento *fuzzy* baseado no aprendizado participativo, ao invés do agrupamento baseado em espalhamento, densidade ou potencial informativo. Com os parâmetros dos antecedentes definidos, os parâmetros dos consequentes das regras *fuzzy* podem ser encontrados utilizando o método de mínimos quadrados. Para atualização da base de regras durante o processo de aprendizado em modo *on-line*, o modelo ePL aplica o mesmo processo dos modelos SimplLeTS e eTS: a cada passo do aprendizado incremental é verificado quando um novo grupo deve ser criado, ou um grupo existente deve ser atualizado, ou ainda quando um grupo redundante existente deve ser eliminado.

No modelo ePL, diferentemente do modelo eTS, a atualização da base de regras é baseada em uma medida de similaridade *fuzzy* para determinar a proximidade entre a nova amostra de dados e a base de regras existente. A principal característica do aprendizado participativo é o modo como é realizado o aprendizado do modelo, isto é, o aprendizado depende do que o sistema já sabe sobre o modelo. Dessa forma, no modelo ePL, o modelo atual é parte do processo evolutivo em si e influencia o modo como novas amostras de dados serão utilizadas na auto-organização da estrutura. Além disso, uma importante propriedade do aprendizado participativo é que o impacto causado por uma nova amostra de dados na auto-organização ou revisão do modelo depende da sua compatibilidade com a estrutura da base de regras atual (ou a estrutura do agrupamento atual).

O processo de aprendizado no modelo ePL inicia pela atualização da base de regras, ou seja, atualização da estrutura do agrupamento de dados. Considerando um conjunto de centros de grupo atual, um parâmetro de medida de compatibilidade entre a nova amostra de dados e a estrutura do agrupamento é calculado, assim como é calculado um parâmetro denominado índice de alerta. Enquanto o primeiro parâmetro mede o quanto a nova amostra é compatível com a estrutura do agrupamento, o segundo parâmetro é responsável por indicar quando a estrutura do agrupamento atual deve ser revisada de acordo com a nova informação contida nos dados. Se o índice de alerta for maior que um dado limiar, então um novo centro de grupo é incluído, definindo suas coordenadas como as coordenadas da amostra de dados atual. Caso contrário, o centro de grupo mais compatível (mais próximo) com a amostra de dados é atualizado. O índice de alerta pode ser interpretado como o complemento da confiabilidade no conhecimento atual, ou seja, na estrutura da base de regras. Esse mecanismo monitora a performance do sistema pela observação da compatibilidade do modelo atual com os dados. Assim, o aprendizado no modelo ePL é dinâmico, pode ser visto como uma estratégia de revisão baseada no conhecimento, uma vez que a taxa de aprendizado efetiva depende da compatibilidade entre as novas amostras de dados e a estrutura do agrupamento atual, e também da confiança no modelo. Um mecanismo de exclusão de grupos redundantes está presente no modelo ePL, nesse caso, através do cálculo de um parâmetro de similaridade entre centros de grupo. Um centro de grupo é considerado redundante quando a sua similaridade com outro centro de grupo é maior ou igual a um

dados limiar. Quando isso ocorre, o grupo redundante é simplesmente eliminado. Após o agrupamento de dados realizado na primeira etapa do aprendizado do modelo ePL, a base de regras *fuzzy* é construída de forma similar ao modelo eTS. Os centros de grupo encontrados definem os parâmetros das funções de pertinência Gaussianas, exceto a dispersão, que deve ser definida *a priori* para alcançar um nível apropriado de sobreposição de regras. Numa segunda etapa do aprendizado, os parâmetros lineares das funções dos consequentes das regras associadas a cada grupo são ajustados utilizando o algoritmo recursivo de mínimos quadrados.

Uma versão aperfeiçoada do modelo ePL foi proposta na literatura, sendo denominada de ePL+ (Maciel et al., 2014). Nessa versão, foram incorporadas três modificações principais, descritas a seguir. A primeira modificação consiste na adição de um mecanismo para determinar dinamicamente os raios dos grupos, que representam a zona de influência das funções de pertinência. Ao contrário do modelo ePL, onde o valor da zona de influência era fixo e determinado pelo usuário, nessa nova versão esse parâmetro do algoritmo é determinado durante o aprendizado. A segunda modificação consiste em uma alteração no mecanismo de eliminação de grupos redundantes, que passa a ser baseado em uma medida de utilidade, semelhante ao algoritmo do modelo eTS+ e não mais baseado numa medida de compatibilidade como no modelo ePL. E a terceira modificação consiste na alteração da estrutura do modelo, permitindo que o mesmo possa ser implementado tanto como um sistema *fuzzy* tipo *Multi-Input-Single-Output* (MISO), quanto como um sistema *fuzzy* tipo *Multi-Input-Multi-Output* (MIMO). Dessa forma, o modelo ePL+ é computacionalmente mais eficiente, requer menos conhecimento acerca do ambiente a ser modelado e permite aplicações mais avançadas em função da habilidade de modelar sistemas mais complexos.

Multivariable Gaussian Evolving Fuzzy Modeling (eMG)

O modelo de sistema *fuzzy* evolutivo eMG, assim como o modelo ePL, também é baseado no conceito de aprendizado participativo. Mas, diferentemente do ePL, o modelo eMG utiliza funções de pertinência tipo Gaussiana multivariável na representação dos grupos, sendo a estrutura do agrupamento atualizada recursivamente a cada passo do algoritmo, e limiares definidos automaticamente

(Lemos et al., 2011a). O uso de funções de pertinência tipo Gaussiana multivariável é adotado com o objetivo de se evitar perda de informações sobre as interações entre as variáveis de entrada. O modelo eMG emprega o modelo de sistema *fuzzy* TS, cujos parâmetros das funções de pertinência são estimados pelo algoritmo de agrupamento evolutivo proposto, denominado Algoritmo de Agrupamento Participativo Gaussiano, e os parâmetros dos consequentes das regras *fuzzy* são atualizados por um algoritmo recursivo de mínimos quadrados ponderados. O algoritmo de agrupamento evolutivo empregado no modelo eMG considera a possibilidade de interação entre as entradas. Nesse algoritmo os grupos são estimados com base em uma medida normalizada de distância, semelhante a distância de Mahalanobis, que produz grupos elipsoidais cujos eixos não são necessariamente paralelos aos eixos das variáveis de entrada, como ocorre quando utilizada a distância Euclidiana. Dessa forma, essa medida de distância preserva as possíveis interações entre as variáveis de entrada. As funções de pertinência Gaussianas multivariáveis que representam os grupos são caracterizadas por um vetor central e uma matriz de dispersão, que representa a dispersão presente em cada variável de entrada. Outra característica do algoritmo de agrupamento evolutivo do modelo eMG é o uso de um mecanismo de ajuste automático do valor de limiar de distância baseado na dimensão do espaço de entrada, a fim de evitar a chamada maldição da dimensionalidade (aumento do número de grupos e aumento da complexidade do modelo à medida que a dimensão do espaço de entrada aumenta).

O processo de aprendizado do modelo eMG, assim como em outros modelos, inicia pelo aprendizado dos antecedentes das regras que, nesse modelo, possuem funções de pertinência multivariáveis, a partir do agrupamento de dados realizado pelo algoritmo de agrupamento. A cada iteração, o número de regras do modelo é igual ao número de grupos já encontrados pelo algoritmo de agrupamento. Assim, a cada iteração um novo grupo pode ser criado, ou um grupo existente pode ter seus parâmetros modificados, e ainda, dois grupos redundantes podem ser unidos. Caso um novo grupo seja criado, a regra correspondente é criada com os parâmetros do antecedente extraídos do grupo. Caso um grupo existente seja atualizado, os parâmetros dos antecedentes também são atualizados de acordo com o novo centro do grupo e da matriz de dispersão. O aprendizado dos parâmetros do consequente de cada regra é realizado utilizando-se o algoritmo recursivo de mínimos quadrados ponderados de forma

similar a outros modelos de sistemas *fuzzy* evolutivos. As características do modelo eMG fazem com que esse seja diferente em relação aos demais modelos propostos na literatura em três aspectos principais:

- utilização de funções de pertinência multivariáveis para os conjunto *fuzzy* dos antecedentes das regras a fim de evitar a perda de informação relacionada a interação entre as variáveis de entrada;
- utilização de um algoritmo de agrupamento robusto a ruídos e *outliers*, uma vez que é derivado do aprendizado participativo, um conceito que provê um mecanismo de suavização de dados incompatíveis;
- criação de novas regras empregando um mecanismo automático para ajustar valores de limiares considerando a dimensão do espaço de entrada, de forma a evitar a maldição da dimensionalidade.

Fuzzy Set Based Evolving Modeling (FBeM)

Modelos de sistemas inteligentes evolutivos podem ser implementados com a abordagem *Fuzzy Set Based Evolving Modeling* (FBeM) (Leite and Gomide, 2011), que é baseada na computação granular (Pedrycz and Gomide, 2007). Essa abordagem emprega grânulos de informação tipo *fuzzy* para construir mapas granulares que associam dados granulares de entradas à dados granulares de saída. O uso de grânulos *fuzzy* garante a generalidade da estrutura dos dados, com uso de algoritmos de matemática simples e regras que descrevem seu comportamento. A modelagem FBeM pode ser vista como uma estratégia para construir conjuntos *fuzzy* para representar dados imprecisos. O funcionamento de um sistema FBeM consiste em, basicamente, perceber os dados de um fluxo sob diferentes resoluções e decidir entre a adoção de granularidade em menor ou maior grau. Os modelos FBeM utilizam funções de pertinência trapezoidais e regras que combinam consequentes linguísticos tipo Mamdani e funcionais tipo TS para realizar aproximações singulares e granulares de funções não estacionárias. Em Leite et al. (2011) são propostos modelos FBeM que utilizam funções de pertinência Gaussianas e regras que combinam consequentes linguísticos e funcionais. O processo evolutivo nos modelos consiste em incluir um novo grânulo (regra) quando as regras existentes não são suficientemente

ativadas por uma nova amostra de dados de entrada, ou contrair/expandir os grânulos quando a amostra ativa suficientemente uma regra. Um mecanismo de união de grânulos baseado em distância é utilizado para diminuir a redundância do modelo FBeM. Nesse modelo é empregada uma estratégia de remoção de grânulos baseada na inatividade, isto é, grânulos são excluídos quando se tornam inativos por um determinado número de passos pré-definidos. A remoção de grânulos inativos periodicamente contribui para manter a base de regras atualizada.

Evolving Neo-Fuzzy Neural Network (eNFN)

O modelo *Evolving Neo-Fuzzy Neural Network* (eNFN) (Silva et al., 2014) consiste em um sistema inteligente evolutivo do tipo rede *neurofuzzy*, baseado no modelo *Neo-Fuzzy Neuron* (NFN) (Yamakawa et al., 1992). A estrutura desse tipo de rede consiste basicamente em “ n ” estruturas tipo TS de ordem zero, uma para cada entrada. Ao contrário do modelo NFN que possui estrutura fixa, no modelo eNFN a estrutura da rede é flexível, isto é, evolui de acordo com o fluxo de dados de entrada. O algoritmo de aprendizado desse modelo realiza a evolução da estrutura pela alteração do valor modal (centro) das funções de pertinência e também pela inclusão ou exclusão de funções de pertinência. Nesse modelo são utilizadas funções de pertinência triangulares e complementares, onde o número inicial de funções de pertinência pode ser definido empiricamente, ou com base no conhecimento *a priori* do sistema, ou ainda empregando alguma técnica de agrupamento. No modelo eNFN o mecanismo de criação e inclusão de funções de pertinência é baseado no valor médio do erro local das regras correspondentes às funções de pertinência mais ativas, isto é, se o valor médio do erro local for maior que o erro médio global de modelagem, então são adicionadas novas funções de pertinência à região correspondente. Nesse modelo é adotada uma estratégia de exclusão de funções de pertinência baseada no conceito de idade da regra, onde por idade da regra entende-se o intervalo de tempo em que uma função de pertinência fica sem ser ativada. Dessa forma, são excluídas as funções de pertinência cujo tempo de inatividade for maior que um limiar de tempo de inatividade pré-definido. As funcionalidades propostas no modelo eNFN têm por objetivo manter um erro mínimo e uniforme para todas as regiões do espaço de entrada e permitir a

adaptação dos parâmetros e estrutura da rede a fim de lidar com as mudanças no fluxo de dados de entrada.

Conforme relatado na seção 3.3, a área dos sistemas inteligente evolutivos tem apresentado um grande crescimento nos últimos anos. Como consequência desse crescimento, um grande número de novos modelos têm sido propostos na literatura recente, onde os autores procuram utilizar novas abordagens visando alcançar melhores características em relação a modelos anteriores e superar os desafios que as aplicações atuais impõe ao uso dos sistemas inteligentes evolutivos. Um exemplo de nova abordagem é a abordagem não linear para problemas de previsão empregando sistemas *fuzzy* evolutivos proposta em Maciel et al. (2013), sendo a mesma é baseada no conceito de nuvens de dados (Angelov and Yager, 2012).

3.5 Aplicações

As principais características dos sistemas inteligentes evolutivos, como a capacidade de adaptação em ambientes dinâmicos e não estacionários, e aprendizado em modo *on-line*, os tornam ideais para aplicação em problemas complexos do mundo real. Os sistemas inteligentes evolutivos podem ser utilizados em modelagem, controle, classificação ou previsão, sendo que, inúmeros trabalhos encontrados na literatura demonstram que tais sistemas possuem vantagens em relação a outros métodos ou técnicas de resolução do problema em questão, especialmente nas aplicações industriais (Angelov and Kasabov, 2006; Kasabov and Filev, 2006).

Em algumas áreas, os problemas são um desafio aos sistemas inteligentes evolutivos, pois exigem alto grau de adaptação e aprendizagem em um ambiente sujeito a inúmeras perturbações. Como exemplo pode-se citar as áreas de processos industriais complexos, veículos autônomos, sensores inteligentes, dentre outras. Apesar do desenvolvimento alcançado pelos sistemas inteligentes evolutivos ao longo das últimas décadas, a demanda por aplicações de tais sistemas em problemas complexos nas mais diversas áreas é crescente e contínua, o que motiva a pesquisa por sistemas ainda mais eficientes (Angelov et al., 2010). A seguir são apresentados alguns exemplos de aplicação dos sistemas inteligentes evolutivos.

Modelagem

A modelagem de sistemas é uma importante aplicação para os sistemas inteligentes evolutivos, uma vez que, obter o modelo de um sistema é uma tarefa imprescindível para controlar ou analisar tal sistema. Apesar de existir uma grande variedade de abordagens para modelamento de sistemas, para muitos processos do mundo real, esta tarefa pode ser complexa ou de alto custo computacional.

Modelos de regressão do tipo $y(k) = f(x(k))$ podem ser obtidos através do uso de sistemas inteligentes evolutivos onde, a partir de um conjunto de dados obtidos do sistema de modo *on-line* e, possivelmente, em tempo real (como por exemplo, obtidos por meio de sensores), o sistema gera e evolui automaticamente um modelo de regressão não linear. É importante ressaltar que, o sistema inteligente evolutivo, ao contrário de outras abordagens, não irá utilizar nenhuma informação prévia sobre o sistema a ser modelado, sendo o aprendizado sobre o sistema realizado inteiramente a partir dos dados de entrada.

Um exemplo de aplicação dos sistemas inteligentes evolutivos em modelagem é apresentado em Angelov and Zhou (2006), onde o modelo do sistema de emissão de gases NOx de um motor de carro é obtido a partir de dados reais medidos *on-line* e em tempo real. Nessa aplicação é utilizado a versão aperfeiçoada do modelo de sistema inteligente evolutivo eTS, denominada eTS+. Esse modelo consiste em um sistema *fuzzy* tipo TS evolutivo, que realiza o aprendizado a partir do fluxo de dados de entrada combinando a identificação da estrutura e parâmetros do sistema através do agrupamento de dados não supervisionado, e aprendizado supervisionado dos parâmetros dos consequentes das regras *fuzzy*.

Nessa aplicação, as entradas de dados consistem em 180 variáveis diferentes medidas através de sensores, tais como pressão dos cilindros, torque do motor, velocidade de rotação, etc. Normalmente, um subconjunto das variáveis de entrada é escolhido para a modelagem, baseado em conhecimento anterior, experiências ou aplicação de métodos de seleção de características de modo off-line. Mas o modelo de sistema inteligente evolutivo eTS+ dispõe de um mecanismo que processa todas as entradas em modo *on-line* e seleciona automaticamente o melhor subconjunto de entradas para a modelagem. Nesse caso,

a estrutura do sistema eTS+ obtida consiste em apenas 4 entradas e 4 regras *fuzzy*. Comparando os resultados do sistema inteligente evolutivo obtido através da seleção automática de entradas com outros sistemas que empregam um número pré-selecionado de entradas, observa-se que o erro de estimação é menor no modelo que utiliza as entradas selecionadas automaticamente, e ainda, o tempo de computação foi reduzido significativamente.

No exemplo de aplicação de sistemas inteligentes evolutivos em modelagem apresentado é demonstrado que esses sistemas permitem obter resultados relevantes em termos de precisão e tempo quando comparado a outras abordagens, além de apresentarem as vantagens inerentes de tais sistemas como, robustez, flexibilidade e autonomia (evolução do sistema a partir dos dados de entrada sem depender de limiares ou parâmetros pré-definidos).

Controle

Atualmente, o controle de processos envolve o uso de sistemas sofisticados, tendo em vista metas de qualidade e produtividade exigidas pelas empresas. Considerando que, na maioria das aplicações reais, o ambiente é dinâmico e sujeito a perturbações (tais como variações dos sinais em sensores ou falhas em equipamentos do processo), o uso de sistemas inteligente evolutivos no controle de processos representa uma alternativa atraente em relação aos sistemas de controle clássicos.

Uma das abordagens de maior sucesso nas aplicações industriais é o método de controle denominado *Model-based Predictive Control* (MBPC), onde o modelo do processo é utilizado para prever a saída do sistema. Uma estratégia de controle ótimo é obtida minimizando o erro resultante da diferença entre a saída desejada e a saída prevista. Uma vez que a abordagem MBPC requer um modelo confiável, a modelagem e identificação do processo é a maior parte do projeto do controlador. Essa tarefa pode ser realizada de forma mais eficiente empregando-se técnicas de modelamento baseadas em sistemas *fuzzy* ou redes neurais artificiais (Babuska and Verbruggen, 1996).

Um exemplo de aplicação dos sistemas inteligentes evolutivos em controle do tipo MBPC é apresentado em Chivala et al. (2010), onde o modelo do processo é obtido por um sistema inteligente do tipo TS evolutivo (eTS), sendo que a base de regras e os parâmetros do sistema são continuamente evoluídos pela

adição de novas regras e pela atualização das regras e parâmetros existentes. O uso do sistema inteligente evolutivo permitiu obter um controle tolerante a falhas, com a capacidade adicional de lidar com falhas desconhecidas, o que permitiu aumentar o desempenho do sistema de controle.

Nessa aplicação, o processo a ser controlado consiste em uma coluna de destilação, do tipo empregado nas indústrias químicas e refinaria de petróleo para separação dos componentes de uma substância. A coluna de destilação considerada nesse caso, é usada para separar água e etanol. O objetivo é controlar a temperatura de um *reboiler*, a partir das diversas variáveis medidas no processo, o que constitui um sistema tipo MISO. As entradas empregadas são: taxa do fluxo de alimentação, fração do líquido de alimentação, refluxo na válvula, fração de alimentação molar de etanol e temperatura do *reboiler*. O sistema inteligente evolutivo empregado nessa aplicação foi inicializado com um número de grupos inicial igual a 2, que é o mesmo número de regras. Esse número de regras revelou-se ser suficiente para modelar o processo em questão.

O sistema proposto também foi avaliado sob a ocorrência de falhas no processo. Normalmente, uma coluna de destilação está sujeita a falhas devido à mudanças nas variáveis do processo tais como taxa do fluxo de alimentação, composição do fluxo de alimentação dentre outras e ainda falhas devido a mudanças na temperatura do *reboiler*. Nessa aplicação foram simuladas dois tipos de falhas: variação da composição do fluxo de alimentação e variação na temperatura do *reboiler*, na forma de falhas abruptas e de falhas incipientes.

Os resultados obtidos nessa aplicação demonstraram que o uso de um sistema inteligente evolutivo permitiu obter, além do controle do processo com precisão, a adaptação às falhas no processo, uma vez que a saída do sistema se manteve estável mesmo na ocorrência das mesmas, tornado assim o sistema de controle tolerante a falhas.

Classificação

Muitas tarefas cotidianas atuais são dependentes da utilização de processos de reconhecimento de padrões, e a cada dia surgem novas aplicações. Apesar de existir uma vasta pesquisa na área de reconhecimento de padrão, os desafios impostos por novas aplicações tornam necessária a busca por métodos e técnicas de reconhecimento de padrões mais eficientes. Uma das etapas de reconheci-

mento de padrões é a classificação, que consiste em, a partir de um conjunto de dados “brutos”, atribuir corretamente um padrão a uma categoria ou classe. Para esta tarefa são utilizados algoritmos de classificação, sendo estes algoritmos baseados em algum método de aprendizado, normalmente supervisionado ou não supervisionado (Duda et al., 2000).

Dentre os algoritmos de classificação propostos na literatura, os algoritmos de classificação baseados em regras *fuzzy* tem sido aplicados em tarefas como tomada de decisão, detecção de falhas ou classificação de imagens, devido a suas vantagens em relação a outros algoritmos de classificação. (Lughofer et al., 2007).

Sistemas inteligentes evolutivos, tais como os sistemas *fuzzy* ou *neurofuzzy* evolutivos podem ser aplicados em problemas de classificação, onde um classificador *fuzzy* é evoluído automaticamente a partir dos dados de entrada (sinais de medição, dados, imagens, etc.). Um exemplo de aplicação dos sistemas inteligentes evolutivos em classificação é apresentado em Lughofer et al. (2007). Nessa aplicação é utilizado um tipo de classificador *fuzzy* evolutivo denominado FLEXFIS-Class em duas versões, modelo *fuzzy* único (FLEXFIS-Class SM) e multimodelo *fuzzy* (FLEXFIS-Class MM), para classificação de imagens. Ambas as versões do classificador são derivadas do modelo FLEXFIS.

Nessa aplicação é proposto uma estrutura autoconfigurável e adaptativa para detecção de falhas em imagens, que permite classificar cada imagem como boa ou ruim, sendo o classificador evoluído a partir dos dados e da informação fornecida pelo operador. As imagens são obtidas do processo de produção em modo *on-line*, com alta frequência, a partir do sistema de supervisão. A estrutura é composta de etapas de pré-processamento, segmentação e classificação.

As imagens utilizadas nessa aplicação são de um processo de impressão de discos CD, onde falhas devido a cores fracas, palhetas erradas, dentre outros fatores, devem ser detectadas no processo com uma frequência em torno de 1 Hz. O conjunto de dados compreende de 1164 imagens gravadas individualmente. Um subconjunto de 776 imagens foi utilizado como imagens de treinamento para construção dos classificadores de maneira evolutiva, e o restante das imagens (imagens de teste) foram utilizadas somente para classificação (sem adaptação dos classificadores), de forma tornar possível a comparação com outros tipos de classificadores. Foram extraídas 17 características das imagens, tais como distribuição, densidade, forma, e outras.

Os resultados obtidos demonstraram que o desempenho do sistema inteligente evolutivo na classificação é comparável, em termos de precisão, com outros métodos reconhecidos, dentre eles os métodos *K-Nearest Neighbor algorithm* (KNN), *Classification and Regression Tress* (CART) e rede neural artificial probabilística. Entretanto, deve-se ressaltar que, enquanto os outros métodos requerem o acesso a todo o conjunto de dados (imagens) disponível, o sistema inteligente evolutivo é capaz de construir o classificador de forma evolutiva e de modo *on-line*, o que lhe confere maior aplicabilidade no processo em questão.

Previsão

A previsão de valores em uma série temporal consiste em utilizar valores passados da série até o instante t para prever o valor em algum ponto futuro $t + P$, onde P expressa o horizonte de previsão. A previsão de valores tem aplicações nas mais diversas áreas, como por exemplo, a previsão de índices financeiros, a previsão de consumo de eletricidade ou prognóstico de falhas em processos industriais, dentre outras.

Várias técnicas para previsão de valores têm sido propostas na literatura, sendo as abordagens clássicas baseadas em modelos dinâmicos ou modelos estocásticos. Todavia, os métodos clássicos apresentam algumas dificuldades para previsão do comportamento de sistemas dinâmicos não estacionários complexos. Diferentemente dos métodos clássicos, a previsão de valores baseada em sistemas *fuzzy* ou *neurofuzzy*, quando aplicada à problemas complexos, permite obter maior precisão e um menor tempo computacional (Chang et al., 2007).

As características dos sistemas inteligentes evolutivos os tornam potencialmente aplicáveis a problemas de previsão, especialmente nos casos onde há necessidade de treinamento do preditor em modo *on-line* e operação em tempo real. Um exemplo de aplicação dos sistemas inteligentes evolutivos em previsão é apresentado em Wang and Vrbanek (2008), onde um sistema *fuzzy* evolutivo é empregado para prognóstico de falhas em um sistema mecânico sujeito a variações de carga e falhas no sistema de transmissão.

O sistema mecânico em questão consiste de um motor de corrente alternada (C.A.) controlado por um controlador de velocidade acoplando a um conjunto de engrenagens, que por sua vez está acoplado a uma carga tipo freio magnético. O sistema conta com sensores óticos e de vibração cujos sinais, após serem

pré-condicionados, são enviados para armazenamento e processamento em um computador. Falhas podem ser simuladas no sistema mecânico, mais especificamente falhas nas engrenagens. Os componentes do sistema mecânico são: controlador de velocidade; motor C.A.; sensor ótico; caixa de engrenagem cônicas; disco de balanço; engrenagens de teste; acelerômetros e cargas. O tempo de monitoramento foi definido em 10 minutos, ou seja, o preditor é aplicado a cada 10 minutos para realizar a previsão de valores futuros do índice de monitoramento (*beta kurtosis*) em três passos a frente, ou seja, 30 minutos. Os testes foram realizados em um intervalo de funcionamento de aproximadamente 350 horas, sendo que durante o funcionamento foram inseridas falhas no conjunto de engrenagens.

Os resultados obtidos pelo sistema inteligente evolutivo na previsão de valores futuros do índice de monitoramento, quando comparados a outras abordagens, demonstraram que o preditor baseado em sistema *fuzzy* evolutivo proposto possui desempenho superior aos outros métodos, uma vez que o treinamento em modo *on-line* permite a captura do comportamento dinâmico do sistema com rapidez e, conseqüentemente, o monitoramento das características do sistema com precisão.

Capítulo 4

Algoritmo de Agrupamento Recursivo e Detecção de *Drift*

4.1 Introdução

Para implementação de sistemas inteligentes evolutivos, uma das abordagens mais utilizadas é o uso de algoritmos de agrupamento recursivos não-supervisionados para o aprendizado da estrutura e parâmetros do sistema. A partir das amostras de dados de entrada obtidas em modo *on-line*, o algoritmo realiza o agrupamento dos dados no espaço de entrada, definindo o centro de cada grupo e, em alguns casos, o raio do grupo (ou zona de influência). Durante o processo evolutivo, o algoritmo pode criar novos grupos, atualizar grupos existentes, ou ainda eliminar grupos redundantes. Os modelos propostos em Angelov and Filev (2004); Kasabov and Song (2002); Lemos et al. (2011a); Leng et al. (2005); Lima et al. (2010); Lughofer (2008) são exemplos de sistemas inteligentes evolutivos baseados em algoritmos de agrupamento recursivos.

Normalmente, os sistemas inteligentes evolutivos baseados em algoritmos de agrupamento recursivos adotam um mecanismo para atualização da estrutura e parâmetros do sistema com base em alguma medida de similaridade entre as amostras de dados de entrada e os grupos existentes. Diversos tipos de medida de similaridade são empregadas, como distância entre a amostra e os centros de grupo, potencial informativo, dentre outras. Esse tipo de mecanismo, apesar

de funcional, pode levar a uma definição errônea da estrutura caso *outliers* ou ruído estejam presentes nos dados de entrada, o que geralmente ocorre com dados adquiridos por meio de sensores em ambientes industriais. Na literatura são encontrados alguns modelos de sistemas inteligentes evolutivos que adotam mecanismos diferentes para atualização da estrutura e parâmetros do sistema, como os modelos propostos em Lima et al. (2010) e em Lemos et al. (2011a), e dessa forma tendem a ser mais robustos.

O presente capítulo apresenta o algoritmo de agrupamento recursivo proposto nesse trabalho. O algoritmo utiliza um novo mecanismo de atualização do agrupamento e consiste numa versão recursiva do algoritmo de agrupamento *Gustafson-Kessel* (GK) (Gustafson and Kessel, 1979) com a incorporação do algoritmo *Drift Detection Method* (DDM) (Gama et al., 2004). As seções 4.2 e 4.3 apresentam os conceitos teóricos relacionados ao algoritmo de agrupamento recursivo e método de detecção de *drift*, respectivamente, e a seção 4.4 apresenta o algoritmo proposto.

4.2 Algoritmo de Agrupamento Recursivo

Na área de reconhecimento de padrões, os algoritmos de agrupamento estão entre as ferramentas mais úteis para soluções de problemas que envolvem a análise de dados não-rotulados, ou aprendizado não-supervisionado (Duda et al., 2000). Ao longo das últimas décadas, muitos de algoritmos de agrupamento foram propostos na literatura (Jain, 2010), mas a maioria deles são baseados no conceito de aprendizado *off-line*, ou aprendizado em lote, onde pressupõe-se que todo o conjunto de dados está disponível previamente. Entretanto, para muitas aplicações, os dados são adquiridos em tempo real, exigindo um aprendizado *on-line*.

Em contraste com os algoritmos de agrupamento para aprendizado *off-line*, que encontram os grupos empregando uma estratégia iterativa, como os bem conhecidos algoritmos *K-means* e *Fuzzy C-Means* (FCM) (Jain, 2010), os algoritmos de agrupamento para aprendizado *on-line* baseiam-se na estratégia recursiva, que permite ao algoritmo encontrar os grupos executando um único passo a cada amostra de dados de entrada. Na literatura recente foram propostos diversos algoritmos com esta abordagem, como por exemplo o *Evolving*

Clustering Method (ECM) (Kasabov and Song, 2002), o *eClustering* (Angelov and Filev, 2004), e o *Evolving Vector Quantization* (eVQ) (Lughofer, 2008). Uma característica comum desses algoritmos é o fato de assumirem que a forma dos grupos é hipersférica, o que pode ser um fator limitador em aplicações reais, onde os grupos podem não ter forma hipersférica e ainda ter diferentes formas e orientação no espaço.

O algoritmo de agrupamento GK é um poderoso algoritmo de agrupamento que, ao contrário de outros, permite a identificação de grupos de diferentes formas e com diferentes orientações no espaço. Esse algoritmo emprega uma técnica de adaptação da métrica de distância à forma de cada grupo, através da estimação da matriz de dispersão do grupo e seu uso para encontrar uma matriz de adaptação da norma. Além disso, o algoritmo GK tem a vantagem de ser relativamente insensível à escala dos dados e à inicialização da matriz de partição (Filev and Georgieva, 2010). Diversas aplicações para o algoritmo GK têm sido propostas na literatura, tais como previsão de séries temporais, modelagem de sistemas dinâmicos, diagnóstico e prognóstico de falhas.

Diferentemente de muitos algoritmos de agrupamento que empregam a distância Euclidiana como medida de similaridade, o algoritmo GK emprega uma distância tipo Mahalanobis, que permite a identificação de grupos de diferentes formas e com diferentes orientação no espaço. Nesse algoritmo a distância é definida da seguinte forma:

$$d_{ik}^2 = (x_k - v_i)A_i(x_k - v_i)^T \quad (4.1)$$

onde d_{ik}^2 representa a distância entre a amostra de dados de entrada $x_k = [x_{k1}, \dots, x_{kn}]$, $k = 1, \dots, N$, e os centros de grupo v_i , $i = 1, \dots, c$. N é o número de amostras de dados; n é o número de dimensões dos dados e c é o número de grupos.

A matriz A_i , $i = 1, \dots, c$, define a forma e orientação de cada grupo no espaço, sendo calculada como:

$$A_i = [\rho_i \det(F_i)]^{1/n} F_i^{-1} \quad (4.2)$$

onde ρ_i , $i = 1, \dots, c$, é o volume do grupo (usualmente fixado como um para todos os grupos).

A matriz $F_i, i = 1, \dots, c$, é a matriz de dispersão do grupo, sendo calculada como:

$$F_i = \frac{\sum_{k=1}^N (u_{ik})^m (x_k - v_i)^T (x_k - v_i)}{\sum_{k=1}^N (u_{ik})^m} \quad (4.3)$$

onde $u_{ik}, i = 1, \dots, c, k = 1, \dots, N$, é o grau de pertinência da amostra de dados de entrada e m é o coeficiente *fuzzy*, sendo geralmente utilizado $m = 2$.

No algoritmo GK, um processo iterativo é utilizado para determinar os parâmetros dos grupos (o centro do grupo v_i e a matriz de dispersão F_i), que por sua vez são utilizados para definir a distância d_{ik}^2 e o grau de pertinência u_{ik} . Esse processo é encerrado quando um determinado critério de convergência é alcançado. Mas, como discutido no início dessa seção, quando a aplicação requer a definição do agrupamento em modo *on-line*, é necessário que o algoritmo de agrupamento seja recursivo. Em Filev and Georgieva (2010) é proposto uma versão estendida do algoritmo GK denominada *evolving GK-like algorithm* (eGKL), que estima o número de grupos e realiza a adaptação dos seus parâmetros recursivamente, mantendo as vantagens do algoritmo GK, como a capacidade de identificar grupos com forma e orientação genéricas.

No algoritmo eGKL, como o número de grupos não é conhecido *a priori*, para estimar o número de grupos é utilizada a estratégia de avaliar, a cada nova amostra de dados, se a mesma pertence ou não a um grupo existente. Caso a amostra de dados atual pertença a um grupo já definido, então os parâmetros desse grupo (centro do grupo e matriz de dispersão) são atualizados. Caso a amostra de dados não pertença a nenhum dos grupos, então ela é utilizada para definir um novo grupo. Para avaliar a similaridade entre cada nova amostra de dados e os grupos existentes, o algoritmo eGKL emprega a distância de Mahalanobis, definida da seguinte forma:

$$D_{ik}^2 = (x_k - v_i)^T F_i^{-1} (x_k - v_i) \quad (4.4)$$

Nessa estratégia, a amostra de dados atual pertence a um grupo existente quando a distância até o centro do grupo é menor que o raio do grupo. No algoritmo eGKL, para determinação do raio de cada grupo, é empregada uma abordagem inspirada nos conceitos e técnicas do controle estatístico de processos. Nessa abordagem, considera-se que a amostra de dados pertence a um

grupo quando a seguinte relação for satisfeita:

$$D_{ik}^2 < \chi_{n,\beta}^2 \quad (4.5)$$

onde $\chi_{n,\beta}^2$ é o valor de uma distribuição qui-quadrado com n graus de liberdade e nível de confiança igual a $(1 - \beta)$. Nesse caso, os graus de liberdade n correspondem ao número de dimensões dos dados de entrada. Essa abordagem tem a vantagem de não requerer a escolha de um limiar para definir o raio do grupo, como em outros algoritmos de agrupamento recursivos, sendo o raio do grupo determinado pela constante $\chi_{n,\beta}^2$. Outra vantagem é evitar o problema denominado de “maldição da dimensionalidade” (Hastie et al., 2001), que consiste no aumento da distância entre dois pontos adjacentes com o aumento da dimensão do espaço, uma vez que a constante $\chi_{n,\beta}^2$ é proporcional à dimensão dos dados de entrada.

No algoritmo eGLK, se a condição (4.5) for satisfeita, então isso significa que a amostra de dados atual pertence ao grupo mais próximo, de forma que seus parâmetros são atualizados da seguinte forma:

$$v_q = v_q + \alpha(x_k - v_q) \quad (4.6)$$

$$F_q = F_q + \alpha((x_k - v_q)^T(x_k - v_q) - F_q) \quad (4.7)$$

onde $q = \arg \min_{i=1,\dots,c} (D_{ik}^2)$ e $\alpha \in [0, 1]$ é a taxa de aprendizado. Caso contrário, a amostra de dados atual não pertence a nenhum dos grupos existentes, então é criado um novo grupo com seus parâmetros inicializados da seguinte forma:

$$v_{c+1} = x_k; \quad F_{c+1} = F_{init} \quad (4.8)$$

onde F_{init} é a matriz de dispersão inicial, que pode ser calculada a partir das amostras de dados de entrada iniciais, ou inicializada como uma matriz diagonal $F_{init} = \gamma I$; onde I é uma matriz identidade de tamanho n e γ é um número positivo pequeno.

Para aumentar a robustez a *outliers* do algoritmo eGKL, os autores propõem um mecanismo baseado no número de amostras de dados que pertencem a um grupo. Nesse mecanismo, se o número de amostras de dados M_i , $i = 1, \dots, c$, já atribuídas a um grupo for menor que M_{min} (um número mínimo escolhido

inicialmente), mesmo que a nova amostra de dados não pertença a esse grupo, os parâmetros do grupo são atualizados conforme (4.6) e (4.7). Apesar de funcional, esse mecanismo depende da escolha adequada desse parâmetro do algoritmo ao problema em questão, o que pode ser difícil para problemas que não se têm conhecimento *a priori*.

4.3 Detecção de *Drift*

Um método de detecção de *drift* consiste em um método para detectar uma mudança gradual no contexto dos dados de entrada (Gama et al., 2004). Por contexto, entende-se como um conjunto de dados gerados quando o processo está estacionário. Dessa forma, um método para detecção de *drift* é um método para detectar os momentos em que ocorrem mudanças no contexto dos dados. Quando um novo contexto é identificado, isto significa que o modelo atual está desatualizado e é necessário um re-aprendizado do modelo utilizando as informações relevantes presentes no novo contexto. Métodos de detecção de *drift* são indicados para aplicações que envolvam aprendizado de máquina, onde os algoritmos a serem empregados lidam com problemas do mundo real, em ambientes dinâmicos, não estacionários e complexos. Nessas aplicações, grandes quantidades de informação são fornecidas em um fluxo contínuo de dados em alta velocidade e que apresentam variações ao longo do tempo como, por exemplo, o monitoramento em tempo real de plantas industriais (Sebastião and Gama, 2009). Dessa forma, os algoritmos de aprendizado devem ser capazes de acompanhar o comportamento do sistema dinâmico em questão e adaptar o modelo interno conforme ocorram mudanças.

Diversos métodos para detecção de *drift* foram propostos na literatura e podem ser classificados em duas categorias: métodos que adaptam o aprendizado em intervalos regulares, independentemente da ocorrência de mudanças, e métodos que primeiro detectam as mudanças e posteriormente adaptam o aprendizado a essas mudanças (Sebastião and Gama, 2009). Na primeira categoria, os métodos podem utilizar janelas de tempo de tamanho fixo ou ponderação dos dados de acordo com sua idade e/ou utilidade (Klinkenberg, 2004; Klinkenberg and Joachims, 2000; Maloof and Michalski, 2000). Quando são utilizadas janelas de tempo de tamanho fixo, a cada intervalo de tempo, o aprendizado é

realizado somente com as amostras de dados incluídas na janela. Uma dificuldade inerente aos métodos que utilizam janelas de tamanho fixo é a escolha do tamanho da janela apropriado para cada problema. Na segunda categoria, os métodos utilizam alguns indicadores monitorados ao longo do tempo para detectar mudanças, como por exemplo medidas de performance, distribuição dos dados, ou propriedades dos dados (Gama et al., 2004; Kifer et al., 2004; Sebastião et al., 2008). Se durante o monitoramento do processo ocorrer a detecção de *drift*, ações são tomadas para adaptar o modelo às mudanças que ocorreram, como no caso de utilização de janelas de tempo de tamanho adaptativo, onde as ações incluem ajustar o tamanho da janela de acordo com a extensão da mudança de contexto.

Dentre diversos métodos para detecção de *drift* propostos na literatura, o algoritmo DDM emprega um método simples, de aplicação direta e computacionalmente eficiente para detectar os momentos em que ocorrem mudanças no contexto dos dados. Consiste em um método de detecção de *drift* independente e pode ser incorporado a qualquer algoritmo de aprendizado, aumentando sua eficiência em problemas que envolvam modelos dinâmicos não estacionários. O algoritmo DDM, que enquadra-se na segunda categoria dos métodos que para detecção de *drift*, é baseado no monitoramento do número de erros produzidos pela aprendizagem do modelo durante a predição. O método utiliza a distribuição binomial para determinar a forma geral da probabilidade da variável aleatória que representa o número de erros de predição em uma amostra de n dados de entrada. Para a k -ésima amostra da sequência de dados de entrada, a taxa de erro é a probabilidade do erro de predição p_k com desvio padrão $s_k = \sqrt{p_k(1 - p_k)/k}$. De acordo com o modelo de aprendizado *Probably Approximately Correct* (PAC) (Mitchell, 1997), a taxa de erro p_k do algoritmo de aprendizado decresce com o aumento de dados de entrada se a distribuição dos dados é estacionária. Um aumento significativo na taxa de erro sugere uma mudança de contexto e, nesse caso, supõe-se que o modelo atual é inadequado e deve ser atualizado.

No algoritmo DDM, são armazenados os valores de p_k e s_k quando $p_k + s_k$ atingem o menor valor durante o processo, obtendo p_{min} e s_{min} , e verificada a seguinte condição:

$$p_k + s_k \geq p_{min} + z_1 \cdot s_{min} \quad (4.9)$$

onde z_1 é o nível de alerta. Se esta condição for satisfeita, então as amostras de

dados são armazenadas em memória e é verificada a seguinte condição:

$$p_k + s_k \geq p_{min} + z_2 \cdot s_{min} \quad (4.10)$$

onde z_2 é o nível de *drift*. Se esta condição for satisfeita, então considera-se que há uma mudança de contexto, o modelo induzido pelo algoritmo de aprendizagem deve ser atualizado com as amostras de dados armazenadas desde o instante em que foi atingido o nível de alerta. Nesse caso, os valores de p_{min} e s_{min} devem ser reinicializados. Os autores sugerem os valores $z_1 = 2$ e $z_2 = 3$, o que equivale ajustar o nível de confiança para alerta a 95% e o nível de confiança para *drift* a 99%. É possível que o erro aumente e, após atingir o nível de alerta, o mesmo diminua a níveis baixos. Essa situação corresponde a um alarme falso, onde não há mudança de contexto, portanto, não é necessária nenhuma ação e as amostras de dados armazenadas na memória não são mais necessárias.

Utilizando o algoritmo DDM incorporado a um algoritmo de aprendizado é possível manter o modelo do sistema dinâmico continuamente ajustado ao contexto atual. Por exemplo, incorporando esse algoritmo a um algoritmo de agrupamento recursivo, a definição dos grupos pode ser ajustada conforme a detecção da mudança de contexto, e não apenas em função de uma medida de similaridade entre as amostras de dados e os grupos existentes. Esse mecanismo confere ao algoritmo de agrupamento maior robustez a *outliers* e ruído em aplicações onde é necessário o aprendizado *on-line* de modelos dinâmicos e não estacionários.

4.4 Algoritmo proposto

O algoritmo de agrupamento recursivo não supervisionado proposto nesse trabalho une as vantagens do algoritmo GK recursivo, especialmente a habilidade de identificar grupos de diferentes formas e com diferentes orientações no espaço em modo *on-line*, com o método eficaz de detecção de mudança no contexto dos dados de entrada do algoritmo DDM, para realizar a atualização do agrupamento. No algoritmo proposto, a atualização do agrupamento não só depende da medida de similaridade, mas também do monitoramento de mudanças no fluxo de dados de entrada, o que confere ao algoritmo maior robustez à presença de *outliers* e ruído. Um mecanismo de união de grupos também

foi incorporado ao algoritmo de agrupamento para permitir a eliminação de grupos redundantes e, conseqüentemente, maior precisão na identificação do agrupamento.

Assumindo que não há nenhuma informação *a priori* sobre o agrupamento e nenhum conjunto de amostras de dados de entradas coletadas previamente, o algoritmo proposto inicia associando o centro do primeiro grupo v_1 à primeira amostra de dados de entrada x_1 , e inicializando a matriz de dispersão F_1 , a taxa de aprendizado α_1 e o número de amostras M_1 do primeiro grupo da seguinte forma:

$$v_1 = x_1; \quad F_1 = F_{init}; \quad \alpha_1 = \alpha_{init}; \quad M_1 = 1 \quad (4.11)$$

onde $F_{init} = \gamma I$; sendo I uma matriz identidade de tamanho n e γ um número positivo pequeno (valor padrão: $\gamma = 10^{-2}$); $\alpha_{init} \in [0, 1]$ (valor padrão: $\alpha_{init} = 0.5$).

Se todas as amostras de dados de entrada foram processadas, o algoritmo encerra, caso contrário, é obtida uma nova amostra de dados x_k , para $k = 2, 3, \dots$, e calculada a distância entre a amostra de dados e os centros de grupo existentes:

$$D_{ik}^2 = (x_k - v_i)F_i^{-1}(x_k - v_i)^T, \quad i = 1, \dots, c. \quad (4.12)$$

A similaridade entre a amostra de dados atual e os grupos existentes é verificada pela seguinte condição:

$$D_{ik}^2 < \chi_{n,\beta}^2, \quad i = 1, \dots, c. \quad (4.13)$$

Se condição de similaridade (4.13) for satisfeita, a amostra de dados atual pertence ao grupo mais próximo, então os seus parâmetros (centro, matriz de dispersão, taxa de aprendizado e número de amostras do grupo) são atualizados como:

$$v_q = v_q + \alpha_q(x_k - v_q) \quad (4.14)$$

$$F_q = F_q + \alpha_q((x_k - v_q)^T(x_k - v_q) - F_q) \quad (4.15)$$

$$\alpha_q = \frac{\alpha_{init}}{M_q} \quad (4.16)$$

$$M_q = M_q + 1 \quad (4.17)$$

onde $q = \arg \min_{i=1,\dots,c} (D_{ik}^2)$.

Se a condição de similaridade (4.13) não for satisfeita, a amostra de dados atual não pertence a nenhum grupo existente, o algoritmo incrementa uma variável que representa o número de dissimilaridades M_{dis} , em seguida são calculados a probabilidade e o desvio padrão do erro como:

$$M_{dis} = M_{dis} + 1 \quad (4.18)$$

$$p = \frac{M_{dis}}{k} \quad (4.19)$$

$$s = \sqrt{p(1-p)/k} \quad (4.20)$$

Uma vez calculados os valores de p e s , se a seguinte condição for satisfeita:

$$p + s < p_{min} + s_{min} \quad (4.21)$$

então os valores de p_{min} e s_{min} devem ser atualizados:

$$p_{min} = p; \quad s_{min} = s \quad (4.22)$$

Nota-se que, quando o algoritmo inicia, os valores de p_{min} e s_{min} devem ser inicializados como um número positivo. Considerando que, teoricamente, os valores máximos de p e s serão de 1 e 0,5, respectivamente, sugere-se inicializar os parâmetros p_{min} e s_{min} com esses valores.

Para decidir se a amostra de dados atual x_k representa um novo grupo ou é apenas um *outlier*, devem ser avaliadas as condições de alerta e de *drift*. A condição de alerta é verificada como:

$$p + s > p_{min} + z_1 \cdot s_{min} \quad (4.23)$$

onde z_1 é o nível de alerta (valor padrão: $z_1 = 2$). Se o nível de alerta for atingido, então a amostra de dados atual x_k é armazenada em uma janela de amostras de dados W_j , $j = 1, \dots, m$, onde m é o tamanho atual da janela, e em seguida é avaliada a condição de *drift*. Caso contrário, o algoritmo processa a próxima amostra de dados de entrada. A condição de *drift* é verificada como:

$$p + s > p_{min} + z_2 \cdot s_{min} \quad (4.24)$$

onde z_2 é o nível de *drift* (valor padrão: $z_2 = 3$). Se o nível de *drift* foi atingido, então é criado um novo grupo, sendo o centro e matriz de dispersão do novo

grupo determinados pelas amostras armazenadas na janela de dados da seguinte forma:

$$v_{c+1} = \frac{1}{m} \sum_{j=1}^m W_j \quad (4.25)$$

$$F_{c+1} = \text{cov}(W_j), \quad j = 1, \dots, m. \quad (4.26)$$

Os demais parâmetros do novo grupo (taxa de aprendizado e número de amostras do grupo) são inicializados como:

$$\alpha_{c+1} = \alpha_{init}; \quad M_{c+1} = 1 \quad (4.27)$$

Durante a atualização dos grupos, para evitar a formação de grupos redundantes, deve ser verificada a similaridade entre grupos. Para isso, são calculadas as distâncias entre os centros de grupo da seguinte forma:

$$D_{ij}^2 = (v_i - v_j)F_i^{-1}(v_i - v_j)^T, \quad i = 1, \dots, c. \quad j = 1, \dots, c. \quad (4.28)$$

$$D_{ji}^2 = (v_j - v_i)F_j^{-1}(v_j - v_i)^T, \quad j = 1, \dots, c. \quad i = 1, \dots, c. \quad (4.29)$$

Se, para dois grupos i e j existentes, uma das seguintes condições de similaridade for satisfeita:

$$D_{ij}^2 < \chi_{n,\beta}^2, \quad i = 1, \dots, c. \quad j = 1, \dots, c. \quad (4.30)$$

$$D_{ji}^2 < \chi_{n,\beta}^2, \quad j = 1, \dots, c. \quad i = 1, \dots, c. \quad (4.31)$$

então os grupos devem ser unidos. Os grupos a serem unidos podem ser vistos como grupos hiperelipsoidais, definidos pelo vetor de média, matriz de dispersão e número de amostras do grupo, de forma que a combinação dos dois grupos produzem um novo grupo com parâmetros que podem ser calculados da seguinte forma (Kelly, 1994):

$$M_i = M_i + M_j \quad (4.32)$$

$$v_i = \frac{M_i}{M_i + M_j}v_i + \frac{M_j}{M_i + M_j}v_j \quad (4.33)$$

$$F_i = \frac{M_i - 1}{M_i + M_j + 1}F_i + \frac{M_j - 1}{M_i + M_j + 1}F_j + \frac{M_i M_j}{M_i + M_j(M_i + M_j - 1)}(v_i - v_j)^T(v_i - v_j) \quad (4.34)$$

O Algoritmo 1 resume os procedimentos do algoritmo de agrupamento recursivo proposto.

Algoritmo 1: Algoritmo de agrupamento recursivo com detecção de *drift*

Entrada: $x_k, \chi_{n,\beta}^2, F_{init}, \alpha_{init}, z_1, z_2$;

Saída: v_i, F_i ;

Apresentar a primeira amostra de dados x_1 ;

Inicializar o primeiro grupo v_1, F_1 ;

for $k = 2, 3, \dots$ **do**

 Apresentar nova amostra de dados x_k ;

 Calcular D_{ik}^2 para todos os grupos;

 Identificar o grupo mais próximo;

if $D_{qk}^2 < \chi_{n,\beta}^2$ **then**

 | Atualizar do grupo mais próximo;

else

 | Incrementar M_{dis} ;

 | Calcular p e s ;

if $p + s < p_{min} + s_{min}$ **then**

 | Atualizar p_{min} e s_{min} ;

end if

if $p + s > p_{min} + z_1 \cdot s_{min}$ **then**

 | Armazenar x_k na janela de dados W_j ;

end if

if $p + s > p_{min} + z_2 \cdot s_{min}$ **then**

 | Criar novo grupo;

end if

end if

 Calcular D_{ij}^2 e D_{ji}^2 para todos os grupos;

if $D_{ij}^2 < \chi_{n,\beta}^2$ **or** $D_{ji}^2 < \chi_{n,\beta}^2$ **then**

 | Unir os grupos redundantes;

end if

end for

A Figura 4.1 mostra, como exemplo, um agrupamento de dados de duas dimensões formado por 3 grupos principais e outros 3 grupos secundários. Esse agrupamento ilustra um cenário onde os grupos principais representam a informação válida e os grupos secundários representam a informação inválida (*outliers*). As Figuras 4.2 a 4.8 ilustram em sequência a identificação do agrupamento de dados nesse cenário pelo algoritmo de agrupamento proposto. Inicialmente, o algoritmo identifica o primeiro grupo a partir da primeira amostra de dados, determinando os parâmetros iniciais (centro e a dispersão) desse grupo (Figura 4.2). À medida em que novas amostras de dados do primeiro grupo principal são recebidas, o algoritmo atualiza os seus parâmetros (Figura 4.3). Na sequência, o primeiro grupo de amostras de dados correspondentes aos *outliers* é recebido, mas o algoritmo não identifica essas amostras como pertencentes a um novo grupo (Figura 4.4). Em seguida, são recebidas as amostras de dados correspondentes ao segundo grupo principal, o algoritmo cria um novo grupo e atualiza os seus parâmetros (Figura 4.5). Após isso, o segundo grupo de amostras de dados correspondentes aos *outliers* é recebido e, novamente, o algoritmo não identifica essas amostras como pertencentes a um novo grupo (Figura 4.6). As próximas amostras de dados recebidas terceiro grupo principal, o algoritmo cria um novo grupo e atualiza os seus parâmetros (Figura 4.7). E, finalmente, são recebidas as amostras de dados correspondentes ao terceiro grupo de *outliers* e, novamente, o algoritmo não identifica essas amostras como pertencentes a um novo grupo (Figura 4.8).

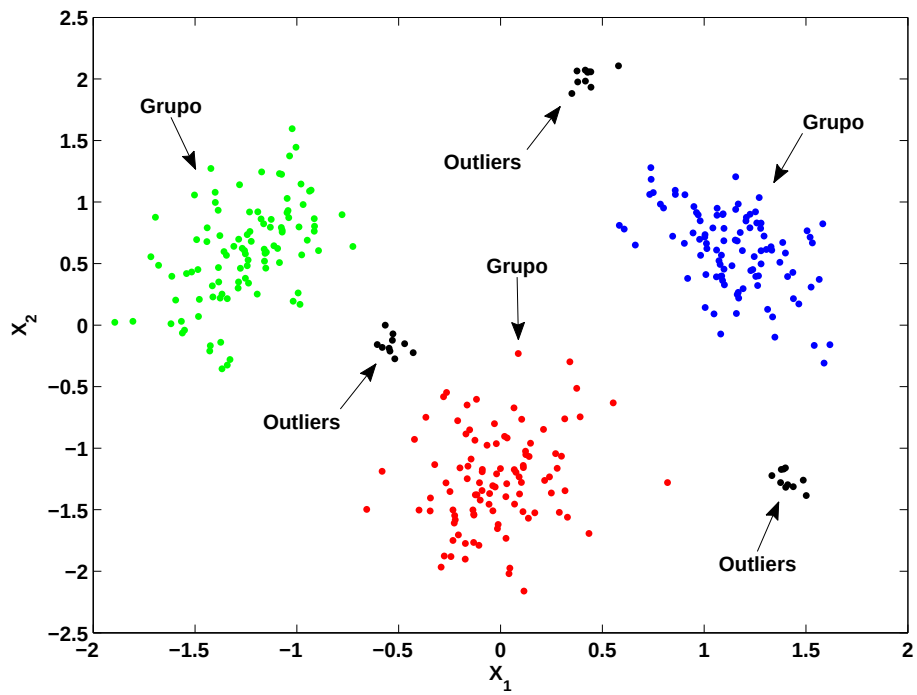


Figura 4.1: Agrupamento de dados.

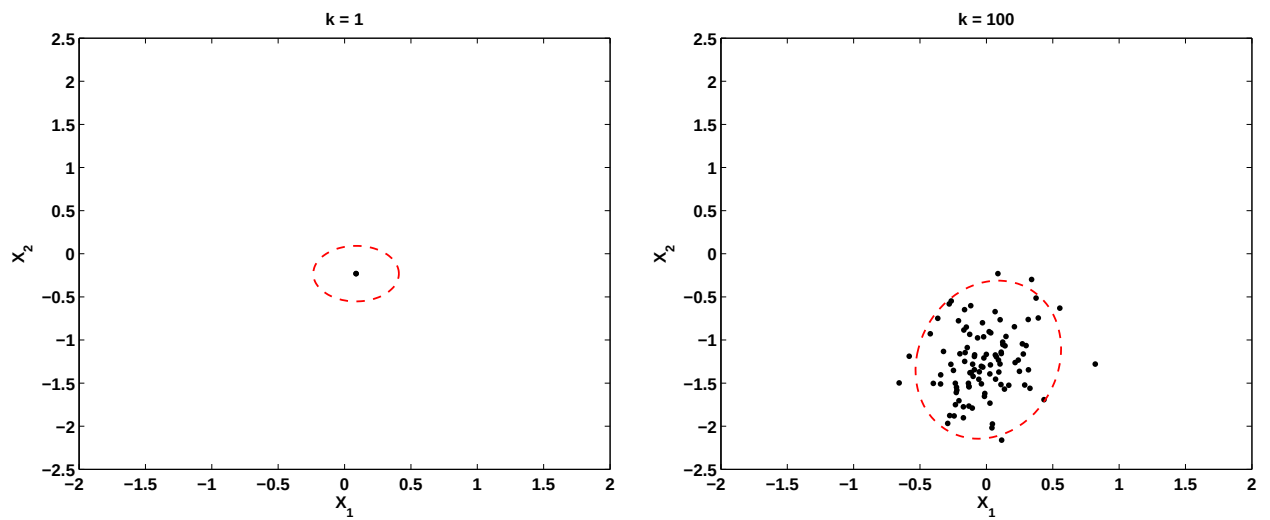


Figura 4.2: Identificação do primeiro grupo de dados. Figura 4.3: Atualização dos parâmetros do primeiro grupo de dados.

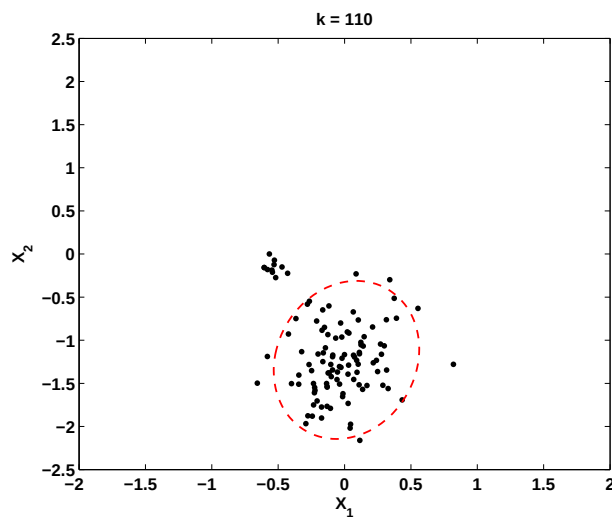


Figura 4.4: Primeiro grupo de *Outliers* não identificados como um grupo.

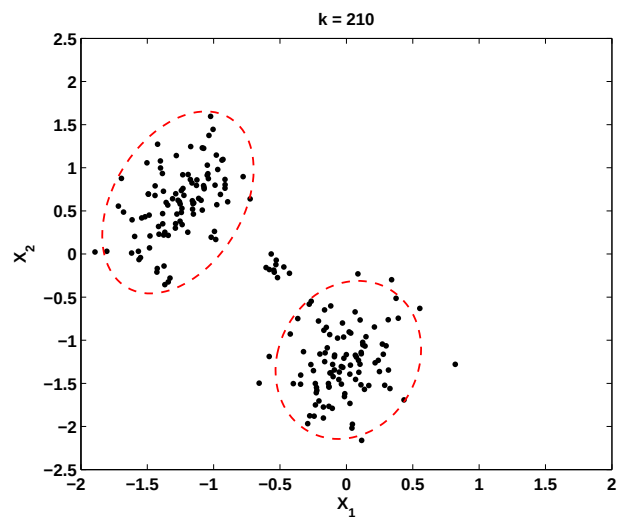


Figura 4.5: Identificação e atualização dos parâmetros do segundo grupo de dados.

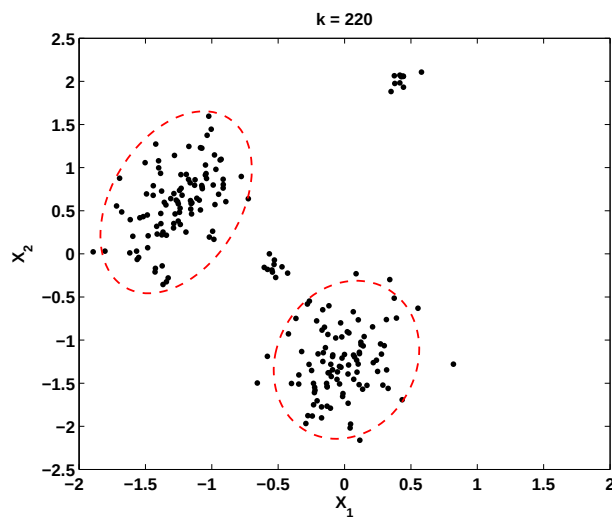


Figura 4.6: Segundo grupo de *Outliers* não identificados como um grupo.

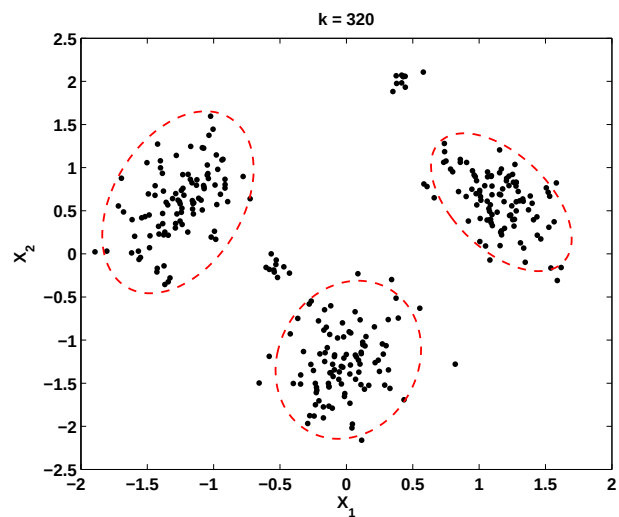


Figura 4.7: Identificação e atualização dos parâmetros do terceiro grupo de dados.

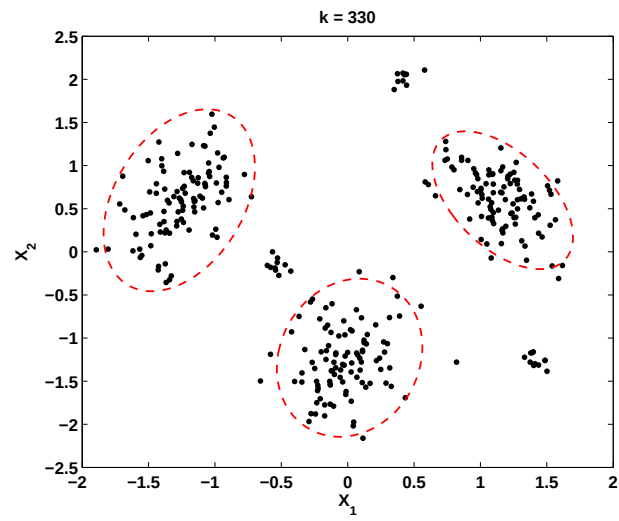


Figura 4.8: Terceiro grupo de *Outliers* não identificados como um grupo.

Capítulo 5

Classificador *Fuzzy* Evolutivo

5.1 Introdução

Os sistemas inteligentes evolutivos, quando aplicados em diagnóstico de falhas, necessitam de maior robustez à presença de *outliers* e ruído nos dados, como normalmente ocorre em ambientes industriais, por exemplo. Dados com presença de *outliers* e ruído levam o algoritmo de agrupamento evolutivo a definir grupos que representam condições de funcionamento ou falha inexistentes, ocasionando posteriormente um aumento no índice de alarme falso e diminuição no índice de isolamento de falha.

O algoritmo de agrupamento com detecção de *drift* apresentado na seção 4.4 emprega um novo mecanismo para atualização do agrupamento, que une as vantagens do algoritmo GK recursivo, com o método eficaz de detecção de mudança no contexto dos dados de entrada do algoritmo DDM, para realizar a atualização do agrupamento. Com essa abordagem, o algoritmo de agrupamento recursivo permite a implementação de sistemas inteligente evolutivos com capacidade de aprendizado contínuo e incremental em modo *on-line* e em tempo real, além de maior robustez a *outliers* e ruído.

Nesse capítulo é apresentado o classificador *fuzzy* evolutivo proposto. A partir da definição dos grupos pelo algoritmo de agrupamento recursivo, a base de regras do classificador *fuzzy* evolutivo é atualizada, utilizando os centros e as matrizes de dispersão dos grupos encontrados como parâmetros dos antecedentes das regras *fuzzy*. Nos antecedentes das regras são empregadas funções

de pertinência Gaussianas multivariáveis, visando evitar a perda de informação em caso de interação entre as variáveis de entrada. A seção 5.2 apresenta os conceitos teóricos relacionados aos classificadores *fuzzy*, a seção 5.3 apresenta o classificador proposto e a seção 5.4 apresenta a aplicação do classificador proposto em diagnóstico de falhas de sistemas dinâmicos.

5.2 Classificador *Fuzzy*

A utilização de algoritmos de classificação de padrões está presente em muitas aplicações atuais, tais como reconhecimento de impressão digital em sistemas de segurança, reconhecimento de escrita em computadores com telas sensíveis ao toque, identificação de sequências de DNA em softwares de diagnóstico médico e diagnóstico de falhas em equipamentos industriais. Nesse contexto, a classificação de padrões consiste na tarefa de, a partir de um conjunto de dados “brutos”, atribuir uma classe ou categoria a cada amostra de dados (Duda et al., 2000).

Em diversas aplicações os algoritmos de classificação de padrões baseados em regras *fuzzy* têm sido utilizados devido às suas vantagens em relação aos algoritmos de classificação de padrões clássicos (Duda et al., 2000), especialmente pelo bom desempenho de predição em problemas reais e pela boa transparência na forma de regras linguísticas (Jang et al., 1997), que permitem uma fácil compreensão da dependência entre as características dos padrões. A arquitetura típica de um classificador *fuzzy* consiste em conjunto de regras *fuzzy* SE ... ENTÃO, definidas como:

$$R_i : \text{SE } x_1 \text{ É } \mu_{i1} \text{ E } \dots \text{ E } x_n \text{ É } \mu_{in} \text{ ENTÃO } y_i = L_i \quad (5.1)$$

onde R_i é a i -ésima regra *fuzzy*, $x = [x_1, \dots, x_n]$ são as variáveis/padrões de entrada de dimensionalidade n ; $\mu_i = [\mu_{i1}, \dots, \mu_{in}]$ são os conjuntos *fuzzy* dos antecedentes da i -ésima regra *fuzzy*; y_i é a saída; L_i é a saída correspondente ao rótulo da classe, $L_i \in [1, \dots, K]$, com K sendo o número de classes.

A classificação de cada nova amostra de dados de entrada x_k é obtida atribuindo-se a ela o rótulo da classe correspondente à regra de maior grau de ativação, determinada da seguinte forma:

$$y_i = L_{i^*} \quad (5.2)$$

onde $i^* = \arg \max_{1 \leq i \leq R} (\tau_i)$, sendo R o número de regras *fuzzy* e τ_i o grau de ativação da i -ésima regra, definido por uma *t-norma*, expressa normalmente como um operador produto:

$$\tau_i = \prod_{j=1}^n \mu_{ij}(x_j) \quad (5.3)$$

onde μ_{ij} são as funções de pertinência dos conjuntos *fuzzy* definidas por Gaussianas:

$$\mu_{ij} = e^{-\frac{1}{2} \left(\frac{(x_j - v_{ij})^2}{\sigma_{ij}^2} \right)} \quad (5.4)$$

onde v_{ij} e σ_{ij}^2 representam o centro e a dispersão das funções de pertinência, respectivamente.

Para implementação dessa arquitetura de classificador *fuzzy* normalmente se utiliza um algoritmo de agrupamento de dados para realizar o agrupamento no espaço de entrada, e então são criadas regras utilizando-se conjuntos *fuzzy* unidimensionais (ou univariáveis), gerados a partir da projeção dos grupos encontrados no eixo de cada variável. Segundo Lemos et al. (2011a), essa abordagem pode levar à perda de informação, caso haja interação entre as variáveis, e para evitar isso, os autores propõem a utilização de funções de pertinência Gaussianas multivariáveis para representar os conjuntos *fuzzy* dos antecedentes de cada regra. Essas funções de pertinência são descritas por:

$$H(x) = e^{-\frac{1}{2}((x-v)\Sigma^{-1}(x-v)^T)} \quad (5.5)$$

onde x e v são vetores n -dimensionais; Σ é uma matriz positiva definida simétrica $n \times n$. O vetor central v é definido como o valor modal e representa o valor típico de $H(x)$, e a matriz Σ denota a dispersão e representa o espalhamento de $H(x)$.

Nesse caso, cada grupo encontrado pelo algoritmo de agrupamento é associado a uma regra *fuzzy*, sendo os parâmetros da função de pertinência (vetor central e matriz de dispersão) definidos como parâmetros do grupo correspondente. Com a utilização de funções de pertinência Gaussianas multivariáveis, o classificador *fuzzy* terá um conjunto de regras *fuzzy* definidas como:

$$R_i : \text{SE } x_k \text{ É } A_i \text{ ENTÃO } y_i = L_i \quad (5.6)$$

onde A_i é o conjunto *fuzzy* com função de pertinência Gaussiana multivariável (5.5) da i -ésima regra *fuzzy*, com parâmetros extraídos do grupo correspondente, $x_k = [x_{k1}, \dots, x_{kn}]$.

Nas aplicações do classificador *fuzzy* mais de uma regra pode ser utilizada para descrever uma classe, uma vez que uma dada classe pode depender do contexto da aplicação em si. Nesse caso, apenas uma regra pode não ser suficiente para descrever todas as possíveis variações de uma mesma classe. Dessa forma, o classificador *fuzzy* agrega regras com o mesmo consequente, utilizando uma *s-norma* e o resultado da agregação pode ser interpretado como regras da seguinte forma:

$$\text{SE } x_k \text{ É } A_i \text{ OU SE } x_k \text{ É } A_j \text{ OU... SE } x_k \text{ É } A_k \text{ ENTÃO } y_i = L_i \quad (5.7)$$

O resultado dessa agregação é o grau de pertinência de cada classe conhecida e a classificação de cada nova amostra x_k é definida pela classe com maior grau de ativação.

5.3 Classificador proposto

Em algumas aplicações de classificação de padrões, as amostras de dados não são rotuladas, ou seja, as classes dos dados não são conhecidas *a priori*. Isto exige para implementação do classificador um processo de aprendizado não supervisionado. Além disso, em aplicações onde a classificação de padrões deva ser realizada em tempo real, o aprendizado deve ser realizado em modo *on-line*, a partir de amostras de dados fornecidas ao classificador em um fluxo de dados. Para resolver esses problemas, a solução é o uso de um algoritmo de agrupamento recursivo.

O classificador *fuzzy* evolutivo proposto nesse trabalho é baseado no algoritmo de agrupamento recursivo com detecção de *drift* apresentado na seção 4.4, que permite criar a base de regras *fuzzy* em modo *on-line* e, se necessário, em tempo real, a partir das amostras de dados de entrada. Essa abordagem é diferente daquela empregada nos classificadores *fuzzy* tradicionais que necessi-

tam de algum tipo de treinamento (normalmente supervisionado) realizado em modo *off-line*.

No classificador *fuzzy* evolutivo a base de regras é criada e atualizada simultaneamente ao particionamento do espaço de entrada realizado pelo algoritmo de agrupamento recursivo. Para cada nova amostra de dados de entrada, se um novo grupo é criado, uma nova regra *fuzzy* (5.6) é adicionada à base de regras, utilizando-se o centro e a matriz de dispersão do novo grupo como parâmetros das funções de pertinência Gaussianas multivariáveis dos antecedentes da regra *fuzzy*. O consequente da regra *fuzzy* (a saída que corresponde ao rótulo da classe) deve ser definido por um especialista ou operador do sistema, uma vez que no processo de aprendizado não supervisionado, as amostras de dados de entrada adquiridas em modo *on-line* normalmente não são rotuladas. Se um grupo existente é atualizado, o rótulo da classe correspondente à amostra de dados atual é determinado pelo consequente da regra de maior grau de ativação. Nesse caso, a intervenção do usuário não é necessária, sendo a classificação da amostra de dados realizada automaticamente. Se dois grupos são unidos pelo algoritmo de agrupamento recursivo, as regras *fuzzy* correspondentes também são unidas para representar a mesma classe. Deve-se destacar que, no classificador *fuzzy* evolutivo proposto, tanto o número de regras quanto o número de classes são determinados durante o processo evolutivo, não sendo necessário definir esses parâmetros *a priori*. O Algoritmo 2 resume os passos do classificador *fuzzy* evolutivo proposto.

5.4 Aplicação do classificador proposto em diagnóstico de falhas

A aplicação do classificador *fuzzy* evolutivo em diagnóstico de falhas é ilustrado na Fig. 5.1. As amostras de dados são obtidas do sistema dinâmico em um fluxo contínuo, normalmente fornecidas por sensores que monitoram o processo. Esses dados constituem “dados brutos” que eventualmente requerem o uso de técnicas de pré-processamento para extração de características. A partir das amostras de dados de entrada, o algoritmo de agrupamento recursivo executa o agrupamento na medida em que as amostras são recebidas do sistema dinâmico.

Algoritmo 2: Classificador *Fuzzy* Evolutivo

Entrada: x_k ;

Saída: y_k ;

Inicializar o classificador;

for $k = 1, 2, \dots$ **do**

 Apresentar a nova amostra x_k ao classificador;

 Executar o algoritmo de agrupamento;

if *Novo grupo foi criado* **then**

 Criar nova regra *fuzzy*;

 Definir o rótulo da nova classe pelo especialista/operador do sistema;

$y_k =$ Rótulo da nova classe;

end if

if *Um grupo foi atualizado* **then**

 Atualizar a regra *fuzzy* correspondente;

 Determinar a regra mais ativa;

$y_k =$ Rótulo da regra mais ativa;

end if

if *Dois grupos foram unidos* **then**

 Unir as regras correspondentes;

end if

end for

Inicialmente, não há nenhuma regra e nenhuma classe definida. A cada novo grupo encontrado, o classificador *fuzzy* evolutivo inclui uma nova regra *fuzzy* à base de dados e a associa a uma nova classe. Cada classe será relacionada a uma condição do sistema dinâmico, que representará operação normal ou uma falha. O rótulo da classe, que a define como uma condição de operação normal ou como uma falha específica deverá ser fornecido pelo operador do sistema. Todas as informações necessárias ao diagnóstico, as regras *fuzzy* e os rótulos das classes, são armazenadas em uma base de dados unificada e atualizada durante a utilização do sistema.

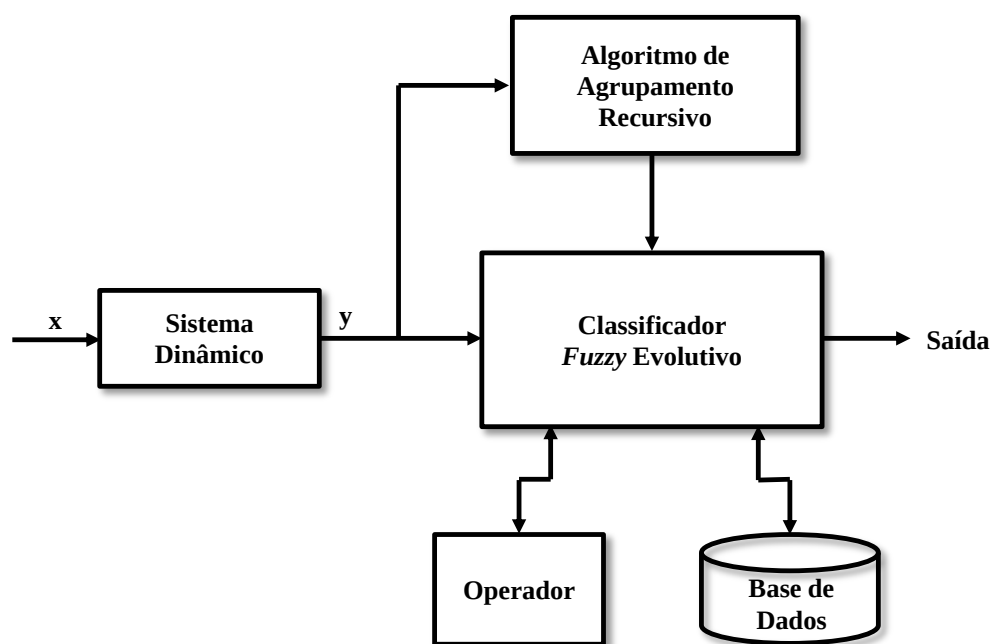


Figura 5.1: Diagnóstico de falhas com Classificador Fuzzy Evolutivo.

Após o período inicial de funcionamento do classificador *fuzzy* evolutivo, a base de dados conterá um conjunto de regras *fuzzy* e os rótulos das classes definidas até então. Quando uma nova amostra de dados recebida pertencer a um grupo existente ou for incluída em um determinado grupo, o classificador atualizará a regra *fuzzy* correspondente e classificará a amostra de dados, conforme os rótulos armazenados na base de dados. Nessa situação, a intervenção do usuário não é necessária, sendo a classificação da condição de operação do sistema dinâmico, como operação normal ou falha, realizada automaticamente.

O classificador *fuzzy* evolutivo proposto nesse trabalho apresenta como principal característica a capacidade de diagnosticar falhas em um sistema dinâmico não estacionário em modo *on-line* e em tempo real. O classificador não necessita do conhecimento do modelo do sistema dinâmico *a priori* e nem do histórico do processo. Empregando o classificador *fuzzy* evolutivo, este permite obter uma base de regras de forma evolutiva e, com o auxílio do operador, aprende a diagnosticar as falhas à medida em que elas ocorrem. Dessa forma, classificador é capaz de se adaptar ao sistema dinâmico, tornando possível o diagnóstico de falhas não conhecidas previamente.

A Figura 5.2 mostra um aplicativo desenvolvido para demonstração do funcionamento do classificador *fuzzy* evolutivo proposto neste trabalho. O aplicativo obtém as amostras de dados de entrada em modo *on-line* a partir de um arquivo gerado previamente, apresentando a identificação do agrupamento na janela “Dados de Entrada”. Os rótulos das classes obtidos no dados de entrada são apresentados na janela “Saída Desejada”, enquanto os rótulos das classes obtidas na saída do classificador são apresentados na janela “Saída Estimada”. Dessa forma é possível observar simultaneamente a identificação do agrupamento de entrada realizado pelo algoritmo de agrupamento e a classificação das amostras de dados realizada pelo classificador.

As Figuras 5.3 a 5.16 mostram em sequência a classificação das amostras de dados pelo classificador *fuzzy* evolutivo. Inicialmente, como o classificador inicia sem nenhuma regra *fuzzy*, a primeira amostra de dados recebida define a primeira regra, sendo o rótulo da classe informado pelo operador (Figura 5.3). À medida em que novas amostras da mesma classe são recebidas, os parâmetros dos antecedentes da regra (funções de pertinência Gaussianas multivariáveis) são atualizados a partir dos parâmetros do grupo (vetor central e matriz de dispersão) (Figura 5.4). Na sequência, amostras de dados pertencentes a uma segunda classe são recebidas, e o classificador define a segunda regra, sendo o rótulo da classe informado pelo operador (Figura 5.5). As novas amostras de dados recebidas da mesma classe irão atualizar a regra correspondente (Figura 5.6). Em seguida, são recebidas amostras de dados pertencentes a uma terceira classe e, novamente, o rótulo da classe informado pelo operador (Figura 5.7). A atualização da regra ocorre conforme as novas amostras de dados da mesma classe são recebidas (Figura 5.8). Após isso, amostras de dados pertencentes a uma quarta classe são recebidas, e o classificador define a quarta regra, sendo

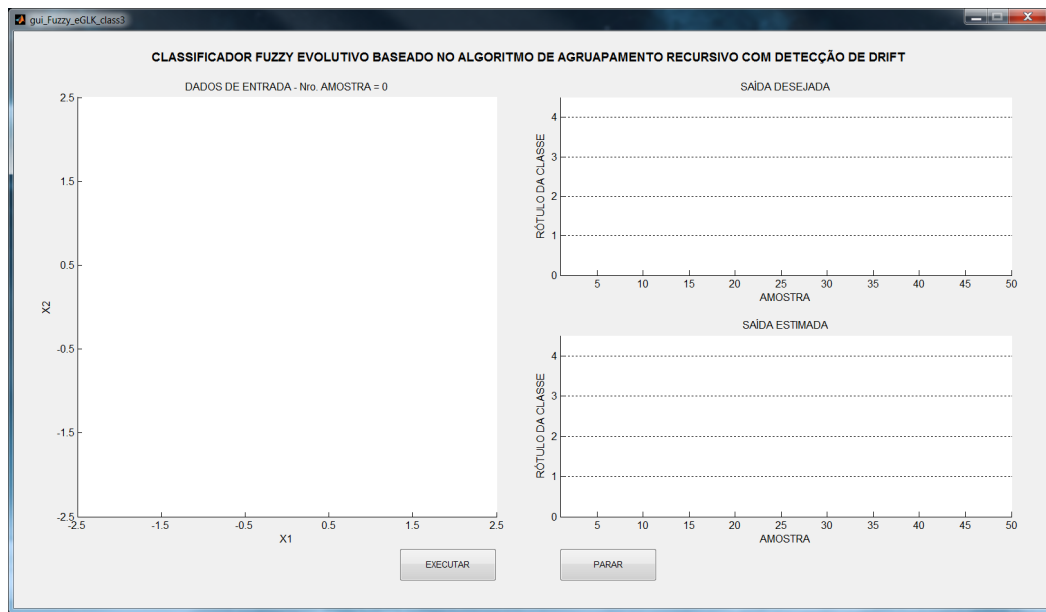


Figura 5.2: Aplicativo desenvolvido para demonstração de funcionamento do classificador *fuzzy* evolutivo.

o rótulo da classe informado pelo operador (Figura 5.9). As novas amostras de dados recebidas da mesma classe irão atualizar a regra correspondente (Figura 5.10). E, finalmente, são recebidas amostras de dados pertencentes a uma quinta classe e, novamente, o rótulo da classe é informado pelo operador (Figura 5.11). A atualização da regra ocorre conforme as novas amostras de dados da mesma classe são recebidas (Figura 5.12). Com as cinco regras *fuzzy* e seus respectivos rótulos definidos, as próximas amostras de dados recebidas pertencentes a qualquer uma das cinco classes são classificadas automaticamente, sem intervenção do operador (Figuras 5.13 a 5.17). Na sequência de figuras apresentadas é possível observar que, quando as amostras de dados de uma classe são recebidas pela primeira vez, a saída do classificador fornece o rótulo com um atraso em relação à saída desejada. Mas, quando as amostras de dados de uma classe são recebidas pela segunda vez, a saída do classificador fornece o rótulo da classe praticamente no mesmo instante da saída desejada.

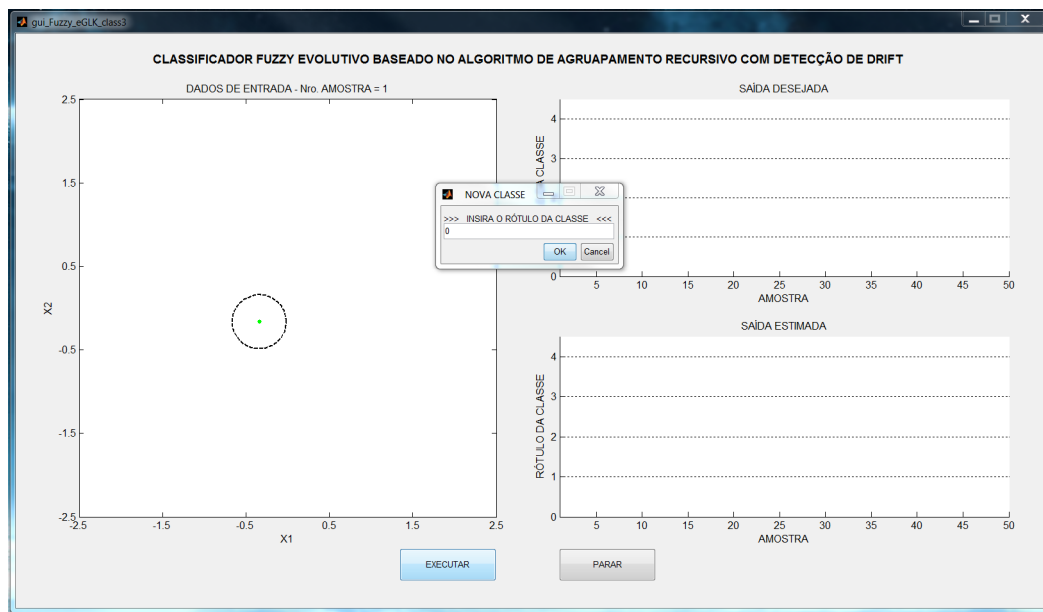


Figura 5.3: Detecção de nova classe e criação da primeira regra com rótulo da classe fornecido pelo operador.

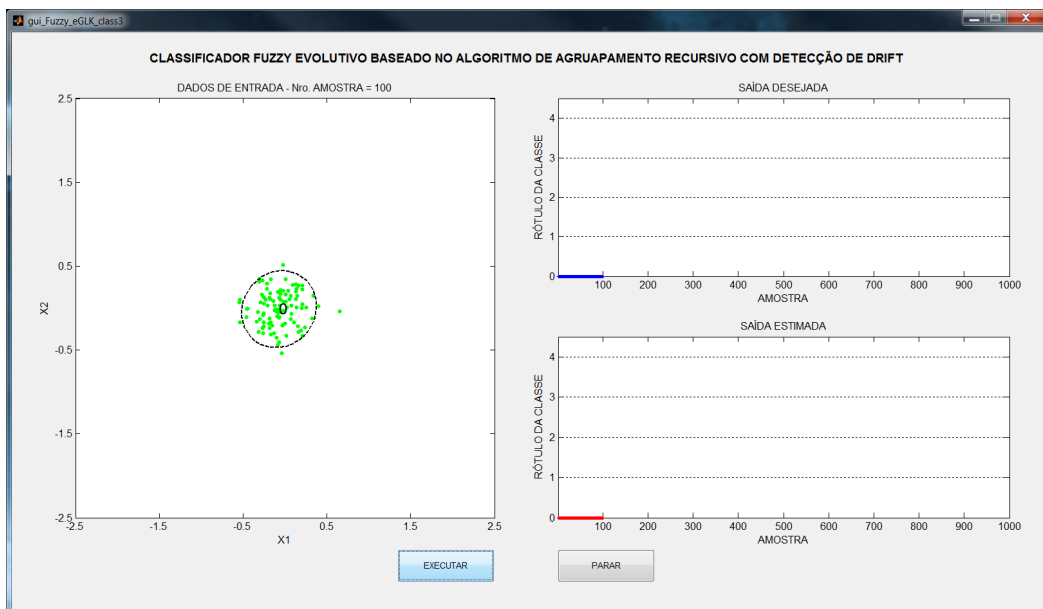


Figura 5.4: Atualização dos parâmetros da primeira regra.

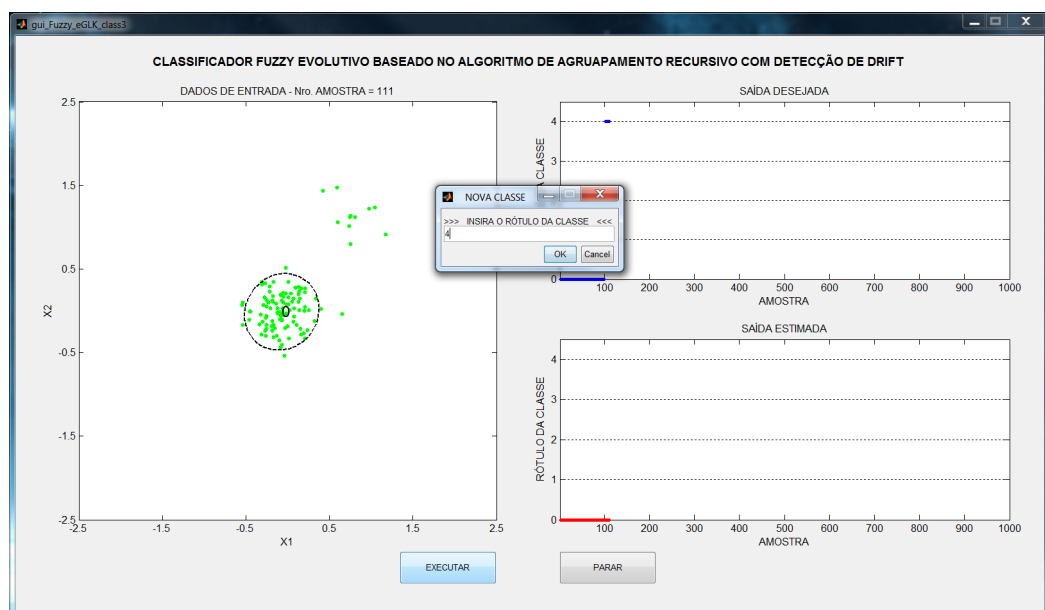


Figura 5.5: Detecção de nova classe e criação da segunda regra com rótulo da classe fornecido pelo operador.

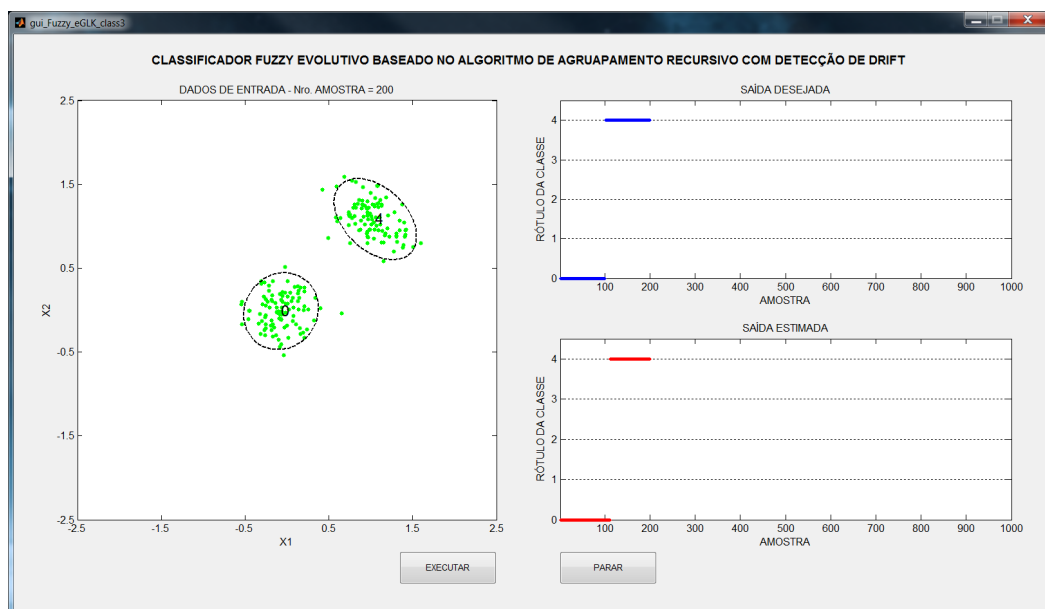


Figura 5.6: Atualização dos parâmetros da segunda regra.

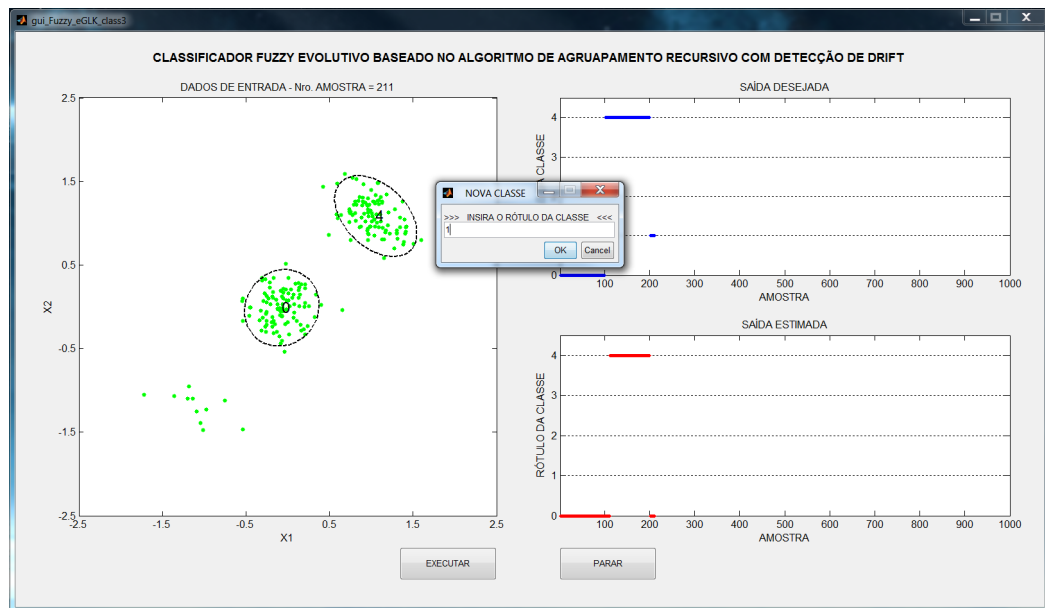


Figura 5.7: Detecção e criação da terceira regra com rótulo da classe fornecido pelo operador.

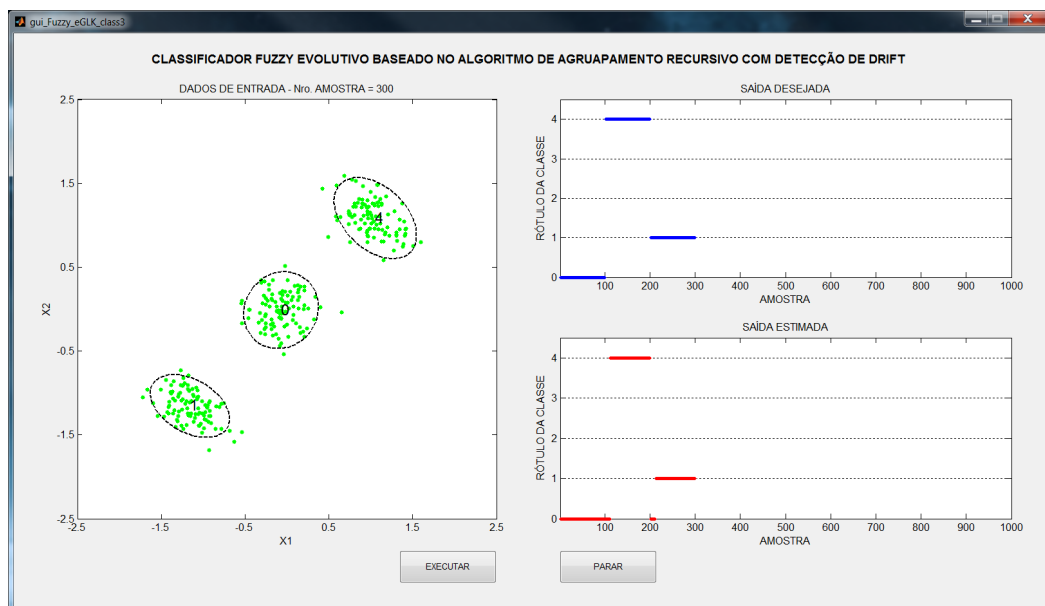


Figura 5.8: Atualização dos parâmetros da terceira regra.

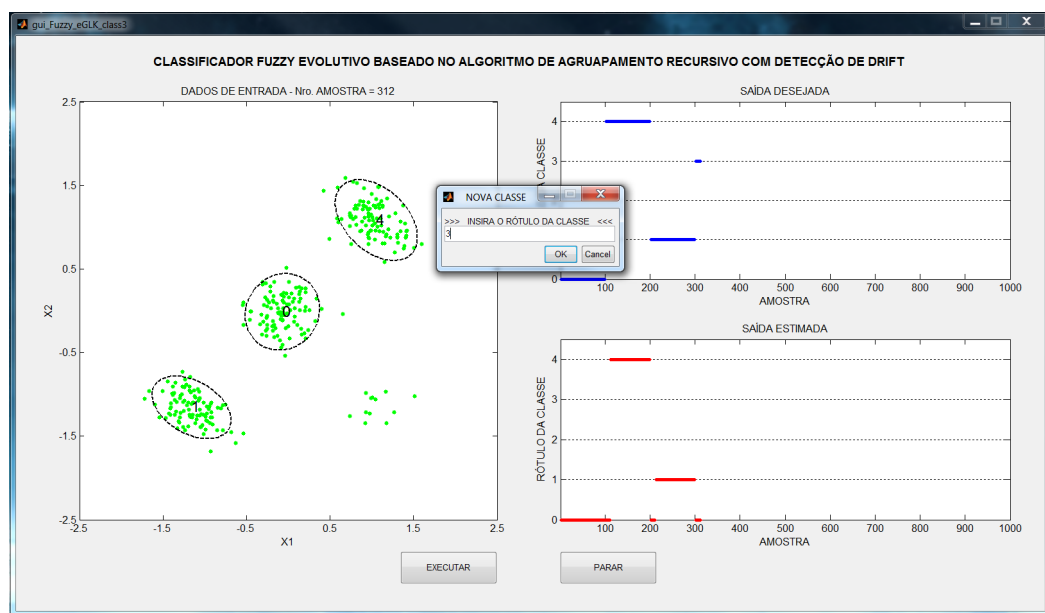


Figura 5.9: Detecção de nova classe e criação da quarta regra com rótulo da classe fornecido pelo operador.

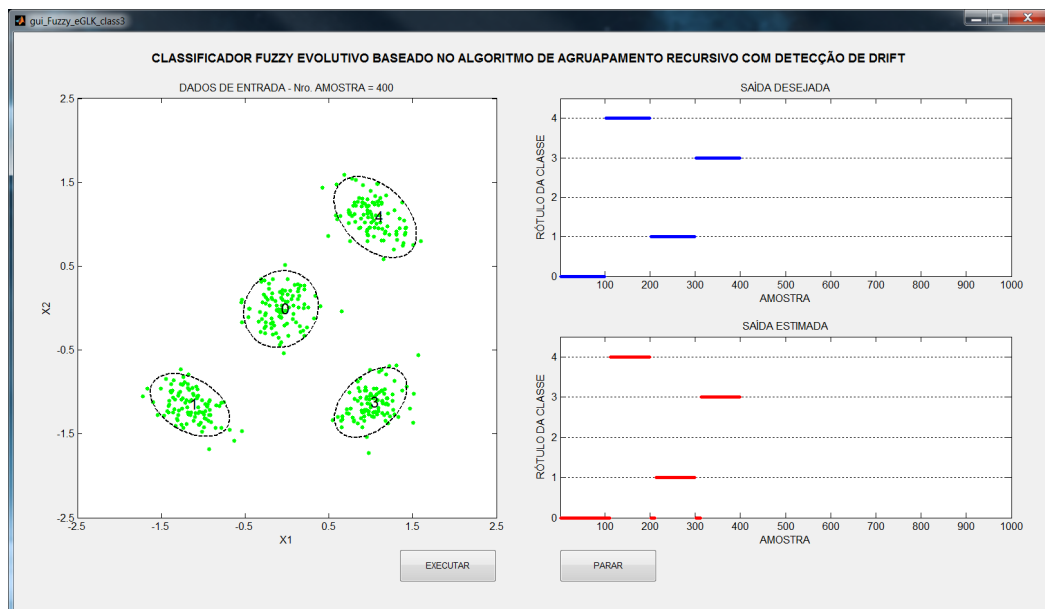


Figura 5.10: Atualização dos parâmetros da quarta regra.

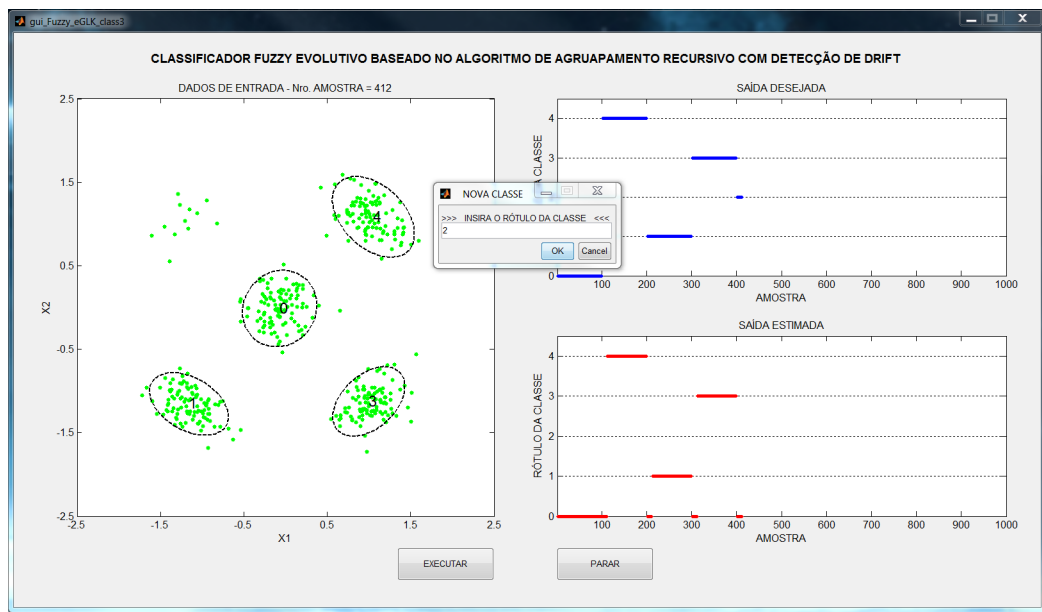


Figura 5.11: Detecção de nova classe e criação da quinta regra com rótulo da classe fornecido pelo operador.

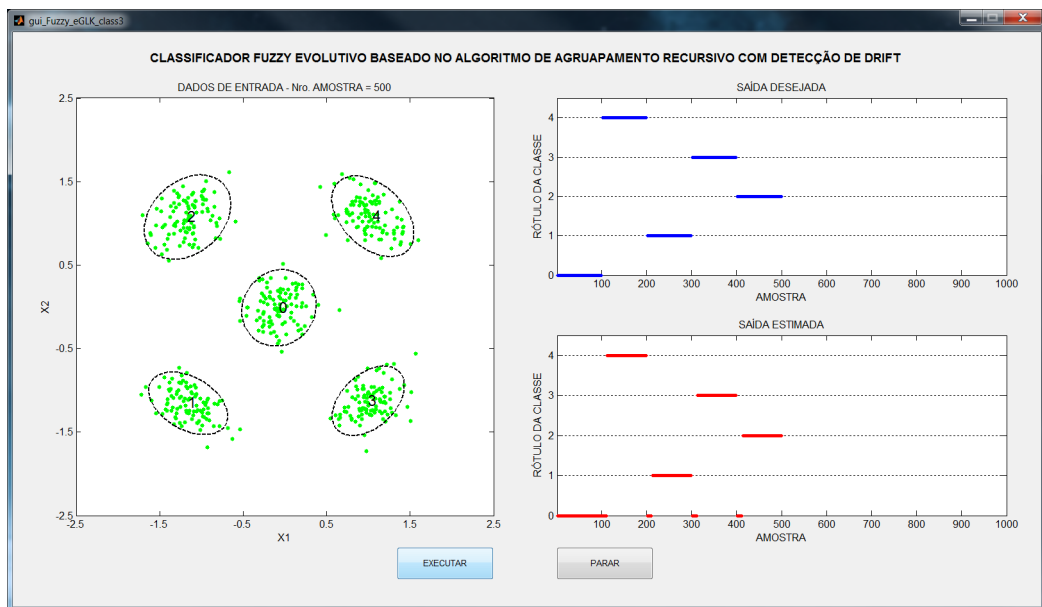


Figura 5.12: Atualização dos parâmetros da quinta regra.

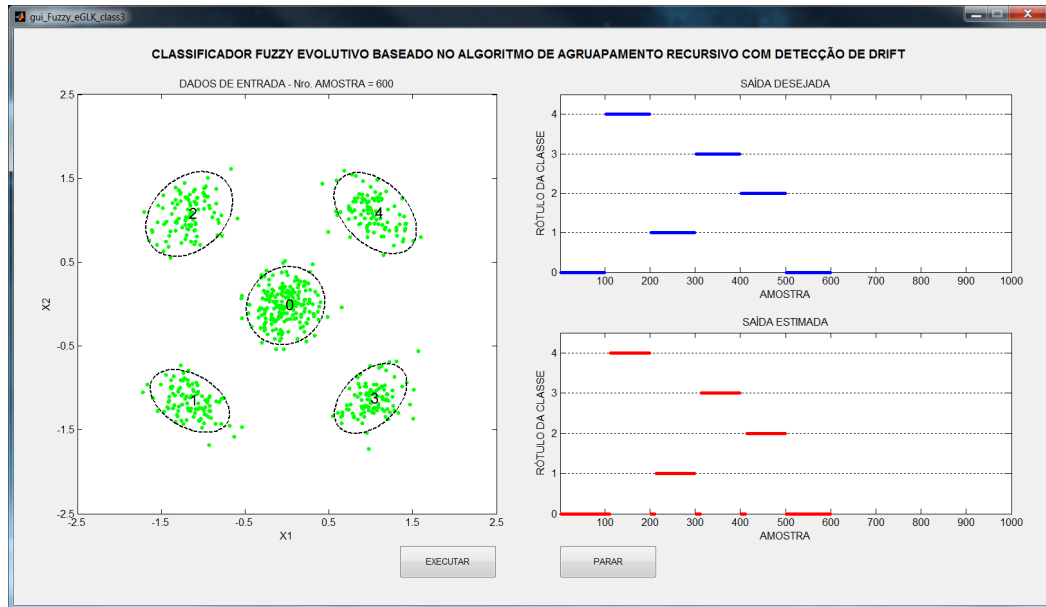


Figura 5.13: Amostras de dados da classe 0 classificadas automaticamente.

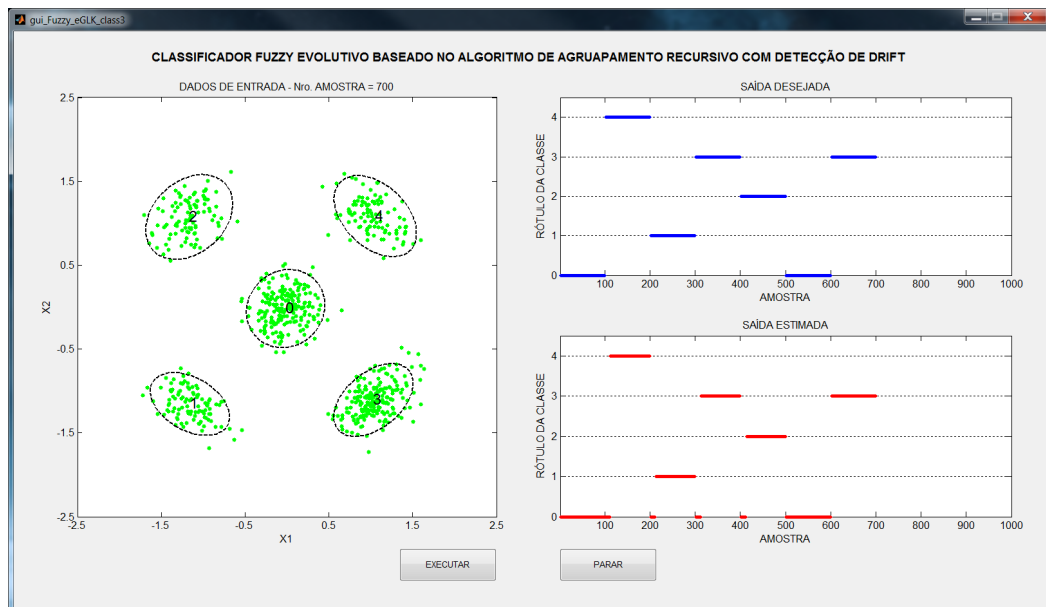


Figura 5.14: Amostras de dados da classe 3 classificadas automaticamente.

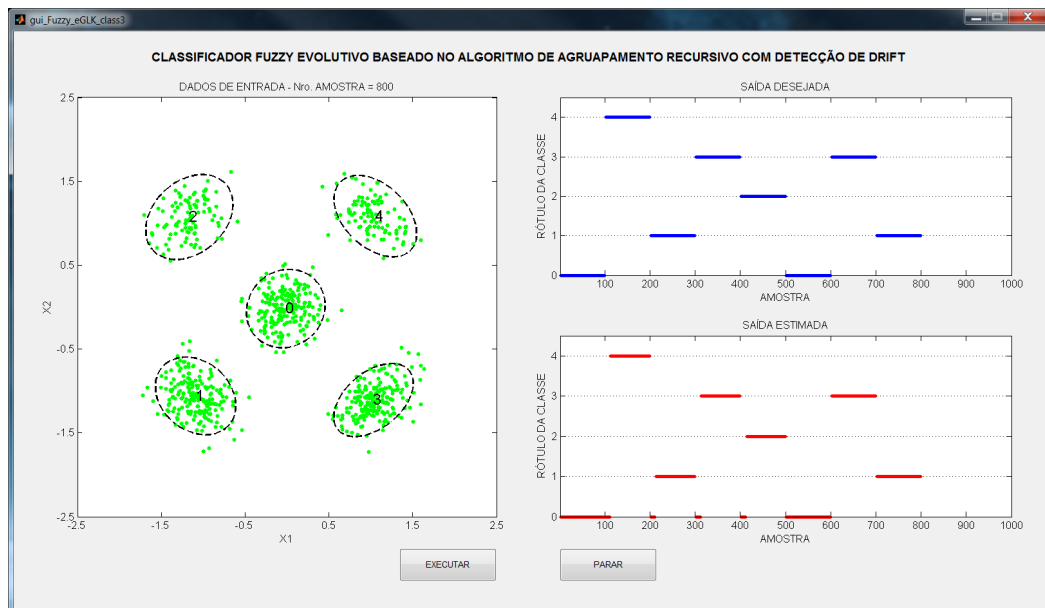


Figura 5.15: Amostras de dados da classe 1 classificadas automaticamente.

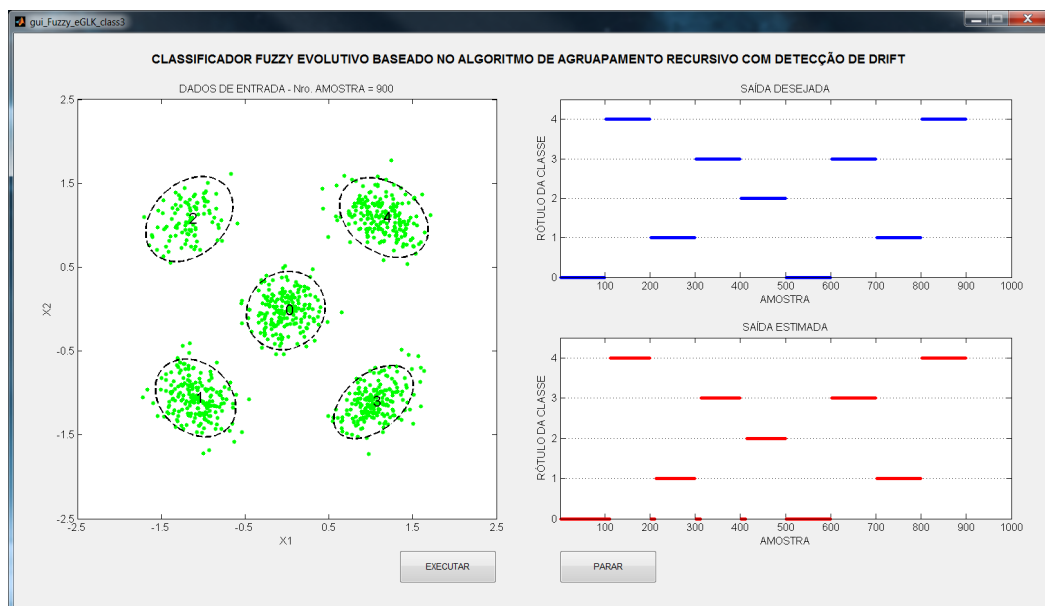


Figura 5.16: Amostras de dados da classe 4 classificadas automaticamente.

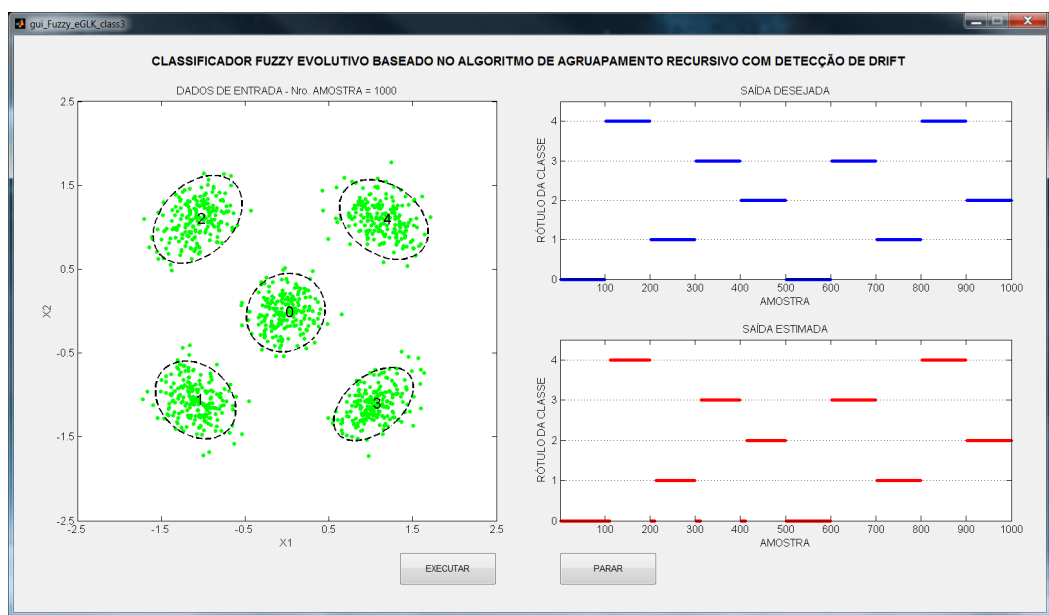


Figura 5.17: Amostras de dados da classe 2 classificadas automaticamente.

Capítulo 6

Resultados Numéricos

6.1 Introdução

Para permitir a avaliação do classificador *fuzzy* evolutivo proposto nesse trabalho são realizadas simulações em dois problemas distintos: diagnóstico de falhas em sistema de acionamento C.C. e diagnóstico de falhas em sistema de tanques interativos. Esses problemas foram escolhidos por representarem sistemas dinâmicos não-estacionários e por sua aplicabilidade no mundo real.

Em ambos os casos foram utilizados simuladores de falhas correspondentes a cada tipo de sistema, a partir dos quais foram gerados dados de operação normal e dados de falha, organizados em sequências aleatórias com operação normal e com falha. A saída do classificador é comparada com as sequências fornecidas, permitindo comprovar a eficiência do mesmo no diagnóstico de falhas. Para o sistema de tanques interativos também é realizado diagnóstico de falhas com dados experimentais obtidos na planta real, com o objetivo de validar o classificador.

A avaliação do desempenho do classificador no diagnóstico de falhas dos sistemas dinâmicos em questão é realizada em termos de detecção de falhas e de classificação de falhas, conforme sugerido em Vachtsevanos et al. (2006). Segundo os autores, a avaliação da detecção de falhas pode ser realizada a partir de uma matriz de decisão (Tabela 6.1). Essa matriz é baseada em uma metodologia de teste de hipóteses e representa todas as combinações possíveis de detecção de falhas que possam ocorrer. A partir da matriz de decisão são

calculadas três métricas:

- Probabilidade de detecção (*Probability of Detection* - POD): proporção de falhas detectadas em relação a todos os casos de falhas potenciais (sensibilidade).

$$POD = \frac{a}{a + c} \quad (6.1)$$

- Probabilidade de alarme falso (*Probability of False Alarm* - POFA): proporção de falhas detectadas em relação a todos os casos de operação normal.

$$POFA = \frac{b}{b + d} \quad (6.2)$$

- Acurácia (*Accuracy* - ACC): eficiência do algoritmo em distinguir corretamente a ocorrência de uma falha da condição de operação normal.

$$Acc = \frac{a + d}{a + b + c + d} \quad (6.3)$$

Tabela 6.1: Matriz de decisão para avaliação da detecção de falhas

Resultado	Falha	Operação Normal	Total
Positivo (detectado)	a Número de falhas detectadas	b Número de alarmes falsos	$a + b$ Total de casos com alarmes
Negativo (não detectado)	c Número de falhas não detectadas	d Número de rejeições corretas	$c + d$ Total de casos sem alarmes
	$a + c$ Número total de casos de falhas	$b + d$ Número total de casos de op. normal	$a + b + c + d$ Número total de casos

Na avaliação da classificação de falhas é calculada a métrica taxa de isolamento de falha (*Fault Isolation Rate* - FIR), que expressa o percentual de todas as falhas que o classificador é capaz de isolar inequivocadamente. Essa métrica é calculada como:

$$FIR = \frac{A}{A + C} \quad (6.4)$$

onde A representa o total de falhas detectadas e corretamente classificadas, e C representa o total de falhas detectadas, mas incorretamente classificadas.

Outras métricas que foram utilizadas na avaliação do desempenho do classificador são as seguintes:

- Tempo de detecção da falha (*Detection Delay Time* - DDT), que representa o intervalo de tempo entre o instante que a falha ocorre pela primeira vez e sua detecção pelo algoritmo;
- Tempo de isolamento da falha (*Isolation Delay Time* - IDT), que representa o intervalo de tempo entre o instante de tempo que a falha ocorre pela segunda vez e sua classificação pelo algoritmo;
- Percentual de falhas classificadas com a intervenção do operador (*Operator Intervention Rate* - OIR).

A Figura 6.1 ilustra os tempos de detecção e isolamento de falha em um cenário de 3 falhas.

Nas seções 6.2 e 6.3 deste capítulo são apresentadas as simulações, bem como os resultados obtidos na avaliação do classificador no diagnóstico de falhas em sistema de acionamento C.C. e no diagnóstico de falhas em sistema de tanques interativos, respectivamente.

6.2 Diagnóstico de falhas em sistema de acionamento C.C.

O sistema de acionamento C.C. empregado foi proposto por Caminhas and Takahashi (2001) e consiste em um *benchmark* para detecção e diagnóstico de falhas que descreve um sistema de acionamento de uma máquina de corrente contínua, do tipo utilizado em indústrias e que permitem alto desempenho no controle de velocidade (D'Angelo et al., 2010). Conforme ilustrado na Figura 6.2, o sistema é composto por duas fontes de alimentação, dois conversores estáticos controlados, uma máquina de corrente contínua e uma carga mecânica.

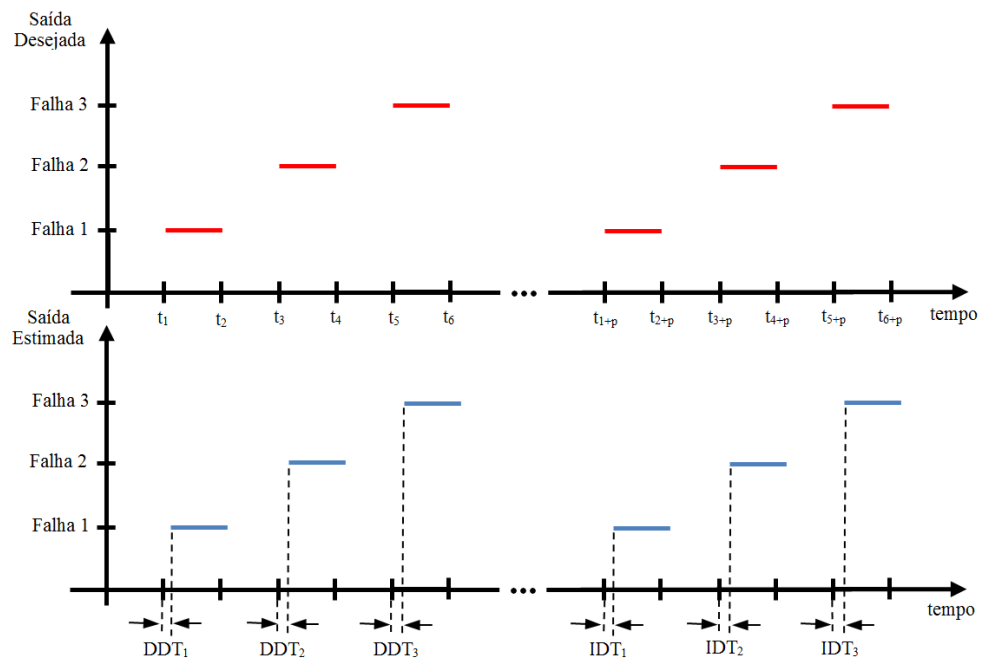


Figura 6.1: Ilustração dos tempos de detecção e isolamento da falha.

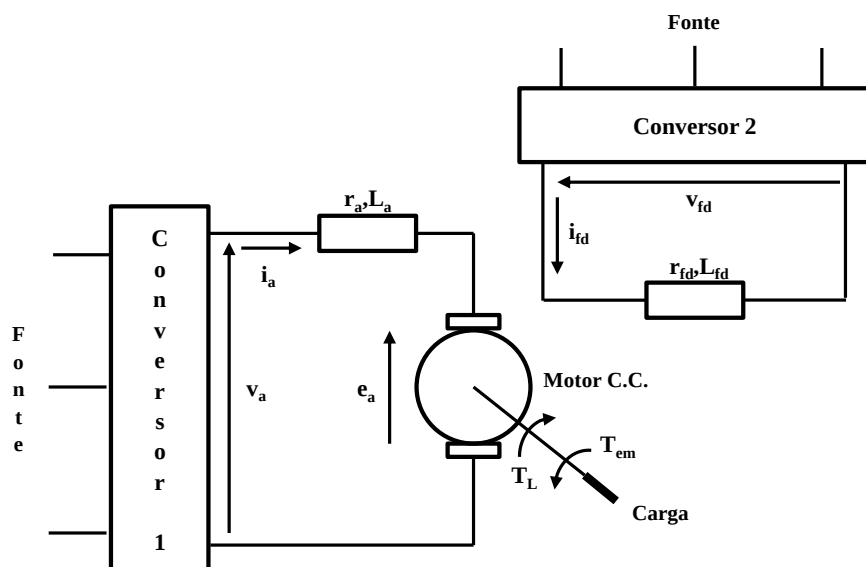


Figura 6.2: Representação do Sistema de Acionamento C.C.

As definições das variáveis mostradas na representação do sistema são as seguintes:

- v_a : tensão de alimentação do circuito da armadura;
- v_{fd} : tensão de alimentação do circuito de campo;
- i_a : corrente do circuito da armadura;
- i_{fd} : corrente do circuito de campo;
- r_a, L_a : resistência/indutância do circuito da armadura;
- r_{fd}, L_{fd} : resistência/indutância do circuito de campo;
- e_a : força contra-eletromotriz da armadura;
- T_{em} : conjugado eletromagnético;
- T_L : conjugado mecânico.

Utilizando a modelagem de falhas do sistema de acionamento C.C. proposta em Caminhas (1997) foi possível realizar a simulação de falhas nos atuadores (conversores), na planta (máquina e carga mecânica) e nos sensores (medidores de corrente e de velocidade), conforme detalhado na Tabela 6.2.

Para simulação das falhas foi empregada velocidade constante e sobrecarga variável, sendo o valor da sobrecarga gerado aleatoriamente entre 0 e 50% do valor nominal. No início de cada simulação o sistema está em operação normal (regime estacionário) e a falha é inserida na metade do intervalo de simulação. Na aquisição das amostras de dados foi empregado um período de amostragem (T_a) das variáveis monitoradas de 2ms. De forma a avaliar a robustez do classificador *fuzzy* evolutivo à presença de ruído nos dados, em cada variável monitorada foi inserido um ruído aleatório Gaussiano com média zero e desvio padrão igual a 2% do valor nominal da variável, considerando a operação normal do sistema.

As Figuras 6.3 - 6.7 mostram, como exemplo, as curvas da corrente de armadura (i_a), da corrente de campo (i_{fd}) e da velocidade da máquina (v) na simulação de cinco falhas diferentes. Nesse caso, foram simuladas as seguintes falhas: abertura da bobina de armadura, curto-circuito do conversor de campo, curto-circuito entre as espiras da bobina da armadura, falha de lubrificação dos

Tabela 6.2: Tipos de falhas no sistema de acionamento C.C.

Índice	Descrição
0	Operação normal
1	Abertura da bobina da armadura
2	Abertura da bobina de campo
3	Curto-circuito no conversor da armadura
4	Curto-circuito no conversor do campo
5	Curto-circuito entre espiras da bobina de armadura
6	Curto-circuito entre espiras da bobina de campo
7	Falha no sistema de ventilação
8	Falha de lubrificação dos mancais
9	Falha no sensor de corrente da armadura
10	Falha no sensor de corrente do campo
11	Falha no sensor de velocidade

mancais e falha no sensor de corrente de campo, respectivamente. No início de cada simulação, o sistema está em operação normal. Na Figura 6.8 são exibidas as mesmas falhas apresentadas nas figuras anteriores, mas vistas no espaço de três dimensões, onde é possível observar que no sistema de acionamento C.C. as falhas têm diferentes comportamentos.

As simulações de diagnóstico de falhas foram realizados considerando diferente cenários. Cada cenário consiste na simulação de uma sequência de falhas. A quantidade de falhas em cada sequência varia de três a onze tipos de falhas, sendo cada tipo de falha selecionada gerada aleatoriamente dentro do conjunto de tipos de falhas. Antes de cada falha o sistema o sistema está em operação normal. Foram simulados 50 cenários de três falhas, 50 cenários de cinco falhas, 50 cenários de sete falhas, 50 cenários de nove falhas e 50 cenários de onze falhas, totalizando 250 cenários distintos. As amostras de dados referentes às variáveis monitoradas no sistema de acionamento C.C., corrente de armadura (I_a), corrente de campo (I_{fd}) e velocidade (V), foram fornecidas à entrada do classificador em modo *on-line*, e a cada sequência, a saída do classificador foi comparada com a sequência fornecida para determinar o desempenho. Con-

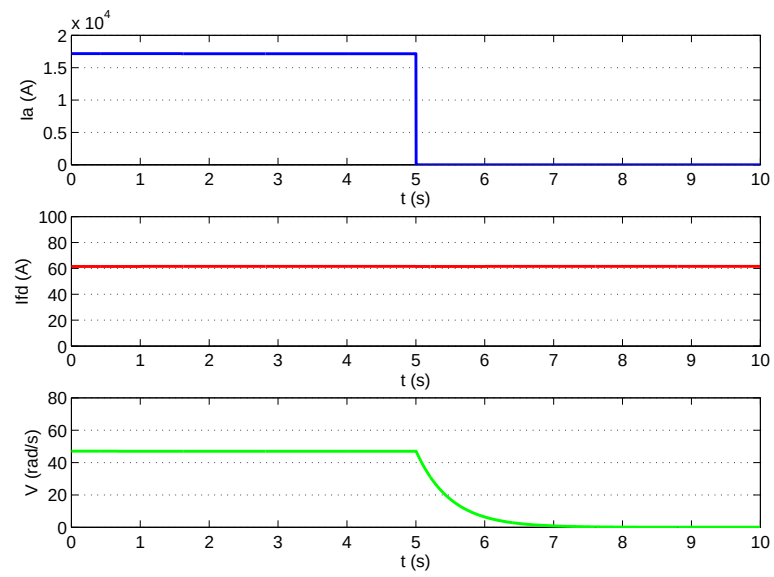


Figura 6.3: Simulação de falha: abertura da bobina de armadura.

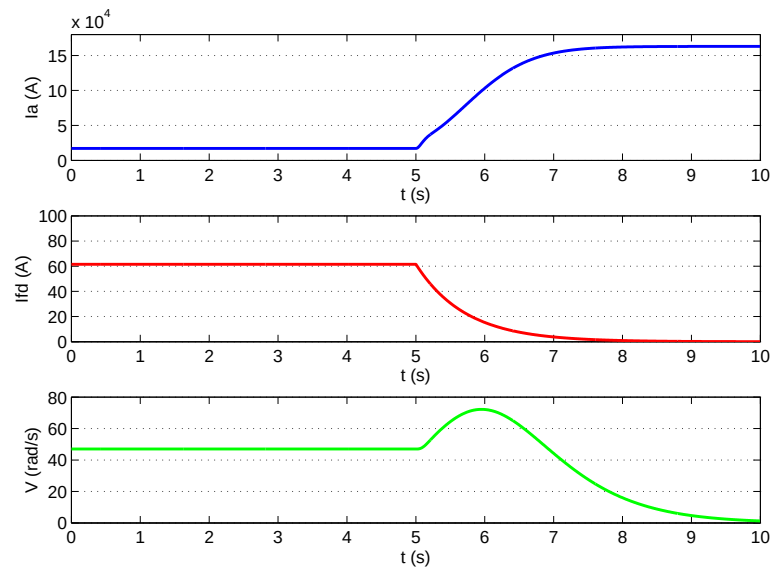


Figura 6.4: Simulação de falha: curto circuito no conversor de campo.

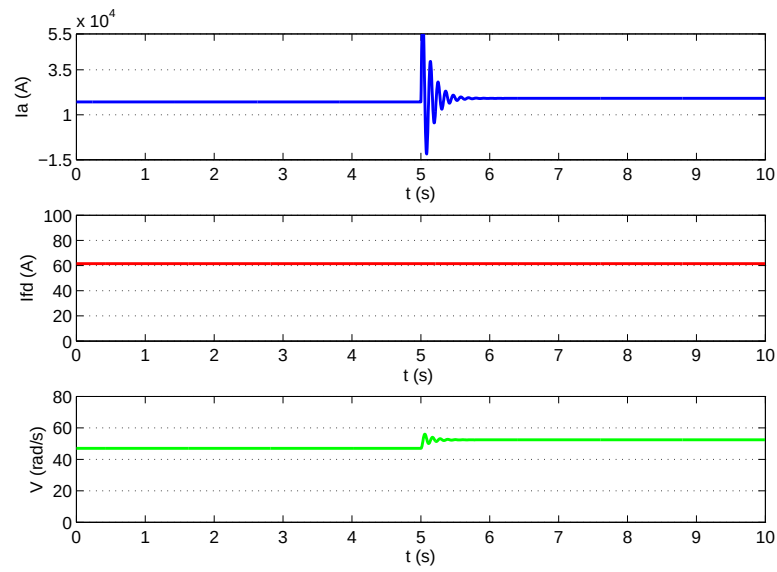


Figura 6.5: Simulação de falha: curto circuito entre as espiras da bobina da armadura.

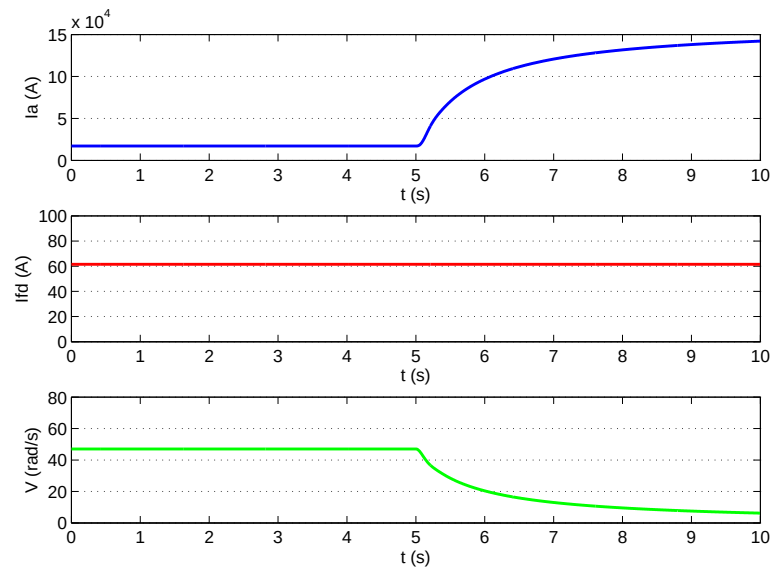


Figura 6.6: Simulação de falha: falha de lubrificação dos mancais.

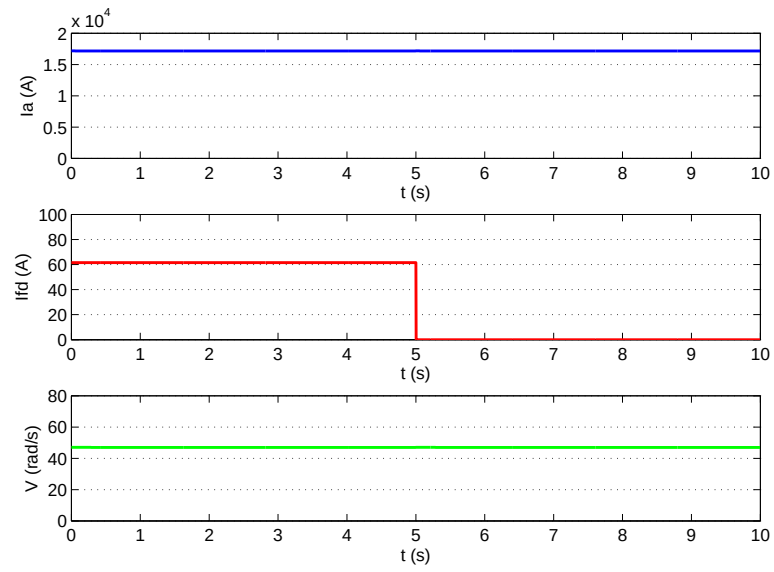


Figura 6.7: Simulação de falha: falha no sensor de corrente de campo.

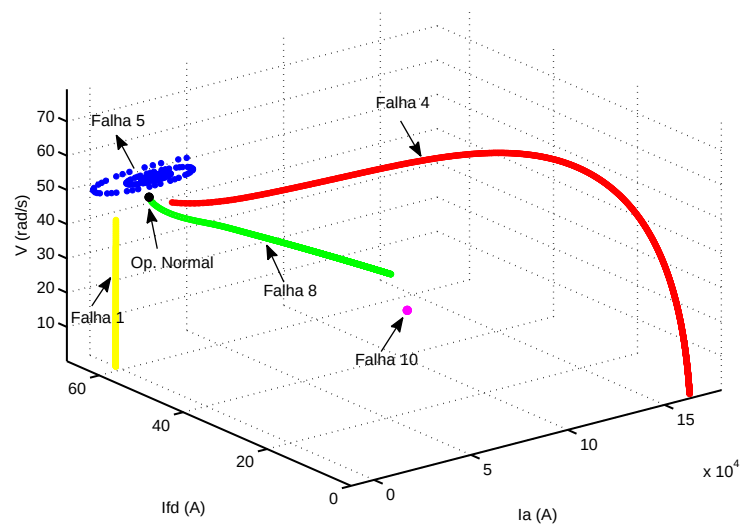


Figura 6.8: Simulação das falhas vistas em espaço 3D.

siderando que o classificador *fuzzy* evolutivo inicia sem nenhuma regra *fuzzy* definida, as primeiras amostras de dados devem corresponder à operação normal do sistema, ou seja, a primeira regra criada deve descrever a operação normal. Na sequência, durante a realização do diagnóstico, as falhas são detectadas e são criadas novas regras que descrevem cada tipo de falha. O rótulo de cada classe, que a define como operação normal ou como falha são fornecidas pelo operador do sistema no instante que cada nova regra é criada. Nos instantes seguintes à primeira ocorrência de uma falha, a intervenção do operador não se faz mais necessária. Para realização do diagnóstico de falhas, os parâmetros do algoritmo de agrupamento recursivo foram definidos como: $\chi^2_{3,0,95} = 7,8147$; $F_{init} = 10^{-2}I$; $\alpha_{init} = 0,5$; $z_1 = 2$; $z_2 = 3$.

As Figuras 6.9 - 6.13 mostram, como exemplo, o resultado do diagnóstico de falhas em cada um dos tipos de cenários simulados, onde pode-se comparar a saída estimada (sequência de falhas classificadas) pelo classificador *fuzzy* evolutivo com a saída desejada (sequência de falhas selecionadas), a partir das amostras de dados de entrada. Os resultados obtidos demonstram que o classificador foi capaz de diagnosticar corretamente todas as falhas do sistema de acionamento C.C. em todos cenários simulados. Considerando a presença de ruído nas amostras de dados, a ocorrência de alarmes falsos ou de erros de classificação (representados por pontos isolados nos gráficos) é significativamente baixa, mesmo no cenário com o maior número de falhas possível.

Todos os resultados das simulações de diagnóstico de falhas com o sistema de acionamento C.C. obtidos com o classificador *fuzzy* evolutivo proposto nesse trabalho foram comparados com os resultados obtidos utilizando o classificador proposto por Lemos et al. (2013). Para realização das simulações, foram adotados os seguintes valores para os parâmetros desse classificador: $w = 100$, $\lambda = 0,001$, $\alpha = 0,01$, $T_{\mu_y} = 0,01$. De acordo com os autores a combinação de parâmetros foi obtida experimentalmente visando manter um bom equilíbrio entre a taxa de alarme falso e sensibilidade na abordagem de detecção e diagnóstico de falhas empregada.

A Tabela 6.3 resume os resultados das simulações obtidos com ambos os classificadores em relação às métricas de detecção de falhas. Os resultados

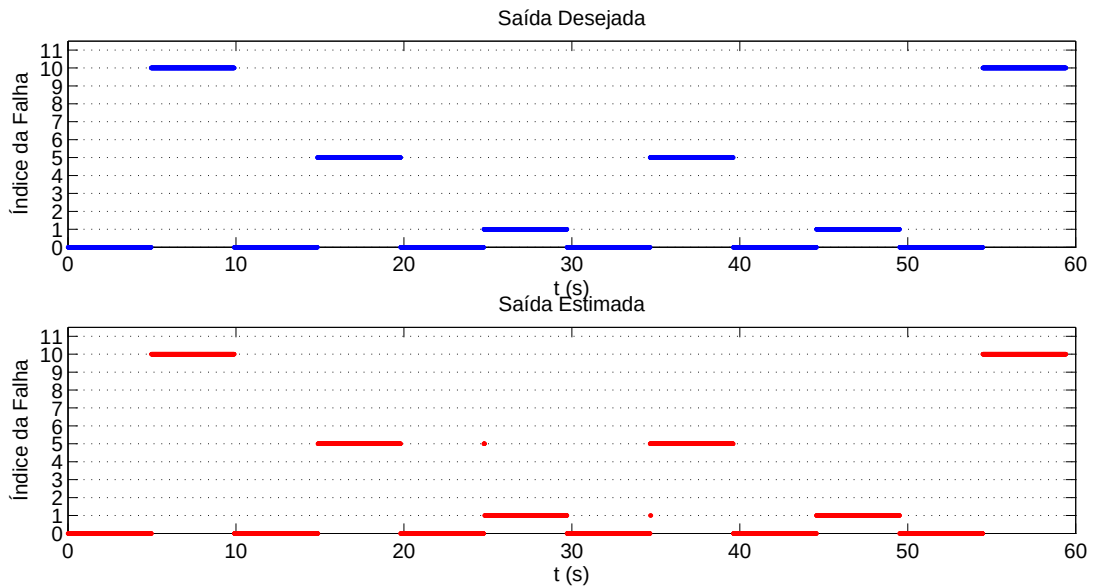


Figura 6.9: Saída desejada e saída estimada pelo classificador em cenário de 3 falhas simuladas no sistema de acionamento C.C.

apresentados demonstram que o classificador *fuzzy* evolutivo obteve altos índices de detecção de falha e acurácia em todos os cenários, com valores acima de 99% e baixo índice de alarme falso, com valores abaixo de 0,09%, comprovando a eficiência do classificador na detecção das falhas simuladas no sistema de acionamento C.C. O classificador proposto por Lemos et al. (2013), apesar de apresentar índices de detecção de falhas e acurácia menores, apresentou índices de alarme falso menores.

A Tabela 6.4 resume os resultados das simulações obtidos com ambos os classificadores em relação às métricas de classificação de falhas. Os resultados apresentados demonstram que o classificador *fuzzy* evolutivo obteve alto índice de isolamento de falha em todos os cenários, com valor médio de aproximadamente 97%, e em todos os cenários a intervenção do operador na classificação das falhas foi menor ou igual a 0,23%. Esses resultados comprovam a capacidade do classificador proposto nesse trabalho de diagnosticar automaticamente a quase totalidade das falhas após a primeira ocorrência dessas e evidencia a sua capacidade de aprendizado. Nota-se que o classificador proposto por Lemos et al. (2013) obteve, em geral, menor desempenho na classificação das falhas que o classificador proposto nesse trabalho e necessitou de maior intervenção

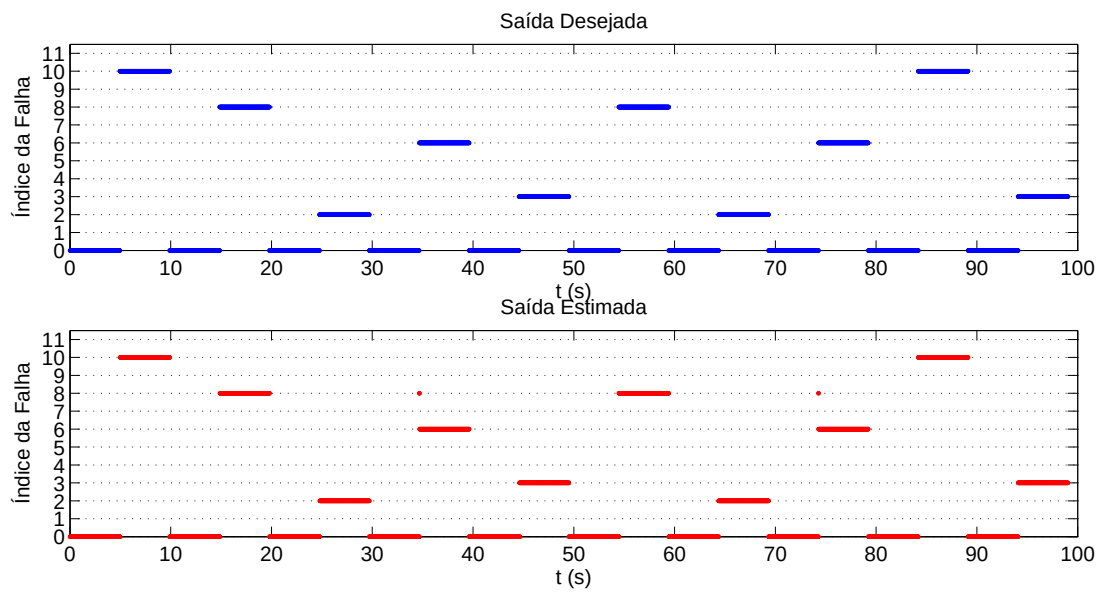


Figura 6.10: Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de acionamento C.C.

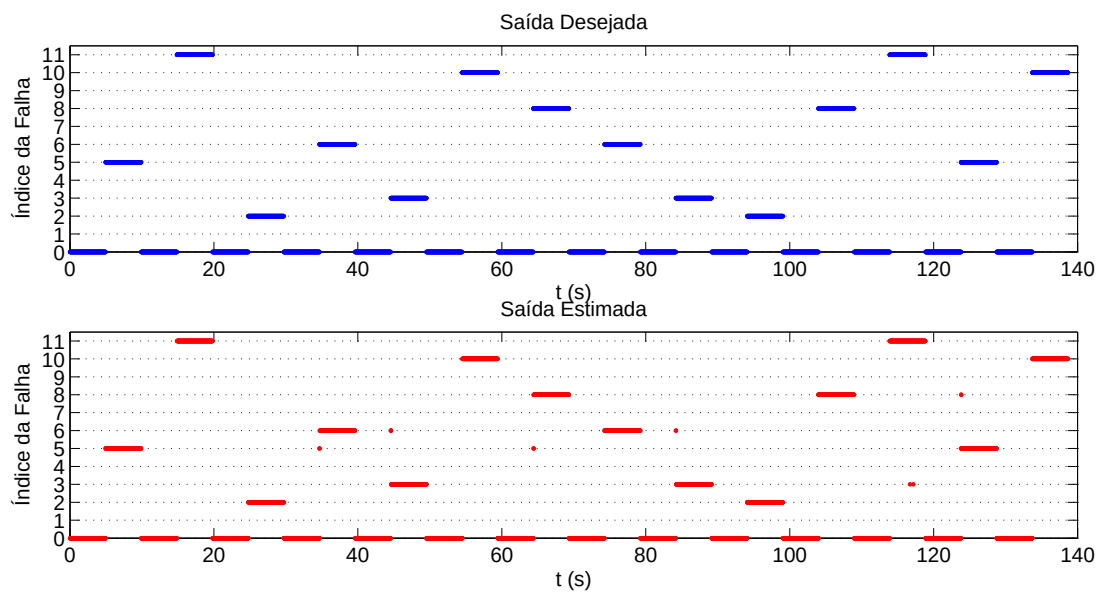


Figura 6.11: Saída desejada e saída estimada pelo classificador em cenário de 7 falhas simuladas no sistema de acionamento C.C.

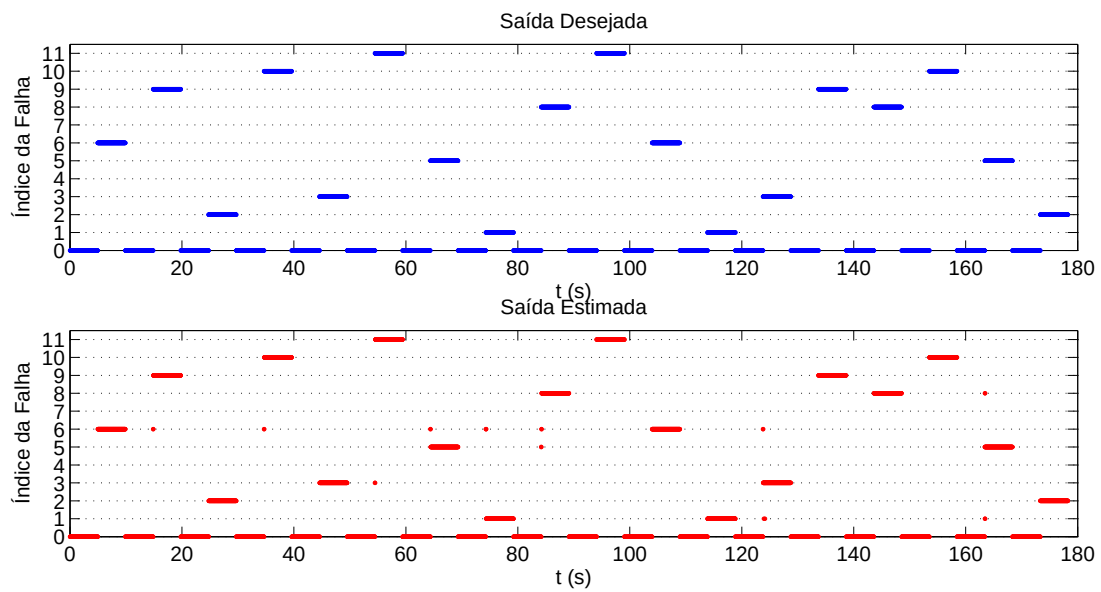


Figura 6.12: Saída desejada e saída estimada pelo classificador em cenário de 9 falhas simuladas no sistema de acionamento C.C.

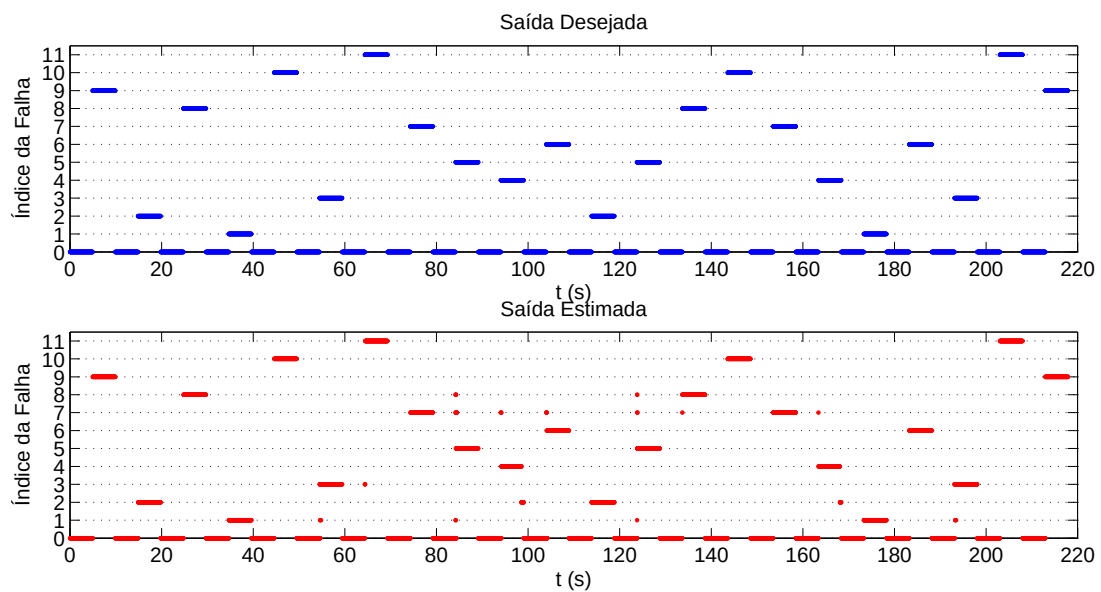


Figura 6.13: Saída desejada e saída estimada pelo classificador em cenário de 11 falhas simuladas no sistema de acionamento C.C.

do operador.

A Tabela 6.5 resume os resultados das simulações obtidos com ambos os classificadores em relação às métricas de tempo na detecção e na classificação das falhas. Os valores são apresentados em quantidade de períodos de amostragem das variáveis monitoradas, nesse caso, $Ta = 2ms$. O tempo médio de detecção da falha obtido nas simulações com o classificador *fuzzy* evolutivo foi de aproximadamente 21 períodos, que é determinado principalmente pela quantidade de amostras de dados necessárias para o algoritmo de agrupamento recursivo detectar uma mudança de contexto. O tempo médio de isolamento da falha obtido nas simulações foi de aproximadamente 6 períodos. Uma comparação entre os valores do tempo de detecção da falha e do tempo de isolamento da falha demonstra que a classificação das falhas é mais rápida após a primeira ocorrência de cada tipo de falha, uma vez que a base de dados do classificador já possui as regras *fuzzy* e rótulos para todos os tipos de falhas detectadas, não requerendo a intervenção do operador. Os resultados das simulações com classificador proposto por Lemos et al. (2013) mostram tempos médios de detecção e isolamento da falha de aproximadamente 12 períodos e 4 períodos, respectivamente, demonstrando que este classificador tem resposta mais rápida que o classificador proposto nesse trabalho, em função dos diferentes algoritmos de agrupamento empregados nos mesmos.

Tabela 6.3: Desempenho na detecção de falhas simuladas no sistema de acionamento C.C.

Cenário	Proposto			Lemos et al. (2013)		
	POD(%)	POFA(%)	ACC(%)	POD(%)	POFA(%)	ACC(%)
3 falhas	99,66	0,00	99,74	99,66	0,00	98,99
5 falhas	99,73	0,01	99,77	98,25	0,00	98,54
7 falhas	99,79	0,01	99,82	98,53	0,00	98,72
9 falhas	99,68	0,06	99,71	98,39	0,01	98,66
11 falhas	99,73	0,08	99,75	98,65	0,04	98,76

Outra simulação foi realizada para avaliar a robustez do classificador *fuzzy* evolutivo à presença de *outliers* nos dados. Nesse caso foi simulado um cenário de 5 falhas e, em seguida, foram acrescentadas às amostras de dados *outliers*, isto é, amostras de dados artificiais distribuídas aleatoriamente no espaço das

Tabela 6.4: Desempenho na classificação de falhas simuladas no sistema de acionamento C.C.

Cenário	Proposto		Lemos et al. (2013)	
	FIR(%)	OIR(%)	FIR(%)	OIR(%)
3 falhas	99,69	0,12	98,94	0,21
5 falhas	99,37	0,12	98,35	0,29
7 falhas	98,13	0,15	97,13	0,25
9 falhas	96,04	0,23	93,40	0,27
11 falhas	94,62	0,21	91,39	0,30

Tabela 6.5: Tempos de detecção e classificação por tipo de falha simulada no sistema de acionamento C.C.

Índice	Proposto		Lemos et al. (2013)	
	DDT(T_a)	IDT(T_a)	DDT(T_a)	IDT(T_a)
1	14	2	12	2
2	12	2	6	2
3	14	3	8	3
4	20	10	18	4
5	21	3	5	2
6	17	8	4	3
7	49	18	30	12
8	36	17	33	9
9	14	1	5	1
10	14	1	5	1
11	17	1	5	1

variáveis de entrada. A Fig. 6.14 apresenta o resultado da simulação das falhas com a presença de *outliers* no espaço de três dimensões. A Fig. 6.15 mostra o resultado do diagnóstico de falhas no cenário em questão, onde pode-se comparar a saída estimada pelo classificador *fuzzy* evolutivo (sequência de falhas classificadas) com a saída desejada (sequência de falhas selecionadas), a

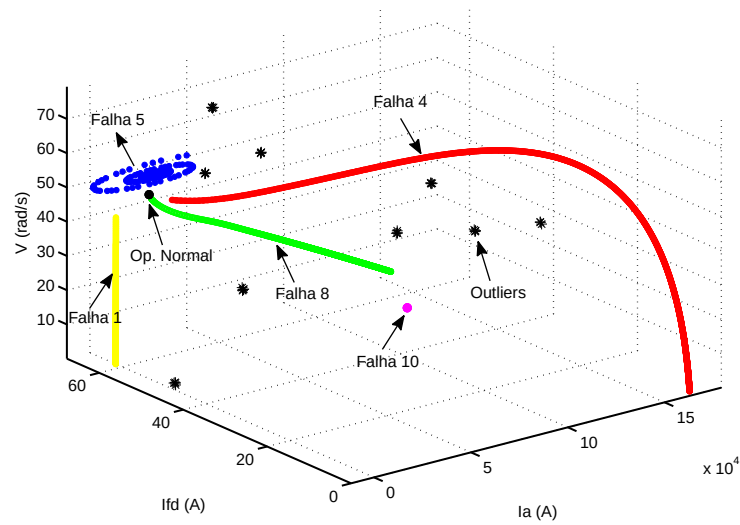


Figura 6.14: Simulação das falhas no sistema de acionamento C.C. com presença de *outliers*.

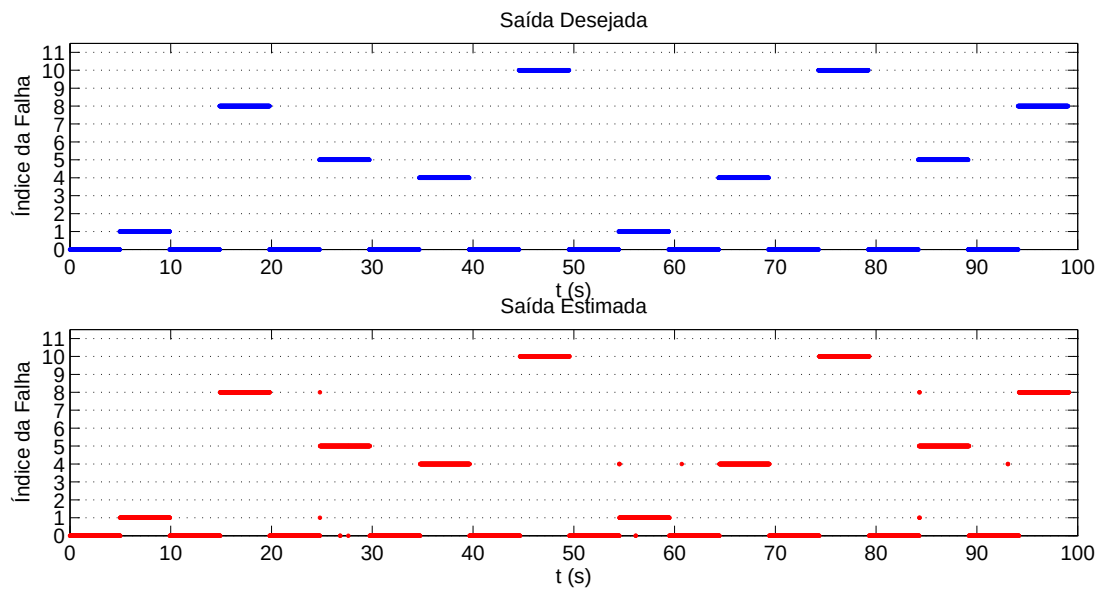


Figura 6.15: Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de acionamento C.C. com presença de *outliers*.

partir das amostras de dados de entrada com a presença de *outliers*. O resultado do diagnóstico de falhas nessa simulação demonstra que, mesmo com a presença de *outliers*, o classificador *fuzzy* evolutivo foi capaz de detectar e diagnosticar

corretamente todas as falhas do sistema de acionamento C.C., significando que o classificador foi capaz de distinguir de forma correta os dados válidos dos *outliers*.

Os resultados dessa simulação são apresentados na Tabela 6.6 e na Tabela 6.7, comprovando que o classificador obteve praticamente o mesmo desempenho no diagnóstico de falhas com e sem a presença de *outliers* nos dados de entrada, embora nota-se um aumento do índice de alarme falso. Essa simulação evidenciou a maior robustez do classificador proposto nesse trabalho em relação ao classificador proposto por Lemos et al. (2013), uma vez que este apresentou maiores diferenças no índice probabilidade de alarme falso e no índice taxa de isolamento de falha obtidos nos cenários com e sem a presença de *outliers*.

Tabela 6.6: Desempenho na detecção de falhas simuladas no sistema de acionamento C.C. com presença de *outliers*.

Cenário	Proposto			Lemos et al. (2013)		
	POD(%)	POFA(%)	ACC(%)	POD(%)	POFA(%)	ACC(%)
sem outliers	99,73	0,01	99,77	98,25	0,00	98,54
com outliers	99,35	0,18	99,40	97,09	0,27	97,21

Tabela 6.7: Desempenho na classificação de falhas simuladas no sistema de acionamento C.C. com presença de *outliers*.

Cenário	Proposto		Lemos et al. (2013)	
	FIR(%)	OIR(%)	FIR(%)	OIR(%)
sem outliers	99,37	0,12	98,35	0,28
com outliers	99,12	0,21	97,80	0,31

6.3 Diagnóstico de falhas em sistema de tanques interativos

O sistema de tanques interativos empregado nesse trabalho consiste de uma planta construída com atuadores e sensores similares aos utilizados em plantas industriais. A resposta da planta assemelha-se muito ao comportamento dinâmico de sistemas industriais reais (Jota et al., 1995). A descrição detalhada do sistema de tanques interativos é apresentada em Braga et al. (1995). Conforme ilustra a Figura 6.16, o sistema é composto por um computador do tipo PC (*Personal Computer*), um controlador do tipo PLC (*Programmable Logic Controller*), um painel de operação e o conjunto de tanques interativos.

O computador executa o *software* de supervisão e controle, e o mesmo está conectado ao PLC através de uma porta serial padrão RS-232C. O PLC é responsável pela aquisição dos dados relativos às variáveis do processo, permitindo a atuação na planta através dos dados enviados pelo computador, e executa o controle lógico do intertravamento necessário para o funcionamento correto do sistema. O painel de operação permite o controle lógico e a operação manual do processo, bem como a visualização das variáveis.

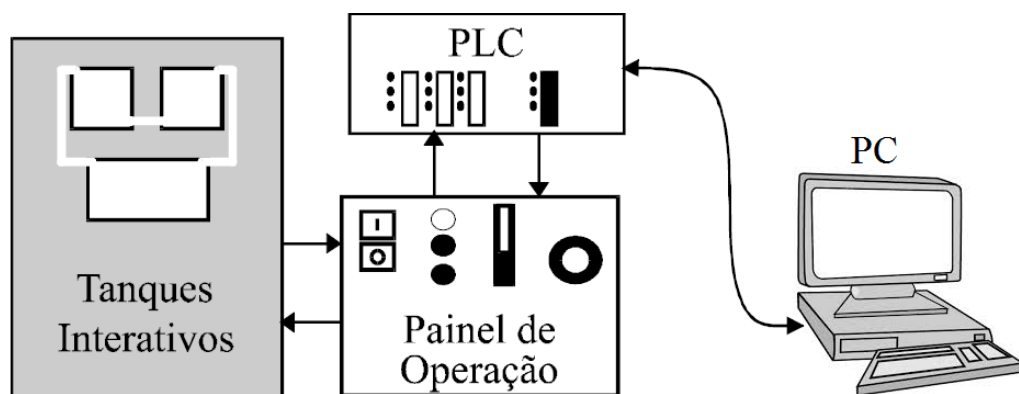


Figura 6.16: Diagrama em blocos do sistema de tanques interativos.

Na Figura 6.17 é mostrado o diagrama do fluxo de massa no sistema de tanques interativos. Nesse sistema o objetivo é controlar o nível e a vazão de saída do tanque TQ-3. O ciclo de operação do sistema consiste em bombear o

líquido do reservatório TQ-1 diretamente para o tanque TQ-2 através da bomba BA-1. Do tanque TQ-2 o líquido flui naturalmente para o tanque TQ-3. A intensidade desse acoplamento é controlada pelas válvulas manuais FV-3 e FV-4 (a válvula FV-5 permanece fechada durante a operação normal do sistema). O líquido é bombeado de volta para o reservatório através da bomba BA-2. Dessa forma, o controle do nível no tanque TQ-3 é realizado a partir da atuação na válvula pneumática FCV-1, que controla a vazão de entrada do tanque TQ-2 e, conseqüentemente, controla a vazão de entrada de TQ-3. O controle da vazão de saída de TQ-3 é realizado a partir da atuação na válvula pneumática FCV-2.

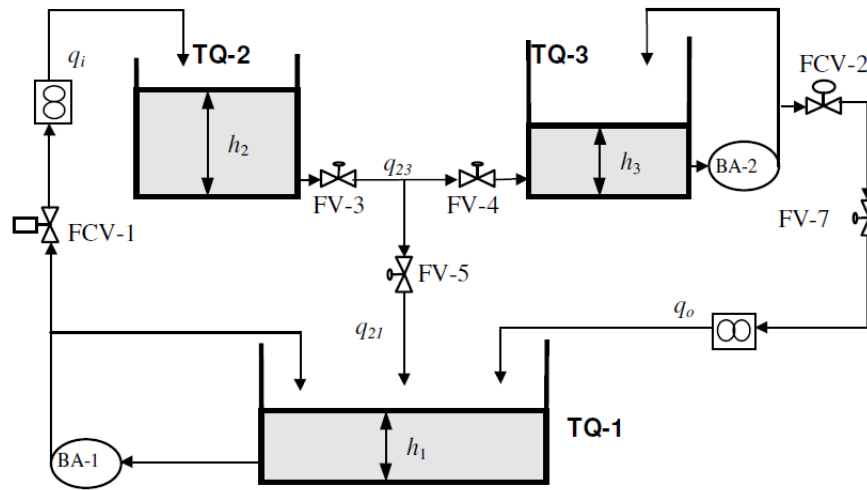


Figura 6.17: Fluxo de massa do sistema de tanques interativos.

As definições das variáveis mostradas no diagrama são as seguintes:

- h_1 : nível do líquido no reservatório TQ-1;
- h_2 : nível do líquido no tanque TQ-2;
- h_3 : nível do líquido no tanque TQ-3;
- q_i : vazão de entrada do tanque TQ-2;
- q_o : vazão de saída do tanque TQ-3;

- q_{21} : vazão entre TQ-2 e TQ-1;
- q_{23} : vazão entre TQ-2 e TQ-3;

6.3.1 Resultados de simulações

Utilizando a modelagem de falhas do sistema de tanques interativos proposta em Caminhas (1997) foi possível realizar a simulação de falhas nos atuadores (válvulas pneumática FCV-1 e FCV-2, bombas BA-1 e BA-2), nos componentes do sistema (dutos e conexões entre os tanques) e nos sensores (sensor de vazão de entrada de TQ-2, sensores de nível e vazão de saída de TQ-3). Os tipos de falhas simuladas são detalhadas na Tabela 6.8.

As simulações incluem variações de *setpoints* (nível e vazão de saída de TQ-3), cujas variáveis foram alteradas de forma aleatória em 10% dos valores nominais. Nas simulações de falhas, o sistema inicia em operação normal e a falha é inserida no instante $t = 500s$. Na aquisição das amostras de dados foi empregado um período de amostragem (T_a) das variáveis monitoradas de 2s. Para avaliar a robustez do classificador *fuzzy* evolutivo à presença de ruído nos dados, em cada variável monitorada foi inserido um ruído aleatório Gaussiano com média zero e desvio padrão igual a 0,075%, 0,25% e 0,5% dos valores máximos das variáveis nível de TQ-3, vazão de saída de TQ-3 e vazão de entrada de TQ-2, respectivamente.

As Figuras 6.18 - 6.22 mostram, como exemplo, as curvas da vazão de entrada no tanque TQ-2 (q_i), nível do tanque TQ-3 (h_3), vazão de saída do tanque TQ-3 (q_o) e os sinais de controle fracionário das válvulas FCV-1 e FCV-2 (u_1 e u_2 , respectivamente) na simulação de cinco falhas diferentes. As seguintes falhas foram simuladas nesse caso: agarramento da válvula FCV-1, desligamento da bomba BA-2, entupimento no duto entre TQ-2 e TQ-3, vazamento no duto entre TQ-2 e TQ-3 e falha no sensor de nível de TQ-3, respectivamente. Em cada simulação, o sistema inicia em operação normal.

Diferentes cenários foram considerados para realização das simulações de diagnóstico de falhas. Um cenário consiste na simulação de uma sequência de falhas. Cada falha da sequência é selecionada aleatoriamente dentro do conjunto de tipos de falhas, sendo que a quantidade de falhas varia de três a onze tipos. Antes da ocorrência de cada falha o sistema está em operação normal. Um total

Tabela 6.8: Tipos de falhas no sistema de tanques interativos

Índice	Descrição
0	Operação normal
1	Agarramento da válvula FCV-1
2	Agarramento da válvula FCV-2
3	Desligamento da bomba BA-01
4	Desligamento da bomba BA-02
5	Entupimento no duto entre TQ-1 e TQ-2
6	Entupimento no duto entre TQ-3 e TQ-1
7	Entupimento no duto entre TQ-2 e TQ-3
8	Vazamento no duto entre TQ-2 e TQ-3
9	Falha no sensor de nível de TQ-3
10	Falha no sensor de vazão de saída de TQ-3
11	Falha no sensor de vazão de entrada de TQ-2

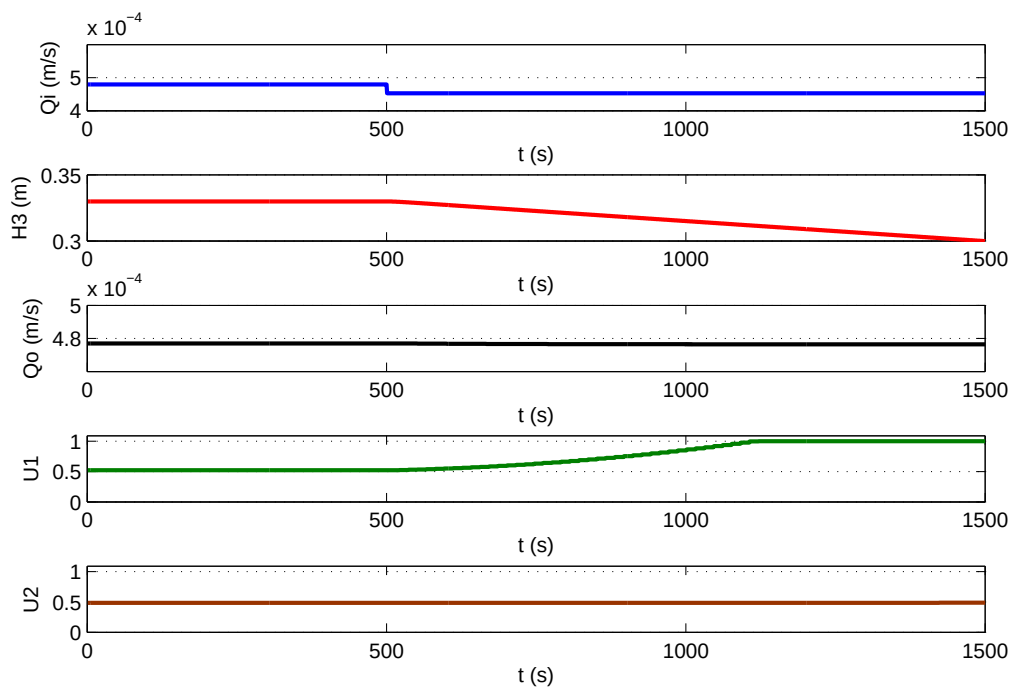


Figura 6.18: Simulação de falha: agarramento da válvula FCV-1.

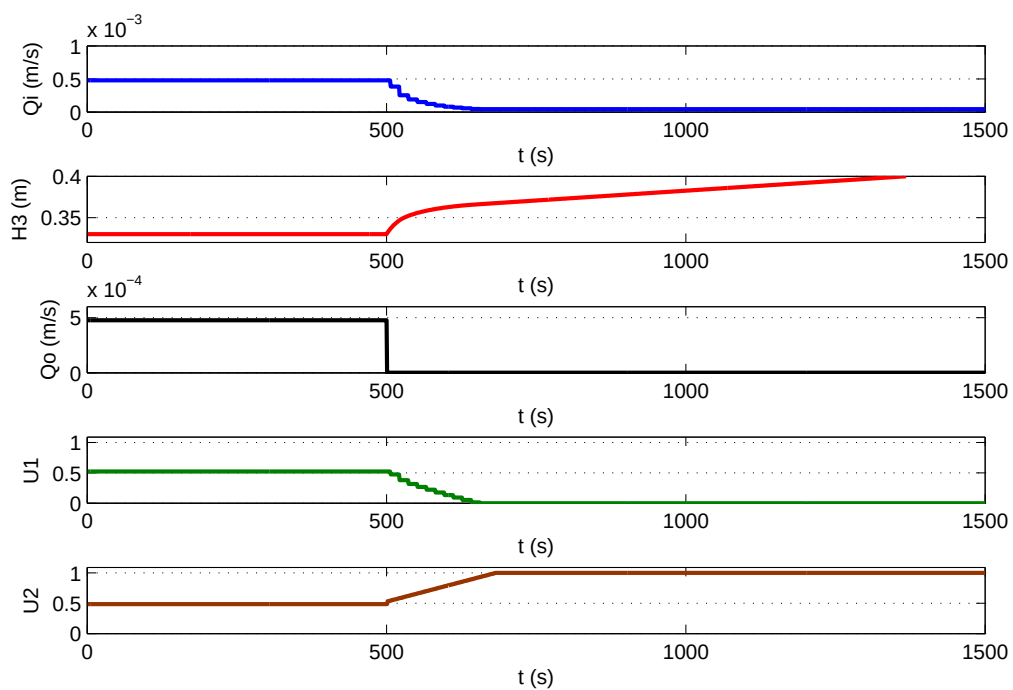


Figura 6.19: Simulação de falha: desligamento da bomba BA-2.

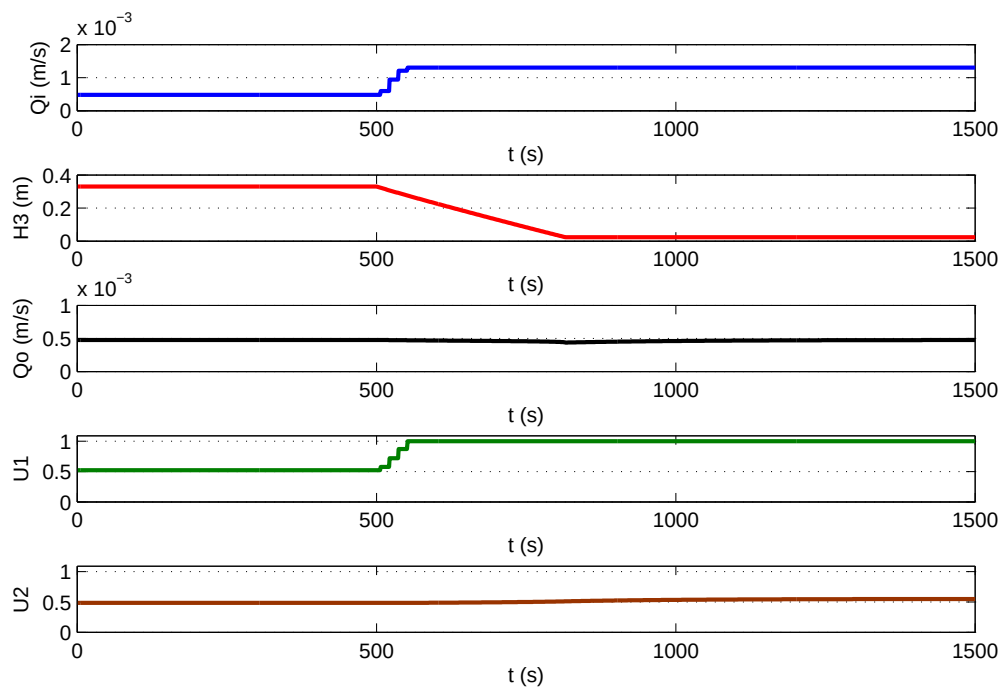


Figura 6.20: Simulação de falha: entupimento no duto entre TQ-2 e TQ-3.

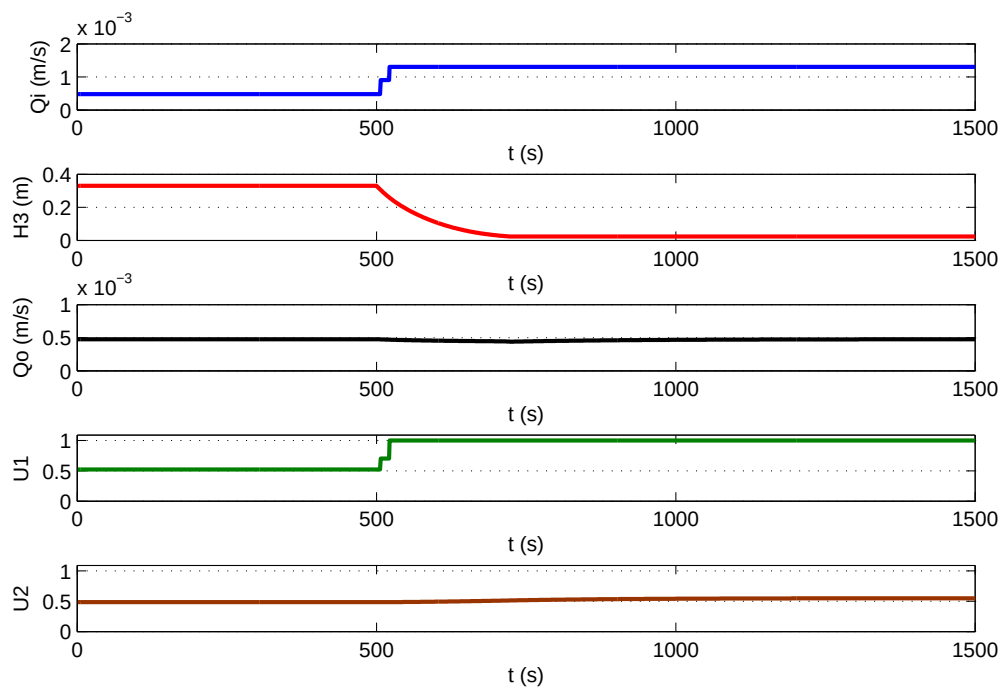


Figura 6.21: Simulação de falha: vazamento no duto entre TQ-2 e TQ-3.

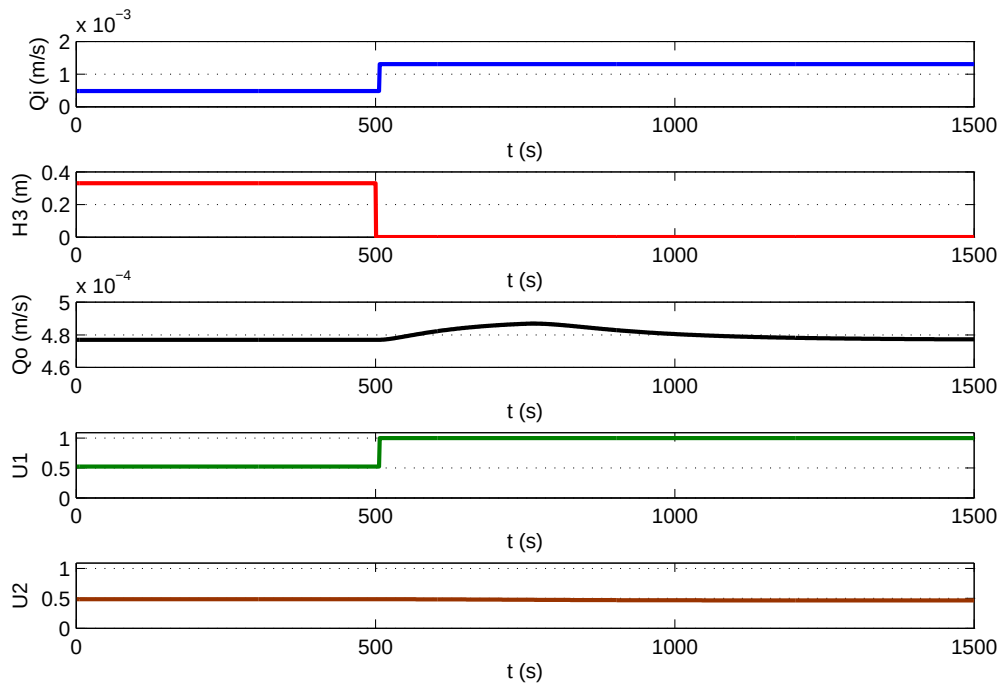


Figura 6.22: Simulação de falha: falha no sensor de nível de TQ-3.

de 250 cenários foram simulados, sendo 50 cenários de três falhas, 50 cenários de 5 falhas, 50 cenários de sete falhas, 50 cenários de nove falhas e 50 cenários de onze falhas. As amostras de dados referentes às variáveis monitoradas no sistema de tanque interativos, vazão de entrada do tanque TQ-2, nível do tanque TQ-3, vazão de saída de TQ-3 e os sinais de controle fracionário das válvulas FCV-1 e FCV-2, foram fornecidas à entrada do classificador em modo *on-line*, e a cada sequência, a saída do classificador foi comparada com a sequência fornecida para determinar o desempenho. As primeiras amostras de dados devem corresponder à operação normal do sistema, ou seja, a primeira regra criada deve descrever a operação normal, uma vez que o classificador *fuzzy* evolutivo inicia sem nenhuma regra *fuzzy* definida. Após o período inicial, durante a realização do diagnóstico, as falhas são detectadas e são criadas novas regras que descrevem cada tipo de falha. O rótulo de cada classe, que a define como operação normal ou como falha, é fornecida pelo operador do sistema no instante que cada nova regra é criada. Nos instantes seguintes, após à primeira ocorrência de uma falha, a intervenção do operador não se faz mais

necessária. Para realização do diagnóstico de falhas, os parâmetros do algoritmo de agrupamento recursivo foram definidos como: $\chi^2_{5,0,95} = 11,0705$; $F_{init} = 10^{-2}I$; $\alpha_{init} = 0,5$; $z_1 = 2$; $z_2 = 3$.

O resultado do diagnóstico de falhas em cada um dos tipos de cenários simulados são mostrados nas Figuras 6.34 - 6.38, onde pode-se comparar a saída estimada (sequência de falhas classificadas) pelo classificador *fuzzy* evolutivo com a saída desejada (sequência de falhas selecionadas), a partir das amostras de dados de entrada. Os resultados obtidos demonstram que o classificador foi capaz de diagnosticar corretamente todas as falhas do sistema de tanques interativos. Os resultados demonstram ainda que a ocorrência de alarmes falsos ou de erros de classificação é baixa, mesmo no cenário com o maior número de falhas possível, apesar da presença de ruído nas amostras de dados.

Todos os resultados das simulações de diagnóstico de falhas com o sistema de tanques interativos obtidos com o classificador *fuzzy* evolutivo proposto nesse trabalho foram comparados com os resultados obtidos utilizando o classificador proposto por Lemos et al. (2013). Para realização das simulações, foram adotados os valores dos parâmetros do classificador sugeridos pelos autores e definidos como: $w = 100$, $\lambda = 0,001$, $\alpha = 0,01$, $T_{\mu_y} = 0,01$.

Os resultados das simulações obtidos com ambos os classificadores em relação às métricas de detecção de falhas estão resumidos na Tabela 6.9. Os resultados apresentados demonstram que o classificador *fuzzy* evolutivo obteve altos índices de detecção de falha e acurácia em todos os cenários, com valores acima de 94% e 97%, respectivamente. Observa-se ainda que não houve ocorrência de alarme falso. Esses resultados comprovam a eficiência do classificador na detecção das falhas simuladas no sistema de tanques interativos. O classificador proposto por Lemos et al. (2013), apresentou índices de detecção de falhas e acurácia menores, mas também sem ocorrência de alarme falso.

Os resultados das simulações obtidos com ambos os classificadores em relação às métricas de classificação de falhas estão resumidos na Tabela 6.10. Os resultados apresentados demonstram que o classificador *fuzzy* evolutivo obteve alto índice de isolamento de falha em todos os cenários, com valor médio de aproximadamente 98%, e em todos os cenários a intervenção do operador na classificação das falhas foi menor ou igual a 0.24%. Esses resultados comprovam a capacidade do classificador de diagnosticar automaticamente a quase totalidade das falhas após a primeira ocorrência destas e evidencia a sua capa-

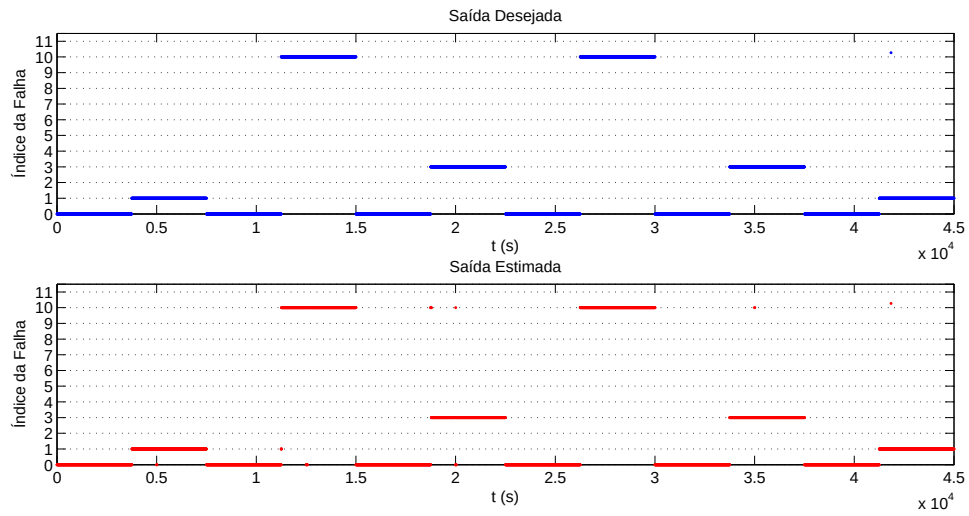


Figura 6.23: Saída desejada e saída estimada pelo classificador em cenário de 3 falhas simuladas no sistema de tanques interativos.

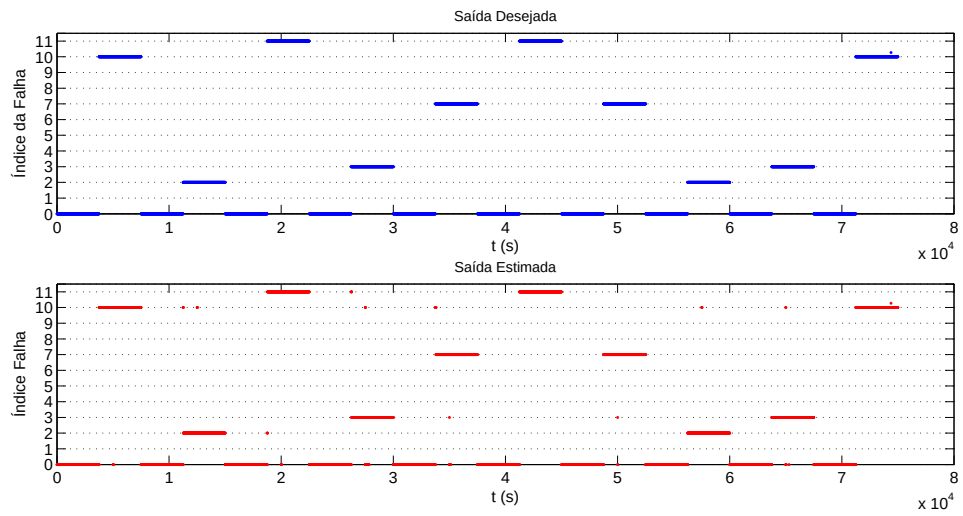


Figura 6.24: Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de tanques interativos.

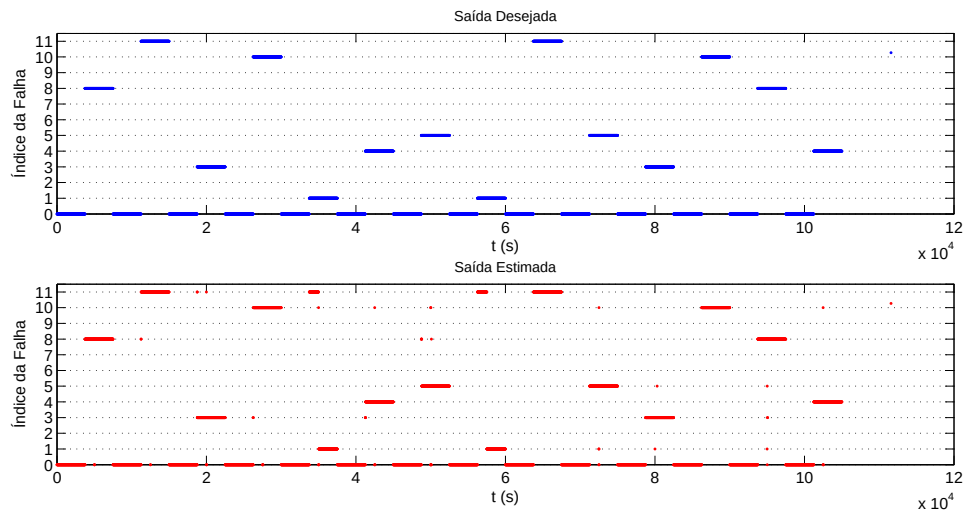


Figura 6.25: Saída desejada e saída estimada pelo classificador em cenário de 7 falhas simuladas no sistema de tanques interativos.

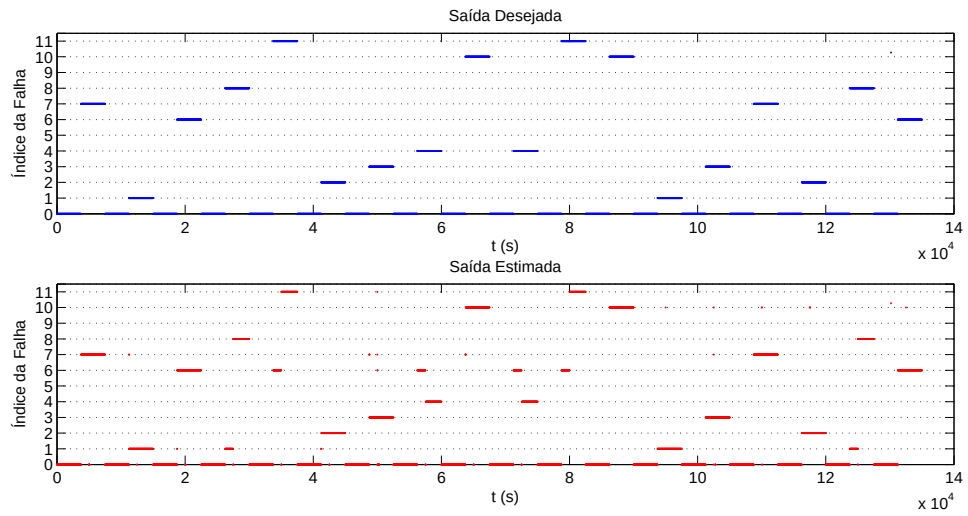


Figura 6.26: Saída desejada e saída estimada pelo classificador em cenário de 9 falhas simuladas no sistema de tanques interativos.

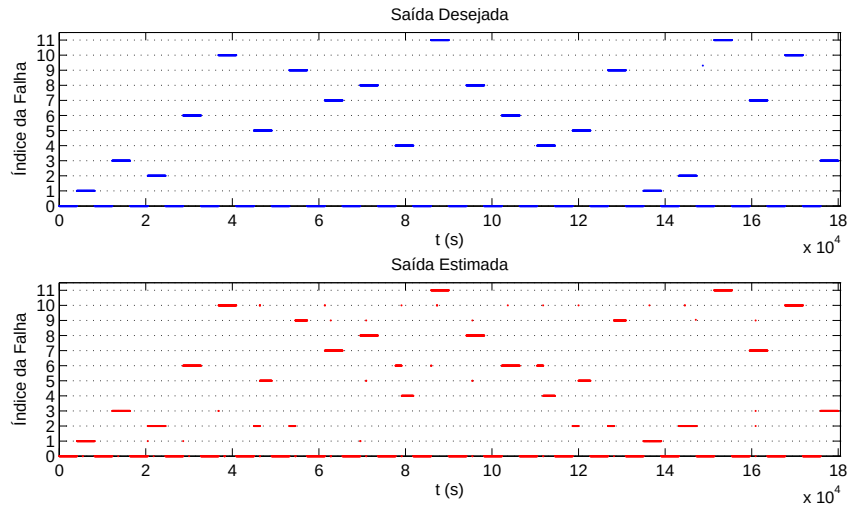


Figura 6.27: Saída desejada e saída estimada pelo classificador em cenário de 11 falhas simuladas no sistema de tanques interativos.

cidade de aprendizado. De forma geral, o desempenho do classificador proposto por Lemos et al. (2013) na classificação das falhas em relação ao classificador proposto nesse trabalho foi menor e a intervenção do operador foi maior.

Os resultados das simulações obtidos com ambos os classificadores em relação às métricas de tempo na detecção e na classificação das falhas estão resumidos na Tabela 6.11. Os valores são apresentados em quantidade de períodos de amostragem das variáveis monitoradas, nesse caso, $Ta = 2s$. O tempo médio de detecção da falha obtido nas simulações com o classificador *fuzzy* evolutivo foi de aproximadamente 19 períodos, e o tempo médio de isolamento da falha foi de aproximadamente 3 períodos. Considerando a diferença entre os valores do tempo de detecção da falha e do tempo de isolamento da falha, os resultados demonstram que a classificação das falhas é mais rápida após a primeira ocorrência da cada tipo de falha em função da existência das regras *fuzzy* e rótulos das falhas detectadas na base de dados do classificador, e da não intervenção do operador nesse caso. Os resultados das simulações com classificador proposto por Lemos et al. (2013) resultaram em tempos médios de detecção da falha e isolamento da falha de aproximadamente 10 períodos e 2 períodos, respectivamente, demonstrando que em função dos diferentes algoritmos de agrupamento empregados em cada classificador, o classificador proposto por Lemos et al. (2013) tem resposta mais rápida que o classificador proposto nesse trabalho.

Tabela 6.9: Desempenho na detecção de falhas simuladas no sistema de tanques interativos.

Cenário	Proposto			Lemos et al. (2013)		
	POD(%)	POFA(%)	ACC(%)	POD(%)	POFA(%)	ACC(%)
3 falhas	99,21	0,00	99,57	90,13	0,00	95,05
5 falhas	99,25	0,00	99,63	89,59	0,00	94,77
7 falhas	99,10	0,00	99,55	87,28	0,00	93,49
9 falhas	96,25	0,00	98,12	86,48	0,00	92,65
11 falhas	94,63	0,00	97,31	86,65	0,00	91,56

Tabela 6.10: Desempenho na classificação de falhas simuladas no sistema de tanques interativos.

Cenário	Proposto		Lemos et al. (2013)	
	FIR(%)	OIR(%)	FIR(%)	OIR(%)
3 falhas	99,56	0,15	95,05	0,31
5 falhas	99,63	0,24	94,77	0,33
7 falhas	98,80	0,14	93,35	0,34
9 falhas	97,91	0,15	92,37	0,44
11 falhas	96,85	0,15	90,73	0,36

Para avaliar a robustez do classificador *fuzzy* evolutivo à presença de *outliers* nos dados do sistema de tanques interativos foi realizando uma simulação onde, em um cenário de 5 falhas, foram acrescentadas às amostras de dados *outliers*, isto é, amostras de dados artificiais distribuídas aleatoriamente no espaço das variáveis de entrada. A Figura 6.28 mostra o resultado do diagnóstico de falhas no cenário em questão, onde pode-se comparar a saída estimada pelo classificador *fuzzy* evolutivo (sequência de falhas classificadas) com a saída desejada (sequência de falhas selecionadas), a partir das amostras de dados de entrada com a presença de *outliers*.

O resultado da simulação de diagnóstico de falhas com presença de *outliers* demonstrou que o classificador *fuzzy* evolutivo foi capaz de detectar e diagnosticar corretamente todas as falhas do sistema de tanques interativos mesmo com

Tabela 6.11: Tempos de detecção e classificação por tipo de falha simulada no sistema de tanques interativos.

Índice	Proposto		Lemos et al. (2013)	
	DDT(Ta)	IDT(Ta)	DDT(Ta)	IDT(Ta)
1	21	1	5	1
2	26	7	6	2
3	17	1	6	1
4	23	1	22	3
5	20	5	5	2
6	27	2	22	3
7	17	3	5	1
8	20	1	22	3
9	11	5	5	1
10	16	1	6	2
11	14	1	5	1

a presença de *outliers*, mostrando que o classificador foi capaz de distinguir de forma correta os dados válidos dos *outliers*. Os resultados dessa simulação são apresentados na Tabela 6.12 e na Tabela 6.13, comprovando que o classificador obteve desempenho muito próximo no diagnóstico de falhas com e sem a presença de *outliers* nos dados de entrada, e ainda sem ocorrência de alarmes falsos. Nessa simulação, o classificador proposto por Lemos et al. (2013) apresentou menor robustez em relação ao classificador proposto neste trabalho, uma vez que aquele apresentou maiores diferenças que este nos índices probabilidade de detecção de falha e taxa de isolamento de falha obtidos nos cenários com e sem a presença de *outliers*.

6.3.2 Validação do classificador

Para validação do classificador *fuzzy* evolutivo proposto nesse trabalho foram comparados os resultados obtidos no diagnóstico de falhas do sistema de tanques interativos utilizando dados reais, com os resultados utilizando dados de

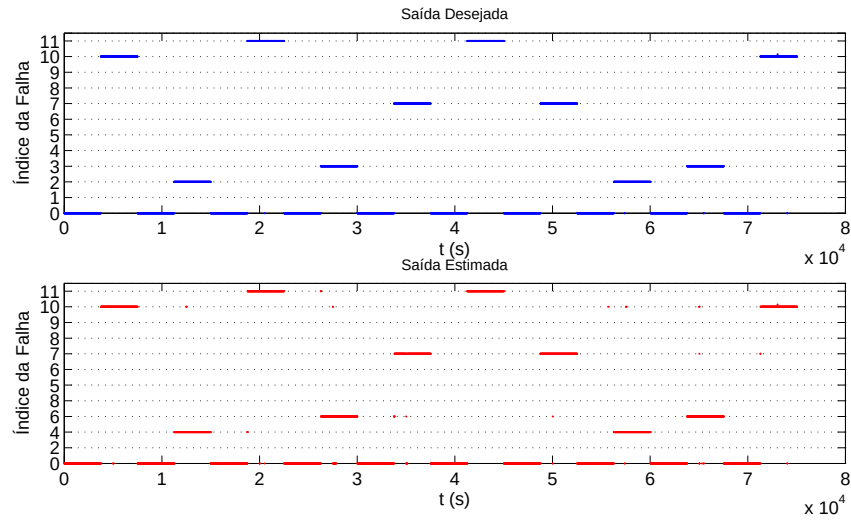


Figura 6.28: Saída desejada e saída estimada pelo classificador em cenário de 5 falhas simuladas no sistema de tanques interativos com presença de *outliers*.

Tabela 6.12: Desempenho na detecção de falhas simuladas no sistema de tanques interativos com presença de *outliers*.

Cenário	Proposto			Lemos et al. (2013)		
	POD(%)	POFA(%)	ACC(%)	POD(%)	POFA(%)	ACC(%)
sem outliers	99,25	0,00	99,63	89,59	0,00	94,77
com outliers	99,13	0,11	99,51	82,05	0,74	90,66

Tabela 6.13: Desempenho na classificação de falhas simuladas no sistema de tanques interativos com presença de *outliers*.

Cenário	Proposto		Lemos et al. (2013)	
	FIR(%)	OIR(%)	FIR(%)	OIR(%)
sem outliers	99,63	0,24	94,77	0,33
com outliers	99,24	0,30	89,81	0,48

simulação. Os dados reais utilizados foram adquiridos em experimentos realizados na planta real por Pereira (1996), com 34 condições de falha e 3 condições de operação normal onde, durante os experimentos, os parâmetros da planta foram alterados a fim de obter-se comportamentos de falhas diferentes. De forma semelhante às simulações de falhas, nos experimentos a planta inicia em operação normal e a falha é provocada no instante $t = 500s$. Na aquisição das amostras de dados reais também foi empregado um período de amostragem das variáveis monitoradas de 2s. É importante ressaltar que a validação foi realizada a partir dos dados reais coletados previamente e não em tempo real.

As Figuras 6.29 - 6.33 mostram, como exemplo, as curvas da vazão de entrada no tanque TQ-2 (q_i), nível do tanque TQ-3 (h_3), vazão de saída do tanque TQ-3 (q_o) e os sinais de controle fracionário das válvulas FCV-1 e FCV-2 (u_1 e u_2 , respectivamente) nas falhas da planta real. As falhas apresentadas correspondem aos experimentos de agarramento da válvula FCV-1, desligamento da bomba BA-2, entupimento no duto entre TQ-2 e TQ-3, vazamento no duto entre TQ-2 e TQ-3 e falha no sensor de nível de TQ-3, respectivamente.

Nos gráficos mostrados observa-se que os sinais apresentam variações abruptas após a ocorrência de determinadas falhas. Essas variações ocorrem devido ao intertravamento de proteção incorporado à lógica de controle da planta real. Tal proteção é utilizada para evitar que o nível do tanque TQ-2 atinja o nível máximo e evitar que o nível do tanque TQ-3 atinja um nível abaixo de 4% do mínimo.

Assim como na simulação de falhas, os dados reais foram organizados em diferentes cenários com sequências de 3 a 11 tipos de falhas selecionadas aleatoriamente dentro do conjunto de tipos de falhas, com intervalos de operação normal entre as falhas. Da mesma forma, as variáveis, os procedimentos e os parâmetros do algoritmo de agrupamento recursivo adotados para o diagnóstico de falhas com dados reais foram os mesmos utilizados na simulação.

Os resultados do diagnóstico de falhas com dados reais em cada um dos tipos de cenários são mostrados nas Figuras 6.34 - 6.38, onde pode-se comparar a saída estimada (sequência de falhas classificadas) pelo classificador *fuzzy* evolutivo com a saída desejada (sequência de falhas selecionadas), a partir das amostras de dados de entrada. Os resultados obtidos demonstram que o classificador proposto foi capaz de diagnosticar corretamente todas as falhas da planta real, apesar de um aumento na ocorrência de alarmes falsos ou de er-

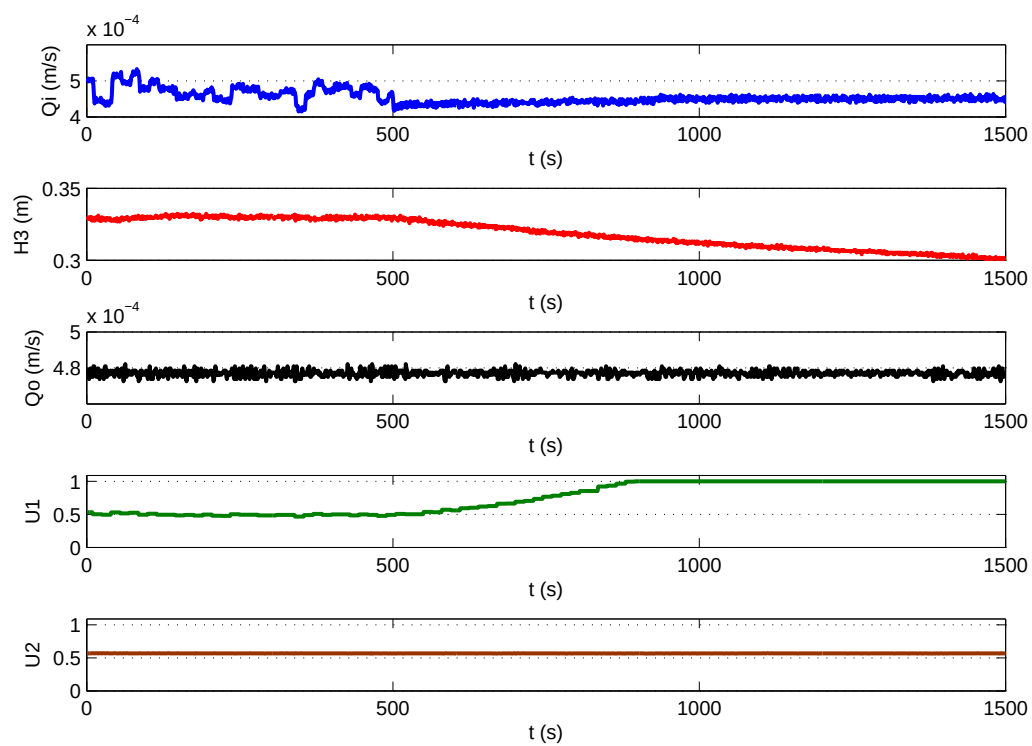


Figura 6.29: Falha na planta real: agarramento da válvula FCV-1.

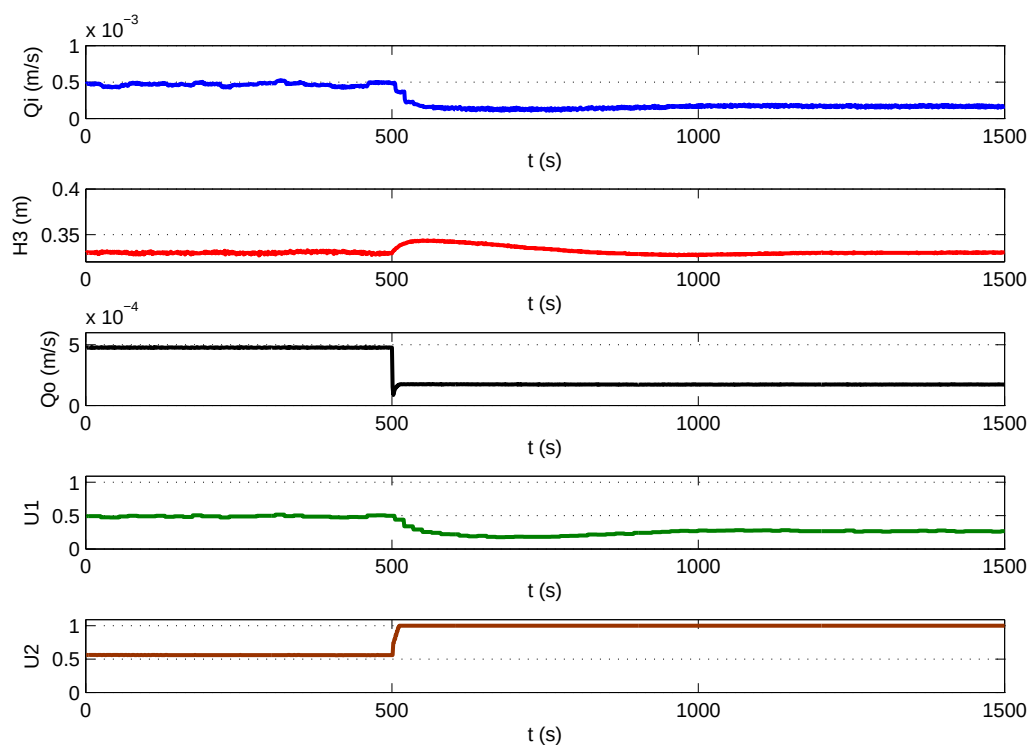


Figura 6.30: Falha na planta real: desligamento da bomba BA-2.

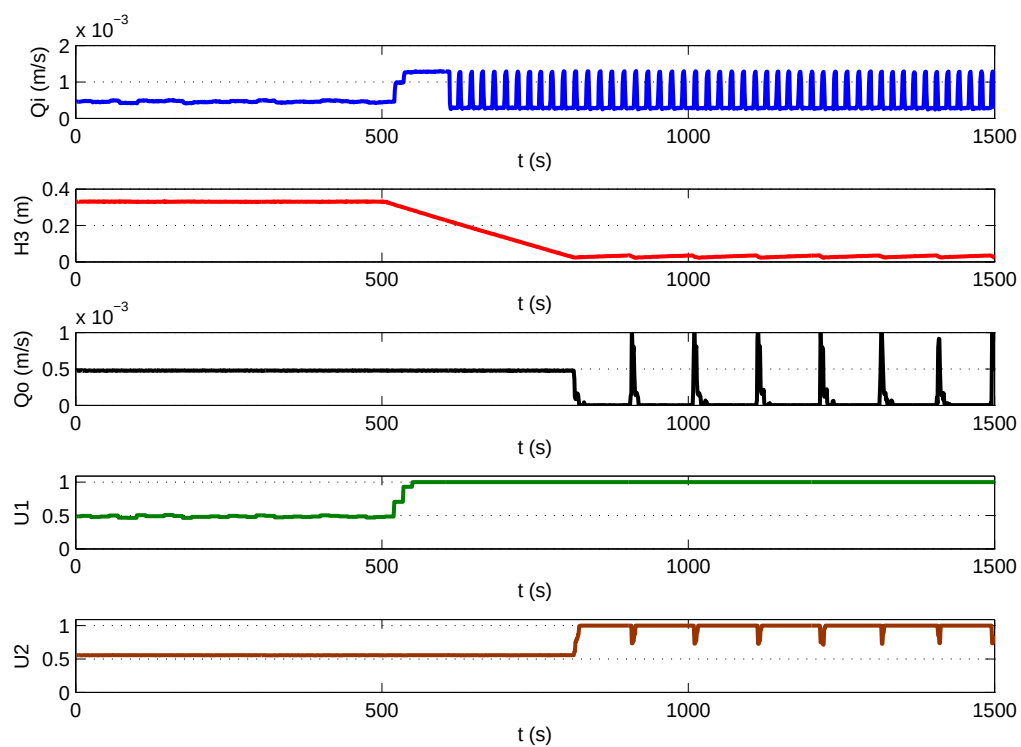


Figura 6.31: Falha na planta real: entupimento no duto entre TQ-2 e TQ-3.

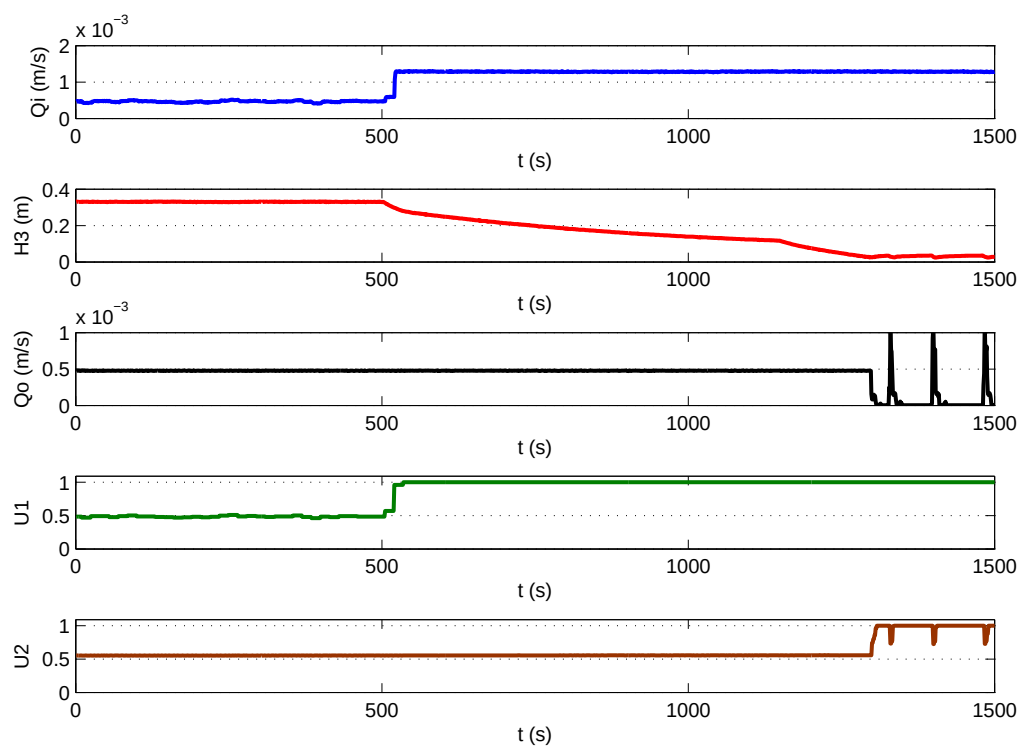


Figura 6.32: Falha na planta real: vazamento no duto entre TQ-2 e TQ-3.

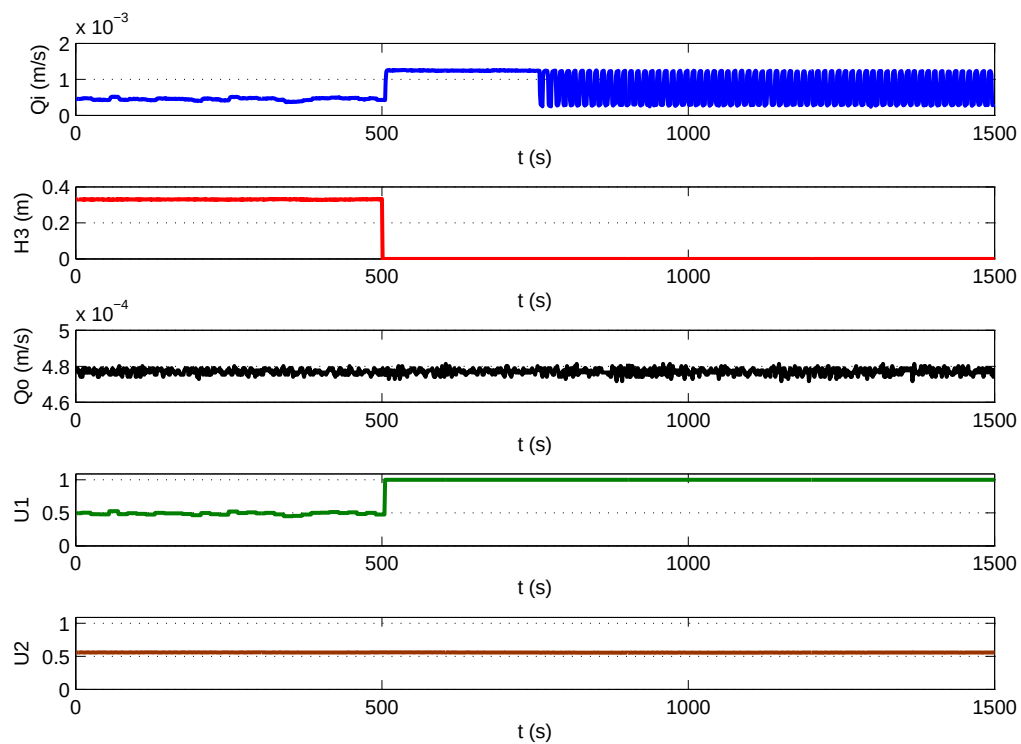


Figura 6.33: Falha na planta real: falha no sensor de nível de TQ-3.

ros de classificação em relação à simulação. Esses resultados demonstram que os dados reais de falhas impõem maior dificuldade para o diagnóstico, uma vez que estas incorporam naturalmente as variações do processo e ruídos de medição comumente encontrados nas plantas reais.

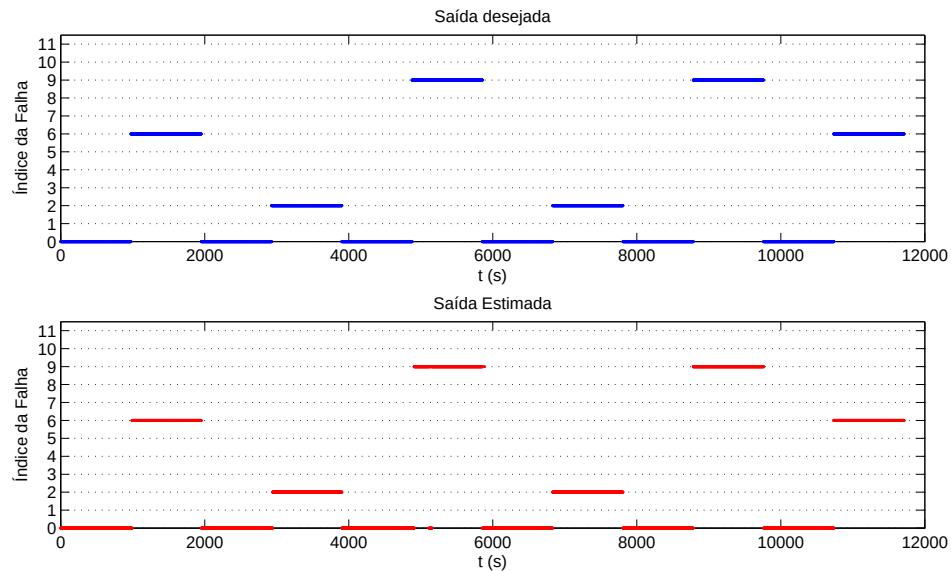


Figura 6.34: Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 3 falhas no sistema de tanques interativos.

As Tabelas 6.14 e 6.15 resumem os resultados obtidos pelo classificador *fuzzy* evolutivo com dados reais. Em relação às métricas de detecção de falhas, as diferenças entre os valores médios dos índices de detecção de falha e acurácia com dados reais e com dados de simulações foram menores que 3%, porém observa-se que, com dados reais houve ocorrência de alarme falso nos cenários de maior número de falhas. Apesar disso, a probabilidade de alarme falso em todos os cenários foi menor que 0,3%. Em relação às métricas de classificação de falhas, as diferenças entre os valores médios dos índices de isolamento de falhas dados reais e com dados de simulações foram menores que 2%. E em relação às métricas de tempos na detecção e na classificação de falhas observou-se que, apesar das diferenças entre os valores dos índices de tempo de detecção e tempo

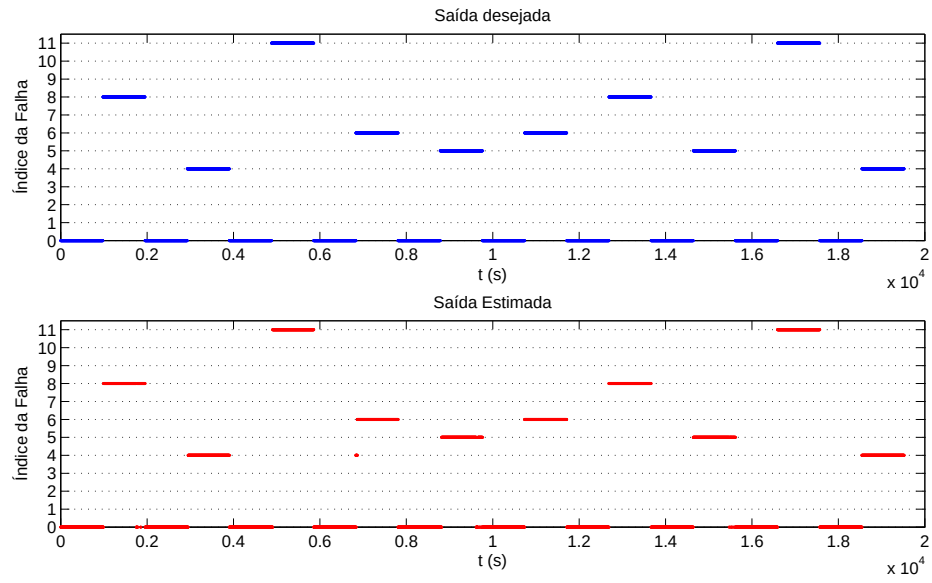


Figura 6.35: Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 5 falhas no sistema de tanques interativos.

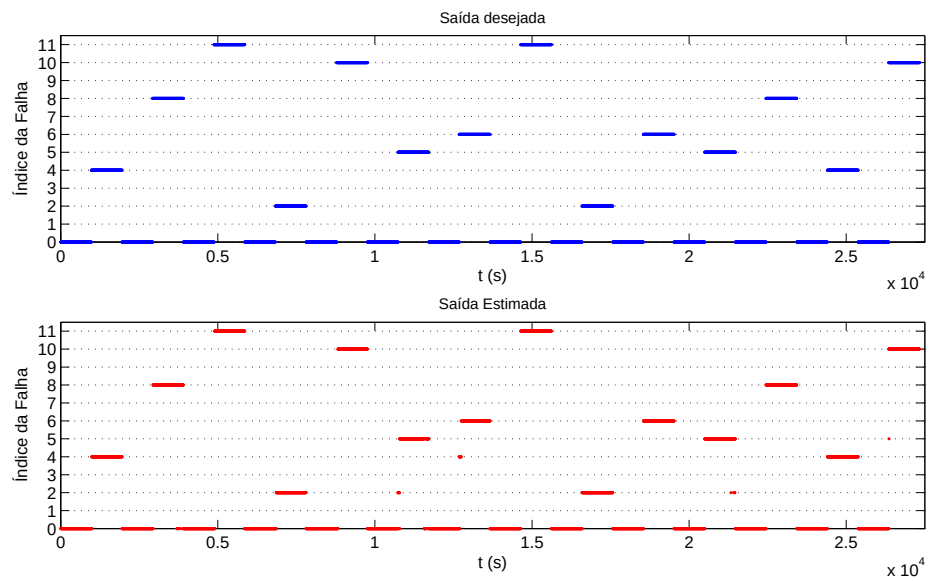


Figura 6.36: Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 7 falhas no sistema de tanques interativos.

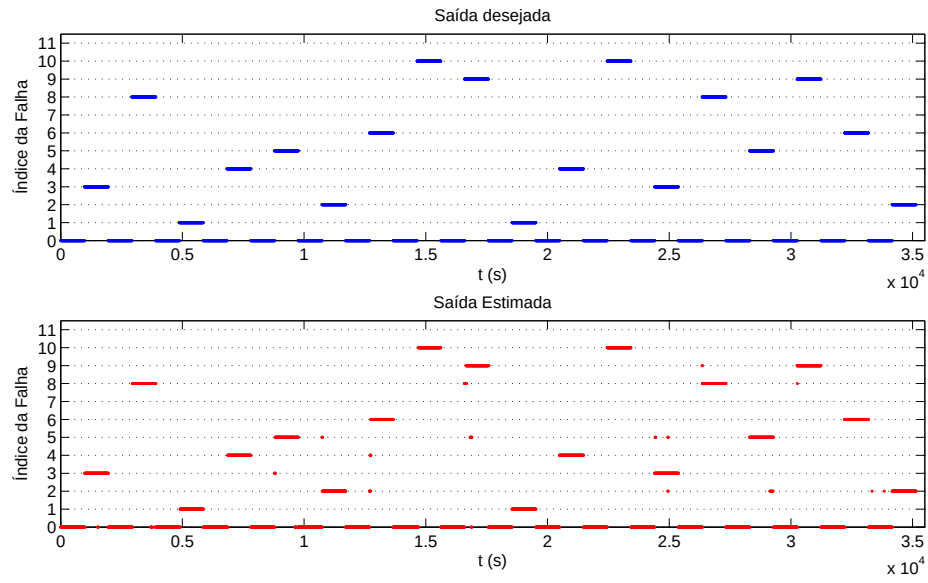


Figura 6.37: Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 9 falhas no sistema de tanques interativos.

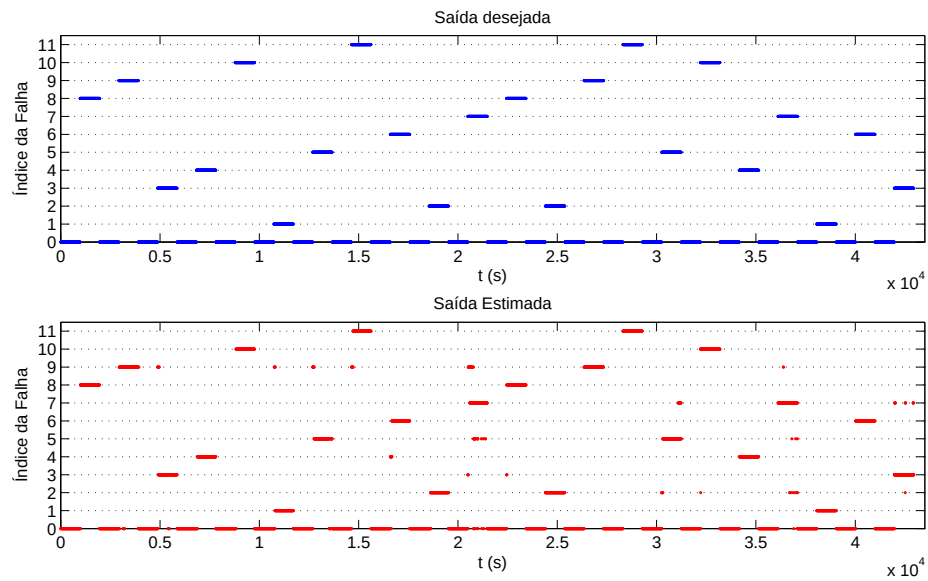


Figura 6.38: Saída desejada e saída estimada pelo classificador proposto na validação com cenário de 11 falhas no sistema de tanques interativos.

de isolamento de falha com dados reais e com dados de simulações, o comportamento do classificador foi o mesmo, uma vez que a classificação das falhas foi mais rápida após a primeira ocorrência da cada tipo de falha.

Tabela 6.14: Desempenho na validação de detecção de falhas do sistema de tanques interativos.

Cenário	Dados de simulação			Dados reais		
	POD(%)	POFA(%)	ACC(%)	POD(%)	POFA(%)	ACC(%)
3 falhas	99,21	0,00	99,57	98,90	0,00	99,45
5 falhas	99,25	0,00	99,63	98,73	0,00	99,34
7 falhas	99,10	0,00	99,55	98,76	0,00	99,37
9 falhas	96,25	0,00	98,12	98,74	0,13	99,29
11 falhas	94,63	0,00	97,31	98,71	0,23	99,24

Tabela 6.15: Desempenho na validação de classificação de falhas do sistema de tanques interativos.

Cenário	Dados de simulação		Dados reais	
	FIR(%)	OIR(%)	FIR(%)	OIR(%)
3 falhas	99,56	0,15	99,10	0,09
5 falhas	99,63	0,24	98,54	0,08
7 falhas	98,80	0,14	97,84	0,07
9 falhas	97,91	0,15	96,91	0,07
11 falhas	96,85	0,15	95,99	0,06

Capítulo 7

Conclusão

O avanço da tecnologia tem levado ao surgimento de máquinas e equipamentos complexos em muitas áreas, tornando o seu gerenciamento e manutenção mais difíceis e, conseqüentemente, aumentando a ocorrência de falhas. Métodos de diagnóstico de falhas capazes de lidar com esses sistemas complexos foram propostos na literatura, especialmente os métodos baseados em dados com uso de sistemas inteligentes, tais como redes neurais artificiais e redes *neurofuzzy*. Para sistemas dinâmicos não estacionários complexos, o diagnóstico de falhas deve ser adaptativo, uma vez que nesses sistemas as características se alteram ao longo do tempo. Uma abordagem promissora para esses casos é aquela baseada no uso de sistemas inteligentes evolutivos, que são sistemas capazes de alterar tanto sua estrutura quanto seus parâmetros a partir das amostras de dados de entrada, conseguindo assim se adaptar ao sistema dinâmico em questão. Além disso, os sistemas inteligentes evolutivos não exigem conhecimento prévio do sistema dinâmico uma vez que o aprendizado é realizado de forma contínua e incremental a partir das amostras de dados adquiridas de modo *on-line* e em tempo real. Para a aplicação em diagnóstico de falhas é necessário que o sistema inteligente evolutivo seja robusto a *outliers* e ruído, a fim de garantir a eficácia do diagnóstico. Neste contexto, esse trabalho propôs nova abordagem para implementação de sistemas inteligentes evolutivos robustos.

Inicialmente, esse trabalho propôs um algoritmo de agrupamento recursivo não supervisionado. Nesse algoritmo foi empregado um novo mecanismo para atualização do agrupamento baseado em um método de detecção de *drift*. Com esse mecanismo, a atualização do agrupamento não depende apenas da medida

de similaridade entre as amostras de dados e os centros de grupos, como na maioria dos algoritmos de agrupamento recursivos encontrados na literatura, mas depende também do monitoramento do contexto dos dados. O algoritmo proposto combina as vantagens do algoritmo GK recursivo, especialmente a habilidade de identificar grupos de diferentes formas e com diferentes orientações no espaço em modo *on-line* com o método eficaz de detecção de mudança no contexto dos dados do algoritmo DDM. Além disso, no algoritmo proposto foi incorporado um mecanismo de união de grupos para permitir a eliminação de grupos redundantes. Dessa forma, o algoritmo de agrupamento recursivo com detecção de *drift* apresenta como principais características a robustez à presença de *outliers* e ruído nos dados de entrada, e ainda maior precisão na identificação do agrupamento.

Em seguida, esse trabalho propôs um classificador *fuzzy* evolutivo para aplicação em diagnóstico de falhas de sistemas dinâmicos. O classificador proposto emprega o algoritmo de agrupamento recursivo com detecção de *drift* para criação e atualização da base de regras *fuzzy*, o que confere ao classificador a capacidade de aprendizado contínuo e incremental em modo *on-line* e em tempo real, além da maior robustez a *outliers* e ruído. No classificador proposto, a base de regras *fuzzy* evolui a partir da definição dos grupos realizada pelo algoritmo de agrupamento, sendo utilizados os centros e as matrizes de dispersão dos grupos como parâmetros das funções de pertinência nos antecedentes das regras. Foram utilizadas funções de pertinência Gaussianas multivariáveis, o que permite preservar a informação referente às interações entre as variáveis de entrada.

Para aplicação do classificador *fuzzy* evolutivo no problema de diagnóstico de falhas foi proposta uma metodologia onde as falhas são identificadas somente a partir das amostras de dados de entrada e com auxílio do operador do sistema. A partir do agrupamento de dados, o classificador associa cada grupo a uma classe, que por sua vez representa uma condição de operação do sistema dinâmico. O rótulo das classes deverá ser fornecido pelo operador do sistema, que definirá a condição de operação como normal ou como uma falha específica. As informações referentes ao diagnóstico (regras *fuzzy* e rótulos das classes) são armazenadas em uma base de conhecimento unificada, atualizada durante a operação do sistema. Nessa metodologia, o classificador inicia sem nenhuma regra e nenhuma classe definida, mas após o período inicial de funcionamento

do sistema, a base de conhecimento conterá as informações referentes as falhas definidas até então, de forma que uma nova ocorrência de uma dessas falhas será diagnosticada automaticamente, sem intervenção do operador. Assim, o classificador proposto apresenta como principal característica a capacidade de aprendizado das falhas de forma contínua e incremental, não necessitando do conhecimento do modelo do sistema dinâmico *à priori* e nem de dados de histórico do processo. Essa característica permite que o classificador se adapte ao sistema dinâmico em questão, e torne possível o diagnóstico de falhas não conhecidas previamente.

Dois sistemas dinâmicos não estacionários foram utilizados nesse trabalho para avaliação de desempenho do classificador *fuzzy* evolutivo. O primeiro consiste em um sistema de acionamento C.C, constituído por dois conversores estáticos controlados, uma máquina de corrente contínua e uma carga mecânica. O segundo consiste em um sistema de tanques interativos, constituído por dois tanques, um reservatório, duas válvulas pneumáticas e duas bombas. Foram implementados simuladores de ambos os sistemas, o que permitiu a geração de dados simulados de operação normal e de falha dos sistemas. Esses dados foram organizados em vários cenários, onde foram intercalados períodos de operação normal com as falhas específicas de cada sistema. *Outliers* e ruídos foram inseridos artificialmente nos dados simulados a fim de avaliar a robustez do classificador. Esses cenários foram empregados na simulação do diagnóstico de falhas com classificador proposto. Os resultados obtidos nas simulações demonstraram que, de forma geral, o classificador foi capaz de diagnosticar todas as falhas corretamente, apresentando altos índices de detecção e classificação de falhas, mesmo com a presença de *outliers* e ruídos nos dados. Os resultados demonstraram ainda que o classificador apresenta tempos de detecção e classificação da falha baixos após a primeira ocorrência de cada falha, e taxa de intervenção do operador baixa durante o processo de diagnóstico. As simulações foram realizadas com outro classificador reportado pela literatura para comparação, e os resultados obtidos sugerem que o desempenho do classificador proposto nesse trabalho é superior em relação aos índices que avaliam a detecção e classificação de falhas, robustez a *outliers* e ruídos, porém inferior em relação aos índices que avaliam o tempo de detecção e classificação da falha. Para validação do classificador proposto foram comparados os resultados do diagnóstico de falhas no sistema de tanques interativos obtidos com dados de

simulação com os resultados do diagnóstico de falhas obtidos com dados reais. Esses dados foram adquiridos em experimentos de falhas realizados na planta real. Os resultados obtidos com dados reais foram muito próximos daqueles obtidos com dados de simulação, comprovando a aplicabilidade do classificador proposto no diagnóstico de falhas de máquinas e equipamentos reais.

Dessa forma, conclui-se que o diagnóstico de falhas baseado em sistema inteligente evolutivo proposto nesse trabalho representa uma alternativa promissora para aplicação em problemas reais, onde naturalmente os sistemas são dinâmicos, não estacionários e complexos, o que requer a realização dessa tarefa a partir de dados obtidos do processo em tempo real.

Os resultados obtidos por esse trabalho sugerem a continuidade do mesmo com a realização de novos estudos visando incorporar novas características à metodologia diagnóstico de falhas baseado em sistema inteligente evolutivo, tais como: seleção automática das características das variáveis de entrada do classificador, redução do número de regras baseada em utilidade, capacidade de classificação de múltiplas falhas simultaneamente. Essas novas características permitiriam que o sistema inteligente evolutivo alcançasse maior autonomia, desempenho e aplicabilidade em problemas reais em relação ao sistema proposto.

Sugere-se ainda a implementação de um sistema inteligente evolutivo para prognóstico de falhas, baseado no algoritmo de agrupamento recursivo proposto. Para essa tarefa, o sistema inteligente evolutivo teria a capacidade de modelar o sistema dinâmico em questão, de forma a permitir a previsão do seu comportamento no futuro, e assim realizar o prognóstico das falhas. Como consequência, tornaria possível a implementação de um sistema de diagnóstico e prognóstico de falhas integrado, empregando dois sistemas inteligentes evolutivos com uma base de conhecimento unificada, de forma que esse sistema integrado permita a realização de ambas as tarefas em uma dada aplicação, com aprendizado contínuo e incremental a partir dos dados de entrada adquiridos de modo *on-line* e em tempo real.

Finalmente sugere-se a investigação do sistema inteligente evolutivo proposto em aplicações de diagnóstico de falhas de máquinas e equipamentos em tempo real, onde o sistema seria implementado em sistema computacional dedicado. Espera-se que testes a serem realizados nesse sistema permitam propor novas melhorias a serem desenvolvidas para aumentar a confiabilidade do diagnóstico.

Referências Bibliográficas

- J. V. Abellan-Nebot and F. R. Subirón. A review of machining monitoring systems based on artificial intelligence process models. *The International Journal of Advanced Manufacturing Technology*, 47:237–257, 2010.
- A. Abraham and C. Grosan. Engineering Evolutionary Intelligent Systems: Methodologies, Architectures and Reviews. In A. Abraham, C. Grosan, and W. Pedrycz, editors, *Engineering Evolutionary Intelligent Systems, Studies in Computational Intelligence*, pages 1–22. Springer Verlag, Germany, 2008.
- P. Angelov. *Evolving rule-based models: a Tool for Design of Flexible Adaptive Systems*. Springer-Verlag, London, UK, 2002.
- P. Angelov. Fuzzily connected multimodel systems evolving autonomously from data streams. *IEEE Transactions on Systems Man and Cybernetics Part B - Cybernetics*, 41(4):898–910, 2011.
- P. Angelov and R. Buswell. Evolving rule-based models: A tool for intelligent adaptation. In *Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, volume 2, page 1062–1067, 2001.
- P. Angelov and D. Filev. An approach to Online identification of Takagi-Sugeno fuzzy models. *IEEE Transactions on Systems Man and Cybernetics Part B - Cybernetics*, 34(1):484–498, 2004.
- P. Angelov and D. Filev. SimpNet: a simplified method for learning evolving takagi-sugeno fuzzy models. In *The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ '05.*, page 1068–1073, 2005.
- P. Angelov and N. Kasabov. Evolving Intelligent Systems - eIS. *IEEE SMC eNewsLetter*, 15:1–13, 2006.

- P. Angelov and R. Yager. A new type of simplified fuzzy rule-based system. *International Journal Of General Systems*, 41(2):163–185, 2012.
- P. Angelov and X. Zhou. Evolving fuzzy systems from data streams in real-time. In *International Symposium on Evolving Fuzzy Systems*, page 29–35, 2006.
- P. Angelov, D. Filev, and N. Kasabov. *Evolving Intelligent Systems: Methodology and Applications*. John Wiley & Sons, New York, USA, 2010.
- K. J. Aström and B. Wittenmark. *Adaptive Control*. Addison Wesley, Reaching, Boston, Ma. USA, 2nd edition, 1995.
- R. Babuska and H. B. Verbruggen. An Overview of Fuzzy Modeling for Control. *Control Engineering Practice*, 4(11):1593–1606, 1996.
- L. M. R. Baccharini, V. V. R. Silva, B. R. Menezes, and W. M. Caminhas. SVM practical industrial application for mechanical faults diagnostic. *Expert Systems with Applications*, 38(6):6980–6984, 2011.
- A. R. Braga, F. G. Jota, C. M. Polito, and R. T. Pena. Development of an interacting tank system for the study of advanced process control strategies. In *Proceedings of the 38th Midwest Symposium on Circuits and Systems*, 1995.
- J. M. F. Calado, J. Korbicz, K. Patan, R. J. Patton, and J. M. G. S. da Costa. Softcomputing approaches to fault diagnosis for dynamic systems. *European Journal Control*, 7(2-3):248–286, 2001.
- W. Caminhas and R. Takahashi. Dynamic system failure detection and diagnosis employing sliding mode observers and fuzzy neural networks. In *Proceedings of Joint 9th IFSA World Congress and 20th NAFIPS international Conference*, pages 304–309, 2001.
- W. M. Caminhas. *Estratégias de Detecção e Diagnóstico de Falhas em Sistemas Dinâmicos*. PhD thesis, Universidade Estadual de Campinas, Campinas, Brasil, 1997.
- C. T. Chang and J. I. Hwang. Simplification techniques for EKF computations in fault diagnosis - suboptimal gains. *Chemical Engineering Science*, 53(22):3853–3862, 1994.

- P.-C. Chang, Y.-W. Wang, and C.-H. Liu. The development of a weighted evolving fuzzy neural network for pcb sales forecasting. *Expert Systems with Applications*, 32(1):86–96, 2007.
- J. Chen and R. J. Patton. *Robust Model-Based Fault Diagnosis for Dynamic Systems*. Kluwer Academic Publishers, Boston/Dordrecht/London, 1999.
- D. Chivala, L. F. Mendonça, J. M. C. Sousa, and J. M. G. S. da Costa. Application of evolving fuzzy modeling to fault tolerant control. *Evolving Systems*, 1(4):209–223, 2010.
- H. J. Cho and J. K. Park. An expert system for fault section diagnosis of power systems using fuzzy relations. *IEEE Transactions on Power Systems*, 12(1):342–347, 1997.
- M. F. S. V. D’Angelo and P. P. Costa. Detection of shorted turns in the field winding of turbogenerators using the neural network mlp. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, page 1930–1935, 2001.
- M. F. S. V. D’Angelo, R. M. Palhares, W. M. Caminhas, R. H. C. Takahashi, R. D. Maia, A. P. Lemos, and M. J. Inácio. Detecção de falhas: Uma revisão com aplicações. In A. P. Feltrin, C. R. Minussi, M. C. M. Teixeira, and R. A. R. Lazaro., editors, *Tutoriais do XVIII Congresso Brasileiro de Automática*, pages 1 – 47. Cultura Acadêmica, São Paulo, Brasil, 2010.
- M. F. S. V. D’Angelo, R. M. Palhares, R. H. C. Takahashi, and R. H. Loschi. Fuzzy/bayesian change point detection approach to incipient fault detection. *IET Control Theory & Applications*, 5(4):539–551, 2011a.
- M. F. S. V. D’Angelo, R. M. Palhares, R. H. C. Takahashi, R. H. Loschi, L. M. R. Baccarini, and W. M. Caminhas. Incipient fault detection in induction machine stator-winding using a Fuzzy-Bayesian change point detection approach. *Applied Soft Computing*, 11(1):179–192, 2011b.
- M. F. S. V. D’Angelo, R. M. Palhares, L. B. Cosme, L. A. Aguiar, F. S. Fonseca, and W. M. Caminhas. Fault detection in dynamic systems by a fuzzy/bayesian network formulation. *Applied Soft Computing*, 21:647–653, 2014.

- D. Deng and N. Kasabov. Esom: An algorithm to evolve selforganizing maps from on-line data streams. In *International Joint Conference on Neural Networks (IJCNN'2000)*, page 3–8, 2000.
- K. P. Detroja, R. D. Gudi, and S. C. Patwardhan. A possibilistic clustering approach to novel fault detection and isolation. *Journal of Process Control*, 16(10):1055–1073, 2006.
- R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, New York, USA, second edition, 2000.
- R. Dunia, S. J. Qin, T. F. Edgar, and T. J. McAvoy. Identification of faulty sensors using principal component analysis. *American Institute of Chemical Engineers Journal*, 42(10):2797–2812, 1996.
- Y. Dutuit and A. Rauzy. Approximate estimation of system reliability via fault trees. *Reliability Engineering & System Safety*, 87(2):163–172, 2005.
- S. M. El-Shal and A. S. Morris. A fuzzy expert system for fault detection in statistical process control of industrial processes. *IEEE Transactions on Systems, Man, and Cybernetics*, Part C30(2):281–289, 2000.
- D. Filev and O. Georgieva. An extended version of the Gustafson–Kessel algorithm for evolving data stream clustering. In P. Angelov, D. Filev, and N. Kasabov, editors, *Evolving Intelligent Systems: Methodology and Applications*, pages 273–300. John Wiley & Sons, New York, USA, 2010.
- D. Filev, R. B. Chinnam, F. Tseng, and P. Baruah. An Industrial Strength Novelty Detection Framework for Autonomous Equipment Monitoring and Diagnostics. *IEEE Transactions on Industrial Informatics*, 6(4):767–779, 2010.
- P. M. Frank. On-line fault detection in uncertain nonlinear systems using diagnostic observers: a survey. *International Journal Systems Science*, 25(12):2129–2154, 1994.
- P. M. Frank and B. Koppen-Seliger. Fuzzy logic and neural network applications to fault diagnosis. *International Journal of Approximate Reasoning*, 16:67–88, 1997.

- B. Fritzke. Growing cell structures - a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9):1441–1460, 1994.
- J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In *Proceedings of 17th Brazilian Symposium on Artificial Intelligence*, pages 286–295, 2004.
- H. Gao, T. Chen, and L. Wang. Robust fault detection with missing measurements. *International Journal Control*, 81(5):804–819, 2008.
- D. E. Gustafson and W. C. Kessel. Fuzzy clustering with fuzzy covariance. In *Proceedings of IEEE Conference on Decision and Control*, page 761–766, 1979.
- T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer-Verlag, New York, Berlin, Heidelberg, second edition, 2001.
- V. J. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.
- R. Isermann. Process fault detection based on modeling and estimation methods – A survey. *Automatica (Journal of IFAC)*, 20(4):87–404, 1994.
- A. K. Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.
- J. S. R. Jang, C. T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing: a computational approach to learning and machine intelligence*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997.
- A. K. S. Jardine, D. Lin, and D. Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7):1483–1510, 2006.
- F. Jota, A. R. Braga, and R. Pena. Performance assessment of advanced process control algorithms using an interacting tank system. In *Proceedings of the Industry Applications Conference, 1995. 30th IAS Annual Meeting, IAS '95.*, volume 2, pages 1565–1571, 1995.

- N. Kasabov. ECOS: Evolving Connectionist Systems and the ECO Learning Paradigm. In *Proceedings of ICONIP'1998*, page 1232–1235, 1998.
- N. Kasabov. Evolving Fuzzy Neural Networks for Supervised/Unsupervised Online Knowledge-Based Learning. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 31(6):902–918, 2001.
- N. Kasabov and D. Filev. Evolving Intelligent Systems: Methods, Learning, & Applications. In *2006 International Symposium on Evolving Fuzzy Systems*, pages 8–18, 2006.
- N. Kasabov and Q. Song. DENFIS: Dynamic Evolving Neural-Fuzzy Inference System and Its Application for Time Series Prediction. *IEEE Transactions on Fuzzy Systems*, 10(2):144–154, 2002.
- S. Katipamula and M. R. Brambley. Methods for Fault Detection, Diagnostics and Prognostics for Building Systems - A Review Part II. *HVAC&R Research*, 11(2):169–187, 2005.
- P. M. Kelly. An algorithm for merging hyperellipsoidal clusters. Technical report, LA-UR-94-3306, Los Alamos National Laboratory, Los Alamos, NM, 1994.
- D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *VLDB 04: Proceedings of the 30th International Conference on Very Large Data Bases*, pages 180–191, 2004.
- R. Klinkenberg. Learning drifting concepts: example selection vs. example weighting. *Intelligent Data Analysis*, 8(3):281–300, 2004.
- R. Klinkenberg and T. Joachims. Detecting concept drift with support vector machines. In *Proceedings of ICML-00, 17th International Conference on Machine Learning*, pages 487–494, 2000.
- C. A. Laurentys, C. H. M. Bomfim, B. R. Menezes, and W. M. Caminhas. Design of a pipeline leakage detection using expert system: a novel approach. *Applied Soft Computing*, 11(1):1057–1066, 2011a.

- C. A. Laurentys, R. M. Palhares, and W. M. Caminhas. A novel artificial immune system for fault behavior detection. *Expert Systems with Applications*, 38:6957–6966, 2011b.
- W. S. Lee, D. L. Grosh, F. A. Tillman, and C. H. Lie. Fault tree analysis, methods and applications – A review. *IEEE Transactions on Reliability*, 34(3):194–203, 1985.
- D. Leite and F. Gomide. Evolving linguistic fuzzy models from data streams. In E. Trillas, P. Bonissone, L. Magdalena, and J. Kacprzyk, editors, *Combining Experimentation and Theory: A Hommage to Abe Mamdani (Studies in Fuzziness and Soft Computing)*, pages 209–223. Springer - Verlag, Berlin Heidelberg, 2011.
- D. Leite, F. Gomide, R. Ballini, and P. Costa. Fuzzy granular evolving modeling for time series prediction. In *Proceedings of the IEEE International Conference on Fuzzy Systems, FUZZ-IEEE '11*, pages 2794–2801, 2011.
- A. Lemos, W. Caminhas, and F. Gomide. Multivariable Gaussian Evolving Fuzzy Modeling System. *IEEE Transactions on Fuzzy Systems*, 19(1):91–104, 2011a.
- A. Lemos, W. Caminhas, and F. Gomide. Fuzzy evolving linear regression trees. *Evolving Systems*, 2(1):1–14, 2011b.
- A. Lemos, W. Caminhas, and F. Gomide. Adaptive Fault Detection and Diagnosis Using an Evolving Fuzzy Classifier. *Information Sciences*, 220:64–85, 2013.
- G. Leng, T. M. McGinnity, and G. Prasad. An Approach for On-Line Extraction of Fuzzy Rules Using a Self-Organising Fuzzy Neural Network. *Fuzzy Sets & Systems*, 150(2):211–243, 2005.
- J. A. Leonard and M. A. Kramer. Diagnosing dynamic faults using modular neural nets. *IEEE Expert*, 8(2):44–53, 1993.
- B. Li, T. Han, and F. Kang. Fault diagnosis expert system of semiconductor manufacturing equipment using a bayesian network. *International Journal of Computer Integrated Manufacturing*, 26(12):1161–1171, 2013.

- W. Li, H. Yue, S. Valle-Cervantes, and S. J. Qin. Recursive PCA for adaptive process monitoring. *Journal of Process Control*, 10(5):471–486, 2000.
- E. Lima, M. Hell, F. Gomide, and R. Ballini. *Evolving fuzzy modeling using participatory learning*, pages 67–87. John Wiley & Sons, New York, USA, 2010.
- E. Lughofer. FLEXFIS: A robust Incremental Learning Approach for Evolving Takagi–Sugeno Fuzzy Models. *IEEE Transactions on Fuzzy Systems*, 16(6):1393–1410, 2008.
- E. Lughofer. *Evolving Fuzzy Systems - Methodologies, Advanced Concepts and Applications*. Springer-Verlag, Berlin Heidelberg, 2011.
- E. Lughofer. Flexible evolving fuzzy inference systems from data streams (flex-fis++). In M. Sayed-Mouchaweh and E. Lughofer, editors, *Learning in Non-Stationary-Environments:Methods and Applications*, pages 205–246. Springer - Verlag, New York, 2012.
- E. Lughofer and C. Guardiola. Applying Evolving Fuzzy Models with Adaptive Local Error Bars to On-line Fault Detection. In *Proceedings of Genetic and Evolving Fuzzy Systems*, pages 35–40, 2008.
- E. Lughofer, P. Angelov, and X. Zhou. Evolving single-and multi-model fuzzy classifiers with FLEXFIS-class. In *Proceedings of FUZZ-IEEE*, page 363–368, 2007.
- L. Maciel, F. Gomide, R. Ballini, and R. Yager. Simplified evolving rule-based fuzzy modeling of realized volatility forecasting with jumps. In *Proceedings of the IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr'2013)*, pages 82–89, 2013.
- L. Maciel, F. Gomide, and R. Ballini. Enhanced evolving participatory learning fuzzy modeling: an application for asset returns volatility forecasting. *Evolving Systems*, 5(2):75–88, 2014.
- M. Maloof and R. Michalski. Selecting examples for partial memory learning. *Machine Learning*, 41:27–52, 2000.

- M. R. Maurya, R. Rengaswamy, and V. Venkatasubramanian. Assigned directed graph-based systematic framework for steady-state malfunction diagnosis inside control loops. *Chemical Engineering Science*, 61(6):1790–1810, 2006.
- M. R. Maurya, R. Rengaswamy, and V. Venkatasubramanian. Fault diagnosis using dynamic trend analysis: a review and recent developments. *Engineering Applications of Artificial Intelligence*, 20(2):133–146, 2007.
- T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, USA, 1997.
- W. Pedrycz and F. Gomide. *Fuzzy Systems Engineering - Toward Human-Centric Computing*. John Wiley & Sons, Hoboken, NJ, USA, 2007.
- M. F. S. Pereira. Desenvolvimento e Aplicação de um Métodos de Detecção e Identificação de Falhas Baseado em Observadores Dedicados. Master’s thesis, Universidade Federal de Minas Gerais, Belo Horizonte, Brasil, 1996.
- M. Petković, M. R. Rapaić, Z. D. Jeličić, and A. Pisano. On-line adaptive clustering for process monitoring and fault detection. *Expert Systems with Applications*, 39(11):10226–10235, 2012.
- S. Ploix and O. Adrot. Parity relations for linear uncertain dynamic systems. *Automatica*, 42(6):1553–1562, 2006.
- J. Ragot and D. Maquin. Fault measurement detection in an urban water supply network. *Journal of Process Control*, 16(9):887–902, 2006.
- T. Reuters. ISI Web of Science. <http://www.webofknowledge.com>, July 2014.
- H. J. Rong, N. Sundararajan, G. B. Huang, and P. Saratchandran. Sequential Adaptive Fuzzy Inference System (SAFIS) for nonlinear system identification and prediction. *Fuzzy Sets & Systems*, 157(9):1260–1275, 2006.
- H. J. Rong, N. Sundararajan, G. B. Huang, and G. S. Zhao. Extended sequential adaptive fuzzy inference system for classification problems. *Evolving Systems*, 2(2):71–82, 2011.
- R. Sebastião and J. Gama. A Study on Change Detection Methods. In *Proceedings of 14th Portuguese Conference on Artificial Intelligence*, pages 353–364, 2009.

- R. Sebastião, J. Gama, P. P. Rodrigues, and J. Bernardes. Monitoring Incremental Histogram Distribution for Change Detection in Data Streams. In *Proceedings of KDD Workshop on Knowledge Discovery from Sensor Data*, pages 25–42, 2008.
- J. Z. Sikorska, M. Hodkiewicz, and L. Ma. Prognostic modeling options for remaining useful life estimation by industry. *Mechanical Systems and Signal Processing*, 25(5):1803–1836, 2011.
- A. M. Silva, W. Caminhas, A. Lemos, and F. Gomide. A fast learning algorithm for evolving neo-fuzzy neuron. *Applied Soft Computing*, 14(Part B):194–209, 2014.
- G. C. Silva, R. M. Palhares, and W. M. Caminhas. Immune inspired fault detection and diagnosis: a Fuzzy-based approach of the negative selection algorithm and participatory clustering. *Expert Systems with Applications*, 39:12474–12486, 2012.
- T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1):116–132, 1985.
- R. H. C. Takahashi and P. L. D. Peres. Unknown input observers for uncertain systems: a unifying approach. *European Journal Control*, 5(2-4):261–275, 1999.
- R. H. C. Takahashi, R. M. Palhares, and P. L. D. Peres. Discrete-time singular observers: H_2/H_∞ optimality and unknown inputs. *International Journal Control*, 72(6):481–492, 1999.
- G. Vachtsevanos, F. Lewis, M. Roeme, A. Hess, and B. Wu. *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. John Wiley & Sons, New York, USA, 2006.
- H. Vedaam and V. Venkatasubramanian. Signed digraph based multiple fault diagnosis. *Computers and Chemical Engineering*, 21:655–660, 1997.
- V. Venkatasubramanian. Prognostic and diagnostic monitoring of complex systems for product lifecycle management: Challenges and opportunities. *Computers and Chemical Engineering*, 29(6):1253–1263, 2005.

- V. Venkatasubramanian, R. Rengaswamy, S. Kavuri, and K. Yin. A review of process fault detection and diagnosis Part I: Quantitative model-based methods. *Computers and Chemical Engineering*, 27(3):293–311, 2003a.
- V. Venkatasubramanian, R. Rengaswamy, S. Kavuri, and K. Yin. A review of process fault detection and diagnosis Part II: Qualitative model and search strategies. *Computers and Chemical Engineering*, 27(3):313–326, 2003b.
- V. Venkatasubramanian, R. Rengaswamy, S. Kavuri, and K. Yin. A review of process fault detection and diagnosis Part III: Process history based methods. *Computers and Chemical Engineering*, 27(3):327–346, 2003c.
- S. Verron, J. Li, and T. Tiplica. Fault detection and isolation of faults in a multivariate process with bayesian network. *Journal of Process Control*, 20(8):902–911, 2010.
- W. Wang and J. Vrbanek. An evolving fuzzy predictor for industrial applications. *IEEE Transactions on Fuzzy Systems*, 16(6):1439–1449, 2008.
- M. J. Watts and N. Kasabov. Simple evolving connectionist systems and experiments on isolated phoneme recognition. In *Proceedings of the first IEEE conference on evolutionary computation and neural networks*, page 232–239, 2000.
- A. Widodo and B. S. Yang. Support vector machine in machine condition monitoring and fault diagnosis. *Mechanical Systems and Signal Processing*, 21(6):2560–2574, 2007.
- R. Yager. Learning Methods for Intelligent Evolving Systems. In *Proceedings of the International Symposium on Evolving Fuzzy Systems EFS’06*, IEEE Press, pages 3–7, 2006.
- T. Yamakawa, E. Uchino, T. Miki, and H. Kusabagi. A neo fuzzy neuron and its applications to system identification and predictions to system behavior. In *Proceedings of the 2nd International Conference on Fuzzy Logic and Neural Networks (IIZKA’92)*, page 477–484, 1992.
- E. Zio and G. Gola. A neuro-fuzzy technique for fault diagnosis and its application to rotating machinery. *Reliability Engineering & System Safety*, 94:78–88, 2009.

A. Zolghadri. An algorithm for real-time failure detection in Kalman filters.
IEEE Transactions on Automatic Control, 41(10):1537–1539, 1996.